Journal of Big Data

RESEARCH Open Access

Assessing the effects of hyperparameters on knowledge graph embedding quality

Oliver Lloyd[1*], Yi Liu[1] and Tom R. Gaunt[1]

*Correspondence:
oliver.lloyd@bristol.ac.uk

[1] MRC Integrative Epidemiology Unit, Bristol Medical School, University of Bristol, Bristol, UK

Abstract

Embedding knowledge graphs into low-dimensional spaces is a popular method for applying approaches, such as link prediction or node classification, to these databases. This embedding process is very costly in terms of both computational time and space. Part of the reason for this is the optimisation of hyperparameters, which involves repeatedly sampling, by random, guided, or brute-force selection, from a large hyperparameter space and testing the resulting embeddings for their quality. However, not all hyperparameters in this search space will be equally important. In fact, with prior knowledge of the relative importance of the hyperparameters, some could be eliminated from the search altogether without significantly impacting the overall quality of the outputted embeddings. To this end, we ran a Sobol sensitivity analysis to evaluate the effects of tuning different hyperparameters on the variance of embedding quality. This was achieved by performing thousands of embedding trials, each time measuring the quality of embeddings produced by different hyperparameter configurations. We regressed the embedding quality on those hyperparameter configurations, using this model to generate Sobol sensitivity indices for each of the hyperparameters. By evaluating the correlation between Sobol indices, we find substantial variability in the hyperparameter sensitivities between knowledge graphs with differing dataset characteristics as the probable cause of these inconsistencies. As an additional contribution of this work we identify several relations in the UMLS knowledge graph that may cause data leakage via inverse relations, and derive and present UMLS-43, a leakage-robust variant of that graph.

Keywords: Knowledge graph, Embedding, Sensitivity analysis, Hyperparameter tuning

Introduction

Substantial amounts of time and energy can be spent during the tuning of knowledge graph embeddings (KGEs) due to the computationally expensive nature of this process. This expense comes not just from algorithm complexity and dataset size, but also from the sheer breadth of the hyperparameter space. For example, optimising embeddings for a medium sized graph such as FB15k-237 [1] can take upwards of 100 hours even with multiple graphical processing units (GPUs) [2]. The specific GPUs used in that study were 4 × NVIDIA Tesla P100-SXM. At their maximum power draw of 300 watts, total electricity consumed for that single embedding process would have been more than 30 kilowatt-hours, which is more than a median UK household will use

© The Author(s) 2023. **Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.

in 100 days [3]. This figure is alarmingly high, particularly when considering that the study assessed 17 methods, each analysed over 5 datasets. With the rising popularity of graph databases [4] we expect a corresponding increase in the application of embeddings, highlighting the importance of reducing the cost of the process.

A number of approaches have been proposed to improve the efficiency of embedding KGs. Speed is one such facet for this, and algorithms such as FastRP [5] and RandNE [6] are capable of producing embeddings thousands of times faster than baseline methods such as DeepWalk or Node2vec. Achieved via random projection techniques, these approaches produce equal or better quality embeddings than those baselines at a fraction of the computational cost. The process of negative sampling, in which negative examples of edges are generated, has also come under scrutiny for its inefficiencies. A non-sampling method called NS-KGE has been developed that utilises *all* non-existent edges as examples, thereby eliminating the model instability associated with negative sampling [7]. This approach does come with increased cost, so mathematical derivations are leveraged to reduce the complexity of the loss function calculation down to acceptable levels. Some researchers have done away with negative sampling altogether, opting to only use positive examples in the training process for their PROCRUSTES algorithm [8]. Another efficiency improvement proposed in that same work is the aligning of tuples within a batch to be of the same relation type—this facilitates parallelisation as well as simplifying computation. Parallelisation itself does not reduce computational load, but does enable analyses that would be otherwise infeasible in serial processing. Several KGE libraries offer this functionality, including LibKGE [9] and DGL-KE [10]. Other research has focused on the storage of embeddings. LightKG [11] stores codebooks instead of continuous vectors, which results in a sevenfold decrease in required disk space. The authors further report that codebook storage can speed up entity lookup by more than 15-fold. This is a large efficiency improvement that becomes even more relevant when the embeddings are subject to a lot of downstream analysis (as they often are).

While these advances are important, there is room for further efficiency improvement by examining the scope of the hyperparameter optimisation process. Though shown to be important in the embedding task [12], it can be very inefficient (as noted above). This is partly because it is often carried out by random or pseudo-random grid-searches, methods that may involve a lot of redundancy if there are sets of task-irrelevant hyperparameters repeatedly being tested at different values despite having little to no effect on the quality of the outputted embeddings. With some prior knowledge these unimportant variables could remain fixed, allowing a search space with reduced complexity which places higher emphasis on the tuning of task-relevant hyperparameters. As a result, the whole tuning process should be more time- and energy-efficient.

To gain understanding into the relative importance of hyperparameters, we ran a global sensitivity analysis to ascertain the effect of changes to their values on the quality of the produced embeddings. We collected the data to support this analysis by implementing tens of thousands of embedding trials using a state-of-the-art embedding library. By repeating this process for three well-known benchmark KGs, we have enabled comparison of hyperparameter sensitivities across datasets.

Methods

Datasets

The datasets included in this study are FB15k-237, UMLS, and WN18RR—three popular datasets used in KG completion research. A summary of their characteristics is displayed in Table 1, and brief descriptions of their origins are found in the following paragraphs.

Freebase is a knowledge graph that was initially created in 2007 by Metaweb. Google acquired the company in 2010 and shut it down a few years later, transferring its data to Wikidata in the process. The FB15k dataset [13] is a subset of 592,213 edges from Freebase, comprising 14,951 entities (hence the name) and 1345 meta-relations that make up a selection of general facts such as the following triple: ('Jackie Chan', 'Acted in', 'Around the world in 80 days'). In a 2015 paper Toutanova and Chen highlighted the issue of inverse relations in the dataset, showing that a simple ruled-based approach could achieve state-of-the-art performance levels because many test set relations are just the inverse of those in the training set. For example, the aforementioned triple would be problematic if the following triple also existed: ('Around the world in 80 days', 'starred', 'Jackie Chan'). By removing 411 entities and including just 237 of the original relations, they derived a leakage-robust variant of the dataset, FB15k-237, which is now used frequently in KG completion research.

WordNet [14] is a large network representing the English language, with words connected if there is a conceptual link between them. For example, the 'hypernym' relation could exist from the node 'chair' to 'furniture', or the meronymic relation 'has_part' could link 'cat' and 'tail'. WN18, also introduced by Bordes et al., is a subset of WordNet that consists of 18 meta-relations and 40,943 entities. Much like FB15k, WN18 was commonly used in KG completion research until it was reported that this dataset also suffered from inverse relation test leakage [15]. To replace it the authors proposed the robust subset WN18RR, which removed 7 problematic meta-relations from the graph while retaining the same entities.

Table 1 Graph characteristics for the analysed datasets. Here, 'node degree' refers to the count of all adjacent edges to a given node (i.e. in-degree + out-degree)

Statistic	FB15k-237	UMLS	WN18RR
node count	14,541	135	40,943
edge count	310,116	6529	93,003
edge type count	237	46	11
density	6.19e−06	7.85e−03	5.04e−06
component count	6	1	13
mean component diameter	2.5	2	2.46
mean component distance	1.40	1.61	1.55
mean component connectivity	1	4	1.15
mean node degree	42.65	96.73	4.54
median node degree	26	71	3
maximum node degree	8642	382	521
standard deviation of node degree	127.70	87.44	8.58
skewness of node degree	34.07	1.84	26.15
kurtosis of node degree	1777.59	2.91	1095.90

A 'component' refers to a weakly connected subgraph that is disconnected from the rest of the graph

The Unified Medical Language System (UMLS) [16] is a set of resources that aims to consolidate many differing biomedical lexicons to standardise and facilitate interoperability between them. In graphical format, it exists as a set of 135 high-level biomedical entities (e.g., 'vitamin', 'alga', 'lipid') connected by 46 cognate relations (e.g., 'causes', 'disrupts', 'contains'). Although used in some KG completion research (e.g. [17, 18]), UMLS is less prevalent in the field than either FB15k-237 or WN18RR. Its inclusion in this work was motivated by its nature as a smaller and much denser graph than the other two, and because it represents a more specific knowledge domain. To our knowledge UMLS has not been checked for inverse relation leakage, so here we have performed this analysis and shared a new leakage-robust variant of the graph.

KGE methods

For this experiment we utilised all methods available in the knowledge graph embedding library LibKGE [9]. Spanning almost a decade and a variety of architectures, these methods represent some of the most popular and impactful KGE techniques developed to date. The inclusion of earlier methods such as RESCAL is justified by the LibKGE developers because older architectures can achieve comparable performance if they are allowed usage of the updated training methods and functions that are often presented alongside newer KGE methods [19]. An overview of these methods is presented in Table 2. Please note that throughout this paper, the term 'model' will be used only to refer to an *instantiated* version of a particular KGE method/algorithm.

Data collection

Hyperparameter ranges were taken from those used in the grid-searches by Ruffinelli et al. [19], but applied to the rest of the methods implemented in LibKGE. These ranges are presented in Additional file 1: Table S1, alongside brief descriptions of the hyperparameters' effects. Following the example of the LibKGE developers, the analysis was divided into individual jobs, each representing a different combination of dataset, KGE method, training method, and loss function. For example, the TransE algorithm applied to the UMLS dataset, training under the 1vsAll approach and using

Table 2 KGE methods included in the study, alongside their citation and mechanism category (as described by [2])

Method	Mechanism	Citation
ComplEx	Matrix factorisation	Trouillon et al., 2016
ConvE	Deep learning	Dettmers et al., 2018
CP	Matrix factorisation	Lacroix, Usunier, and Obozinski, 2018
Distmult	Matrix factorisation	Yang et al., 2014
Relational Tucker3	Matrix factorisation	Wang, Broscheit, and Gemulla, 2019
Rescal	Matrix factorisation	Nickel, Tresp, and Kriegel, 2011
RotatE	Geometric	Sun et al., 2018
SimplE	Matrix factorisation	Kazemi and Poole, 2018
TransE	Geometric	Bordes et al., 2013
Transformer*	Deep learning	Chen et al., 2020
TransH	Geometric	Wang et al., 2014

* Transformer is based on the 'no context' HittER method [20]

binary cross-entropy (BCE) loss, would be one such job. With these four elements fixed, we then altered the other hyperparameters in individual embedding trials. A full list of the 233 jobs is provided as Additional file 1: Table S2.

100 trials were run for each job, except for FB15k-237 jobs, on which we ran only 50 due to the dataset's size and the resulting computational cost. Hyperparameter values at each embedding attempt were chosen by Sobol sequence [21] to ensure a good spread of trial points throughout the search space. Datasets were split into training, validation, and testing sets—the former two splits were used for the learning process and the latter was used to assess embedding quality via link prediction (LP). A modified version of mean reciprocal rank (MRR) was used to measure LP performance, whereby existing triples are filtered from the ranked scores in order to prevent a model being unfairly marked down. Hits@k was a viable alternative metric, but MRR's continuous nature meant that the measurement would be more robust to the presence of false-negative edges in the entity rankings. For example, if a model scores a false-negative edge highly, this is a good indication that the embeddings are truly representative of the underlying graph. However, if that same edge pushes the target edge out of the top k rankings then the model will be marked down for this under hits@k evaluation, which would be incorrect. MRR does not entirely solve this issue, but it does lessen the impact.

Sensitivity analysis

For each KG, we selected the top 5% of trials by embedding quality to eliminate those that may have been poor quality due to stochastic processes. These performant trials' MRR scores were then regressed on their corresponding hyperparameter configurations using linear regression from scikit-learn [22]. Then, using SALib [23], we performed a Sobol sensitivity analysis [21] on each of the regression models and recorded the output indices.

Sobol indices represent the proportion of the decomposed output variance that is attributable to a particular input, or set of inputs, of a regression model. First-order indices correspond to the variance caused by each input variable alone, second-order indices correspond to variance attributed to interactions between *pairs* of input variables, and so on. Total-order indices are simply the sum of first- and all higher-order indices for each model input. As an example, if we have a general model $Y=f(X)$ with n inputs and total output variance V, first-order sensitivities, S_1, can be written as follows:

$$S_1 = \{\alpha_1, \alpha_2, .., \alpha_n\} \tag{1}$$

where α_i is:

$$\alpha_i = E(\Delta Var(Y)|\Delta x_i)/V \tag{2}$$

with x_i being the i-th input variable of X, such that:

$$\sum_{i=1}^{n} \alpha_i <= 1 \tag{3}$$

We can write second-order sensitivities, S_2, as:

$$S_2 = \{\beta_{ij} \, for \, i, \, j \, in \, X, \; i \neq j\} \tag{4}$$

where β_{ij} is:

$$\beta_{ij} = E(\Delta Var(Y)|\Delta x_i, \Delta x_j)/V \tag{5}$$

Third- and higher-order indices follow the same pattern, but are not calculated by SALib so we will not define them here nor use them in the following equations. With that in mind, total-order sensitivities, S_T, can be written similarly to the first-order set:

$$S_T = \{\omega_1, \omega_2, .., \omega_n\} \tag{6}$$

but in the total-order case, ω_i is calculated as:

$$\omega_i = \alpha_i + \sum_{j=1}^{n} \beta_{ij}, \; i \neq j \tag{7}$$

Applied in this case, we can view the Sobol indices as indicators of the relative importance of different KGE method hyperparameters, i.e. those with higher indices should be prioritised in the hyperparameter tuning process.

SALib uses the Monte Carlo approach for index estimation, so to this end a total of 86,016 samples were taken from each of the OLS models' input spaces using Saltelli's extension of the Sobol sequence [24]. This figure is slightly lower than the 100,000 recommended for the 20 input parameters that our regression models have [25], but due to the inclusion of several dummy variables in our inputs we considered this acceptable. The dummies themselves were set to 'one-hot' in each sample by setting the dummy with the highest sampled continuous value to 1, and its counterparts to 0.

Once sensitivity results were calculated, we estimated pairwise Pearson's correlation across the three datasets' indices to quantify the level of agreement between them. As second-order indices form a matrix, these were flattened to 1 dimension (in row major order) prior to this analysis.

Experimental conditions

This work was carried out using the computational facilities of the Advanced Computing Research Centre, University of Bristol—http://www.bristol.ac.uk/acrc/. The specific environment was CentOS-7 running Python 3.8.12 with PyTorch 1.7.1, accelerated with CUDA 11.4 on 4 × NVIDIA GeForce RTX 2080 Ti. The software version for scikit-learn was 0.24.2 and for SALib was 1.4.5.

All code and data used in the analysis are available in our GitHub repository [26].

Results

Embedding quality

We first report, in Fig. 1, on our proxy for embedding quality: LP results. Immediately noticeable is the MRR for the UMLS dataset, whose median value is consistently more than twice that of the other two KGs, regardless of method. Due to the nature of our measure for embedding quality, however, we cannot make direct comparisons *between*

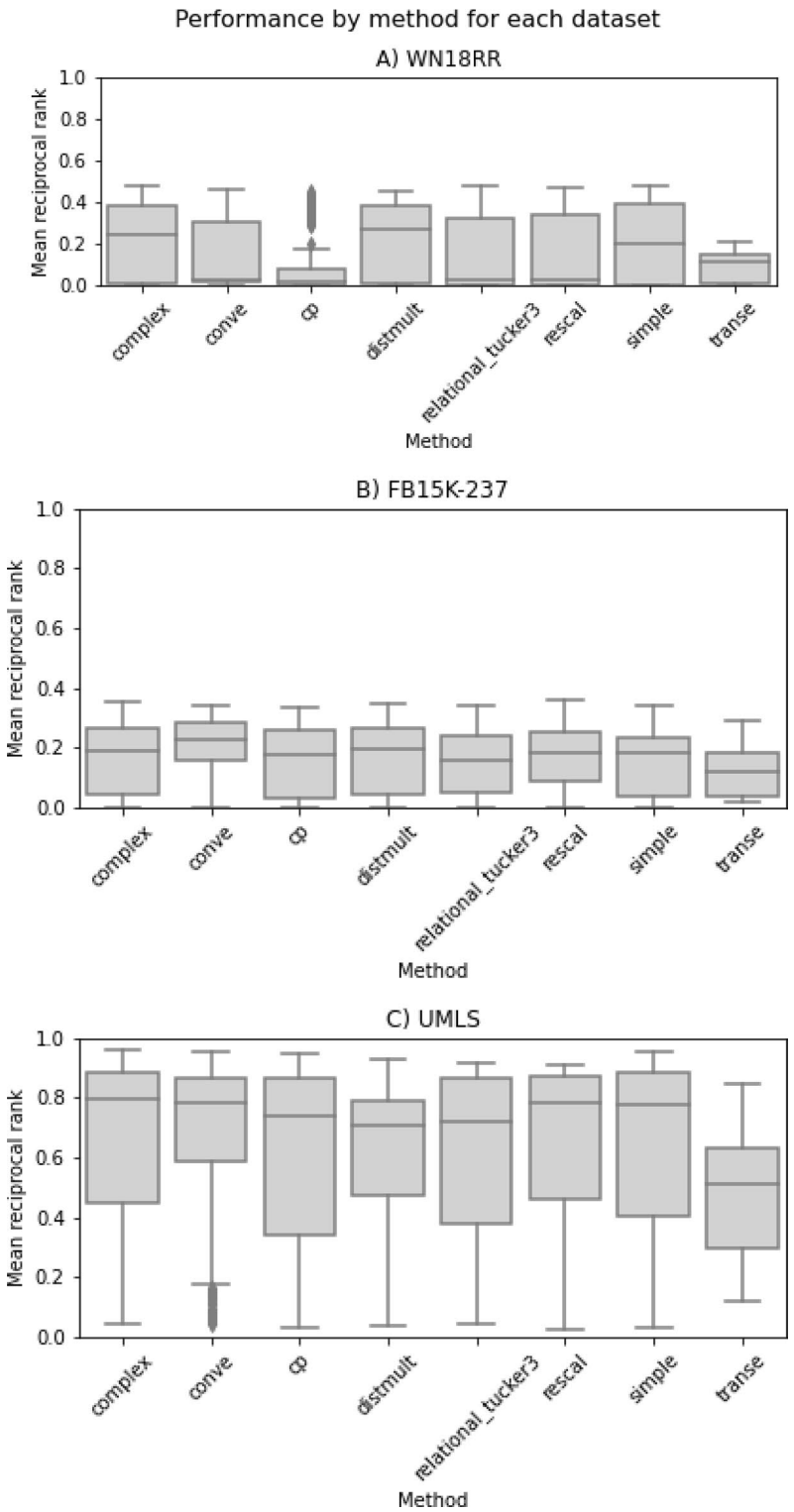

Fig. 1 Box plot showing link prediction performance by method and dataset. Outliers are displayed as black diamonds

KGs for the embedding quality, only *within* KGs. For example, the lower-end results of the CP method on WN18RR can be said to be the poorest quality for that dataset, but we cannot definitively say that those embeddings are better/worse than any of those from FB15k-237 or UMLS. The explanation for the MRR difference is therefore that the UMLS graph just presents an easier LP problem. Because WN18RR and FB15k-237 have both been constructed to specifically remove inverse relations, our first suspicion was that this performance-boosting problem is present in UMLS. Analysis revealed three relations that should be considered problematic by Dettmers' definition [15]: 'degree_of', 'precedes', and 'derivative_of'. However, these relations represented fewer than 2% of the held-out edges and as such cannot be the sole reason for the KGE methods' improved MRR on UMLS. All edges in UMLS that have a corresponding non-zero inverse edge, including those that did not cross the threshold to be considered problematic, are presented in Additional file 1: Table S4. Dataset statistics (see Table 1) may offer an additional explanation for the performance disparity. The UMLS graph is substantially denser than the other two, at 0.008 compared to 5e−6 and 6.2e−6 for WN18RR and FB15k-237 respectively. This should, on average, allow the models more contextual information with which they can make predictions. Furthermore, median degree is much higher and skewness of degree is much lower in UMLS than the other datasets, meaning there will be far fewer situations where the model is presented with a low-context problem and that would produce a lower confidence prediction.

The top results shown here do not differ much from the majority of those reported for the same three datasets [27–29], giving us confidence in the quality of our embeddings for the downstream analysis. Several of the employed KGE methods, however, were unable to be successfully run. From the selection of 233 total jobs, 36 were deemed invalid by LibKGE prior to running—15 of these came from TransE, which the software will only run if the training method is negative sampling and the loss function is one of BCE or margin ranking. All 21 Transformer/HittER jobs were invalid because the method does not support the'_po' scoring method which was used in all other jobs. On the other hand, some jobs did not complete regardless of their validity. Both RotatE and TransH failed on all jobs for WN18RR and FB15k-237 with full computational power allocated, likely due to the size of the datasets and/or inefficiencies in the specific implementations used. TransH also failed on UMLS when using 1vsAll and KvsAll training with BCE loss. As a result of these failures, these 2 KGE methods were excluded from the UMLS sensitivity analysis in order to make the resulting indices comparable with those of the other two KGs. Overall, of the 197 valid jobs, 167 ran to completion; this corresponds to a total of 13,450 embedding trials.

Hyperparameter sensitivities

For each of our per-dataset sensitivity results we report: first- and total-order Sobol sensitivity indices as bar charts (Fig. 2), and second-order sensitivities as network diagrams (Fig. 3).

For the UMLS embeddings we see in Fig. 2 that loss function dominates the relative importance of hyperparameters, both in terms of first- and total-order sensitivity. This is somewhat mirrored in WN18RR, where we see loss function have the highest total-order, despite being beaten by a slim margin on first-order sensitivity by weight

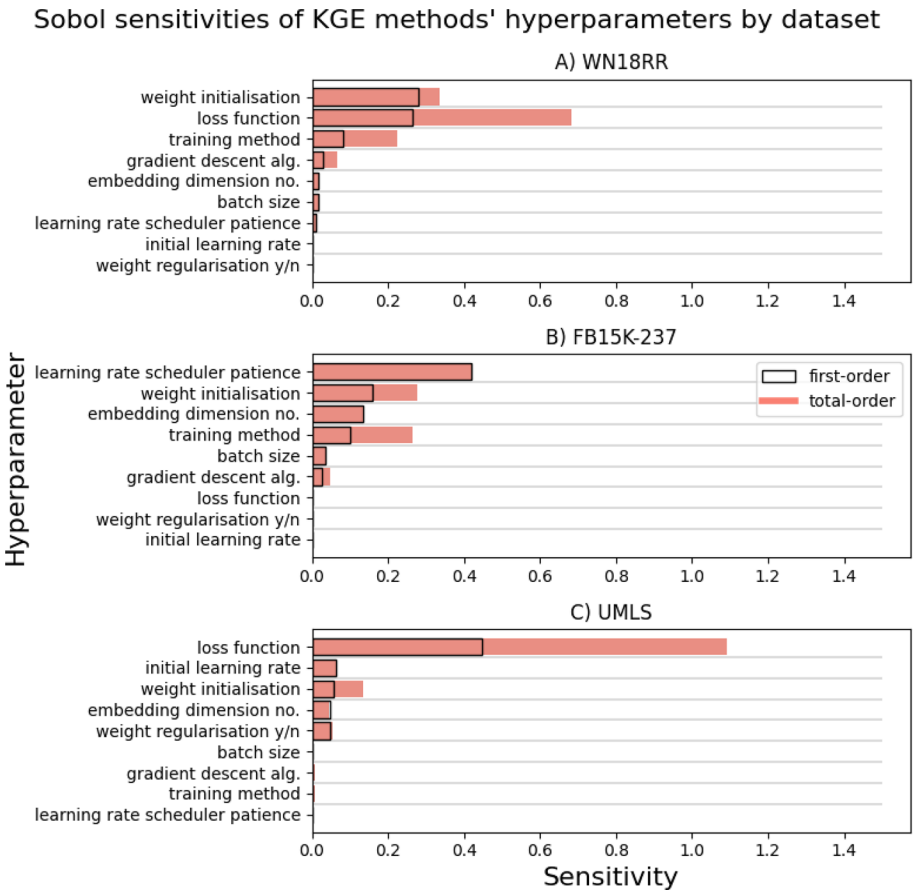

Fig. 2 Bar charts showing Sobol sensitivities of the hyperparameters of knowledge graph embedding methods on three benchmark datasets. Note that dummy variables and continuous subparameters have been cumulatively grouped into their corresponding categories: training method, loss function, weight initialisation, and gradient descent algorithm. The raw data used to generate the figure are provided in Additional file 1: Table S3

initialisation methods. FB15k-237, on the other hand, seems to show no sensitivity at all to the choice of loss function. In fact, its most sensitive hyperparameter is the patience of the learning rate scheduler (ReduceLROnPlateau), a value that is almost completely unimportant when embedding the other two KGs. Another area of inconsistency is the response to the size of the embedding space. This particular hyperparameter's sensitivity is relatively high for FB15k-237, perhaps attributable to the size and complexity of that graph, but again is negligible for the other two. Surprisingly, training method achieves just middling ranks on WN18RR and FB15k-237, and registers almost no sensitivity at all on UMLS apart from some very minor higher-order sensitivity. Also unexpectedly, the initial learning rate value actually scores above 0 sensitivity in one case. This occurs on UMLS where it finds itself ranked 2nd by first-order sensitivity on this dataset, but is still a long way behind loss function in 1st place and barely ahead of the three hyperparameters below it.

A few consistencies are arguably observed in Fig. 2, mainly for weight initialisation which ranks in the top 3 hyperparameters across all three datasets. However, there is still quite a large reduction in its sensitivity on UMLS, where, save for its total-order

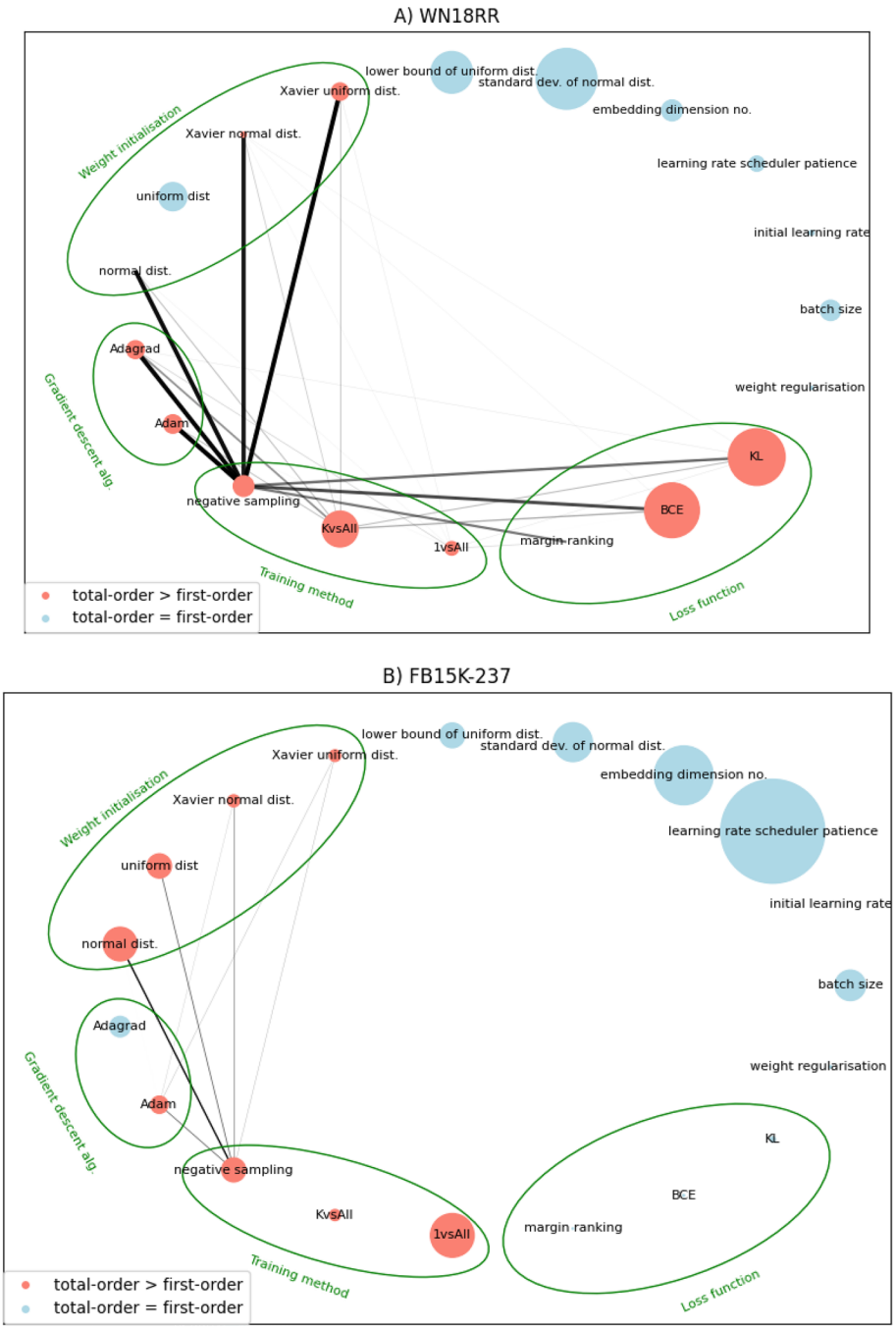

Fig. 3 Network diagrams for second-order Sobol sensitivities between hyperparameters of KGE methods on three benchmark datasets. Width and opacity of edges represent the magnitude of second-order effects. Node size represents first order sensitivity. Dummy variables are not grouped in this figure, but edges between them are disallowed as they are meaningless. Dummies' categories are indicated by green bands around nodes. Raw adjacency matrices for these networks are provided in Additional file 1: Tables S5.1, S5.2, and S5.3 respectively

value, it does not stand out from the other non-zero hyperparameters. Choice of gradient descent algorithm, batch size, and whether to regularise weights, could all also be considered to be consistent in their sensitivities, in that they are always less than 0.1 for

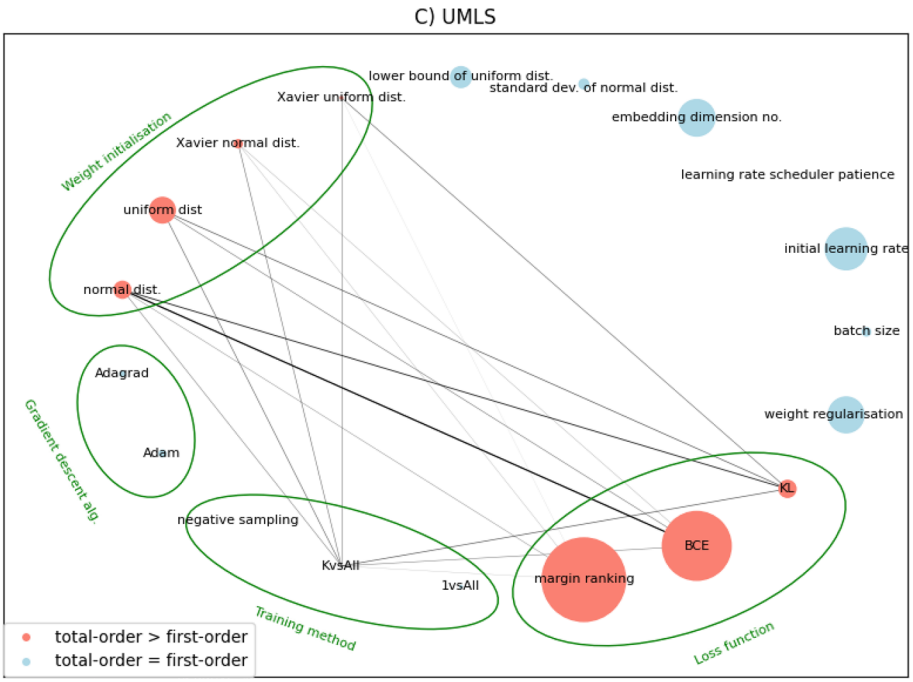

Fig. 3 continued

both first- and total-order. However, they still differ across the datasets in whether they are actually zero or not.

Overall, it is apparent that there is substantial variation in the relative hyperparameter sensitivities across the KG datasets. In fact, pairwise Pearson's correlation analysis on the ungrouped Sobol indices reveals that the highest correlation between first-order sensitivities of any two datasets is only 0.047 (WN18RR - UMLS). The other two pairs cross over into negative correlation, with WN18RR - FB15k-237 achieving a score of -0.13, and UMLS - FB15k-237 reaching the shallow depth of -0.19.

In Fig. 3 we see the strongest second-order sensitivities when embedding the WN18RR dataset, particularly using the 'negative sampling' training method. This hyperparameter value interacts heavily with others from all other dummy categories, making it the most central node in this network by weighted-degree. Its counterparts, KvsAll and 1vsAll, are also well connected, though the magnitude of their sensitivities is lower. For UMLS, training methods are still well connected but we see slightly heavier edges between weight initialisation and loss functions. This dataset also sees minor interactions involving gradient descent algorithm dummies, something that is mirrored by WN18RR but not by FB15k-237. Interestingly, across all three datasets we see that continuous hyperparameters (upper right nodes, clockwise from 'lower bound of uniform dist.' round to 'batch size') have negligible second-order sensitivities, meaning that such interactions are only occurring between the one-hot encoded categorical hyperparameters.

At the second-order level then, there superficially appears to be more consistency in hyperparameter sensitivity between KGs. Since our networks in Fig. 3 are undirected and all contain the same nodes, we can perform pairwise Pearson's correlation of them by flattening the upper triangles of their adjacency matrices to get 1-D vectors. Doing so

reveals that the correlation levels do not, in fact, improve upon those of the first-order indices. WN18RR - UMLS scores 0.078, WN18RR - FB15k-237 scores 0.063, and UMLS - FB15k-237 once more dips into negative correlation at -0.08.

Discussion

Hyperparameter tuning with the goal of generating high quality/representative KG embeddings is an expensive task, in part due to the size of the hyperparameter search space. With prior understanding of the relative contributions of different hyperparameters to the variance of that quality, we could trim down the search space so that we are not spending time and energy altering inconsequential variables. Here though, we have found that Sobol sensitivity indices of KGE algorithm hyperparameters differ substantially when embedding different datasets, and this difference is observed for higher-order indices as well as the first-order. Some of the intra-dataset sensitivity results presented here might prove useful in future embedding tasks on these specific datasets, for instance the lack of importance of loss function for FB15k-237 (Fig. 2B). However, these inferences are not generalisable to external datasets.

Varying graph characteristics seem the most likely cause of the reported differences in hyperparameter sensitivities across different KGs. For example, the '1vsAll' training strategy may not work well when embedding a graph with high density (e.g. >0.5), due to the fact that the method does not exclude existing edges from the negative examples it generates [30]. Thanks to the density, the embedding model might then frequently be presented with the same edges as both positive and negative examples, rendering the examples useless and slowing down the learning process. If this was the case, the method would therefore be more sensitive to the choice of training method (i.e. not 1vsAll) than it would be if the graph itself was less dense. As a more concrete example, we see here that our biggest graph, FB15k-237, is also by far the most sensitive to changes to the dimensional size of the embeddings. Given the diverse statistics of the graphs analysed here (see Table 1), it is perhaps unsurprising that we see a lack of consistency in their hyperparameter sensitivities.

Of course, other effects could be influencing results in parallel, if only to a lesser degree. For instance, the differing domains of the datasets: UMLS sits in the biomedical category, WN18RR represents language, and FB15k-237 represents 'general facts'. Real-world networks may have different patterns of edge emergence depending on their domain [31]. Consequently, the ability to predict held-out edges well for each of our datasets might require very different embedding configurations, which, in turn, could cause KGE hyperparameters to be of varying importance. Our choice of metric (MRR) may offer an additional explanation for the sensitivity differences. The use of mean *reciprocal* rank has been discouraged by Fuhr [32], who stated that the use of the reciprocal changes the metric from an interval to an ordinal scale, rendering a mean calculation invalid. Others have rejected this claim, however, with Sakai [33] writing in the same forum to challenge Fuhr's commandment-style assertions and highlight that the ordinal-means issue is still being debated. Regardless of controversy, MRR is still used extensively in contemporary research to measure LP performance. Finally, in any model that initialises to a random state there is the possibility of stochastic processes impacting the outcome. Our exclusion of all but the top performing trials

should lessen the impact of these processes on our results, but we cannot claim that this completely alleviates the possible effects.

In conclusion, we have shown that the response of embedding quality to changes in hyperparameter values differs substantially between KGs. To properly assess the extent to which this is determined by specific dataset attributes, it will be necessary to follow up this work with a sensitivity analysis in which permutations of the KGs are created to provide a continuous spectrum of graph characteristics that can be fed, alongside hyperparameter values, into a unified regression model. We believe this may uncover important second-order interactions that would help to explain the results presented here. In this work we have also identified a set of relations in the UMLS graph that could cause data leakage through inverse relations. We therefore present the UMLS-43 graph, a derivation of UMLS that is robust to such leakage, which is available for download in the provided GitHub repository [26].

Abbreviations

KG	Knowledge graph
KGE	Knowledge graph embedding
GPU	Graphical processing unit
UMLS	Unified Medical Language System
MRR	Mean reciprocal rank
BCE	Binary cross-entropy

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s40537-023-00732-5.

Additional file 1: A .zip file containing all supplementary tables, in gzip compressed .csv format, that are referenced in this manuscript.Supplementary table 1 provides the possible values that each of the tested hyperparameters could take. Supplementary table 2 describes the 233 total jobs that were attempted to be run in LibKGE in terms of: dataset, KGE method, training method, and loss function. Supplementary table 3 provides the raw data used to generate Figure 2. Supplementary table 4 shows the results of an inverse-relation-leakage analysis that was run on the UMLS dataset. Supplementary tables 5.1, 5.2, and 5.3 are the raw symmetrical adjacency matrices used to visualise the network data in figures 3A, 3B, and 3C respectively."

Acknowledgements
The authors would like to thank Patrick Rubin-Delanchy for his helpful comments and insights.

Author contributions
OL performed all analyses documented in this paper, and wrote the paper. YL and TRG oversaw the project and reviewed and edited the manuscript. All authors were involved in the conception and design of the study. All authors read and approved the final manuscript.

Funding
This work was supported by the UK Medical Research Council (MRC) and carried out in the MRC Integrative Epidemiology Unit [MC_UU_00011/4].

Availability of data and materials
Datasets generated or analysed during the current study are available in the GitHub repository, https://github.com/oliver-lloyd/kge_param_sens.

Declarations

Ethics approval and consent to participate
Not Applicable.

Consent for publication
Not Applicable.

Competing interests
The authors declare that they have no competing interests.

Received: 9 August 2022 Accepted: 11 April 2023
Published online: 06 May 2023

References

1. Toutanova K, Chen D. Observed versus latent features for knowledge base and text inference. Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality. 2015
2. Rossi A, Barbosa D, Firmani D, Matinata A, Merialdo P. Knowledge graph embedding for link prediction. ACM Trans Knowl Discov Data. 2021;15(2):1–49.
3. Energy Follow Up Survey: Household Energy Consumption & Affordability. 2021. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1018725/efus-Household-Energy-Consumption-Affordability.pdf. Accessed 8 Aug 2022.
4. DB-Engines Ranking per database model category. Db-engines.com. 2022. https://db-engines.com/en/ranking_categories. Accessed 8 Aug 2022.
5. Chen H, Sultan S, Tian Y, Chen M, Skiena S. Fast and accurate network embeddings via very sparse random projection. Proceedings of the 28th ACM International Conference on Information and Knowledge Management. 2019.
6. Zhang Z, Cui P, Li H, Wang X, Zhu W. Billion-scale network embedding with iterative random projection. 2018 IEEE International Conference on Data Mining (ICDM). 2018.
7. Li Z, Ji J, Fu Z, Ge Y, Xu S, Chen C et al. Efficient Non-Sampling Knowledge Graph Embedding. Proceedings of the Web Conference 2021. 2021.
8. Peng X, Chen G, Lin C, Stevenson M. Highly efficient knowledge graph embedding learning with orthogonal procrustes analysis. Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. 2021.
9. Broscheit S, Ruffinelli D, Kochsiek A, Betz P, Gemulla R. LibKGE—a knowledge graph embedding library for reproducible research. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. 2020.
10. Zheng D, Song X, Ma C, Tan Z, Ye Z, Dong J et al. DGL-KE: Training knowledge graph embeddings at scale. Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. 2020.
11. Wang H, Wang Y, Lian D, Gao J. A lightweight knowledge graph embedding framework for efficient inference and storage. Proceedings of the 30th ACM International Conference on Information & Knowledge Management. 2021.
12. Kadlec R, Bajgar O, Kleindienst J. Knowledge Base Completion: Baselines Strike Back. Proceedings of the 2nd Workshop on Representation Learning for NLP. 2017.
13. Bordes A, Usunier N, Garcia-Durán A, Weston J, Yakhnenko O. Translating embeddings for modeling multi-relational data. Adv Neural Inf Process Syst. 2013; 26.
14. Miller G. WordNet. Commun ACM. 1995;38(11):39–41.
15. Dettmers T, Minervini P, Stenetorp P, Riedel S. Convolutional 2D knowledge graph embeddings. Proceedings of the AAAI Conference on Artificial Intelligence. 2018;32(1).
16. Bodenreider O. The Unified Medical Language System (UMLS): integrating biomedical terminology. Nucleic Acids Res. 2004;32(90001):267D – 270.
17. Sang S, Yang Z, Liu X, Wang L, Lin H, Wang J, et al. GrEDeL: a knowledge graph embedding based method for drug discovery from biomedical literatures. IEEE Access. 2019;7:8404–15.
18. Trouillon T, Welbl J, Riedel S, Gaussier U, Bouchard G. Complex embeddings for simple link prediction. 2016. https://github.com/ttrouill/complex
19. Ruffinelli D, Broscheit S, Gemulla R. YOU can teach an old dog new tricks! ON training knowledge graph embeddings. 2019. https://github.com/uma-pi1/kge
20. Chen S, Liu X, Gao J, Jiao J, Zhang R, Ji Y. Hitter: Hierarchical transformers for knowledge graph embeddings. arXiv preprint arXiv:200812813. 2020;
21. Sobol' I. On the distribution of points in a cube and the approximate evaluation of integrals. USSR Comput Math Math Phys. 1967;7(4):86–112.
22. Pedregosa F, Michel V, Grisel O, Blondel M, Prettenhofer P, Weiss R, et al. Scikit-learn: machine learning in python. J Mach Learn Res. 2011;12:2825–30.
23. Herman J, Usher W. SALib: an open-source python library for sensitivity analysis. J Open Source Softw. 2017;2(9):97.
24. Saltelli A. Making best use of model evaluations to compute sensitivity indices. Comput Phys Commun. 2002;145(2):280–97.
25. Zhang X-Y, Trame MN, Lesko LJ, Schmidt S. Sobol sensitivity analysis: a tool to guide the development and evaluation of systems pharmacology models. CPT Pharmacometrics Syst Pharmacol. 2015;4:69–79.
26. Lloyd O. Zenodo; 2022. https://doi.org/10.5281/zenodo.6627236
27. Papers with code - fb15k-237 benchmark (link prediction). https://paperswithcode.com/sota/link-prediction-on-fb15k-237?metric=MRR
28. Papers with code - wn18rr benchmark (link prediction). https://paperswithcode.com/sota/link-prediction-on-wn18rr?metric=MRR
29. Papers with code - umls benchmark (link prediction). https://paperswithcode.com/sota/link-prediction-on-umls?metric=MR
30. Lacroix T, Usunier N, Obozinski G. Canonical tensor decomposition for knowledge base completion. 35th International Conference on Machine Learning, ICML 2018. 2018;7:4475–86. http://arxiv.org/abs/1806.07297
31. Yu J, Wu L-Y. Understanding the network formation pattern for better link prediction. ArXiv. 2021;abs/2110.08850.
32. Fuhr N. Some common mistakes in IR evaluation, and how they can be avoided. ACM SIGIR Forum. 2018;51(3):32–41.
33. Sakai T. On Fuhr's guideline for IR evaluation. ACM SIGIR Forum. 2020;54(1):1–8.

Publisher's Note
Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen journal and benefit from:

- ► Convenient online submission
- ► Rigorous peer review
- ► Open access: articles freely available online
- ► High visibility within the field
- ► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com