ON UNDERSTANDING KNOWLEDGE GRAPH REPRESENTATION

Carl Allen*, Ivana Balažević*, Timothy M. Hospedales
School of Informatics, University of Edinburgh
{carl.allen, ivana.balazevic, t.hospedales}@ed.ac.uk

ABSTRACT

Many methods have been developed to represent knowledge graph data, which implicitly exploit low-rank latent structure in the data to encode known information and enable unknown facts to be inferred. To predict whether a relationship holds between entities, their embeddings are typically compared in the latent space following a relation-specific mapping. Whilst link prediction has steadily improved, the latent structure, and hence why such models capture semantic information, remains unexplained. We build on recent theoretical interpretation of word embeddings as a basis to consider an explicit structure for representations of relations between entities. For identifiable relation types, we are able to predict properties and justify the relative performance of leading knowledge graph representation methods, including their often overlooked ability to make independent predictions.

1 INTRODUCTION

Knowledge graphs are large repositories of binary relations between words (or entities) in the form of fact triples (subject, relation, object). Many models have been developed for learning representations of entities and relations in knowledge graphs, such that known facts can be recalled and previously unknown facts can be inferred, a task known as link prediction. Recent link prediction models (e.g. Bordes et al., 2013; Trouillon et al., 2016; Balažević et al., 2019b) learn entity representations, or embeddings, of far lower dimensionality than the number of entities, by capturing latent structure in the data. Relations are typically represented as a mapping from the embedding of a subject entity to its related object entity embedding(s). Although knowledge graph representation has advanced in terms of link prediction performance for nearly a decade, relatively little is understood of the latent structure that underpins low-rank models, which we seek to address in this work.

We start by drawing a parallel between entity embeddings in knowledge graphs and unsupervised word embeddings, as learned by algorithms such as Word2Vec (W2V) (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). We assume that words have latent features, e.g. meaning(s), tense, grammatical type, that are innate and fixed, irrespective of what an embedding may capture (which may be only a part, subject to the embedding method and/or the data source); and that this same latent structure gives rise to patterns observed in the data, e.g. in word co-occurrence statistics and in which words are related to which. As such, an understanding of the latent structure from one embedding task (e.g. word embedding) might be useful to another (e.g. knowledge graph entity embedding).

Recent work (Allen & Hospedales, 2019; Allen et al., 2019) theoretically explains how semantic properties are encoded in word embeddings that (approximately) factorise a matrix of word co-occurrence pointwise mutual information (PMI), e.g. as is known for W2V (Levy & Goldberg, 2014). Semantic relationships between words (specifically similarity, relatedness, paraphrase and analogy) are proven to manifest as linear relationships between rows of the PMI matrix (subject to known error terms), of which word embeddings can be considered low-rank projections. This explains why similar words (e.g. synonyms) have similar embeddings; and embeddings of analogous word pairs share a common “vector offset”.

Importantly to our work, this insight provides a basis to identify geometric embedding relationships necessary for other semantic relations to hold, such as those of knowledge graphs. Such geometric
Almost all recent knowledge graph models represent entities typically use Euclidean distance and contain a relation-specific additive models (possibly transformed) subject to a (possibly transformed) object entity embedding. A generic additive RTuckER, each relation-specific score function is given by \( R_{e} \), \( e_{s} \), \( e_{o} \) are the entity biases.

### Table 1: Score functions of representative linear link prediction models.

| Model       | Linear Subcategory               | Score Function                                                                 |
|-------------|----------------------------------|--------------------------------------------------------------------------------|
| TransE      | additive                         | \(-\|e_{s} + r - e_{o}\|_{2}^{2}\)                                                    |
| DistMult    | multiplicative (diagonal)        | \( e_{s} \) \( R_{e} \)                                                         |
| TuckER      | multiplicative (diagonal)        | \( W \times_{1} e_{s} \times_{2} r \times_{3} e_{o} \)                           |
| MuRE        | multiplicative (diagonal) + additive | \(-\|R_{e}r + e_{s} - e_{o}\|_{2}^{2} + b_{s} + b_{o}\)                        |

relationships, or relation requirements, implicitly describe relation-specific mappings between entity embeddings, i.e. the essence of relation representation. We find that relations of popular knowledge graphs (e.g. WordNet [Miller 1995], NELL [Carlson et al. 2010]) fall into a hierarchy of 3 relation types: highly related, (generalised) specialisation and (generalised) context-shift. Understanding the requirements of a relation representation provides a “blue-print” to consider knowledge graph representation models, which we can use to make several theoretical predictions and justify the relative performance of recent link prediction models for different relation types.

In summary, the key contributions of this work are:

- to use recent understanding of PMI-based word embeddings as a basis for developing new insight into knowledge graph relations and their representation;
- to derive a novel categorisation of relations based on identified relation requirements, i.e. what a relation representation must achieve to map a subject embedding to all related object embeddings;
- to show that the performance of several leading link prediction models on different relations depends on the ability of the model’s architecture to represent that type of relation;
- to theoretically predict identifiable properties of knowledge graph models, e.g. that the strength of a relation’s relatedness is reflected in the eigenvalues of its relation matrix; and
- noting how ranking metrics can be flawed, to provide novel insight into recent knowledge graph models from their per-relation prediction accuracy, an evaluation metric we recommend in future.

## 2 Background

Our work draws on knowledge graph representation and word embedding. Whilst related, these tasks differ materially in their training data. The former is restricted to datasets crafted by hand or automatically generated, the latter has the vast abundance of natural language text (e.g. Wikipedia).

### 2.1 Knowledge Graph Representation

Almost all recent knowledge graph models represent entities \( e_{s}, e_{o} \) as vectors \( e_{s}, e_{o} \in \mathbb{R}^{d_{e}} \) of low dimension (e.g. \( d_{e} = 200 \)) relative to the number of entities \( n_{e} \) (typically of order \( 10^{4} \)), and relations as transformations in the latent space from subject entity embedding to object. These models are distinguished by their score function, which defines (i) the form of the relation transformation, e.g. matrix multiplication, vector addition; and (ii) how “closeness” between the transformed subject embedding and an object embedding is evaluated, e.g. dot product, Euclidean distance. Score functions can be non-linear (e.g. [Defmers et al. 2018]), or linear and sub-categorised as additive, multiplicative or both. We focus on linear models due to their simplicity and strong performance at link prediction (including state-of-the-art). Table 1 shows the score functions of competitive linear models that span all linear sub-categories: TransE [Bordes et al. 2013], DistMult [Yang et al. 2015], TuckER [Balažević et al. 2019b] and MuRE [Balažević et al. 2019a].

### Additive models

typically use Euclidean distance and contain a relation-specific translation from a (possibly transformed) subject to a (possibly transformed) object entity embedding. A generic additive score function is given by \( \phi(e_{s}, r, e_{o}) = -\|R_{s}e_{s} + r - R_{o}e_{o}\|_{2}^{2} + b_{s} + b_{o} = -\|e_{s}^{(r)} + r - e_{o}^{(r)}\|_{2}^{2} + b_{s} + b_{o} \). The simplest example is TransE for which \( R_{s} = R_{o} = I \) and \( b_{s} = b_{o} = 0 \). The score function of MuRE has \( R_{s} = I \) and so combines multiplicative (\( R_{o} = R \)) and additive (\( r \)) components.

### Multiplicative models

have the generic score function \( \phi(e_{s}, r, e_{o}) = e_{s}^{\top} R e_{r} = (e_{s}^{(r)}, e_{o}) \), i.e. a bilinear product of the entity embeddings and a relation-specific matrix \( R \). DistMult is a simple example with diagonal \( R \) and so cannot model asymmetric relations ([Trouillon et al. 2016]). In TuckER, each relation-specific \( R = W \times_{3} r \) is a linear combination of \( d_{e} \) “prototype” relation matrices in a core tensor \( W \in \mathbb{R}^{d_{e} \times d_{e} \times d_{e}} \) (where \( \times_{n} \) denotes tensor product along mode \( n \)), facilitating multi-task learning across relations.
2.2 Word Embedding

Algorithms such as Word2Vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014) generate succinct low-rank word embeddings that perform well on downstream tasks (Baroni et al., 2014). Such models predict the context words ($c_j$) observed around each target word ($w_i$) in a text corpus using shallow neural networks. Whilst recent language models (e.g., Devlin et al. (2018); Peters et al. (2018)) create impressive context-specific word embeddings, we focus on the former embeddings since knowledge graph entities have no obvious context and, more importantly, they are interpretable. Levy & Goldberg (2014) show that, for a dictionary of unique words $\mathbb{D}$ and embedding dimension $d \ll |\mathbb{D}|$, W2V’s loss function is minimised when its weight matrices $W, C \in \mathbb{R}^{d \times |\mathbb{D}|}$ (whose columns are word embeddings $w_i, c_j$) factorise a word co-occurrence pointwise mutual information (PMI) matrix, subject to a shift term ($\text{PMI}(w_i, c_j) = \log \frac{P(w_i, c_j)}{P(w_i)P(c_j)}$). This connects W2V to earlier context-based embeddings and specifically to PMI, which has a long history in linguistic analysis (Turney & Pantel, 2010). From its loss function, GloVe can be seen to perform a comparable factorisation.

Recent work shows why word embeddings that factorise such PMI matrix encode semantic word relationships (Allen & Hospedales, 2019; Allen et al., 2019). The authors show that word embeddings can be seen as low-rank projections of high dimensional PMI vectors (rows of the PMI matrix), between which the semantic relationships of similarity, relatedness, paraphrase and analogy provably manifest as linear geometric relationships (subject to defined error terms), which are then preserved, under a sufficiently linear projection, between word embeddings. Thus similar words are known to have similar embeddings, and the embeddings of analogous word pairs share a common vector offset.

Specifically, the PMI vectors ($p_x$, for word $x$) of an analogy “man is to king as woman is to queen” satisfy $p_y \sim p_w \approx p_k - p_m$ because the difference between words associated with king and man (e.g. reign, crown) mirrors that between queen and woman. This leads to a common difference between their co-occurrence distributions (over all words), giving a common difference between their PMI vectors, which projects to a common difference between embeddings. Any discrepancy in the mirroring of word associations is shown to introduce error, weakening the analogy, as does a lack of statistical independence within certain word pairs (see (Allen & Hospedales, 2019)). The common difference in word co-occurrence distributions can be interpreted semantically as a common change in context (context-shift), e.g. the increased association with words {reign, crown, etc.}, that transforms man to king and woman to queen, can be seen to add a royal context. Under this interpretation, context can also be subtracted, e.g. “king is to man as queen is to woman” (minus royal); or both, e.g. “boy is to king as girl is to queen” (minus youth plus royal). Adding context can also be interpreted as specialisation, and subtracting context as generalisation. This establishes a correspondence between common word embedding vector offsets and semantic context-shifts.

Although the projection from PMI vectors to word embeddings preserves the relative relationships, and thus the above semantic interpretability of common embedding differences, a direct interpretation of dimensions themselves is obscured, not least because any embedding matrix can be arbitrarily scaled/rotated if the other is inversely transformed.

3 Relationships between Embeddings of Related Words

Our aim is to build on the understanding of PMI-based word embeddings (henceforth word embeddings), to identify what a knowledge graph relation representation needs to achieve to map all subject word embeddings to all related object word embeddings. We note that if a semantic relation between two words implies a particular geometric relationship between their embeddings, then the latter serves as a necessary quantitative condition for the former to hold (a relation condition). Relation conditions implicitly define a relation-specific mapping by which all subject embeddings are mapped to all related object embedding(s), allowing related entities to be identified by a proximity measure (e.g. Euclidean distance or dot product). Since this is the approach of many knowledge graph representation models, their performance can be contrasted with their ability to express mappings that satisfy required relation conditions. For example, similarity and context change relations respectively imply closeness and a relation-specific vector offset between embeddings (S2.2). Such relation conditions can be tested for by respectively making no change to the subject entity or adding the relation-specific offset, before measuring proximity with the object. We note that since relation conditions are not necessarily sufficient, they do not guarantee a relation holds, i.e. false positives may arise.
Analysing the relations of popular knowledge graph datasets with the above perspective, we find with. Thus their PMI vectors (Fig 1a) and (generalised) specialisation (Turney, 2006; Gladkova et al., 2016). More specifically, the relations within analogies that give a variable relation than similarity, its limiting case with $S$ might be expected to reflect relatedness strength. In general, relatedness is a weaker, more variable relation than similarity, its limiting case with $S = \emptyset$ and $\text{rank}(V_S) = d$.

- **Similarity**: Semantically similar words induce similar distributions over the words they co-occur with. Thus their PMI vectors (Fig 1a) and word embeddings are similar.
- **Relatedness**: The relatedness of two words can be defined in terms of the words with which both co-occur similarly ($S \subseteq \mathbb{D}$), which define the nature of relatedness, e.g. milk and cheese are related by $S = \{\text{dairy, breakfast, ...}\}$; and $|S|$ reflects the strength of relatedness. Since PMI vector components corresponding to $S$ are similar (Fig 1b), embeddings of “$S$-related” words have similar components in the subspace $V_S$ that spans the projected PMI vector dimensions corresponding to $S$. The rank of $V_S$ might be expected to reflect relatedness strength. In general, relatedness is a weaker, more variable relation than similarity, its limiting case with $S = \emptyset$ and $\text{rank}(V_S) = d$.
- **Context-shift**: In the context of word embeddings, analogy typically refers to relational similarity (Turney, 2006; Gladkova et al., 2016). More specifically, the relations within analogies that give a common vector offset between word embeddings require related words to have a common difference between their distributions of co-occurring words, defined as a context-shifts (see S2.2). These relations are strictly 1-to-1 and include an aspect of relatedness due to the word associations in common (Fig 1d). A specialisation relation is a context-shift in which context is only added (Fig 1c).
- **Generalised context-shift**: Context-shift relations are generalised to 1-to-many, many-to-1 and many-to-many relations by letting the fully-specified added or subtracted context be one of a (relation-specific) context set (Fig 1e), e.g. allowing an entity to be any colour or anything blue. The potential scope and size of each context set means these relations can vary greatly. The limiting case for small context sets has a single context in each, whereby the relation is an explicit context-shift (as above), and the difference between embeddings is a known vector offset. In the limiting case where context sets are large, the added/subtracted context is so loosely defined that, in effect, only the relatedness aspect of the relation and thus only the common subspace component of embeddings is known.

**Link to set theory**: Viewing PMI vectors as sets of word associations and taking intuition from Fig 1, the above relations can be seen to reflect set operations: similarity as set equality; relatedness as equality of a subset; and context-shift as the set difference equalling a relation-specific set. This highlights how the relatedness aspect of a relation reflects features that must be common, and context-shift reflects features that must differ. Whilst this mirrors an intuitive notion of “feature vectors”, we emphasise that this is grounded in the co-occurrence statistics of PMI-based word embeddings.

### 3.1 Categorising Knowledge Graph Relations

Analysing the relations of popular knowledge graph datasets with the above perspective, we find that they comprise (i) a relatedness aspect that reflects a common theme (e.g. both entities are animals or geographic terms); and (ii) specific word associations of the subject and/or object entities. Specifically, relations appear to fall under a hierarchy of three relation types: highly related (R); (generalised) specialisation (S); and (generalised) context-shift (C). As above, “generalised” indicates that any added/subtracted contexts can be from a set. From Fig 1, type R relations can be seen as a special case of S, which, in turn, is a special case of C. Type C is therefore a generalised case of all considered relations. Whilst there are several other ways to classify relations (e.g. by their...
hierarchy, transitivity), by considering relation conditions, we delineate by the required mathematical form of their representation. Table 2 shows a categorisation of the relations of the WN18RR dataset (Dettmers et al., 2018), containing 11 relations between 40,943 entities (a subset of WordNet (Miller, 1995)). An explanation for this category assignment is in Appx. A and that for NELL-995 (Xiong et al., 2017) and FB15k-237 (Toutanova et al., 2015) relations is in Appx. B.

### 3.2 Relations as Mappings of Embeddings

Given the relation conditions of a particular relation type, we can recognise mappings that meet them and thus which loss functions (that evaluate the proximity of mapped entity embeddings, e.g. by dot product or Euclidean distance) are able to identify relations of that type. Reviewing the form of relation representation in recent knowledge graph representation models (Table 1), we can contrast what they are able to express with that which we determine necessary for each relation type.

**R:** To evidence S-relatedness, both entity embeddings $e_s, e_o$ must be projected onto a subspace $\mathbb{V}_S$, where their images are compared. Projection requires multiplication by a matrix $P_r \in \mathbb{R}^{d \times d}$ and cannot be achieved additively, except in the limiting case of similarity, when $P_r = I$ or vector $r \approx 0$ is added. Comparison by dot product gives $(P_r e_s)^	op (P_r e_o) = e_s^	op P_r^	op P_r e_o = e_s^	op M_r e_o$ (for a relation-specific symmetric $M_r = P_r^	op P_r$). Euclidean distance gives $\|P_r e_s - P_r e_o\|^2 = (e_s - e_o)^	op M_r (e_s - e_o) = \|P_r e_s\|^2 - 2 e_s^	op M_r e_o + \|P_r e_o\|^2$.

**S/C:** Evidencing these relations requires a test both for S-relatedness and for relation-entity-specific embeddings component(s) ($v_s^r, v_o^r$). This can be achieved by (i) multiplying both entity embeddings by a relation-specific projection matrix $P_r$ that projects onto the subspace that spans the low-rank projection of dimensions corresponding to $S$, $v_s^r$ and $v_o^r$, (which tests for S-relatedness whilst preserving any entity-specific embedding components); and (ii) adding a relation-specific vector $r = v_s^r - v_o^r$ to the transformed subject entity embeddings. Comparison of the final transformed entity embeddings by dot product equates to $(P_r e_s + r)^	op P_r e_o$; and by Euclidean distance to $\|P_r e_s + r - P_r e_o\|^2 = \|P_r e_s + r\|^2 - 2 (P_r e_s + r)^	op P_r e_o + \|P_r e_o\|^2$ (cf MuRE: $\|Re_s + r - e_o\|^2$).

For each relation type, minimising the Euclidean distance comparison term equates to maximising the dot product comparison term with $l_2$ regularisation of entity-specific terms.

Contrasting this insight with the loss functions of knowledge graph models (Table 1), we make the following predictions: (P1) the ability to learn the representation of a particular relation is expected to reflect the complexity of its type (R>S>C), and whether all relation conditions (e.g. additive or multiplicative interactions) can be met under a given model; (P2) relation matrices for relatedness (type R) relations are highly symmetric; (P3) offset vectors for relatedness relations have low norm; and (P4) as a proxy to the rank of $\mathbb{V}_S$, the eigenvalues of a relation matrix reflect a relation’s strength of relatedness. Specifically, P1 expresses that additive-only models (e.g. TransE) are not suited to identifying the relatedness aspect of relations (except in limiting cases of similarity, requiring a zero vector); and multiplicative-only models (e.g. DistMult) should perform well on type R but are not suited to identifying entity-specific features of type S/C, for which an asymmetric relation matrix may help compensate. Further, the loss function of MuRE closely resembles that identified for type C relations (which generalise all considered relations) and is expected to perform best overall.

---

1. We omit the relation “similar_to” since its instances have no discernible structure, and only 3 occur in the test set, all of which are the inverse of a training example and trivial to predict.
Table 3: Hits@10 per relation on WN18RR.

| Relation Name                  | Type | %   | #    | Khs Max/Avg Path | TransE | MuRE | DistMult | TuckER | MuRE |
|-------------------------------|------|-----|------|------------------|--------|------|----------|--------|------|
| verb_group                    | R    | 1%  | 39   | 0.00            | 0.87   | 0.95 | 0.97     | 0.97   | 0.97 |
| derivationally_related_form   | R    | 34% | 1074 | 0.04            | -      | 0.96 | 0.96     | 0.96   | 0.97 |
| also_see                      | R    | 2%  | 56   | 0.24            | 44.04  | 0.35 | 0.40     | 0.39   | 0.46 |
| instance_hypernym             | S    | 4%  | 122  | 1.00            | 3      | 1.0  | 0.37     | 0.38   | 0.46 |
| synset_domain_topic_of         | C    | 4%  | 114  | 0.99            | 3      | 1.1  | 0.19     | 0.43   | 0.45 |
| member_of_domain_usage        | C    | 1%  | 24   | 1.00            | 2      | 1.0  | 0.42     | 0.48   | 0.50 |
| member_of_domain_region       | C    | 1%  | 26   | 1.00            | 2      | 1.0  | 0.35     | 0.40   | 0.40 |
| member_meronym                | C    | 6%  | 172  | 0.99            | 18     | 4.5  | 0.02     | 0.20   | 0.19 |
| has_part                      | C    | 1%  | 24   | 1.00            | 2      | 1.0  | 0.35     | 0.40   | 0.40 |
| hypernym                      | S    | 8%  | 253  | 1.00            | 10     | 3.9  | 0.04     | 0.38   | 0.30 |
| all                           |      | 100%| 3134 | 0.38            | 0.52   | 0.51 | 0.53     | 0.57   |      |

Table 4: Hits@10 per relation on NELL-995.

| Relation Name                  | Type | %   | #    | Khs Max/Avg Path | TransE | MuRE | DistMult | TuckER | MuRE |
|-------------------------------|------|-----|------|------------------|--------|------|----------|--------|------|
| teamplaysagainstteam          | R    | 2%  | 243  | 0.11            | 10     | 3.5  | 0.76     | 0.84   | 0.89 |
| clothingtowithclothing        | R    | 1%  | 132  | 0.17            | 5      | 2.6  | 0.72     | 0.80   | 0.85 |
| professionistypeofprofession  | S    | 1%  | 143  | 0.99            | 7      | 2.5  | 0.37     | 0.55   | 0.62 |
| animalistypeofanimal          | S    | 1%  | 103  | 0.68            | 9      | 3.1  | 0.50     | 0.56   | 0.64 |
| athleteplaysport              | C    | 1%  | 113  | 1.00            | 1      | 1.0  | 0.54     | 0.58   | 0.60 |
| chemicalistypeofchemical      | S    | 1%  | 115  | 0.53            | 6      | 3.0  | 0.23     | 0.43   | 0.49 |
| itemfoundinroom               | C    | 2%  | 162  | 1.00            | 1      | 1.0  | 0.39     | 0.57   | 0.53 |
| agentcollaborateswithagent    | C    | 1%  | 119  | 0.51            | 14     | 4.7  | 0.44     | 0.58   | 0.64 |
| bodypartcontainsbodypart      | C    | 1%  | 103  | 0.60            | 7      | 3.2  | 0.30     | 0.38   | 0.54 |
| atdate                        | C    | 1%  | 967  | 0.99            | 4      | 1.1  | 0.41     | 0.50   | 0.48 |
| locationlocatedwithinlocation | C    | 2%  | 157  | 0.99            | 6      | 1.9  | 0.35     | 0.37   | 0.46 |
| all                           |      | 100%| 20000| 0.36            | 0.48   | 0.51 | 0.52     | 0.52   |      |

4 Comparing Knowledge Graph Models

We test the predictions made on the basis of word embeddings by comparing the performance of competitive knowledge graph models, TransE, DistMult, TuckER and MuRE (see S2), which entail different forms of relation representation, on all WN18RR relations and a similar number of NELL-995 relations (selected to represent each relation type). Since applying the logistic sigmoid to the score function of TransE does not give a probabilistic interpretation comparable to other models, we include a constrained variant of MuRE with $R_s = R_o = I$, as a proxy to TransE for a fairer comparison. Implementation details are included in Appx. D. For evaluation, we generate $2n_e$ evaluation triples for each test triple (for the number of entities $n_e$) by fixing the subject entity $e_s$ and relation $r$ and replacing the object entity $e_o$ with all possible entities and then keeping $e_o$ and $r$ fixed and varying $e_s$. The obtained scores are ranked to give the standard metric hits@10 (Bordes et al., 2013), i.e. the fraction of times a true triple appears in the top 10 ranked evaluation triples.

4.1 Performance per Relation Type

Tables 3 and 4 report results (hits@10) for each relation and include our relation categorisation and known confounding influences: percentage of relation instances in the training and test sets (approximately equal), number of instances in the test set, Krackhardt hierarchy score (see Appx. E) (Krackhardt, 2014; Balažević et al., 2019a) and maximum and average shortest path between any two related nodes. A further confounding effect due to multi-task learning, which is expected to benefit TuckER on the NELL-995 dataset (200 relations). Note that all models have a comparable number of free parameters.

P1: As predicted, all models tend to perform best at type R relations, with a clear performance gap to other relation types. Also, performance on type S relations appears higher in general than type C. Additive-only models (TransE, MuRE$_I$) perform most poorly on average, in line with prediction since all relation types involve a relatedness component. They achieve their best results on type R relations, where the relation vector can be zero/small. Multiplicative-only DistMult performs well, sometimes best, on type R relations, fitting expectation as it can fully represent those relations and
has no additional parameters that may overfit to noise (which may explain where MuRE performs slightly worse). As expected, MuRE, performs best on average (particularly on WN18RR), and most strongly on S and C type relations that require both multiplicative and additive components. The comparable performance of TuckER on NELL-995 is explained by its ability for multi-task learning.

Other unexpected results also tend to align with confounding factors, e.g. that all models perform poorly on the hypernym relation, despite it having type S and a relative abundance of training data (40% of all instances), might be explained by its hierarchical nature (Khs ≈ 1 and long paths). The same may explain the reduced performance on type R relations also_see and agentcollaborateswithagent. As found previously, none of the models are well suited to modelling hierarchical structures. We also note that the percentage of training instances of a relation does not seem to correlate with its performance, as might typically be expected.

P2/P3: Table 5 shows the symmetry score (∈ [-1, 1]) indicating perfect anti-symmetry to symmetry; see Appx. F for the relation matrix of TuckER and the norm of relation vectors of TransE, MuRE, and MuRE on the WN18RR dataset. As expected, type R relations have high symmetry, whereas both other relation types have lower scores, fitting the expectation that TuckER compensates for having no additive component. All additive models learn relation vectors of a noticeably lower norm for type R relations (where, in the extreme, no additive component is required) than for types S and C.

Table 5: Relation matrix symmetry score [-1,1] for TuckER; and relation vector norm for TransE, MuRE, and MuRE (WN18RR).

| Relation                  | Type | Symmetry Score | Vector Norm |
|---------------------------|------|----------------|-------------|
| verb_group                | R    | 0.52           | 5.65 0.76 0.89 |
| derivationally_related_form | R   | 0.54           | 2.98 0.45 0.69 |
| also_see                  | R    | 0.50           | 7.20 0.97 0.97 |
| instance_hypernym         | S    | 0.13           | 18.26 2.98 1.88 |
| member_of_domain_usage    | C    | 0.10           | 11.24 3.18 1.88 |
| member_of_domain_region   | C    | 0.06           | 12.52 3.07 2.11 |
| synset_domain_topic_of    | C    | 0.12           | 23.29 2.65 1.52 |
| member_meronym            | C    | 0.12           | 4.97 1.91 1.97 |
| has_part                  | C    | 0.13           | 6.44 1.69 1.25 |
| hypernym                  | S    | 0.04           | 9.64 1.53 1.03 |

Figure 2: Eigenvalue magnitudes of relation-specific matrices $R$ for MuRE by relation type (WN18RR).

P4: Fig 2 shows eigenvalue magnitudes (scaled relative to their largest and ordered) of the relation-specific matrices $R$ of MuRE, labelled by relation type. Predicted to reflect strength of a relation’s relatedness, they should be highest for type R relations, as observed. For relation types S and C the profiles are more varied, fitting the expectation that the relatedness of those types has greater variability in both choice and size of $S$, i.e. in the nature and strength of relatedness.

The results above confirm all of our predictions about the ability of each model to represent different relational mappings. Our analysis identifies the best model for each relation type (multiplicative-only DistMult for type R, additive-multiplicative MuRE for types S/C), providing a basis for dataset-dependent model selection. Separately, the per-relation insight into where models perform poorly (e.g. on hierarchical relations) can be used to aid future model design.

4.2 Knowledge graph model predictions

Even though in practice we want to know whether a particular triple is true or false, such independent predictions are not commonly reported or evaluated. Despite many recent link prediction models being able to independently predict the truth of each triple, it is common practice to report ranking-based metrics, e.g. mean reciprocal rank, hits@k, which compare the prediction of a test triple to those of all evaluation triples (see §3). Not only is this computationally costly, the evaluation is flawed if entities are related to more than $k$ others and does not evaluate a model’s ability to independently predict whether “$a$ is related to $b$”. We address this by considering actual model predictions.

Since for each relation there are $n^2$ possible entity-entity relationships, we sub-sample by computing predictions for all $(e_s, r, e_o)$ triples only for each $e_s, r$ pair seen in the test set. We split positive predictions $(\sigma(\phi(e_s, r, e_o)) > 0.5)$ between (i) known truths, either training or test/validation; and (ii) other, the truth of which is not known. Per relation, we then compute accuracy over the true training instances (train) and the true test/validation instances (test); and the average number of other truths predicted per $e_s, r$ pair. Table 6 shows results for MuRE, DistMult, TuckER and MuRE. All
with respect to satisfying the necessary relation conditions. Finally, we evaluate several existing knowledge graph models based on their predictive performance. Given this knowledge, we consider knowledge graph models, since if they are satisfied by the relation representations defined by a model’s architecture, then word embedding parameters are a known solution for its entity embeddings. This way, the encoding of semantic relations is not only understood – it is built into the geometry of entity and relation representation. We use this understanding to categorise knowledge graph relations according to their relation conditions. Our analysis confirms that a model’s ability to represent a specific relation type depends on the limitations imposed by the model architecture.

Many methods for learning low-rank representations of knowledge graphs have been developed, yet little is known about the latent structure they learn. This work focuses on understanding why existing knowledge graph representation models work and identifying their limitations, with the ultimate goal of improving representation. With this in mind, we build on recent understanding of PMI-based word embeddings to establish relation conditions, i.e. what a relation representation needs to achieve to map one word embedding to the other. Those conditions provide a framework to consider knowledge graph models, since if they are satisfied by the relation representations defined by a model’s architecture, then word embedding parameters are a known solution for its entity embeddings. This way, the encoding of semantic relations is not only understood – it is built into the geometry of entity and relation representation. We use this understanding to categorise knowledge graph relations according to their relation conditions. Our analysis confirms that a model’s ability to represent a specific relation type depends on the limitations imposed by the model architecture with respect to satisfying the necessary relation conditions. Finally, we evaluate several existing knowledge graph models based on their predictive performance.

Table 6: Per relation prediction accuracy for MuRE\(_I\) (M\(_I\)), (D)istMult, (T)uckER and (M)uRE (WN18RR).

| Relation Name                  | \#train | \#test | Accuracy (train) | Accuracy (test) | \# Other “True” |
|-------------------------------|---------|--------|-----------------|----------------|-----------------|
| verb_group                    | 15      | 39     | 1.00            | 0.97           | 8.3             |
| derivationally_related_form    | 1714    | 1127   | 1.00            | 0.96           | 8.8             |
| also_see                      | 95      | 61     | 1.00            | 0.64           | 7.9             |
| instance_hypernym             | 52      | 122    | 1.00            | 0.32           | 1.3             |
| member_of_domain_usage        | 545     | 43     | 0.98            | 0.02           | 1.5             |
| member_of_domain_region       | 543     | 42     | 0.88            | 0.02           | 1.0             |
| synset_domain_topic_of        | 13      | 115    | 1.00            | 0.42           | 0.7             |
| member_meronym                | 1402    | 307    | 1.00            | 0.22           | 7.9             |
| has_part                      | 848     | 196    | 1.00            | 0.15           | 3.7             |
| hypernym                      | 57      | 1254   | 1.00            | 0.15           | 3.7             |

| Relation Name                  | \#train | \#test | Accuracy (train) | Accuracy (test) | \# Other “True” |
|-------------------------------|---------|--------|-----------------|----------------|-----------------|
| verb_group                    | 15      | 39     | 1.00            | 0.97           | 8.3             |
| derivationally_related_form    | 1714    | 1127   | 1.00            | 0.96           | 8.8             |
| also_see                      | 95      | 61     | 1.00            | 0.64           | 7.9             |
| instance_hypernym             | 52      | 122    | 1.00            | 0.32           | 1.3             |
| member_of_domain_usage        | 545     | 43     | 0.98            | 0.02           | 1.5             |
| member_of_domain_region       | 543     | 42     | 0.88            | 0.02           | 1.0             |
| synset_domain_topic_of        | 13      | 115    | 1.00            | 0.42           | 0.7             |
| member_meronym                | 1402    | 307    | 1.00            | 0.22           | 7.9             |
| has_part                      | 848     | 196    | 1.00            | 0.24           | 7.1             |
| hypernym                      | 57      | 1254   | 1.00            | 0.15           | 3.7             |

Figure 3: Histograms of MuRE predictions for an example WN18RR relation of each type, split into true training, true test/validation and other instances.

Models achieve almost perfect training accuracy. The additive-multiplicative MuRE gives best test set performance, followed (surprisingly) closely by MuRE\(_I\), with multiplicative models (DistMult and TuckER) performing poorly on all but type R relations. Analysing a sample of “other” positive predictions for a relation of each type (see Appx. [C]), we estimate that TuckER is relatively accurate but pessimistic (~0.3 correct of the 0.5 predictions ≈ 60%), MuRE\(_I\) is optimistic but inaccurate (~2.3 of 7.5≈ 31%), whereas MuRE is both optimistic and accurate (~1.1 of 1.5≈ 73%).

Fig 3 shows histograms of MuRE prediction probabilities for the same sample relations, split by known truths (training and test/validation) and other instances. There is a clear distinction between relation types: for type R, most train and test triples are classified correctly with high confidence; for types S and C, an increasing majority of incorrect test predictions are far below the decision boundary, i.e. the model is confidently incorrect. For relations where the model is less accurate, fewer positive predictions are made overall and the prediction distribution is more peaked towards zero.

This predictive performance analysis shows more specifically how models have difficulty representing S/C relations and, given it provides new insight compared to ranking-based metrics, we recommend its inclusion in future link prediction work.

5 CONCLUSION

Many methods for learning low-rank representations of knowledge graphs have been developed, yet little is known about the latent structure they learn. This work focuses on understanding why existing knowledge graph representation models work and identifying their limitations, with the ultimate goal of improving representation. With this in mind, we build on recent understanding of PMI-based word embeddings to establish relation conditions, i.e. what a relation representation needs to achieve to map one word embedding to the other. Those conditions provide a framework to consider knowledge graph models, since if they are satisfied by the relation representations defined by a model’s architecture, then word embedding parameters are a known solution for its entity embeddings. This way, the encoding of semantic relations is not only understood – it is built into the geometry of entity and relation representation. We use this understanding to categorise knowledge graph relations according to their relation conditions. Our analysis confirms that a model’s ability to represent a specific relation type depends on the limitations imposed by the model architecture with respect to satisfying the necessary relation conditions. Finally, we evaluate several existing knowledge graph models based on their predictive performance.
ACKNOWLEDGEMENTS

Carl Allen and Ivana Balažević were supported by the Centre for Doctoral Training in Data Science, funded by EPSRC (grant EP/L016427/1) and the University of Edinburgh.

REFERENCES

Carl Allen and Timothy Hospedales. Analogies Explained: Towards Understanding Word Embeddings. In International Conference on Machine Learning, 2019.

Carl Allen, Ivana Balažević, and Timothy Hospedales. What the Vec? Towards Probabilistically Grounded Embeddings. In Advances in Neural Information Processing Systems, 2019.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. Multi-relational Poincaré Graph Embeddings. In Advances in Neural Information Processing Systems, 2019a.

Ivana Balažević, Carl Allen, and Timothy M Hospedales. TuckER: Tensor Factorization for Knowledge Graph Completion. In Empirical Methods in Natural Language Processing, 2019b.

Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don’t Count, Predict! A Systematic Comparison of Context-Counting vs. Context-Predicting Semantic Vectors. In Association for Computational Linguistics, 2014.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhanenko. Translating Embeddings for Modeling Multi-relational Data. In Advances in Neural Information Processing Systems, 2013.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. Toward an Architecture for Never-ending Language Learning. In Association for the Advancement of Artificial Intelligence, 2010.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2D Knowledge Graph Embeddings. In Association for the Advancement of Artificial Intelligence, 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In North American Chapter of the Association for Computational Linguistics, 2018.

Anna Gladkova, Aleksandr Drozd, and Satoshi Matsuoka. Analogy-based Detection of Morphological and Semantic Relations With Word Embeddings: What Works and What Doesn’t. In NAACL Student Research Workshop, 2016.

Lawrence J Hubert and Frank B Baker. Evaluating the Symmetry of a Proximity Matrix. Quality & Quantity, 13(1):77–84, 1979.

Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In International Conference on Learning Representations, 2015.

David Krackhardt. Graph Theoretical Dimensions of Informal Organizations. In Computational Organization Theory. Psychology Press, 2014.

Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. Canonical Tensor Decomposition for Knowledge Base Completion. In International Conference on Machine Learning, 2018.

Omer Levy and Yoav Goldberg. Neural Word embedding as Implicit Matrix Factorization. In Advances in Neural Information Processing Systems, 2014.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed Representations of Words and Phrases and their Compositionality. In Advances in Neural Information Processing Systems, 2013.

George A Miller. WordNet: a Lexical Database for English. Communications of the ACM, 1995.
Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Empirical Methods in Natural Language Processing*, 2014.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep Contextualized Word Representations. In *North American Chapter of the Association for Computational Linguistics*, 2018.

Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing Text for Joint Embedding of Text and Knowledge Bases. In *Empirical Methods in Natural Language Processing*, 2015.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex Embeddings for Simple Link Prediction. In *International Conference on Machine Learning*, 2016.

Peter D Turney. Similarity of Semantic Relations. *Computational Linguistics*, 32(3):379–416, 2006.

Peter D Turney and Patrick Pantel. From Frequency to Meaning: Vector Space Models of Semantics. *Journal of Artificial Intelligence Research*, 37:141–188, 2010.

Wenhan Xiong, Thien Hoang, and William Yang Wang. DeepPath: A Reinforcement Learning Method for Knowledge Graph Reasoning. In *Empirical Methods in Natural Language Processing*, 2017.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *International Conference on Learning Representations*, 2015.
A  CATEGORISING WORDNET RELATIONS

Table 7 describes how each WN18RR relation was assigned to its respective category.

Table 7: Explanation for the WN18RR relation category assignment.

| Type | Relation                          | Relatedness                      | Subject Specifics | Object Specifics |
|------|-----------------------------------|----------------------------------|-------------------|-----------------|
| R    | verb_group                        | both verbs; similar in meaning   | -                 | -               |
|      | derivationally_related_form also_see | different syntactic categories; semantically related | - | - |
| S    | hypernym                          | semantically similar              | instance of the object | instance of the object |
| C    | member_of_domain_usage member_of_domain_region member_meronym has_part synset_domain_topic_of | loosely semantically related (word usage features) | usage descriptor region descriptor collection of objects | broad feature set part of the subject domain descriptor |

B  ANALYSING NELL-995 AND FB15K-237 RELATIONS

NELL-995: We categorise a random subsample of 12 relations from the NELL-995 dataset (Xiong et al., 2017) containing 75,492 entities and 200 relations (a subset of NELL (Carlson et al., 2010)), which span our identified relation types (see Table 8). Explanation for the relation category assignment is shown in Table 9.

Table 8: Categorisation of NELL-995 relations.

| Type | Relation                          | Examples (subject entity, object entity)                                                                 |
|------|-----------------------------------|----------------------------------------------------------------------------------------------------------|
| R    | teamplaysagainststeam             | (sportsteam_rangers, sportsteam_mariners), (sportsteam_phillies, sportsteam_tampa_bay_rays)             |
|      | agentcollaborateswithagent        | (musicartist_white_stripes, musicartist_jack_white), (politician_obama, politician_fullary_clinton)     |
| S    | professionistypeofprofession      | (profession_trial_lawyers, profession_attorneys), (profession_engineers, profession_experts)          |
|      | animalistypeofanimal              | (mammal_cats, mammal_small_animals), (bird_chickens, agriculturalproduct_livestock)                    |
|      | chemicalistypeofchemical          | (chemical_moisture, chemical_gas), (chemical_oxide, chemical_materials)                               |
| C    | athleteplayssport                 | (athlete_joe_smith, sport_baseball), (athlete_chris_cooley, sport_football)                           |
|      | itemfoundinroom                   | (bedroomitem_bed, room_den), (hallwayitem_refrigerator, visualizablething_kitchen_area)             |
|      | bodypartcontainstbodypart         | (bodypart_system002, brainissue_eyes), (bodypart_blood, bodypart_left_ventricle)                     |
|      | atdate                            | (currency_scotland, date_n2009), (governmengorganization_wto, date_n2003)                           |
|      | locationlocatedwithinlocation     | (city_medellin, county_colombia), (city_jackson, stateorprovince_wyoming)                         |
|      |                                   | (city_ogunquin, stateorprovince_maine), (city_palmer_lake, stateorprovince_colorado)             |

Table 9: Explanation for the NELL-995 relation category assignment.

| Type | Relation                          | Relatedness                      | Subject Specifics | Object Specifics |
|------|-----------------------------------|----------------------------------|-------------------|-----------------|
| R    | teamplaysagainststeam             | both sport teams                 | -                 | -               |
|      | agentcollaborateswithagent        | both people or companies; related industries | - | - |
| S    | professionistypeofprofession      | semantically related (both profession types) | instance of the object | - |
|      | animalistypeofanimal              | semantically related (both animals) | instance of the object | - |
|      | chemicalistypeofchemical          | semantically related (both chemicals) | instance of the object | - |
| C    | athleteplayssport                 | semantically related (sports features) | athlete descriptor | sport descriptor |
|      | itemfoundinroom                   | semantically related (room/furniture features) | item descriptor | part of the subject |
|      | bodypartcontainstbodypart         | semantically related (specific body part features) | collection of objects | date descriptor |
|      | atdate                            | loosely semantically related (start date features) | broad feature set | collection of objects |
|      | locationlocatedwithinlocation     | semantically related (geographical features) | part of the subject | collection of objects |
|      |                                   | semantically related (geographical features) | part of the subject | collection of objects |

FB15k-237: Reviewing the FB15k-237 dataset (Toutanova et al., 2015) shows that the vast majority of relations are of type C (e.g. /people/person/profession, /location/location/contains), preventing contrast between relation types being drawn, so we consider that dataset no further.
C Splitting the NELL-995 Dataset

The test set of NELL-995 created by Xiong et al. (2017) contains only 10 out of 200 relations present in the training set. To ensure a fair representation of all training set relations in the validation and test sets, we create new validation and test set splits by combining the initial validation and test sets with the training set and randomly selecting 20,000 triples each from the combined dataset.

D Implementation Details

All algorithms are re-implemented in PyTorch with the Adam optimizer (Kingma & Ba, 2015) that minimises binary cross-entropy loss, using hyper-parameters that work well for all models (learning rate: 0.001, batch size: 128, number of negative samples: 50). Entity and relation embedding dimensionality is set to \( d_e = d_r = 200 \) for all models except TuckER, for which \( d_e = 30 \) (Balažević et al., 2019b).

E Krackhardt Hierarchy Score

The Krackhardt hierarchy score measures the proportion of node pairs \((x, y)\) where there exists a directed path \(x \rightarrow y\), but not \(y \rightarrow x\); and it takes a value of one for all directed acyclic graphs, and zero for cycles and cliques (Krackhardt, 2014; Balažević et al., 2019a).

Let \( M \in \mathbb{R}^{n \times n} \) be the binary reachability matrix of a directed graph \( G \) with \( n \) nodes, with \( M_{i,j} = 1 \) if there exists a directed path from node \( i \) to node \( j \) and 0 otherwise. The Krackhardt hierarchy score of \( G \) is defined as:

\[
Khs_G = \frac{\sum_{i=1}^{n-1} \sum_{j=1}^{n} I(M_{i,j} == 1 \land M_{j,i} == 0)}{\sum_{i=1}^{n} \sum_{j=1}^{n} I(M_{i,j} == 1)}.
\]  

F Symmetry Score

The symmetry score \( \in [-1, 1] \) (Hubert & Baker, 1979) for a relation matrix \( R \in \mathbb{R}^{d_e \times d_e} \) is defined as:

\[
s = \frac{\sum \sum_{i \neq j} R_{ij} R_{ji} - \frac{(\sum \sum_{i \neq j} R_{ij})^2}{d_e (d_e - 1)}}{\sum \sum_{i \neq j} R_{ij}^2 - \frac{(\sum \sum_{i \neq j} R_{ij})^2}{d_e (d_e - 1)}},
\]  

where 1 indicates a symmetric and -1 an anti-symmetric matrix.

G “Other” Predicted Facts

Tables 10 to 13 shows a sample of the unknown triples (i.e. those formed using the WN18RR entities and relations, but not present in the dataset) for the derivationally_related_form (R), instance_hypernym (S) and synset_domain_topic_of (C) relations at a range of probabilities \( \sigma(\phi(e_s, r, e_o)) \approx \{0.4, 0.6, 0.8, 1\} \), as predicted by each model. True triples are indicated in bold; instances where a model predicts an entity is related to itself are indicated in blue.
Table 10: “Other” facts as predicted by MuRE.

| Relation (Type) | $\sigma(v(e(r, r, e))) \approx 0.4$ | $\sigma(v(r, e, r)) \approx 0.6$ | $\sigma(v(e, r, r)) \approx 0.8$ | $\sigma(v(r, r, r)) \approx 1$ |
|-----------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|
| derivationally_related_form (R) | (equalizer_NN_2, set_off_VB_3) (constellation_NN_2, satellite_NN_3) (shrink_VB_3, subtraction_NN_2) (continue_VB_10, proceed_VB_1) (support_VB_6, defend_VB_5) (shutter_NN_1, flip_up_VB_3) (swimming_NN_1, patellar_reflex_NN_1) (swim_NN_1, spiral_VB_1) (stratum_NN_2, social_group_NN_1) (sheel_VB_1, scuffle_NN_3) | (extrapolation_NN_1, math_NN_2) (spread_VB_5, circularize_VB_3) (faint_NN_1, showing_NN_2) (extrapolation_3, synthesis_NN_3) (strategy_NN_1, mathN_2) (stroke_NN_1, math_NN_2) (stroke_NN_1, math_NN_2) (stroke_NN_1, math_NN_2) (stroke_NN_1, math_NN_2) (stroke_NN_1, math_NN_2) | (sewer_NN_2, stick_NV_1) (land_VB_1, vegetable_oil_NN_1) (snuggle_NN_1, draw_close_VB_3) (train_VB_3, training_NN_1) (scratch_VB_3, skin_sensation_NN_1) (scheme_NN_5, schematic_NN_1) (shrink_VB_3, math_NN_2) (shrink_VB_3, math_NN_2) (shrink_VB_3, math_NN_2) (shrink_VB_3, math_NN_2) (shrink_VB_3, math_NN_2) | (mud_VB_2, mud_VB_2) (worship_VB_1, worship_VB_1) (sorcerer_VB_1, sorcerer_VB_1) (sort_out_VB_1, sort_out_VB_1) (make_VB_1, make_VB_1) (utilize_VB_1, utilize_VB_1) (biology_NN_2, biology_NN_2) (biology_NN_2, biology_NN_2) |
Table 11: “Other” facts as predicted by DistMult.

| Relation (Type) | \( \sigma(\phi(e_s, r, e_o)) \approx 0.4 \) | \( \sigma(\phi(e_s, r, e_o)) \approx 0.6 \) | \( \sigma(\phi(e_s, r, e_o)) \approx 0.8 \) | \( \sigma(\phi(e_s, r, e_o)) \approx 1 \) |
|-----------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|
| (stag_VB_3, undercover_work_NN_1) | (print_VB_4, publisher_NN_2) | (criter_NN_3, pitcher_NN_2) | (play_VB_26, turn_NN_10) | (count_VB_4, recite_VB_2) |
| (vividness_NN_2, imblee_VB_3) | (sea_mew_NN_1, fawn_NN_1) | (shrub_NN_2, shrub_NN_2) | (alliterate_VB_1, versifier_NN_1) | (print_VB_4, publishing_house_NN_1) |
| ( derivationally_related_form(R) ) | (alliterate_VB_1, versifier_NN_1) | (allegorize_VB_1, allegorize_NN_1) | (print_VB_4, publisher_NN_2) | (alliterate_VB_1, versifier_NN_1) |
| (instance_hypernym(S) ) | (washington_NN_2, urban_center_NN_1) | (marshall_NN_2, lieutenant_general_NN_1) | (arizona_NN_1, state_NN_1) | (omaha_NN_1, urban_center_NN_1) |
| (synset_domain_topic_of(C) ) | (limitation_NN_4, trammel_VB_2) | (light_colonel_NN_1, colonel_NN_1) | (sermon_NN_1, sermonize_VB_1) | (etymology_NN_1, explanation_NN_1) |

| Relation (Type) | \( \sigma(\phi(e_s, r, e_o)) \approx 0.4 \) | \( \sigma(\phi(e_s, r, e_o)) \approx 0.6 \) | \( \sigma(\phi(e_s, r, e_o)) \approx 0.8 \) | \( \sigma(\phi(e_s, r, e_o)) \approx 1 \) |
|-----------------|---------------------------------|---------------------------------|---------------------------------|---------------------------------|
| (dish_NN_2, stew_NN_2) | (devise_VB_3, show_NN_2) | (system_NN_3, orderliness_NN_1) | (spread_NN_4, drowse_VB_3) | (develop_VB_2, plot_VB_2) |
| (wreath_VB_4, wreath_NN_1) | (point_NN_2, rock_star_NN_1) | (smiling_NN_1, scent_VB_1) | (alliterate_VB_1, versifier_NN_1) | (print_VB_4, publishing_house_NN_1) |
| (alleviate_VB_1, dichotomy_NN_1) | (rub_NN_2, roll_VB_2) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publisher_NN_2) | (alliterate_VB_1, versifier_NN_1) |
| (expose_VB_3, show_NN_2) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
| (finish_VB_6, finishing_NN_2) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
| (geology_NN_1, structural_JJ_5) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
| (crier_NN_3, pitchman_NN_2) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
| (system_NN_9, orderliness_NN_1) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
| (play_VB_26, turn_NN_10) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
| (resect_VB_1, amputation_NN_2) | (alliterate_VB_1, versifier_NN_1) | (alleviate_VB_1, dichotomy_NN_1) | (print_VB_4, publishing_house_NN_1) | (alliterate_VB_1, versifier_NN_1) |
Table 12: “Other” facts as predicted by TuckER.

| Relation (Type) | $\sigma(\phi(e, r, e_0)) \approx 0.4$ | $\sigma(\phi(e, r, e_0)) \approx 0.6$ | $\sigma(\phi(e, r, e_0)) \approx 0.8$ | $\sigma(\phi(e, r, e_0)) \approx 1$ |
|-----------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|
| **derivationally_related_form** (R) | (tympanist_NN, gong_NN) | (mash_NN, mill_VB) | (take_chances_VB, venture_NN) | (venturer_NN, venture_NN) |
| | (turn_over_V, rotation_NN) | (walk_VB, inner_frame_NN) | (shatter_NN, fall_up_VB) | (dynamist_NN, dynamist_NN) |
| | (date_VB, geological_dating_NN) | (use_VB, utility_NN) | (exhale_VN, leave_VB) | (love_VB, lover_NN) |
| | (set_VB, shroud_NN) | (musical_instrument_NN, write_VB) | (tremble_VN, vibrate_VB) | (snuggle_NN, squeeze_VB) |
| | (tyro_NN, start_VB) | (lining_VN, wrap_up_VB) | (motor_VN, trip_VB) | (departure_NN, die_VB) |
| | (identification_NN, name_VN) | (cot_temp_VB, wrangle_VB) | (support_VN, endorsement_NN) | (repentant_JJ, repentant_JJ) |
| | (stubber_VN, thrust_VN) | (identification_VN, certification_VN) | (federate_VN, confederation_NN) | (tread_NN, step_VB) |
| | (justification_NN, apology_NN) | (manifestation_VB, prevarication_NN) | (take_over_V, return_VN) | (stockist_NN, stockist_NN) |
| | (manufacture_VN, prevarication_NN) | (synchronize_VB, synchronoscope_NN) | (wait_on_V, supporter_NN) | (philanthropist_NN, philanthropist_NN) |
| | **instance_hypernym** (S) | (deep_south_NN, south_NN) | (rpm_dev_VN, rmp_dev) | (cowsens_NN, siege_NN) | (r_e_byrd_NN, siege_NN) |
| | (st_paul_NN, organist_NN) | (irish_capital_NN, urban_center_NN) | (mccormick_NN, arms_manufacturer_NN) | (show_NN, women's_rightist_NN) |
| | (fallingun_NN, urban_center_NN) | (tom_the_apostle_NN, apostle_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (malcolm_x_NN, emancipationist_NN) | (st_paul_NN, apostle_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (thomas_the_apostle_NN, priest_NN) | (mccormick_NN, arms_manufacturer_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (russian_orthodox_NN, church_father_NN) | (mccormick_NN, arms_manufacturer_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (russian_orthodox_NN, priest_NN) | (mccormick_NN, arms_manufacturer_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (russian_orthodox_NN, theologian_NN) | (mccormick_NN, arms_manufacturer_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (russian_orthodox_NN, theologian_NN) | (mccormick_NN, arms_manufacturer_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | (russian_orthodox_NN, theologian_NN) | (mccormick_NN, arms_manufacturer_NN) | (mccormick_NN, arms_manufacturer_NN) | (weight_NN, stat_NN) |
| | **synset_domain_topic_of** (C) | (roll-on_roll-off_NN, motorcar_NN) | (drive_NN, badminton_NN) | (impo_NN, capital_NN) | (russian_orthodox_NN, church_father_NN) |
| | (libel_NN, legislature_NN) | (drive_NN, badminton_NN) | (impo_NN, capital_NN) | (russian_orthodox_NN, church_father_NN) |
| | (roll-on_roll-off_NN, passenger_vehicle_NN) | (drive_NN, badminton_NN) | (impo_NN, capital_NN) | (russian_orthodox_NN, church_father_NN) |
Table 13: “Other” facts as predicted by MuRE.

| Relation (Type) | $\sigma(\alpha(x, r, e)) \approx 0.4$ | $\sigma(\alpha(x, r, e)) \approx 0.6$ | $\sigma(\alpha(x, r, e)) \approx 0.8$ | $\sigma(\alpha(x, r, e)) \approx 1$ |
|-----------------|------------------------------------|------------------------------------|------------------------------------|------------------------------------|
| derivationally_related_form (R) | (surround_VB_1, wall_NN_1) (unpleasant JJ_1, unpleasantness NN_1) (low VB_3, enjoyment NN_2) (magnitude JJ_1, tell JJ_1) (testify VB_2, information NN_1) (connect VB_6, converging NN_1) (connect VB_6, connection NN_4) (operate VB_4, pyropy NN_1) (market VB_1, trade NN_4) (operate VB_4, mission NN_2) | (word_picture NN_1, sketch VB_2) (develop VB_10, adjustment NN_4) (gloss VB_3, commentary NN_1) (violate VB_2, violation NN_2) (suffocate VB_1, strangle NN_1) (number VB_3, point NN_12) (develop VB_10, organic process NN_1) (plecion NN_1, twist VB_4) (split up VB_3, separation NN_5) (plecion NN_1, wrinkle VB_2) | (smelling NN_1, wind VB_4) (try out VB_1, organic cell nuclear transplantation NN_1) (lighting NN_4, set on fire VB_1) (symptom NN_1, one-half NN_1) (just JJ_1, validity NN_1) (represent VB_1, talking to NN_1) (sustain VB_5, barn NN_2) (spring NN_6, handle VB_1) (spark NN_1, scintillate VB_1) (utility NN_2, functional JJ_1) (use NB_1, instrument NN_2) | (pollution NN_2, walk VB_1) (desire NN_2, hope VB_2) (stumble VB_3, whine NN_1) (naturalization NN_1, sound out VB_1) (read NN_1, step VB_2) (yawn VB_1, pining NN_1) (unreliableness NN_1, arbitrary JJ_1) (treaties NN_2, travels NN_2) |