On the impact of Compression and Aggregation on Knowledge Graph Embeddings

Nader Sobhi[1,†], Unmesh Joshi[2,†], Jacopo Urbani[2] and Katja Hose[1]

[1]TU Wien, Favoritenstrasse 9, 1040, Vienna, Austria
[2]Vrije Universiteit Amsterdam, De Boelelaan 1105, 1081 HV Amsterdam, The Netherlands

Abstract

Knowledge Graph Embedding (KGE) models for Knowledge Graphs (KGs) can be used to predict missing links between nodes in a knowledge graph. Link prediction quality depends on the integrity of these vector representations, as even small modifications can lead to significant changes in predictions. The total size and the number of vector representations grow with the number of entities in a KG. As such, methods that can compress and approximate KGEs are beneficial. Additionally, methods that prune the search space by reducing the number of entities considered for a query are of great interest. In this paper, we study the possible extent of approximation of entity embeddings and its influence on link prediction across two methods: (1) a *vector quantization* method that compresses individual entity embeddings into compact integer representations and (2) a *triple-based partitioning* method that groups entities defined by shared neighbors to create representative aggregate embeddings. We evaluate both methods on a number of standard KGs and KGE models. We measure the change embeddings undergo using two similarity measures and analyze the relationship with link prediction performance. Our empirical results yield concrete thresholds, as measured by the Euclidean distance and signal-to-noise ratio, prescribing how much embeddings can be modified before downstream task accuracy degrades. Additionally, we show that these thresholds can be overcome when using grouping methods such as triple-based partitioning.

Keywords

Knowledge Graph, Knowledge Graph Embeddings, Link prediction, Vector quantization, Compression, Embedding aggregation, Euclidean distance, Signal-to-noise-ratio

CEUR-WS.org/Vol-4313/kgnesy-paper3.pdf

1. Introduction

Knowledge Graphs (KGs) are a popular data model for capturing factual knowledge. They are crucial assets for enhancing various tasks like question answering [1], ontology-based data access [2], task-oriented dialogs [3], data integration [4], named entity recognition [5], and more recently as a way to improve the context of LLMs using GraphRAG [6] with the largest public KGs containing billions of statements, e.g., Wikidata [7], DBpedia [8].

One way to leverage KGs for these tasks is to train a Knowledge Graph Embedding (KGE) model, which can then be used for the task of *link prediction*. A KGE model [9] embeds entities and relations into low-dimensional numerical vectors so that links are identified by computing likelihood scores using those vectors. Multiple studies have shown that KGE models return good results for link prediction [9, 10]. However, the usage of KGEs of large KGs is problematic: when the KG contains millions of entities, computing the likelihood score of all entities for each potential link is too expensive, and storing all embeddings during computation requires substantial memory.

Various methods have been proposed to address these problems by *approximating* the vector embeddings. In this paper, we consider *compression* of individual embeddings into more compact integer representations that reference a common codebook using **Vector quantization (VQ)** [11]. We also

Third Workshop on Knowledge Graphs and Neurosymbolic AI (KG-NeSy 2026), in ISWC 2026 Workshops Joint Proceedings, October 25–26, 2026, Bari, Italy

†These authors contributed equally.
nader.sobhi@tuwien.ac.at (N. Sobhi); u.n.joshi@vu.nl (U. Joshi); j.urbani@vu.nl (J. Urbani); katja.hose@tuwien.ac.at (K. Hose)
https://nader2929.github.io/ (N. Sobhi); https://unme.sh/ (U. Joshi); https://www.jacopourbani.it/ (J. Urbani); https://www.dbai.tuwien.ac.at/staff/khose/ (K. Hose)
0000-0002-0717-3559 (J. Urbani); 0000-0001-7025-8099 (K. Hose)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

CEUR Workshop Proceedings ceur-ws.org ISSN 1613-0073
published 2026-11-05

consider creating *aggregate* embeddings that represent groups of entities; these aggregate embeddings are then used during link prediction to narrow the search space. We dub this method **Triple-based partitioning** [12]. The disadvantage of using these methods to reduce memory and computation requirements is that any embedding approximation introduces a *deviation* from the original embedding, and any such deviation affects the quality of link prediction. Both **Vector quantization** and **Triple-based partitioning** differ in how they cause the deviation. With **Vector quantization**, the deviation is introduced when decompressing the integer representation of an embedding by concatenating the values referenced in the codebook. With **Triple-based partitioning**, the deviation is introduced by the usage of aggregate embeddings to reduce the number of entities that are scored as potential links for link prediction. This gives rise to the following research question: *How much can embeddings be compressed or aggregated before they lose their semantic usefulness for link prediction?*

To answer this question we measure the difference between original embeddings and approximated embeddings using Euclidean distance and Signal-to-Noise Ratio, and relate these difference measures with link prediction performance. In doing so, we compute measurements of how link prediction performance degrades as the compression we apply using VQ increases. We also compute these measures between the entity embeddings and the aggregate embeddings for each group of entities, and analogously also measure link prediction performance. We then use Rate-distortion theory (RDT) [13] from information theory to describe the precise relationship between the change in the embeddings, measured by Euclidean distance and Signal-to-Noise Ratio, and the distortion of the information in the embeddings, measured by link prediction performance. Thus, we can also define a formal relationship indicating the amount of information an embedding must contain before it becomes useless for link prediction. In this work, we train three embedding models on two datasets, whose embeddings we then use for our experiments with the VQ and Triple-based partitioning methods.

In summary, the key contributions of this paper are as follows:

1. Through experimental evaluation, we show how link prediction performance degrades as the embeddings deviate from their original representation.
2. We analyze the precise relationship between recall in link prediction and the amount of compression applied by VQ to KGEs.
3. We show that even though group aggregate embeddings have large deviations from their related entity embeddings, they can still be used for link prediction with near-perfect recall.
4. We show that across datasets and models, the relationship between compression and link prediction follows the same distribution. Thus, an intuitively understood phenomenon is given an exact model.

The remainder of this paper is organized as follows. After explaining basic concepts in Section 2, Section 3 explains the mechanisms behind the Vector Quantization and triple-based partitioning methods. Section 4 then presents the results of the conducted experiments, and Section 5 concludes and summarizes them.

2. Preliminaries

Knowledge Graphs (KGs) A KG is a directed labelled graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{R})$ where $\mathcal{V}$ is a set of nodes, $\mathcal{E}$ is a set of edges between nodes, and $\mathcal{R}$ is a set of named relations. Each entity is represented as a node $e \in \mathcal{V}$, and each relation as an edge $r \in \mathcal{R}$. We denote each fact in $\mathcal{E}$ as a triple $(e_h, r, e_t) \in \mathcal{E}$, where the outgoing node e_h is the head, and the incoming node e_t is the tail. An example from Figure 1 would be $(Cat, likes, Yak)$, where $e_h = Cat$, $r = likes$, and $e_t = Yak$.

Link prediction (LP) The goal of link prediction is to predict the head or the tail entity given the relation and the other entity. For instance, if r and e_t are given, then the goal is to predict the correct e_h, which can also be seen as finding a link that matches a pattern q of the form $(?, r, e_t)$. The list of all patterns of $\mathcal{G}$ is denoted by Q. In more formal notation, q is a pattern of the form $\langle ?, r, e_t \rangle$ where

$r \in \mathcal{R}, e_t \in \mathcal{V}$ and during LP, a prediction $\forall e_h \in \mathcal{V}$ is made as to whether a triple $\langle e_h, r, e_t \rangle$ exists that satisfies the pattern q (the case for $q = \langle e_h, r, ? \rangle$ is analogous). LP can also be done by ranking entities in $\mathcal{V}$ to produce a list $\rho = \langle e_1, \ldots, e_{|\mathcal{V}|} \rangle$ ordered by decreasing likelihood. The ranked position of e in the ranked list ρ is defined as $\text{pos}_\rho(e)$. The degree of success a link prediction approach achieves is then measured by the Hits@K metric. Hits@K is calculated as: the number of times that $\text{pos}_\rho(e_h) \leq$ K for a pattern $q = \langle e_h, r, ? \rangle$, divided by the total number of patterns considered $|Q|$. In this paper, we report Hits@K for K = 10, i.e., Hits@10. Additionally, we report the filtered Hits@10 variant. The filtered variant updates ρ by removing all answers contained in the training set. This variant is commonly reported in the literature, so that a method that predicts a different e_h or e_t that is also a valid answer to a pattern q is not penalized. Various approaches for performing LP have been studied, with the successful methods being: Rule Mining [14, 15], Path Ranking Algorithms [16], and machine learning models that produce KGEs.

Embeddings In general, embeddings are learned numerical vector representations of the items in a set of data. The learning is done by optimizing a loss function that ensures that the interactions between the individual items in the data are captured. For example, word embeddings are learned such that words that frequently occur together have similar embeddings. Where similarity is measured by a scoring function that gives the distance between two embeddings. KGEs are the learned vector representations of the entities and relations in a KG. Each embedding is a vector in $\mathbb{R}^{dim}$ with $dim > 0$. We use boldface fonts to denote them, i.e., we write $\mathbf{e}$ and $\mathbf{r}$ to refer to the vectors associated with entity e and relation r.

Knowledge Graph Embeddings The set of embeddings learned by a KG embedding model are intuitively called Knowledge Graph Embeddings (KGEs). Techniques like TransE [17], ComplEx [18], etc. define a *score function* $\psi(\mathbf{e_h}, \mathbf{r}, \mathbf{e_t})$ to compute the likelihood that triple (e_h, r, e_t) is true. During the training procedure, the embeddings are updated based on a loss function that maximizes the score for true triples and minimizes the score for false triples. Once the embeddings are trained, they can be used for link prediction. This is done by computing the score for every entity $e \in \mathcal{V}$ and using the score as a likelihood that e is an answer. Thus, by ranking all entities $e \in \mathcal{V}$ and ranking them by their score ψ, we can produce a list ρ.

3. Compression and Aggregation

In this work, we analyze two methods that use an approximation of entity embeddings during link prediction. Vector quantization compresses the embeddings, then, during link prediction, uses the decompressed representation of the embeddings. Triple-based entity partitioning creates groups of entities whose embeddings are used to create aggregate embeddings that are used during link prediction. Each method introduces a deviation from the original entity embedding $\mathbf{e}$ when performing link prediction, relying on an approximate embedding $\hat{\mathbf{e}}$ to retrieve relevant entities for ranking.

3.1. Vector Quantization

Following Sachan [11], each entity embedding $\mathbf{e}$ is compressed into a more compact representation, i.e., a smaller vector of integers $\mathbf{kd}$. A vector $\mathbf{kd}$ has size $D << |\mathbf{e}|$ and each integer in it is between $\{1 \ldots K\}$, where K and D are integer parameters that are set at the start of the compression procedure. Henceforth, we call such a vector a *KD code*.

After compression, each entity $\mathbf{e}$ is replaced by its KD code $\mathbf{kd}$. Each entity embedding $\mathbf{e}$ is compressed into a $\mathbf{kd}$ as follows; we partition each entity embedding $\mathbf{e} \in \mathbf{E}$ into D sub-vectors of equal size $z = \frac{|\mathbf{e}|}{D}$. Each discrete code for the i-th value of the $\mathbf{kd}$ code represents the partition of an entity embedding $\mathbf{e}_{i \cdot z}^{(i+1) \cdot z}$. The value of each $\mathbf{kd}_i$ is determined by the nearest neighbor assignment in the Equation 1.

$$\mathbf{kd}_i^D = \arg_{k=1}^{K} \min_k \; dist\Big(\mathbf{e}_{i\cdot z}^{(i+1)\cdot z}, C_i\Big) \tag{1}$$

Where $dist(\mathbf{a}, \mathbf{b}) = \|\mathbf{a}-\mathbf{b}\|_2^2$, i.e., the distance function is the squared Euclidean distance. The codebook C is defined as a two-dimensional vector, the first dimension is a vector of vectors, and has length of D, the second dimension of C contains K vectors. Each $\mathbf{kd}$ indexes the C in each of the two dimensions. The first dimension is indexed by the value of i, i.e., the positional index of each element in $\mathbf{kd}$; the second dimension, k, is indexed by the value of each $\mathbf{kd}_i$. The vectors that are retrieved when C is indexed in this manner are concatenated and return the decompressed embedding $\hat{\mathbf{e}}$ that represents an entity, as shown in Equation 2.

$$\hat{\mathbf{e}} = \Big[C_{1,\mathbf{kd}_1},\ \ldots,\ C_{D,\mathbf{kd}_D}\Big] \tag{2}$$

Codebook learning Both the codebook C and the KD codes are learned over multiple epochs. The KD codes are set at every epoch as the result of Equation 1. Whereas C is updated by a loss function which is defined as the error between the reconstructed embeddings $\hat{\mathbf{e}}$ and the original embeddings $\mathbf{e}$. The error function is the Mean Squared Error, shown in Equation 3.

$$\mathcal{L}_1 = \frac{1}{|\mathcal{V}|} \sum_{e \in \mathcal{V}} \|\mathbf{e} - \hat{\mathbf{e}}\|^2 \tag{3}$$

We use the Adam [19] optimizer to update C based on this loss at every epoch. We stop the codebook learning procedure when the loss gets below an ϵ or when we have reached the maximum defined number of epochs. At the end, we have a learned codebook C and a set of KD codes, $\mathbf{kd} \in \mathbf{KD}$ for each entity $e \in \mathcal{V}$, that is decompressed using the operation in Equation 2.

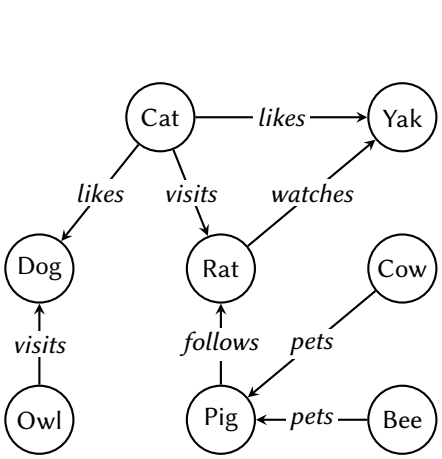
Storage efficiency Assuming the original embeddings used 32-bit floats, the total space required would be $32 \cdot |\mathbf{e}| \cdot |\mathcal{V}|$ bits. With discrete codes, we store $16 \cdot D \cdot |\mathcal{V}|$ bits for the KD codes plus $32 \cdot K \cdot z$ bits for the C, assuming we use 16-bit unsigned integers for $\mathbf{KD}$. Since typically D and K are much smaller than $32 \cdot |\mathbf{e}|$, and $K \ll |\mathcal{V}|$, the compression ratio can be substantial. For example, the FB15k237 dataset has 14,541 entities [17] and assuming that $|\mathbf{e}| = 256$ with $K = 128$ and $D = 32$, the compression ratio is $\approx 15.9\times$ based on the following calculation:

$$\text{Compression Ratio} = \frac{32 \cdot |\mathbf{e}| \cdot |\mathcal{V}|}{(16 \cdot D \cdot |\mathcal{V}|) + (32 \cdot K \cdot z)} \tag{4}$$

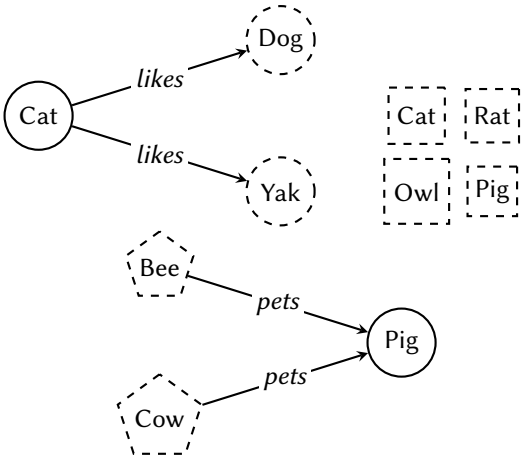
Link prediction using Vector quantization Link prediction for a pattern $q = \langle e_h, r, ?\rangle$ using VQ proceeds as normal by scoring all entities and then ranking them based on their likelihood scores, however the scoring is done using the decompressed embeddings rather than using original embeddings. I.e, instead of getting the score for a candidate triple as $\psi(\mathbf{e_h}, \mathbf{r}, \mathbf{e_t})$, we instead use $\psi(\hat{\mathbf{e}}_\mathbf{h}, \mathbf{r}, \hat{\mathbf{e}}_\mathbf{t})$.

3.2. Triple-Based Partitioning

Following Joshi [12], we now introduce the triple-based partitioning method that groups entities based on the triples $\langle e_h, r, e_t\rangle \in \mathcal{E}$ they belong to. Unlike the VQ method, the individual entity embeddings do not change. Instead, the focus is on creating groupings of entities such that the element-wise average of their embeddings serves as a representative centroid. We call these element-wise group average embeddings, aggregate embeddings.



(a) Example knowledge graph with 8 entities and 5 relation types.



(b) Example of found groups from Figure 1 (a) using triple-based partitioning, each shape denotes the relevant entities of a different group. Dashed lines show entities considered in aggregate embeddings.

Figure 1: An example of a simple Knowledge Graph (a) and the entity groups inside it identified using Triple-based partitioning (b).

Partitioning For each triple $\langle e_h, r, e_t \rangle \in \mathcal{E}$, we remove the tail to obtain the pattern $\langle e_h, r, ? \rangle$. All entities that appear as valid tails for this pattern form a triple-based group $s \in S$, where S is the set of all triple-based groups. In this work, we sometimes refer to these triple-based groups simply as groups. Symmetrically, for each pattern $\langle ?, r, e_t \rangle$, all entities that appear as valid heads form another group.

More formally, an entity group is identified for each triple $\langle e_h, r, e_t \rangle$ and consists of all entities $\{e_t \mid \langle e_h, r, e_t \rangle \in \mathcal{E}\}$, i.e., the set of nodes that are connected to $e_h \in \mathcal{E}$ with label $r \in \mathcal{R}$. Likewise, another group of entities that is rooted at e_t and r is created from the set $\{e_h \mid \langle e_h, r, e_t \rangle \in \mathcal{E}\}$.

We only keep groups larger than a minimum size τ. Entities that do not belong to any group are collected into an "orphan" group.

Example 1. *Consider the graph in Figure 1 with $\tau = 2$. Triple-based partitioning produces entity groups: {Dog, Yak} and {Cow, Bee} based on the patterns ⟨Cat, likes, ?⟩ and ⟨?, pets, Pig⟩. The remaining entities {Cat, Rat, Owl, Pig} that do not appear in any sufficiently large subset are placed in the orphan group. This example is illustrated in Figure 1.*

Aggregate Embeddings Our starting point is the embeddings of a pre-trained KGE model trained on a dataset. Given the entity embeddings from such a model, we compute the aggregate embedding for each group as the element-wise average of the entities' embeddings in that group. For a group $s \in S$, the corresponding embedding $\mathbf{s} \in \mathbf{S}$ is calculated as:

$$\mathbf{s} = \frac{1}{|s|} \sum_{e \in s} \mathbf{e} \tag{5}$$

In Figure 1(b), this is illustrated by showing the entities that are considered in their group's aggregate embeddings with dashed lines.

Link prediction using aggregate embeddings After computing the embeddings for all groups S, we can use them to perform link prediction using the following two-step approach. First, the aggregate embeddings $\mathbf{S}$ are scored, producing a list $\sigma = \langle s_1, \ldots, s_{|S|} \rangle$ that ranks entity groups. Second, the embeddings of entities that belong to the $\mathsf{Top-N}$ of σ groups are scored to produce a list of ranked entities $\rho' = \langle e_1, \ldots, e_n \rangle$, where n is the number of entities that belonged to the $\mathsf{Top-N}$ groups. The

advantage of triple-based partitioning is that there can be fewer embeddings ranked in total for each pattern q, thereby reducing computational requirements. This is because when τ is set to an optimal value, then $|S| + n \ll |\mathcal{V}|$. The optimal value for τ depends on the dataset.

4. Experiments

We empirically evaluate the deviation introduced by the two strategies presented in Sections 3.1 and 3.2.

4.1. Setup

Environment We implemented both the VQ and triple-based approaches as wrappers around the LibKGE [20] library in Python (ver 3.13.12). All experiments were conducted on a host running Ubuntu 24.2 LTE, equipped with an AMD Ryzen 9 5900X 12-core Hyperthreaded CPU, an Nvidia RTX 5080 GPU, 96 GB of RAM, and a 1 TB NVMe SSD.

Datasets We consider two standard knowledge graph benchmarks widely used in the link prediction literature [21, 22]: **FB15K-237** [23], a subset of Freebase, and **WN18RR** [24], a subset of WordNet. The number of entities, relations, and triples is shown in Table 1.

			Triples		
Dataset	**Entities**	**Relations**	**Training**	**Validation**	**Test**
FB15k237	14540	237	272115	17535	20466
WN18RR	40943	10	86835	3034	3134

Table 1
The number of entities, relations, and triples in both datasets that were considered in this paper.

Models We consider three models, TransE [17], ComplEx [18], and RotatE[25], which are often used as the benchmarks for new KGE model development [26, 27], and performing well for link prediction [21, 22].

4.2. Evaluation Metrics

Let $\mathbf{e} \in \mathbb{R}^d$ denote the original embedding of an entity and let $\hat{\mathbf{e}} \in \mathbb{R}^d$ denote a decompressed embedding using vector quantization, an aggregate embedding is denoted as $\mathbf{s}$. We compute the difference of $\hat{\mathbf{e}}$ and $\mathbf{s}$ from $\mathbf{e}$ for each $model, dataset$ pair as described below.
In computing the distance metrics in this manner for the VQ and triple-based partitioning methods, we measure the difference between original embeddings and the approximate embeddings that each method produces. We consider the following distance metrics:

Euclidean Distance The straight-line distance between two vectors in $\mathbb{R}^d$:

$$dist_{\text{Euclid}}(\mathbf{e}, \hat{\mathbf{e}}) = \sqrt{\sum_{j=1}^{dim} (\mathbf{e}_j - \hat{\mathbf{e}}_j)^2} = \|\mathbf{e} - \hat{\mathbf{e}}\|_2. \tag{6}$$

If the difference is zero, that indicates identical vectors; larger values indicate greater displacement. In Equation 6, we illustrate the formula using a decompressed embedding $\mathbf{e}$. However, the same formula can be applied in an analogous manner with the group embeddings $\mathbf{s} \in \mathbf{S}$.

	Model					
	TransE		ComplEx		RotatE	
Dataset	$\|\mathbf{e}\|$	Hits@10	$\|\mathbf{e}\|$	Hits@10	$\|\mathbf{e}\|$	Hits@10
FB15k237	128	0.497	256	0.536	256	0.522
WN18RR	512	0.517	128	0.547	256	0.552

Table 2
The Hits@10 performance and embedding dimensions ($|\mathbf{e}|$) of the considered models on the chosen datasets after the regular KGE training procedure.

Signal-to-Noise Ratio (SNR) Signal-to-Noise Ratio (SNR) measures the energy of the original embedding relative to the energy of the reconstruction error. A high value means the distortion introduced by approximation is small compared to the magnitude of the embedding itself. Note that it is complementary to the Euclidean distance. While Euclidean distance captures the absolute displacement of an embedding, the SNR expresses that displacement relative to the embedding's own scale, making it comparable across entities and models whose embeddings differ in magnitude. The formula for computing SNR is below:

$$\mathrm{SNR}(\mathbf{e}, \hat{\mathbf{e}}) = \frac{\sum_{j=1}^{d} \mathbf{e}_j^2}{\sum_{j=1}^{d} (\mathbf{e}_j - \hat{\mathbf{e}}_j)^2} = \frac{\|\mathbf{e}\|_2^2}{\|\mathbf{e} - \hat{\mathbf{e}}\|_2^2}. \tag{7}$$

Higher values indicate that the reconstruction error is small in relation to the original signal magnitude; an SNR of ∞ corresponds to a perfect reconstruction. Again we substitute $\hat{\mathbf{e}}$ with $\mathbf{s}$ to compute the same metric with regard to an entity embedding $\mathbf{e}$ and its group aggregate embedding.

Hits@10 Recall We train the three models on the two datasets and measure the Hits@10 performance of the original embeddings produced; the results are reported in Table 2. The results presented are consistent with those reported by other studies using these models and datasets [10, 9, 21].

Since VQ and triple-based partitioning modify the embeddings and the link prediction approach, we report their Hits@10 recall; the ratio of the achieved Hits@10 relative to the original performance reported in Table 2.

4.3. Experiment Results and Analysis

We evaluate both approaches, each addressing a distinct question about the relationship between embedding precision and link prediction quality. We also provide some interpretation and discussion of our results.

4.3.1. Compression Limits of Vector Quantization

We first ask how aggressively entity embeddings can be compressed via vector quantization before link prediction quality degrades beyond an acceptable threshold. We incrementally increase compression by increasing the value of K and D from VQ. Figure 2(a) plots the average Euclidean distance of $\hat{\mathbf{e}}$ and $\mathbf{e}$, $\forall \mathbf{e} \in \mathcal{V}$ and relates it to the Hits@10 recall. Likewise, Figure 2(b) relatest the Hits@10 recall to the average Signal-to-Noise Ratio between $\hat{\mathbf{e}}$ and $\mathbf{e}$, $\forall \mathbf{e} \in \mathcal{V}$.

Vector Quantization on FB15k-237 Regarding FB15k237, and based on Figure 2, all three models exhibit a smooth, monotonically decreasing relationship between Euclidean distance and recall: recall stays above 0.9 while the average displacement remains below a model-dependent threshold, ranging from roughly 3.6 for ComplEx to 5.2-5.6 for TransE and RotatE (Table. 3). Beyond this point, the models diverge sharply: ComplEx degrades fastest, with the Hits@10 recall dropping below 0.4 at a Euclidean distance of 6.5. Both TransE and RotatE retain Hits@10 recall above 0.5 even at distances of 8 to 9;

	FB15k237				WN18RR			
	VQ Parameters		Distances		VQ Parameters		Distances	
Model	K	D	Eucl. ($\leq$)	SNR ($\geq$)	K	D	Eucl. ($\leq$)	SNR ($\geq$)
TransE	128	32	4.53	9.03	64	64	5.77	2.96
RotatE	512	32	4.91	6.49	256	32	4.94	6.49
ComplEx	64	64	2.74	8.41	16	32	1.46	3.29

Table 3
Embedding-departure thresholds at which Hits@10 recall stays at or above 0.9, obtained by examining the fitted Weibull functions in Fig. 2. Recall stays high for Euclidean distance *below* the listed value and for SNR *above* it.

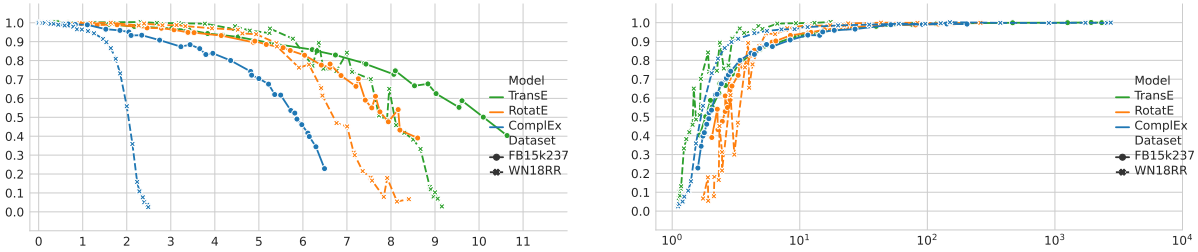
this is, of course, a large degradation when it comes to LP. The SNR view (log scale) corroborates this pattern: once the Signal-to-Noise Ratio exceeds approximately 10, all models on FB15k-237 converge to near-perfect recall, with recall between 0.9 and 1.0.

Vector Quantization on WN18RR For WN18RR, again based on Figure 2, the experiments show similar behavior, albeit with faster degradation of Hits@10 recall, i.e., the same monotonic decay holds, but degradation sets in at much smaller increments. Every model tolerates less deviation before recall drops, compared to FB15k237. ComplEx is extremely sensitive to deviation: recall collapses to near zero at a Euclidean distance of 3. TransE and RotatE span a wider range of tolerable distances (more than 8 in both cases), but ultimately also suffer substantial degradation. TransE and RotatE maintain a recall above 0.9 up to and including a Euclidean distance of 5. The SNR perspective reveals that RotatE climbs most steeply toward full recall at low SNR values, suggesting that its rotational scoring function is inherently more tolerant of noise in decompressed embeddings. Across both datasets, the results establish a clear, model-dependent threshold: below it, vector quantization preserves near-perfect link prediction recall, and above it, performance deteriorates rapidly.

Usable thresholds for VQ Based on the results we present in Figure 2, we can prescribe specific K and D values for VQ that result in a small difference between decompressed embeddings $\hat{\mathbf{e}}$ and $\mathbf{e}$ as seen in Table 3. We make this prescription so that Hits@10 recall is always above 0.9. In practice, experts can follow these thresholds grounded empirically for aforementioned datasets and models and validate them for unseen KGs. Additionally, we relate these thresholds to specific values of Euclidean distance and SNR, thereby informing the minimum similarity required for any KGE approximation to maintain recall in link prediction tasks. These similarity metrics can easily be integrated into any approximation method to serve as a guide or stopping criterion. Furthermore, Figure 4 shows the results of fitting each $\langle model, dataset \rangle$ curve in Figure 2 and allows us to corroborate the results presented in Figure 2. Because Euclidean distance and SNR move in opposite directions, the two columns are read differently: recall stays high while Euclidean distance is *at or below* the listed value, and while SNR is *at or above* it. The table also makes the model dependence explicit: ComplEx on WN18RR tolerates a Euclidean departure of only 1.46 before recall drops below 0.9, whereas TransE on the same dataset withstands 5.77.

4.3.2. Impact of triple-based Partitioning on Link Prediction

We next examine how the use of triple-based partitioning affects link prediction recall. We remind readers that, using triple-based partitioning, we first score and rank the groups based on their aggregated embeddings to produce a list σ. We keep the Top $-$ N groups of that list, and then we rank the individual entity embeddings within those Top $-$ N groups to produce a ranking of entities ρ'. How this affects Hits@10 performance depends on the minimum group size, τ, and the number of Top $-$ N groups retained. Table 4 presents the heatmap in which each cell shows the ratio of computational work relative to standard (per-entity) link prediction, while the color encodes the fraction of the original Hits@10 that is retained. Note that lighter colors (yellow, light green) indicate we can retain recall, while darker



(a) Y axis: Hits@10 recall vs. X axis: Average Euclidean distance

(b) Y axis: Hits@10 recall vs. X axis: Average Signal-to-Noise Ratio

Figure 2: Vector quantization: Hits@10 recall as a function of the average Euclidean distance (a) and Signal-to-Noise Ratio (b) between original and reconstructed embeddings as the compression ratio increases.

colors indicate degradation. The reason for the variance in the work ratio between models is that the ranking of groups, σ, varies between models. Meaning that different groups that vary in size are in the Top $-$ N. This is also why the Hits@10 recall differs between models, since different entities are considered for the final ranking ρ'.

Triple-based partitioning on FB15k-237 On FB15k237, setting τ to 5 produces many small fine-grained entity groups. Across all models, the work ratio exceeds 1.0, while Hits@10 recall is around 0.8; i.e., link prediction requires more computation than the original per-entity evaluation, and Hits@10 performance also degrades. When τ is set to 10, the work ratio across all models is between 57% and 63%, and we start to see more differentiation between models with regard to the Hits@10 recall. Depending on Top $-$ N value, the ComplEx model's recall ranges from 0.75 to 0.85. RotatE achieves Hits@10 recall between 0.85 and 0.95, whereas the TransE model has a steady recall of 0.85 regardless of the Top $-$ N value. As τ increases to 50 or 100, the work ratio drops by varying amounts for each model. For ComplEx, the work ratio ranges from 6% to 12% depending on τ and Top $-$ N; the Hits@10 recall hovers at 0.6. Still with τ at 50 and 100, the work ratio measured for the RotatE model ranges from 20% to 42%, with the Hits@10 recall never dipping below 0.9. For the same τ values, TransE manages to retain a recall of 0.8 while having a work ratio of 12% to 25%. At τ values 200 and 300, both ComplEx and TransE perform similarly as they did when τ is set to 5 and 10. The work ratio is lower, however, ranging from 13% to 36% for ComplEx and 18% to 52% for TransE. On the other hand, RotatE has near-perfect Hits@10 recall $\geq$ 0.95, while doing 34% to 62% of the work. When τ is set to 1000, all models are able to achieve $\geq$ 0.97 Hits@10 recall depending on the Top $-$ N value, while the work ratio approaches 100%. This is because the groups of entities are so large that almost all entities are considered in the final ranking ρ'.

Triple-based partitioning on WN18RR On WN18RR the behavior is markedly different across models. ComplEx is extremely sensitive: recall remains below 15% for $\tau \leq 50$ regardless of Top $-$ N, then jumps sharply to above 85% at τ=100 when Top $-$ N= 20, and reaches perfect recall at $\tau \geq 300$, however this is due to the amount of work done is equal to ranking all entities as the Top $-$ N groups encompass all the entities of the KG for $N \geq 300$. In contrast, RotatE and TransE degrade more gracefully, with recall exceeding 95% already at τ=50 for TransE and at $\tau = 10$ for RotatE. This suggests that the structural grouping imposed by triple-based partitioning aligns better with the geometry of translational and rotational embedding spaces than with the bilinear decomposition used by ComplEx. Additionally, the difference between datasets can be attributed to the difference in the structure of the two Knowledge Graphs. WN18RR has a total of 10 distinct relations and 40,942 entities, whereas FB15k237 contains 237 distinct relations and 14,540. So during the creation of groups of entities WN18RR will naturally produce fewer and larger groups of entities as compared to FB15k237.

		ComplEx			RotatE			TransE		
	τ	$N{=}5$	$N{=}10$	$N{=}20$	$N{=}5$	$N{=}10$	$N{=}20$	$N{=}5$	$N{=}10$	$N{=}20$
FB15k237	5	1.273	1.275	1.277	1.294	1.300	1.310	1.293	1.298	1.306
	10	0.571	0.573	0.577	0.599	0.612	0.632	0.591	0.598	0.610
	50	0.078	0.086	0.099	0.201	0.253	0.331	0.117	0.175	0.193
	100	0.057	0.074	0.114	0.280	0.348	0.423	0.125	0.174	0.248
	200	0.136	0.188	0.300	0.342	0.460	0.576	0.184	0.284	0.447
	300	0.155	0.228	0.363	0.395	0.534	0.619	0.239	0.386	0.516
	1000	0.536	0.967	0.999	0.662	0.994	0.999	0.586	0.980	0.999
WN18RR	5	0.068	0.069	0.071	0.232	0.329	0.397	0.102	0.141	0.218
	10	0.023	0.026	0.031	0.413	0.519	0.543	0.182	0.297	0.405
	50	0.016	0.030	0.058	0.761	0.791	0.820	0.646	0.777	0.822
	100	0.026	0.078	0.832	0.860	0.887	0.940	0.851	0.889	0.940
	200	0.081	0.770	0.996	0.912	0.949	0.996	0.913	0.949	0.996
	300	0.996	0.996	0.996	0.996	0.996	0.996	0.996	0.996	0.996
	1000	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995	0.995

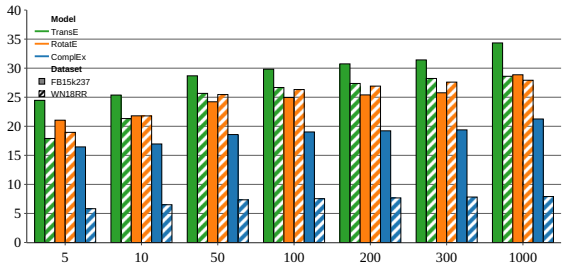
Hits@10 Recall

0 .2 .4 .6 .8 1

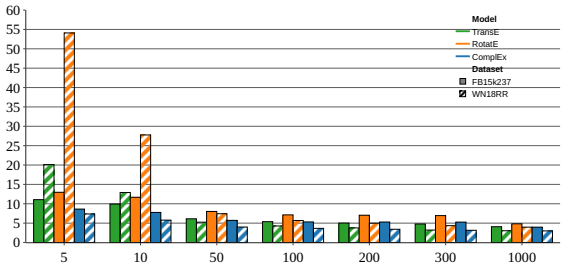
Table 4
Ratio of computational work relative to per-entity (regular) link prediction, as a function of minimum group size τ (rows) and number of Top $-$ N groups retained (columns). Values < 1 indicate less work than the baseline; values > 1 indicate more. Cell color encodes retained Hits@10 recall (viridis scale, see color bar).

Embedding deviation vs. minimum entity group size Figure 3 provides a complementary view to Figure 2 by plotting the average Euclidean distance and SNR between entity embeddings and their aggregate embeddings as a function of τ. Across all models, as τ increases, entities are added into larger, more heterogeneous groups, which steadily increases the Euclidean distance and decreases the SNR. In Figure 3(a), we can see that across all models this effect is evident, since, at every increase of τ, the aggregate embeddings of entity groups have an increased average Euclidean distance from their enclosed entities. In Figure 3(b), we can see the same effect but for the Signal-to-Noise Ratio, where every increase of τ leads to smaller values of the Signal-to-Noise Ratio. We remind that higher SNR values indicate higher similarity between an entity embedding and its group's aggregate embedding. An interesting observation emerges from a combination of Figures 2 and 3, and Table 4. When using the triple-based partitioning approach for link prediction, near-perfect Hits@10 recall is achievable despite consistently higher Euclidean distances than the average distance between decompressed VQ embeddings and their original representations. The same observation applies when examining the SNR values.

However, we note that this comparison spans two structurally different roles. In VQ, the approximate embedding $\hat{\mathbf{e}}$ is used directly as the scoring vector for every candidate entity, so any deviation from $\mathbf{e}$ propagates into the final ranking. In Triple-based partitioning, the aggregate embedding $\mathbf{s}$ is only used to rank and filter candidate groups. The entities within the retained Top $-$ N groups are subsequently scored using their original, unmodified embeddings. Consequently, deviation in $\mathbf{s}$ only risks excluding the correct entity's group from consideration. It does not directly distort the final score. This asymmetry explains why aggregate embeddings can exhibit substantially larger Euclidean distance (and lower SNR) than the VQ-derived thresholds while still preserving near-perfect Hits@10 recall. Thus, our results provide evidence of the degree of similarity that an aggregate embedding must have with respect to the entities it aggregates in order to be useful for link prediction.

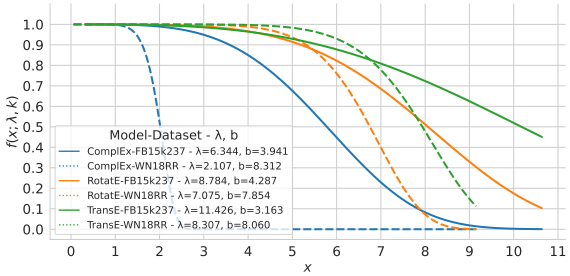


(a) Y axis: Euclidean Distance vs. X axis: Minimum Group Size

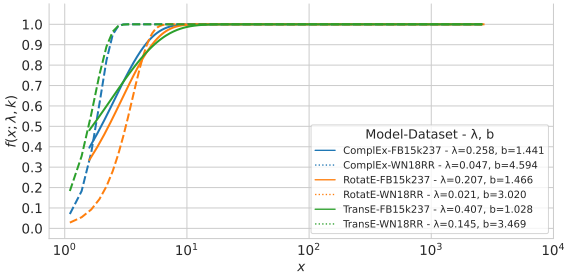


(b) Y axis: SNR vs. X axis: Minimum Group Size

Figure 3: Average embedding departure as a function of the minimum group size τ measured by Euclidean distance (left) and Signal-to-Noise Ratio (right). Larger groups produce more heterogeneous groups and consequently greater distance from the original entity embeddings.



(a) Euclidean distance relationship with Hits@10 Recall follows the Weibull reliability function:
$f(x; \lambda, b) = e^{-\left(\frac{x}{\lambda}\right)^b}$



(b) Signal-to-Noise Ratio relationship with Hits@10 recall follows the Weibull CDF:
$f(x; \lambda, b) = 1 - e^{-\lambda \cdot x^b}$

Figure 4: Relationship between the (a) Euclidean distance and (b) Signal-to-Noise Ratio, and the Hits@10 Recall follow the (a) Weibull reliability function and (b) Weibull cumulative density function, respectively. *Note: In (a) and (b) e is Euler's number.*

4.3.3. Compression-Usability Trade-Off

We now quantify how much an embedding can be approximated before it loses its usefulness for link prediction through the lens of Rate-Distortion Theory (RDT). RDT studies the trade-off between the *rate* of a lossy compression scheme (how much the representation is altered) and the *distortion* of the reconstructed data (how much fidelity is lost). It establishes that a relationship between rate and distortion must exist, without prescribing its functional form [28]; that form is captured by a problem-specific rate-distortion function. In signal processing, for example, it has been shown that there is a specific relationship between the number of samples taken of a signal and the resulting Mean Squared Error of the reconstructed analog wave of that signal [29]. We frame our analysis using RDT because VQ is a lossy compression method that approximates the original embeddings, and we want a principled way to quantify how this approximation affects downstream performance. This is similar to other uses of RDT when it comes to data representation [30]. We will then argue that for successful link prediction, the information stored in an individual embedding must not deviate beyond a certain threshold. We then argue that aggregate-based approaches require less information to be present in the aggregate embedding to perform well.

We do this analysis in order to show the precise relationship between the *rate*, i.e., the amount of compression applied, and the *functional distortion*, i.e., the Hits@10 recall. We then want to use this relationship to explain how it can guide the application of compression to KGEs and to validate the concrete compression thresholds provided in Table 3.

Based on our experimental results, we show that a mathematical relationship describes how our

compression method loses information and introduces distortion. That is:

- **Rate:** In this work, the rate is measured as the metrics, Euclidean distance, and the Signal-to-Noise Ratio. Both of these metrics are determined by the amount of compression we apply, i.e., as the compression ratio increases, these metrics show a greater difference between the decompressed and original embeddings. By using two separate metrics to measure the amount of information in the embedding representations, we provide a more robust argument for describing the thresholds and trade-offs of the VQ method. We use the difference metrics as our rate since they are strongly determined by the compression ratio. We acknowledge that there is some local non-monotonic behavior depending on how we vary compression parameters K and D, seen in Figure 2.
- **Distortion:** In this work, the distortion is measured as Hits@10 recall, as it indicates how well we can still perform the desired task after decompression. By doing this, we measure functional distortion, i.e., whether the data can still perform the task it was originally meant to perform.

After fitting the results of experiments in Figure 2, we find the functions in Figure 4(a) that show that the relationship of the Euclidean distance and the Hits@10 recall is that of the Weibull reliability function. Whereas the relationship of the Signal-to-Noise Ratio and the Hits@10 recall follows the cumulative density function (CDF), as seen in Figure 4(b). As such, we have identified the specific Rate-Distortion functions that describe the relationship between the amount of information that must be stored in embeddings and their usefulness for link prediction. The Weibull distribution is used to model reliability or decay in a variety of systems [31, 32, 33]. This distribution describes how a process begins to fail or how a signal begins to decay, i.e., the probability of failure at any given moment in time. In our work, the Weibull functions precisely capture the relationship between the rate of change in the numerical representation of each entity in the low-dimensional space and the usefulness of the distorted representation for the downstream task, per model. Furthermore, both metrics are related to the same distribution: the Weibull distribution. This shows a consistent decay pattern described in our experiments by a Weibull function across the models and datasets we evaluated. Hence, it suggests that the resulting Weibull structure reflects how the compression procedure degrades the encoded information across a set of KGEs. Our results also show that the rate of information loss is dependent on both the model and the dataset. Crucially, the Weibull form is *not* assumed in advance: after collecting the empirical data in Figure 2, we fit candidate decay functions and find the relationships described in Figure 4. We emphasize this because it means that we have measured an effect of compression and have not guided our experiments to find the set of experiments that best fit the Weibull distribution.

5. Conclusion

In this paper, we studied the extent to which entity embeddings can be approximated before link prediction quality degrades. We compare two state-of-the-art methods: vector quantization and triple-based partitioning on the embeddings produced by the TransE, RotatE, and ComplEx models trained on FB15K237 and WN18RR. In doing so, we have shown the limits to which these methods can be employed to reduce memory and compute resources while maintaining link prediction performance. We can prescribe concrete thresholds for Euclidean distance and SNR to guide compression and aggregation applications while preserving performance; in our experiments, the safe thresholds are determined per dataset (Table 3). We further show that the aggregate embedding produced by triple-based partitioning can *exceed* these VQ-derived thresholds for Euclidean Distance and SNR while still retaining Hits@10 performance. Thus, showing that even when an entity's group-aggregate embedding lies well beyond the distance at which per-entity VQ would fail, structure-aware grouping can preserve recall. Based on our results, interested practitioners can select parameters for their embedding modification or aggregation approaches from a narrower range rather than by trial and error. Additionally, based on our experimentation of these $\langle$model, dataset$\rangle$ pairs, we show that the relationship between how much approximation of embeddings and link prediction performance is captured by a Rate-Distortion function, with the Euclidean and SNR views following the Weibull reliability function and CDF, respectively.

Because both metrics fall within the same distribution, we have experimentally demonstrated the degradation caused by compression through two lenses.

Promising future work includes deriving the Weibull parameters analytically from model and dataset statistics (avoiding a fitting step) and testing the results on million-entity KGs such as Wikidata to validate the results on large datasets. It is also conceivable to combine triple-based partitioning with vector quantization to leverage the benefits of both reduced computational requirements and memory usage. This means that the aggregate embeddings would be created before the quantization of individual entity embeddings, so they could be used to limit the candidate set of entities before they are decompressed and ranked.

Acknowledgments

This project was partially funded by the Vienna Science and Technology Fund (WWTF) [DORSET, GrantID: 10.47379/ICT25032].

Declaration on Generative AI

During the preparation of this work, the author(s) used Claude Sonnet 4.6 and Opus 4.6 in order to: Grammar and spelling check. After using these tool(s)/service(s), the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] D. Diefenbach, V. Lopez, K. Singh, P. Maret, Core techniques of question answering systems over knowledge bases: a survey, Knowledge and Information systems 55 (2018) 529–569.

[2] D. Calvanese, G. De Giacomo, D. Lembo, M. Lenzerini, R. Rosati, G. A. Ruberti, Ontology-based data access and integration, Encyclopedia of Database Systems 2 (2017) 39–71.

[3] A. Madotto, C.-S. Wu, P. Fung, Mem2Seq: Effectively Incorporating Knowledge Bases into End-to-End Task-Oriented Dialog Systems, in: Proceedings of ACL, ACL, Melbourne, Australia, 2018, pp. 1468–1478.

[4] E. Kharlamov, S. Brandt, E. Jimenez-Ruiz, Y. Kotidis, S. Lamparter, T. Mailis, C. Neuenstadt, Ö. Özçep, C. Pinkel, C. Svingos, et al., Ontology-Based Integration of Streaming and Static Relational Data with Optique, in: Proceedings of SIGMOD, ACM, San Francisco, CA, USA, 2016, pp. 2109–2112.

[5] D. Seyler, T. Dembelova, L. Del Corro, J. Hoffart, G. Weikum, A Study of the Importance of External Knowledge in the Named Entity Recognition Task, in: Proceedings of ACL, ACL, Melbourne, Australia, 2018, pp. 241–246.

[6] B. Peng, Y. Zhu, Y. Liu, X. Bo, H. Shi, C. Hong, Y. Zhang, S. Tang, Graph Retrieval-Augmented Generation: A Survey, ACM Transactions on Information Systems 44 (2025) 35:1–35:52. doi:10.1145/3777378.

[7] D. Vrandečić, M. Krötzsch, Wikidata: a free collaborative knowledge base, Communications of the ACM 57 (2014) 78–85.

[8] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, others, DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia, Semantic Web 6 (2015) 167–195.

[9] M. Nickel, K. Murphy, V. Tresp, E. Gabrilovich, A Review of Relational Machine Learning for Knowledge Graphs, Proceedings of the IEEE 104 (2016) 11–33.

[10] H. Cai, V. W. Zheng, K. C. Chang, A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications, IEEE Transactions on Knowledge and Data Engineering 30 (2018) 1616–1637.

[11] M. Sachan, Knowledge graph embedding compression, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), 2020, pp. 2681–2691.

[12] U. Joshi, J. Urbani, Searching for embeddings in a haystack: Link prediction on knowledge graphs with subgraph pruning, in: Proceedings of The Web Conference (WWW), ACM, Taipei, Taiwan, 2020, pp. 2817–2823.

[13] N. Sloane, A. Wyner, Coding theorems for a discrete source with a fidelity criterioninstitute of radio engineers, international convention record, vol. 7, 1959. (2009) 325–350. doi:10.1109/9780470544242.ch21.

[14] C. Meilicke, M. W. Chekol, P. Betz, M. Fink, H. Stuckenschmidt, Anytime bottom-up rule learning for large-scale knowledge graph completion, The VLDB Journal 33 (2024) 131–161. doi:10.1007/s00778-023-00800-5.

[15] L. A. Galárraga, C. Teflioudi, K. Hose, F. Suchanek, AMIE: Association rule mining under incomplete evidence in ontological knowledge bases, in: Proceedings of the 22nd International Conference on World Wide Web, WWW '13, Association for Computing Machinery, New York, NY, USA, 2013, pp. 413–422. doi:10.1145/2488388.2488425.

[16] N. Lao, T. Mitchell, W. W. Cohen, Random walk inference and learning in a large scale knowledge base, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '11, Association for Computational Linguistics, USA, 2011, pp. 529–539.

[17] A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, O. Yakhnenko, Translating Embeddings for Modeling Multi-relational Data, in: Proceedings of NIPS, NIPS, Lake Tahoe, NV, USA, 2013, pp. 2787–2795.

[18] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, G. Bouchard, Complex embeddings for simple link prediction, in: Proceedings of ICML, 2016, pp. 2071–2080.

[19] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: International Conference on Learning Representations (ICLR), 2015. URL: https://arxiv.org/abs/1412.6980.

[20] S. Broscheit, D. Ruffinelli, A. Kochsiek, P. Betz, R. Gemulla, LibKGE - A knowledge graph embedding library for reproducible research, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, 2020, pp. 165–174. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.22.

[21] A. Rossi, D. Barbosa, D. Firmani, A. Matinata, P. Merialdo, Knowledge Graph Embedding for Link Prediction: A Comparative Analysis, ACM Trans. Knowl. Discov. Data 15 (2021) 14:1–14:49. doi:10.1145/3424672.

[22] M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, M. Galkin, S. Sharifzadeh, A. Fischer, V. Tresp, J. 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 (2022) 8825–8845. doi:10.1109/TPAMI.2021.3124805.

[23] K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, M. Gamon, Representing Text for Joint Embedding of Text and Knowledge Bases, in: L. Màrquez, C. Callison-Burch, J. Su (Eds.), Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal, 2015, pp. 1499–1509. doi:10.18653/v1/D15-1174.

[24] T. Dettmers, P. Minervini, P. Stenetorp, S. Riedel, Convolutional 2D knowledge graph embeddings, in: Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI'18/IAAI'18/EAAI'18, AAAI Press, New Orleans, Louisiana, USA, 2018, pp. 1811–1818.

[25] Z. Sun, Z.-H. Deng, J.-Y. Nie, J. Tang, Rotate: Knowledge graph embedding by relational rotation in complex space, in: Proceedings of ICLR, 2019.

[26] R. Abboud, I. Ceylan, T. Lukasiewicz, T. Salvatori, BoxE: A Box Embedding Model for Knowledge Base Completion, in: Advances in Neural Information Processing Systems, volume 33, Curran Associates, Inc., 2020, pp. 9649–9661.

[27] A. Pavlović, E. Sallinger, SpeedE: Euclidean Geometric Knowledge Graph Embedding Strikes Back, in: K. Duh, H. Gomez, S. Bethard (Eds.), Findings of the Association for Computational Linguistics: NAACL 2024, Association for Computational Linguistics, Mexico City, Mexico, 2024, pp. 69–92.

doi:10.18653/v1/2024.findings-naacl.6.

[28] C. E. Shannon, Coding Theorems for a Discrete Source With a Fidelity CriterionInstitute of Radio Engineers, International Convention Record, vol. 7, 1959., in: Claude E. Shannon: Collected Papers, IEEE, 1993, pp. 325–350. doi:10.1109/9780470544242.ch21.

[29] A. Kipnis, A. J. Goldsmith, Y. C. Eldar, Sub-Nyquist sampling achieves optimal rate-distortion, in: 2015 IEEE Information Theory Workshop (ITW), 2015, pp. 1–5. doi:10.1109/ITW.2015.7133113.

[30] Y. Dubois, B. Bloem-Reddy, K. Ullrich, C. Maddison, Lossy Compression for Lossless Prediction, in: Advances in Neural Information Processing Systems, volume 34, Curran Associates, Inc., 2021, pp. 14014–14028.

[31] N. Sagias, G. Karagiannidis, Gaussian class multivariate Weibull distributions: Theory and applications in fading channels, IEEE Transactions on Information Theory 51 (2005) 3608–3619. doi:10.1109/TIT.2005.855598.

[32] L. M. Cunha, F. A. R. Oliveira, J. C. Oliveira, Optimal experimental design for estimating the kinetic parameters of processes described by the Weibull probability distribution function, Journal of Food Engineering 37 (1998) 175–191. doi:10.1016/S0260-8774(98)00085-5.

[33] H.-L. Yang, Two-warehouse partial backlogging inventory models with three-parameter Weibull distribution deterioration under inflation, International Journal of Production Economics 138 (2012) 107–116. doi:10.1016/j.ijpe.2012.03.007.