Context-Enriched Learning Models for Aligning Biomedical Vocabularies at Scale in the UMLS Metathesaurus

Vinh Nguyen
National Library of Medicine
Bethesda, Maryland, USA
vinh.nguyen@nih.gov

Hong Yung Yip
University of South Carolina
Columbia, South Carolina, USA
hyip@sc.edu

Goonmeet Bajaj
The Ohio State University
Columbus, Ohio, USA
bajaj.32@osu.edu

Thilini Wijesiriwardene
University of South Carolina
Columbia, South Carolina, USA
thilini@sc.edu

Vishesh Javangula
George Washington University
Columbia, Virginia, USA
visheshj123@gwu.edu

Srinivasan Parthasarathy
The Ohio State University
Columbus, Ohio, USA
parthasarathy.2@osu.edu

Amit Sheth
University of South Carolina
Columbia, South Carolina, USA
amit@sc.edu

Olivier Bodenreider
National Library of Medicine
Bethesda, Maryland, USA
olivier@nlm.nih.gov

ABSTRACT

The Unified Medical Language System (UMLS) Metathesaurus construction process mainly relies on lexical algorithms and manual expert curation for integrating over 200 biomedical vocabularies. A lexical-based learning model (LexLM) was developed to predict synonymy among Metathesaurus terms and largely outperforms a rule-based approach (RBA) that approximates the current construction process. However, the LexLM has the potential for being improved further because it only uses lexical information from the source vocabularies, while the RBA also takes advantage of contextual information. We investigate the role of multiple types of contextual information available to the UMLS editors, namely source synonymy (SS), source semantic group (SG), and source hierarchical relations (HR), for the UMLS vocabulary alignment (UVA) problem.

In this paper, we develop multiple variants of context-enriched learning models (ConLMs) by adding to the LexLM the types of contextual information listed above. We represent these context types in context-enriched knowledge graphs (ConKGs) with four variants ConSS, ConSG, ConHR, and ConAll. We train these ConKG embeddings using seven KG embedding techniques. We create the ConLMs by concatenating the ConKG embedding vectors with the word embedding vectors from the LexLM. We evaluate the performance of the ConLMs using the UVA generalization test datasets with hundreds of millions of pairs.

Our extensive experiments show a significant performance improvement from the ConLMs over the LexLM, namely +5.0% in precision (93.75%), +0.69% in recall (93.23%), +2.88% in F1 (93.49%) for the best ConLM. Our experiments also show that the ConAll variant including the three context types takes more time, but does not always perform better than other variants with a single context type. Finally, our experiments show that the pairs of terms with high lexical similarity benefit most from adding contextual information, namely +6.56% in precision (94.97%), +2.13% in recall (93.23%), +4.35% in F1 (94.09%) for the best ConLM. The pairs with lower degrees of lexical similarity also show performance improvement with +0.85% in F1 (96%) for low similarity and +1.31% in F1 (96.34%) for no similarity. These results demonstrate the importance of using contextual information in the UVA problem.

CCS CONCEPTS

• **Computer systems organization → Neural networks**; • **Computing methodologies → Neural networks**; • **Applied computing → Bioinformatics**.

KEYWORDS

UMLS Metathesaurus, neural networks, vocabulary alignment, scalability, supervised learning, knowledge graph embeddings.

ACM Reference Format:
Vinh Nguyen, Hong Yung Yip, Goonmeet Bajaj, Thilini Wijesiriwardene, Vishesh Javangula, Srinivasan Parthasarathy, Amit Sheth, and Olivier Bodenreider. 2022. Context-Enriched Learning Models for Aligning Biomedical Vocabularies at Scale in the UMLS Metathesaurus. In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3485447.3511946

1 INTRODUCTION

The Unified Medical Language System (UMLS) Metathesaurus is a biomedical terminology integration system developed by the US National Library of Medicine to integrate biomedical terms by organizing clusters of synonymous terms into concepts. The current construction process of the UMLS Metathesaurus heavily relies on lexical similarity algorithms to identify candidates for synonymy, although terms that do not share a common semantics are prevented from being recognized as synonymous. Additionally,

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France

ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3511946

synonymy asserted between terms in a source vocabulary is generally preserved in the Metathesaurus. Final synonymy determination comes from human curation by the Metathesaurus editors. Given the current size of the Metathesaurus with over 15 million terms from 214 source vocabularies, this process is inevitably costly and error-prone (as pointed out by [8, 9, 19, 28, 29].)

Motivation. Nguyen et al. [34] formalized synonymy prediction in the UMLS Metathesaurus as a vocabulary alignment problem, referred to as the UMLS Vocabulary Aligment (UVA) problem. The scale of this problem is extremely large (even when restricted to English terms from active vocabularies), since $8.7M$ biomedical terms need to be compared pairwise. Additionally, they developed a rule-based baseline ("RBA baseline") that approximates the Metathesaurus construction process described above. Moreover, to evaluate their vocabulary alignment algorithms, they created different UVA dataset variants (also referred to as "seed alignment" or "ground truth") with different degrees of lexical similarity among the negative examples. The UVA datasets contain hundreds of millions of pairs of biomedical terms (as described in Section 5.2), i.e., several orders of magnitude more than the datasets typically used for evaluating ontology alignment algorithms (e.g., tens of thousands of pairs in the OAEI datasets).

In practice, the full scale of the UVA problem ($8.7M^2$) is impractical for any kind of experimentation. For this reason, all the experiments use the UVA datasets for evaluating the models before applying the best models to the full scale of the UVA problem.

To address the UVA problem, Nguyen et al. [33, 34] developed a scalable supervised learning approach with lexical learning models (LexLMs) that largely outperformed the RBA baseline. However, they noted as a limitation of their work that they only leveraged lexical information and did not include any contextual information. For example, the terms "COLD" from NCI and "Cold" from SNOMEDCT_US have the same lexical embeddings and will therefore be predicted as synonymous by the LexLM. However, these two terms can be disambiguated by adding to the LexLM their source synonyms, such as "Chronic Obstructive Lung Disease" and "Common cold", respectively. Therefore, we believe that the LexLM can be further improved, because it only uses lexical information from biomedical vocabularies, while the RBA also takes advantage of contextual information.

Note that we choose to add contextual information to the LexLM, because its Siamese architecture and BioWordVec embeddings have been shown to outperform the BERT-based approaches for the UVA problem [3]. In [3], the authors implemented and scaled up different approaches for extracting word and sentence embeddings from the BERT-based models such as BioBERT [21], UmlsBERT [26], SapBERT [24], BlueBERT [37]. Their experiments showed that the LexLM with a Siamese Network architecture and BioWordVec embeddings performed best. Additionally, other existing approaches, e.g., from the OAEI [12, 18, 20, 31, 41], were developed and tested with OAEI datasets that are small compared to the UVA problem. Therefore, these approaches are not suitable for the scale of the UVA problem. However, scaling up and adapting existing approaches is not trivial and beyond the scope of this paper.

Objectives. Our first objective is to improve the performance of the LexLM by adding the different types of contextual information available to the UMLS editors, namely source synonymy (SS), source semantic group (SG), and hierarchical relations (HR).

Our second objective is to investigate if these types of contextual information should be added individually or collectively into the LexLM. We assess each approach in terms of computational costs and experimental performance given the size of the UVA datasets.

Our third objective is to evaluate the performance of the context-enriched models on the UVA datasets containing different degrees of lexical similarity in the biomedical term pairs. This will allow us to evaluate the feasibility of applying the proposed approach at the full scale of the UVA problem [34].

Approach. To evaluate the types of contextual information individually and collectively, we represent each context type in a context-enriched knowledge graph (ConKG) variant: ConSS, ConSG, and ConHR for each individual context type, and ConAll for all the context types. We use various KG embedding techniques to train these ConKG embeddings and then concatenate the ConKG embeddings with the lexical embeddings from LexLM to create the ConLM models. We evaluate the performance of these ConLM models on the UVA datasets for each KGE technique with different lexical similarity levels in their biomedical term pairs.

Note that we use various existing KGE techniques in the experiments for the purpose of exploring how these KGE techniques perform with common hyper-parameters when adding multiple context types. Given the cost of running each experiment (5-6 days for each), we did not attempt to identify optimal hyper-parameters for each KGE technique. Therefore, our approach would not support a fair comparison of these KGE techniques and we make no claim that our work represents any kind of systematic evaluation or performance benchmark for the KG embedding techniques.

Contributions. For each objective, we obtain the experimental results as follows.

Our contribution for the first objective (improving the performance of the LexLM) is an approach to develop context-enriched learning models (ConLMs) with significant overall performance improvement over the reference LexLM, namely +5.0% in precision (93.75%), +0.69% in recall (93.23%), +2.88% in F1 (93.49%) for the best ConLM. The experiments also show performance gains from +1.52% to +2.88% in F1 from the ConLMs with all the seven KGE techniques.

Our contribution for the second objective (adding types of contextual information individually or collectively) is an extensive set of experiments for evaluating the ConLMs with multiple ConKG variants showing that, although the ConAll variant including the three context types takes the longest time for training, it does not always outperform single context type variants (lower performance with 4 out of 7 KGE techniques). However, no single context type performs better than ConAll across all KGE techniques.

Our contribution for the third objective (assessing the impact of adding contextual information on specific datasets) is an extensive set of experiments showing that the pairs with a high degree of lexical similarity among negative examples benefit most from adding contextual information, with +6.56% in precision (94.97%), +2.13% in recall (93.23%), +4.35% in F1 (94.09%) for the best ConLM. The pairs with lower degrees of lexical similarity also show performance improvement with +0.85% in F1 (96%) for low similarity and +1.31% in F1 (96.34%) for no similarity.

The remainder of the paper is organized as follows. Section 2 provides background knowledge about the Metathesaurus and how lexical and contextual information is transformed into ConKGs. Section 3 describes the KG embedding techniques selected for training the embedding vectors for the ConKGs. Section 4 describes how the ConLMs are developed. In Section 5, we present our experiments and their results. In Section 6, we discuss our findings and future work. In section 7, we discuss the related work. Section 8 concludes the paper.

2 CONTEXT-ENRICHED KNOWLEDGE GRAPHS

This section describes how multiple variants of context-enriched knowledge graphs (ConKG) are constructed using the various types of contextual information from source vocabularies. These ConKG variants will be used as input for training the KG embeddings in Section 3.

2.1 Background Knowledge on the UMLS Metathesaurus

Nguyen et al. [34] described the knowledge representation aspects of UMLS used in the synonymy prediction task. Here we briefly summarize key concepts from [34] and add new concepts specific to contextual information. The examples below are illustrated in Figure 1.

AUI. Every occurrence of a term in a source vocabulary is assigned a unique atom identifier (AUI). For example, "Cold" in SNOMEDCT_US and "COLD" in NCI are assigned different AUIs, "A2880095" and "A17684490", respectively. Let A be the set of AUIs.

SCUI and m_s. Each AUI is optionally associated with one identifier provided by its source ("Source CUI" or SCUI). Terms considered synonymous in a source vocabulary are assigned the same SCUI. For example, the terms "COLD" and "Chronic Obstructive Lung Disease" are associated with the same SCUI, "C3199", from the source vocabulary NCI. SCUIs play an important role in the Metathesaurus construction process because source synonymy is very often conserved in the Metathesaurus.

(M1) Let S be the set of SCUIs in the Metathesaurus. Let m_s be the function that maps an atom $a \in A$ to an SCUI $s \in S$ such that $s = m_s(a)$.

Source Semantic Group and m_g. Source semantic groups (SGs) are assigned to a source vocabulary (or to its top-level terms for multi-domain vocabularies). An SCUI from a source will inherit its semantic groups from its source. For example, "COLD" with SCUI "C3199" inherits the SG "Disorders" from the top-level term "Disease, Disorder or Finding" (from NCI). Let G be the set of semantic groups in the Metathesaurus.

(M2) Let m_g be the function that maps an SCUI $s \in S$ to a set of semantic groups such that $m_g(s) \subset G$.

Source Hierarchical Relations and m_h. An SCUI may have parent or child terms in a source vocabulary. For example, "COLD" with SCUI "C3199" (from NCI) has "Chronic Lung Disorder" with SCUI "C98541" as a parent and "Pulmonary Emphysema" with SCUI "C3348" as a child.

(M3) Let m_h be the function that maps an SCUI $s \in S$ to a set of its parents, $m_h : S \rightarrow S$ such that $m_h(s) \subset S$.

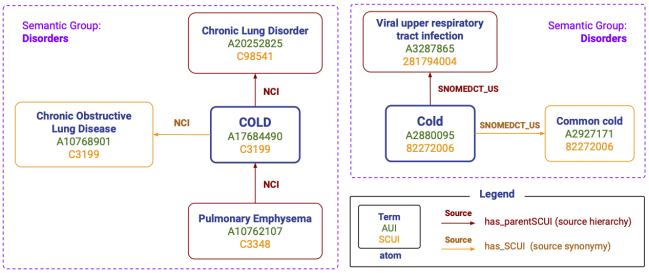

Figure 1: Example illustrating the contextual information available for disambiguating the terms *COLD* from NCI and *Cold* from SNOMEDCT_US, including source synonyms (through has_SCUI), source semantic group (*Disorders*), and source hierarchical relations (through has_parentSCUI). Note that SNOMEDCT_US and NCI are two examples out of over 200 vocabularies from the UMLS Metathesaurus that are included in the UVA datasets.

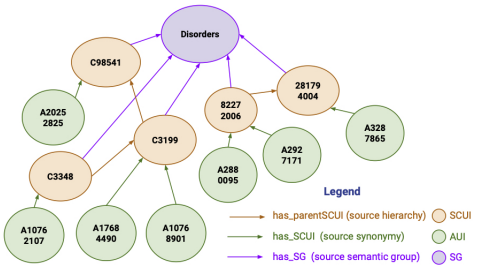

Figure 2: An example of the ConAll variant in the knowledge graph representation.

2.2 Context-enriched Knowledge Graphs

Here we explain how we construct the context-enriched knowledge graphs (ConKGs) (Figure 2) and define the set of triples constructed for each ConKG variant.

A is the set of AUIs, S is the set of SCUIs, G is the set of SGs, and the mapping functions $\{m_s, m_g, m_h\}$ are defined in M1, M2, and M3 above.

ConSS. Let r_s denote the binary relation has_SCUI from an AUI $a \in A$ to an SCUI $s \in S$. The ConSS variant includes the triples representing the relationship between an AUI and its SCUI.

(V1) ConSS = $\{(a, r_s, s) : s = m_s(a)\}$.

ConSG. Let r_g denote the binary relation has_SG from an SCUI $s \in S$ to a SG $g \in G$. The ConSG variant includes the triples representing the relationship between an SCUI and its semantic groups.

(V2) ConSG = $\{(s, r_g, g) : g \in m_g(s)\}$.

ConHR. Let r_h denote the binary relation has_parentSCUI from an SCUI $s \in S$ to its parent SCUI $p \in S$. The ConHR variant includes the triples representing the relationship between an SCUI and its parent SCUI.

(V3) ConHR = $\{(s, r_h, p) : p \in m_h(s)\}$.

ConAll. ConAll = ConSS ∪ ConSG ∪ ConHR. An example of ConAll is shown in Figure 2.

Table 1: Set of ConKG embeddings for each ConKG variant

Variant	ConKG Triples	Set of ConKG embedding vectors
ConSS	$\{(a, r_s, s) : s = m_s(a)\}$	$\{\mathbf{E}_{ConSS}(a) \oplus \mathbf{E}_{ConSS}(m_s(a)) : \forall a \in A\}$
ConSG	$\{(s, r_g, g) : g \in m_g(s)\}$	$\{\mathbf{E}_{ConSG}(m_s(a)) \oplus \sum_{j=1}^{\|m_g(m_s(a))\|} \mathbf{E}_{ConSG}(g_j) : \forall a \in A, g_j \in m_g(m_s(a))\}$
ConHR	$\{(s, r_h, p) : p \in m_h(s)\}$	$\{\mathbf{E}_{ConHR}(m_s(a)) : \forall a \in A\}$
ConAll	$\{(a, r_s, s) : s = m_s(a)\}$ $\{(s, r_g, g) : g \in m_g(s)\}$ $\{(s, r_h, p) : p \in m_h(s)\}$	$\{\mathbf{E}_{ConAll}(a) \oplus$ $\mathbf{E}_{ConAll}(m_s(a)) \oplus$ $\sum_{j=1}^{\|m_g(m_s(a))\|} \mathbf{E}_{ConAll}(g_j) : \forall a \in A, g_j \in m_g(m_s(a))\}$

Table 2: List of abbreviations used in the paper

Notion	Meaning	Notion	Meaning
AUI	Atom unique ID	A	set of *AUIs*
SCUI	Source concept unique identifier	T	set of ConKG triples
SS	Source synonym	T'	set of ConKG negative triples
SG	Semantic group	S	set of SCUIs
HR	Hierarchical Relation	m_s	$A \rightarrow S$
TOPN_SIM	Highest level of lexical similarity	G	set of *SGs*
RAN_NOSIM	Zero lexical similarity	m_g	$S \rightarrow G$
RAN_SIM	Low lexical similarity	m_h	$S \rightarrow S$
LexLM	Lexical-based learning model	E	$A \cup S \cup G$
ConLM	Context-enriched learning model	$\mathbf{E}$	set of entity embeddings
ConKG	Context-enriched knowledge graph	$\mathbf{C}$	set of ConKG embeddings
TRAIN_	Prefix of training datasets	$\sum$	average an array of vectors
GEN_	Prefix for generalization test sets	$\oplus$	concatenate vectors

3 EMBEDDINGS FOR CONTEXT-ENRICHED KNOWLEDGE GRAPHS

This section describes how the ConKG triples are transformed into their respective ConKG embedding vectors using various KG embedding techniques. We selected a few candidate algorithms from each KG algorithm class, (1) Translational distance-based: TransE [5], TransR [23]; (2) Semantic matching-based: RESCAL [36], DistMult [46], HolE [35], and ComplEx [40]; and (3) Neural network-based: ConvKB [32] due to their popularity and demonstrated all-around performance on multi-relational graphs. These trained ConKG embedding vectors will be then added to the LexLM to form the ConLMs in Section 4. A list of abbreviations is provided in Table 2 for convenience.

3.1 Knowledge Graph Embeddings

We explore different KG embedding approaches to transform the structural representation of ConKG triples T into a low-dimensional vector space, while preserving the semantics defined in the ConKG. Such transformation allows the ConKG triples to be added to the LexLM as a set of ConKG embedding vectors. Here we describe how ConKG triples are transformed into ConKG embedding vectors.

Given that A is the set of AUIs, S is the set of SCUIs, and G is the set of SGs, let E be the set of ConKG entities, $E = A \cup S \cup G$. Let R be the set of all ConKG relations, $R = \{r_s, r_g, r_h\}$. Let T be the set of ConKG triples, a triple $t \in T$ if $t = (e_1, r, e_2)$ with $r \in R$, and $e_1, e_2 \in E$. Let T' be the set of negative ConKG triples, $t' = (e'_1, r, e'_2) \in T'$ if $\exists t = (e_1, r, e_2) \in T$ and $t' \notin T$. Let $d = 2 * i$ ($i \in \mathbf{N}$) be the dimension of embedding vector. We choose d to be an even number to facilitate the representation of ComplEx vectors as explained later in this section.

Generating embedding vectors. KG embedding techniques generate the embeddings for entity and relation vectors of dimension d using a scoring function $f_r : E \times R \times E \rightarrow \mathbb{R}$. This scoring function measures the plausibility of facts by minimizing the loss function $L(T, T', \theta)$ with respect to parameter θ either by (a) distance-based (TransE, TransR), (b) similarity-based (RESCAL, HolE, DistMult, and ComplEx) or (c) neural network-based (ConvKB) scoring functions. (See [17] for details about scoring and loss functions for the various embedding techniques.)

Entity embeddings. Let $\mathbf{E}$ be the set of embedding vectors of E, then $\mathbf{e} \in \mathbf{E}$ is an embedding vector of entity e. Here we only use entity embeddings because our ConKG has only three relations and these relation embeddings are not particularly useful for our task. While the embedding for an entity or relation from TransE, HolE, and DistMult is a single vector with dimension $d = 2i$, ComplEx embeddings require two vectors (real and imaginary) of dimension $d = i$. In this case, we concatenate the real and imaginary vectors for each entity into a single vector of dimension $d = 2i$. For ComplEx, we denote the two-vector embeddings as $\mathbf{E} = (\mathbf{E}_{re}, \mathbf{E}_{im})$, then we define the embedding vector for an entity e as $\mathbf{E}(e) = \mathbf{E}_{re}(e) \oplus \mathbf{E}_{im}(e), \forall e \in \mathbf{E}$.

The output of each KG embedding technique is a set of entity embeddings: $\mathbf{E}_{ConSS}$, $\mathbf{E}_{ConSG}$, $\mathbf{E}_{ConHR}$, and $\mathbf{E}_{ConAll}$, which will be used to derive the ConKG embeddings in the next section.

3.2 ConKG Embeddings

We derive ConKG embeddings $\mathbf{C}$ for each type of contextual information described in Section 2.2 with respect to $a \in A$, so that we can add them to the word embedding vectors from the LexLM described in Section 4. For each type of contextual information, we generate a ConKG embedding vector $\mathbf{c} \in \mathbf{C}$ for each $a \in A$ by concatenating the entity embeddings of ConKG entities inside the ConKG triples corresponding to a, including $m_s(a)$ for an SCUI, and $m_g(m_s(a))$ for a semantic group. As an SCUI is mapped to a set of semantic groups, we get the entity embedding for each semantic group and average the set of embedding vectors ($\sum$). We reuse the mapping functions defined in Section 2.2. Table 1 shows the set of the ConKG embeddings for each type of contextual information. The output of each KG embedding technique is a set of context-enriched embeddings for all AUIs: $\mathbf{C}_{ConSS}$, $\mathbf{C}_{ConSG}$, $\mathbf{C}_{ConHR}$, and $\mathbf{C}_{ConAll}$, which will be added to the reference LexLM in Section 4 and evaluated in Section 5.

4 NEURAL NETWORK ARCHITECTURE

This section describes the LexLM architecture from [34], which we will be using as the reference model, and our approach for adding the context-enriched embeddings from Section 3 to the LexLM.

LexLM. The LexLM (grey boxes in Figure 3) adopts the Siamese architecture [30] that takes in a pair of inputs and learns representations based on explicit similarity and dissimilarity information defined during training. The inputs (a pair of atoms) are pre-processed and transformed into their numerical representations with BioWordVec embeddings pre-trained from PubMed text corpus and MeSH data [47]. The word embeddings are then fed to Long Short Term Memory (LSTM) layers to learn the semantic and syntactic features of the atoms through time. The LexLM relies exclusively on the lexical features of the atoms, i.e., the terms themselves.

ConLMs. We develop the ConLM (Figure 3), which adds to the LexLM (at the LSTM layer) a specific variant of the ConKGs described in Section 3. For each ConKG variant, we first feed the

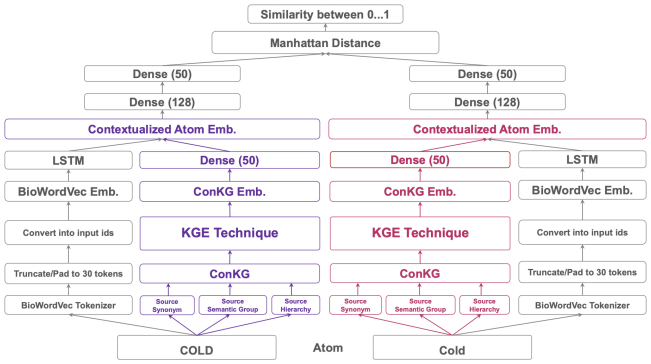

Figure 3: The architecture of the neural network in the Context-enriched Learning Model (ConLM) created by adding the ConKG embeddings to the Lexical-based Learning Model (LexLM) embeddings (grey boxes) from [34]. The seven KGE technique used for ConKG include TransE, TransR, DistMult, HolE, ComplEx, RESCAL, and ConvKB.

Table 3: The ConKG variants with their respective number of unique entities E, relations R, and positive triples

ConKG	$\|E\|$	$\|R\|$	$\|T\|$
ConAll	10,716,301	3	14,774,566
ConSS	10,553,767	1	7,062,582
ConHR	2,816,992	1	3,520,969
ConSG	3,653,711	1	4,191,015

respective trained ConKG embedding vectors to a 50-unit dense layer to learn the derived features in Table 1. We then concatenate ($\oplus$) the output of the dense layer with the output of the LSTM units from the LexLM. Together, they form the contextualized atom embedding, which is then fed to subsequent dense layers with 128 and 50 learning units, respectively. The output is a Manhattan distance similarity function [1], which computes a score indicating the degree of synonymy between the atoms with a threshold of 0.5. The datasets used for training and testing the ConLMs are described in Section 5.2. The trained models are evaluated to assess their respective contribution in Section 5.

5 EVALUATION

This section presents our implementation of the ConLM variants and a set of experiments to evaluate the ConLM variants towards the three objectives as described in Section 1. We describe the set of experiments in Section 5.1, and present the datasets used for training and testing in Section 5.2. We report and discuss the results from the experiments towards the objectives in Section 5.4. We analyze the significance of the results in the Section 5.5. The datasets from [34] are available at https://bit.ly/uva-datasets.

5.1 Experimental Setup

We have presented the ConLM variants with 4 ConKG variants for representing the context types in Section 2.2. Figure 3 presented the architecture of the neural network of the ConLM where KGE techniques are employed to generate the ConKG embeddings. We select 7 KGE techniques to be evaluated in this paper, including TransE, TransR, HolE, ComplEx, DisMult, RESCAL, and ConvKB as described in Section 3. Therefore, we have 28 variants of the ConLM to be implemented and evaluated, since each KGE technique is paired with each of the four context types.

For our implementation of the ConLM, there are multiple steps in our pipeline: (1) we extract the context types from the UMLS Metathesaurus and generate the ConKG datasets representing the ConKG variants, (2) we use the implementation of KGE techniques from the two libraries, OpenKE [13] and PyKEEN [2], for training the ConKG embeddings from the ConKG datasets, (3) we implement the training and testing of 28 variants of the ConLM using Keras and Tensorflow.

All these experiments are deployed as batches of parallel jobs with the Slurm [1] workload manager to the Biowulf high-performance computing cluster [2] at the National Institutes of Health (NIH). We used Tesla V100x GPUs with 32GB of GPU RAM and at least 220GB of CPU RAM for each training and testing task. While the implementation is configurable and reproducible in a different environment, these experiments are computationally- and resource-intensive. We estimated that we used 2753 GPU hours for the set of experiments reported in this paper. Run-time information is provided in Table 5.

Table 4: Statistics of the UVA datasets for both training and generalization tests in terms of number of positive/negative pairs of biomedical terms

Training DS	Negative	Positive	Total
TRAIN_ALL (train)	101,322,647	16,743,627	118,066,274
Generalization Test DS	**Negative**	**Positive**	**Total**
GEN_ALL	166,410,710	5,581,208	171,991,918
GEN_TOPN_SIM	54,752,228	5,581,208	60,333,436
GEN_RAN_SIM	54,445,899	5,581,208	60,027,107
GEN_RAN_NOSIM	58,256,526	5,581,208	63,837,734

5.2 Datasets

This section describes the datasets for the three types of experiments. We used release 2020AA of the UMLS Metathesaurus restricted to English terms from active source vocabularies. The UMLS can be downloaded with a no-cost license [3].

5.2.1 Datasets for Training Embeddings for ConKG Variants. Table 3 shows the characteristics of the four datasets generated for training the KG embeddings for each ConKG variant. We use a positive to negative triple ratio $\in \{1, 50, 200\}$ for generating the triple instances described in Section 2.2. The negative triples are automatically generated using the "bern" sampling technique [43] to corrupt either the e_1 or e_2 entity.

5.2.2 Datasets for Training and Testing ConLMs. To compare the ConLMs against the LexLM baseline, we reuse the training and testing (generalization) datasets from Table 4 generated in [34]. The different datasets splits are based on the degree of lexical similarity among negative examples. The prefix "TRAIN_" refers to

[1] https://slurm.schedmd.com/documentation.html
[2] https://hpc.nih.gov/
[3] https://uts.nlm.nih.gov/uts/

Table 5: Training time (approximate hours per 100 epochs) for each combination of ConKG variant (ConSS for source synonymy, ConSG for semantic group, ConHR for hierarchical relations, and ConAll for all of them) and KG embedding technique (TransE, TransR, DistMult, HolE, ComplEx, RESCAL, and ConvKB). The total training time for all is 2753 hours

	TransE			TransR			DistMult			HolE			ComplEx			RESCAL			ConvKB		
	KGE	ConLM	Total	KGE	ConLM	Total	KGE	ConLM	Total	KGE	ConLM	Total	KGE	ConLM	Total	KGE	ConLM	Total	KGE	ConLM	Total
ConAll	30	74	104	60	62	122	41	85	126	48	77	125	43	79	122	60	78	138	68	87	155
ConSS	13	75	88	27	78	105	20	84	104	21	69	90	18	85	103	27	68	95	45	75	120
ConHR	4	102	106	9	65	74	5	80	85	6	66	72	5	81	86	10	62	72	17	70	87
ConSG	6	66	72	11	76	87	7	74	81	7	69	76	7	81	88	12	65	77	22	71	93

Table 6: Results for ComplEx embedding technique towards 3 objectives: (O1) the highest overall performance gain with +2.88% in F1 with GEN_ALL, (O2) ConAll variant outperformed the other 3 variants for all metrics, and (O3) the highest performance gains +4.35%, +1.31%, +0.85% in F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9958**	**0.9375**	**0.9323**	**0.9349**	**0.9892**	**0.9497**	**0.9323**	**0.9409**
	Diff	0.0020	0.0500	0.0069	0.0288	0.0085	0.0656	0.0213	0.0435
ConSS	Score	0.9939	0.9008	0.9139	0.9073	0.9858	0.9318	0.9139	0.9227
	Diff	0.0001	0.0133	-0.0115	0.0012	0.0051	0.0477	0.0029	0.0253
ConHR	Score	0.9947	0.9126	0.9237	0.9181	0.9868	0.9333	0.9237	0.9284
	Diff	0.0009	0.0251	-0.0017	0.0120	0.0061	0.0492	0.0127	0.0310
ConSG	Score	0.9946	0.9070	0.9283	0.9175	0.9869	0.9299	0.9283	0.9291
	Diff	0.0008	0.0195	0.0029	0.0114	0.0062	0.0458	0.0173	0.0317
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9928**	**0.9894**	**0.9323**	**0.9600**	**0.9938**	**0.9967**	**0.9323**	**0.9634**
	Diff	0.0023	0.0036	0.0213	0.0131	0.0014	-0.0004	0.0161	0.0085
ConSS	Score	0.9901	0.9779	0.9139	0.9448	0.9913	0.9852	0.9139	0.9482
	Diff	-0.0004	-0.0079	0.0029	-0.0021	-0.0011	-0.0119	-0.0023	-0.0067
ConHR	Score	0.9915	0.9839	0.9237	0.9528	0.9926	0.9915	0.9237	0.9564
	Diff	0.0010	-0.0019	0.0127	0.0059	0.0002	-0.0056	0.0075	0.0015
ConSG	Score	0.9915	0.9792	0.9283	0.9531	0.9932	0.9936	0.9283	0.9598
	Diff	0.0010	-0.0066	0.0173	0.0062	0.0008	-0.0035	0.0121	0.0049
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549

Table 7: Results for TransE embedding technique towards 3 objectives: (O1) the best performance gain with +1.99% in F1 with GEN_ALL, (O2) ConAll variant outperformed the other 3 variants for all metrics except precision, and (O3) the highest performance gains +3.72%, +1.05%, and +0.69% in F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9952**	0.9210	**0.9311**	**0.9260**	**0.9879**	0.9382	**0.9311**	**0.9346**
	Diff	0.0014	0.0335	0.0057	0.0199	0.0072	0.0541	0.0201	0.0372
ConSS	Score	0.9944	0.9117	0.9171	0.9144	0.9866	0.9371	0.9171	0.9270
	Diff	0.0006	0.0242	-0.0083	0.0083	0.0059	0.0530	0.0061	0.0296
ConHR	Score	0.9951	**0.9239**	0.9263	0.9251	0.9876	**0.9391**	0.9263	0.9327
	Diff	0.0013	0.0364	0.0009	0.0190	0.0069	0.0550	0.0153	0.0353
ConSG	Score	0.9947	0.9079	0.9302	0.9189	0.9874	0.9329	0.9302	0.9316
	Diff	0.0009	0.0204	0.0048	0.0128	0.0067	0.0488	0.0192	0.0342
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9923**	0.9853	**0.9311**	**0.9574**	**0.9935**	0.9946	**0.9311**	**0.9618**
	Diff	0.0018	-0.0005	0.0201	0.0105	0.0011	-0.0025	0.0149	0.0069
ConSS	Score	0.9907	0.9817	0.9171	0.9483	0.9918	0.9884	0.9171	0.9514
	Diff	0.0002	-0.0041	0.0061	0.0014	-0.0006	-0.0087	0.0009	-0.0035
ConHR	Score	0.9920	**0.9873**	0.9263	0.9558	0.9931	**0.9948**	0.9263	0.9593
	Diff	0.0015	0.0015	0.0153	0.0089	0.0007	-0.0023	0.0101	0.0044
ConSG	Score	0.9916	0.9778	0.9302	0.9534	0.9933	0.9926	0.9302	0.9604
	Diff	0.0011	-0.0080	0.0192	0.0065	0.0009	-0.0045	0.0140	0.0055
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549

training and "GEN_" refers to the generalization dataset. The suffix "_ALL" dataset contains the following splits: (a) TOPN_SIM - negative pairs with highest level of lexical similarity, (b) RAN_SIM - random negative pairs with low level of lexical similarity, and (c) RAN_NOSIM - random negative pairs with no lexical similarity. Training and generalization datasets are mutually exclusive. The LexLM variant trained on the TRAIN_ALL dataset performs best across the four generalization tests (GEN_ALL, GEN_TOPN_SIM, GEN_RAN_SIM, and GEN_RAN_NOSIM) and is used as baseline here. In practice, we use the dataset TRAIN_ALL for training 28 variants of our ConLMs, and the four generalization datasets for testing our ConLMs (Section 5.3.2).

5.3 Training

5.3.1 Training 28 Variants of ConKG Embeddings. We use both OpenKE [13] and PyKEEN [2] libraries to implement the KG embedding techniques to train the embeddings independently for each ConKG variant described in Section 2.2. Since this is not a systematic evaluation, nor a performance benchmark across various KG embedding techniques, we did not attempt to select optimal hyper-parameters for each technique. Instead we ran various hyper-parameter selection experiments and obtained a list of hyper-parameters that balance performance and training speed, as well as maximize the GPU memory across all techniques.

Training parameters. Each ConKG variant is trained with each KG embedding technique with (a) 100-1000 epochs, (b) batch size $\in \{50, 256, 1024, 2048\}$ depending on the complexity of the KG technique and available GPU memory, (c) learning rate $\in \{0.01, 0.05\}$, (d) loss margin of 1.0, (e) positive to negative triple sampling ratio $\in \{1, 50, 200\}$, and with (f) embedding dimension of size $\in \{50, 100\}$, (g) optimizer $\in \{SGD, Adam\}$. Table 5 shows the training time for each combination of ConKG variant and KG embedding technique.

5.3.2 Training and Testing 28 ConLM Variants. We trained 28 ConLM variants (by combining each of the four context types with each of the seven KGE techniques) using the 28 variants of the ConKG embeddings obtained from the training presented in Section 5.3.1.

Table 8: Results for DisMult embedding technique towards 3 objectives: (O1) the best performance gain with +1.62% in F1 with GEN_ALL, (O2) ConAll variant outperformed other 3 variants in all metrics except precision, and (O3) the highest performance gains +3.62%, +0.87%, +0.67% F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9949**	0.9129	**0.9319**	**0.9223**	**0.9877**	0.9353	**0.9319**	**0.9336**
	Diff	0.0011	0.0254	0.0065	0.0162	0.0070	0.0512	0.0209	0.0362
ConSS	Score	0.9948	**0.9177**	0.9218	0.9197	0.9872	**0.9389**	0.9218	0.9302
	Diff	0.0010	0.0302	-0.0036	0.0136	0.0065	0.0548	0.0108	0.0328
ConHR	Score	0.9945	0.9104	0.9225	0.9164	0.9866	0.9324	0.9225	0.9274
	Diff	0.0007	0.0229	-0.0029	0.0103	0.0059	0.0483	0.0115	0.0300
ConSG	Score	0.9947	0.9084	0.9300	0.9191	0.9874	0.9331	0.9300	0.9316
	Diff	0.0009	0.0209	0.0046	0.0130	0.0067	0.0490	0.0190	0.0342
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9919**	0.9805	**0.9319**	**0.9556**	**0.9935**	**0.9932**	**0.9319**	**0.9616**
	Diff	0.0014	-0.0053	0.0209	0.0087	0.0011	-0.0039	0.0157	0.0067
ConSS	Score	0.9913	**0.9841**	0.9218	0.9519	0.9924	0.9912	0.9218	0.9552
	Diff	0.0008	-0.0017	0.0108	0.0050	0.0000	-0.0059	0.0056	0.0003
ConHR	Score	0.9914	0.9836	0.9225	0.9521	0.9924	0.9901	0.9225	0.9551
	Diff	0.0009	-0.0022	0.0115	0.0052	0.0000	-0.0070	0.0063	0.0002
ConSG	Score	0.9916	0.9784	0.9300	0.9536	0.9933	0.9924	0.9300	0.9602
	Diff	0.0011	-0.0074	0.0190	0.0067	0.0009	-0.0047	0.0138	0.0053
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549

We use the TRAIN_ALL dataset variant with 118M pairs as shown in Table 4 for training our ConLMs.

Training parameters. We also use the same training hyperparameters from the LexLM, namely 100 epochs, batch size of 8192, and Adam optimizer. Training each epoch of the ConLMs takes about 30 minutes to one hour as shown in Table 5.

Evaluation metrics. To evaluate our synonymy prediction models, we employ the usual metrics for machine learning systems, namely accuracy, precision, and recall. Since precision and recall are equally important to our application, we use the F1 score, the harmonic mean of the precision and recall, as a balanced metric to represent them both.

5.4 Results

We test these 28 trained ConLM variants using four generalization test sets including GEN_ALL, GEN_TOPN_SIM, GEN_RAN_SIM, and GEN_RAN_NOSIM as shown in Table 4. We report the results for these 28 variants of the ConLMs in Table 6-8 and Table 9-12 (in Appendix) with each KGE technique in a separate table. Each table contains four blocks corresponding to the four generalization test datasets with different degrees of lexical similarity. Each block compares the four ConLM variants using the same generalization test dataset. In each table, we summarize the results of each technique towards the three objectives of the paper. Here we summarize the results per objective across all KGE techniques.

5.4.1 Objective 1: Improving the Performance of the LexLM. To evaluate the overall performance of the ConLM over the LexLM, we compare the results from ConLM variants using the GEN_ALL dataset because this dataset contains the pairs from different degrees of lexical similarity (the top-left green block).

The best F1 scores of the ConLMs with different KGE techniques increased between +1.52% and +2.88% compared to the LexLM, and the ConLM with the best F1 score is trained with the ConAll variant (Table 6 with ComplEx). Compared to the LexLM, the best ConLM has gained +5.0% in precision, +0.69% in recall, and +2.88% in F1. We also observed the pattern that most of the ConLM variants gained more in precision and less (or lost) in recall.

5.4.2 Objective 2: Adding Context Types Individually or Collectively. To evaluate the ConKG variants collectively and individually, we compare the performance of the ConKG variants across multiple KGE techniques because one variant can perform well with one technique but not with others.

From Table 6-8, we observed that the ConAll variant outperformed other ConKG individual variants in 3 out of 7 KGE techniques. Particularly, it outperformed others (with ComplEx Table 6) in every metric. It also outperformed others (TransE in Table 7 and and DisMult in Table 8) in every metric except precision (lower than ConHR). On the other hand, Table 5 shows that the training time for ConAll with these KGE techniques is several times higher than other ConKG variants.

Table 9 shows that ConSG is the only individual ConKG variant that outperformed the ConAll in every metric. We also observed that ConAll does not outperform other ConKG individual variants in 3 KGE techniques (TransR, ConvKB, and RESCAL) as shown in Table 10-12. In summary, there is no clear winner for these cases.

5.4.3 Objective 3: Assessing the Impact of Adding Contextual Information on Specific Datasets. For this objective, we evaluate the performance of the ConLMs using UVA datasets with high/low/no lexical similarity, including GEN_TOPN_SIM, GEN_RAN_SIM, and GEN_RAN_NOSIM. Respectively, the highest performance gains in F1 are from +3.28% to +4.35%, +0.58% to +1.31%, and +0.19% to +0.85% for different KGE techniques. In other words, the largest gain in F1 is observed with datasets that contain a high degree of lexical similarity between terms. However, more modest performance gains can be observed consistently across all datasets, including those with no similarity between terms.

5.5 Statistical Analysis

To assess the statistical significance of the difference in overall performance between the best ConLM (ConAll in Table 6) and the reference LexLM on the GEN_ALL dataset, we perform a McNemar test. This test compares the distribution of positive and negative predictions between the two models. The McNemar statistics (60887.0) indicates that the difference is statistically significant ($p < 0.001$). Of note, since our goal is not to assess the superiority of specific KGE techniques or types of contextual information, we did not perform a systematic statistical analysis of all differences in performance.

6 DISCUSSION AND FUTURE WORK

Findings. As we hypothesized, the use of KG embeddings to add contextual information to the reference LexLM yielded significant performance improvement over the LexLM baseline, especially in terms of precision (+5.0% to reach 93.75%). Even more remarkably,

there was no concomitant loss of recall, but marginal gain instead (+0.69% to reach 93.23%) and the overall performance (F1) also increased (+2.88% in F1 to reach 93.49%). We showed a large degree of variability among KG embedding techniques and among types of contextual information. The optimal type of contextual information seems to depend on the KGE technique used. ConAll performs best with three of the seven KGE techniques tested, but at the cost of increased training time. On the other hand, no single context type outperforms ConAll across all KGE techniques. Therefore, selecting the optimal context type for the UVA task at scale will require a trade off between performance and training time. Also important is the fact that only the semantic group is guaranteed to be specified for each atom, while source synonymy and hierarchical relations are only present for about 50% of the atoms on average. This may explain the somewhat strong performance of ConAll. Finally, our experiments also confirmed that the addition of contextual information not only benefits pairs of terms with high lexical similarity as we expected, but also improves performance across all test datasets.

Significance. The performance of the reference LexLM was already very strong, especially given the limited information available to this model (i.e., only the lexical features of terms). However, given the very large number of comparisons required for the Metathesaurus construction, the number of false positives remained important despite precision values near 90%. Therefore, increasing precision by 5.0% (to reach 93.75%) represents a substantial advance for the UVA task, especially because recall was not negatively affected. Moreover, the gain in precision with ConLM results in a drop of almost 50% in the false positive rate compared to LexLM (from 0.391% to 0.207%). In practice, this performance gain will translate in savings in manual curation, as fewer false positive synonyms will need to be corrected. Another important result for the application of ConLMs to the UVA task is that the addition of contextual information to the LexLM improves performance across all datasets, not only on highly similar terms. This shows that the performance of ConLMs will likely generalize to the UMLS Metathesaurus as a whole, where most terms exhibit no similarity with other terms.

Limitation and Future Work. As mentioned earlier, this investigation is no substitute for a systematic performance evaluation of KG embedding techniques and thorough benchmarking. Our focus was rather on the application to the UVA task. Performing a comprehensive analysis of the differences with the LexLM baseline was beyond the scope of this investigation, but is part of our future work. Finally, we plan to improve the performance of the LexLM, e.g., by developing novel embedding techniques that consider the unique characteristics of the UMLS Metathesaurus.

7 RELATED WORK

Biomedical ontology alignment is a long-standing research effort driven by the Ontology Alignment Evaluation Initiative (OAEI) since 2005. With the growth of interest in integrating biomedical ontologies at scale [34], studies have looked into using rule-based and statistical approaches [12, 18, 31], as well as supervised learning approaches for ontologies matching [11] by assessing the similarity [20, 41] and relatedness [25] between words and sentences. Such tasks are also known as Semantic Text Similarity (STS) tasks.

Recent progress has been attributed to the use of a combination of knowledge-based similarity with deep learning techniques, such as word embeddings [27] for input feature representation, and Siamese Network [4, 14, 15, 30] to learn the underlying semantics and structure. Unsupervised approaches, such as transformer-based mechanisms, have also been explored with a great degree of success [10]. Nonetheless, these techniques are largely based on lexical features and require very large text corpora to learn from. In [16], contextualized representations of concepts with ancestral, child, data property neighbor, and object property neighbor contexts are used to discover semantically equivalent concepts. In our approach, we exploit the graph structure of various types of contextual information through the use of KG embeddings.

There are several families of KG embedding techniques [17, 38]. They aim to map entities and relations into low-dimensional vectors while capturing their structural and semantic meanings [42], and have shown to benefit a variety of knowledge-driven tasks [22]. Since this is the first attempt to use the various types of contextual information in the UVA task at scale, we explored three of the popular classes of techniques because of their demonstrated success in various tasks: translational distance-based with TransE [5], and TransR [23]; semantic matching-based using RESCAL [36], DistMult [46], HolE [35], and ComplEx [40]; and neural network-based using ConvKB [32]. We evaluated their performance in the UVA task, but did not benchmark them against other forms of graph representation techniques [17, 45]. Many embedding techniques listed in [38] are not selected due to the specific characteristics of context in our datasets, e.g., having no attributes/literals (but 10 techniques in [38] including AttrE [39] and KDCoE [6] leveraging attributes), having English-only (but MTransE [7] leveraging multilingual), or very large (RDGCN [44] being not scalable).

8 CONCLUSION

In summary, we demonstrated the importance of using contextual information in the UMLS vocabulary alignment task. Particularly, we showed that it was possible to improve on the performance of a learning model based on the lexical features of biomedical terms by taking into account the context of these terms in their source vocabulary (source synonymy, source semantics, hierarchical relations). Adding contextual information to the lexical model through KG embeddings yielded a substantial gain in precision with no negative effect on recall and was particularly beneficial to lexically similar strings, such as homonyms, but also improved performance overall.

ACKNOWLEDGMENTS

This research was supported by the Intramural Research Program of the National Library of Medicine (NLM), National Institutes of Health. This research was also supported by appointments to the NLM Research Participation Program, administered by the Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy and the NLM. GB and SP acknowledge a grant from cooperative AI Institute grant (AI-EDGE), from the National Science Foundation under CNS-2112471. All content represents the opinion of the authors and is not necessarily endorsed by their institutions or sponsors.

REFERENCES

[1] Charu C Aggarwal, Alexander Hinneburg, and Daniel A Keim. 2001. On the surprising behavior of distance metrics in high dimensional space. In *International conference on database theory*. Springer, 420–434.

[2] Mehdi Ali, Max Berrendorf, Charles Tapley Hoyt, Laurent Vermue, Sahand Sharifzadeh, Volker Tresp, and Jens Lehmann. 2021. PyKEEN 1.0: A Python Library for Training and Evaluating Knowledge Graph Embeddings. *Journal of Machine Learning Research* 22, 82 (2021), 1–6. http://jmlr.org/papers/v22/20-825.html

[3] Goonmeet Bajaj, Vinh Nguyen, Thilini Wijesiriwardene, Hong Yung Yip, Vishesh Javangula, Srinivasan Parthasarathy, Amit Sheth, and Olivier Bodenreider. 2021. Evaluating Biomedical BERT Models for Vocabulary Alignment at Scale in the UMLS Metathesaurus. *arXiv preprint arXiv:2109.13348* (2021).

[4] Alexandre Bento, Amal Zouaq, and Michel Gagnon. 2020. Ontology matching using convolutional neural networks. In *Proceedings of The 12th Language Resources and Evaluation Conference*. 5648–5653.

[5] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Neural Information Processing Systems (NIPS)*. 1–9.

[6] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. 2018. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. *arXiv preprint arXiv:1806.06478* (2018).

[7] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. 2016. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. *arXiv preprint arXiv:1611.03954* (2016).

[8] James J Cimino. 1998. Auditing the unified medical language system with semantic methods. *Journal of the American Medical Informatics Association* 5, 1 (1998), 41–51.

[9] James J Cimino, Hua Min, and Yehoshua Perl. 2003. Consistency across the hierarchies of the UMLS Semantic Network and Metathesaurus. *Journal of biomedical informatics* 36, 6 (2003), 450–461.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[11] AnHai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy. 2004. Ontology matching: A machine learning approach. In *Handbook on ontologies*. Springer, 385–403.

[12] Daniel Faria, Catia Pesquita, Emanuel Santos, Matteo Palmonari, Isabel F Cruz, and Francisco M Couto. 2013. The agreementmakerlight ontology matching system. In *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 527–541.

[13] Xu Han, Shulin Cao, Lv Xin, Yankai Lin, Zhiyuan Liu, Maosong Sun, and Juanzi Li. 2018. OpenKE: An Open Toolkit for Knowledge Embedding. In *Proceedings of EMNLP*.

[14] Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 1576–1586.

[15] Degen Huang, Anil Ahmed, Syed Yasser Arafat, Khawaja Iftekhar Rashid, Qasim Abbas, and Fuji Ren. 2020. Sentence-Embedding and Similarity via Hybrid Bidirectional-LSTM and CNN Utilizing Weighted-Pooling Attention. *IEICE TRANSACTIONS on Information and Systems* 103, 10 (2020), 2216–2227.

[16] Vivek Iyer, Arvind Agarwal, and Harshit Kumar. 2020. VeeAlign: a supervised deep learning approach to ontology alignment.. In *OM@ ISWC*. 216–224.

[17] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S Yu. 2020. A survey on knowledge graphs: Representation, acquisition and applications. *arXiv preprint arXiv:2002.00388* (2020).

[18] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. 2011. Logmap: Logic-based and scalable ontology matching. In *International Semantic Web Conference*. 273–288.

[19] Antonio Jimeno-Yepes, Ernesto Jiménez-Ruiz, Rafael Berlanga-Llavori, and Dietrich Rebholz-Schuhmann. 2009. Reuse of terminological resources for efficient ontological engineering in Life Sciences. *BMC bioinformatics* 10, S10 (2009), S4.

[20] Prodromos Kolyvakis, Alexandros Kalousis, Barry Smith, and Dimitris Kiritsis. 2018. Biomedical ontology alignment: an approach based on representation learning. *Journal of biomedical semantics* 9, 1 (2018), 1–20.

[21] Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics* 36, 4 (2020), 1234–1240.

[22] Yankai Lin, Xu Han, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. 2018. Knowledge representation learning: A quantitative review. *arXiv preprint arXiv:1812.10901* (2018).

[23] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 29.

[24] Fangyu Liu, Ehsan Shareghi, Zaiqiao Meng, Marco Basaldella, and Nigel Collier. 2020. Self-alignment pretraining for biomedical entity representations. *arXiv preprint arXiv:2010.11784* (2020).

[25] Yuqing Mao and Kin Wah Fung. 2020. Use of word and graph embedding to measure semantic relatedness between Unified Medical Language System concepts. *Journal of the American Medical Informatics Association* (2020).

[26] George Michalopoulos, Yuanxin Wang, Hussam Kaka, Helen Chen, and Alex Wong. 2020. Umlsbert: Clinical domain knowledge augmentation of contextual embeddings using the unified medical language system metathesaurus. *arXiv preprint arXiv:2010.10391* (2020).

[27] Rada Mihalcea, Courtney Corley, Carlo Strapparava, et al. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Aaai*, Vol. 6. 775–780.

[28] C Paul Morrey, James Geller, Michael Halper, and Yehoshua Perl. 2009. The Neighborhood Auditing Tool: a hybrid interface for auditing the UMLS. *Journal of biomedical informatics* 42, 3 (2009), 468–489.

[29] Fleur Mougin, Olivier Bodenreider, and Anita Burgun. 2009. Analyzing polysemous concepts from a clinical perspective: Application to auditing concept categorization in the UMLS. *Journal of Biomedical Informatics* 42, 3 (2009), 440–451.

[30] Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

[31] DuyHoa Ngo and Zohra Bellahsene. 2016. Overview of YAM++?(not) Yet Another Matcher for ontology alignment task. *Journal of Web Semantics* 41 (2016), 30–49.

[32] Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2017. A novel embedding model for knowledge base completion based on convolutional neural network. *arXiv preprint arXiv:1712.02121* (2017).

[33] Vinh Nguyen and Olivier Bodenreider. 2021. Adding an Attention Layer Improves the Performance of a Neural Network Architecture for Synonymy Prediction in the UMLS Metathesaurus. *Proceedings of the MedInfo 2021* (2021).

[34] Vinh Nguyen, Hong-Yung Yip, and Olivier Bodenreider. 2021. Biomedical Vocabulary Alignment at Scale in the UMLS Metathesaurus. In *The Web Conference (WWW2021)*. ACM.

[35] Maximilian Nickel, Lorenzo Rosasco, and Tomaso Poggio. 2016. Holographic embeddings of knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 30.

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

[37] Yifan Peng, Shankai Yan, and Zhiyong Lu. 2019. Transfer Learning in Biomedical Natural Language Processing: An Evaluation of BERT and ELMo on Ten Benchmarking Datasets. In *Proceedings of the 2019 Workshop on Biomedical Natural Language Processing (BioNLP 2019)*. 58–65.

[38] Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. 2020. A benchmarking study of embedding-based entity alignment for knowledge graphs. *arXiv preprint arXiv:2003.07743* (2020).

[39] Bayu Distiawan Trisedya, Jianzhong Qi, and Rui Zhang. 2019. Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 297–304.

[40] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International Conference on Machine Learning*. PMLR, 2071–2080.

[41] Lucy Lu Wang, Chandra Bhagavatula, Mark Neumann, Kyle Lo, Chris Wilhelm, and Waleed Ammar. 2018. Ontology alignment in the biomedical domain using entity definitions and context. *arXiv preprint arXiv:1806.07976* (2018).

[42] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering* 29, 12 (2017), 2724–2743.

[43] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 28.

[44] Yuting Wu, Xiao Liu, Yansong Feng, Zheng Wang, Rui Yan, and Dongyan Zhao. 2019. Relation-aware entity alignment for heterogeneous knowledge graphs. *arXiv preprint arXiv:1908.08210* (2019).

[45] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. 2020. A comprehensive survey on graph neural networks. *IEEE transactions on neural networks and learning systems* (2020).

[46] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint 1412.6575* (2014).

[47] Yijia Zhang, Qingyu Chen, Zhihao Yang, Hongfei Lin, and Zhiyong Lu. 2019. BioWordVec, improving biomedical word embeddings with subword information and MeSH. *Scientific data* 6, 1 (2019), 1–9.

A APPENDIX

A.1 Reproducibility

The experiments are reproducible with the materials to be made available at https://bit.ly/www2022-supp.

A.2 Additional Tables

Given the limited space in the paper, here we provide the remaining tables for the KGE techniques.

Table 9: Results for HolE embedding technique towards 3 objectives: (O1) the best performance gain with +1.92% in F1 with GEN_ALL, (O2) ConAll variant not outperforming the other 3 variants for all metrics, and (O3) the highest performance gains +3.74%, +1.04%, and +0.76% in F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9948	0.9111	0.9297	0.9203	0.9874	0.9339	0.9297	0.9318
	Diff	0.0010	0.0236	0.0043	0.0142	0.0067	0.0498	0.0187	0.0344
ConSS	Score	0.9936	0.8919	0.9148	0.9032	0.9855	0.9274	0.9148	0.9210
	Diff	-0.0002	0.0044	-0.0106	-0.0029	0.0048	0.0433	0.0038	0.0236
ConHR	Score	0.9945	0.9072	0.9237	0.9154	0.9865	0.9304	0.9237	0.9270
	Diff	0.0007	0.0197	-0.0017	0.0093	0.0058	0.0463	0.0127	0.0296
ConSG	Score	**0.9951**	**0.9183**	**0.9325**	**0.9253**	**0.9880**	**0.9370**	**0.9325**	**0.9348**
	Diff	0.0013	0.0308	0.0071	0.0192	0.0073	0.0529	0.0215	0.0374
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9917	0.9801	0.9297	0.9542	0.9933	0.9930	0.9297	0.9603
	Diff	0.0012	-0.0057	0.0187	0.0073	0.0009	-0.0041	0.0135	0.0054
ConSS	Score	0.9899	0.9749	0.9148	0.9439	0.9911	0.9823	0.9148	0.9473
	Diff	-0.0006	-0.0109	0.0038	-0.0030	-0.0013	-0.0148	-0.0014	-0.0076
ConHR	Score	0.9913	0.9820	0.9237	0.9520	0.9925	0.9903	0.9237	0.9558
	Diff	0.0008	-0.0038	0.0127	0.0051	0.0001	-0.0068	0.0075	0.0009
ConSG	Score	**0.9923**	**0.9835**	**0.9325**	**0.9573**	**0.9936**	**0.9945**	**0.9325**	**0.9625**
	Diff	0.0018	-0.0023	0.0215	0.0104	0.0012	-0.0026	0.0163	0.0076
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549

Table 10: Results for TransR embedding technique towards 3 objectives: (O1) the best performance gain with +1.56% in F1 with GEN_ALL, (O2) ConAll variant not outperforming the other 3 variants for all metrics, and (O3) the highest performance gains +3.31%, +0.8%, and +0.45% in F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9947	0.9108	0.9287	0.9197	**0.9872**	0.9322	0.9287	**0.9305**
	Diff	0.0009	0.0233	0.0033	0.0136	0.0065	0.0481	0.0177	0.0331
ConSS	Score	**0.9949**	**0.9243**	0.9192	**0.9217**	0.9871	**0.9401**	0.9192	0.9295
	Diff	0.0011	0.0368	-0.0062	0.0156	0.0064	0.0560	0.0082	0.0321
ConHR	Score	0.9946	0.9122	0.9237	0.9179	0.9867	0.9321	0.9237	0.9279
	Diff	0.0008	0.0247	-0.0017	0.0118	0.0060	0.0480	0.0127	0.0305
ConSG	Score	0.9946	0.9072	**0.9290**	0.9180	0.9871	0.9310	**0.9290**	0.9300
	Diff	0.0008	0.0197	0.0036	0.0119	0.0064	0.0469	0.0180	0.0326
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	**0.9918**	0.9826	0.9287	**0.9549**	**0.9931**	0.9919	0.9287	0.9593
	Diff	0.0013	-0.0032	0.0177	0.0080	0.0007	-0.0052	0.0125	0.0044
ConSS	Score	0.9914	**0.9873**	0.9192	0.9520	0.9925	**0.9941**	0.9192	0.9552
	Diff	0.0009	0.0015	0.0082	0.0051	0.0001	-0.0030	0.0030	0.0003
ConHR	Score	0.9916	0.9797	**0.9290**	0.9537	0.9931	0.9919	**0.9290**	**0.9594**
	Diff	0.0011	-0.0061	0.0180	0.0068	0.0007	-0.0052	0.0128	0.0045
ConSG	Score	0.9915	0.9840	0.9237	0.9529	0.9927	0.9923	0.9237	0.9568
	Diff	0.0010	-0.0018	0.0127	0.0060	0.0003	-0.0048	0.0075	0.0019
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549

Table 11: Results for the ConvKB embedding towards 3 objectives: (O1) the best overall performance gain with +1.54% in F1 with GEN_ALL, (O2) ConAll variant not outperforming the other 3 variants for all metrics, and (O3) the highest performance gains +3.28%, + 0.58%, and +0.19% in F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9943	0.9111	0.9149	0.9130	0.9854	0.9263	0.9149	0.9205
	Diff	0.0005	0.0236	-0.0105	0.0069	0.0047	0.0422	0.0039	0.0231
ConSS	Score	**0.9949**	**0.9227**	0.9202	**0.9215**	**0.9872**	**0.9405**	0.9202	**0.9302**
	Diff	0.0011	0.0352	-0.0052	0.0154	0.0065	0.0564	0.0092	0.0328
ConHR	Score	0.9947	0.9148	0.9230	0.9189	0.9869	0.9344	0.9230	0.9287
	Diff	0.0009	0.0273	-0.0024	0.0128	0.0062	0.0503	0.0120	0.0313
ConSG	Score	0.9944	0.9067	**0.9235**	0.9150	0.9866	0.9307	**0.9235**	0.9271
	Diff	0.0006	0.0192	-0.0019	0.0089	0.0059	0.0466	0.0125	0.0297
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9909	0.9859	0.9149	0.9490	0.9922	**0.9954**	0.9149	0.9534
	Diff	0.0004	0.0001	0.0039	0.0021	-0.0002	-0.0017	-0.0013	-0.0015
ConSS	Score	0.9914	**0.9869**	0.9202	0.9524	0.9924	0.9923	0.9202	0.9549
	Diff	0.0009	0.0011	0.0092	0.0055	0.0000	-0.0048	0.0040	0.0000
ConHR	Score	**0.9915**	0.9844	0.9230	**0.9527**	0.9926	0.9923	0.9230	0.9564
	Diff	0.0010	-0.0014	0.0120	0.0058	0.0002	-0.0048	0.0068	0.0015
ConSG	Score	0.9910	0.9789	**0.9235**	0.9504	**0.9927**	0.9926	**0.9235**	**0.9568**
	Diff	0.0005	-0.0069	0.0125	0.0035	0.0003	-0.0045	0.0073	0.0019
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549

Table 12: Results for RESCAL embedding technique towards 3 objectives: (O1) the best performance gain with +1.52% in F1 with GEN_ALL, (O2) ConAll variant not outperforming the other 3 variants for all metrics, and (O3) the highest performance gains +3.38%, +0.78%, and +0.43% in F1 for pairs with high/low/no degree of lexical similarity

		GEN_ALL				GEN_TOPN_SIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9944	0.9021	0.9283	0.9150	0.9867	0.9282	0.9283	0.9283
	Diff	0.0006	0.0146	0.0029	0.0089	0.0060	0.0441	0.0173	0.0309
ConSS	Score	0.9939	0.8964	0.9167	0.9065	0.9856	0.9266	0.9167	0.9216
	Diff	0.0001	0.0089	-0.0087	0.0004	0.0049	0.0425	0.0057	0.0242
ConHR	Score	**0.9949**	**0.9171**	0.9257	**0.9213**	0.9871	**0.9343**	0.9257	0.9300
	Diff	0.0011	0.0296	0.0003	0.0152	0.0064	0.0502	0.0147	0.0326
ConSG	Score	0.9947	0.9091	**0.9294**	0.9191	**0.9873**	0.9330	**0.9294**	**0.9312**
	Diff	0.0009	0.0216	0.0040	0.0130	0.0066	0.0489	0.0184	0.0338
LexLM	Score	0.9938	0.8875	0.9254	0.9061	0.9807	0.8841	0.9110	0.8974
		GEN_RAN_SIM				GEN_RAN_NOSIM			
		accuracy	precision	recall	F1	accuracy	precision	recall	F1
ConAll	Score	0.9913	0.9774	0.9283	0.9522	0.9930	0.9913	0.9283	0.9587
	Diff	0.0008	-0.0084	0.0173	0.0053	0.0006	-0.0058	0.0121	0.0038
ConSS	Score	0.9903	0.9778	0.9167	0.9463	0.9916	0.9859	0.9167	0.9501
	Diff	-0.0002	-0.0080	0.0057	-0.0006	-0.0008	-0.0112	0.0005	-0.0048
ConHR	Score	**0.9918**	**0.9856**	0.9257	**0.9547**	0.9930	**0.9938**	0.9257	0.9585
	Diff	0.0013	-0.0002	0.0147	0.0078	0.0006	-0.0033	0.0095	0.0036
ConSG	Score	0.9917	0.9806	**0.9294**	0.9543	**0.9931**	0.9910	**0.9294**	**0.9592**
	Diff	0.0012	-0.0052	0.0184	0.0074	0.0007	-0.0061	0.0132	0.0043
LexLM	Score	0.9905	0.9858	0.9110	0.9469	0.9924	0.9971	0.9162	0.9549