Joint Completion and Alignment of Multilingual Knowledge Graphs

Soumen Chakrabarti[1], Harkanwar Singh[2], Shubham Lohiya[1], Prachi Jain[2], Mausam[2]
[1]IIT Bombay
soumen@cse.iitb.ac.in
[2]IIT Delhi
mausam@cse.iitd.ac.in

Abstract

Knowledge Graph Completion (KGC) predicts missing facts in an incomplete Knowledge Graph (KG). Multilingual KGs associate entities and relations with surface forms written in different languages. An entity or relation may be associated with distinct IDs in different KGs, necessitating entity alignment (EA) and relation alignment (RA). Many effective algorithms have been proposed for completion and alignment as separate tasks. Here we show that these tasks are synergistic and best solved together. Our multitask approach starts with a state-of-the-art KG embedding scheme, but adds a novel relation representation based on sets of embeddings of (subject, object) entity pairs. This representation leads to a new relation alignment loss term based on a maximal bipartite matching between two sets of embedding vectors. This loss is combined with traditional KGC loss and optionally, losses based on text embeddings of entity (and relation) names. In experiments over KGs in seven languages, we find that our system achieves large improvements in KGC compared to a strong completion model that combines known facts in all languages. It also outperforms strong EA and RA baselines, underscoring the value of joint alignment and completion.

1 Introduction

A knowledge graph (KG) has nodes representing entities and directed edges representing relations between subject and object entities. In a KG, each entity (and relation) has a unique node (relation) ID. A *monolingual* KG is associated with a single human language — an entity node (or relation ID) is associated with one or more surface forms in that language. E.g., the ID for the country USA may have aliases like "United States of America".

KGs are usually very incomplete, as curators struggle to keep up with the real world. KG completion (KGC) is thus a strongly motivated problem involving the prediction of true facts unknown to the KG. Traditional KGC methods (Bordes et al., 2013; Trouillon et al., 2016; Sun et al., 2019b) follow a link prediction paradigm: represent a KG as an abstract graph without entity or relation aliases, and predict missing facts as ID tuples.

Most KGC methods are applicable to only one KG at a time. However, different language speakers would maintain separate monolingual KGs in their own languages. Independent completions of each KG may not be optimal, since information from one KG will likely help completion of the other. Massive KGs like WikiData cover well-resourced and resource-poor languages. Cross-lingual transfer thus acquires increased utility in the context of Linked Open Data initiatives.

Different KGs will assign a different ID, and often also a different surface form, to the same entity, such as "Estados Unidos de América" for the USA entity in a Spanish KG. This leads to ID proliferation. Recent work on entity alignment (EA) across KGs attempts to assign a unique ID to all nodes representing the same entity (Chen et al., 2017; Sun et al., 2017, 2018, 2020c; Cao et al., 2019; Chen et al., 2021; Tang et al., 2020). Most recent EA methods use surface forms and possibly other text and attributes associated with each ID. A related task is relation alignment (RA) — assigning synonymous relations in different KGs the same relation ID. Compared to EA, RA is under-explored, particularly in case of multilingual KGs. RA involves *global* evidence, because the decision to merge two relations in two languages is essentially an act of *schema integration*, having far-reaching consequences on many facts in both KGs.

Our key contribution is to recognize and exploit the synergy between multilingual KGC, EA and RA tasks, particularly in the realistically motivated scenario of transferring from multiple source language KGs to a target language KG. Entity and relation alignments expose a KGC algorithm to more facts, which can lead to better completion. Conversely, a high-confidence completion can give additional evidence to align entities and relation (see extended example in Appendix A).

We present ALIGNKGC, a multi-task system that learns to optimize for KGC, EA and RA jointly. We develop ALIGNKGC in two settings. Our first 'GRAPHONLY' setting follows KGC literature,

where no surface forms or other textual information is available. This helps in direct comparison against other KGC approaches. Here, ALIGNKGC uses a novel subject-object signature of each relation, which it represents as a bag of pairs of entity embeddings. Relation pairs are compared for equivalence and implication using maximal matchings between these embedding sets. RA supervision imposes a loss over these maximal matching measurements, which can be reduced via gradient descent on entity embeddings. This RA-based loss term is combined with a state-of-the-art KGC loss. Our second 'GRAPHTEXT' setting additionally employs entity and relation surface forms (when available), as in the standard EA research. Here, ALIGNKGC incorporates similarity between surface forms as an additional loss term based on EA (RA) supervision.

We evaluate ALIGNKGC on slices of DBPedia collected in the DBP5L (Chen et al., 2020), DBP15K (Sun et al., 2017) and OpenEA (Sun et al., 2020b) datasets, covering seven languages. In both GRAPHONLY and GRAPHTEXT settings, ALIGNKGC improves substantially on KGC and EA tasks compared to recent baselines (Yao et al., 2019; Wu et al., 2019a; Zhu et al., 2021b), and gives competitive RA performance. These results underscore the value of joint training for these tasks. Our implementation is public.[1] A preliminary version of this work was published by Singh et al. (2021).

2 Notation and preliminaries

A knowledge graph (KG) provides a graph-view to a knowledge-base (KB). A KG consists of entities E and relations (aka relation types) R. A KG instance is a triple (s, r, o) where $s, o \in E$ and $r \in R$. These are all canonical IDs. Each ID (say s) is associated with one or more textual aliases ($\text{text}(s)$). KGC, EA and RA systems usually associate these IDs with embeddings to be trained, which we denote as $\boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}$, etc. These could be real or complex vectors.

KG completion For any single KG, training data is provided as (s, r, o) triples. A test instance has the form $(s, r, ?)$ or $(?, r, o)$ where the system has to predict o or s. Multiple correct values are possible. The evaluation protocol takes a system-ranked list of candidate objects or subjects, and then measures MRR or HITS@K (a.k.a. precision@K) based on position of the gold entity. The *filtered* evaluation (Bordes et al., 2013) removes valid train or test tuples ranking above the gold (s, r, o) for scoring purposes. With rare exceptions (Yao et al., 2019), KGC has traditionally been solved in the GRAPHONLY setting, without using textual features.

We extend monolingual KGC by considering a set $L = \{l_1, l_2, \ldots\}$ of languages, with KG_l representing the KG supported by the language l. An entity in this KG is denoted as e_l, and a relation as r_l. While trying to improve KGC for target language l', KGs in multiple source languages l may be beneficial.

EA & RA alignment An entity alignment (**EA**) or equivalence between entities e_l and $e_{l'}$ in KGs of two languages l and l' is denoted as $e_l \equiv e_{l'}$, also written as the triple $(e_l, \equiv, e_{l'})$. Similarly, a relation alignment (**RA**) or equivalence between relations r_l and $r_{l'}$ is specified as $r_l \equiv r_{l'}$. Other types of alignment between relation pairs (such as $r_l \Rightarrow r_{l'}$) may be possible, though not studied in this paper. During training, a set of gold EAs $\{(e_l^n, \equiv, e_{l'}^n) : n = 1, \ldots, N\}$ and a set of gold RAs $\{(r_l^m, \equiv, r_{l'}^m) : m = 1, \ldots, M\}$ are revealed to the system between each KG_l and $KG_{l'}$. The system's goal is to infer additional entity and relation equivalences. It usually produces a ranked list of equivalences, evaluated using HITS@K or MRR.

Joint KGC, EA & RA problem definition In the GRAPHONLY setting, given a set of KGs $\{\text{KG}_l | l \in L\}$, gold EAs and gold RAs, the goal is to predict new facts (s_l, r_l, o_l) for each KG_l, and new entity and relation alignments $(e_l, \equiv, e_{l'})$ and $(r_l, \equiv, r_{l'})$. For GRAPHTEXT setting, each entity e_l and relation r_l may have associated text $\text{text}(e_l)$ and $\text{text}(r_l)$.

3 Related work

KG completion KGC, through learning embeddings for entities and relations, is a densely-populated research landscape. At a high level, they fit entity and relation representations, along with a scoring function, such that training KG facts are scored higher than randomly sampled negative facts. Thus, the model size scales with the number of KG entities and relations. Among the best performers are ComplEx (Trouillon et al., 2016), ConvE (Dettmers et al., 2018), and RotatE (Sun et al., 2019b). Almost all systems are designed for a single KG and in GRAPHONLY setting, agnostic to the language used in entity and relation aliases.

[1] https://github.com/soumen-chakrabarti/alignkgc-tgz.git

We use ComplEx (Trouillon et al., 2016; Lacroix et al., 2018) as our baseline KGC gadget, because it gives near state-of-the-art predictions. ComplEx defines a triple score as

$$f(s,r,o) = \Re\left(\langle \boldsymbol{s}, \boldsymbol{r}, \boldsymbol{o}^\star \rangle\right) = \Re\left(\sum_d s_d r_d o_d^\star\right), \quad (1)$$

where $c^\star$ is complex conjugate, $\langle \cdots \rangle$ is a 3-way elementwise inner product and $\Re(\cdot)$ is the real part of a complex number. Using f, ComplEx defines $\Pr(o|s,r) = \frac{e^{f(s,r,o)}}{\sum_{o'} e^{f(s,r,o')}}$ and $\Pr(s|o,r) = \frac{e^{f(s,r,o)}}{\sum_{s'} e^{f(s',r,o)}}$. To train, we use negative log-likelihood **KGC loss**

$$L_{\text{KGC}} = \sum_{(s,r,o)\in \text{KG}} -\log \Pr(o|s,r) - \log \Pr(s|o,r). \quad (2)$$

EA & RA alignment Early EA methods worked in the GRAPHONLY setting. They include TransE adaptations, such as MTransE (Chen et al., 2017), and TransEdge (Sun et al., 2019a), iterative EA bootstrapping (Sun et al., 2018), a combination of ComplEx and DistMult (Shi and Xiao, 2019), and use of GNNs (Sun et al., 2020c).

Recent EA methods (Wu et al., 2019a,b; Zhu et al., 2021b,a) are based on GNNs over KGs, where entity (node) embeddings are computed over an entity's relational neighborhood, and equivalence is checked by comparing node embeddings. BERT-INT (Tang et al., 2020) compares two nodes through direct 1-hop neighbor-to-neighbor comparisons. Like us, RDGCN (Wu et al., 2019a) and RNM (Zhu et al., 2021b) recognize the duality between EA and RA. They benefit from the GRAPHTEXT setting, by using deep text encodings (e.g., mBERT (Devlin et al., 2019)) on the name/description of an entity (in different languages) to obtain node vectors for initializing GNNs. RNM obtains the best EA accuracy, though if GNNs are initialized with random vectors, RNM's performance plummets. Hence, we compare against these in GRAPHTEXT setting only. Meanwhile, RA approaches have been limited to computing overlap of entity-pairs in non-neural settings (Lin and Pantel, 2001; Bhagat et al., 2007; Nakashole et al., 2012).

The popular DBP15K (Sun et al., 2017) EA dataset has been criticized (Liu et al., 2020; Berrendorf et al., 2021), for containing many EA pairs with almost exact name matches, unduly rewarding text-based methods. DBP5L (Chen et al., 2020) and OpenEA (Sun et al., 2020b, v. 1) have similar limitations. Like Mao et al. (2020a), we offer a flexible option to use or not use textual descriptions of entities and relations. We clearly separate the effect of text features through our two settings, and also show EA performance after filtering exact matches, to focus on non-trivial EA queries.

Against this backdrop, the **main novelties of ALIGNKGC** are its: 1) multi-task integration between KGC, EA and RA; 2) non-aggregative representation of relations in terms of entity embeddings; and 3) support for multiple source to single target language transfer.

4 The design of ALIGNKGC

Perhaps the most straight-forward way to apply a KGC system across multiple KGs is to compute $\text{KG}_U = \bigcup_{l\in L} \text{KG}_l$ such that for all revealed entity equivalences $\{(e_l, \equiv, e_{l'})\}$, collapse their node pairs, and rename all equivalent $r_l, r_{l'}$ relation IDs to a common new ID (a transitive closure may be needed to identify equivalence classes of entities or relations). As pointed out in multiple works (Sun et al., 2017; Zhu et al., 2017; Shi and Xiao, 2019), "pre-aligned pair should share embedding to bridge different KGs. Under this assumption, the semantic loss between equivalent entity or relation pair is zero." A standard KGC system, such as ComplEx, can work on the resulting KG_U, and we call this method **KGCUNION**. Comparing entity and relation embeddings across languages provides a first solution for EA and RA as well.

How similar are two relations $r_l, r_{l'}$ in KG_U? Comparing their embedding vectors $\boldsymbol{r}_l, \boldsymbol{r}_{l'}$, computed by ComplEx, gives a first estimate. However, we found that this did not yield the best results. Instead, guiding the training through an auxiliary loss, computed using the following explicit *relation signatures* helped the model learn better embeddings. Here s, o are interpreted as canonical IDs (hence, 'hard'). This will be generalized shortly.

Definition 1 (Hard SO-signature). We define the ('hard') subject-object signature of a relation r as

$$\text{SO}(r) = \{(s,o) : (s,r,o) \in \text{KG}_U\}. \quad (3)$$

4.1 Hard SO-overlap & Jaccard similarity

Definition 2 (SO-overlap). The (hard) overlap between two relations $r_l, r_{l'}$ is $|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|$.

Jaccard similarity can then be used as a standard symmetric belief that two relations IDs in different languages are equivalent:

$$b_J(r_l \Leftrightarrow r_{l'}) = \frac{|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|}{|\,\text{SO}(r_l) \cup \text{SO}(r_{l'})|}. \quad (4)$$

If $b_J(r_l \Leftrightarrow r_{l'})$ exceeds a threshold θ (tuned using dev set), we add $(r_l, r_{l'})$ to set Ψ_J of 'silver' alignments. We define an additional RA loss term encouraging such embeddings to be aligned:

$$L_{\text{RA-J}} = \sum_{(r_l, r_{l'}) \in \Psi_J} b_J(r_l \Leftrightarrow r_{l'}) \left\| \boldsymbol{r}_l - \boldsymbol{r}_{l'} \right\|_1. \quad (5)$$

We call this scheme **Jaccard**.

4.2 Asymmetric subsumption

A problem with Jaccard similarity is that it can give a symmetric high score to relation pairs having asymmetric implications between them. E.g., in the DBP5L data set, Jaccard similarity gives a large symmetric similarity score between `locationCity` and `head-quarter`, or `keyPerson` and `founders` relations in different KGs. However, these relation pairs clearly involve asymmetric implication, and should not be linked. Therefore, we need an asymmetric belief measure (b_A) for one relation subsuming another. We define it as

$$b_A(r_l \Rightarrow r_{l'}) = \frac{|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|}{|\,\text{SO}(r_l)|} \in [0, 1] \quad (6)$$

The asymmetry in (6) refrains from pushing those relation pairs together. This avoids some erroneous inferences, generally boosting precision — possibly at the expense of some recall, as this may also remove some "reasonable inferences". Extending

$$(r_l \Leftrightarrow r_{l'}) \quad \text{iff} \quad (r_l \Rightarrow r_{l'}) \wedge (r_{l'} \Rightarrow r_l) \quad (7)$$

to the fuzzy domain, we get

$$b_A(r_l \Leftrightarrow r_{l'}) = \min\{b_A(r_l \Rightarrow r_{l'}), b_A(r_{l'} \Rightarrow r_l)\} \quad (8)$$

which we readily see as equivalent to

$$|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})| \min\left\{\frac{1}{|\,\text{SO}(r_l)|}, \frac{1}{|\,\text{SO}(r_{l'})|}\right\}$$
$$= \frac{|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|}{\max\{|\,\text{SO}(r_l)|, |\,\text{SO}(r_{l'})|\}} \in [0, 1]. \quad (9)$$

Similar to Eqn. (5), we incentivize the model to align r_l and $r_{l'}$ if their $b_A(r_l \Leftrightarrow r_{l'})$ is high. In particular, we search for pairs of 'silver' alignments $(r_l, r_{l'}) \in \Psi_A$, where $r_l = \text{argmax}_{r'_l} b_A(r_{l'} \Rightarrow r'_l)$ and $r_{l'} = \text{argmax}_{r'_{l'}} b_A(r_l \Rightarrow r'_{l'})$. In other words, we want to apply the loss only to partner relations with mutually largest implication beliefs in that language. We define an asymmetric RA loss as:

$$L_{\text{RA-A}} = \sum_{(r_l, r_{l'}) \in \Psi_A} b_A(r_l \Leftrightarrow r_{l'}) \left\| \boldsymbol{r}_l - \boldsymbol{r}_{l'} \right\|_1. \quad (10)$$

We call this method **Asymmetric**. Note that our *expression* (10) is not asymmetric, but the notion of relation entailment is. There are many other standard notions of symmetric and asymmetric overlap: Jaccard, overlap, simple matching, Hamming, Dice, and Tversky coefficients, and exploring these is left for future work.

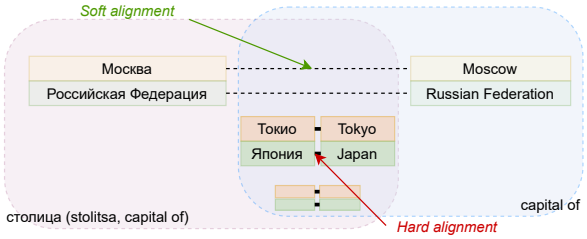

Figure 1: Motivation behind SO-set representation, showing En and Ru relation `capital-of`, with hard and soft overlaps among their SO-sets.

4.3 Soft overlap and approximation

In the above definitions involving SO, we assumed the (s, o) pairs were represented using canonical entity IDs. Pairs such as (Tokyo, Japan) are in the "hard intersection" shown in Figure 1. Unless explicit entity equivalences are provided, 'hard' SO overlap will underestimate relation similarities. Based on their KG neighborhoods, entity 'Moscow' in the English KG may have a similar embedding to its Russian counterpart, and the same may hold for "Russian Federation". These are shown as "soft alignments" in Figure 1. We now extend these ideas to redefine SO-signatures using entity embeddings, which can be trained via gradient descent.

Recall that the KGC system obtains embedding vectors $\boldsymbol{e}$ for each entity e in the KG. We modify our earlier definition of subject-object signature (and reuse the notation):

Definition 3 (Soft SO signature). For each relation r,

$$\text{SO}(r) = \{(\boldsymbol{s}, \boldsymbol{o}) : (s, r, o) \in \text{KG}_U\}. \quad (11)$$

Each element is the concatenation of the subject and object *embedding vectors*. The ith embedding pair in $\text{SO}(r)$ is denoted $\text{SO}(r)[i]$. Two such pairs can be compared by, say, extending cosine similarity with an AND-semantic:

$$\text{sim}\left((\boldsymbol{s}, \boldsymbol{o}), (\boldsymbol{s}', \boldsymbol{o}')\right) = \sigma(\cos(\boldsymbol{s}, \boldsymbol{s}'))\sigma(\cos(\boldsymbol{o}, \boldsymbol{o}')), \quad (12)$$

where σ is the sigmoid nonlinearity. This captures the requirement that *both* subjects and objects have to be similar.

The key challenge is to compare two *sets* of such SO vectors, $\text{SO}(r_l), \text{SO}(r_{l'})$ for extending Defn. 2. A simple approach is to use a network to encode each SO-set (Zaheer et al., 2017; Lee et al., 2019) into a set embedding and then compare these (see Appendix J for details). However, recent work (Pabbaraju and Jain, 2019; Tang et al., 2020) suggests that we should drill down to pairwise interactions

between set elements.

To that end, consider two relations from different languages, $r_l, r_{l'}$, that are candidates for alignment. Suppose the $(\boldsymbol{s}, \boldsymbol{o})$ pairs of r_l are indexed by i and pairs of $r_{l'}$ are indexed by j. We build a matrix $\boldsymbol{S}_{r_l,r_{l'}}$ of pairwise cosine similarities:

$$S_{r_l,r_{l'}}[i,j] = \text{sim}\Big(\text{SO}(r_l)[i], \text{SO}(r_{l'})[j]\Big). \quad (13)$$

Definition 4 (Soft SO-Overlap). The continuous extension of $|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|$, denoted by $\text{SoftOv}(r_l, r_{l'})$, is defined as the value of the *maximal matching* on the weighted bipartite graph induced by $S_{r_l,r_{l'}}$.

In Appendix I we show that SoftOv generalizes hard-SO overlap. Note that $\text{SoftOv}(r_l, r_{l'})$ depends on entity embeddings, which are to be trained. Ideally, we should be able to backpropagate various KGC and alignment losses past the solution of the matching problem to the entity and relation embeddings. Gumbel-Sinkhorn matrix scaling (Cuturi, 2013; Mena et al., 2018) can be used for this purpose, but it is computationally expensive at KG scales (Appendix K).

Here we use a computationally cheaper approximation: only if i is j's strongest partner ($i = \text{argmax}_{i'}\, S[i', j]$) and j is i's strongest partner ($j = \text{argmax}_{j'}\, S[i, j']$), we retain edge (i, j) in our matching. (In Section 5.9, we show that this pruning reduces the value of our matching by less than 1% from the optimal matching.) Let the retained edge set be $P(r_l, r'_l)$. For each retained edge (i, j), we accumulate into $\text{SoftOv}(r_l, r_{l'})$ a score increment of $\sigma\big(S_{r_l,r_{l'}}[i,j]\, w + c\big)$, where σ is the sigmoid nonlinearity, and $w > 0, c \in \mathbb{R}$ are model parameters trained along with all embeddings. Thus, we approximate $|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|$ using

$$\text{SoftOv}(r_l, r_{l'}) = \sum_{i,j:P(i,j)} \sigma\big(S_{r_l,r_{l'}}[i,j]\, w + c\big), \quad (14)$$

We continue to use (6)-(9) as defined in Asymmetric, except we replace the hard overlap $|\,\text{SO}(r_l) \cap \text{SO}(r_{l'})|$ by $\text{SoftOv}(r_l, r_{l'})$. We denote the beliefs computed thus by b_{SA}. The rest of the machinery of the previous section follows, i.e., we compute 'silver' alignments $(r_l, r_{l'}) \in \Psi_{SA}$, leading to a soft asymmetric RA loss as:

$$L_{\text{RA-SA}} = \sum_{(r_l,r_{l'})\in\Psi_{SA}} b_{SA}(r_l \Leftrightarrow r_{l'}) \big\|\boldsymbol{r}_l - \boldsymbol{r}_{l'}\big\|_1. \quad (15)$$

We call this method **SoftAsymmetric** or **SoftAsym** for short.

For the GRAPHONLY setting, our multitask objective (see Figure 2) has the form

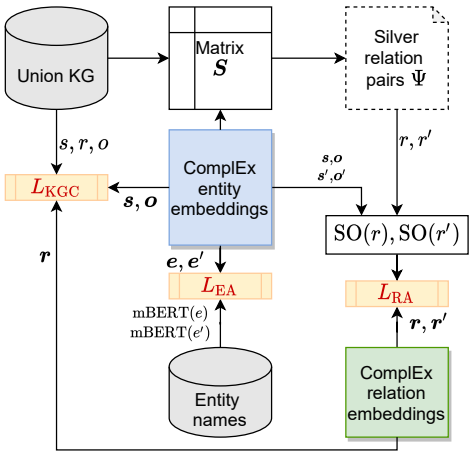

Figure 2: The three main loss components of ALIGN-KGC: L_{KGC}, L_{EA}, and L_{RA}. KG IDs/indices are denoted $r, r', e, e', s, s', o, o'$ whereas corresponding vectors are denoted $\boldsymbol{r}, \boldsymbol{r'}, \boldsymbol{e}, \boldsymbol{e'}, \boldsymbol{s}, \boldsymbol{s'}, \boldsymbol{o}, \boldsymbol{o'}$. GRAPHTEXT methods use L_{EA}; GRAPHONLY methods do not.

$$L_{\text{KGC}} + \alpha L_{\text{reg}} + \beta L_{RA*}, \quad (16)$$

where one of $L_{\text{RA-J}}, L_{\text{RA-A}}$ or $L_{\text{RA-SA}}$ is plugged in as $L_{\text{RA}*}$. L_{reg} is an L_2 regularization on embeddings. $\alpha, \beta, \ldots \geq 0$ are tuned hyperparameters. The next subsection extends this model to the GRAPHTEXT setting by introducing text features.

4.4 Adding text signals

In the GRAPHTEXT setting, cross-lingual alignment can benefit from similarity between entity names and relation names (Berrendorf et al., 2021). Exact match can be trivially exploited by adding $(e_l, \equiv, e_{l'})$ as a (noisy) 'silver' EA instance (and collapse those entity nodes) in KG_U if $\text{text}(e_l) \equiv \text{text}(e_{l'})$. Beyond exact match, BERT-INT and GNN-based systems (Wu et al., 2019a,b; Tang et al., 2020; Zhu et al., 2021b,a) bootstrap graph-based propagation from some form of text embedding, abstracted as $\text{Embed}(\text{text}(e_l))$, which we write compactly as $\text{TxtEmb}(e_l)$. Additional details are in Section 5.2 and Appendix C.

If the text embeddings of two entities in different languages are similar, we would like their KG-based embeddings (as obtained using previous sections) to be similar as well. This can be achieved by augmenting Eqn. (16) with another loss term

$$L_{\text{EA}} = \sum_{e_l,e_{l'}} \cos\big(\text{TxtEmb}(e_l), \text{TxtEmb}(e_{l'})\big) \times \big\|\boldsymbol{e_l} - \boldsymbol{e_{l'}}\big\|_1, \quad (17)$$

thus giving the overall multitask objective $L_{\text{KGC}} + \alpha L_{\text{reg}} + \beta L_{RA*} + \gamma L_{EA}$. Depending on whether $L_{\text{RA-J}}, L_{\text{RA-A}}$ or $L_{\text{RA-SA}}$ is used in Eqn. (16), we call the resulting methods **Jaccard+Text**,

Asym+Text and **SoftAsym+Text**. Relation texts, if available, can be used via a similar loss term.

5 Experiments

We study these research questions: 1) Does joint KGC, EA and RA (ALIGNKGC) improve upon independent models for each of these tasks? 2) Is there cross-lingual transfer of knowledge, for facts mentioned in one language, but queried in another? 3) What are the incremental contributions of soft overlap and text embeddings in the overall performance of ALIGNKGC? 4) How does performance vary with the number of seed alignments (both entity and relation) provided at training time?

5.1 Data sets

Popular KGC datasets such as FB15k, FB15k-237, WN18, WN18RR, and YAGO-3-10 are not multilingual. In this work, we consider the following data sets: 1) **DBP5L** (Chen et al., 2020), a dataset tailored for combined KGC and EA tasks with English, Japanese, Greek, Spanish, French KGs; 2) **DBP15K** (Sun et al., 2017), developed for the EA task and has English, French, Japanese, Chinese KGs; and 3) **OpenEA** (Sun et al., 2020b), also designed for the EA task with English, German, French KGs. In all splits, we use 30% of entity alignments and 0% relation alignments for training. Note, our results in DBP15K and OpenEA do not exactly match published work, since (for KGC evaluation), we remove the test facts from training KGs. For GRAPHTEXT methods, we use entity names as textual information for an entity. Relations in these datasets do not have associated surface forms in different languages, so no textual RA loss is added. Further details of datasets and text processing are in Appendices B and C.

5.2 Generating text features

Each competing GRAPHTEXT method was given *all* text featurization options (Appendix C). To save space, we reported in the main paper only the best-performing setting for each individual model. Available settings were:

GloVE, BERT: English-only, therefore unsuited.

Translate+GloVE: High-quality translation to English followed by GloVE.

Translate+BERT: Ditto, followed by BERT.

mBERT: Directly embeds multilingual texts. For some low-resource languages, mBERT's word-piece dictionary is deficient. In such cases Translate+BERT may be better than mBERT.

Translate+mBERT: To keep competing methods standardized, we avoid any BERT-to-mBERT discrepancy (in word-piece vocabularies, say), using mBERT even after translation to English.

Method	Best text featurization scheme
RDGCN (Wu et al., 2019a)	Translate+GloVE
RNM (Zhu et al., 2021b)	Translate+GloVE
RAGA (Zhu et al., 2021a)	Translate+GloVE
ALIGNKGC=SoftAsym+Text	Translate+mBERT

Table 1: Best text features for GraphText methods.

The best text featurization choices for each method are shown in Table 1. Compared to RDGCN/RNM/RAGA, AlignKGC benefits from fine-tuning mBERT using EA before multi-task training. Others suffer from mBERT's larger model capacity and prefer GloVe.

5.3 Methods compared

KGC: For KGC GRAPHONLY setting, we have three baselines: 1) ComplEx, when applied to any one KG in isolation, which we call **KGCMONO**, 2) **KGCUNION** – ComplEx applied to KG_U, and 3) KEnS (Chen et al., 2020), a recent multilingual KGC algorithm. KEnS embeds all KGs in a shared space. It ensembles predictions from embedding models of multiple language-specific KGs. All other GRAPHONLY methods are ablations of ALIGNKGC, ranging across **Jaccard**, **Asymmetric**, and **SoftAsym**. For assessing the benefit of text signals (GRAPHTEXT), we evaluate **Asym+Text** and **SoftAsym+Text**, and also report on **KG-BERT** (Yao et al., 2019). For each competing GRAPHTEXT method, we use its most favorable text encoding method among translate+GloVE, mBERT, and translate+mBERT.

EA and RA: KGCUNION and ALIGNKGC variations can also provide EA predictions by comparing embeddings $\boldsymbol{e}$ using cosine similarity to rank equivalence candidates. We compare these against **RNM** and **RDGCN**, the best GRAPHTEXT baselines. Neither ALIGNKGC nor any baseline is allowed to access test fold KGC, EA or RA instances for any of these three tasks. For RA comparisons, following the approach of RNM, we recognize that both embeddings $\boldsymbol{r}_l, \boldsymbol{r}_{l'}$ and sets $\mathrm{SO}(r_l), \mathrm{SO}(r_{l'})$ can provide valuable signal toward the scoring of $(r_l, \equiv, r_{l'})$. Accordingly, let $S(r) = \{s : (s, \cdot) \in \mathrm{SO}(r)\}$ and $O(r) = \{o : (\cdot, o) \in \mathrm{SO}(r)\}$ be the subject and object entities involved with r, and let $\overrightarrow{S(r)}, \overrightarrow{O(r)}$ be the average of their entity embeddings. Following RNM, we represent relation r

	Greek (EL)			English (EN)			Spanish (ES)			French (FR)			Japanese (JA)		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
KGCmono	23.6	49.0	31.9	18.8	43.0	26.9	22.1	50.1	31.4	24.0	50.4	32.8	26.4	49.4	34.4
KG-BERT	17.3	40.1	27.3	12.9	31.9	21.0	21.9	54.1	34.0	23.5	55.9	35.4	26.9	59.8	38.7
KEnSb(RotatE)	27.5	56.5	-	14.4	39.6	-	25.2	62.6	-	22.3	60.6	-	32.9	64.8	-
KGCunion	25.1	56.2	35.0	18.3	43.7	26.4	22.6	52.7	32.3	26.0	54.7	35.8	29.3	55.9	38.6
Jaccard	25.1	57.8	35.7	18.5	45.0	27.2	23.0	52.3	32.7	27.6	58.0	37.6	32.9	59.8	41.9
Asymmetric	26.6	58.9	37.8	19.7	45.2	28.0	25.6	56.0	35.6	29.5	58.4	39.1	33.5	59.9	42.2
SoftAsym	32.7	61.0	42.5	20.8	45.9	29.1	26.9	55.9	36.4	30.7	59.3	40.3	34.7	61.7	43.9
Asym+Text	53.8	88.1	66.4	35.5	65.4	46.0	48.8	82.8	61.1	49.5	83.4	61.7	52.4	78.9	61.8
SoftAsym+Text	**57.6**	**88.4**	**69.0**	**37.2**	**66.1**	**47.4**	**53.0**	**84.4**	**64.5**	**52.9**	**84.9**	**64.5**	**53.2**	**80.9**	**62.9**
Unseen test set															
KGCunion	21.6	50.5	30.8	16.9	41.9	24.9	20.2	49.9	29.7	22.4	51.1	32.2	25.1	50.4	33.9
Jaccard	20.1	52.5	30.4	16.9	43.0	25.4	19.8	48.8	29.3	23.0	54.3	33.1	26.7	54.2	35.7
Asymmetric	20.3	52.8	31.3	17.5	43.0	25.7	21.9	52.5	31.8	24.6	54.3	34.4	27.1	53.8	35.8
SoftAsym	25.5	55.3	35.6	18.0	43.5	26.4	22.2	52.0	31.8	24.7	54.9	34.8	27.7	55.5	36.9
Asym+Text	48.7	86.2	62.2	33.2	63.9	43.9	45.8	81.3	58.5	45.4	81.6	58.4	47.0	75.6	57.0
SoftAsym+Text	**52.6**	**86.5**	**65.0**	**34.9**	**64.6**	**45.3**	**50.0**	**82.9**	**62.0**	**49.2**	**83.2**	**61.5**	**47.7**	**77.8**	**58.0**
Seen test set															
KGCunion	45.3	89.3	59.0	48.3	82.0	59.2	48.8	83.1	60.6	57.3	86.3	67.7	53.1	87.6	65.6
Jaccard	54.0	88.7	66.3	54.4	87.8	65.8	57.1	90.2	68.9	67.3	90.3	76.0	68.3	91.6	77.2
Asymmetric	62.7	94.0	75.5	68.5	94.8	78.4	65.4	94.6	76.8	72.4	94.2	79.8	70.2	94.7	78.8
SoftAsym	74.7	94.0	82.2	81.7	98.8	87.9	77.2	97.8	85.7	82.6	97.5	88.3	75.2	96.9	83.8
Asym+Text	83.3	99.3	90.3	85.9	99.4	91.7	81.6	99.3	89.3	**85.2**	98.6	**91.1**	82.9	97.8	89.4
SoftAsym+Text	**86.7**	**99.3**	**92.1**	**87.8**	**100.0**	**93.1**	**86.0**	**99.8**	**92.0**	*84.5*	**99.5**	*90.9*	**84.8**	**98.5**	**90.6**

Table 2: DBP5L, EA=30%, RA=0%, KGC performance. GRAPHTEXT methods. **Best,** *second-best* numbers.

as the concatenation $[\boldsymbol{r}, \overrightarrow{S(r)}, \overrightarrow{O(r)}]$ and compare $r_l, r_{l'}$ using the cosine of the concatenated representations in the entity-aware relation matching protocol of RNM. Appendix D discusses hyperparameter tuning and other implementation details.

5.4 Evaluation policies

KGC evaluation: We use standard evaluation framework (as described in Section 2) and rank candidates o for test instances $(s, r, ?)$, with gold o^* known. We report filtered MRR, HITS@1 and HITS@10 (abbreviated H@1, H@10). The **seen** test subset refers to those facts that are already seen at train time, but in a KG of a different language. I.e., a test fact (s_l, r_l, o_l) is in seen test if $\exists(s_{l'}, r_{l'}, o_{l'}) \in \text{KG}_{l'}(train)$ s.t. $s_l \equiv s_{l'}$, $o_l \equiv o_{l'}$ and $r_l \equiv r_{l'}$, even though the model may/may not know these alignments. All other test facts are in the **unseen** test subset. The seen split tests the ability to memorize known facts in one language and align them to another language; the unseen split tests a system's inference capability – those facts are likely not read in *any* language at train time.

EA and RA evaluation: Test instance $(e_l, \equiv, ?_{l'})$ is regarded as a task of ranking $e_{l'}$s, using the cosine distance between the entity embeddings of the language pair. RA is evaluated similarly, by ranking $r_{l'}$s for the query $(r_l, \equiv, ?_{l'})$. We calculate H@1 and H@10 on the ranking. Recognizing exact name match pitfalls in EA (see Section 3), we report on all test instances and also on the subset of instances without text match.

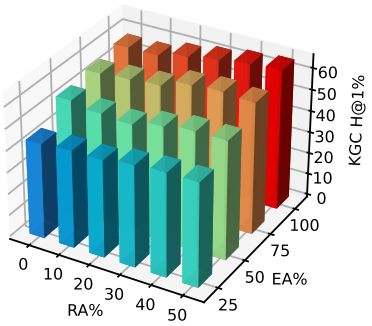

Figure 3: KGC Hits@1 variation as the percentage of EA and RA revealed at training time are jointly varied.

5.5 Results: KGC performance

Table 2 reports KGC performance for DBP5L. DBP15K and OpenEA show similar patterns, and are relegated to Appendix E.

GRAPHONLY methods: All methods outperform KGCMONO, since it does not perform multilingual training. KGCUNION obtains much better scores because of combining different language KGs. Both baselines are outperformed by all ALIGNKGC variants. Jaccard performs well for seen facts. Asymmetric removes false positives in RA, suffered by (symmetric) Jaccard. But it has a slight negative effect on unseen facts, likely because of removal of some reasonable inferences. Making SO-overlap soft and trainable enables better learning of embeddings and implication scores.

LangPair	Text		SoftAsym +Text		RAGA		RNM		RDGCN	
	All	!M	All	!M	All	!M	All	!M	All	!M
EL-EN	83.8	82.6	**90.3**	*89.6*	75.8	75.5	74.9	74.1	71.3	70.1
EL-ES	83.3	81.9	**91.1**	*90.5*	79.8	79.6	79.4	78.0	74.7	73.5
EL-FR	82.1	80.8	**90.9**	*90.3*	69.1	69.0	72.4	70.7	72.7	71.3
EL-JA	74.8	74.6	**89.3**	*89.3*	64.4	64.4	68.3	68.1	64.4	64.2
JA-EN	76.5	76.4	**88.5**	*88.5*	59.1	59.2	64.5	64.4	58.2	58.2
JA-ES	74.3	74.2	**88.9**	*88.8*	58.3	58.3	65.0	64.9	60.0	59.8
JA-FR	73.9	73.7	**88.9**	*88.9*	64.5	64.4	70.6	70.5	60.2	60.1
ES-FR	89.5	76.4	**96.1**	*91.7*	80.9	80.8	84.9	72.2	87.1	74.5
ES-EN	93.3	86.5	**96.8**	*94.1*	85.7	85.7	88.0	79.2	87.8	78.8
EN-FR	90.5	81.0	**95.3**	*91.2*	77.0	76.7	81.2	69.9	83.2	71.1
AVG	82.2	78.8	**91.6**	90.3	71.5	71.4	74.9	71.2	72.0	68.2

(!M = no exact match)

Table 3: DBP5L, EA H@1 performance.

Each successive model enhancement — Jaccard, Asymmetric, and SoftAsym — achieves progressively better performance. The differences in seen vs. unseen are along expected lines – models successfully extrapolate a seen fact in one language to another language. We report KEnS scores from its best reported setting, KEnS$_b$(RotatE). Although it has strictly more information than ALIGNKGC, with EA=RA=100%, it performs even worse than KGCUNION. We attribute this to 1) a stronger KGC baseline of ComplEx (with high negative sampling), compared to KEnS's use of RotatE with only one negative sample, and 2) ALIGNKGC's hard alignment policy of providing aligned entities in one unified KG with a single embedding, instead of KEnS's approach of finding nearest entities in each KG separately and then creating an ensemble.

GRAPHTEXT methods: Unsurprisingly, when entity surface forms are available, the methods see a huge jump in performance. This is because textual information helps with EA (especially in unseen set), which, in turn, helps KGC. It also outperforms KG-BERT, a GRAPHTEXT KGC approach.

Effect of the number of seed EAs and RAs: We vary the percentage of revealed seed relation alignments (RA%) and entity alignments (EA%) on x and y axes, and observe test KGC H@1 performance on the z axis (Figure 3; color changes with bar height). Predictably, increasing either EA% or RA% improves KGC. At any RA% value, increase in EA% has a powerful impact. At low EA%, increasing RA% has limited KGC impact. In this setting, entities in different languages have limited connections, and so the KGs are sparsely connected. Increasing EA% leads to collapse of entity nodes, increasing the number of seen triples

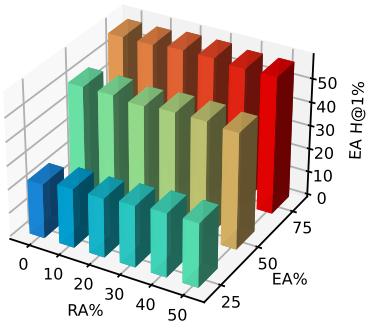

Figure 4: EA Hits@1 variation as the percentage of EA and RA revealed at training time are jointly varied.

making the graph denser, and likely making it easier for ALIGNKGC to reason with. Moreover, the model is able to infer many relation IDs as synonymous when more EAs are known, due to the explicit incentive provided via asymmetric loss — these improve downstream KGC performance.

5.6 Results: EA performance

Table 3 shows EA performance for DBP5L. DBP15K and OpenEA results are shown in Appendix F. We also compare against performance observed from the original codes for the recent systems RAGA (Zhu et al., 2021a) (EA) and RNM (Zhu et al., 2021b) (EA and RA).

Translate+mBERT, with mBERT fine-tuned from training EAs, is a formidable baseline, in fact, better than RNM and RDGCN. However, SoftAsym+Text clearly adds further value to Translate+mBERT, establishing a new state of the art among known EA methods. Figure 4 shows test EA hits@1 against EA% and RA% (color changes with bar height). We observe that (predictably) increasing either EA% and RA% improves KGC performance.

At any RA% value, increase in EA% has a powerful impact on both EA performances. At low

Methods↓	Rare (<500)		Freq (≥500)	
	H@1	H@3	H@1	H@3
KgcUnion	35.6	47.9	79.8	93.2
Jaccard	39.4	53.8	80.2	92.8
Asymmetric	37.9	51.8	81.9	94.0
SoftAsym	41.8	55.4	84.4	94.1
RNM	61.3	72.0	85.8	93.9
Asym+Text	*71.2*	*80.7*	**86.8**	**96.6**
SoftAsym+Text	**72.8**	**82.2**	**86.8**	*95.9*

Table 4: DBP5L, RA performance. Relations occurring in <500 and ≥ 500 triples are evaluated separately.

EA%, increasing RA% does add information on fact similarity across KGs, thus improving EA score, but the numbers remain low, due to the overall difficulty of the task.

At high EA% values, increasing RA% does not further improve EA performance. This is because EA supervision generally targets more frequent relations (Figure 6). Even at low RA%, this enables adequate alignment of frequent relations, leading further to adequate alignment of the entities attached to such relations.

5.7 Results: RA performance

Table 4 compares (unweighted average over language pairs) RA performance of various methods on DBP5L. Per-language-pair drill-down for all three data sets can be found in Appendix G. We split relations into rare (SO-set has under 500 SO pairs) and frequent ($\geq$500 SO pairs). Naturally, RA predictions involving frequent relations are expected to be generally more accurate. GRAPH-ONLY methods perform worse than GRAPHTEXT methods as expected, with SoftAsym rising above others. Within GRAPHTEXT methods, Asym+Text and SoftAsym+Text improve considerably over RNM, particularly for rare relations and H@1.

5.8 Discussion

Overall, we see benefits from both our multi-tasking objective and their associated data sets. The previous experiments already provide some level of ablation studies. First, Table 2 shows KGC performance for DBP5L, with a fixed 30% of gold EAs exposed for training. KGCmono does not use this EA information. KGCunion does, but it is still pure KGC, with no additional EA or RA loss terms. Jaccard, Asymmetric and SoftAsym use EA and RA loss terms in the objective. Their '+Text' counterparts also use BERT-based EA loss term. Second, we consider additional training data beyond KGC training, in the form of EA and RA training. We studied this effect by controlling the percentage of

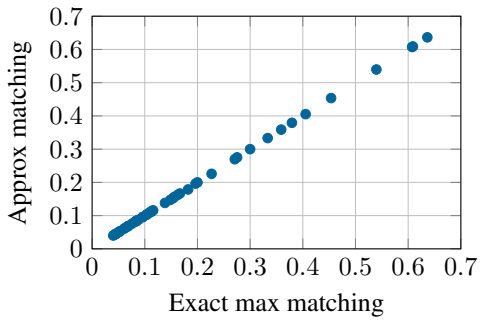

Figure 5: Our fast approximation for soft relation overlap is accurate. The x-axis represents the optimal matching and the y-axis our approximation.

EA and RA training folds exposed to ALIGNKGC variants — e.g., Figure 3 for the effect on KGC and Figure 4 for the effect on (test fold) EA. We highlight that the special case of 0% training exposure also corresponds to removing EA or RA tasks from the multi-task objective.

5.9 Soft overlap approximation quality

We described in Section 4.3 our fast approximation for maximal matching, required to evaluate the overlap between $\mathrm{SO}(r_1)$ and $\mathrm{SO}(r_2)$. In Figure 5 we show a scatter plot of a sample of relation pairs; the x-axis is the true optimal overlap found via the Hungarian matching algorithm, and the y-axis is our approximation. We can see that the approximation is very close to the optimal.

6 Conclusion

We presented ALIGNKGC, a system that jointly learns to complete multiple monolingual KGs in different languages and align their entities and relations. To our knowledge, these three tasks have never been unified before. ALIGNKGC operates on the KG constructed by taking a union of all monolingual KGs, and extends KGC models to use novel EA and RA loss terms. In extensive experiments, ALIGNKGC significantly improves KGC accuracy, as well as alignment accuracy on three datasets, underscoring the value of joint alignment and completion.

Acknowledgments: Mausam is supported by grants from Huawei, IBM, Google, Bloomberg, Verisk, and 1MG, a Visvesvaraya faculty award by Govt. of India, and Jai Gupta Chair professorship. We thank the IITD HPC facility for compute resources. Soumen Chakrabarti is partly supported by grants from Cisco, Google, IBM, and SERB.

7 Limitations

There remain a few items for continued research and enhancement. (1) We have focused on interlingual alignment between KGs in different languages. Aligning KGs with different domains and 'styles', such as Wikipedia and IMDB, even if in the same language, may present different challenges. (2) The introduction of 'silver' relation pairs (eqns. 5, 10, 15) slows down RA loss optimization. Suitable sampling of RA loss terms may lead to better loss optimization and faster convergence. (3) ALIGN-KGC effectively uses tailored representations of relations for different tasks. For KGC it uses single vectors $\boldsymbol{r}$, whereas for RA it uses soft-SO-sets $\mathrm{SO}(r)$. This prompts us toward a more unified representation of KG elements.

References

Max Berrendorf, Ludwig Wacker, and Evgeniy Faerman. 2021. A critical assessment of state-of-the-art in entity alignment. In *ECIR*.

Rahul Bhagat, Patrick Pantel, and Eduard Hovy. 2007. LEDIR: An unsupervised algorithm for learning directionality of inference rules. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 161–170, Prague, Czech Republic. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *NeurIPS*.

Yixin Cao, Zhiyuan Liu, Chengjiang Li, Zhiyuan Liu, Juanzi Li, and Tat-Seng Chua. 2019. Multi-channel graph neural network for entity alignment. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1452–1461, Florence, Italy. Association for Computational Linguistics.

Muhao Chen et al. 2017. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *IJCAI*, page 1511–1517.

Muhao Chen et al. 2021. Cross-lingual entity alignment with incidental supervision. In *EACL Conference*.

Xuelu Chen, Muhao Chen, Changjun Fan, Ankith Uppunda, Yizhou Sun, and Carlo Zaniolo. 2020. Multilingual knowledge graph completion via ensemble knowledge transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3227–3238, Online. Association for Computational Linguistics.

Marco Cuturi. 2013. Sinkhorn distances: Lightspeed computation of optimal transport. In *NeurIPS*, pages 2292–2300.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *AAAI Conference*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Timothée Lacroix, Nicolas Usunier, and Guillaume Obozinski. 2018. Canonical tensor decomposition for knowledge base completion. In *ICML*.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In *ICML*, pages 3744–3753.

Dekang Lin and Patrick Pantel. 2001. DIRT: Discovery of inference rules from text. In *SIGKDD Conference*, pages 323–328.

Zhiyuan Liu, Yixin Cao, Liangming Pan, Juan-Zi Li, and Tat-Seng Chua. 2020. Exploring and evaluating attributes, values, and structure for entity alignment. *EMNLP*, abs/2010.03249.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020a. Relational reflection entity alignment. *CIKM*.

Xin Mao, Wenting Wang, Huimin Xu, Yuanbin Wu, and Man Lan. 2020b. Relational reflection entity alignment. In *CIKM*, page 1095–1104.

Gonzalo Mena, David Belanger, Scott Linderman, and Jasper Snoek. 2018. Learning latent permutations with gumbel-sinkhorn networks. *ICLR*.

Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. 2012. PATTY: A taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1135–1145, Jeju Island, Korea. Association for Computational Linguistics.

Chirag Pabbaraju and Prateek Jain. 2019. Learning functions over sets via permutation adversarial networks. *arXiv preprint arXiv:1907.05638*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Xiaofei Shi and Yanghua Xiao. 2019. Modeling multi-mapping relations for precise cross-lingual entity alignment. In *EMNLP/IJCNLP*.

Harkanwar Singh, Soumen Chakrabarti, Prachi Jain, Sharod Roy Choudhury, and Mausam. 2021. Multilingual knowledge graph completion with joint relation and entity alignment. In *AKBC Conference*, volume abs/2104.08804.

Lichao Sun, Kazuma Hashimoto, Wenpeng Yin, Akari Asai, Jia Li, Philip S. Yu, and Caiming Xiong. 2020a. Adv-bert: Bert is not robust on misspellings! generating nature adversarial samples on bert. *ArXiv*.

Zequn Sun, Muhao Chen, and Wei Hu. 2021. Knowing the no-match: Entity alignment with dangling cases. *ACL*.

Zequn Sun, Wei Hu, and Chengkai Li. 2017. Cross-lingual entity alignment via joint attribute-preserving embedding. In *ISWC*, pages 628–644.

Zequn Sun, JiaCheng Huang, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. 2019a. Transedge: Translating relation-contextualized embeddings for knowledge graphs. In *SEMWEB*.

Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020b. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proceedings of the VLDB Endowment*, 13(11):2326–2340.

Zequn Sun et al. 2018. Bootstrapping entity alignment with knowledge graph embedding. In *IJCAI*, volume 18, pages 4396–4402.

Zequn Sun et al. 2020c. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. *AAAI*, 34:222–229.

Zhiqing Sun, Zhihong Deng, Jian-Yun Nie, and Jian Tang. 2019b. RotatE: Knowledge graph embedding by relational rotation in complex space. *ICLR*, abs/1902.10197.

Xiaobin Tang et al. 2020. BERT-INT: A BERT-based interaction model for knowledge graph alignment. In *IJCAI*, pages 3174–3180.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *ICML*, pages 2071–2080.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019a. Relation-aware entity alignment for heterogeneous knowledge graphs. *IJCAI*.

Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, and Dongyan Zhao. 2019b. Jointly learning entity and relation representations for entity alignment. In *EMNLP*.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. KG-BERT: BERT for knowledge graph completion. *ArXiv*, abs/1909.03193.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. 2017. Deep sets. *NeurIPS*.

Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2017. Iterative entity alignment via joint knowledge embeddings. In *IJCAI*.

Renbo Zhu, Meng Ma, and Ping Wang. 2021a. RAGA: relation-aware graph attention networks for global entity alignment. *PKDD*, abs/2103.00791.

Yao Zhu, Hongzhi Liu, Zhonghai Wu, and Yingpeng Du. 2021b. Relation-aware neighborhood matching model for entity alignment. In *AAAI Conference*.

Joint Completion and Alignment of Multilingual Knowledge Graphs (Appendix)

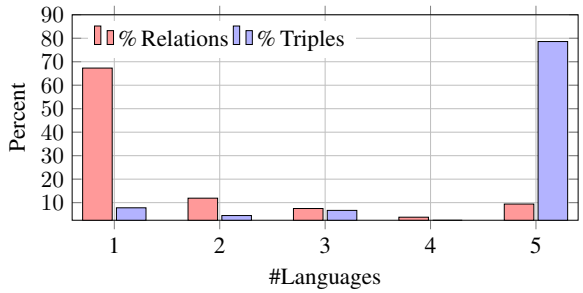

Figure 6: Fraction of relation labels and fact tuples with the number of languages they appear in (DBP5L).

A Example of KGC, EA and RA synergy

We first motivate the need for multi-tasking KGC, EA and RA with a toy example from English (En) and Greek (El) slices of DBPedia.

EA helps RA Suppose a KGC+alignment system is given the equivalences between English (En) and Greek (El) entity names:

- `Botafogo_de_Futebol_e_Regatas` ≡ Μποταφόγκο_ντε_Φουτεμπόλ_ε_Ρεγκάτας (a football club)
- `Brazil` ≡ Βραζιλία

Now suppose the system sees these two triples in En and El KGs, respectively:

En: (`Botafogo_de_Futebol_e_Regatas`, `home_arena`, `Brazil`)

El: (Μποταφόγκο_ντε_Φουτεμπόλ_ε_Ρεγκτας, σπίτι αρένα, Βραζιλία)

If this pattern occurs often enough, the system can infer the relation alignment "`home_arena`" = "σπίτι αρένα". This is an example of inferring RA from EA.

EA, RA helps KGC Suppose the system knows that

- `Rot-WeissEssen` (English) ≡ Ροτ_Βάις_Έσσεν (Greek name of a football club)
- `Essen` (English) ≡ Έσσεν (Greek name of a town in Germany)
- (`Rot-WeissEssen`, `home_arena`, `Essen`) holds in the English KG

Then the system can potentially infer (Ροτ_Βάις_Έσσεν, σπίτι αρένα, Έσσεν) in the Greek KG. This is an example of EA and RA helping KGC.

KGC helps EA Now suppose the KGC system has already learnt the (soft) inference pattern

([club], `home_arena`, [city]) and ([city], `country`, [country])

$\Longrightarrow$ ([club], `home_arena`, [country])

where `home_arena` and `country` (Χώρα in Greek) are assumed to be 'universal' across languages.

Now if we know that (Έσσεν, Χώρα, Γερμανία) holds in the Greek KG, we can apply the above rule to infer (Ροτ_Βάις_Έσσεν, σπίτι αρένα, Γερμανία). If the corresponding fact was already in the English KG as (`Rot-WeissEssen`, `homeArena`, `Germany`), then we could infer the EA Γερμανία ≡ `Germany`.

Overall, all three tasks are synergistic, and we posit (and verify) that a joint model can produce better results on all three tasks.

B Data set details

B.1 DBP5L

Most suited for our purpose is the recently-released DBP5L benchmark (Chen et al., 2020), which is derived from DBPedia in five languages: English (En), Greek (El), Spanish (Es), Japanese (Ja) and French (Fr). En is the most well-populated, with 5–7 times more relations than other KGs. Figure 6 shows that a majority of the relation labels have associated string surface forms in only one of five languages (usually English). However, relations with monolingual aliases account for only 8% of fact triples. Meanwhile, almost 80% of fact triples use a relation label that has aliases in all five languages. For the KGC task, we use 60-30-10 splits of the KG triples into train-dev-test folds, and combine the train sets of all source languages for training. In addition to KGC triple folds, DBP5L provides gold EA pairs. We randomly sample a fraction ('EA%') of these seed entity alignments for training; the rest are used for testing. Over 65% of the relation labels have associated string surface forms in only one of five languages (usually English). However, relations with monolingual aliases account for only 8% of fact triples. Meanwhile, almost 80% of fact triples use a relation label that has aliases in all five languages. Because DBPedia uses a uniform relation vocabulary that is normalized across all languages, we adapt it slightly to assess the RA capabilities of various models. A randomly sampled fraction ('RA%') of relations are exposed as aligned across all languages for training; the rest are named apart for each language. An experiment

DBP5L										
el-en	el-es	el-fr	el-ja	en-es	en-fr	en-ja	es-fr	es-ja	fr-ja	avg
6.81	8.23	7.18	0.58	55.06	53.04	0.42	57.8	0.4	0.31	19.0

DBP15K			
fr-en	ja-en	zh-en	avg
50.04	2.51	3.47	18.7

OpenEA v1.1				
v1		v2		
en-de	en-fr	en-de	en-fr	avg
61.81	58.25	61.11	59.68	60.2

Table 5: Percentage of EA pairs $(e_l, e_{l'})$ in the test folds with exact text match, i.e., $\text{text}(e_l) \equiv \text{text}(e_{l'})$.

is thus parameterized by (EA%,RA%). Our default KGC setting is (30%,0%) for training. We also sweep (EA%,RA%) ranges to measure the effect of partial supervision. No asymmetric gold relation implication $r_1 \Rightarrow r_2$ are available, only equivalences of the form $r_1 \equiv r_2$.

B.2 DBP15K and OpenEA

A few recent EA datasets such as DBP15K (Sun et al., 2017), OpenEA (Sun et al., 2020b) and DB2.0 (Sun et al., 2021) provide gold EAs between *two* languages at a time. They offer fewer language pairs (DBP15K: fr-en, ja-en, zh-en; OpenEA: en-de, en-fr) than DBP5L (all $\binom{5}{2} = 10$ pairs). They do not provide direct support for synergistic training and evaluation of RA and KGC at the same time. Therefore, we draw our own samples for KGC training and evaluation.

DBP15K (Sun et al., 2017) has been criticized (Liu et al., 2020; Berrendorf et al., 2021) for the "exact-match" problem; OpenEA (Sun et al., 2020b, version 1) and DBP5L (Chen et al., 2020) have similar problems. Specifically, a sizeable fraction (Table 5) of gold EA pairs $(e_l, e_{l'})$ have the *exact same entity name*, i.e., $\text{text}(e_l) \equiv \text{text}(e_{l'})$, even if in different languages like en-fr or en-de. The problem is severe for languages sharing charsets and almost non-existent for language pairs that use different charsets. A drastic measure is to replace entity names with opaque IDs (Liu et al., 2020; Sun et al., 2020b, version 2), but this also destroys the legitimate ability of mBERT to map genuine translations to similar embeddings. We will therefore report EA performance separately for all test-fold EA pairs, and for the subset of test-fold EA pairs where there is no exact text match. BERT/mBERT is not robust to character-level corruption (Sun et al., 2020a), so filtering exact matches is adequate.

C Text processing

Most recent, competitive EA methods (Wu et al., 2019a; Tang et al., 2020; Zhu et al., 2021b) need initial node (entity) representation vectors for a GCN (or GCN-like) graph propagation step. RDGCN (Wu et al., 2019a) and RNM (Zhu et al., 2021b) obtain this node embedding by translating the entity name to English, then looking up the GloVE embedding (Pennington et al., 2014) of the English name. BERT-INT (Tang et al., 2020) directly applies the multilingual mBERT pretrained network on the entity name (in any language). These are not equivalent — because mBERT's wordpiece dictionary is seriously deficient for most low-resource languages, translation is a significantly better option. While Google Translate is commonly used, it is a network service whose quality can change through time. For reproducibility, we use EasyNMT[2] with Facebook's model `m2m_100_1.2B`. For uniformity across competing systems, we use mBERT[3] throughout, even if the input is English (for which mBERT's wordpiece dictionary is adequate). For each competing method, we use its most favorable text encoding method among translate+GloVE, mBERT, and translate+mBERT. Since KGC is most useful around nascent entities, we use the entity surface form(s) as $\text{text}(e)$ and not, e.g., its full Wikipedia description or infoboxes. We append a fully connected layer to mBERT's `[CLS]` embedding to project it from 768 to 300 dimensions and train that plus fine-tune mBERT on entity surface forms and gold EAs, to get 300-dim entity feature vectors.

By virtue of the available datasets, we do not have surface forms for relations in different languages, hence GRAPHTEXT setting is evaluated with only entity aliases, even though ALIGNKGC is general.

D Hardware, software, hyperparameters

Experiments run on Intel Xeon servers running Ubuntu 20.04 with nVidia A6000 (48GB) and Titan Xp (12GB) GPUs. Our programs were

[2] `https://github.com/UKPLab/EasyNMT`, commit hash 5ea48f5f...

[3] `https://huggingface.co/bert-base-multilingual-cased`

written using pytorch 1.10.2 on python 3.9.7. RNM and RDGCN codes were downloaded from their published sources, https://github.com/Peter7Yao/RNM and https://github.com/StephanieWyt/RDGCN, and adapted for comparison.

We use the Adagrad optimzer with a batch size of 500, and fine tune hyper-parameters on a dev set. We choose θ=0.01 for Jaccard. We select among the following sets of hyper-parameter values: learning rate from {0.2,0.4,0.6,0.8,1}, α from $\{0.002, 0.02, 0.2\}$, β from $\{1, 5, 10, 50, 100, 500, 1000, 2000\}$, and γ from $\{1, 10, 50, 100, 200, 500, 1000, 2000\}$. Our best choices are learning rate=0.8, α=0.02, β=500, and γ=1000. We sample 2000 negative instances for each positive triple when training ComplEx. We sample initial entity and relation embeddings via $\mathcal{N}(0, 0.05)$. As a form of curriculum, we let relation alignments stabilize over a few iterations and then make the equivalence scores trainable.

E DBP15K and OpenEA KGC performance

Tables 6 and 7 show KGC performance on DBP15K and OpenEA, respectively. These data sets were meant for EA, not KGC, so we had to sample train, dev and test folds for KGC. We show the KGCUNION baseline and the best ALIGNKGC variants. The trends are the same as in case of DBP5L.

F DBP15K and OpenEA EA performance

Tables 8 and 9 show EA performance on DBP15K and OpenEA, respectively. As with DBP5L, Text=Translate+mBERT, by itself, forms a strong baseline (as also reported in BERT-INT (Tang et al., 2020)). However, SoftAsym+Text improves considerably beyond Text alone, and establishes new state of the art EA performance over RNM and RDGCN.

G RA performance drill-down

We drill down into RA performance over all language pairs in all the three datasets. The results are shown in Tables 10, 11 and 12. For RA evaluation (see Section 5.3), we select 0.75 as the cosine threshold to introduce silver alignments. Gold+silver alignments are used to augment the cosine similarity scores for entity-aware relation matching as in RNM (Zhu et al., 2021b, Eqn. (10)) with λ_r tuned to 200 via validation. As usual, GRAPHTEXT methods are better than GRAPH-ONLY methods. Over the majority of language pairs, SoftAsym+Text is the leading method, followed by Asym+Text and then RNM.

H Stability across model initializations

Table 13 shows the standard deviation over three runs (with different random seeds for initializing model weights) corresponding to ALIGNKGC. It is under 1% much of the time over all the three measurements: H@1, H@10 and MRR. This gives an indication of the stability of ALIGNKGC.

I SoftOv generalizes discrete overlap

Proposition 1. Soft overlap in Definition 4 generalizes discrete set overlap in Definition 2.

Proof sketch (details deferred to final version): If there are E entities in the universe, represent each entity 'embedding' as a 0/1 vector of E dimensions, specifically, as a 1-hot vector. Then the bipartite graph defined by Eqn. (13) degenerates to edges with weight 1 for identical SO-pairs on both sides, and weight 0 otherwise. Then the value of the maximal matching in this graph is exactly the size of the intersection of the two SO-sets of the two concerned relations.

J Differentiable set representations

Given a finite set $X = \{x_i\}$ where $x_i \in \mathcal{X}$, Zaheer et al. (2017) showed that any set function $f(X)$ that is invariant to permutations of X can be expressed as $\rho\left(\sum_{x \in X} \phi(x)\right)$ for suitable transformations ρ and ϕ. In practice, capturing correlations between features of xs requires unreasonably deep/wide networks for ρ and ϕ. Pabbaraju and Jain (2019) proposed to present X, after applying adversarial permutations, to an order-sensitive recurrent network which can better capture correlations among xs, while using lower network capacity. The RNN must reduce variability of its set encoding in the face of these permutations. Lee et al. (2019) replaced the RNN with a transformer network.

K Matching via matrix scaling

Consider two sets $\{x_i : i \in [I]\}$ and $\{y_j : j \in [J]\}$ with a metric $d_{ij} = d(x_i, y_j)$. Let $T_{ij} \in \mathbb{R}_+$ be a *transportation matrix*, subject marginal constraints to $\sum_i T_{ij} = c_j$ for all j and $\sum_j T_{ij} = r_i$ for all i. Often, $r_i = c_j = 1$ or $r_i = 1/I$ and

	DBP15k-FR-EN						DBP15k-JA-EN						DBP15k-ZH-EN					
	French (FR)			English (EN)			Japanese (JA)			English (EN)			Chinese (ZH)			English (EN)		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
KGCUnion	26.0	56.1	36.0	27.3	57.6	37.4	27.5	52.5	35.8	27.7	54.9	36.8	21.0	43.9	28.7	25.6	51.5	34.4
SoftAsym	27.4	57.0	37.2	28.7	58.9	38.6	28.7	53.1	36.9	28.7	54.6	37.2	22.5	45.4	30.2	26.8	51.9	35.3
SoftAsym+Text	**38.5**	**68.6**	**48.9**	**39.8**	**68.8**	**49.9**	**36.4**	**62.3**	**45.3**	**35.5**	**61.2**	**44.0**	**29.4**	**53.7**	**37.6**	**30.4**	**55.4**	**39.0**
Unseen test set																		
KGCUnion	25.7	55.8	35.7	27.0	57.2	37.1	27.1	52.1	35.4	27.4	54.6	36.5	20.1	42.8	27.7	25.0	50.7	33.7
SoftAsym	27.0	56.6	36.8	28.1	58.5	38.2	28.1	52.6	36.3	28.2	54.3	36.8	21.0	44.1	28.7	25.5	50.9	34.1
SoftAsym+Text	**38.0**	**68.3**	**48.5**	**39.3**	**68.4**	**49.4**	**35.8**	**61.8**	**44.7**	**35.0**	**60.8**	**43.5**	**27.8**	**52.4**	**36.1**	**29.0**	**54.4**	**37.7**
Seen test set																		
KGCUnion	57.6	90.9	69.1	51.7	90.7	65.1	61.7	88.3	70.5	56.8	83.2	66.2	50.5	82.0	62.2	49.3	86.3	61.4
SoftAsym	71.7	97.0	82.6	78.8	97.5	85.5	78.7	95.7	85.4	76.8	89.5	81.5	73.5	93.0	80.9	77.5	93.4	83.5
SoftAsym+Text	**88.9**	**100.0**	**93.2**	**89.8**	**100.0**	**94.8**	**88.3**	**97.9**	**92.4**	**83.2**	**96.8**	**88.9**	**83.0**	**97.0**	**89.2**	**87.2**	**98.2**	**91.9**

Table 6: DBP15K, revealed EA=30%, RA=0%, KGC performance.

	OpenEA-EN-DE-V1						OpenEA-EN-DE-V2						OpenEA-EN-FR-V1						OpenEA-EN-FR-V2					
	English (EN)			German (DE)			English (EN)			German (DE)			English (EN)			French (FR)			English (EN)			French (FR)		
	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
KGCUNION	15.2	35.3	21.9	22.2	43.0	28.8	19.5	46.5	28.3	29.5	55.7	38.4	32.9	54.5	40.3	31.4	53.2	39.0	43.3	71.2	53.0	42.6	68.5	51.6
SoftAsym	15.8	35.2	21.9	22.5	43.7	29.5	20.0	46.8	28.8	28.4	54.1	37.0	33.9	55.0	40.9	31.5	53.6	39.2	43.2	70.4	52.7	42.0	69.1	51.3
SoftAsym+Text	**37.1**	**55.8**	**43.4**	**39.9**	**61.4**	**47.4**	**43.0**	**68.0**	**51.7**	**46.7**	**73.3**	**56.3**	**47.2**	**65.9**	**54.0**	**48.2**	**67.2**	**55.1**	**59.1**	**82.0**	**67.4**	**59.5**	**82.8**	**68.1**
Unseen test set																								
KGCUNION	14.3	34.2	20.9	21.3	42.1	27.9	18.5	45.5	27.3	28.9	55.1	37.8	31.8	53.4	39.2	30.5	52.1	37.9	42.3	70.4	52.1	41.5	67.6	50.6
SoftAsym	14.6	34.0	20.6	21.4	42.7	28.4	18.8	45.8	27.6	27.5	53.5	36.2	32.4	53.8	39.5	30.0	52.3	37.7	42.1	69.6	51.7	40.6	68.2	50.1
SoftAsym+Text	**36.0**	**54.9**	**42.4**	**38.9**	**60.8**	**46.5**	**41.9**	**67.3**	**50.7**	**46.0**	**72.9**	**55.7**	**45.9**	**64.9**	**52.8**	**46.9**	**66.3**	**53.9**	**58.0**	**81.5**	**66.5**	**58.2**	**82.2**	**67.1**
Seen test set																								
KGCUNION	62.8	96.5	74.2	74.7	95.4	83.2	69.2	94.8	78.7	71.2	95.0	79.4	72.4	93.7	80.9	62.5	91.7	74.0	79.2	96.6	86.6	74.7	97.0	83.4
SoftAsym	84.9	98.8	91.2	88.5	98.9	92.6	80.8	95.4	85.7	83.5	97.1	89.0	85.8	96.9	90.6	82.5	95.8	87.8	83.3	98.5	89.3	81.9	97.0	88.2
SoftAsym+Text	**96.5**	**100.0**	**97.8**	**94.3**	**100.0**	**97.1**	**95.4**	**99.4**	**97.4**	**94.2**	**100.0**	**97.0**	**96.1**	**100.0**	**97.7**	**91.7**	**99.2**	**94.8**	**95.5**	**100.0**	**97.7**	**97.0**	**99.6**	**98.3**

Table 7: OpenEA, revealed EA=30%, RA=0%, KGC performance.

LangPair	Text		SoftAsym +Text		RAGA		RNM		RDGCN	
	All	!M	All	!M	All	!M	All	!M	All	!M
fr-en	87.48	78.77	**90.31**	*84.86*	84.15	84.09	79.33	72.72	75.11	63.41
ja-en	64.29	63.40	**69.43**	*68.84*	63.89	63.88	62.29	62.05	50.44	49.88
zh-en	48.00	46.39	**55.10**	*53.82*	54.96	55.04	53.57	52.79	42.30	41.13
AVG	66.59	62.85	**71.61**	*69.17*	67.67	67.67	65.06	62.52	55.95	51.47

(!M = no exact match)

Table 8: DBP15K, EA H@1 performance.

LangPair	Text		SoftAsym +Text		RAGA		RNM		RDGCN	
	All	**!M**	**All**	**!M**	**All**	**!M**	**All**	**!M**	**All**	**!M**
EN-DE-15K-V1	87.1	72.6	**89.7**	*78.7*	73.5	73.5	74.2	67.4	76.5	62.6
EN-DE-15K-V2	87.2	72.1	**91.6**	*82.2*	85	85	82.2	68.7	79.8	66.5
EN-FR-15K-V1	87.3	78.7	**90.5**	*84.9*	70	70.1	70.3	69	70.1	61.7
EN-FR-15K-V2	91.4	84.9	**93.3**	*88.2*	89.5	89.4	83.7	76.7	84.8	76
AVG	88.2	77.1	91.3	83.5	79.5	79.5	77.6	70.5	77.8	66.7

(!M = no exact match)

Table 9: OpenEA, EA H@1 performance.

LangPair	Jaccard				Asymmetric				SoftAsym			
	Rare		Freq.		Rare		Freq.		Rare		Freq.	
	H@1	H@3	H@1	H@3	H@1	H@3	H@1	H@3	H@1	H@3	H@1	H@3
EL-EN	41.0	58.1	76.2	97.6	38.5	53.9	83.3	97.6	44.4	56.8	81.0	97.6
EL-ES	39.7	55.6	88.9	94.4	38.8	53.7	94.4	100.0	39.7	59.8	94.4	97.2
EL-FR	31.8	44.7	66.7	88.9	34.7	48.8	69.4	94.4	35.3	43.5	80.6	94.4
EL-JA	49.0	68.0	78.1	90.6	48.5	66.0	75.0	90.6	54.9	68.0	81.3	90.6
JA-EN	38.8	49.5	74.0	90.0	39.8	53.1	74.0	90.0	40.8	54.6	78.0	94.0
JA-ES	40.6	51.8	79.0	89.5	37.1	50.6	81.6	92.1	40.6	51.2	86.8	94.7
JA-FR	41.8	53.9	95.0	100.0	38.0	50.0	95.0	100.0	42.8	53.9	97.5	97.5
ES-FR	39.8	59.0	89.6	93.8	36.3	51.6	89.6	95.8	44.7	61.8	89.6	95.8
ES-EN	40.8	51.5	83.9	92.9	39.2	50.8	83.9	91.1	41.2	57.3	83.9	92.9
EN-FR	30.4	46.0	71.2	90.4	28.1	39.7	73.1	88.5	33.9	47.3	71.2	86.5
AVG	39.4	53.8	80.2	92.8	37.9	51.8	81.9	94.0	41.8	55.4	84.4	94.1
	RNM				**Asym+Text**				**SoftAsym+Text**			
EL-EN	58.1	72.2	90.5	97.6	71.4	85.5	92.9	100.0	74.4	85.9	95.2	100.0
EL-ES	69.2	79.4	94.4	97.2	79.9	86.0	94.4	100.0	83.2	88.8	94.4	100.0
EL-FR	53.5	62.9	75.0	94.4	62.4	73.5	83.3	97.2	62.4	74.1	83.3	94.4
EL-JA	74.8	83.5	81.3	90.6	82.5	88.8	84.4	93.8	83.0	89.8	84.4	93.8
JA-EN	54.6	64.8	86.0	96.0	61.7	75.0	82.0	94.0	63.8	76.5	82.0	94.0
JA-ES	46.5	58.8	84.2	92.1	56.5	67.1	84.2	94.7	57.7	74.7	84.2	94.7
JA-FR	70.2	75.0	92.5	92.5	78.9	88.0	92.5	100.0	79.3	87.0	92.5	97.5
ES-FR	75.5	84.5	89.6	95.8	82.6	87.9	91.7	95.8	84.2	87.0	91.7	95.8
ES-EN	64.2	75.8	89.3	94.7	77.7	85.4	85.7	98.2	80.0	86.9	85.7	98.2
EN-FR	46.9	63.4	75.0	88.5	58.9	69.6	76.9	92.3	60.3	71.0	75.0	90.4
AVG	61.3	72.0	85.8	93.9	*71.2*	*80.7*	*86.8*	**96.6**	**72.8**	**82.2**	**86.8**	*95.9*

Table 10: DBP5L, RA performance drill-down over all language pairs.

	fr-en				ja-en				zh-en			
	Rare		Freq.		Rare		Freq.		Rare		Freq.	
Models	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**
RNM	41.5	**50.9**	65.0	76.7	63.1	72.8	**79.2**	79.2	64.4	71.6	93.0	97.7
SoftAsym+Text	**45.0**	50.3	**71.7**	**80.0**	**68.1**	**78.9**	75.0	79.2	**66.3**	**74.1**	**96.5**	**100.0**

Table 11: DBP15K, RA performance drill-down over all language pairs.

	EN-DE-15K-V1				EN-DE-15K-V2				EN-FR-15K-V1				EN-FR-15K-V2			
	Rare		Freq.		Rare		Freq.		Rare		Freq.		Rare		Freq.	
Models	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**	**H@1**	**H@3**
RNM	69.9	78.9	92.3	92.3	68.2	80.7	90.9	**93.2**	71.2	81.0	94.1	94.1	72.2	81.9	94.7	94.7
SoftAsym+Text	**75.9**	**85.5**	92.3	92.3	**76.1**	**88.6**	90.9	90.9	**77.4**	**84.1**	**97.1**	**97.1**	**77.8**	**91.7**	94.7	**97.4**

Table 12: OpenEA, RA performance drill-down over all language pairs.

GREEK			ENGLISH			SPANISH			FRENCH			JAPANESE		
H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR
1.01	0.261	0.607	0.815	0.744	0.876	1.07	0.118	0.785	0.632	0.225	0.538	2.826	0.685	1.872

Table 13: DBP5L; standard deviation (absolute %) of full ALIGNKGC (SoftAsym+Text) over three random model initializations. Most of the time it is under 1% over all three measurements: H@1, H@10 and MRR.

$c_j = 1/J$. Our goal is to $\min_T \sum_{i,j} d_{ij} T_{ij}$. While this can be solved via linear programming, it is not clear how to backpropagate losses based on the optimal transportation back to networks that output $\{x_i\}, \{y_j\}$. (In our case, these set elements are functions of KG embedding vectors, through SO-sets.) Cuturi (2013) showed that if the objective is mildly augmented with an entropy term $\lambda H(T) = -\lambda \sum_{i,j} T_{ij} \log T_{ij}$, then the optimal transport can be approximated by iterative row and column scaling of (d_{ij}), which are differentiable operations. However, this is computationally expensive for large KGs.

L Exploiting relation descriptions

Along with entities, relations, too, come with textual aliases in some KGs like WikiData. If we denote the textual description of relation r using $t(r)$, we can also use $\mathrm{mlp}(\mathrm{TxtEmb}(t(r)))$ as additional intrinsic features of r. (This MLP is different from the one for entities.) Similar to entity alignment, if $b(r_l \Leftrightarrow r_{l'})$ is high, then we want $\mathrm{TxtEmb}(r_l) \approx \mathrm{TxtEmb}(r_{l'})$. Accordingly, we can assess another loss term $L_{\text{RA3}} =$

$$\sum_{r_l \equiv r_{l'}} b(r_l \Leftrightarrow r_{l'}) \left\| \mathrm{TxtEmb}(r_l) - \mathrm{TxtEmb}(r_{l'}) \right\|_1. \quad (18)$$

In ongoing work, we are preparing a new data set based on WikiData to evaluate the above extension.

M Other ways to use text signals

Beyond what is presented in the main paper, we explored some other ways to incorporate mBERT outputs, but these did not perform as well as the approach we chose in the end. One idea was to augment the soft SO signature with vectors output from BERT. Specifically, we redefine $\mathrm{SO}(r)$ as

$$\left\{ \left\langle \boldsymbol{s}, \mathrm{TxtEmb}(s)); \boldsymbol{o}, \mathrm{TxtEmb}(o) \right\rangle : (s,r,o) \in \mathrm{KG} \right\} \quad (19)$$

Relative weights balancing $\boldsymbol{e}$ and $\mathrm{TxtEmb}(e)$ may also help. This compound vector is input to an MLP which may be trained with mBERT weights pinned down, or also fine tuned. This suggests the alternative loss term

$$L_{\text{EA}} = \sum_{e_l \equiv e_{l'}} \left\| \mathrm{TxtEmb}(e_l) - \mathrm{TxtEmb}(e_{l'}) \right\|_1 \quad (20)$$

We may not want a flat cosine (13), but introduce some and-like semantics for the match between the vectors with four fields $(s_1, s_1'; o_1, o_1')$ and $(s_2, s_2'; o_2, o_2')$. As one example, we might use

$$\sigma\big(\cos((s_1, s_1'), (s_2, s_2'))\big) \times \sigma\big(\cos((o_1, o_1'), (o_2, o_2'))\big), \quad (21)$$

and even shift and scale the cosines before applying sigmoid non-linearities.