TRANSINT: EMBEDDING IMPLICATION RULES IN KNOWLEDGE GRAPHS WITH ISOMORPHIC INTERSECTIONS OF LINEAR SUBSPACES

Anonymous authors
Paper under double-blind review

ABSTRACT

Knowledge Graphs (KG), composed of entities and relations, provide a structured representation of knowledge. For easy access to statistical approaches on relational data, multiple methods to embed a KG into $f(\text{KG}) \in \mathbb{R}^d$ have been introduced. We propose TransINT, a novel and interpretable KG embedding method that isomorphically preserves the implication ordering among relations in the embedding space. TransINT maps set of entities (tied by a relation) to continuous sets of vectors that are inclusion-ordered isomorphically to relation implications. With a novel parameter sharing scheme, TransINT enables automatic training on missing but implied facts without rule grounding. We achieve new state-of-the-art performances with signficant margins in Link Prediction and Triple Classification on FB122 dataset, with boosted performance even on test instances that cannot be inferred by logical rules. The angles between the continuous sets embedded by TransINT provide an interpretable way to mine semantic relatedness and implication rules among relations.

1 INTRODUCTION

Recently, learning distributed vector representations of multi-relational knowledge has become an active area of research (Bordes et al.; Nickel et al.; Kazemi & Poole; Wang et al.; Bordes et al.). These methods map components of a KG (entities and relations) to elements of $\mathbb{R}^d$and capture statistical patterns, regarding vectors close in distance as representing similar concepts. However, they lack common sense knowledge which are essential for reasoning (Wang et al.; Guo et al.; Nickel & Kiela). For example, “parent" and “father" would be deemed similar by KG embeddings, but by common sense, "parent $\Rightarrow$ father" yet not the other way around. Thus, one focus of current research is to bring common sense rules to KG embeddings (Guo et al.; Wang et al.; Wei et al.(. Some methods impose hard geometric constraints and embed asymmetric orderings of knowledge (Nickel & Kiela; Vendrov et al.; Vilnis et al.(. However, they only embed hierarchy (unary *Is_a* relations), and cannot embed n-ary relations in KG's. Moreover, their hierarchy learning is largely incompatible with conventional relational learning, because they put hard constraints on distance to represent partial ordering, which is a common metric of similarity/ relatedness in relational learning.

We propose TransINT, a new KG embedding method that isomorphically preserves the implication ordering among relations in the embedding space. TransINT restrict entities tied by a relation to be embedded to vectors in a particular region of $\mathbb{R}^d$ included isomorphically to the order of relation implication. For example, we map any entities tied by *is_father_of* to vectors in a region that is part of the region for *is_parent_of*; thus, we can automatically know that if John is a father of Tom, he is also his parent even if such a fact is missing in the KG. Such embeddings are constructed by sharing and rank-ordering the basis of the linear subspaces where the vectors are required to belong.

Mathematically, a relation can be viewed as sets of entities tied by a constraint (Stoll). We take such a view on KG's, since it gives consistancy and interpretability to model behavior. Furthermore, for the first time in KG embedding, we map sets of entitites under relation constraint to a continuous set of points (whose elements are entity vectors) - which learns relationships among not only individual entity vectors but also sets of entities. We show that angles between embedded relation sets can identify semantic patterns and implication rules - an extension of the line of thought as in word/ image embedding methods such as Mikolov et al., Frome et al. to relational embedding. Such mining is both limited and less interpretable if embedded sets are discrete (Vilnis et al.; Vendrov et al.) or each entitity itself is embedded to a region, not a member vector of it (Vilnis et al.).[1] TransINT's

[1] Vilnis et al. can be interpreted in both ways.

such interpretable meta-learning opens up possibilities for explainable reasoning in applications such as recommender systems (Ma et al.) and question answering (Hamilton et al.).

The main contributions of our work are: (1) A novel KG embedding such that implication rules in the original KG are guaranteed to unconditionally, not approximately, hold. (2) We introduce a novel parameter sharing regularization and negative example construction methods. (3) Our model suggests possibilities of learning semantic relatedness between groups of objects. (4) We achieve new state-of-the-art performances with large margins in Link Prediction and Triple Classification on FB122 datasets.

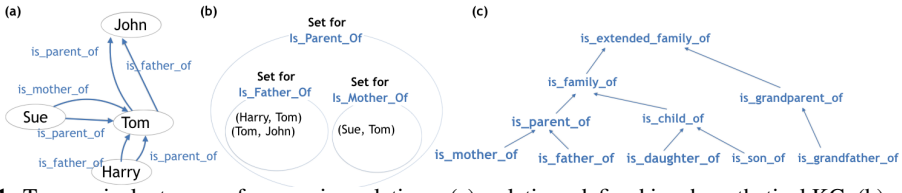

Figure 1: Two equivalent ways of expressing relations. (a): relations defined in a hypothetical KG. (b): relations defined in a set-theoretic perspective (Definition 1). Because *is_father_of* $\Rightarrow$ *is_parent_of*, the set for *is_father_of* is a subset of that for *is_parent_of* (Definition 2). (c): Hierarchical depiction of familial relations.

2 TRANSINT

In this section, we describe the intuition and justification of our method. We first define relation as sets, and revisit TransH as mapping relations to sets in $\mathbb{R}^d$. Finally, we propose TransINT, which connects the ordering of the two aforementioned sets. We put * next to definitions and theorems we propose/ introduce. Otherwise, we use existing definitions and cite them.

2.1 SETS AS RELATIONS

We define relations as sets and implication as inclusion of sets, as in set-theoretic logic.

Definition *(Relation Set):* Let r_i be a binary relation x, y entities. Then, $r_i(x, y)$ iff there exists some set $\mathbf{R_i}$ such that the pair $(x, y) \in \mathbf{R_i}$. $\mathbf{R_i}$ is called the **relation set** of r_i.(Stoll)

For example, consider the distinct relations in Figure 1a, and their corresponding sets in Figure 1b; *Is_Father_Of(Tom, Harry)* is equivalent to *(Tom, Harry)* $\in \mathbf{R}_{\text{Is\_Father\_Of}}$.

Definition *(Logical Implication):* For two relations, r_1 implies r_2 (or $r_1 \Rightarrow r_2$) iff $\forall x, y$,

$$(x, y) \in \mathbf{R_1} \Rightarrow (x, y) \in \mathbf{R_2} \quad \text{or equivalently,} \quad \mathbf{R_2} \subset \mathbf{R_1}.\text{(Stoll)}$$

For example, *Is_Father_Of* $\Rightarrow$ *Is_Parent_Of*. (In Figure 1b, $\mathbf{R_{Is\_Father\_Of}} \subset \mathbf{R_{Is\_Parent\_Of}}$).

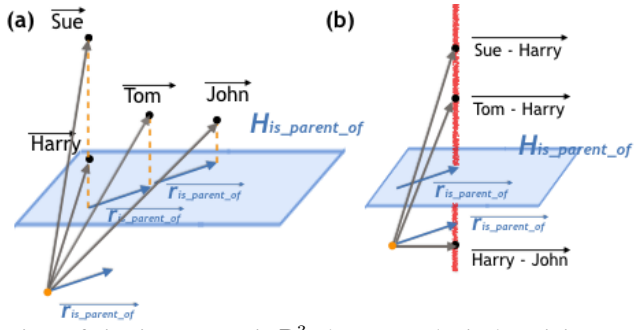

Figure 2: Two perspectives of viweing TransH in $\mathbb{R}^3$; the orange dot is the origin, to emphasize that a vector is really a point from the origin but can be translated and considered equivalently. (a): first projecting $\vec{h}$ and $\vec{t}$ onto $H_{is\_family\_of}$, and then requiring $\vec{h_\perp} + \vec{r_j} \approx \vec{t_\perp}$ (b): first substracting $\vec{t}$ from $\vec{h}$, and then projecting the distance $(\vec{t - h})$ to $H_{is\_family\_of}$ and requiring $(\vec{t - h})_\perp \approx r_j$. The red line is unique because it is when $\vec{r_{is\_family\_of}}$ is translated to the origin.

2.2 BACKGROUND: TRANSE AND TRANSH

Given a fact triple (h, r, t) in a given KG (i.e. *(Harry, is_father_of, Tom)*), TransE wants $\vec{h} + \vec{r} \approx \vec{t}$ where $\vec{h}, \vec{r}, \vec{t}$ are embeddings of h, r, t. In other words, the distance between two entity vectors is equal to a fixed relation vector. TransE applies well to 1-to-1 relations but has issues for N-to-1, 1-to-N and N-to-N relations, because the distance between two vectors are unique and thus two entities can only be tied with one relation.

To address this, TransH constraints the distance of entities in a multi-relational way, by decomposing distance with projection (Figure 2a). TransH first projects an entity vector into a hyperplane unique to each relation, and then requires their difference is some constant value. Like TransE, it embeds

an entity to a vector. However, for each relation r_j, it assigns **two** components: a relation-specific hyperplane H_j and a fixed vector $\vec{r_j}$ on H_j. For each fact triple (h, r_j, t), TransH wants (Figure 2)

$$\overrightarrow{h_\perp} + \vec{r_j} \approx \overrightarrow{t_\perp} \cdots\cdots \quad \text{(Eq. 1)}$$

where $\overrightarrow{h_\perp}, \overrightarrow{t_\perp}$ are projections on $\vec{h}$, $\vec{t}$ onto H_j (Figure 2a).

Revisiting TransH We interpret TransH in a novel perspective. An equivalent way to put Eq.1 is to change the order of subtraction and projection:

$$\text{Projection of } (\overrightarrow{t-h}) \text{ onto } H_j \approx \vec{r_j}.$$

This means that all entity vectors $(\vec{h}, \vec{t})$ such that their distance $\overrightarrow{t-h}$ belongs to the red line are considered to be tied by relation r_j (Figure 2b) i.e. $\mathbf{R_j} \approx$ the red line. For example,

$$\textit{(Tom, Sue)} \in \mathbf{R_j} \cong \overrightarrow{(Sue - Tom)} \in \text{the red line}$$

The red line is the set of all vectors whose projection onto H_j is the fixed vector $\vec{r_j}$. Thus, upon a deeper look, **TransH actually embeds a relation set in KG (figure 1b) to a particular set in $\mathbb{R}^d$.** We call such sets **relation space** for now; in other words, a **relation space** of some relation r_i is the space where each (h, r_i, t)'s $\overrightarrow{t-h}$ can exist. We formally visit it later in Section **3.1**.

Thus, in TransH,

$$r_i(x, y) \equiv (x, y) \in \mathbf{R_i} \qquad \textbf{(relation in KG)}$$
$$\cong (x - y) \in \text{relation space of } r_i \quad \textbf{(relation in } \mathbb{R}^{\mathbf{d}}\textbf{)}$$

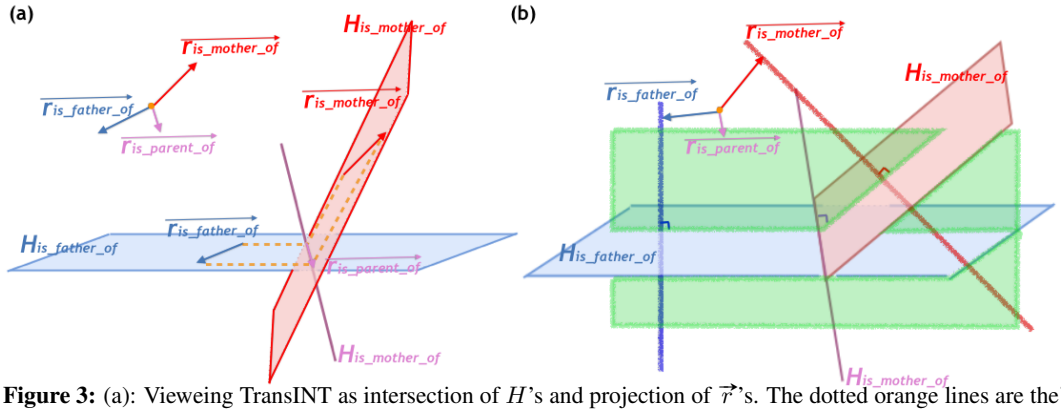

Figure 3: (a): Vieweing TransINT as intersection of H's and projection of $\vec{r}$'s. The dotted orange lines are the projection constraint. (b): Viewing TransINT in the relation space (Figure 2b) perspective. The blue line, red line, and the green plane is respectively *is_father_of, is_mother_of* and *is_parent_of*'s relation space - where $\overrightarrow{t-h}$'s of h, t tied by these relations can exist. The blue and the red line lie on the green plane - *is_parent_of*'s relation space includes the other two's.

2.3 TRANSINT

Like TransH, TransINT embeds a relation r_j to a (subspace, vector) pair $(H_j, \vec{r_j})$. However, TransINT modifies the relation embeddings $(H_j, \vec{r_j})$ so that the relation spaces (i.e. red line of Figure 2b) are ordered by implication; we do so by intersecting the H_j's and projecting the $\vec{r_j}$'s (Figure 3a). We explain with familial relations as a running example.

Intersecting the H_j's TransINT assigns distinct hyperplanes $H_{is\_father\_of}$ and $H_{is\_mother\_of}$ to *is_father_of* and *is_mother_of*. However, because *is_parent_of* is implied by the aforementioned relations, we assign

$$H_{is\_parent\_of} = H_{is\_father\_of} \cap H_{is\_mother\_of}.$$

TrainsINT's $H_{is\_parent\_of}$ is not a hyperplane but a line (Figure 3a), unlike in TransH where all H_j's are hyperplanes.

Projecting the $\vec{r_j}$'s TransH constrains the $\vec{r_j}$'s with projections (Figure 3a's dotted orange lines). First, $\overrightarrow{r_{is\_father\_of}}$ and $\overrightarrow{r_{is\_mother\_of}}$ are required to have the same projection onto $H_{is\_parent\_of}$. Second, $\overrightarrow{r_{is\_parent\_of}}$ is that same projection onto $H_{is\_parent\_of}$.

Connection to Relation Spaces We connect the two above constraints to ordering relation spaces. Figure 3b graphically illustrates that *is_parent_of*'s relation space (green hyperplane) includes those of *is_father_of* (blue line) and *is_mother_of* (red line).

More generally, TransINT requires two hard geometric constraints on $(H_j, \overrightarrow{r_j})$'s that

For distinct relations r_i, r_j, require the following if and only if $r_i \Rightarrow r_j$:
Intersection Constraint: $H_j = H_i \cap H_j$.
Projection Constraint: Projection of $\overrightarrow{r_1}$ to H_j is $\overrightarrow{r_j}$.
where $\overrightarrow{H_i}, \overrightarrow{H_j}$ and $\overrightarrow{r_i}, \overrightarrow{r_j}$ are distinct.

We prove that these two constraints guarantee that an ordering isomorphic to implication holds in the embedding space: $(r_i \Rightarrow r_j)$ iff (r_i's rel. space $\subset r_j$'s rel. space) or equivalently,

$(R_i \subset R_j)$ iff (r_i's rel. space $\subset r_j$'s rel. space)

The orderings are isomorphic, because for example, if *is_parent_of* subsumes *is_father_of*, the first relation space also subsumes the latter's (Figure 3). At first sight, it may look paradoxical that the H_j's and the relation spaces are inversely ordered; however, it is a natural consequence of the rank-based geometry in $\mathbb{R}^d$.

3 TRANSINT'S ISOMORPHIC GUARANTEE

In this section, we formally state TransINT's isomorphic guarantee and its grounds. We also discuss the intuitive meaning of our method. We denote all $d \times d$ matrices with capital letters (ex) A) and vectors with arrows on top (ex) $\overrightarrow{b}$).

3.1 PROJECTION AND RELATION SPACE

In $\mathbb{R}^d$, points are projected to linear subspaces by projection matrices; each linear subspace H_i has a projection matrix P_i such that $\forall \overrightarrow{x} \in \mathbb{R}^d, Px \in H$ (Strang). For example, in Figure 4, a random point $\overrightarrow{a} \in \mathbb{R}^d$ is projected onto H_1 when multiplied by P_1; i.e. $P_1 a = \overrightarrow{b} \in H_1$. In the rest of the paper, denote P_i as the projection matrix onto subspace H_i.

Now, we algebraically define a general concept that subsumes relation space(Figure 3b).

Definition* $(Sol(P, \overrightarrow{k}))$: Let H be a linear subspace and P its projection matrix. Then, given $\overrightarrow{k}$ on H, the set of vectors that become $\overrightarrow{k}$ when projected on to H, or the solution space of $P\overrightarrow{x} = \overrightarrow{k}$, is denoted as $\mathbf{Sol}(\mathbf{P}, \overrightarrow{k})$.

With this definition, relation space (Figure 3b) is $(Sol(P_i, \overrightarrow{r_i}))$, where P_i is the projection matrix of H_i (subspace for relation r_i); it is the set of points $t - h$ such that $P_i(\overrightarrow{t - h}) = \overrightarrow{r_i}$.

3.2 ISOMORPHIC GUARANTEES

***Main Theorem 1** (Isomorphism):* Let $\{(H_i, \overrightarrow{r_i})\}_n$ be the (subspace, vector) embeddings assigned to relations $\{\mathbf{R_i}\}_n$ by the *Intersection Constraint* and *the Projection Constraint*; P_i the projection matrix of H_i. Then, $(\{Sol(P_i, \overrightarrow{r_i})\}_n, \subset)$ is isomorphic to $(\{\mathbf{R_i}\}_n, \subset)$.

In actual implementation and training, TransINT requires something less strict than $P_i(\overrightarrow{t - h}) = \overrightarrow{r_i}$:

$$P_i(\overrightarrow{t - h}) - \overrightarrow{r_i} \approx \overrightarrow{0} \equiv ||P_i(\overrightarrow{t - h} - \overrightarrow{r_i})||_2 < \epsilon,$$

for some non-negative and small ϵ. This bounds $\overrightarrow{t - h} - \overrightarrow{r_i}$ to regions with thickness 2ϵ, centered around $Sol(P_i, \overrightarrow{r_i})$ (Figure 5). We prove that isomorphism still holds with this weaker requirement.

Definition* $(Sol_\epsilon(P, k))$: Given a projection matrix P, the solution space of $||P\overrightarrow{x} - \overrightarrow{k}||_2 < \epsilon$ is denoted as $\mathbf{Sol}_\epsilon(\mathbf{P}, \overrightarrow{k})$.

***Main Theorem 2** (Margin-aware Isomorphism):* For all non-negative scalar ϵ, $(\{Sol_\epsilon(P_i, \overrightarrow{r_i})\}_n, \subset)$ is isomorphic to $(\{\mathbf{R_i}\}_n, \subset)$.

3.3 INTUITIVE MEANING OF THE ISOMORPHISM

At a first glance, it may look paradoxical that a relation whose H_i is the intersection of other relations' H_j's (i.e. is_parent_of of Figure 3a) actually subsumes all the relations that were intersected (i.e. is_fahter_of, is_mohter_of). This inverse ordering of H_j's and Sol($P_j, \overrightarrow{r_j}$) arise from the fact that the two are orthocomplements (Strang).

Geometrically, projection is decomposition into independent directions; $\overrightarrow{x} = P\overrightarrow{x} + (I - P)\overrightarrow{x}$ holds for all $\overrightarrow{x}$. In Fig. 4a, one can see that P and $I - P$ are orthogonal. Algebraically, a vector $\overrightarrow{x} \in \mathbb{R}^d$ bound by $P\overrightarrow{x} = b$, composed of k independent constraints (rank k), $\overrightarrow{x}$ is free in all other $d - k$ directions of $I - P$ (Fig. 4b). Thus, the lesser constraint the space to be projected onto, the

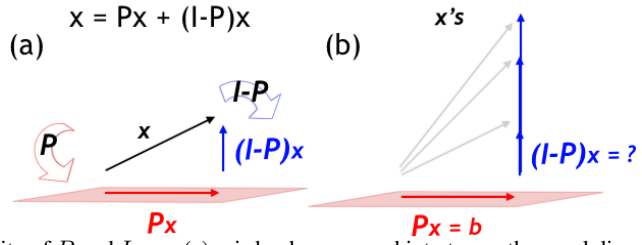

Figure 4: Orthogonality of P and $I-p$. (a) x is be decomposed into two orthogonal directions, when projection is applied. (b) The constraint $Px = b$ does not impose anything to the orthogonal direction; x can have any magnitude in $(I - P)$'s direction.

more freedom a vector is given; which is *isomorphic* to that, for example, is_family_of puts more freedom on who can be tied by it than is_father_of. (Fig. 1b).
Thus, the intuitive meaning of the above proof is that we can map degree of freedom in the logical space to that in $\mathbb{R}^d$.

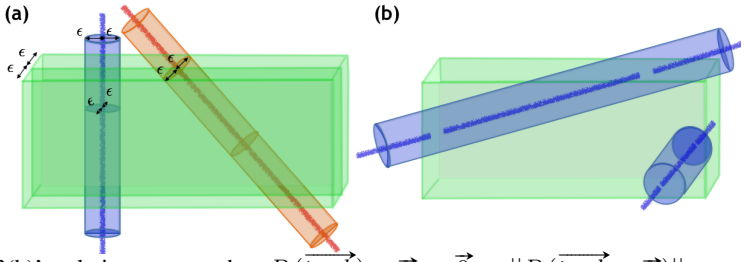

Figure 5: Fig. 3(b)'s relation spaces when $P_i(\overrightarrow{t-h}) - \vec{r_i} \approx \vec{0} \equiv ||P_i(\overrightarrow{t-h} - \vec{r_i})||_2 < \epsilon$ is required. (a): Each relation space now becomes regions with thickness ϵ, centered around figure 3(b)'s relation space. (b): Relationship of the angle and area of overlap between two relation spaces. With respect to the green region, the nearly perpendicular cylinder overlaps much less with it than the other cylinder with much closer angle.

4 Initialization and Training

The intersection and projection constraints can be imposed with parameter sharing. We describe how shared parameters are initialized and trained.

4.1 Parameter Sharing Initializaion

From initialization, we bind parameters so that they satisfy the two constraints. For each entity e_j, we assign a d-dimensional vector $\vec{e_j}$. To each $\mathbf{R}_i$, we assign $(H_i, \vec{r_i})$ (or $(A_i, \vec{r_i})$) with parameter sharing. We first construct the H's.

Intersection constraint We define the H's top-down, first defining the intersections and then the subspaces that go through it. To the head $\mathbf{R}_h$, assign a_h linearly independent rows for the basis of H_h. Then, to each $\mathbf{R}_i$ that is not a head, additionally assign a_i rows linearly independent to the bases of all of its parents, and construct H_i with its bases and the bases of all of its parents. Projection matrices can be uniquely constructed given the bases (Strang).
Now, we initlialize the $\vec{r_i}$'s.

Projection Constraint To the head $\mathbf{R}_h$, pick any random $x_h \in \mathbb{R}^d$ and assign $\vec{r_h} = P_h x$. To each non-head $\mathbf{R}_i$ whose parent is $\mathbf{R}_p$, assign $\vec{r_i} = \vec{r_p} + (I - P_p)(P_i)x_i$ for some random x_i. This results in

$$P_p\vec{r_i} = P_p\vec{r_p} + P_p(I - P_p)(P_i)\vec{x_i} = \vec{r_p} + \vec{0} = \vec{r_p}$$

for any parent, child pair.

Parameters to be trained Such initialization leaves the following parameters given a KG with entities e_j's and relations r_i's: (1) A_h for the head relation, (2) c_i for each non-head relation, (3) $\vec{x_i}$ for each head and non-head relation, (4) $\vec{e_j}$ for each entity e_j.

4.1.1 Training

We construct negative examples (wrong fact triplets) and train with a margin-based loss, following the same protocols as in TransE and TransH.

Training Objective We adopt the same loss function as in TransH. For each fact triplet (h, r_i, t), we define the score function $f(h, r_i, t) = ||P_i(\overrightarrow{t-h}) - \vec{r_i}||_2$
and train a margin-based loss L which is aggregates f's and discriminates between correct and negative examples.

$$L = \sum_{(h,r_i,t)\in G} max(0, f(h, r_i, t)^2 + \gamma - f(h', r_i', t')^2)$$

where G is the set of all triples in the KG and (h', r_i', t') is a negative triple made from corrupting (h, r_i, t). We minimize this objective with stochastic gradient descent.

Automatic Grounding of Positive Triples The parameter sharing scheme guarantees two advantages during all steps of training. First, the intersection and projection constraint are met not only at initialization but always.

Second, traversing through a particular (h, r_i, t) also automatically executes training with (h, r_p, t) for any $r_i \Rightarrow r_p$. For example, by traversing *(Tom, is_father_of, Harry)* in the KG, the model automatically also traverses *(Tom, is_parent_of, Harry), (Tom, is_family_of, Harry)*, even if the two triples are missing in the KG. This is because $P_p P_i = P_p$ with the given initialization (section 4.1.1) and thus,

$$f(h, r_p, t) = ||P_p(\overrightarrow{t-h}) - \vec{r_p}||_2^2 = ||P_p(P_i((\overrightarrow{t-h}) - \vec{r_i}))||_2^2$$
$$\leq ||(P_p + (I - P_p))P_i((\overrightarrow{t-h}) - \vec{r_i}))||_2^2 = ||(P_i((\overrightarrow{t-h}) - \vec{r_i}))||_2^2 = f(h, r_i, t)$$

In other words, training $f(h, r_i, t)$ towards less than ϵ automatically guarantees, or has the effect of training $f(h, r_p, t)$ towards less than ϵ. This enables the model to be automatically trained with what exists in the KG, eliminating the need to manually create missing triples that are true by implication rule.

5 EXPERIMENTS

We evaluate TransINT on Freebase 122 (respectively created by Vendrov et al. and Guo et al.) against the current state-of-the-art method.

5.1 LINK PREDICTION

The task is to predict the gold entity given a fact triple with missing head or tail - if (h, r, t) is a fact triple in the test set, predict h given (r, t) or predict t given (h, r). We follow TransE and KALE's protocol. For each test triple (h, r, t), we rank the similarity score ($f(e, r, t)$ when h is replaced with e for every entity e in the KG, and identify the rank of the gold head entity h; we do the same for the tail entity t. Aggregated over all test triples, we report: (i) the mean reciprocal rank (**MRR**), (ii) the median of the ranks (**MED**), and (iii) the proportion of ranks no larger than n (**HITS@N**), which are the same metrics reported by KALE. A lower MED and higher MRR and Hits HITS@N are better.

TransH and KALE adopt a "filtered" setting that addresses when entities that are correct, albeit not gold, are ranked before the gold entity. For example, if the gold entity is *(Tom, is_parent_of, John)* and we rank every entity e for being the head of *(?, is_parent_of, John)*, it is possible that *Sue*, *John*'s mother, gets ranked before *Tom*. To avoid this, the "filtered setting" ignore corrupted triplets that exist in the KG when counting the rank of the gold entity. (The setting without this is called the "raw setting").

We compare our performance with that of KALE and previous methods (TransE, TransH, TransR) that were compared against it, using the same dataset (FB122). FB122 is a subset of FB15K (Bordes et al.) accompanied by 47 implication and transitive rules; it consists of 122 Freebase relations on "people", "location", and "sports" topics. Since we use the same train/ test/ validation sets, we directly copy from Guo et al. for reporting on these baselines.

5.1.1 DETAILS OF TRAINING

TransINT's hyperparameters are: learning rate (η), margin (γ), embedding dimension (d), and learning rate decay (α), applied every 10 epochs to the learning rate. We find optimal configurations among the following candidates: $\eta \in \{0.003, 0.005, 0.01\}, \gamma \in \{1, 2, 5, 10\}, d \in \{50, 100\}, \alpha, \in \{1.0, 0.98, 0.95\}$. We create 100 mini-batches of the training set and train for maximum of 1000 epochs with early stopping based on the best median rank. Furthermore, we try training with and without normalizing each of entity vectors, relation vectors, and relation subspace bases after every batch of training.

5.1.2 EXPERIMENT SETTINGS

Out of the 47 rules in FB122, 9 are transitive rules (such as `person/nationality(x,y)` $\wedge$ `country/official_language(y,z)` $\Rightarrow$ `person/languages(x,z)`) to be used for KALE. However, since TransINT only deals with implication rules, we do not take advantage of them, unlike KALE.

We also put us on some intentional disadvantages against KALE to assess TransINT's robustness to absence of negative example grounding. In constructing negative examples for the margin-based loss L, KALE both uses rules (by grounding) and their own scoring scheme to avoid false negatives. While grounding with FB122 is not a burdensome task, it known to be very inefficient and difficult for extremely large datasets (Ding et al.). Thus, it is a great advantage for a KG model to perform well without grounding of training/ test data. We evaluate TransINT on two settings for avoiding false negative examples; using rule grounding and only avoiding ones that exist in the KG. We call them respectively TransINT^G (grounding), TransINT^{NG} (no grounding).

Table 1: Results on Link Prediction on FB122. *: For KALE, we report the best performance by any of KALE-PRE, KALE-Joint, KALE-TRIP (three variants of KALE proposed by Guo et al.).

	Raw					**Filtered**				
	MRR	MED	Hits N%			MRR	MED	Hits N%		
			3	5	10			3	5	10
TransE	0.262	10.0	33.6	42.5	50.0	0.480	2.0	58.9	64.2	70.2
TransH	0.249	12.0	31.9	40.7	48.6	0.460	3.0	53.7	59.1	66.0
TransR	0.261	15.0	28.9	37.4	45.9	0.523	2.0	59.9	65.2	71.8
KALE*	0.294	9.0	36.9	44.8	51.9	0.523	2.0	61.7	66.4	72.8
TransINT^G	**0.339**	**6.0**	**40.1**	**49.1**	**54.6**	**0.655**	**1.0**	**70.4**	**75.1**	**78.7**
TransINT^{NG}	0.323	8.0	38.3	46.6	53.8	0.620	1.0	70.1	74.1	78.3

Table 2: Results on Triple Classification on FB122, in Mean Average Precision (MAP).

TransE	**TransH**	**TransR**	**KALE***	**TransINT**^{1}	**TransINT**^{2}
0.634	0.641	0.619	0.677	**0.781** (0.839/ 0.752)	**0.743** (0.709/ 0.761)

5.1.3 RESULTS

We report link prediction results in Table 1. While the *filtered* setting gives better performance (as expected), the trend is generally similar between *raw* and *filtered*. TransINT outperforms all other models by large margins in all metrics, even without grounding; especially in the *filtered* setting, the **Hits@N** gap between TransINT^G and KALE is around 4$\sim$6 times that between KALE and the best performing Trans Baseline (TransR).

Also, while TransINT^G performs higher than TransINT^{NG} in all settings/metrics, the gap between them is much smaller than the that between TransINT^{NG} and KALE, showing that TransINT robustly brings state-of-the-art performance even without grounding. The results suggest two possibilities in a more general sense. First, the emphasis of true positives could be as important as/ more important than avoiding false negatives. Even without manual grounding, TransINT^{NG} has automatic grounding of positive training instances enabled (Section 4.1.1.) due to model properties, and this could be one of its success factors. Second, hard constraint on parameter structures can bring performance boost uncomparable to that by regularization or joint learning, which are softer constraints. We also note that norm regularization of any parameter did not help in training TransINT, unlike stated in TransE, TransH, and KALE. Instead, it was important to use a large margin (either $\gamma = 5$ or $\gamma = 10$).

5.2 TRIPLE CLASSIFICATION

The task is to classify whether an unobserved instance (h, r, t) is correct or not, where the test set consists of positive and negative instances. We use the same protocol and test set provided by KALE; for each test instance, we evaluate its similarity score $f(h, r, t)$ and classify it as "correct" if $f(h, r, t)$ is below a certain threshold (σ), a hyperparameter to be additionally tuned for this task. We report on mean average precision (MAP), the mean of classification precision over all distinct relations (r's) of the test instances. We use the same experiment settings/ training details as in Link Prediction other than additionally finding optimal σ.

5.2.1 RESULTS

Triple Classification results are shown in Table 2. Again, TransINT^G and TransINT^{NG} both significantly outperform all other baselines. We also separately analyze MAP for relations that are/ are not affected by the implication rules (those that appear/ do not appear in the rules), shown in parentheses of Table 2 with the order of (influenced relations/ uninfluenced relations). We can see that both TransINT's have MAP higher than the overall MAP of KALE, even when the TransINT's have the penalty of being evaluated only on uninfluenced relations; this shows that TransINT generates better embeddings even for those not affected by rules. Furthermore, we comment on the role of negative example grounding; we can see that grounding does not help performance on unaffected relations (i.e. 0.752 vs 0.761), but greatly boosts performance on those affected by rules (0.839 vs 0.709). While TransINT does not necessitate negative example grounding, it does improve the quality of embeddings for those affected by rules.

6 SEMANTICS MINING WITH OVERLAP BETWEEN EMBEDDED REGIONS

Traditional embedding methods that map an object (i.e. words, images) to a singleton vector learn soft tendencies between embedded vectors, such as semantic similarity (Mikolov et al., Frome et al.).

Table 3: Examples of relations' angles and *imb* with respect to `/people/person/place_of_birth`

		Relation	Anlge	*imb*
Not Disjoint	Relatedness	`/people/person/nationality`	22.7	1.18
	Implication	`/people/person/place_lived/location`*	46.7	3.77
Disjoint		`/people/cause_of_death/people`	76.6	n/a
		`/sports/sports_team/colors`	83.5	n/a

A common metric for such tendency is cosine similarity, or angle between two embddings. TransINT extends such line of thought to semantic relatedness between groups of objects, with angles between relation spaces. In Fig. 5b, one can observe that the closer the angle between two embedded regions, the larger the overlap in area. For entities h and t to be tied by both relations r_1, r_2, $t - h$ has to belong to the intersection of their relation spaces. Thus, we hypothesize the following over any two relations r_1, r_2 that are not explicitly tied by the pre-determined rules:
Let V_1 be the set of $\overrightarrow{t-h}$'s in r_1's relation space (denoted as Rel_1) and V_2 that of r_2's. Then,

(1) Angle between Rel_1 and Rel_2 represents semantic "disjointness" of r_1, r_2; the more disjoint two relations, the closer their angle to 90°.
When the angle between Rel_1 and Rel_2 is small,
(2) if majority of V_1 belongs to the overlap of V_1 and V_2 but not vice versa, r_1 implies r_2.
(3) if majority of V_1 and V_2 both belong to their overlap, r_1 and r_2 are semantically related.

Hypotheses (2) and (3) consider the imbalance of membership in overlapped regions. Exact calculation of this involves specifying an appropriate ϵ (Fig. 3). As a proxy for deciding whether an element of V_1 (denote v_1) belongs in the overlapped region, we can consider the distance between v_1 to and its projection to Rel_2; the further away v_1 is from the overlapped region, the larger the projected distance (visualization available in our code repository). We call the mean of such distances from V_1 to Rel_2 d_{12} and the reverse d_{21}. We quantify the imbalance in d_{12}, d_{21} with $\frac{1}{2}(\frac{d_{12}}{d_{21}} + \frac{d_{21}}{d_{12}})$, which is minimized to 1 when $d_{21} = d_{12}$ and increases as d_{12}, d_{21} are more imbalanced. We call this factor imb.

For hypothesis (1), we verified that the vast majority of relation pairs have angles near to 90°, with the mean and median respectively 83.0° and 85.4°; only 1% of all relation pairs had angles less than 50°. We observed that relation pairs with angle less than 20° were those that can be inferred by transitively applying the pre-determined implication rules. Relation pairs with angles within the range of $[20°, 60°]$ had strong tendencies of semantic relatedness or implication; such tendency drastically weakened past 70°. Table 3 shows the angle and imb of relations with respect to `/people/person/place_of_birth`, whose trend agrees with our hypotheses. While we only show a subset of the complete list, we note that almost all relation pairs generally follow such a tendency; the complete list can be accessed in our code repository. Finally, we note that such analysis could be possible with TransH as well, since their method too maps $\overrightarrow{t-h}$'s to lines (Fig. 2b).

Throughout target tasks (Link Prediction, Triple Classification) and semantics mining, TransINT's theme of optimal regions to bound entity sets is unified and consistent. Furthermore, the integration of rules into embedding space geometrically coherent with KG embeddings alone. These two qualities were missing in existing works such as TransE or KALE, and TransINT opens up new possibilities for applying KG embeddings to explainable reasoning in applications such as recommender systems (Ma et al.) and question answering (Hamilton et al.).

7 Related Work

Our work is related to three strands of work. The first strand is Order Embeddings (Vendrov et al.) and their extensions (Vilnis et al.; Athiwaratkun & Wilson), whose limitation we discussed in the introduction. While Nickel & Kiela also approximately embed unary partial ordering, their focus is on achieving reasonably competent result with unsupervised learning of rules in low dimensions, while ours is achieving state-of-the-art in a supervised setting. The second strand is those that enforce the satisfaction of common sense logical rules in the embedded KG. Wang et al. explicitly constraints the resulting embedding to satisfy logical implications and type constraints via linear programming, but it only requires to do so during inference, not learning. On the other hand,Guo et al. induces that embeddings follow a set of logical rules during learning, but their approach is soft induction not hardly constrain. Our work combines the advantages of both works.

8 Conclusion

We presented TransINT, a new KG embedding method that embed sets of entities (tied by relations) to continuous sets in $\mathbb{R}^d$ that are inclusion-ordered isomorphically to relation implications. Our method achieved new state-of-the-art performances with signficant margins in Link Prediction and Triple Classification on the FB122 dataset, with boosted performance even on test instances that are not affected by rules. We further propose and interpretable criterion for mining semantic similairty among sets of entities with TransINT.

REFERENCES

Ben Athiwaratkun and Andrew Gordon Wilson. Hierarchical density order embeddings, 2018.

Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning structured embeddings of knowledge bases. In *AAAI*, 2011.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pp. 2787–2795, USA, 2013. Curran Associates Inc. URL http://dl.acm.org/citation.cfm?id=2999792.2999923.

Boyang Ding, Quan Wang, Bin Wang, and Li Guo. Improving knowledge graph embedding using simple constraints. In *ACL*, 2018.

Andrea Frome, Gregory S. Corrado, Jonathon Shlens, Samy Bengio, Jeffrey Dean, Marc'Aurelio Ranzato, and Tomas Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. Jointly embedding knowledge graphs and logical rules. In *EMNLP*, 2016.

William L. Hamilton, Payal Bajaj, Marinka Zitnik, Daniel Jurafsky, and Jure Leskovec. Embedding logical queries on knowledge graphs. In *NeurIPS*, 2018.

Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 31*, pp. 4284–4295. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7682-simple-embedding-for-link-prediction-in-knowledge-graphs.pdf.

Weizhi Ma, Min Zhang, Yue Cao, Woojeong Jin, Chenyang Wang, Yiqun Liu, Shaoping Ma, and Xiang Ren. Jointly learning explainable rules for recommendation with knowledge graph. In *WWW*, 2019.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *NIPS*, 2017.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *ICML*, 2011.

Robert Roth Stoll. *Set theory and logic*. Courier Corporation, 1979.

Gilbert Strang. *Linear algebra and its applications*. Thomson, Brooks/Cole, Belmont, CA, 2006. ISBN 0030105676 9780030105678 0534422004 9780534422004. URL http://www.amazon.com/Linear-Algebra-Its-Applications-Edition/dp/0030105676.

Ivan Vendrov, Jamie Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. *CoRR*, abs/1511.06361, 2015.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. Probabilistic embedding of knowledge graphs with box lattice measures. *arXiv preprint arXiv:1805.06627*, 2018.

Quan Wang, Bin Wang, and Li Guo. Knowledge base completion using embeddings and rules. In *IJCAI*, 2015.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *AAAI*, 2014.

Zhuoyu Wei, Jun Zhao, Kang Liu, Zhenyu Qi, Zhengya Sun, and Guanhua Tian. Large-scale knowledge base completion: Inferring via grounding network sampling over selected instances. In *CIKM*, 2015.

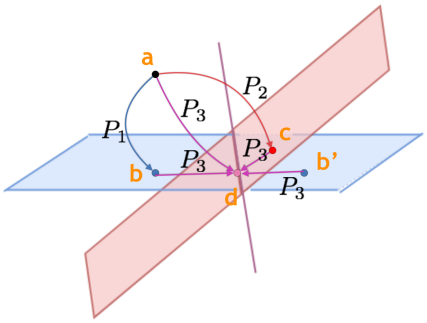

Figure 6: Projection matrices of subspaces that include each other.

A APPENDIX

PROOF FOR TRANSINT'S ISOMORPHIC GUARANTEE

Here, we provide the proofs for Main Theorems 1 and 2. We also explain some concepts necessary in explaining the proofs. We put * next to definitions and theorems we propose/ introduce. Otherwise, we use existing definitions and cite them.

A.1 LINEAR SUBSPACE AND PROJECTION

We explain in detail elements of $\mathbb{R}^d$ that were intuitively discussed. In this and later sections, we mark all lemmas and definitions that we newly introduce with *; those not marked with * are accompanied by reference for proof. We denote all $d \times d$ matrices with capital letters (ex) A) and vectors with arrows on top (ex) $\vec{b}$).

A.1.1 LINEAR SUBSPACE AND RANK

The linear subspace given by $A(x - \vec{b}) = 0$ (A is $d \times d$ matrix and $b \in \mathbb{R}^d$) is the set of $x \in \mathbb{R}^d$ that are solutions to the equation; its rank is the number of constraints $A(x - \vec{b}) = 0$ imposes. For example, in $\mathbb{R}^3$, a hyperplane is a set of $\vec{x} = [\, x_1, x_2, x_3] \in \mathbb{R}^3$ such that $ax_1 + bx_2 + cx_3 - d = 0$ for some scalars a, b, c, d; because vectors are bound by one equation (or its "A" only really contains one effective equation), a hyperplane's rank is 1 (equivalently $rank(A) = 1$). On the other hand, a line in $\mathbb{R}^3$ imposes to 2 constraints, and its rank is 2 (equivalently $rank(A) = 2$).

Consider two linear subspaces H_1, H_2, each given by $A_1(\vec{x} - \vec{b_1}) = 0, A_2(\vec{x} - \vec{b_2}) = 0$. Then,

$$(H_1 \subset H_2) \Leftrightarrow (A_1(\vec{x} - \vec{b_1}) = 0 \Rightarrow A_2(\vec{x} - \vec{b_2}) = 0)$$

by definition. In the rest of the paper, denote H_i as the linear subspace given by some $A_i(\vec{x} - \vec{b_i}) = 0$.

A.1.2 PROPERTIES OF PROJECTION

Invariance For all $\vec{x}$ on H, projecting $\vec{x}$ onto H is still $\vec{x}$; the converse is also true.

Lemma 1 *$P\vec{x} = \vec{x} \Leftrightarrow \vec{x} \in H$ (Strang).*

Orthogonality *Projection decomposes any vector $\vec{x}$ to two orthogonal components - $P\vec{x}$ and $(I - P)\vec{x}$ (Figure 4). Thus, for any projection matrix P, $I - P$ is also a projection matrix that is orthogonal to P (i.e. $P(I - P) = 0$) (Strang).*

Lemma 2 *Let P be a projection matrix. Then $I - P$ is also a projection matrix such that $P(I - P) = 0$ (Strang).*

The following lemma also follows.

Lemma 3 *$||P\vec{x}|| \leq ||P\vec{x} + (I - P)\vec{x}|| = ||\vec{x}||$ (Strang).*

Projection onto an included space *If one subspace H_1 includes H_2, the order of projecting a point onto them does not matter. For example, in Figure 3, a random point $\vec{a}$ in R^3 can be first projected onto H_1 at $\vec{b}$, and then onto H_3 at $\vec{d}$. On the other hand, it can be first projected onto H_3 at $\vec{d}$, and then onto H_1 at still $\vec{d}$. Thus, the order of applying projections onto spaces that includes one another does not matter.*

If we generalize, we obtain the following two lemmas (Figure 6):

Lemma 4 Every two subspaces $H_1 \subset H_2$ if and only if $P_1P_2 = P_2P_1 = P_1$.*

***proof)** By Lemma 1, if $H_1 \subset H_2$, then $P_2\vec{x} = \vec{x} \quad \forall \vec{x} \in H_1$. On the other hand, if $H_1 \not\subset H_2$, then there is some $\vec{x} \in H_1, \vec{x} \notin H_2$ such that $P_2\vec{x} \neq \vec{x}$. Thus,*

$$\begin{aligned} H_1 \subset H_2 &\Leftrightarrow \forall \vec{x} \in H_1, \quad P_2\vec{x} = \vec{x} \\ &\Leftrightarrow \forall \vec{y}, \quad P_2(P_1\vec{y}) = P_1\vec{y} \Leftrightarrow P_2P_1 = P_1. \end{aligned}$$

Because projection matrices are symmetric (Strang),

$$P_2P_1 = P_1 = {P_1}^T = {P_1}^T{P_2}^T = P_1P_2. \blacksquare$$

Lemma 5 For two subspaces H_1, H_2 and vector $\vec{k} \in H_2$,*

$$H_1 \subset H_2 \Leftrightarrow Sol(P_2, \vec{k}) \subset Sol(P_1, P_1\vec{k}).$$

***proof)** $Sol(P_2, \vec{k}) \subset Sol(P_1, P_1\vec{k})$ is equivlaent to $\forall \vec{x} \in \mathbb{R}^d, P_2\vec{x} = \vec{k} \Rightarrow P_1\vec{x} = P_1\vec{k}$.*

By Lemma 4, if $H_1 \subset H_2 \Leftrightarrow P_1P_2 = P1$. Since $\vec{k} \in P_2$, $P_2\vec{x} = \vec{k} \Leftrightarrow P_2(x - \vec{k}) = \vec{0} \Leftrightarrow P_1(P_2\vec{x} - \vec{k}) = \vec{0} \Leftrightarrow P_1P_2\vec{x} = P_1\vec{k} \Leftrightarrow P_1\vec{x} = P_1\vec{k}$.$\blacksquare$

Partial ordering *If two subspaces strictly include one another, projection is uniquely defined from lower rank subspace to higher rank subspace, but not the other way around. For example, in Figure 3, a point $\vec{a}$ in R^3 (rank 0) is always projected onto H_1 (rank 1) at point $\vec{b}$. Similarly, point $\vec{b}$ on H_1 (rank 1) is always projected onto similarly, onto H_3 (order 2) at point d. However, "inverse projection" from H_3 to H_1 is not defined, because not only $\vec{b}$ but other points on H_1 (such as $\vec{b'}$) project to H_3 at point $\vec{d}$; these points belong to $Sol(P_3, \vec{d})$. In other words, $Sol(P_1, \vec{b}) \subset Sol(P_3, \vec{d})$. This is the key intuition for isomorphism , which we prove in the next chapter.*

A.2 Proof for Isomorphism

Now, we prove that TransINT's two constraints (section 2.3) guarantee isomorphic ordering in the embedding space.

Two posets are isomorphic if their sizes are the same and there exists an order-preseving mapping between them. Thus, any two posets $(\{A_i\}_n, \subset)$, $(\{B_i\}_n, \subset)$ are isomorphic if $|\{A_i\}_n| = |\{B_i\}_n|$ and

$$\forall i, j \quad A_i \subset A_j \Leftrightarrow B_i \subset B_j$$

***Main Theorem 1** (Isomorphism): Let $\{(H_i, \vec{r_i})\}_n$ be the (subspace, vector) embeddings assigned to relations $\{\mathbf{R_i}\}_n$ by the Intersection Constraint and the Projection Constraint; P_i the projection matrix of H_i. Then, $(\{Sol(P_i, \vec{r_i})\}_n, \subset)$ is isomorphic to $(\{\mathbf{R_i}\}_n, \subset)$.*

***proof)** Since each $Sol(P_i, \vec{r_i})$ is distinct and each $\mathbf{R_i}$ is assigned exactly one $Sol(P_i, \vec{r_i})$, $|\{Sol(P_i, \vec{r_i})\}_n| = |\{I_i\}_n|$.①*

Now, let's show

$$\forall i, j, \quad R_i \subset R_j \Leftrightarrow Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j}).$$

Because the $\forall i, j$, intersection and projection constraints are true iff $R_i \subset R_j$, enough to show that the two constraints hold iff $Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j}$.

First, let's show $\mathbf{R_i} \subset \mathbf{R_i} \Rightarrow Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j})$. From the Intersection Constraint, $\mathbf{R_i} \subset \mathbf{R_i} \Rightarrow H_j \subset H_i$. By Lemma 5, $Sol(P_i, \vec{r_i}) \subset Sol(P_j, P_j\vec{r_i})$. From the Projection Constraint, $\vec{r_j} = P_j\vec{r_i}$. Thus, $Sol(P_i, \vec{r_i}) \subset Sol(P_j, P_j\vec{r_i}) = Sol(P_j, \vec{r_j}). \cdots\cdots$ ②

Now, let's show the converse; enough to show that if $Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j})$, then the intersection and projection constraints hold true.

$$Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j})$$
$$\Leftrightarrow \forall \vec{x}, \quad P_i \vec{x} = \vec{r_i} \Rightarrow P_j \vec{x} = \vec{r_j})$$

If $P_i \vec{x} = \vec{r_i}$,

$$\forall \vec{x}, \quad P_j P_i \vec{x} = P_j \vec{r_i}$$
$$\forall \vec{x}, \quad P_j \vec{x} = \vec{r_j}$$

both have to be true. For any $\vec{x} \in H_i$, or equivalently, if $\vec{x} = P_i \vec{y}$ for some $\vec{y}$, then the second equation becomes $\forall \vec{y}, \quad P_j P_i \vec{y} = \vec{r_j}$, which can be only compatible with the first equation if $\vec{r_j} = P_j \vec{r_i}$, since any vector's projection onto a subspace is unique. (Projection Constraint)

Now that we know $\vec{r_j} = P_j \vec{r_i}$, by Lemma 5, $H_i \subset H_j$ (intersection constraint).$\cdots$ ③ From ①, ②, ③, the two posets are isomorphic.■

In actual implementation and training, TransINT requires something less strict than $P_i(\overrightarrow{t-h}) = \vec{r_i}$:

$$P_i(\overrightarrow{t-h}) - \vec{r_i} \approx \vec{0} \equiv ||P_i(\overrightarrow{t-h} - \vec{r_i})||_2 < \epsilon,$$

for some non-negative and small ϵ. This bounds $\overrightarrow{t-h} - \vec{r_i}$ to regions with thickness 2ϵ, centered around $Sol(P_i, \vec{r_i})$ (Figure 5). We prove that isomorphism still holds with this weaker requirement.

Definition* $(Sol_\epsilon(P, k))$: *Given a projection matrix P, the solution space of $||P\vec{x} - \vec{k}||_2 < \epsilon$ is denoted as $\mathbf{Sol}_\epsilon(\mathbf{P}, \vec{k})$.*

***Main Theorem 2** (Margin-aware Isomorphism): For all non-negative scalar ϵ, $(\{Sol_\epsilon(P_i, \vec{r_i})\}_n, \subset)$ is isomorphic to $(\{\mathbf{R_i}\}_n, \subset)$.*

***proof)** Enough to show that $(\{Sol_\epsilon(P_i, \vec{r_i})\}_n, \subset)$ and $(\{Sol(P_i, \vec{r_i})\}_n, \subset)$ are isomorphic for all ϵ.*

First, let's show

$$Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j}) \Rightarrow Sol_\epsilon(P_i, \vec{r_i}) \subset Sol_\epsilon(P_j, \vec{r_j}).$$

By Main Theorem 1 and Lemma 4,

$$Sol(P_i, \vec{r_i}) \subset Sol(P_j, \vec{r_j}) \Leftrightarrow \vec{r_j} = P_j \vec{r_i}, P_j = P_j P_i.$$

Thus, for all vector $\vec{b}$,

$$P_i(x - \vec{r_i}) = \vec{b}$$
$$\Leftrightarrow P_j P_i(\vec{x} - \vec{r_i}) = P_j \vec{b}$$
$$\Leftrightarrow P_j(\vec{x} - \vec{r_i}) = P_j \vec{b} (\because \text{Lemma 4})$$
$$\Leftrightarrow P_j(\vec{x} - \vec{r_j}) = P_j \vec{b} (\because P_j \vec{r_j} = \vec{r_j} = P_j \vec{r_i})$$

Thus, if $||P_i(\vec{x} - \vec{r_i})|| < \epsilon$, then $||P_j(\vec{x} - \vec{r_j})|| = ||P_j(P_i(\vec{x} - \vec{r_i}))|| < ||P_j(P_i(\vec{x} - \vec{r_i})) + (I - P)(P_i(\vec{x} - \vec{r_i}))|| = ||P_i(\vec{x} - \vec{r_i})|| < \epsilon. \cdots$ ①

Now, let's show the converse. Assume $||P_i(\vec{x} - \vec{r_i})|| < \epsilon$ for some i. Then,

Thus, for $||P_i(\vec{x} - \vec{r_i})|| < \epsilon$ to bound $||P_j(\vec{x} - \vec{r_j})||$ at all for all $\vec{x}$,

$$P_j(I - P_i) = 0, P_j(\vec{r_i} - \vec{r_j}) = 0$$

need to hold. By Lemma 4 and 5,

$$P_j = P_j P_i \Leftrightarrow H_j \subset H_i$$
$$\Leftrightarrow Sol(P_i, \vec{r_i}) \subset Sol(P_j, P_j \vec{r_i}) = Sol(P_j, \vec{r_j}) \cdot \cdot ②$$

$|\{Sol_\epsilon(P_i, \vec{r_i})\}_n| = |\{Sol(P_i, \vec{r_i})\}_n|$ holds obviously; each $Sol(P_i, \vec{r_i})$ has a distinct $Sol_\epsilon(P_i, \vec{r_i})$ and each $Sol_\epsilon(P_i, \vec{r_i})$ also has a distinct "center" ($Sol(P_i, \vec{r_i})) \cdot \cdot$ ③

From ①, ②, ③, the two sets are isomorphic. ■