Knowledge Sheaves:
A Sheaf-Theoretic Framework for Knowledge Graph Embedding

Thomas Gebhart
University of Minnesota
Department of Computer Science

Jakob Hansen
BlueLightAI, Inc.

Paul Schrater
University of Minnesota
Department of Computer Science

Abstract

Knowledge graph embedding involves learning representations of entities—the vertices of the graph—and relations—the edges of the graph—such that the resulting representations encode the known factual information represented by the knowledge graph and can be used in the inference of new relations. We show that knowledge graph embedding is naturally expressed in the topological and categorical language of *cellular sheaves*: a knowledge graph embedding can be described as an approximate global section of an appropriate *knowledge sheaf* over the graph, with consistency constraints induced by the knowledge graph's schema. This approach provides a generalized framework for reasoning about knowledge graph embedding models and allows for the expression of a wide range of prior constraints on embeddings. Further, the resulting embeddings can be easily adapted for reasoning over composite relations without special training. We implement these ideas to highlight the benefits of the extensions inspired by this new perspective.

1 INTRODUCTION

Knowledge graphs are structured knowledge bases which encode information about entities and their relationships. Each graph is a set of triplets—two entities and a relation between them—that represent facts about a domain. Modern knowledge graphs may contain millions of entities and billions of relational facts. As a result, efficiently representing knowledge graphs in a way amenable to large-scale inferential analysis is of great practical importance in areas such as biomedicine (Santos et al., 2022; Hoyt et al., 2022), content

recommendation (Sun et al., 2019b; Guo et al., 2020), and scientific discovery (Sinha et al., 2015; Auer et al., 2018).

Knowledge graph embedding (KGE) has emerged as an important approach to encoding this type of data. In this approach, a representation of each entity and relation is learned, enabling tasks like knowledge graph completion, multi-hop reasoning, and other forms of inference (Chen et al., 2020; Ji et al., 2020). A wide variety of knowledge graph embedding approaches exist (Bordes et al., 2013; Trouillon and Nickel, 2017; Nickel et al., 2016; Ji et al., 2016; Wang et al., 2014; Lin et al., 2017; Socher et al., 2013; Jenatton et al., 2012; Bordes et al., 2011; Zhang et al., 2019), each with its own motivation and theoretical framework.

Various taxonomies have been proposed to classify these approaches (Ji et al., 2020), but they remain far from exhausting the space of possibilities. Describing and enforcing priors about the entities and relations by embedding in more complex spaces has attracted recent interest (Zhang et al., 2019; Sun et al., 2019a; Ebisu and Ichise, 2017), but a general framework for these types of constraints is still lacking. We express the knowledge graph embedding task in a formal framework that illuminates the structural similarity across a variety of prior methods while also inspiring new possibilities for this family of embedding techniques.

Our approach stems from an algebro-topological perspective, using cellular sheaves (Curry, 2014) and their Laplacians (Hansen and Ghrist, 2019) to construct a framework for knowledge graph embedding. Many of the goals of KGE—local agreement across relations, global consistency, typed representations, and multi-hop reasoning—can be readily formalized and interpreted in this sheaf-theoretic language. A number of well-known KGE models may be subsumed within this framework, providing a uniform language for reasoning about the regularization and extension of existing models. This perspective also provides new benefits: the freedom to embed entities in spaces of varying dimension according to type semantics, control over symmetry or antisymmetry of relations, and a natural method (*harmonic extension*) for reasoning over multi-hop queries. We implement these ideas and validate their performance on a range of complex query structures within two bench-

mark datasets, showcasing a theoretically-sound method for adapting knowledge graph embeddings trained on simple knowledge graph completion to more complex queries.

2 KNOWLEDGE GRAPHS

A knowledge graph is often vaguely defined as a set of entities $\mathcal{E}$ together with a set of relations $\mathcal{R}$ between these entities. To facilitate the connection with cellular sheaves, we provide a more formal definition.

Definition 1 *Let $\mathcal{S}$ be a set of entity types, and $\mathcal{R}$ a set of relations. Suppose that each relation $r \in \mathcal{R}$ may hold between an entity of type $\mathfrak{h}(r) \in \mathcal{S}$ and an entity of type $\mathfrak{t}(r) \in \mathcal{S}$. The tuple $\mathcal{Q} = (\mathcal{S}, \mathcal{R}, \mathfrak{h}, \mathfrak{t})$ is a* knowledge database schema.

Recall that a directed multigraph consists of a set V of vertices and a set E of edges, with two functions $h, t : E \to V$, where an edge $e \in E$ is thought of as going from $h(e)$ to $t(e)$. Note that under this definition a knowledge database schema is simply a directed multigraph, where each entity type is a vertex and each relation is an edge.

A knowledge graph then instantiates a schema in the form of a set of factual triplets which respect the typing from $\mathcal{Q}$:

Definition 2 *Given a knowledge database schema $\mathcal{Q} = (\mathcal{S}, \mathcal{R}, \mathfrak{h}, \mathfrak{t})$, a set $\mathcal{E}$ of entities, and a labeling function $s : \mathcal{E} \to \mathcal{S}$ which gives the type of each entity, a* knowledge graph G *instantiating $\mathcal{Q}$ is a directed graph with vertex set $\mathcal{E}$ and edges $\mathcal{T} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ whose elements $(h, r, t) \in \mathcal{T}$ must satisfy the type consistency conditions $\mathfrak{h}(r) = s(h)$ and $\mathfrak{t}(r) = s(t)$.*

The relationship between the knowledge graph G and its schema $\mathcal{Q}$ is captured by a *graph morphism*.

Definition 3 *Let G and H be directed multigraphs. A graph morphism[1] $k : G \to H$ consists of a pair of functions $k_v : V(G) \to V(H)$ and $k_e : E(G) \to E(H)$ such that, for every edge e of G, $h(k_e(e)) = k_v(h(e))$ and $t(k_e(e)) = k_v(t(e))$. That is, k maps nodes to nodes and edges to edges in a way that respects the incidence relation.*

The type consistency condition on G is precisely the requirement that the obvious map $G \to \mathcal{Q}$ be a graph morphism. For clarity, a simple example of a knowledge graph with schema is sketched in the appendix.

The schema $\mathcal{Q}$ and graph morphism k are often given externally and can be quite simplistic, so it is common to refer to G itself as the knowledge graph without reference to the other type-enforcing structures. Indeed, many popular benchmark datasets for knowledge graph embedding

---

[1]It should be noted that there are other, slightly different definitions of this term; we have chosen this one because it appropriately models the structure of knowledge graphs.

assume only one entity type (one vertex in $\mathcal{Q}$) with a collection of relations mapping this type to itself. This typing scheme is often chosen for convenience and may deserve further consideration if one wishes to embed hierarchical or other type-specific information within knowledge graph embeddings (Jain et al., 2021).

2.1 Knowledge Graph Embedding

The problem of knowledge graph embedding is, broadly speaking, that of finding representations $\boldsymbol{x}_h$ for each entity h and representations $\boldsymbol{R}_r$ for each relation type r such that the truth value of the tuple (h, r, t) may be recovered from $(\boldsymbol{x}_h, \boldsymbol{R}_r, \boldsymbol{x}_t)$. Note that the entities being embedded are the vertices of G, while the relations being embedded are the edges of $\mathcal{Q}$; the edges whose existence we want to be able to infer are the edges of G. A common inferential goal for a knowledge graph embedding is to predict the truth value of new triples from their learned representations alone. The predicted truth value of such a relation is typically determined from the output of some scoring function which takes a vectorized triplet to a real number representing the model's degree of confidence in its truth.

The classic Structured Embedding model (Bordes et al., 2011) represents each entity as a vector $\boldsymbol{x}_e \in \mathbb{R}^d$ and each relation r as a pair of $d \times d$ matrices $(\boldsymbol{R}_{rh}, \boldsymbol{R}_{rt})$, and uses the scoring function $f^{SE}(h, r, t) = \|\boldsymbol{R}_{rh}\boldsymbol{x}_h - \boldsymbol{R}_{rt}\boldsymbol{x}_t\|^2$. That is, a relation embedding provides a pair of linear transformations applied to the head and tail entity embeddings before comparing them. This model is a motivating example for many other embedding methods, and is a good starting point for the generalization to sheaf embedding models.

3 CELLULAR SHEAVES

Abstractly, a sheaf is a mathematical object which tracks the assignment of data to open sets of a topological space. While sheaf theory has existed for nearly a century, only in the past decade has a discretized and computationally tractable theory of *cellular sheaves* received significant attention and development (Curry, 2014). While the theory behind these structures can become quite intricate, all the machinery we will need can be explained using basic graph theory and linear algebra. Our introduction here will be specifically adapted to the motivating example of knowledge graphs.

Definition 4 *A cellular sheaf $\mathcal{F}$ on a directed graph $G = (V, E)$ consists of the following data:*

- *a vector space $\mathcal{F}(v)$ for each vertex $v \in V$ of G*

- *a vector space $\mathcal{F}(e)$ for each edge $e \in E$ of G,*

- *a linear transformation $\mathcal{F}_{v \trianglelefteq_h e} : \mathcal{F}(v) \to \mathcal{F}(e)$ for each pair (v, e) with $h(e) = v$, and*

- *a linear transformation $\mathcal{F}_{v \trianglelefteq_t e} : \mathcal{F}(v) \rightarrow \mathcal{F}(e)$ for each pair (v, e) with $t(e) = v$.*

The sheaf structure over a graph associates a space of data, called a *stalk*, to each node and edge. We refer to the linear maps between stalks as *restriction maps*. For most pairs (v, e), at most one of the restriction maps $\mathcal{F}_{v \trianglelefteq_h e}$ and $\mathcal{F}_{v \trianglelefteq_t e}$ can exist, in which case we simplify the notation to $\mathcal{F}_{v \trianglelefteq e}$. The only exception is when e is a self-loop on the vertex v, when both maps exist and may be distinct.

If G is a knowledge graph, we can think of each vertex stalk $\mathcal{F}(v)$ as a vector space from which we can choose a representation of the entity v. The restriction maps $\mathcal{F}_{v \trianglelefteq e}$ are used to compare entity embeddings with respect to an edge representing a particular relation, and the edge stalks $\mathcal{F}(e)$ are the spaces in which this comparison happens.

Another way to say this is that the restriction maps of $\mathcal{F}$ encode consistency constraints for entity embeddings. For an edge e between vertices u and v, we say that a choice of two embeddings $\boldsymbol{x}_v \in \mathcal{F}(v)$, $\boldsymbol{x}_u \in \mathcal{F}(u)$ is *consistent over* e if $\mathcal{F}_{v \trianglelefteq e} \boldsymbol{x}_v = \mathcal{F}_{u \trianglelefteq e} \boldsymbol{x}_u$. In Structured Embedding, all stalks are the same $\mathbb{R}^d$, and the relation embedding matrices become the restriction maps. The score for a relation is zero if the entity embeddings are consistent over that edge.

The space of all possible entity embeddings (i.e., a choice of an embedding vector for each entity) is known as the space of 0-cochains of $\mathcal{F}$, and is denoted $C^0(G; \mathcal{F})$. Each edge of G imposes a constraint on $C^0(G; \mathcal{F})$ by requiring the data over its two incident vertices to be consistent over the edge. The subspace of $H^0(G; \mathcal{F}) \subseteq C^0(G; \mathcal{F})$ consisting of cochains that satisfy all these local constraints is called the space of *global sections* of $\mathcal{F}$.

Similarly, the space of all choices of one vector in each edge stalk $\mathcal{F}(e)$ is the space of 1-cochains $C^1(G; \mathcal{F})$. The space of global sections $H^0(G; \mathcal{F})$ is the nullspace of a linear transformation $\boldsymbol{\delta} : C^0(G; \mathcal{F}) \rightarrow C^1(G; \mathcal{F})$ whose output on an edge $e : u \rightarrow v$ is given by the formula

$$(\boldsymbol{\delta x})_e = \mathcal{F}_{v \trianglelefteq e} \boldsymbol{x}_v - \mathcal{F}_{u \trianglelefteq e} \boldsymbol{x}_u$$

where the edge e is directed $u \rightarrow v$. Therefore, if $\boldsymbol{\delta x} = 0$, then $\mathcal{F}_{v \trianglelefteq e} \boldsymbol{x}_v = \mathcal{F}_{u \trianglelefteq e} \boldsymbol{x}_u$ for every edge $e = u \sim v$. From the coboundary operator we may construct the *sheaf Laplacian* $\boldsymbol{L}_{\mathcal{F}} = \boldsymbol{\delta}^T \boldsymbol{\delta}$ (Hansen and Ghrist, 2019).

The sheaf Laplacian provides a continuous measure of consistency for potential entity embeddings. As the elements of $H^0(G; \mathcal{F})$ represent globally consistent choices of data on vertices, we can use the sheaf Laplacian to quantify how close any data assignment in $C^0(G; \mathcal{F})$ is to consistency:

$$\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}} \boldsymbol{x} = \sum_{e=u \sim v \in E} \| \mathcal{F}_{u \trianglelefteq e} \boldsymbol{x}_u - \mathcal{F}_{v \trianglelefteq e} \boldsymbol{x}_v \|^2. \quad (1)$$

The closer $\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}} \boldsymbol{x}$ is to zero, the closer the section $\boldsymbol{x}$ is to consistency, and when $\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}} \boldsymbol{x} = 0$, $\boldsymbol{x}$ is fully consistent.

Note that each individual term in this sum is of the same form as the Structured Embedding scoring function. As we will observe, a large family of knowledge graph embedding methods implicitly target similar notions of global consistency in the optimization of entity and relation embeddings derived from knowledge graph triplets.

To fully exploit the flexibility of this sheaf-theoretic perspective, we introduce one more tool which allows us to transfer a sheaf from one graph to another. This operation will depend on a graph morphism as described in Definition 3.

Definition 5 *Given multigraphs G and H, a graph morphism $k : G \rightarrow H$ sending nodes to nodes and edges to edges, and $\mathcal{F}$ a sheaf on H, the* pullback sheaf $k^* \mathcal{F}$ *is a sheaf on G with stalks $k^* \mathcal{F}(\sigma) = \mathcal{F}(k(\sigma))$ and restriction maps $\boldsymbol{k}^* \mathcal{F}_{v \trianglelefteq_\bullet e} = \mathcal{F}_{k(v) \trianglelefteq_\bullet k(e)}$.*

This operation replicates the local structure of $\mathcal{F}$ on all the parts of G that map to the same part of H. Note that, for instance, this allows us to restrict a sheaf on G to any subgraph by using the pullback over the inclusion morphism. Cochains of $\mathcal{F}$ may also be pulled back to cochains of $k^* \mathcal{F}$ by a similar process. If $\boldsymbol{x} \in C^i(H; \mathcal{F}^H)$, we define $k^* \boldsymbol{x} \in C^i(G; \mathcal{F})$ by $(k^* \boldsymbol{x})_\sigma = \boldsymbol{x}_{k(\sigma)}$. It is straightforward to show that if $\boldsymbol{x} \in H^0(H; \mathcal{F})$, then $k^* \boldsymbol{x} \in H^0(G; k^* \mathcal{F})$; that is, global sections of the initial sheaf automatically become sections of its pullback.

4 KNOWLEDGE SHEAVES AND EMBEDDINGS

We are now ready to define sheaf embeddings of knowledge graphs, using the concepts introduced above. This approach separates relation embeddings from entity embeddings: relation embeddings are sheaves on the schema graph $\mathcal{Q}$, while entity embeddings are 0-cochains of the pullback of this sheaf to the knowledge graph G. More formally:

Definition 6 *Given a knowledge database schema $\mathcal{Q} = (\mathcal{S}, \mathcal{R}, \mathfrak{h}, \mathfrak{t})$, a knowledge sheaf $\mathcal{F}$ modeled on $\mathcal{Q}$ corresponds to a choice of vertex stalk spaces $\mathcal{F}(s)$ for each entity type $s \in \mathcal{S}$, edge stalk spaces $\mathcal{F}(r)$ for each relation type $r \in \mathcal{R}$, and linear maps $\mathcal{F}_{h \trianglelefteq_h r} : \mathcal{F}(\mathfrak{h}(r)) \rightarrow \mathcal{F}(r)$ and $\mathcal{F}_{t \trianglelefteq_t r} : \mathcal{F}(\mathfrak{t}(r)) \rightarrow \mathcal{F}(r)$ for each $r \in \mathcal{R}$.*

That is, a knowledge sheaf is simply a cellular sheaf on the directed multigraph $\mathcal{Q}$. To make the space of knowledge sheaves on $\mathcal{Q}$ into a vector space, we assign a dimension d_s for the stalk over each vertex (entity type) s and a dimension d_r for the stalk over each edge (relation type) r. This gives a generalized version of Structured Embedding: a relation r that may hold between entities of type s, t is represented by the two restriction maps $\mathcal{F}_{s \trianglelefteq r}$ and $\mathcal{F}_{t \trianglelefteq r}$, which are matrices of shapes $d_r \times d_s$ and $d_r \times d_t$.

To produce the space of entity embeddings for a knowledge

graph corresponding to schema $\mathcal{Q}$, we use a pullback of a knowledge sheaf.

Definition 7 *Given a graph morphism $k : G \to \mathcal{Q}$ instantiating a knowledge graph G from a schema $\mathcal{Q}$, a sheaf embedding of G is a knowledge sheaf $\mathcal{F}$ on $\mathcal{Q}$ together with a 0-cochain $\boldsymbol{x} \in C^0(G; k^*\mathcal{F})$.*

While this definition depends on the sheaf $\mathcal{F}$, the resulting space of possible entity embeddings depends only on the choice of dimension for each stalk of $\mathcal{F}$. The embedding of an entity of type s is simply a d_s-dimensional vector; if there are N entities, their embeddings combine by concatenation into an Nd_s-dimensional vector, which is an element of $C^0(G; k^*\mathcal{F})$.

For convenience and clarity, we will also denote the sheaf $k^*\mathcal{F}$ by $\mathcal{F}^G$. Note that if $H \subseteq G$ is a subgraph, we can restrict the morphism k to the vertices and edges in H and obtain a morphism $k_H : H \to \mathcal{Q}$. We will denote the pullback $k_H^*\mathcal{F}$ by $\mathcal{F}^H$. The restriction of a 0-cochain $\boldsymbol{x}$ of $k^*\mathcal{F}$ to the nodes in H will be denoted $\boldsymbol{x}^H$.

A concrete description of $\mathcal{F}^G$ is as follows: For an entity v, $\mathcal{F}^G(v) = \mathcal{F}(s(v))$, and for an edge $e = (h, r, t)$, $\mathcal{F}^G(e) = \mathcal{F}(r)$. The restriction maps for e are given by $\boldsymbol{\mathcal{F}}^G_{h \lhd e} = \boldsymbol{\mathcal{F}}_{h \lhd r_e}$ and $\boldsymbol{\mathcal{F}}^G_{t \lhd e} = \boldsymbol{\mathcal{F}}_{t \lhd r_e}$.

It is important to observe that not every sheaf on G arises as a knowledge sheaf. The vertex stalk spaces and edge stalk spaces are determined by the types of entities and relations, respectively, and the restriction maps for an edge are determined entirely by the relation described by that edge. Since many edges in G correspond to the same relation type, this is a form of parameter sharing and greatly reduces the complexity of the knowledge graph embedding.

4.1 Consistent and Contrastive Embeddings

Definition 7 specifies the spaces of entity and relation embeddings, but does not capture the desired representational power of the embedding. In most knowledge graph representation literature, this is done by specifying a loss function; we prefer to specify the desired outcome and then construct a corresponding loss function.

Definition 8 *Let $k : G \to \mathcal{Q}$ be a knowledge graph with schema $\mathcal{Q}$. A consistent sheaf embedding of G is a knowledge sheaf $\mathcal{F}$ on $\mathcal{Q}$ together with a section $\boldsymbol{x} \in H^0(G; k^*\mathcal{F})$.*

That is, a consistent sheaf embedding is one where embeddings of true relational facts are consistent as measured by the sheaf restriction maps; if (h, r, t) is a true relation, then $\boldsymbol{\mathcal{F}}^G_{\boldsymbol{h} \lhd \boldsymbol{r}}\boldsymbol{x}_h = \boldsymbol{\mathcal{F}}^G_{\boldsymbol{t} \lhd \boldsymbol{r}}\boldsymbol{x}_t$. However, this definition does not ensure we can distinguish true from false triples. A trivial entity embedding with $\boldsymbol{x}_e = 0$ for every entity e would be consistent, but useless. To distinguish true from false

relations, we need negative examples, which can be seen as forming a new knowledge graph with the same schema. To capture this requirement, we make a new definition:

Definition 9 *Let $k : G \to \mathcal{Q}$ and $\tilde{k} : \tilde{G} \to \mathcal{Q}$ be knowledge graphs with the same schema and vertex sets. Call G the* positive *knowledge graph, containing relations known to be true, and $\tilde{G}$ the* negative *knowledge graph of triples assumed to be false. Let $\tilde{\mathcal{A}}$ be a collection of subgraphs of $\tilde{G}$. A contrastive sheaf embedding of G with respect to $\tilde{G}$ and $\tilde{\mathcal{A}}$ consists of a consistent sheaf embedding of G such that for every $\tilde{H} \in \tilde{\mathcal{A}}$, $\boldsymbol{x}^{\tilde{H}}$ is not a section of $\mathcal{F}^{\tilde{H}}$.*

In other words, the entity and relation embeddings are consistent for every relation in G, and inconsistent for every selected subgraph of $\tilde{G}$. Thus, if $\tilde{\mathcal{A}}$ is the set of all subgraphs of $\tilde{G}$ with two vertices and one edge, a contrastive sheaf embedding will be able to distinguish perfectly between relations in G and relations in $\tilde{G}$ by checking if $\boldsymbol{x}$ is consistent over the relevant edge or not.

The use of negative examples to constrain knowledge graph embeddings is nearly universal in the literature, and there are various standard choices for constructing $\tilde{G}$ (Ali et al., 2020). Taking $\tilde{G}$ to be the complement of G relative to $\mathcal{Q}$ corresponds with the *closed world assumption*: all relations not known to be true must be false. By contrast, the *open world assumption* corresponds to a $\tilde{G}$ with no edges at all, returning us to Definition 8. These extremes are mostly unsuitable for learning embeddings, and intermediate assumptions like the *local closed world assumption* are typically used. This corresponds to constructing $\tilde{G} \to \mathcal{Q}$ by taking the node set of G, and creating an edge $\tilde{e} = u \to v'$ with $\tilde{k}(\tilde{e}) = r$ if there exists a $v \neq v'$ and an edge $e = u \to v$ in G with $k(e) = r$.

In practice, it is often difficult to find an exact section of a sheaf, if one even exists, so we need versions of these definitions that replace the strict equality constraints for consistency with inequalities. This is where scoring functions come into play.

Definition 10 *A* sheaf scoring function *is a function V taking a graph G, a sheaf $\mathcal{F}$ on G, and a 0-cochain $\boldsymbol{x}$ of $\mathcal{F}$, such that $V_{G,\mathcal{F}}(\boldsymbol{x}) \geq 0$, and $V_{G,\mathcal{F}}(\boldsymbol{x}) = 0$ exactly when $\boldsymbol{x}$ is a section of $\mathcal{F}$.*

The canonical example is the Laplacian quadratic form $V_{G,\mathcal{F}}(\boldsymbol{x}) = \boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}} \boldsymbol{x}$, which we will use almost exclusively, but there are many other options. For instance, any norm induces a scoring function $V_{G,\mathcal{F}}(\boldsymbol{x}) = \|\boldsymbol{\delta x}\|$. Note that many sheaf scoring functions (including the Laplacian quadratic form) can be decomposed into a sum with one term for each edge in G, but this is not required to be the case.

Definition 11 *Given a sheaf scoring function V, a margin $\gamma \geq 0$, positive and negative knowledge graphs $k, \tilde{k} :*

$G, \tilde{G} \rightarrow \mathcal{Q}$, and a set $\mathcal{A}$ of pairs $(H, \tilde{H})$ of subgraphs of $G, \tilde{G}$, a γ-gapped contrastive sheaf embedding *is a sheaf embedding of G such that for every pair of subgraphs* $(H, \tilde{H}) \in \mathcal{A}$, $V_{\tilde{H}, \mathcal{F}^{\tilde{H}}}(\boldsymbol{x}^{\tilde{H}}) - V_{H, \mathcal{F}^H}(\boldsymbol{x}^H) > \gamma$.

A common choice for the set of contrastive pairs $\mathcal{A}$ is to choose all pairs $(H, \tilde{H})$, where both graphs consist of a single edge with the same pair of incident vertices. Note that for any γ, a gapped contrastive sheaf embedding with properly chosen pairs of contrastive subgraphs still enables us to perfectly distinguish between relations in G and relations in $\tilde{G}$. However, this relaxation makes it easier to quantify an embedding's degree of consistency, and to produce tractable objective functions for training. Indeed, Definition 11 leads directly to the *margin ranking loss* for learning knowledge graph embeddings. The decomposition of the criterion over the pairs of subgraphs in $\mathcal{A}$ also provides a natural way to construct training batches (Schlichtkrull et al., 2018).

It should be noted that an embedding satisfying Definition 11 only guarantees the ability to distinguish between relations in G and relations in $\tilde{G}$ by comparison with each other. That is, given two relations, with a guarantee that one is in G and the other is in $\tilde{G}$, we can determine which is which by comparing the corresponding scores. However, given a single relation, there is no way to identify whether it lies in G or $\tilde{G}$. Further criteria could be added to address this point, but we will focus on the purely contrastive case, as it is used in most popular knowledge graph embedding approaches.

4.1.1 Translational Embeddings

One perspective on Definition 8 is that it asks for a vector $\boldsymbol{x}$ and a linear map $\boldsymbol{\delta}$ with particular structure such that $\boldsymbol{\delta x} = 0$. We may ask whether the zero vector should be special—why not also learn a vector $\boldsymbol{y}$ such that $\boldsymbol{\delta x} = \boldsymbol{y}$? This turns out to be a bit too much freedom, since for any $\boldsymbol{\delta}$ and $\boldsymbol{x}$ we could just choose $\boldsymbol{y} = \boldsymbol{\delta x}$, but this restriction makes sense if we require $\boldsymbol{y}$ to be the pullback $k^*\boldsymbol{z}$ of a 1-cochain of $\mathcal{F}$ on $\mathcal{Q}$. This amounts to requiring, for every edge $e : u \rightarrow v$ in G over a relation $r : h \rightarrow t$ in $\mathcal{Q}$, that $\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} \boldsymbol{x}_u - \boldsymbol{\mathcal{F}}_{t \trianglelefteq r} \boldsymbol{x}_v = \boldsymbol{z}_r$. We call this a *translational sheaf embedding*, as it requires the embeddings of entities to agree with each other after a translation in the edge stalk.

A consistent translational sheaf embedding exists precisely when a standard sheaf embedding exists. This can be seen by noting that if $k^*\boldsymbol{z}$ is in the image of $\boldsymbol{\delta}_{k^*\mathcal{F}}$, then we can subtract any preimage from $\boldsymbol{x}$ to get an entity embedding for which $\boldsymbol{\delta}_\mathcal{F}\boldsymbol{x} = 0$. However, once we add negative constraints on the embedding, the picture is not so simple, and a nonzero target 1-cochain may be useful. There are natural generalizations of the previous definitions to include a translational component, and some extensions are described in the appendix.

4.2 Loss Functions

We have now specified embedding spaces for entities and relations, as well as consistency conditions for embeddings. To learn an embedding, we construct a loss function defined on these embedding spaces whose minima correspond to embeddings satisfying the conditions in one of the definitions 7–11.

For instance, the Laplacian quadratic form $\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}^G} \boldsymbol{x}$ attains its minimum value precisely when $\boldsymbol{x}$ and $\mathcal{F}$ form an embedding according to definition 8. Note that since both $\boldsymbol{L}_{\mathcal{F}^G}$ and $\boldsymbol{x}$ depend on the embedding parameters, this loss function is not simply a convex quadratic function.

Knowledge graph embedding objective functions are typically thought of as being constructed from the scoring function that evaluates the quality of the embedding for a single edge of G. This is a special case of the scoring functions of Definition 10, which evaluates an embedding on an arbitrary subgraph of G. As noted above, the Laplacian quadratic form $\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}^G} \boldsymbol{x}$ is the aggregate sum over all edges of the scoring function used in the Structured Embedding model; in sheaf-theoretic notation this is

$$f^{\text{SE}}(h, r, t) = \|\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} \boldsymbol{x}_h - \boldsymbol{\mathcal{F}}_{t \trianglelefteq r} \boldsymbol{x}_t\|^2. \quad (2)$$

Following our discussion of translational embeddings in the previous section, we may define a translational scoring function similarly:

$$f^{\text{TransX}}(h, r, t) = \|\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} \boldsymbol{x}_h + \boldsymbol{r}_r - \boldsymbol{\mathcal{F}}_{t \trianglelefteq r} \boldsymbol{x}_t\|^2. \quad (3)$$

The scoring function f^{TransX} is equivalent to the TransR (Lin et al., 2015) scoring function, and when $\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} = \boldsymbol{\mathcal{F}}_{t \trianglelefteq r_r} = \boldsymbol{I}$, this scoring function is equivalent to that of TransE (Bordes et al., 2013).

The Laplacian quadratic form does not incorporate any negative information about our knowledge graph. Knowledge graph embedding techniques typically construct an objective by evaluating the chosen scoring function on contrastive pairs in such a way as to encourage the score of the true relation to be smaller, implying better consistency. The contrastive embedding definitions given above are meant to capture this notion. In particular, Definition 11 lends itself nicely to an objective function. To learn a γ-gapped sheaf embedding of our knowledge graph, we use the *margin ranking loss*:

$$\mathcal{L}_m = \sum_{(H, \tilde{H}) \in \mathcal{A}} \max(0, V_{H, \mathcal{F}^H}(\boldsymbol{x}^H) + \gamma - V_{\tilde{H}, \mathcal{F}^{\tilde{H}}}(\boldsymbol{x}^{\tilde{H}}))$$
$$(4)$$

which is equal to zero if and only if $\mathcal{F}$ and $\boldsymbol{x}$ form a γ-gapped contrastive sheaf embedding with respect to the contrastive graph pairs given in $\mathcal{A}$.

4.3 Learning Multiple Sections

It is desirable to produce knowledge graph embeddings which encode knowledge in a robust and generalized manner so that these embeddings may be applicable to downstream knowledge tasks involving unseen data. From the purview of sheaf embedding, one way to coerce these knowledge graph representations to be more general is to force the space of approximate sections of the learned knowledge sheaves to be large. In other words, we would like $\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{F}^G} \boldsymbol{x}$ to be small for as many choices of 0-cochain $\boldsymbol{x}$ as possible. Up to this point, we have defined a sheaf embedding as consisting of a single 0-cochain $\boldsymbol{x} \in C^0(G; \mathcal{F}^G)$ and, in the translational case, 1-cochain $\boldsymbol{r} \in C^1(G; \mathcal{F}^G)$ that represent the entity and relation embeddings, respectively learned from the training data. One way to improve the robustness of our sheaf embedding is to learn multiple 0-cochains simultaneously, which can be thought of as an ensemble learning approach that may help mitigate errors due to initialization, sampling, and labeling (Adlam and Pennington, 2020). Learning a set of k independent cochains is simple to implement: instead of learning a single d_v-dimensional vector $\boldsymbol{x}_v$ for each entity, we learn a $d_v \times k$ matrix $\boldsymbol{X}_v$; the loss function is updated accordingly by using the Frobenius norm on matrices. It is important to note that the relation embeddings do not increase in size, which may help avoid some types of overfitting.

4.4 Modeling Knowledge Priors

The choice of representational prior has significant ramifications for the performance of knowledge graph embedding models (Sun et al., 2019a; Zhang et al., 2019; Cai, 2019; Patel et al., 2021). In addition to generalizing a number of popular knowledge graph embedding approaches, this knowledge sheaf framework helps clarify the options for constraining knowledge graph embeddings to better capture the semantics of the underlying knowledge domain. The structure of the restriction maps $\mathcal{F}_{\bullet \trianglelefteq r}$ for each relation r provides control for modeling symmetric, asymmetric, one-to-many, many-to-one, or one-to-one relations by choosing the proper structure for the restriction maps across each edge type in $\mathcal{R}$. For example, a symmetric relationship may be enforced by requiring that $\mathcal{F}_{h \trianglelefteq r} = \mathcal{F}_{t \trianglelefteq r}$. The choice of edge stalk space $\mathcal{F}(r)$ for each relation type r provides flexibility for determining the space within which entity embeddings are compared across incident edges. For example, setting $\dim \mathcal{F}(r) < \dim \mathcal{F}(h)$ means that an entity embedding $\boldsymbol{x}_h$ can be consistently extended across r to many different embeddings $\boldsymbol{x}_t$.

The linear transformations represented in the restriction maps can also be constrained to regularize the learned embeddings: forcing $\mathcal{F}_{\bullet \trianglelefteq r}$ to be orthogonal requires entity embeddings to be comparable as a rotation across r. We experiment with these latter two parametrizations in Section 5.

Finally, when the schema $\mathcal{Q}$ has multiple entity types, the embedding stalk space can vary across these types, decreasing parameterization for types which can be modeled using few dimensions.

4.5 Inference with Sheaf Embeddings

The standard knowledge graph completion task involves finding pairs of entities linked by a given relationship which is not already encoded in the knowledge graph. The standard approach to this task involves ranking potential relations using the scoring function defined by the embedding model, and this approach applies equally well to sheaf embeddings using a sheaf scoring function. The advantage of reframing knowledge graph embedding as learning a knowledge sheaf is that one can exploit the spectral properties of cellular sheaves to naturally extend these embeddings to answer more complex queries.

4.5.1 Multi-Hop Reasoning

Complex relationships may be constructed by composing the basic relations of a knowledge graph. Often the resulting relationship is one of the basic relation types represented in the knowledge graph, but it is also possible to construct more complex relations by composition. For instance, the relations "x is a child of y" and "y is a child of z" compose to "x is a grandchild of z." The term "multi-hop reasoning" is often used for deductions of this sort (Guu et al., 2015; Gardner et al., 2014; Toutanova et al., 2016). The sheaf Laplacian provides a natural tool for addressing these composite queries. We describe here the construction for non-translational sheaf embeddings; the extension to translational embeddings is detailed in the appendix.

If we wish to infer the possible endpoint of a sequence of relations $r_1; r_2; \cdots; r_k$, beginning at known entity u_0 and ending at some to-be-determined entity u_k, we can construct a chain of edges with these relations, and optimize for their combined discrepancy. That is, we consider a new knowledge graph H modeled on $\mathcal{Q}$ with vertex set $v_0, \ldots, v_k$. The knowledge sheaf $\mathcal{F}$ also induces a sheaf $\mathcal{F}^H$ on H as before. If we match each vertex of H with a vertex of G, (in a way consistent with the schema), the entity embeddings for G give us a 0-cochain of $\mathcal{F}^H$. We want to find the matching that makes this 0-cochain as close to a section as possible. The corresponding optimization problem can be expressed as

$$\underset{u_1, \ldots, u_k \in \mathcal{E}}{\arg \min} \sum_{i=1}^{k} \|\mathcal{F}^H_{v_{i-1} \trianglelefteq e_i} \boldsymbol{x}_{u_{i-1}} - \mathcal{F}^H_{v_i \trianglelefteq e_i} \boldsymbol{x}_{u_i}\|^2. \quad (5)$$

Naively, finding the best fit for a chain of length k requires evaluating the objective function at $|\mathcal{E}|^k$ tuples of entities. Other approaches to this problem try to find approximate solutions, e.g. by simply greedily extending to the best entity

at each step or ignoring the interior nodes altogether and constructing some joint composite relation, thus simplifying to single-hop graph completion (Lin et al., 2018; Guu et al., 2015).

We offer a new approach based on a related optimization problem. Rather than choosing one entity from the knowledge database for each intervening node $u_1, \ldots, u_{k-1}$, we optimize directly over the entity embedding space, with the intermediate cost function

$$V(\boldsymbol{y}) = \sum_{i=1}^{k} \|\boldsymbol{\mathcal{F}}^H_{v_{i-1} \trianglelefteq e_i} \boldsymbol{y}_{i-1} - \boldsymbol{\mathcal{F}}^H_{v_i \trianglelefteq e_i} \boldsymbol{y}_i\|^2 = \boldsymbol{y}^T \boldsymbol{L}_{\mathcal{F}^H} \boldsymbol{y}. \tag{6}$$

This is a relaxation of (5), as $\boldsymbol{y}_i$ need not be the embedding of a known entity. The relaxed problem of finding the best-fitting tail entity u_k for the composite relation is then

$$\underset{u_k \in \mathcal{E}}{\arg\min} \left(\min_{\boldsymbol{y} \in C^0(H; \mathcal{F}^H)} V(\boldsymbol{y}) \text{ s.t. } \boldsymbol{y}_0 = \boldsymbol{x}_{u_0}, \ \boldsymbol{y}_k = \boldsymbol{x}_{u_k} \right). \tag{7}$$

The inner optimization problem, depending on $\boldsymbol{x}_{u_0}$ and $\boldsymbol{x}_{u_k}$, is the problem of *harmonic extension* of a 0-cochain defined on a boundary subset of vertices B, which here is $\{v_0, v_k\}$. This problem is convex and quadratic, so the optimal value is unique, but the optimizer may not be. A brief argument using Lagrange multipliers shows that an equivalent problem is to find a 0-cochain $\boldsymbol{y} \in C^0(H; \mathcal{F}^H)$ such that $\boldsymbol{y}_0 = \boldsymbol{x}_{u_0}$, $\boldsymbol{y}_k = \boldsymbol{x}_{u_k}$, and $\boldsymbol{L}_{\mathcal{F}^H} \boldsymbol{y} = \boldsymbol{0}$ on nodes not in B. When there is a unique solution, its values on U, the complement of B, are given by the formula $\boldsymbol{y}_U = -\boldsymbol{L}[U, U]^{-1} \boldsymbol{L}[U, B]\boldsymbol{y}_B$, where $\boldsymbol{y}_B$ is determined by the embeddings $\boldsymbol{x}_{u_0}$ and $\boldsymbol{x}_{u_k}$, and we drop the subscript on $\boldsymbol{L}_{\mathcal{F}^H}$. Then the minimum value of the inner optimization problem in (7) is

$$V(\boldsymbol{y}^*) = \boldsymbol{y}_B^T \left(\boldsymbol{L}[B, B] - \boldsymbol{L}[B, U]\boldsymbol{L}[U, U]^{-1}\boldsymbol{L}[U, B] \right) \boldsymbol{y}_B.$$

The matrix in this formula is the *Schur complement* $\boldsymbol{L}/\boldsymbol{L}[U, U]$ of $\boldsymbol{L}[U, U]$ in $\boldsymbol{L}$.[2] We can think of it as defining a scoring function for the composite relation. In fact, by factorizing $\boldsymbol{L}/\boldsymbol{L}[U, U]$, it is possible to extract from this a pair of matrices that can be thought of as an induced embedding for the multi-hop relation.

4.5.2 Complex Composite Relations

We need not limit ourselves to composing relations in linear chains: harmonic extension adapts effortlessly to more complex networks of relations like those displayed in Figure 1. Let H be any knowledge graph with schema $\mathcal{Q}$. The learned knowledge sheaf extends to H as before, and its sections over H correspond to collections of entity embeddings jointly satisfying the relations. We construct a

---

[2]When $\boldsymbol{L}[U, U]$ is not invertible, we may use its Moore–Penrose pseudoinverse $\boldsymbol{L}[U, U]^\dagger$.
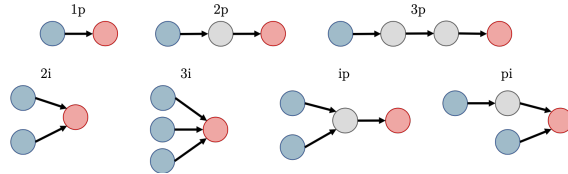

Figure 1: Illustration of complex query structures. Unknown entities are colored gray. Source entities are colored blue and target entities are colored red.

boundary set of vertices B given by the entities of interest and denote its (possibly empty) complement by U, obtaining the Schur complement $\boldsymbol{L}_{\mathcal{F}^H}/\boldsymbol{L}_{\mathcal{F}^H}[U, U]$. The quadratic form $V(\boldsymbol{y}_B) = \boldsymbol{y}_B^T(\boldsymbol{L}_{\mathcal{F}^H}/\boldsymbol{L}_{\mathcal{F}^H}[U, U])\boldsymbol{y}_B$ finds the minimal value of a problem analogous to the inner problem in (7), constraining the values of $\boldsymbol{y}$ on B to equal $\boldsymbol{y}_B$. We can then fix the values of $\boldsymbol{y}_B$ on some source subset of vertices S to be equal to the embeddings of some given entities $\{u_s\}_{s \in S}$, and test the embeddings $\boldsymbol{x}_t$ for other entities t to find the entities that minimize $V(\boldsymbol{y}_B)$ subject to $\boldsymbol{y}_S = \boldsymbol{x}_S$.

For further insight regarding harmonic extension as a method for solving complex queries, it is helpful to note the relationship between the Schur complement and marginalization when entity embeddings are distributed as multivariate Gaussians (Von Mises, 2014), as detailed in the appendix. Further, recall that sheaf embedding generalizes Structured Embedding, and with the addition of non-trivial 1-cochains (3) represents a generalization of TransR. Harmonic extension provides a way to apply any of these models to multi-hop and complex composite queries in a theoretically justified manner which, to the authors' knowledge, is a first for models like Structured Embedding or TransR that are not purely translational or bilinear (Guu et al., 2015).

5 EXPERIMENTS

To validate the approach to answering complex queries detailed in the previous section and to highlight the flexibility of sheaf embedding, we compare the performance of a number of sheaf embedding models on two benchmark datasets: NELL-995 (Xiong et al., 2017) and FB15k-237 (Toutanova et al., 2015). These experiments are not intended to achieve state-of-the-art performance. Rather, our aim is to investigate the effect of model regularization choices detailed in Section 4 and to showcase the extensibility of this framework to the evaluation of complex queries. We implement these models in the open-source knowledge graph embedding package Pykeen (Ali et al., 2021), allowing us to train and evaluate these models in a manner that is both reproducible and comparable to other embedding techniques. Models are trained and tested on an Nvidia GeForce GTX 1080 GPU with 8GB RAM.

We train each model according to the traditional graph com-

pletion task, learning embeddings by minimizing the loss associated to scoring both true triplets and corrupted triplets (Equation 4). At test time, we evaluate each model on the dataset's held-out test set along with the 1p, 2p, 3p, 2i, 3i, ip, and pi complex query structures displayed in Figure 1 and originally detailed in Ren and Leskovec (2020). Performance is evaluated on both an "easy" and "hard" subset of complex query answers. The "easy" subset contains queries whose full structure is encountered within the training set whereas the "hard" subset requires at least one fact to be inferred that was not present in the training set. The query structures themselves correspond to logical multi-hop path query structures (*p), intersectional queries (*i) and a combination of the two (ip and pi). Entities are scored based on their solution to the associated optimization problem (7). For each complex query type, model performance is evaluated according to two canonical measures: the mean reciprocal ranking (MRR) and the hits at 10 (H@10), both of which measure the relative ranking of the true solution to a query among the ordered score of all entities. Note that the models are only trained on the traditional triplet scoring task and not on the complex queries themselves.

We train four types of models. The `ExtensionSE` model implements scoring function (2) which is a generalization of Structured Embedding (Bordes et al., 2011). To observe the effects of compressive comparison across relations, we vary the edge stalk space $\dim \mathcal{F}(r)$ and assume all edge types share this space for this model. We also train an `ExtensionSE_orthogonal` model which applies an orthogonal parameterization to all restriction maps of the model. The `ExtensionTranslational` model implements scoring function (3) while the `ExtensionTransE` model implements the same scoring function but sets $\mathcal{F}_{h \trianglelefteq r} = \mathcal{F}_{t \trianglelefteq r} = I$ which is equivalent to TransE (Bordes et al., 2013). We use the Euclidean norm for both scoring functions. We set the margin $\gamma = 1$ and train each model for 250 epochs without any hyperparameter tuning.

We also experiment with varying the entity embedding dimension $\dim \mathcal{F}(s)$ across dimensions $[8, 16, 32, 64]$ with dimensionality shared across entities. We use the same embedding dimensions for $\dim \mathcal{F}(r)$ and train models with all combinations where $\dim \mathcal{F}(r) <= \dim \mathcal{F}(s)$. We assume a single entity type for the schema of both datasets. Code for these experiments is available at `https://github.com/tgebhart/sheaf_kg`.

Figure 2 depicts a subset of these experimental results for NELL-995 (results for FB15k-237 in appendix) with entity embedding dimension fixed at 32 with varying number of (unregularized) sections and $\dim \mathcal{F}(r)$. Figure 2 indicates that square restriction maps ($\dim \mathcal{F}(r) = \dim \mathcal{F}(s)$) generally perform better than restriction maps which compress information across relations. The variants of the generalized Structured Embedding models generally achieve the highest performance on the complex query tasks, which is sur-

prising given the competitive performance of translational models on the traditional completion tasks (test and 1p) within the broader literature. The higher performance of the `ExtensionSE_orthogonal` model on path-structured queries compared to its performance on the test and 1p tasks highlights the differences in representational requirements for answering complex queries versus simple triplet queries, an observation reinforced by other works investigating compositional relational embeddings (Guu et al., 2015; Sun et al., 2019a; Tang et al., 2020; Cai, 2019).

It is clear from Figure 2 that increasing the number of sections learned for each entity embedding improves overall performance across all model types without orthogonal parameterization of restriction maps. This result is not surprising, as increasing the number of sections increases model capacity. This interpretation is reinforced by Figure 6 (appendix) which shows that increasing the entity embedding dimension $\dim \mathcal{F}(s)$ (an alternative route for increasing model capacity) also leads to increased performance.

We also compare the performance of the harmonic extension approach to a naive method for answering complex queries within the TransE model. This approach, as detailed in Guu et al. (2015), amounts to summation across all entity and relation embeddings involved in the complex query. Table 1 displays these results for models with embedding dimension 32 and 1 section. The similar performance between the two models on the path and intersectional queries is expected, as the harmonic extension over the identity restriction maps of the `ExtensionTransE` model is very similar to the naive method for these simpler queries. However, on the more complex pi and ip queries, the harmonic extension approach significantly outperforms the naive approach.

dataset	model	2p	3p	2i	3i	pi	ip
NELL-995	ExtensionTransE	**0.237**	**0.120**	0.280	0.198	**0.340**	**0.296**
	NaiveTransE	0.164	0.082	**0.281**	**0.205**	0.271	0.104
FB15k-237	ExtensionTransE	**0.084**	0.050	0.157	0.147	**0.119**	**0.109**
	NaiveTransE	0.079	**0.052**	**0.163**	**0.167**	0.107	0.049

dataset	model	2p	3p	2i	3i	pi	ip
NELL-995	ExtensionTransE	**0.060**	**0.054**	**0.165**	0.234	**0.126**	**0.079**
	NaiveTransE	0.055	0.042	0.158	**0.238**	0.105	0.049
FB15k-237	ExtensionTransE	0.022	**0.015**	**0.100**	**0.154**	**0.074**	**0.044**
	NaiveTransE	**0.025**	0.013	0.093	0.153	0.064	0.024

Table 1: Comparative MRR performance of the extension and naive TransE approaches for "easy" (top) and "hard" (bottom) complex query answers on NELL-995 and FB15k-237 datasets.

6 DISCUSSION

The sheaf-theoretic generalization of knowledge graph embedding presented in this paper provides a new perspective for integrating a number of prior embedding approaches within a cohesive theoretical framework. In addition to facilitating the comparison of these prior approaches within a
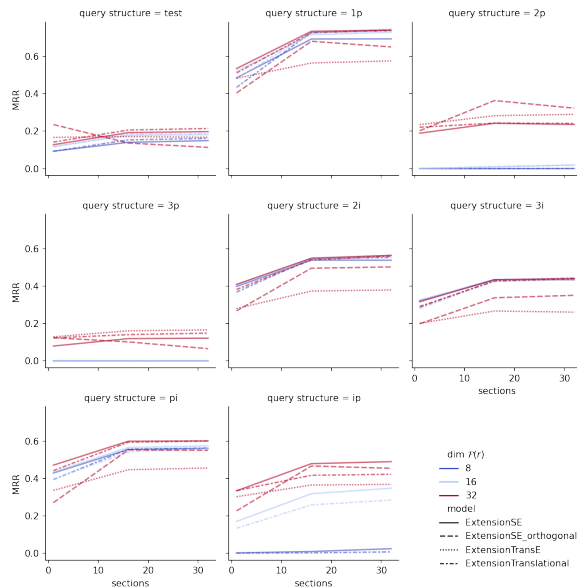
Figure 2: Triplet and complex query completion performance across varying number of sections and edge stalk dimensionality for each model class on NELL-995. Each sub-figure displays results for each complex query type and the test dataset. Line styles correspond to different model types and colors denote the dimensionality of the relation stalk space $\dim \mathcal{F}(r)$. The entity embedding dimension $\dim \mathcal{F}(s) = 32$ for each model.

common language, this generalization permits the incorporation of structured priors and expands the class of queries that can be answered using an embedding. By formalizing the relationship between typed knowledge graph schemas and their instantiation in knowledge graphs, this sheaf-theoretic framework provides a natural roadmap for the modeling of typed, hierarchical knowledge bases which provides further control over the representation of priors placed on the embeddings. By viewing knowledge graph embedding as sheaf learning, we have immediate access to tools for reasoning about the local and global consistency of embeddings through the sheaf Laplacian. Perhaps most importantly, this generalized perspective of knowledge graph embedding opens the door to an array of extensions like introducing uncertainty with a probabilistic approach, enforcing hierarchical typing within embeddings, or embedding knowledge graphs within more exotic categories.

This work is a preliminary exploration of the possibilities for using topology, sheaf theory, and spectral graph theory to understand knowledge graph embeddings. We anticipate that future work will deepen the theoretical foundations of this framework by exploring the representational capacity of sheaf embeddings with particular structures, and also refine the implementation and application to real datasets.

References

B. Adlam and J. Pennington. Understanding double descent requires a fine-grained bias-variance decomposition. *Advances in neural information processing systems*, 33: 11022–11032, 2020.

M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, M. Galkin, S. Sharifzadeh, A. Fischer, V. Tresp, and J. Lehmann. Bringing light into the dark: A large-scale evaluation of knowledge graph embedding models under a unified framework. *arXiv preprint arXiv:2006.13365*, 2020.

M. Ali, M. Berrendorf, C. T. Hoyt, L. Vermue, S. Sharifzadeh, V. Tresp, and J. Lehmann. Pykeen 1.0: A python library for training and evaluating knowledge graph embeddings. *Journal of Machine Learning Research*, 22 (82):1–6, 2021.

S. Auer, V. Kovtun, M. Prinz, A. Kasprzik, M. Stocker, and M. E. Vidal. Towards a knowledge graph for science. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, pages 1–6, 2018.

A. Bordes, J. Weston, R. Collobert, and Y. Bengio. Learning structured embeddings of knowledge bases. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2011.

A. Bordes, N. Usunier, A. Garcia-Duran, J. Weston, and O. Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

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

C. Cai. Group representation theory for knowledge graph embedding. *arXiv preprint arXiv:1909.05100*, 2019.

X. Chen, S. Jia, and Y. Xiang. A review: Knowledge reasoning over knowledge graph. *Expert Systems with Applications*, 141:112948, 2020.

J. Curry. *Sheaves, Cosheaves, and Applications*. PhD thesis, University of Pennsylvania, 2014.

T. Ebisu and R. Ichise. TorusE: Knowledge graph embedding on a Lie group. *arXiv preprint arXiv:1711.05435*, 2017.

B. Fong and D. I. Spivak. Seven sketches in compositionality: An invitation to applied category theory. *arXiv:1803.05316 [math]*, 2018. URL http://arxiv.org/abs/1803.05316.

M. Gardner, P. Talukdar, J. Krishnamurthy, and T. Mitchell. Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 397–406, 2014.

Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He. A survey on knowledge graph-based recommender

systems. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

K. Guu, J. Miller, and P. Liang. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327, 2015.

W. Hamilton, P. Bajaj, M. Zitnik, D. Jurafsky, and J. Leskovec. Embedding logical queries on knowledge graphs. *Advances in neural information processing systems*, 31, 2018.

J. Hansen and R. Ghrist. Toward a spectral theory of cellular sheaves. *Journal of Applied and Computational Topology*, 3(4):315–358, Dec. 2019. ISSN 2367-1734.

K. Hayashi and M. Shimbo. On the equivalence of holographic and complex embeddings for link prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 554–559, 2017.

C. T. Hoyt, M. Balk, T. J. Callahan, D. Domingo-Fernandez, M. A. Haendel, H. B. Hegde, D. S. Himmelstein, K. Karis, J. Kunze, T. Lubiana, N. Matentzoglu, J. McMurry, S. Moxon, C. J. Mungall, A. Rutz, D. R. Unni, E. Willighagen, D. Winston, and B. M. Gyori. The bioregistry: Unifying the identification of biomedical entities th rough an integrative, open, community-driven metaregistry. *bioRxiv*, 2022. doi: 10.1101/2022.07.08.499378.

N. Jain, J.-C. Kalo, W.-T. Balke, and R. Krestel. Do embeddings actually capture knowledge graph semantics? In *Eighteenth Extended Semantic Web Conference - Research Track*, 2021. URL https://openreview.net/forum?id=vsxYOZoPvne.

R. Jenatton, N. L. Roux, A. Bordes, and G. R. Obozinski. A latent factor model for highly multi-relational data. In *Advances in neural information processing systems*, pages 3167–3175, 2012.

G. Ji, K. Liu, S. He, and J. Zhao. Knowledge graph completion with adaptive sparse transfer matrix. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu. A survey on knowledge graphs: Representation, acquisition and applications. *arXiv preprint arXiv:2002.00388*, 2020.

H. Lin, Y. Liu, W. Wang, Y. Yue, and Z. Lin. Learning entity and relation embeddings for knowledge resolution. *Procedia Computer Science*, 108:345–354, 2017.

X. V. Lin, R. Socher, and C. Xiong. Multi-hop knowledge graph reasoning with reward shaping. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3243–3253, 2018.

Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

D. M. Malioutov, J. K. Johnson, and A. S. Willsky. Walk-sums and belief propagation in gaussian graphical models. *The Journal of Machine Learning Research*, 7:2031–2064, 2006.

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

M. Nickel, L. Rosasco, and T. Poggio. Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 1955–1961, 2016.

D. Patel, P. Dangati, J.-Y. Lee, M. Boratko, and A. McCallum. Modeling label space interactions in multi-label classification using box embeddings. In *International Conference on Learning Representations*, 2021.

H. Ren and J. Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. *Advances in Neural Information Processing Systems*, 33, 2020.

H. Ren, W. Hu, and J. Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *International Conference on Learning Representations (ICLR)*, 2020.

E. Riehl. *Category theory in context*. Aurora: Dover Modern Math Originals. Dover, Minola, NY, 2017.

A. Santos, A. R. Colaço, A. B. Nielsen, L. Niu, M. Strauss, P. E. Geyer, F. Coscia, N. J. W. Albrechtsen, F. Mundt, L. J. Jensen, et al. A knowledge graph to interpret clinical proteomics data. *Nature biotechnology*, 40(5):692–702, 2022.

M. Schlichtkrull, T. N. Kipf, P. Bloem, R. v. d. Berg, I. Titov, and M. Welling. Modeling relational data with graph convolutional networks. In *European semantic web conference*, pages 593–607. Springer, 2018.

A. Sinha, Z. Shen, Y. Song, H. Ma, D. Eide, B.-J. Hsu, and K. Wang. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th international conference on world wide web*, pages 243–246, 2015.

R. Socher, D. Chen, C. D. Manning, and A. Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

Z. Sun, Z.-H. Deng, J.-Y. Nie, and J. Tang. RotatE: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*, 2019a.

Z. Sun, Q. Guo, J. Yang, H. Fang, G. Guo, J. Zhang, and R. Burke. Research commentary on recommendations with side information: A survey and research directions. *Electronic Commerce Research and Applications*, 37: 100879, 2019b.

Y. Tang, J. Huang, G. Wang, X. He, and B. Zhou. Orthogonal relation transforms with graph context modeling for

knowledge graph embedding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2713–2722, 2020.

K. Toutanova, D. Chen, P. Pantel, H. Poon, P. Choudhury, and M. Gamon. Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1174. URL `https://www.aclweb.org/anthology/D15-1174`.

K. Toutanova, X. V. Lin, W.-t. Yih, H. Poon, and C. Quirk. Compositional learning of embeddings for relation paths in knowledge base and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1434–1444, 2016.

T. Trouillon and M. Nickel. Complex and holographic embeddings of knowledge graphs: a comparison. *arXiv preprint arXiv:1707.01475*, 2017.

T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard. Complex embeddings for simple link prediction. In *International Conference on Machine Learning (ICML)*, 2016.

R. Von Mises. *Mathematical theory of probability and statistics*. Academic Press, Cambridge, MA, 2014.

Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *Aaai*, pages 1112–1119. Citeseer, 2014.

W. Xiong, T. Hoang, and W. Y. Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573, 2017.

T. Yang, L. Sha, and P. Hong. Nage: Non-abelian group embedding for knowledge graphs. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1735–1742, 2020.

S. Zhang, Y. Tay, L. Yao, and Q. Liu. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems*, pages 2735–2745, 2019.

A OTHER EMBEDDING MODELS

Many knowledge graph embedding approaches may be decomposed into a combination of multiplicative and additive interactions of entity and relation vectors. We show in this section that these additive components correspond to sheaf-theoretic coboundary operators across incident edges and discuss how different embedding approaches alter this coboundary operator. The models discussed in this section are a non-exhaustive subset of the total number of models in existence, but we make an attempt to cover most of the popular choices.

Structured Embedding. One of the earliest approaches to embedding knowledge graphs is Structured Embedding (SE) Bordes et al. (2011). Structured Embedding models entities $\boldsymbol{x}_h \in \mathbb{R}^d$ as d-dimensional vectors and relations as a pair of $(d \times d)$-dimensional matrices $(\boldsymbol{R}_r^h, \boldsymbol{R}_r^t)$. The scoring function between entities is then $f^{\text{SE}}(h, r, t) = \|\boldsymbol{R}_r^h \boldsymbol{x}_h - \boldsymbol{R}_r^t \boldsymbol{x}_t\|$. Setting $\boldsymbol{R}_r^h = \boldsymbol{\mathcal{F}}_{h \trianglelefteq r}$ and $\boldsymbol{R}_r^t = \boldsymbol{\mathcal{F}}_{t \trianglelefteq r}$, we see the scoring function computes precisely boundary function of the sheaf $f^{\text{SE}}(h, r, t) = \|\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} \boldsymbol{x}_h - \boldsymbol{\mathcal{F}}_{t \trianglelefteq r} \boldsymbol{x}_t\|$. In other words, SE attempts to learn entity and relation embeddings that minimize the local discrepancy between adjacent entities along each relation. Therefore,

$$\sum_{(h,r,t)} f^{\text{SE}}(h, r, t)^2 = \boldsymbol{x}^T \boldsymbol{L} \boldsymbol{x}$$

where $\boldsymbol{L}$ is the sheaf Laplacian formed from the matrices $\boldsymbol{R}_r^\bullet$, and $\boldsymbol{x} = (\boldsymbol{x})_{v \in V} \in C^0(G; \mathcal{F}^G)$.

Unstructured Model. The unstructured model (Bordes et al., 2014), often used as a baseline model, is equivalent to Structured Embedding when $\boldsymbol{R}_r^h = \boldsymbol{R}_r^t = \boldsymbol{I}$, and therefore also fits within our modeling framework.

TransX. A number of related embedding methods have been developed which seek to model relations as translations in a vector space which we refer to as the TransX class of embedding methods. These models seek to find embeddings of triples $(\boldsymbol{x}_h, \boldsymbol{r}_r, \boldsymbol{x}_t)$ such that $g(\boldsymbol{x}_h, \boldsymbol{r}_r) \approx \boldsymbol{x}_t$ where g is a simple function representing a translation-like operation within the chosen embedding space.

As discussed in the main text, TransE (Bordes et al., 2013) is an early translation-based model which aims to find embeddings that result in

$$f^{\text{TransE}}(h, r, t) = \|\boldsymbol{x}_h + \boldsymbol{r}_r - \boldsymbol{x}_t\|^2 \tag{8}$$

being small when (h, r, t) is true and large otherwise. Here, both the entity and relation embeddings are vectors in $\mathbb{R}^d$.

We can formulate this kind of translational scoring within our sheaf-theoretic framework by viewing the relation vector as a $\boldsymbol{r}_r$ as a 1-cochain across edge r. More formally, we wish to learn some 1-cochain $\boldsymbol{r} \in C^1(G; \mathcal{F}^G)$, representing a choice of vectors over each relation type in the knowledge graph, such that the discrepancy of entity embeddings $\boldsymbol{x}_h$ and $\boldsymbol{x}_t$ across each relation r is approximately equal to $\boldsymbol{r}_r$:

$$f^{\text{TransX}}(h, r, t) = \|\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} \boldsymbol{x}_h + \boldsymbol{r}_r - \boldsymbol{\mathcal{F}}_{t \trianglelefteq r} \boldsymbol{x}_t\|^2. \tag{9}$$

This is equivalent in form to TransR (Lin et al., 2015) when both restriction maps are equivalent at the head and tail of r. Taking $\boldsymbol{\mathcal{F}}_{h \trianglelefteq r} = \boldsymbol{\mathcal{F}}_{t \trianglelefteq r} = \boldsymbol{I}$, our scoring function simplifies to exactly Equation 8 and is thus equivalent to TransE embedding.

TorusE and RotatE. More recently, RotatE (Sun et al., 2019a) was introduced as a hybrid between ComplEx and the TransX approach. RotatE computes embeddings $\boldsymbol{x}_h, \boldsymbol{x}_t, \boldsymbol{r}_r \in \mathbb{C}^d$ and scores triplets translationally:

$$f^{\text{RotatE}}(\boldsymbol{x}_h, \boldsymbol{r}_r, \boldsymbol{x}_t) = \|\boldsymbol{x}_h \circ \boldsymbol{r}_r - \boldsymbol{x}_t\| \tag{10}$$

where $\circ$ is the Hadamard product. We can encode this scoring function through restriction maps as follows. Taking our edge and node stalk spaces to be in $\mathbb{C}^d$, setting $\boldsymbol{\mathcal{F}}_{h \trianglelefteq r}$ to be the diagonal matrix with $\boldsymbol{r}_r$ on the diagonal[3], and setting $\boldsymbol{\mathcal{F}}_{t \trianglelefteq r} = \boldsymbol{I}$, we obtain an equivalent score for f^{RotatE}. The TorusE model (Ebisu and Ichise, 2017) is a special case of RotatE where the modulus of the embeddings are fixed.

Finally, Yang et al. (Yang et al., 2020) propose a number of embedding methods which fit within this sheaf embedding framework as fixed restriction maps which introduce both inductive priors on the interactions between, and therefore the embeddings of, entities.

---

[3]Equivalently, we can represent $\boldsymbol{r}_r$ as a diagonal matrix with $e^{i\boldsymbol{\phi}_r}$ on the diagonal where $\boldsymbol{\phi}_r$ is a vector of phases ranging from 0 to 2π.
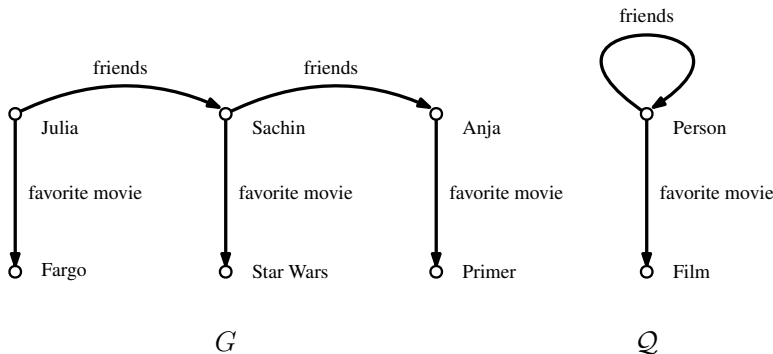
Figure 3: A simple knowledge graph G with its schema $\mathcal{Q}$.

A.1 Models Without Sheaf Structure

The sheaf-theoretic framework presented in the main text does not generalize all knowledge graph embedding approaches that have been proposed in the literature. In general, any model with a bilinear interaction between entity embeddings and relations does not have a simple representation in terms of cellular sheaves. Models of this form include the Neural Tensor Network (Socher et al., 2013), ComplEx (equivalently, HolE) (Trouillon et al., 2016; Nickel et al., 2016; Hayashi and Shimbo, 2017), Rescal/Bilinear (Jenatton et al., 2012; Nickel et al., 2011), and QuatE (Zhang et al., 2019). TransH (Wang et al., 2014) also does not conform to our modeling framework, but does conform once entities are in their post-projection form. Investigating the extent to which these bilinear models may be incorporated into the sheaf embedding form is an interesting avenue for future work.

Recently, a distinct lineage of knowledge graph embedding models have been proposed which represent a departure from translational/bilinear classification given above. Targeting the task of complex logical query answering, models like BetaE (Ren and Leskovec, 2020) and Query2Box (Ren et al., 2020) look to embed the queries themselves within some representational space. It is currently unclear to the authors whether this family of models is amenable to a sheaf-theoretic representation. Casting these models within our sheaf embedding framework may require introducing sheaves valued in other categories than vector space stalks with linear restriction maps. The basics of such a generalization are discussed in the following section.

B WORKED EXAMPLES

B.1 Knowledge Graph

To clarify definitions 1 and 2, we present a simple example of a knowledge graph G and schema $\mathcal{Q}$. Here the schema has two types: Person and Film, and two relations: "friends" and "favorite movie." This is represented as a graph with two vertices and two edges, one of which is a self-loop for the vertex Person. The graph G has three entities of type Person and three entities of type Film, with the relations indicated by edges. The graph morphism k sends each person in G to the vertex Person in $\mathcal{Q}$, and each film in G to the vertex Film in $\mathcal{Q}$. It also sends the edges in G to their similarly labeled edges in $\mathcal{Q}$.

It may be helpful to note that any subgraph of G is a valid knowledge graph for schema $\mathcal{Q}$. However, to add a node or edge to G, we must also specify which node or edge of $\mathcal{Q}$ it corresponds to. If we were to add an edge between Primer and Fargo in G, we would no longer be able to interpret G as a knowledge graph with schema $\mathcal{Q}$, because there is no corresponding edge in $\mathcal{Q}$ from Film to itself.

B.2 Knowledge Sheaf

Continuing this example, we can describe a knowledge sheaf $\mathcal{F}$ on $\mathcal{Q}$ and its pullback to G. For notational convenience, denote the node Person in $\mathcal{Q}$ by u and the node Film by v, with the edge "favorite movie" denoted e and the edge "friends" denoted f. Let $\mathcal{F}(u) = \mathbb{R}^3$ and $\mathcal{F}(v) = \mathbb{R}^2$, with $\mathcal{F}(e) = \mathbb{R}^2$ and $\mathcal{F}(f) = \mathbb{R}$. We can now define the restriction maps of $\mathcal{F}$
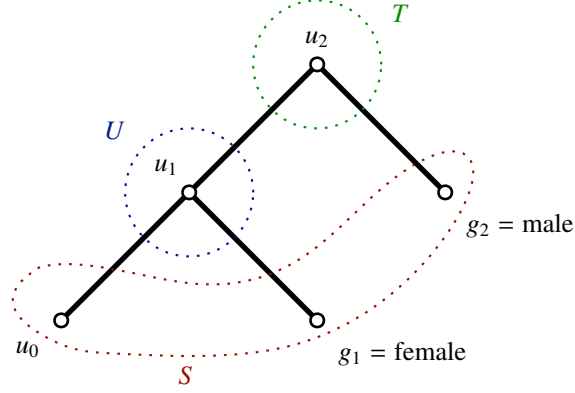
Figure 4: The template knowledge graph for finding the maternal grandfather u_2 of entity u_0. The interior U, source set S, and target set T are labeled.

using matrices:

$$\mathcal{F}_{u \lhd e} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \mathcal{F}_{v \lhd e} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$
$$\mathcal{F}_{u \lhd_h f} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \qquad \mathcal{F}_{u \lhd_t f} = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}$$

Note that because the edge "friends" is a self-loop, we need to distinguish between the connection from u to the head of f ($u \lhd_h f$) and the connection from u to the tail of f ($u \lhd_t f$), as these may have two different restriction maps in general.

The pullback $\mathcal{F}^G$ of $\mathcal{F}$ to the knowledge graph G would be unenlightening to describe completely, but we will give a few examples. For instance, because Julia is the head entity for an edge (call it a) of G that maps to "friends" in $\mathcal{Q}$, $\mathcal{F}^G_{\text{Julia} \lhd a} = \mathcal{F}_{u \lhd_h f}$, and since Sachin is the tail entity for that edge, we get $\mathcal{F}^G_{\text{Sachin} \lhd a} = \mathcal{F}_{u \lhd_t f}$. Since Anja is the head entity for an edge b that maps to "favorite movie" in $\mathcal{Q}$, we have $\mathcal{F}^G_{\text{Anja} \lhd b} = \mathcal{F}_{u \lhd e}$, and since Primer is the tail entity for b, we have $\mathcal{F}^G_{\text{Primer} \lhd b} = \mathcal{F}_{v \lhd e}$.

Choose the embeddings as follows:

$$x_{\text{Julia}} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \qquad x_{\text{Sachin}} = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix} \qquad x_{\text{Anja}} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$
$$x_{\text{Fargo}} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \qquad x_{\text{Star Wars}} = \begin{bmatrix} 0 \\ -1 \end{bmatrix} \qquad x_{\text{Primer}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

The reader may check that this collection of data in fact defines a consistent sheaf embedding (Definition 8) of G. For instance, x is consistent over the edge between Anja and Primer, because $\mathcal{F}_{u \lhd e} x_{\text{Anja}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} = \mathcal{F}_{v \lhd e} x_{\text{Primer}}$.

Symmetry of the relation "friends" is represented by the fact that $\mathcal{F}_{u \lhd_h f} = \mathcal{F}_{u \lhd_t f}$. Perhaps desirably in this instance, we see that this also forces the representation of "friends" to be transitive. The fact that each person can have only one favorite movie is represented by the fact that $\mathcal{F}_{v \lhd e}$ is a one-to-one linear map, so that for any possible Person embedding x_u, there is at most one Film embedding x_v such that $\mathcal{F}_{v \lhd e} x_v = \mathcal{F}_{u \lhd e} x_u$.

B.3 Complex Query

Consider the problem of finding the maternal grandfather of a person in a knowledge database, from constituent relations "is a child of" and "has gender." That is, u_0 is the person whose maternal grandfather we wish to find, and we seek entities u_1 and u_2 satisfying the following relations: u_0 is a child of u_1; u_1 is a child of u_2; u_1 has gender female; u_2 has gender male.

There are five entities in this knowledge graph: the known source vertex u_0 of the desired relation, the unknowns u_1 and u_2, and the entities female and male. The boundary set B consists of all vertices but u_1, and the source subset is $S = \{u_0, \text{female}, \text{male}\}$, while the target subset is simply $T = \{u_2\}$, as shown in Figure 4. To find the maternal grandfather, we construct the sheaf on the relational graph H, find its Laplacian, and compute the Schur complement $\boldsymbol{L}_{\mathcal{F}H}/\boldsymbol{L}_{\mathcal{F}H}[U, U]$.

Then we fix $\boldsymbol{y}_{u_0} = \boldsymbol{x}_{u_0}$, $\boldsymbol{y}_{g_1} = \boldsymbol{x}_{\text{female}}$ and $\boldsymbol{y}_{g_1} = \boldsymbol{x}_{\text{male}}$ to be the known embeddings of these entities, and search the entities for the entity u_2 whose embedding $\boldsymbol{x}_{u_2}$ gives the smallest value of $V(\boldsymbol{y}_B)$ when $\boldsymbol{y}_{u_2} = \boldsymbol{x}_{u_2}$. Note that by changing the values of $\boldsymbol{y}$ on the input set S, the computed Schur complement can also be used to find grandparents of any type for any initial entity u_0. We can thus think of the reduced matrix as describing a polyadic relation $R(u_0, u_2, g_1, g_2)$ which holds when u_0 is the child of someone of gender g_1 who is the child of u_2, who has gender g_2.

C HARMONIC EXTENSION FOR TRANSLATIONAL EMBEDDINGS

The problem of finding a harmonic extension in the affine or translational setting may be formulated as follows. Let H be a graph, $\mathcal{F}$ a sheaf on H, and B a subset of vertices of H with complement U. We further assume that the translations are given by a 1-cochain $\boldsymbol{b} \in C^1(H; \mathcal{F})$, and we have a known boundary condition $\boldsymbol{x}_B$ defined on vertices in B. Harmonic extension is then the following optimization problem:

$$\min_{\boldsymbol{y} \in C^0(H;\mathcal{F})} \|\boldsymbol{\delta}\boldsymbol{y} - \boldsymbol{b}\|^2 \quad \text{s.t. } \boldsymbol{y}_B = \boldsymbol{x}_B.$$

Expanding the objective gives an expression in terms of the Laplacian:

$$\min_{\boldsymbol{y} \in C^0(H;\mathcal{F})} \boldsymbol{y}^T \boldsymbol{L}\boldsymbol{y} - 2\boldsymbol{b}^T \boldsymbol{\delta}\boldsymbol{y} + \boldsymbol{b}^T \boldsymbol{b} \quad \text{s.t. } \boldsymbol{y}_B = \boldsymbol{x}_B.$$

The Lagrange multiplier conditions for optimality are

$$\boldsymbol{L}[U, U]\boldsymbol{y}_U + \boldsymbol{L}[U, B]\boldsymbol{y}_B = (\boldsymbol{\delta}^T \boldsymbol{b})_U$$
$$\boldsymbol{L}[B, U]\boldsymbol{y}_U + \boldsymbol{L}[B, B]\boldsymbol{y}_B = \boldsymbol{\lambda}$$
$$\boldsymbol{y}_B = \boldsymbol{x}_B.$$

Since $\boldsymbol{\lambda}$ is free, these equations are readily solved for $\boldsymbol{y}_U$:

$$\boldsymbol{y}_U = \boldsymbol{L}[U, U]^{-1}((\boldsymbol{\delta}^T \boldsymbol{b})_U - \boldsymbol{L}[U, B]\boldsymbol{x}_B)$$
$$= -\boldsymbol{L}[U, U]^{-1}\boldsymbol{L}[U, B]\boldsymbol{x}_B + \boldsymbol{L}[U, U]^{-1}(\boldsymbol{\delta}^T \boldsymbol{b})_U$$
$$= \boldsymbol{y}_U^{\mathcal{F}} + \boldsymbol{L}[U, U]^{-1}(\boldsymbol{\delta}^T \boldsymbol{b})_U,$$

where $\boldsymbol{y}_U^{\mathcal{F}}$ is the harmonic extension of $\boldsymbol{x}_B$ for the sheaf $\mathcal{F}$ without the affine term $\boldsymbol{b}$. We now wish to compute the optimal value; this is

$$\|\boldsymbol{\delta}\boldsymbol{y} - \boldsymbol{b}\|^2 = \boldsymbol{y}^T \boldsymbol{L}\boldsymbol{y} - 2\boldsymbol{b}^T \boldsymbol{L}\boldsymbol{y} + \boldsymbol{b}^T \boldsymbol{b}.$$

We write $\boldsymbol{y} = \boldsymbol{y}_{\mathcal{F}} + \boldsymbol{y}_b$, where $\boldsymbol{y}_{\mathcal{F}}$ is the standard harmonic extension of $\boldsymbol{x}_B$ and $\boldsymbol{y}_b = \boldsymbol{L}[U, U]^{-1}(\boldsymbol{\delta}^T \boldsymbol{b})_U$ is the affine correction computed above (extended to $v \in B$ by zero). Then the optimal value is

$$\boldsymbol{y}_{\mathcal{F}}^T \boldsymbol{L}\boldsymbol{y}_{\mathcal{F}} + 2\boldsymbol{y}_{\mathcal{F}}^T \boldsymbol{L}\boldsymbol{y}_b + \boldsymbol{y}_b^T \boldsymbol{L}\boldsymbol{y}_b - 2\boldsymbol{b}^T \boldsymbol{\delta}\boldsymbol{y}_{\mathcal{F}} - 2\boldsymbol{b}^T \boldsymbol{\delta}\boldsymbol{x}_b + \boldsymbol{b}^T \boldsymbol{b}.$$

After substituting known values of $\boldsymbol{y}_{\mathcal{F}}$ and $\boldsymbol{y}_b$ in terms of $\boldsymbol{x}_B$ and dropping terms that do not depend on $\boldsymbol{x}_B$, we have

$$\boldsymbol{y}_{\mathcal{F}}^T \boldsymbol{L}\boldsymbol{y}_{\mathcal{F}} - 2\boldsymbol{b}^T \boldsymbol{\delta}\boldsymbol{y}_{\mathcal{F}}.$$

This means that in order to calculate the affine harmonic extension cost, it suffices to compute the standard linear harmonic extension. The first term can be computed from $\boldsymbol{x}_B$ using the Schur complement $\boldsymbol{L}/\boldsymbol{L}[U, U]$, while the second term is equal to $2\boldsymbol{b}^T \left(\boldsymbol{\delta}|_B - \boldsymbol{\delta}|_U \boldsymbol{L}[U, U]^{-1}\boldsymbol{L}[U, B]\right)\boldsymbol{x}_B$. This term is linear in $\boldsymbol{x}_B$ and hence is easily computed.

Note that when $\boldsymbol{b} = 0$ this reduces to the standard harmonic extension problem, and hence gives a proof of the Schur complement formula given in the main text.

D HARMONIC EXTENSION AND MARGINALIZATION

To better convey the role of harmonic extension in the complex query completion setting, it may be helpful to reframe this operation in a more familiar statistical language. Assume entity embeddings of a knowledge graph G are distributed as 0-mean multivariate normal: $p(\boldsymbol{x}_v) = \sqrt{(2\pi)^k \det \boldsymbol{\Sigma}}^{-1} \exp -\frac{1}{2}(\boldsymbol{x}_v^T \boldsymbol{\Sigma}^{-1}\boldsymbol{x}_v)$. For a set of boundary vertices B and their

complement U, their collection of embeddings $\boldsymbol{x}_H = (\boldsymbol{x}_B; \boldsymbol{x}_U)$ is also multivariate normal with zero mean and covariance $\boldsymbol{\Sigma}_H$ a block matrix with $\boldsymbol{\Sigma}_B$ and $\boldsymbol{\Sigma}_U$ as diagonal blocks and the covariance $\boldsymbol{\Sigma}_{BU}$ filling off-diagonal blocks. The conditional covariance of the boundary embeddings $\boldsymbol{x}_B$ given $\boldsymbol{x}_U$ is the Schur complement of $\boldsymbol{\Sigma}_U$ in $\boldsymbol{\Sigma}_H$:

$$\mathbb{E}(\boldsymbol{x}_B \mid \boldsymbol{x}_U) = \mathbb{E}(\boldsymbol{x}_B) + \boldsymbol{\Sigma}_{BU}\boldsymbol{\Sigma}_U^{-1}(\boldsymbol{x}_U - \mathbb{E}(\boldsymbol{x}_U)) = \boldsymbol{\Sigma}_{BU}\boldsymbol{\Sigma}_U^{-1}\boldsymbol{x}_U$$

$$\mathrm{Cov}(\boldsymbol{x}_B \mid \boldsymbol{x}_U) = \boldsymbol{\Sigma}_B - \boldsymbol{\Sigma}_{BU}\boldsymbol{\Sigma}_U^{-1}\boldsymbol{\Sigma}_{BU}^T.$$

In this form, we see that the Laplacian of this knowledge sheaf $\boldsymbol{L}_{\mathcal{F}G}$ corresponds to the inverse covariance matrix:

$$\mathrm{Cov}(\boldsymbol{x}_B \mid \boldsymbol{x}_U)^{-1} = \boldsymbol{L}_{\mathcal{F}G}[B, B] - \boldsymbol{L}_{\mathcal{F}G}[B, U]\boldsymbol{L}_{\mathcal{F}G}[U, U]^{-1}\boldsymbol{L}_{\mathcal{F}G}[U, B].$$

It can be shown that the probability of observing any choice of embedding decomposes as a product of node and edge potential functions which are parameterized by $\boldsymbol{L}_{\mathcal{F}G}$, providing a statistical interpretation of observing an entity embedding within a knowledge graph that is inversely proportional to the discrepancy it introduces with respect to its neighbors (Malioutov et al., 2006). This statistical interpretation is reminiscent of problems like covariance selection or graphic lasso, and the relationship between sheaf embedding, graphical likelihood estimation, and belief propagation may inspire future work.

E SHEAVES AND CATEGORY THEORY

While the algebraic properties of the theory of cellular sheaves rely on the linear structure of the vector spaces that serve as stalks, the theory may be developed in a more general setting. For this we adopt the language of category theory (see Riehl (2017); Fong and Spivak (2018)). Let G be a graph, which we view as a category with one object for each vertex and edge, and a unique morphism $v \lessdot e : v \to e$ for each incident vertex-edge pair. To treat orientations properly, we also assume that morphisms are tagged with an orientation; that is, the morphism $v \lessdot e$ also records whether the pair is to be viewed as defining the head or tail of the edge, which we will write as $v \lessdot_h e$ or $v \lessdot_t e$. This is particularly important for graphs with self-loops, where for a given vertex-edge pair (v, e) there may be two distinct morphisms $v \lessdot_h e$ and $v \lessdot_t e$. (Formally, this means that the category representing G is fibered over the category with two objects and two parallel non-identity morphisms.)

A *cellular sheaf* on G valued in the data category $\mathcal{C}$ is a functor $\mathcal{F} : G \to \mathcal{C}$. We assume $\mathcal{C}$ is complete (i.e. has all small limits), and define the *global sections* of $\mathcal{F}$ to be the limit $\lim \mathcal{F}$, an object of $\mathcal{C}$. The *stalks* of $\mathcal{F}$ are the values the functor $\mathcal{F}$ takes on objects of G, and the *restriction maps* are the values of $\mathcal{F}$ on the morphisms of G. Thus, for a vertex-edge pair $v \lessdot e$, we have a restriction map $\mathcal{F}_{v \lessdot e} : \mathcal{F}(v) \to \mathcal{F}(e)$, which is a morphism in $\mathcal{C}$.

The pullback of $\mathcal{C}$-valued sheaves over a graph morphism k is well-defined for graph morphisms that send edges to edges and vertices to vertices. It is constructed in the same way as for sheaves of vector spaces. For each vertex v, $k^*\mathcal{F}(v) = \mathcal{F}(k(v))$, and for each edge e, $k^*\mathcal{F}(e) = \mathcal{F}(k(e))$. Then $k^*\mathcal{F}_{v \lessdot e} = \mathcal{F}_{k(v) \lessdot k(e)}$.

Since $\mathcal{C}$ is complete, we can define the object $C^0(G; \mathcal{F}) = \prod_v \mathcal{F}(v)$ in $\mathcal{C}$, the product of all vertex stalks of $\mathcal{F}$. The global sections of $\mathcal{F}$ naturally form a subobject of $C^0(G; \mathcal{F})$; when $\mathcal{C}$ is a concrete category, we can think of sections of $\mathcal{F}$ as elements $(x_v)_{v \in V(G)} \in C^0(G; \mathcal{F})$ such that for every edge $e = u \to v$, $\mathcal{F}_{u \lessdot_h e}x_u = \mathcal{F}_{v \lessdot_t e}x_v$.

We can similarly define $C^1(G; \mathcal{F}) = \prod_{e \in E(G)} \mathcal{F}(e)$ as an object in $\mathcal{C}$. If $\mathcal{C}$ is the category of groups (or a subcategory thereof), we can define a coboundary map $\delta : C^0(G; \mathcal{F}) \to C^1(G; \mathcal{F})$ by letting $(\delta x)_e = (\mathcal{F}_{u \lessdot_h e}x_v)^{-1}(\mathcal{F}_{v \lessdot_t e}x_v)$. When $\mathcal{C} = \mathbf{Vect}$, the category of vector spaces, this definition recovers the definition of the coboundary given in the paper.

We actually require slightly less structure to define a coboundary map; it is sufficient for $\mathcal{C}$ to be a category of group torsors. Recall that for a group S, an S-torsor A is a set equipped with an action $\cdot : S \times A \to A$ and a division map $D : A \times A \to S$, such that $D(a, b) \cdot b = a$. We can formally think of $D(a, b)$ as being ab^{-1}, in which case the formula is the natural $ab^{-1} \cdot b = a$. The reason for this extension is to allow a broader class of maps (i.e. more than just group homomorphisms) between embedding spaces. A morphism of torsors is not required to preserve the origin, but does preserve the division operation. To compute the coboundary operator of a torsor-valued sheaf, which we think of as a function between the underlying sets of $C^0(G; \mathcal{F})$ and $C^1(G; \mathcal{F})$, we let $(\delta x)_e = D(\mathcal{F}_{u \lessdot_h e}x_u, \mathcal{F}_{v \lessdot_t e}x_v)$ for $e = u \to v$. The coboundary is then valued in a product of groups: if the stalk $\mathcal{F}(e)$ is an S_e-torsor, the coboundary δx is in $\prod_e S_e$. When these groups are given a metric, we can then compute a cost function for a 0-cochain x by letting $U_{\mathcal{F}}(x) = \sum_{e \in E(G)} d((\delta_x)_e, 1_{S_e})$, where 1_{S_e} is the identity of the group S_e. The cost function $U_{\mathcal{F}}$ vanishes exactly on those 0-cochains x which are sections of $\mathcal{F}$.

Every vector space is an abelian group under addition, and the category of torsors over vector spaces is equivalent to the category $\mathbf{Aff}$ of vector spaces and affine maps. In this category, a morphism $f : V \to W$ is given by a formula of the form

$f(\boldsymbol{v}) = T(\boldsymbol{v}) + \boldsymbol{b}$, where T is a linear transformation $V \to W$ and $\boldsymbol{b} \in W$. The coboundary map of a sheaf valued in **Aff** is given on edges by $(\boldsymbol{\delta x})_e = \mathcal{F}_{v \trianglelefteq e} \boldsymbol{x}_v + \boldsymbol{b}_{v \trianglelefteq e} - \mathcal{F}_{u \trianglelefteq e} \boldsymbol{x}_u - \boldsymbol{b}_{u \trianglelefteq e}$. This is equivalent to the coboundary map of a linear sheaf $\mathcal{F}$ with an edgewise affine correction term. Thus, for the purposes of knowledge graph embedding, working with sheaves valued in **Aff** is equivalent to using a sheaf valued in **Vect** and learning $\boldsymbol{x}$ and $\boldsymbol{b}$ such that $\boldsymbol{\delta x} \approx \boldsymbol{b}$ rather than $\boldsymbol{\delta x} \approx 0$. Passing to sheaves valued in **Aff** thus adds a translational component to the embedding model.

As a result, we can think of the relation between embeddings with group-valued sheaves and embeddings with torsor-valued sheaves as analogous the relationship between pure sheaf embeddings and sheaf embeddings with a translational component.

If we abandon the prospect of a translational component to the embedding, we can further relax our requirements on the data category $\mathcal{C}$. If $\mathcal{C}$ is a category of metric spaces, we can construct a measure of the discrepancy of a 0-cochain $x \in C^0(G; \mathcal{F})$ by

$$V_{\mathcal{F}}(x) = \sum_{e \in E(G)} d_{\mathcal{F}(e)}(\mathcal{F}_{u \trianglelefteq e}(x_u), \mathcal{F}_{v \trianglelefteq e}(x_v)).$$

Optimizing this function with respect to the restriction maps $\mathcal{F}_{v \trianglelefteq e}$ and the 0-cochain x produces a generalized knowledge graph embedding. In this setting, it is most reasonable to take $\mathcal{C}$ to be a category of Euclidean spaces and smooth (or at least almost-everywhere differentiable) maps, so that we can apply automatic differentiation and gradient descent.

The common thread in all these sheaf-theoretic constructions is the notion of comparison. To evaluate the plausibility of a relation holding between two entities, the entity embeddings are both transformed into a comparison space, and some measure of discrepancy between these transformed embeddings is calculated, giving a scoring function. Many commonly used knowledge graph scoring functions fit neatly into this framework, but some do not.

F   ADDITIONAL EXPERIMENTAL DETAILS

The "easy" test set for each complex query structure consists of query structures composed of individual triplets that have been seen at least once in the training set whereas the "hard" test set answers require at least one edge absent from the training set to be inferred (Ren and Leskovec, 2020). Because each complex query in the test set may have a number of feasible answers, we compute performance on the filtered dataset which amounts to reducing the ranking of entities by the number of acceptable answers for each query, computed independently for the "easy" and "hard" answer sets.

We compute the mean reciprocal rank (MRR) and Hits at 10 (Hits@10) from the rankings, according to their assigned score per test query, across all entities in the knowledge graph. Given a set of test triplets $\mathcal{T}_{\text{test}}$, MRR is defined as

$$\text{MRR} = \frac{1}{|\mathcal{T}_{\text{test}}|} \sum_{(h,r,t) \in \mathcal{T}_{\text{test}}} \frac{1}{\text{rank}(t)}$$

For a ranking, denoted $\text{rank}(t)$, of the true entity t score over the score of all other entities in the knowledge graph. The hits at K metrics are defined as the proportion of true entities with ranking below some threshold K:

$$\text{Hits@}K = \frac{|\{(h,r,t) \in \mathcal{T}_{\text{test}} \mid \text{rank}(t) \leq K\}|}{|\mathcal{T}_{\text{test}}|}$$

We also trained versions of these models to learn multiple sections simultaneously, as detailed in Section 4.3. For such models, we were also interested in the performance effects of forcing these sections to being close to orthogonal, thereby increasing the size of the space of global sections. For these models, we vary the α orthogonal regularization parameter across $[0, 0.01, 0.1, 1]$ and vary the number of sections between $[1, 16, 32]$ to observe the effect of this regularization method. To investigate the performance effects of simultaneously learning multiple sections, we aggregated results across models with varying orthogonal regularization penalty values α. The decrease in performance associated with orthogonal section regularization as depicted in Figure 5 implies that encouraging local orthogonality of the learned embedding cochains may be too strong a condition to produce useful embeddings. Instead, it appears that less-constrained representations are sufficient to solve the knowledge graph completion task, even in the context of complex logical queries. However, an interesting avenue of future work would be to investigate the extent to which this observation holds on complex queries whose structure has not been seen during training or on knowledge graphs with a greater diversity of abstract entity types which can appear within a variety of relational contexts.
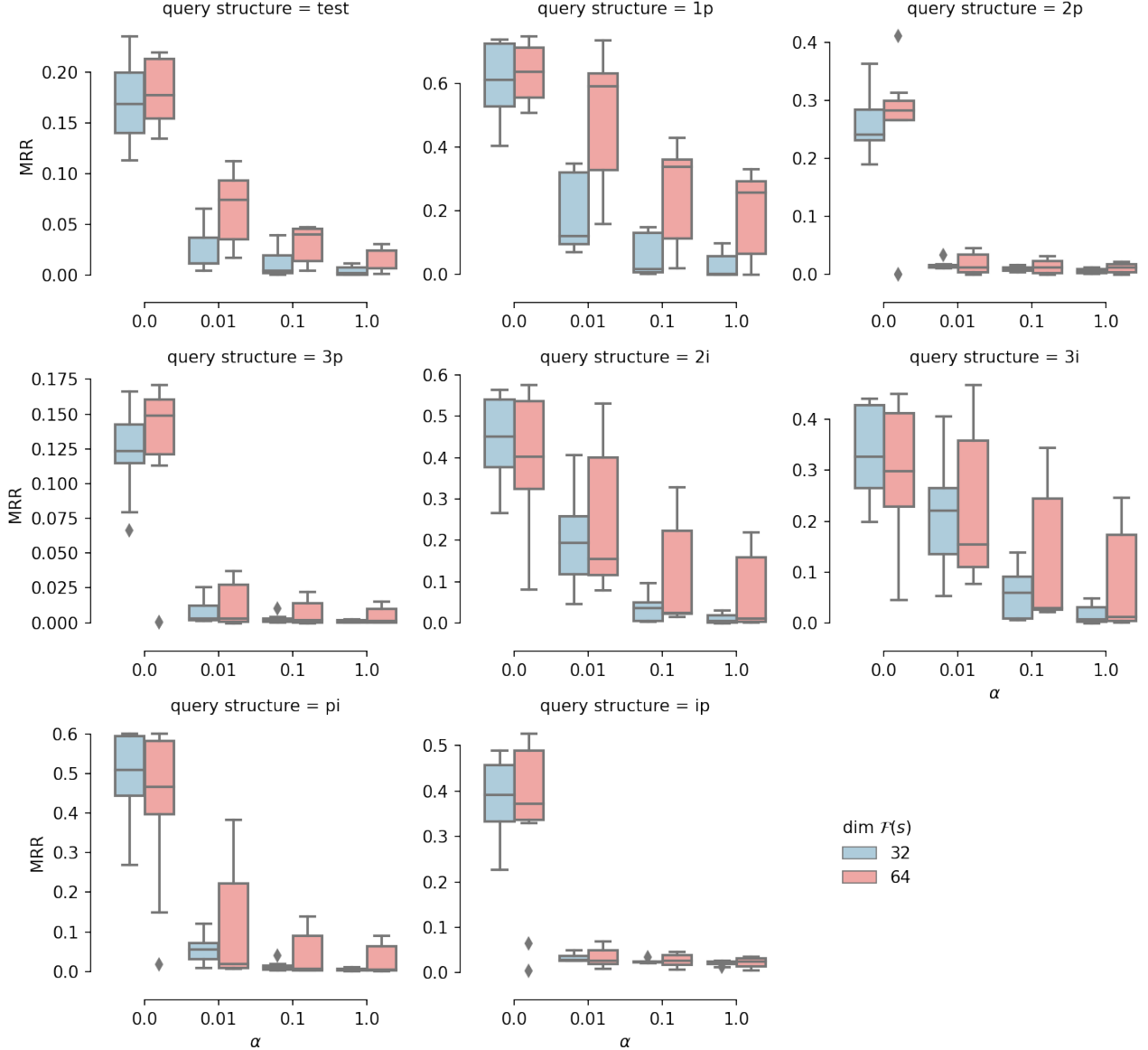
Figure 5: Effects of orthogonal regularization of sections on model performance by query structure and entity embedding dimensionality on NELL-995. Each color of box-and-whisker plot denotes the embedding and edge stalk dimensionality $\dim \mathcal{F}(s) = \dim \mathcal{F}(r)$. Results are aggregated across all model types. The x-axis denotes the orthogonal regularization weight α. Evaluation on "easy" test queries.

F.1 Baseline Comparison for Complex Queries

We evaluated the `ExtensionSE` and `ExtensionTransE` models on the evaluation procedure used in Ren and Leskovec (2020), allowing for direct comparison of performance between the harmonic extension models proposed in this work versus three state-of-the-art models for complex query reasoning: BetaE (Ren and Leskovec, 2020), Q2B Ren et al. (2020), and GQE (Hamilton et al., 2018). The evaluation procedure amounts to evaluating on the "hard" test answer and filtering all "easy" answers for a given query in conjunction with the other correct "hard" answers. In short, we combine the "easy" and "hard" answer sets but only rank-score the "hard" answers.

Table 2 displays the performance of these harmonic extension models versus the state-of-the-art results reported in Ren and Leskovec (2020). Due to computational limitations, and to emphasize harmonic extension as a convenient drop-in method for extending traditional knowledge graph embedding models to conjunctive query reasoning, we set
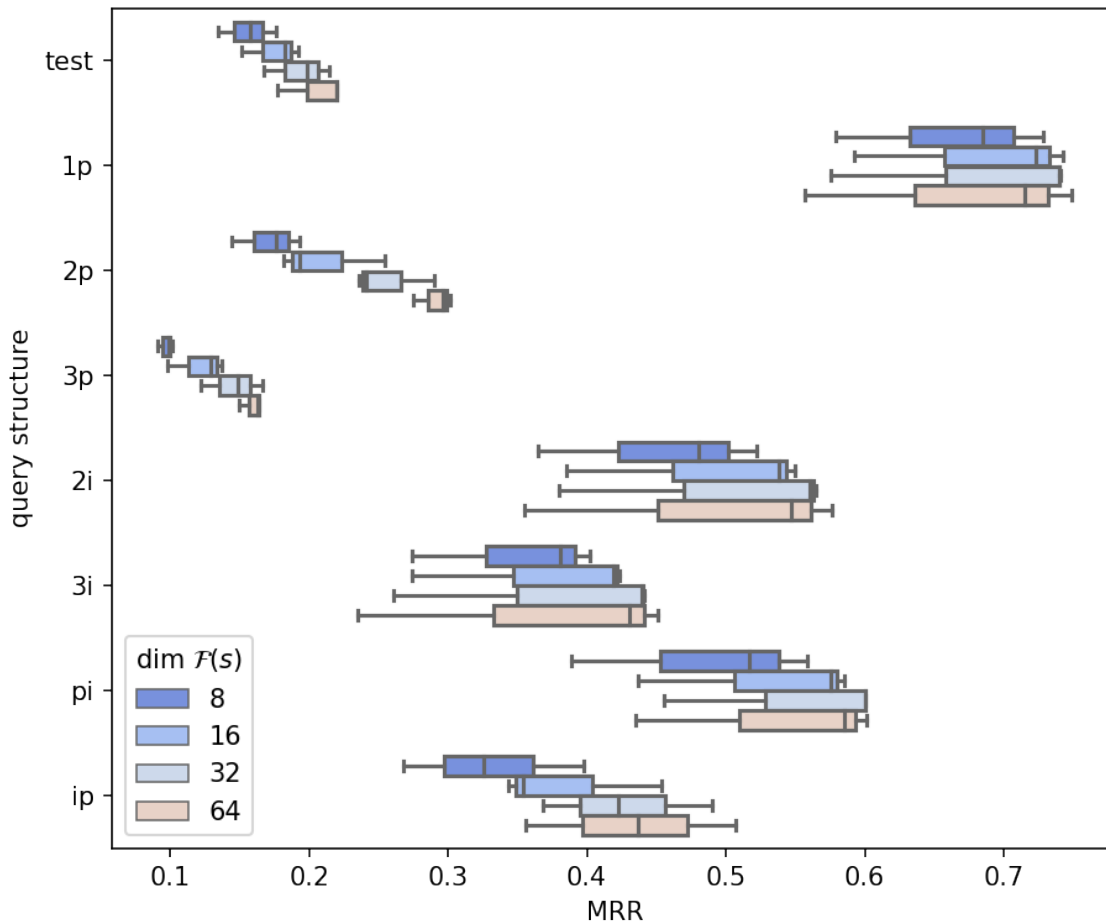
Figure 6: Performance on NELL-995 across varying embedding dimensions $\dim \mathcal{F}(s)$ for models with $\dim \mathcal{F}(s) = \dim \mathcal{F}(r)$ and 32 unregularized sections. Each box-and-whisker plot is aggregated across the three model types without orthogonal restriction map parameterization. Evaluation on "easy" test queries.

$\dim \mathcal{F}(s) = \dim \mathcal{F}(r) = 32$ and learn one section for each of the `ExtensionSE` and `ExtensionTransE` models. We also train these models using the traditional triplet scoring approach using margin ranking loss. This is in contrast to the baseline models which embed the query itself and therefore require a training set composed of complex queries along with factual triplets.

The model structure and training choices described above results in the `ExtensionSE` and `ExtensionTransE` models being structurally equivalent to the Structured Embedding and TransE models, respectively, with 32-dimensional embedding dimension. The number of tunable parameters for these models are on the order of 1 million parameters for FB15k-237 and 2 million parameters for NELL-995. This is in contrast to the baseline methods of Ren and Leskovec (2020) which are on the order of 10 million for FB15k-237 and $30 - 50$ million for NELL-995.

Despite this difference in model size (90-95% compression), the `ExtensionSE` and `ExtensionTransE` models are able to achieve performance within up to 50% of these baselines on a number of conjunctive query structures. These results imply the possibility that the harmonic extension approach to complex query reasoning may be able to achieve performance matching or exceeding that of BetaE and related query embedding models if we are able to find a base model–like Structured Embedding, TransE, or other embedding methods which fall within the sheaf-theoretic framework–that achieves 1p MRR

dataset	model	1p	2p	3p	2i	3i	pi	ip	parameters (M)
FB15k-237	BetaE	0.390	0.109	0.100	0.288	0.425	0.224	0.126	14.3
	Q2B	0.406	0.094	0.068	0.295	0.423	0.212	0.126	6.8
	GQE	0.350	0.072	0.053	0.233	0.346	0.165	0.107	13.3
	ExtensionSE	0.197	0.024	0.013	0.085	0.110	0.055	0.040	1.4
	ExtensionTransE	0.207	0.027	0.021	0.071	0.103	0.045	0.047	0.5
NELL	BetaE	0.530	0.130	0.114	0.376	0.475	0.241	0.143	53.4
	Q2B	0.422	0.140	0.112	0.333	0.445	0.224	0.168	26.3
	GQE	0.328	0.119	0.096	0.275	0.352	0.184	0.144	52.3
	ExtensionSE	0.159	0.037	0.024	0.109	0.118	0.125	0.072	2.8
	ExtensionTransE	0.187	0.035	0.028	0.083	0.079	0.088	0.069	2.0

Table 2: Performance of logical query baselines versus the harmonic extension solution for simple StructuredEmbedding and TransE models using the evaluation setup of Ren and Leskovec (2020). The first seven columns measure MRR performance for each complex query structure while the final column measures the number of trainable parameters (in millions). The Extension models set $\dim \mathcal{F}(s) = \dim \mathcal{F}(r) = 32$ and contain one section. Performance metrics for BetaE, Q2B, and GQE models taken from Ren and Leskovec (2020).

performance in line with these baselines models. Such performance may be achievable, as previous works have shown that, for proper hyperparameter choices, models like TransE or RotatE can achieve MRR performance above 0.3 on FB15k-237, closing the gap between the baselines and the extension models trained in this work. We leave proper experimental validation of this observation to future work.

dataset	model	2p	3p	2i	3i	pi	ip
NELL-995	ExtensionTransE	0.355	0.201	0.467	0.368	0.499	0.427
	NaiveTransE	0.280	0.148	0.488	0.403	0.451	0.216
FB15k-237	ExtensionTransE	0.163	0.107	0.343	0.341	0.253	0.207
	NaiveTransE	0.162	0.095	0.362	0.363	0.229	0.097

dataset	model	2p	3p	2i	3i	pi	ip
NELL-995	ExtensionTransE	0.101	0.087	0.298	0.393	0.198	0.124
	NaiveTransE	0.086	0.060	0.308	0.402	0.173	0.088
FB15k-237	ExtensionTransE	0.049	0.028	0.213	0.312	0.158	0.086
	NaiveTransE	0.049	0.024	0.213	0.307	0.142	0.045

Table 3: Comparative H@10 performance of the extension and naive TransE approaches for "easy" (top) and "hard" (bottom) complex query answers on NELL and FB15k-237 datasets.
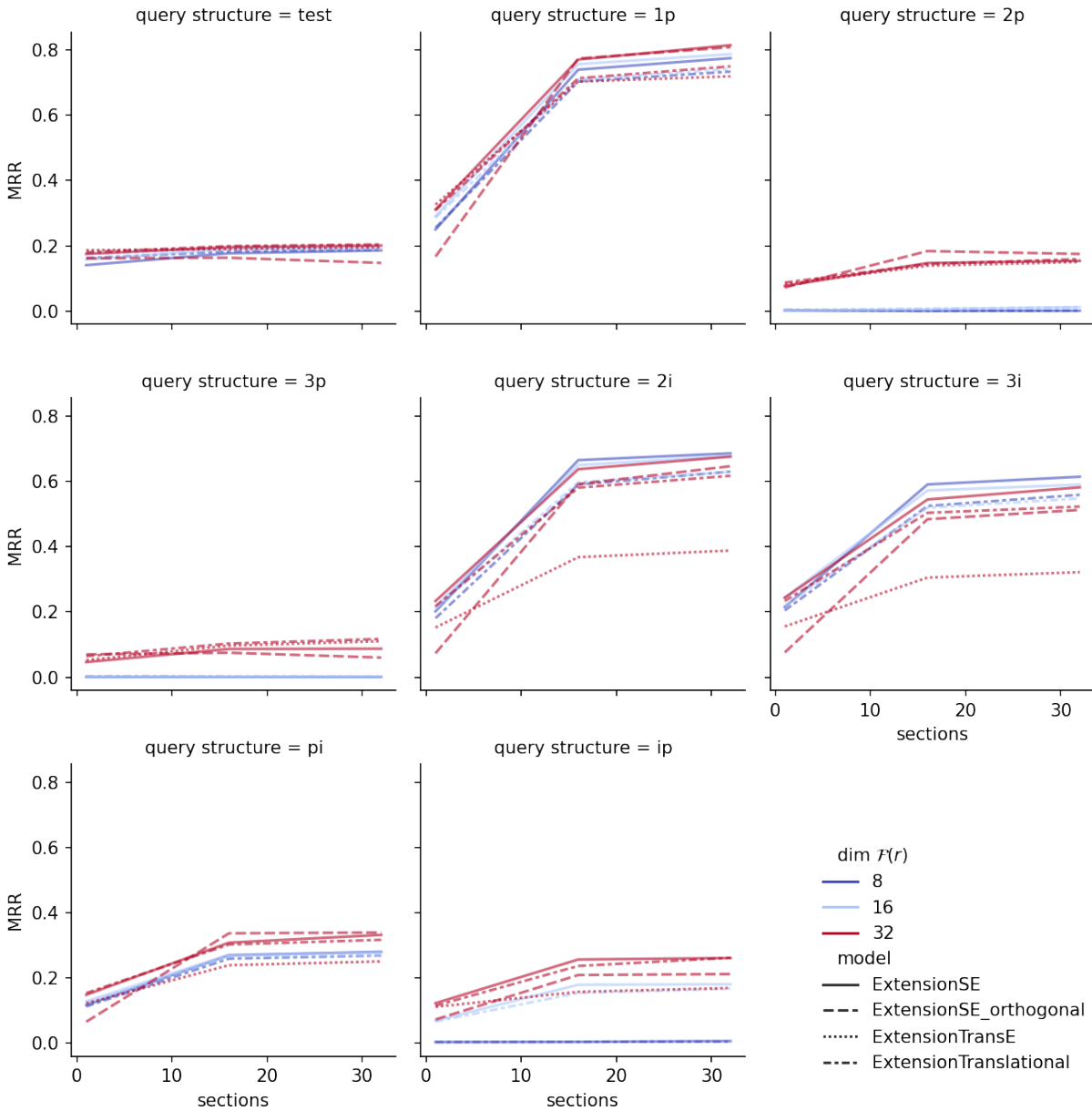
Figure 7: Triplet and complex query completion performance across varying number of sections and edge stalk dimensionality for each model class on FB15k-237. Each sub-figure displays the results for each complex query type, including the test dataset. Line styles correspond to different model types and their colors denote the dimensionality of the relation stalk space $\dim \mathcal{F}(r)$. The entity embedding dimension $\dim \mathcal{F}(s)$ is held at 32 for each model. Evaluation on "easy" test queries.
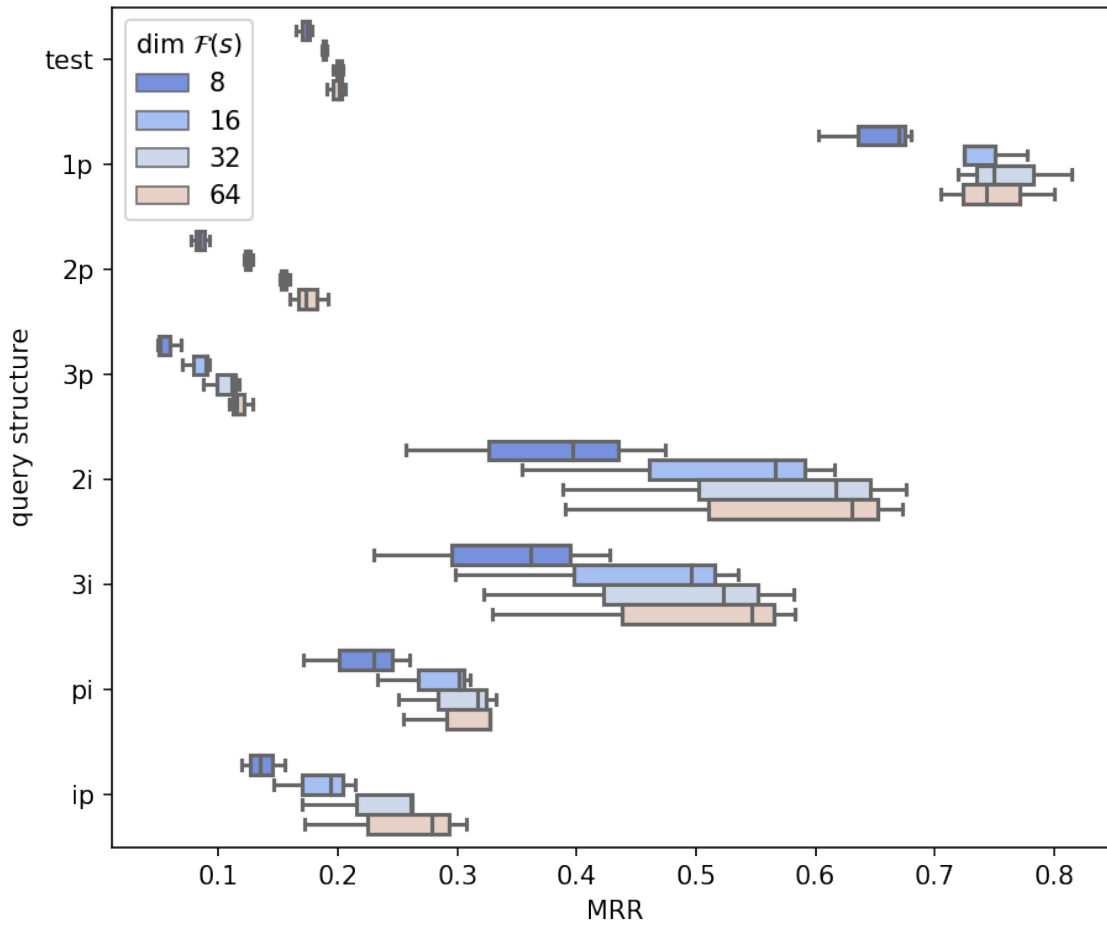
Figure 8: Performance on FB15k-237 across varying embedding dimensions $\dim \mathcal{F}(s)$ for models with $\dim \mathcal{F}(s) = \dim \mathcal{F}(r)$ and 32 unregularized sections. Each box-and-whisker plot is aggregated across the three model types without orthogonal restriction map parameterization. Evaluation on "easy" test queries
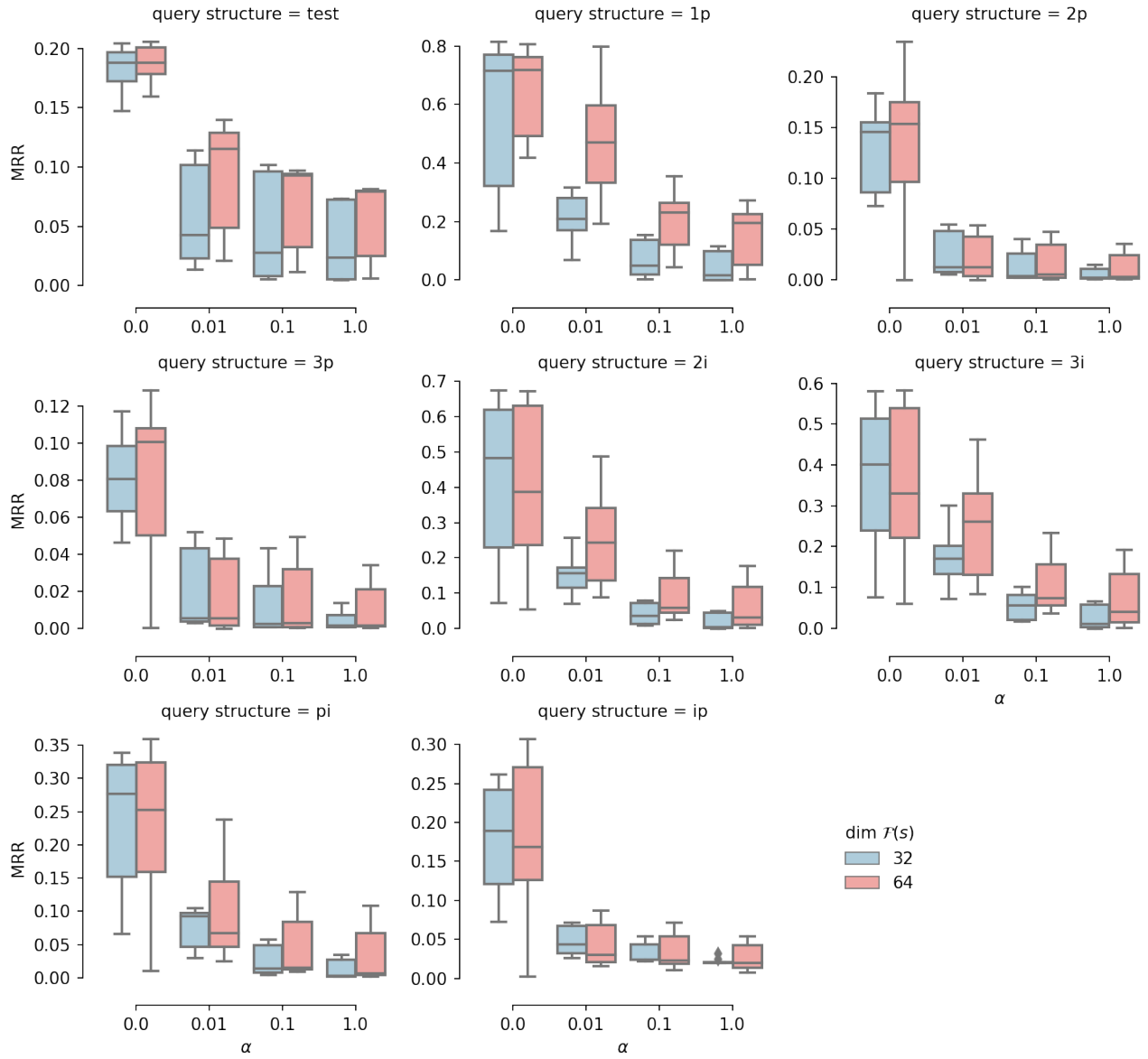
Figure 9: Effects of orthogonal regularization of sections on model performance by query structure and entity embedding dimensionality on FB15k-237. Each color of box-and-whisker plot denotes the embedding and edge stalk dimensionality $\dim \mathcal{F}(s) = \dim \mathcal{F}(r)$. Results are aggregated across all model types. The x-axis denotes the orthogonal regularization weight α. Evaluation on "easy" test queries.