A Survey of Embedding Space Alignment Methods for Language and Knowledge Graphs

Alexander Kalinowski
College of Computing and Informatics
Drexel University
Philadelphia, PA 19104
ajk437@drexel.edu

Yuan An
College of Computing and Informatics
Drexel University
Philadelphia, PA 19104
ya45@drexel.edu

October 27, 2020

Abstract

Neural embedding approaches have become a staple in the fields of computer vision, natural language processing, and more recently, graph analytics. Given the pervasive nature of these algorithms, the natural question becomes how to exploit the embedding spaces to map, or align, embeddings of different data sources. To this end, we survey the current research landscape on word, sentence and knowledge graph embedding algorithms. We provide a classification of the relevant alignment techniques and discuss benchmark datasets used in this field of research. By gathering these diverse approaches into a singular survey, we hope to further motivate research into alignment of embedding spaces of varied data types and sources.

***K*eywords** Knowledge Graph · Language Models · Entity Alignment · Cross-Lingual Alignment

1 Introduction

The purpose of this survey is to explore the core techniques and categorizations of methods for aligning low-dimensional embedding spaces. Projecting sparse, high-dimensional data sets into compact, lower-dimensional spaces allows not only for a significant reduction in storage space, but also builds dense representations with many applications. These embedding spaces have become a staple in representation learning ever since their heralded application to natural language in a technique called word2vec, and have replaced traditional machine learning features as easy-to-build, high-quality representations of the source objects. There has been a wealth of study around techniques for embedding objects, such as images, natural language and knowledge graphs, and many research agendas focused on mapping one embedding space to another, either for the purpose of aligning and unifying to a common space, applications to joint downstream tasks or ease of transfer learning. In order to fully leverage these dense representations and translate them across domains and problem spaces, techniques for establishing alignments between them must be developed and understood. To this extent, we believe this avenue of research will continue to blossom, motivating us to present this survey of current methods for alignment of representations of both text and knowledge graphs, classifying them into a taxonomy based on their mathematical approaches and requisite levels of parallel data sources.

Before diving into a survey of alignment techniques, let us first consider their motivation. In today's age, the availability of data is pervasive. In machine learning, however, methods for learning from this data are skewed to those of supervised learning where labeled data instances are required to optimize models and generalize to new applications. Labeled data is a bottleneck for the majority of machine learning projects in industry, and while methods for eliminating this bottleneck have been proposed with degrees of success [1], the luxury of labeled data is not always available without prohibitive cost.

At the same time, methods for learning a lower-dimensional representation of data in an unsupervised way have proven useful as inputs to machine learning algorithms. These representation learning algorithms, or embeddings, have become a de-facto approach for generating dense and compact feature sets, eliminating the need for tedious human engineering of features at the onset of every new task. The success of these techniques is not only related to their proven accuracy in downstream tasks, but their ability to train without supervision, thereby allowing them to scale to massive datasets.

Given the lack of available supervised data and the prevalence of strong unsupervised methods for embedding large datasets, we consider the problem of matching between embeddings of one set to another, henceforth referred to as embedding alignment. We believe that alignment methods provide a convenient method for deriving correspondences between pairs of embedding spaces, thereby providing a method for bootstrapping fuzzy labels between the source and target spaces. This problem has applications in, but not limited to, the following areas.

1.1 Language Translation

Given a word token in a source language, how can we find a translation of that token in a target language? One such way would be to define pairs of tokens in both the source and target language, creating a labeled dataset of translations. Building such a dataset would require numerous human-hours and thus may not scale well to, or even be feasible in, languages with limited lexical resources. To avoid building these hand-labeled datasets, the task of Bilingual Lexical Induction (BLI) aims to learn mappings from a source to target language in an unsupervised or semi-supervised manner [2]. A critical task in machine translation (MT) systems, BLI has been influenced heavily by embedding techniques, beginning with linear maps from one embedding space to another [3]. By finding structural similarities between two monolingual embedding spaces, such linear maps could generalize to new vocabulary tokens and aid in automated translation. Mappings between languages can also be utilized for transfer learning, where a model using embeddings as input features in one language can be re-adapted to another by simply aligning the feature spaces and re-applying the model. The field of cross-lingual word embedding alignment has rapidly grown, both in the number of evaluation benchmarks and strategies, extending past basic linear maps into the realm of deep learning models [4]. In this survey, we consider word-to-word, word-to-sentence and sentence-to-sentence alignment tasks as solutions to language translation problems.

1.2 Knowledge Integration

An ontology is a formal specification of the types of objects and relationships between those objects in a given domain. Ontologies are typically developed by ontology engineers with the goal of providing a controlled vocabulary that can be used and reused to provide exact, specific definitions that may be leveraged by humans and machines alike. Given that ontologies are domain specific, we may encounter cases where we wish to merge ontologies across two connected domains, or cases where two ontologists have independently developed ontologies for the same domain. In such cases, techniques and technologies for highlighting similarities and resolving conflicts are required. A goal of ontology integration is to develop a function that takes two ontologies as inputs and output as one merged and aligned ontology, matching like entities and relations to reduce duplicity and resolve entity ambiguity [5]. Recently, ontologies have been popularized as 'knowledge graphs' through a clever re-branding by Google [6]. Knowledge graphs explicitly frame ontologies using graph data types, allowing for advanced data representation algorithms, such as graph embeddings, to be applied for a variety of tasks, including knowledge base completion [7]. Framing ontology integration as a problem of aligning embeddings of distinct knowledge graphs, researchers have developed techniques for aligning entities in separate graphs for the purpose of forming a joint graph, many of which are surveyed in this paper [8]. In this survey, we consider graph-to-graph alignment as a solution to knowledge integration tasks.

1.3 Text Understanding and Reasoning

For consumer-facing products like online chat support and information retrieval systems, it is important to have an intuitive interface where a non-technical audience can pose a query and be served information relevant to their needs. Chatbots, question answering (QA) and retrieval systems alike need to be able to interpret user input into a series of intents, understanding the semantic roles of those intents, as well as parse out syntactic language clues such as negation. We call this broad umbrella of applications text understanding and reasoning. Many information reasoning mechanisms are built into knowledge graphs, allowing for the traversals through edges to arrive at facts and inferences. However, access to those systems typically requires technical expertise. In order to leverage the power of knowledge graph semantic reasoners, techniques need to be adapted for understanding, including, but not limited to, slot filling, semantic role labeling and sentence analogies. Harmonizing free text input and understanding with information in a knowledge graph can be seen as an alignment between these two resources, helping to improve the usability and accuracy of these systems. In this survey, we consider word-to-graph and sentence-to-graph alignments as potential solutions to these tasks.

1.4 Information Extraction

The ability to turn unstructured text data into structured, machine-operable data is a key motivation behind natural language processing tasks such as part-of-speech tagging, named-entity recognition and relation extraction, all of which can be seen as belonging to the task of information extraction. The structured information extracted can be used as inputs to many downstream tasks such as question answering, fact verification and human-computer interactions through chatbots. Here, we focus on the task of relation extraction: the ability to identity two entities and the relation between them from raw text data. The issue of supervised dataset construction is especially detrimental to research in this area; not only are labeled instances hard to collect as human labelers are known to have low precision for the task [9, 10], especially in domains such as biomedicine where subject matter experts are required, but the speed at which new entities and relations may be discussed outpaces the development of such datasets, making supervised models stale and unable to generalize quickly. To combat the problem of data collection, the technique of distant supervision was introduced [11] allowing for entity and relation triples from an ontology or knowledge graph to be used for quick label generation over raw text inputs. In this paradigm, when two entities related by a relation in the knowledge graph appear in a sentence, that sentence is labeled as being representative of that relation. This 'distant supervision' assumption and its relaxations allow for quick bootstrapping of labeled datasets, but it is well known that they also introduce a great deal of noise, as is the case when a sentence mentioning to entities expresses a novel relation, causing a false label, and are equally susceptible to missing labels when the surface forms don't match or are ambiguous [12, 13]. While distant supervision techniques still depend largely on linking the ontology and text corpus via surface forms (i.e. matching on likely string spans or candidate mentions), we anticipate a growing field of alignment between ontology and language embeddings given the increases in alignment techniques used in the two distinct data domains, a main motivating factor for the undertaking of this survey. The task of information extraction serves as our main motivation for undertaking this survey. In this light, we focus our efforts on describing techniques or gaps in current research related to graph-to-sentence alignments as this closely mirrors the task of distant supervision.

1.5 Outline of Survey

With these motivations in mind, we proceed by first presenting several of the basic methods for generating embedding spaces for language and knowledge graph data in Section 2. We then move on to describe methods for aligning these spaces in Section 3, presenting six situations in which alignments are useful and discussing existing research in these areas, where applicable. We use Section 4 to provide a categorization of approaches to embedding alignment learning. Section 5 discusses benchmark datasets used in this area of research. We conclude in Section 6 with a summary and areas of future research.

2 Embedding Models

In this section, we outline the basic methods for embedding language and knowledge data into low-dimensional vector spaces.

2.1 Word Embedding Models

Words can be viewed as an atomic unit of natural language. Taking this viewpoint, creating features for machine learning models that involve language can be time consuming. These features typically can include one-hot representations of words or counts of words involved in a sentence or document, both of which suffer from high-dimensionality and sparsity. Finding dense, lower-dimensional representations of words to replace traditional features is the focus of work on word embeddings. The first such modern approach was word2vec, an approach that leverages a shallow neural network to generate hidden state representations. By training such a network, co-occurrences between words are learned to project like-words into the same areas of the low-dimensional space, reflecting their syntactic and semantic properties. Two similar formulations, the continuous bag-of-words model (CBOW) and Skip-gram models were proposed by [14]; here, we focus on the Skip-gram model. We first define the probability of a word w_i given another word w_j as

$$p(w_i|w_j) = \frac{\exp(u_{w_i}^\top v_{w_j})}{\sum_{l=1}^{V} \exp(u_{w_l}^\top v_{w_j})}$$

where u_w is the trainable vector of input probabilities and v_w the trainable vector of output probabilities for a given word w, and V is the entire vocabulary of the given language domain. Given that the size of V is typically very large, these probabilities are estimated by leveraging negative sampling where noise words are used to generate large contrast against potential high probability guesses, eliminating the need to estimate these probabilities over the entire vocabulary for each word. With a sequence of words $w_1, w_2, \ldots, w_n$, the goal of the Skip-gram model is to maximize

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{j=-k}^{k}\log p(w_{t+j}|w_t)$$

where k specifies the window size, i.e. how far to the left and right of the centered word w_t we look when calculating the probability.

Advances and improvements over this model are plentiful. In [15] the authors develop the GloVe model, making use of a global co-occurrence matrix to address information lost by focusing on small windows during the training of word2vec models. To make the training robust to spelling errors and easier to apply to unseen words at training time, the authors of [16] build FastText, using sub-word embeddings made up of units of characters. Addressing the problem of polysemy, bi-directional neural network models are utilized to capture further context about word usage. Using the Transformer module of [17], word embeddings have matured from *static* representations of those in word2vec, GloVe and FastText, to *contextual* representations used in BERT [18], ELMo [19] and GPT-2 [20]. These models are massive in neural architecture, capturing long distance dependencies between words and interactions of words in multiple contexts. Because of their size, they have the requirement of very large training sets. Due to data and architecture size, these models are typically pre-trained, where initial weights are learned and shared, then re-trained, or fine-tuned, for target tasks or datasets. The availability of these pre-trained models though several APIs has made their use the status quo for natural language processing tasks.

2.2 Sentence Embedding Models

Building on the successes of word embedding models, a logical next step is to use words as atomic units that are composed into sentences. In this shift, moving from a discrete world where words typically represent a handful of semantic units to a continuous representation in a sentence or document, where words can be combined in infinitely many ways, represents a significant challenge. Beginning with sub-word embeddings trained using FastText, an efficient sentence classification model is established in [16] by representing a sentence as the average of its component word representations. Taking the average or sum of static vectors is a common approach to move from word representations to sentence representations, as discussed in [21], and can often beat more advanced models while retaining a level of simplicity. Even though these representations provide successful baselines, they throw out an important element of data in moving from words to sentences: word ordering. To address this, [22] propose utilizing a discrete cosine transformation. By stacking the individual word vectors $w_1, \ldots, w_N$ into a matrix, the discrete cosine transformation can be applied column-wise: for a given column $c_1, \ldots, c_N$, a sequence of coefficients can be calculated as

$$coef[0] = \sqrt{\frac{1}{N}}\sum_{n=0}^{N} c_n$$

and

$$coef[k] = \sqrt{\frac{2}{N}}\sum_{n=0}^{N} c_n \cos\frac{\pi}{N}(n+\frac{1}{2})k$$

The choice of k typically ranges from 0 to 6, where a k of zero is essentially the same as vector averaging, while higher orders of k account for greater impacts of word sequencing.

Alternative approaches abandon static word vectors and focus on the sequence of words in the sentence as the starting point. To accommodate sequential data, recursive neural networks (RNNs) dominated the field for a period of time. Recurrent neural network architectures provide an added benefit in that they can theoretically process sequences of variable length up (in practice, this is up to some max length where other sequences are padded with a special token), allowing them to train on corpora with long and short sentences. Another key advantage of RNNs is their ability to share parameters over time where signal from a prior word carries forward to the next word, and so on. This benefit of carrying information forward through the network has a downside of making them hard to train, as gradients need to be propagated backward through time; this has caused them to fall out of favor. One of the first such RNNs trained for sentence encoding was the Skip-Thought model [23]. Rather than use word-context windows, the Skip-Thought model generates an encoding for a center sentence and uses that encoding to predict k sentences to the left and right, where k is again the window size as in the Skip-gram model. To accomplish this, the model leverages an encoder-decoder architecture where each encoder step takes the sequence of words in the sentence and represents them as a hidden state, which is then encoded through the RNN. Decoding then takes place in two steps, one for predicting the next sentence and one for the prior sentence, each of which generates a hidden state through time that can be used to calculate the probabilities of each word in the sequence, with the following objective function

$$\sum_t \log P(w_{i+1}^t | w_{i+1}^{<t}, h_i) + \sum_t \log P(w_{i-1}^t | w_{i-1}^{<t}, h_i)$$

An extension of Skip-Thought is the Quick-Thought model [24]. The authors note that the objective function used in Quick-Thought is focused on re-creating the surface forms of each sentence given its dependence on the individual words represented. Specifically, the authors claim "there are numerous ways of expressing an idea in the form of a sentence. The ideal semantic representation is insensitive to the form in which meaning is expressed" [24]. The objective function of Quick-Thought is thus changed to focus only on sentence representations, using a discriminative function to predict a correct center sentence given a window of context sentences. The authors of [25] continue on the quest to capture sentence semantics with the InferSent model. InferSent uses a supervised learning paradigm where sentences in the training set are fed into a three-way classifier, predicting the degree of their similarity (similar, not similar, neutral). Coupled with a bi-directional LSTM model, the InferSent model can be pre-trained on natural language inference (NLI) tasks such as sentence semantic similarity and later used for inference or fine-tuning on other tasks.

Similar to the InferSent model, Sentence-BERT uses supervised sentence pairs to learn a similarity function [26]. The input sentence embeddings used for the three-way similarity classifier are generated from a pre-trained BERT model. Contextual models such as BERT provide an encoding of each positional word in an input sentence as their output, thus it is necessary to aggregate these contextualized representations into a single static sentence representation. As with word embedding models, this aggregation can be a sum or average of the representations at a particular layer of the language model, typically the top layer or final layer. An alternative option is to provide the model a special classification token "[CLS]" that has been pre-trained to compress the contextual representations into one layer, which can then be fed to a non-linear unit. Sentence-BERT also adapts the classification task to one of regression where cosine similarity scores are used to score the degree of similarity between sentences. This approach is useful for particular applications, such as semantic search.

Continuing on the path of larger, deeper architectures powered by more data, [27] train a Bi-directional LSTM model on a massive scale, multilingual corpus to generate sentence embeddings. Using parallel sentences across 93 input languages, the authors were able to focus on mapping semantically similar sentences to close areas of the embedding space, allowing the model to focus more on meaning and less on syntactic features. Each layer of the LASER model is 512 dimensional, with an output concatenation of both the forward and backward representation generating a final sentence representation of dimension 1024. The model outperforms BERT-like architectures for a variety of tasks including cross-lingual natural language inference, a task focused on detecting sentence similarities.

2.3 Knowledge Graph Embedding Models

Building on the success of embedding-based methods in natural language processing, these techniques have spilled into the domain of knowledge graphs. Their main motivating uses are in the task of statistical representation learning where the larger graph is compressed into a low-dimensional representation that can be used by reasoning systems, and knowledge base completion (KBC), where embeddings of existing facts can be utilized to predict new relationships between entities in the graph. Approaches in this area can be classified into three main categories: translation-based models, semantic-matching models and graph-structure models. Our aim is to introduce models leveraged in the alignment literature; for more comprehensive introductions see [7] and [28].

2.3.1 Translation-based Methods

Let $G = (E, R)$ be a knowledge graph consisting of a set of entities E and relations R, each element of which may have an entity or relation type. From this graph, we can construct the set of known facts, represented as triples $\langle h, r, t \rangle$ with $h, t \in E$ and $r \in R$. The intuition behind translation-based models is we wish to have low-dimensional, dense representations of h, r, t such that $h + r \approx t$. Model choices then depend on which space or spaces the entities and relations are embedded in as well as the scoring function used to help the model learn to differentiate between true triples from the graph and noise triples that do not reflect real-world facts. TransE [29] is the simplest of these models. It embeds both the entities and relations in the same low-dimensional vector space and uses a simple distance function defined by

$$f_r(h, t) = -\left\| h + r - t \right\|_{1/2}$$

While this model is simple, it struggles to properly encode one to many triples, where a single relation may hold between a head entity and several tail entities. Resolutions to this are addressed in TransH [30], where each relation is assigned its own hyperplane. Similarly, TransR [31] lets each relation have its own distinct embedding space ad thereby greatly expands the parameter space of the model and increases the capability of learning relation-specific translations. An entire family of translation-based models exists, each adding various complexities and constraints to the embedding spaces with novel loss functions used to best recover the relations described in the original graph.

2.3.2 Semantic-matching Models

While the translational assumption $h + r \approx t$ gives good geometric intuition as to the types of relations learned during model training, it is prohibitive for a wide class of relations, including those with anti-symmetric or complex properties. Semantic-matching models deviate away from the distance-based assumption and focus on using similarity-based scoring functions to recover facts from the low-dimensional representations of entities and relations. Rather than relying on norms and translations, these methods leverage dot-product like scoring functions to measure angles between low-dimensional representations, sometimes referred to as 'semantic energy' functions. The simplest of such models is RESCAL [32], which relies on a tensor representation of the underlying knowledge graph X, where each entry of the tensor $X_{ijk} = 1$ if the fact is represented in the knowledge graph, else-wise zero. This tensor can then be factorized into latent components,

$$X_k \approx A R_k A^\top \text{ for } k = 1, \ldots, r$$

where R_k is a matrix of dimension $r \times r$ representing interactions between each corresponding component and A contains the r dimensional representations of the entities. Thus for each (h, t) pair, we can compute the likelihood they participate in the k-th relation as

$$f_k(h, t) = h^\top R_k t$$

Contrasted with translation-based models, RESCAL takes advantage of vector products while capturing interactions between elements of each entity and all relations. In a simplification, DistMult [33] requires each R_k to be diagonal, reducing the parameters of the model while sacrificing some of its representational capacity. This reduction in capacity is especially felt when modeling anti-symmetric relations as interactions in these diagonal matrices have no notion of directionality. To circumvent this issue, the ComplEx [34] model allows for the low-dimensional representations to live in the complex space $\mathbb{C}$. The scoring function used by the ComplEx model is defined as

$$f_k(h, t) = \Re(\langle h, w_k, t \rangle) \text{ where } w_k \in \mathbb{C}^r.$$

By allowing the representations to be complex-valued, the model can handle the asymmetries of many relations present in knowledge graphs, yet score the likelihoods of facts existing using only the real-valued vectors. The work of [35] takes this one step further, defining the ConvE model where entities interact through the convolution operator. This introduces additional non-linearities through which the model can increase the capacity for learning complicated relational structures.

2.3.3 Graph-structure Models

Given that the entities and relations between them are modeled as a graph with vertices and edges, we can take advantage of the underlying graph structure to aide in creating low-dimensional representations. Graph representation learning has been a trending topic over the past few years with many advances in creating representations of graphs that can be used in machine learning models. We refer the reader to [36] for a complete introduction.

For knowledge graph embedding, path traversal techniques have been applied to learn additional facts about the relations between multiple entities in a graph, rather than just the one-hop paths learned in translation-based and semantic-matching models. By taking a walk on the graph we can learn more about the neighborhood structures of each entity, using that information to learn better dense representations. Using paths on the graph, the PTransE approach extends the traditional TransE method to capture structural information [37]. Given two entities h and t and a path $p = r_1 \rightarrow r_2 \rightarrow \ldots \rightarrow r_k$, where each r_i is a relational embedding, the authors of PTransE propose three ways of aggregating all relation vectors involved to a path vector. These include addition: $p = r_1 + \ldots + r_k$, multiplication: $p = r_1 \cdot \ldots \cdot r_k$ and application of a RNN: $c_i = f(W[c_{i-1} \colon r_i])$ where f is a non-linearity and [:] represents vector concatenation.

In a similar approach, [38] create entirely new triples from paths in the knowledge graph. If h and t are connected by the path $p = r_1 \rightarrow \ldots \rightarrow r_k$, they add a new triple $\langle h, p, t \rangle$ to the set of known triples used to train the knowledge graph. The triples can now be recovered using the translation-based loss of TransE

$$f_k(h, t) = -\left\| h + (r_1 + \ldots + r_k) - t \right\|_{1/2}$$

or used in the RESCAL context through multiplication of the relevant slices of the factorized matrix R_k

$$f_k(h, t) = h^\top (M_1 \cdot \ldots \cdot M_k) t$$

where each $M_i \in R_k$.

More recently, attention has turned to using graph convolutional networks [39] (GCNs) for knowledge graph embeddings. These techniques are called convolutional as they use neighborhood features of each node, similar to how convolutional operators look at borders of each pixel in computer vision models. By representing the knowledge graph G by its adjacency matrix A and let X be a matrix of representations (features) of the entities in the graph. The convolutions, defined layer by layer, can be represented as

$$H^{(l+1)} = \sigma(\tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} H^{(l)} W^{(l)})$$

for the $l + 1$th layer, where $\tilde{A} = A + I_N$ and I_N is the identity matrix, $\tilde{D_{ii}} = \Sigma_j \tilde{A_{ij}}$and $W^{(l)}$ is the weight matrix for each layer. Here, $H^{(0)} = X$, meaning the process begins by considering individual nodes and expands to represent their neighborhoods up to the number of layers in the network.

Translation-based methods, semantic-matching models and graph-structure models have all been used to embed individual knowledge graphs as well as aide in entity alignment between embedded graphs, as described in Section 3.4.

3 Alignment Approaches

Abstractly, learning a mapping function between two vector spaces is a well studied problem. Let $\mathcal{D}_1$ and $\mathcal{D}_2$ be two datasets, originating from either similar (as is the case for two language corpora from different languages) or different (as is the case for a set of images and a language corpus) domains. Let the functions $f_1 \colon \mathcal{D}_1 \rightarrow \mathbb{R}^n$ and $f_2 \colon \mathcal{D}_2 \rightarrow \mathbb{R}^m$ represent two mappings from the original datasets to their respective real-valued embedding spaces. Typically, n and m are of much lower dimension than the original cardinalities of $\mathcal{D}_1$ and $\mathcal{D}_2$, and therefore f_1 and f_2 can be thought of as techniques to compress the original datasets whilst maintaining their defining geometric characteristics, including a notion of 'semantic similarity'. These similarities are measured in the lower-dimensional vector spaces through techniques such as, but not limited to, Euclidean distance or cosine similarity.

Let us assume that these 'semantic similarities' are preserved by the functions f_1 and f_2. If there exists a correspondence between elements $x \in \mathcal{D}_1$ and $y \in \mathcal{D}_2$, then the problem of aligning their respective embedding spaces seeks to find a map $A \colon \mathbb{R}^n \rightarrow \mathbb{R}^m$ such that $A(f_1(x)) \approx f_2(y)$.

More generally, these methods seek to detect and exploit *invariances* between pairs of low-dimensional embedding spaces. The degree to which these invariances can be captured dictates how much training data is required to learn a reliable alignment model. In the case where the underlying geometric structures of both embedding spaces are perfectly invariant, up to a rotation of the space, simple maps may be learned in a highly unsupervised way. However, on the flip-side of the coin, methods which do not generate well structured embedding spaces may require more training data in order to learn alignments. Critically, the problem of learning an alignment map A is also tied to the choice of good embedding functions f_1 and f_2, and careful coordination between all three choices is required for finding an optimal solution.

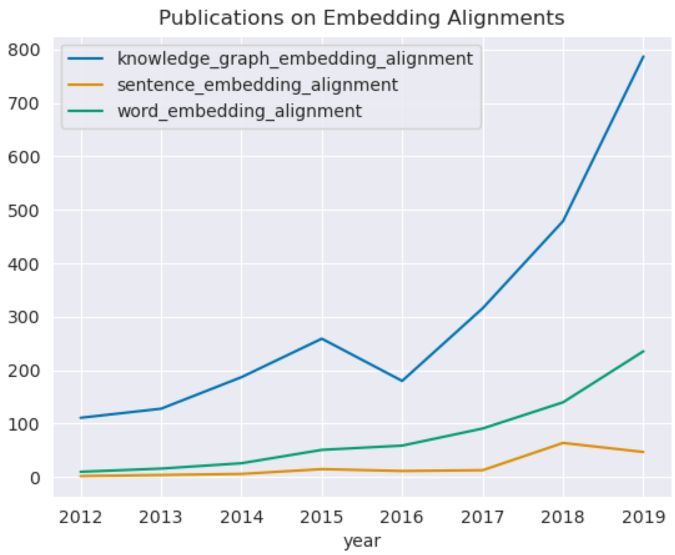
Table 1: Keyword Labels for Research Classification

Label	Keywords
Knowledge Graph	node, knowledge graph, network, ontology, knowledge base
Sentence	sentence, phrase, cross-lingual, multilingual
Word	word, token, cross-lingual, multilingual

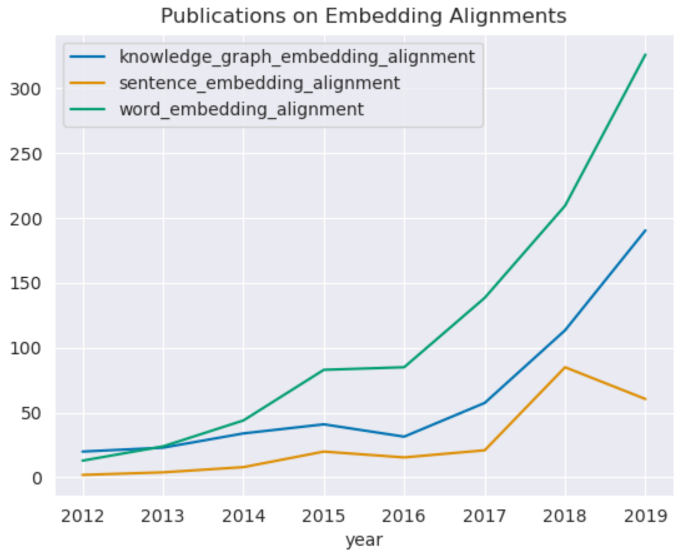
3.1 Research Landscape

As mentioned in our motivating section, we believe that embedding alignments are critical to the task of information extraction, particularly in mapping unstructured text to structured knowledge graphs. However, our hypothesis is that these techniques have yet to be fully explored in the research community, especially in learning alignments between sentence embeddings and graph embeddings. To understand the landscape of research in alignments, we undertook a search of three research repositories: ArXiv, DBLP and IEEE. Our search dates ranged from January 1, 2012 (picked to cover a period of a year before the publication of the word2vec paper) through the end of 2019 (to avoid a partial year of 2020 at the time of publication). For each repository, we conducted a keyword search for ‘embedding alignment’. We then further sub-divided the results into four categories as they pertain to embeddings: knowledge graph, sentence, word or not applicable. These categories are determined by a simple count of keyword matches in the paper’s abstract, as outlined in 1, with ties being assigned to both categories.

We then measured the trend over time for mentions of word embedding alignments, sentence embedding alignments and knowledge graph embedding alignments. The results are shown in the following figure.



We note that the inclusion of the ‘network’ keyword artificially inflates the count of publications classified as aligning knowledge graph embeddings. Many of these papers deal with embeddings of social networks, and while those networks could be considered knowledge graphs they do not fit the definition of a knowledge graph used herein. The same trend plot with network related papers removed can be seen as follows.



Per the above trends, we see that there has been a continued rise in the application of both knowledge graph and word embedding technologies. Mentions of sentence embeddings are dwarfed by the other two categories, with some of that trend explained by limitations in 3.2.2. Given the popularity of embedding techniques in the machine learning community, we believe growth in this research area will continue. We also hypothesize that alignments between other objects, i.e. tokens to graphs, will become an increasingly important field for knowledge and data integration. We proceed by enumerating the potential applications of embedding space alignments.

3.2 Alignment Use Case Enumeration

Given the focus of this survey on the domains of language and knowledge graphs, we outline six situations in which embedding space alignments could occur. In these cases, we assume that the direction of the learned alignment mapping is irrelevant, i.e. we could easily reverse the source and target spaces and learn an alignment map in the reverse direction.

3.2.1 Word-to-Word Alignment

The ultimate goal of a word-to-word alignment model is to be able to input the embedding of a token in a source language and receive as output the embedding of a semantically or syntactically similar token in the target language. As first noted in [3], word embedding models trained on distinct languages exhibited similar geometric patterns and behaviors. This observation led the authors to hypothesize that word embedding spaces could be transformed from one to another through simple linear operations, such as translation and rotation. The first attempts and models in this area took advantage of large, parallel vocabularies, where pairs of words were used to learn mapping matrices from one space to another. While learning the transformation matrix may have a closed-form solution and could be directly solved through linear algebraic methods, in practice, the weights are learned through stochastic gradient descent. We survey the common supervised learning model types in Section 4.1.1. Given the relative success and ease of implementation of these models when parallel data is available, researchers began to ask how limited that parallel set could be. Restricting to the top 5,000 most common words, restricting the parts of speech, or even relying only on numerals have been popular approaches into reducing the level of supervision needed to learn strong translation models [40]. Hybrid approaches use a form of semi-supervised learning, beginning from small seed lexicons and iteratively adding words as confidence in their direct translation builds. We introduce these semi-supervised methods in Section 4.1.2. Moving past semi-supervised methods, approaches to learning mappings between embedding spaces in a completely unsupervised way. These methods rely on the geometric structures of the underlying spaces as a proxy for parallel data, either relying on embedding similarity distributions [41], adversarial learning [42] or metric recoveries via optimal transport [43]. We cover these methods in Section 4.1.3.

3.2.2 Sentence-to-Sentence Alignment

Sentence to sentence alignment often serves as an entry point to machine translation applications. Given a parallel corpus of sentences in two languages, the goal is to learn a mapping function f that converts a low-dimensional representation of sentence $s_1 \in {}_1$ to a close (in terms of vector space proximity) representation $t_1 \in {}_2$. This map can then generalize for future translations such that $f(s_2) \approx t_2, s_2 \in L_1, t_2 \in L_2$. This approach is limited due to two factors. First, the availability of such parallel corpora is limited. Most research in this area either relies on the Europarl dataset [44] or translations of the Bible. Neither of these resources represents enough diversity in language to scale up to production-level systems, but they do allow for ideas to be tested experimentally. The second limitation comes from the composition of semantic units (i.e. individual word tokens) to higher order representations in sentences. Word order plays a role in the structure of languages, thus simple mapping models have been replaced with those that model the sequences of tokens, such as the seq2seq model [45].

3.2.3 Word-to-Sentence Alignment

Given the availability of technologies for word-to-word and sentence-to-sentence alignment, there has been little need for additional research in word-to-sentence alignment. In the case of a set of sentences being mapped to a finite set of words, this is typically handled as a supervised classification problem where the finite set of words is one-hot encoded to represent a target variable. We direct the reader to [16] for efficient approaches to this type of supervised classification problem.

3.3 Sentence-to-Sentence Alignment

While the goal of word-to-word alignment is to map tokens for direct translation, these tokens often can express multiple senses and thereby exhibit polysemy. This creates issues in direct, one-to-one mappings due to the fact that the training data can contain a particular token in the source space with multiple translations in the target space, leading to conflicting information during training and at inference. Rather than focusing on tokens as the atomic unit, tokens in a given context, either through phrases or complete sentences, carry more information that can be leveraged for better alignment. The research area of sentence-to-sentence alignment relies on parallel documents, typically found in resources such as translations of the European Parliament proceedings or the Bible. While alignment of parallel word tokens is a rich research field, there has been less focus on alignment techniques of full sentences; research in this area typically falls under the umbrella of machine translation where more complex sequence-to-sequence neural models are favored. However, many of the same techniques for aligning word embeddings can be leveraged for aligning sentences provided we can generate representative sentence embeddings. We cover a handful of sentence embedding methods in 2.2 and discuss their alignment in section 4.3.

3.4 Knowledge-to-Knowledge Alignment

Knowledge graphs have seen a great deal of interest and hype in recent years as their applications to artificial intelligence and machine learning have come to be seen as the onset of a ‘third wave’ contributing to semantically grounded and explainable AI. They also serve as the backbone to the Semantic Web, a set of standards for defining and linking data and meta-data in a machine-readable and human-interpretable way. While large corporations like Google and LinkedIn have massive knowledge graphs at their disposal, smaller, more tailored graphs exist for specific purposes such as SnoMed for medical clinical terminology and FIBO for financial industry concepts. Given the specificity of some of these smaller graphs, we may wish to weave several of them together for a particular application, including data exchange protocols and data integration tasks. We may also want to merge knowledge graphs covering similar, yet independently defined, concepts, or graphs defining the same subject matter across languages. This task is referred to in the literature as *entity alignment*: the process of identifying nodes in each graph that are referencing the same semantic concept and either forming relations between them or compressing them into a single representation. Work in this area originated in the task of ontology alignment [46], which aimed to use heuristics, string matching and natural language processing techniques to map source and target nodes. As in the other application areas in this survey, the push to deep, representational learning has invaded the space of ontology alignment as well, typically couched under the banner of entity embedding alignment. The task at hand is to create low-dimensional representations of the source and target knowledge graphs and use only these embeddings to automatically discover alignments. As in word-to-word alignments, methods range from directly supervised methods where parallel entities between graphs are used to learn mappings, to fully unsupervised methods where inferences are made to align entities based on the structure of their neighborhoods in the graph. We cover these techniques in Section4.2

3.4.1 Word-to-Knowledge Alignment

In the previously explored cases, embeddings of source and target data from similar domains were aligned. In these cases, strings are mapped to strings and graph entities to other graph entities. This section deviates from those proceeding by considering alignments from strings to knowledge graph entities. For a given token $x \in \mathcal{D}_1$, we wish to identify a corresponding entity $y \in \mathcal{D}_2$, if such an entity exists. One such way of finding these correspondences is to find a map between the token embedding $f_1(x)$ and the entity embedding $f_2(y)$. Given that the target domain (a knowledge graph) is constructed to reflect facts about real world entities and the relations between them, we expect to find those same facts and entities referred to in the source space (language), although with much lower precision and exactness in their statements. While inherent noise present in human language makes learning such an alignment challenging, success in this area can assist with knowledge driven entity extraction and named entity recognition.

3.4.2 Sentence-to-Knowledge Alignment

Our main motivation in this line of research is the alignment of sentences to knowledge graphs. The interest in this problem is two-fold. Firstly, if we are able to align embeddings of triples $\langle h, r, t \rangle$ from the knowledge graph G to sentence embeddings s in a given corpora, these alignments can be used to detect the expression of relationships r in the sentences, aiding in the task of relation extraction. Secondly, in the opposite direction, if we can align sentences to triples, we can use this technique to assist in the detection of new triples to be added to the knowledge graph from text data, aiding in the automated expansion of a knowledge graph. These two problem domains can be viewed as complementary techniques for converting unstructured data in text documents to structured data in a knowledge graph. Having data in a structured format not only makes it easier for human verification, as in the case of automated fact checking, but also allows for insights into how other machine learning models, such as document classification, are leveraging unstructured data, providing an avenue for explainable AI and model governance.

4 Alignment Learning Paradigms

In this section, we explore the major techniques used in each of the six categories defined above. Each section reviews the works from the perspective of classifying them into supervised, semi-supervised and unsupervised frameworks, motivated by our desire to assess the requisite amount of parallel data necessary to learn an alignment.

4.1 Word-to-Word Alignment Techniques

We proceed by classifying word-to-word alignment techniques into supervised, semi-supervised and unsupervised methods.

4.1.1 Supervised Methods

Supervised learning methods are the most common and most data intensive in machine learning applications. To help alleviate the burden on developers of these methods, leveraging unsupervised methods as discussed above helps to ingest large amounts of data and build robust features to jumpstart learning. In this section we discuss supervised models that use unsupervised features as inputs with the goal of aligning these resources.

Regression Models Regression models form the class of solutions first used to address the word-to-word embedding alignment problem. Let us begin by defining languages L_s and L_t, our source and target language, respectively, and embedding functions $f_1 : L_s \to \mathbb{R}^n$ and $f_2 : L_2 \to \mathbb{R}^m$. Given a set of parallel translation tokens (w_i^s, w_i^t) where $w_i^s \in L_s$ and $w_i^t \in L_t$, we wish to learn a transformation matrix W to minimize the following mean-squared loss

$$\sum_{i=1}^{n} \left\| W f_1(w_i^s) - f_2(w_i^t) \right\|^2$$

This method was first proposed by [3] as a means of capturing geometric patterns between embeddings across embedding spaces. In the original paper, no additional pre-processing is done on the input word vectors, which were generated using the CBOW word2vec algorithm. The transformation matrix W can then be applied to a new vector $f_1(w_N)$ to map it into the target space where a cosine similarity search can rank all translation candidates. Subsequent papers suggested minor tweaks to the regression model having significant impacts on the capacity to learn. These include the addition of l_2 regularization [47] and adding pre-processing steps such as embedding vector unit normalization, further discussed in the following section.

Orthogonal Models The original regression model utilized a Euclidean distance in learning the transformation matrix, yet relies on cosine similarity to carry out similarity searches in the target space. This inconsistency was first noted by [48] who in turn modified the regression process to add unit length normalization to the source and target vector spaces and constrain the matrix W to be orthogonal, that is $W^\top W = I$ where I is the identity matrix. The pre-processing step and orthogonal constraint then line up with the retrieval method, where we are less concerned with distances between vectors and more concerned with the angles between them. Solutions to this minimization problem are still carried out by stochastic gradient descent where the orthogonality constraint is implemented by solving the SVD problem, typically done by mini-batch fed to the optimizer.

Applications of pre-processing and orthogonal constraints spurred further research into ways to manipulate the source and target embedding spaces to further express their geometric structures. In [49] and [50], the authors evaluate several pre- and post-processing steps, building towards a framework of applicable methods. The steps in this framework are enumerated as follows:

- Normalize the source and target spaces, either using unit norms or mean centering (where each component/feature is forced to have zero mean) as an initial pre-processing step;
- Feature whitening, requiring each feature to have unit variance and removing their correlations, applied to both source and target space independently;
- Learning an orthogonal mapping via the regression technique;
- Re-weight the features to increase their correlations between source and target spaces, only applied if whitening was applied prior to learning the mapping;
- De-whitening to capture the variance of the original embedding spaces, applied only if whitening was applied prior to learning the mapping; and
- Reducing the dimension by only keeping the most important components of the source and target spaces, helping to remove noise captured in the tail components.

The authors show that combining these steps helped them achieve superior performance when using CBOW word vectors and 5,000 supervised training examples. The full framework has been packaged and released as open source code under the moniker VecMap, and is used as a baseline in many comparative surveys.

Margin Models The methods of the prior two sections rely on variations of mean-squared error to compute and learn from the differences between the source and target space. An alternative modeling technique leverages a max-margin based loss function. These objectives seek to reward the weights associated with positive pairs (in this case, words that are direct translations) while reducing the signal from noise pairs generated either randomly or using a heuristic. In the case of word-to-word translation, the association between pairs is defined by their cosine similarity, thus we may define the max-margin loss as

$$\sum_{i=1}^{n}\sum_{j\neq j}^{k}\max\{0, \gamma - \cos(Wf_1(w_i^s), w_i^t) + \cos(Wf_1(w_i^s), w_j^t)\}$$

where k represents the number of noise pairs (negative samples) and γ is a parameter fixed for setting the margin between positive and negative cases. Using this objective was first proposed by [51] to address issues of hubness seen in regression and orthogonal techniques. The presence of hubs is driven by embeddings that dominate the space due to their high cosine similarity with all other vectors in the source or target space. These hubs can be caused by the overall frequencies of words in the underlying corpus [52], a common mean vector present in all word vectors causing issues of anisotropy in the embedding spaces [53], or issues derived from least-squares regression where low variance points are all grouped together in the target space.

Margin-based models are also explored in [54], where the authors also aim to combat the issue of hubs by introducing a new retrieval criteria. The cross-domain similarity local scaling (CSLS) is defined as

$$CSLS(x, y) = -2\cos(x, y) + \frac{1}{k}\sum_{y'\in N_Y(x)}\cos(x, y') + \frac{1}{k}\sum_{x'\in N_X(y)}\cos(x', y)$$

where $N_Y(x)$ is the set of k nearest neighbors of x in the target space. The authors build this retrieval criteria into their margin model by using unpaired words (those with no explicit translation in the training set) as negative samples when computing nearest neighbors. The full objective function, called the relaxed CSLS (RCSLS) is then computed as

$$\frac{1}{n}\sum_{i=1}^{n} -2\cos(Wf_1(w_i^s), w_i^t) + \frac{1}{k}\sum_{w_j \in N_Y(Wf_1(w_i^s))} \cos(Wf_1(w_i^s), w_j) + \frac{1}{k}\sum_{Wf_1(w_j^s) \in N_X(w_i)} \cos(Wf_1(w_j^s), w_i)$$

For margin-based methods, RCSLS is the most popular method, used as a benchmark for comparisons to other methods [55, 56]. While competitive, we find margin-based methods are studied less frequently; we conjecture this is due to the difficulty in selecting informative negative samples and the preference for methods using as little supervision as possible.

Other Approaches In the interest of exploring methods that extend past word-to-word alignment and are able to generalize to other embedding spaces, we briefly mention alignment methods that lie outside the three categories noted above. The first such method relies on the word neighborhood structures, based on the assumption that for a neighborhood of points in the source space the neighborhood can be reconstructed after applying a linear mapping to the target space. By using manifold learning, without making the assumption that the manifolds (or embedding spaces) were learned via the same algorithm, to capture neighborhood structures, [57] propose a new locality preserving loss function. Given embedding spaces as manifolds M^s and M^t, the goal is to learn a mapping $f : M^s \to M^t$, which is optimized based on three pieces: an orthogonal piece; a mean-squared error piece; and an additional locality preserving loss piece. This approach represents the base of several models, encapsulating the regression class of models and the orthogonal constrained class of models while adding structure preserving pieces for regularization. The entire loss function can be written as

$$\mathcal{L} = \mathcal{L}_{mse}(\theta_f) + \beta\mathcal{L}_{lpl}(\theta_f, W) + \mathcal{L}_{orth}(W)$$

where $\mathcal{L}_{mse}$ and $\mathcal{L}_{orth}$ are as defined in prior sections and

$$\mathcal{L}_{lpl}^{i} = \left\| m_i^t - \sum_{m_j^s \in N_k(m_i^s)} W_{ij} \cdot f(m_j^s) \right\|^2$$

with β a constant to control the influence of the LPL loss and $m_q^s \in M^s, M_q^t \in M^t$. The locality preserving loss was shows to assist in both sentence space and word space alignment, particularly when training dataset sizes are limited.

4.1.2 Semi-Supervised Methods

In instances where full parallel corpora or dictionaries are not readily available it is possible to use smaller seed lexicons to build toward larger datasets in a semi-supervised way. One such way of building pseudo-dictionaries' is to identify words that are expressed as the same string in both the source and target language [58]. These typically occur for proper nouns and abbreviations such as FBI and Microsoft. According to [58], this procedure was able to generate nearly 47,000 translation pairs between English and Italian, much larger than the 5,000 most popular terms used by many supervised methods, with excellent evaluated levels of precision.

Aside from string matching, other alternative corpus construction methods include using very small seed lexicons (on the order of 25 word pairs) and iteratively adding candidate pairs. The work of [40] propose alternating between a step for learning the mapping W similar to those in Section 4.1.1, followed by a dictionary induction step. Given embedding spaces X and Z, let D be the binary matrix representing the word-pair dictionary between the two languages, i.e. $D_{ij} = 1$ when word i of the source is aligned to word j in the target language. The mapping matrix can then be defined as

$$W^* = \arg\min_W \sum_i \sum_j D_{ij} \left\| WX - Z \right\|^2$$

At each step of processing, updates to D are computed as $D_{ij} = 1$ if

$$j = \arg\max_k (X_{i*}W^*) \cdot Z_{k*}$$

otherwise $D_{ij} = 0$. To evaluate the efficacy of this method, small seed dictionaries are sampled ranging in size from 25 to 2,500 entries, as well as experimenting with only aligning numerals (i.e. digits 0 to 9). Given the limited training set, this self-learning paradigm is competitive with, and at times outperforms, supervised methods.

The work of [59] further explores iterative learning, alternating between supervised alignment and unsupervised distribution matching, as explored in the next section, as well as introducing novel metrics to assess the orthogonality assumptions used in supervised approaches. We further unpack these notions in Section 4.1.3.

4.1.3 Unsupervised Methods

Under the goal of restricting the amount of parallel data needed to create an alignment between two word spaces, several approaches have been proposed that attempt to leverage the structure of the embedding space itself, completely removing the need for parallel data. A key approach was described in [42] where the authors propose leveraging an adversarial learning paradigm. In this setup, the goal is still to learn a linear map W between the source embedding vectors $f_1(w_i^s)$ and target space embeddings $f_2(w_i^t)$. A discriminator D is trained to recognize and separate the mapped embeddings $Wf_1(w_i^s)$ from $f_2(w_i^t)$, while an adversarial generator G is trained to fool D. The given loss functions for both models are

$$\mathcal{L}(\theta_D|W) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\theta_D}(source = 1|Wx_i) - \frac{1}{m}\sum_{i=1}^{m} \log P_{\theta_D}(source = 0|y_i)$$

for the discriminator model, and

$$\mathcal{L}(W|\theta_D) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\theta_D}(source = 0|Wx_i) - \frac{1}{m}\sum_{i=1}^{m} \log P_{\theta_D}(source = 1|y_i)$$

for the generator model. With an initial linear map W learned, the authors then apply a refinement procedure by identifying anchor points as pairs that were frequently identified as translations in the prior step. The anchor points and their corresponding word frequencies are used to solve the orthogonal Procrustes problem to generate a refined mapping matrix W^*. This final matrix is used in conjunction with the CSLS objective described in prior sections to mitigate hubness and areas of density in generating a final translation from source to target. The adversarial method has been utilized to generate large benchmark datasets under the name Multilingual Unsupervised or Supervised Embeddings (MUSE), releasing parallel embedding spaces trained using FastText in 110 languages.

As previously discussed, word embedding models tend to reflect the frequency of word usage in the underlying language. While the adversarial method directly leverages word frequencies, an alternative unsupervised method in [41] captures these patterns by analyzing the similarity distributions of the word vectors themselves. By constructing a pair-wise similarity matrix of all word embeddings in the source and target languages, trends in their usages can be exploited to create an initial seed dictionary. By pre-processing to unit normalize the embeddings in both the source X and target Z spaces, these similarity matrices can quickly be computed as $M_X = XX^\top$ and $M_Y = YY^\top$. To further reduce the complexity of finding maps between these similarity matrices, each similarity matrix can then be sorted row by row to identify the most influential embedding dimension and nearest neighbor searches can then be executed to generate candidate pairs. The seed dictionary can then be expanded using semi-supervised methods described in [40].

Aside from leveraging similarity distributions of the underlying embedding spaces, methods are also available to treat these embedding spaces as metric spaces, adopting mathematical tools from measure theory and topology to describe their nature. One such metric is the Gromov–Wasserstein distance used to compare two pairs of spaces, rather than the pairwise point-by-point metrics such as similarities. Using this metric, [43] transform the alignment problem to one of finding an optimal transport from source X to target Z. Due to computational costs, the problem is split into two steps where the two spaces are first aligned using the explicit optimization to find an optimal coupling followed by a refinement using an orthogonal Procrustes procedure, as in [42].

4.2 Graph-to-Graph Alignment Techniques

As in word-to-word alignments, graph-to-graph alignment techniques can be classified into supervised, semi-supervised and unsupervised methods. Within each paradigm, however, it is slightly more complicated to develop a straight-forward classification of techniques. We posit this is due not only to the variety of graph datasets available but the velocity at which new research is being published, as noted in Section 3.1. We proceed by categorizing techniques by their level of parallel data needed to learn a robust model. Where applicable, we will also classify techniques according to their approach to the source and target graph embeddings, noting if they utilize translation-based, semantic-matching or graph-structure models.

4.2.1 Supervised Methods

To address issues of coverage in cross-lingual knowledge graphs, [60] propose a method for embedding knowledge graphs in a source and target language and automatically learning alignments between them, called MTransE. Leveraging the translational-based TransE algorithm for generating embeddings of each monolingual knowledge graph G_i and G_j, the embedded triples of each graph are then fed through an alignment scoring function S_a, where the total alignment score is calculated as

$$S_A = \sum_{(T,T') \in \delta(G_i,G_J)} S_a(T,T')$$

where $\delta_{(G_i,G_j)}$ represents the supervised set of pre-aligned triples. The authors propose three main classes of functions for S_a: distance-based measures, translation vectors and linear transformations. For distance-based measures, the triples in G_i and G_j can be represented as a function of the difference in the head and tail entities

$$S_{a_1} = \|h - h'\| + \|t - t'\|$$

or adjusted to also represent differences in the relation embeddings

$$S_{a_2} = \|h - h'\| + \|r - r'\| + \|t - t'\|$$

An alternative approach focuses not only on the differences in individual components of the triples, but allows for translation vectors to be learned between the entities and relations, as defined by

$$S_{a_3} = \left\|h + v_{ij}^e - h'\right\| + \left\|r + v_{ij}^r - r'\right\| + \left\|t + v_{ij}^e - t'\right\|$$

where $+v_{ij}^e$ and $+v_{ij}^r$ are learnable translations such that $e + v_{ij}^e \approx e'$. The final class of functions defines learnable linear transform matrices, with one focused on translations between entities

$$S_{a_4} = \left\|M_{ij}^e h - h'\right\| + \left\|M_{ij}^e t - t'\right\|$$

and another with learnable transformations for both entities and relations

$$S_{a_5} = \left\|M_{ij}^e h - h'\right\| + \left\|M_{ij}^r r - r'\right\| + \left\|M_{ij}^e t - t'\right\|$$

The authors conclude that the linear transformation models work best with limited differences between the model with entity transformations and the model with both entity and relational translations.

The methods of MTransE lean heavily on the underlying translation model of TransE, which, while directly addressing similarities in graph structures in each individual space, largely ignores other information sources contained in the knowledge graph such as entity types and attributes. Relying only on the structured embedding approaches also leads to issues when the distribution of relations in the knowledge graph is skewed, as has been widely observed in many large-scale knowledge graphs [61, 62]. To leverage both structure and attributes in aligning both graphs, a joint attribute-preserving embedding (JAPE) module is presented in [63]. For the structure embedding piece, the authors again leverage a translation-based approach letting $f(p) = \|h + r - t\|^2$ where p is a known triple from the supervised training set. They then slightly modify the training process by using a training set P that has pre-aligned triples in both the source and target space to capture correspondences between entities sharing similar relationships. This accomplishes the alignment of both cross-lingual entities and their relations in a single computational step and can be seen as optimizing the score of

$$\mathcal{L}_{\mathcal{SE}} = \sum_{p \in P} \sum_{p' \in P'} (f(p) - \alpha f(p'))$$

where α is a tunable margin hyper-parameter and P' a set of negative samples. In addition, attributes of the entities such as their data type and correlations between relation occurrences are used to generate attribute embedding vectors using a skip-gram like objective function

$$\mathcal{L}_{\mathcal{AE}} = - \sum_{(a,c) \in H} w_{a,c} \cdot \log p(c|a)$$

where H is the set of pairwise positively correlated attributes, i.e. when entity e_j has attribute a it is also highly likely to have attribute c. The attribute embeddings are then used to compute three similarity matrices $S^(1), S^(2), S^{(1,2)}$ representing the inner-graph entity attribute similarity scores as well as the cross-graph attribute similarity scores. This additional data from the training set helps to build more support for entities to be aligned that can not be captured when using only translational-based models and can be combined to minimize the objective

$$\mathcal{L}_{\mathcal{S}} = \left\| E_{SE}^{(1)} - S^{(1,2)} E_{SE}^{(2)} \right\| + \beta(\left\| E_{SE}^{(1)} - S^{(1)} E_{SE}^{(1)} \right\| + \left\| E_{SE}^{(2)} - S^{(2)} E_{SE}^{(2)} \right\|)$$

The structured and attribute losses can then be jointly optimized for learning the entire model through

$$\mathcal{O}_{\mathcal{JOINT}} = \mathcal{O}_{\mathcal{SE}} + \delta \mathcal{O}_{\mathcal{S}}$$

where δ is a tunable hyper-parameter to moderate the influence of the attribute similarities.

Having demonstrated the importance of incorporating both structure and attributes into the alignment process, other authors followed in the footsteps of the JAPE model, although with different underlying embedding techniques. In the work of [64], the authors utilize the graph convolutional network architecture (GCN-EA) to embed entities from the training sets into an aligned space. By creating a bipartite graph between the pre-aligned entities in the training set the GCN-EA approach models the edges between each distinct graph as equivalence relations, discovering other equivalence relations as alignments by encoding neighborhood information. This approach is then further refined by incorporating the entity attributes in an additional convolutional layer after entity embeddings have been defined. Given two graphs G_1 and G_2 and a training dataset of pairs of matched entities from each, i.e. $S = (e_{m_1}, e_{m_2}), e_{m_1} \in G_1, e_{m_2} \in G_2$, we define two parallel GCN models GCN_1 and GCN_2 to generate embeddings of each input graph. Each GCN outputs a vector representation of a given input entity, call them v_{m_1} and v_{m_2} for e_{m_1} and e_{m_2}, that can be seen as the concatenation of two parts. The first piece of the output vector v_{m_j} represents the structural piece from the convolutional network with dimension d_s. The second piece represents the attribute representation embedding from the next layer of the network, with dimension d_a. Thus each vector v_{m_j} is of $d_s + d_a$ dimension and compactly represent the structure and attributes of the particular input entity. These representations are then fed to a distance matching function

$$D(x, y) = \beta \frac{f(h_s(x), h_s(y)}{d_s} + (1 - \beta) \frac{f(h_a(x), h_a(y)}{d_a}$$

where $f(a,b) = \|a - b\|$, h_s and h_a take the structure and attribute piece of the embedding, respectively, and β is a hyperparameter that balances the trade-off between the importance of structure and attributes. The distances between pre-aligned entities in the training set can then be back-propagated through the network using a margin-based criteria, one for the structure embeddings and one for the attribute embeddings

$$\mathcal{L}_s = \sum_{(x,y)\in S} \sum_{(x',y')\in S'} [f(h_s(x), h_s(y)) + \gamma_s - f(h_s(x'), h_s(y'))]_+$$

$$\mathcal{L}_a = \sum_{(x,y)\in S} \sum_{(x',y')\in S'} [f(h_a(x), h_a(y)) + \gamma_a - f(h_a(x'), h_a(y'))]_+$$

with γ_s and γ_a as margin hyper-parameters.

Thus far, the three KG-to-KG alignment methods explored, namely MTransE, JAPE and GCN-EA, have focused only on the problem of aligning entities between the two input graphs, and while attribute information from the graphs has also been included, little has been done to also factor in the relations and relational-types in each individual graph. Incorporating relation information is an important facet for selecting an approach to aligning an input source to a knowledge graph embedding, especially in the application to relation extraction from text documents. Equally important in capturing relational data is accounting for directionality; methods must be able to distinguish between one-to-one, many-to-one and many-to-many relation types. Rather than solely relying on aligned entities, [65] create a training set of aligned entities $S_e = (e_{m_1}, e_{m_2}), e_{m_1} \in G_1, e_{m_2} \in G_2$, and aligned relations $S_r = (r_{m_1}, r_{m_2}), r_{m_1} \in G_1, r_{m_2} \in G_2$ in a multi-mapping relation aware technique dubbed MMEA. The authors proceed by defining their own knowledge graph embedding process, avoiding the pitfalls of translation-based methods by defining their own embedding process, called DistMA. DistMA works by replacing translation-based distances with inner-products, defined by

$$E_1(h,r,t) = \langle v_h, v_r \rangle + \langle v_r, v_t \rangle + \langle v_h, v_t \rangle$$

and replace the margin-based optimization with a logistic loss function, defined by

$$-\sum_{(h,r,t)\in S^+} \log \sigma(E_1(h,r,t)) - \sum_{(h',r',t')\in S^-} \log \sigma(-1 \cdot E_1(h',r',t')) + \lambda \|\theta\|$$

Using the inner product rather than subtraction-based distances allows the embeddings to scale well to multi-relational facts in the graph, where methods like TransE typically struggle. The downside is that the proposed DistMA is highly symmetric, making no distinction between the head and tail of the triple, thus incorporating no sense of the directionality of the relation. To combat this, the authors also leverage the ComplEx embedding method, previously presented in Section 2.3.2. Letting

$$E_2 = \Re(\langle w_h, w_r, w_t \rangle), w_i \in \mathbb{C}$$

the final scoring function for each triple is combined and written as

$$E(h,r,t) = E_1(h,r,t) + E_2(h,r,t)$$

With both entities and relations embedded for each knowledge graph, the airs from the training set are aligned in a common embedding space such that their representations are equal. This is accomplished by using a cosine similarity metric

$$sim(e_i, e_j) = \langle \|v_{e_i}\|, \|v_{e_j}\| \rangle$$

to build a similarity matrix$S_{1,2}$ between the two knowledge graphs. This similarity matrix can then be ranked from both directions, i.e. for the similarities $M_{1,2} : G_1 \rightarrow G_2$ and $M_{2,1} : G_2 \rightarrow G_1$. The final ranking matrix is then computed as $M = M_{1,2} + M_{2,1}^{\top}$.

In continuation of research on addressing multi-relational patterns, [66] focus on Non-Translational Alignment for Multi-relational networks (NTAM). Rather than relying on a semantic energy, translational-based, or graph convolutional model to build embeddings, the authors build a probabilistic model based on *motifs* that can be found within the graph. These motifs, or graph patterns, include triangular structures in the graph where a given node in the triangle can have in-degree of zero (out-degree two), one (out-degree one) or two (out-degree zero), as is accomplished in [67]. While these motifs are flexible in capturing local structures in the graph, aligning these structures required nodes in each individual graph to exhibit very similar neighborhood structures, an assumption that may not hold in large-scale heterogeneous graphs.

For the majority of knowledge graph alignment approaches discussed above, the underlying embedding algorithms rely on negative samples, or false facts, to be generated. These negative sampling paradigms inherently make use of the closed-world assumption wherein all facts are assumed to be contained in the knowledge graph. In opposition is the open-world assumption, where we have only build a knowledge graph of our currently known facts, and the validity of those not contained in this set is uncertain. Using negative sampling instantiates a closed-world assumption, and when those negative facts turn out to actually be true, model performance suffers. In opposition, adversarial networks leverage two networks that attempt to trick one another. The first network, the generator, attempts to create samples that look similar to those in the original data distribution, yet are created in a synthetic way. The job of the second network, the discriminator, is to differentiate between instances of the true dataset versus those coming from the generator. Adversarial networks have been utilized to generate embeddings of single knowledge graphs [68] and can also be used in aligning the representations of distinct knowledge graphs. Based on the notion that the embedding spaces of each graph should have similar spatial features for entities that are likely the same, called by the authors of [69] the embedding distribution, an adversarial network can be used to learn to discriminate between these embedding distributions in order to learn an approximate isomorphism between the two spaces. To accomplish this task, the authors introduce the representation module, the mapping module and the adversarial module. In the representation module, two separate instances of the TransE model are trained on each graph, creating two sets of embeddings e_s and e_t. These embedding matrices are then fed into the mapping module, where seed pairs $S = (e_{s_i}, e_{t_i})$ are used to learn a linear mapping, defining the loss function as

$$L_M = \sum_{(e_s, e_t) \in S} \|Ge_s - e_t\|$$

As in approaches for word-to-word embedding alignment, G can be restricted to be an orthogonal matrix. The authors introduce two additional constraints: the feature reconstruction constraint and the mapping reconstruction constraint. Intuitively, the feature reconstruction constraint dictates that once an embedding for an entity is mapped from the source space to the target space, that same mapping can be applied to map it back to the source space representation. This can be reflected in the adjusted loss function

$$L'_M = \sum_{(e_s, e_t) \in S} \lambda_1 \|Ge_s - e_t\| + \lambda_2(\mu \left\|e_s - G^{\top} Ge_s\right\| + (1 - \mu) \left\|e_t - GG^{\top} e_t\right\|)$$

where λ_1, λ_2 are learnable weights to balance the reconstruction constraint and μ is a harmonic factor. The mapping reconstruction constraint is a variant on restricting G to be orthogonal, forcing the learning algorithm to push G toward the nearest orthogonal manifold, modifying the loss function as

$$L''_M = \sum_{(e_s, e_t) \in S} \lambda_1 \|Ge_s - e_t\| + \lambda_2 \left\|G^{\top} G - E\right\|_F$$

where E is the identity matrix and F is the Frobenius norm. With a mapping learned, the mapped embeddings can be fed to the adversarial module where their synthetic counterparts are build by a generator network while the discriminator network learns to differentiate between true and false examples. The authors show that the adversarial setup helps with generalization as its main focus is on aligning the topological features of each embedding space in a way which reduces sensitivity to noise.

4.2.2 Semi-Supervised Methods

While the issue of building supervised word-to-word datasets is a challenge for word alignment techniques, the issue is even more prevalent in knowledge graphs due to their large degree of heterogeneity. Building links between entities in disparate knowledge graphs often requires the intervention of human experts and comes at a significant cost. Rather than rely solely on labeled instances, semi-supervised approaches build from a set of seed aligned entities, iteratively building confidence in newly aligned pairs and expanding the set of training examples.

By embedding the separate graphs using translational-based methods, the authors of [70] build a joint embedding space and utilize a soft alignment scoring function to estimate a reliability score of aligned entities. These three modules are trained in an iterative fashion, making updates to the training set and the joint embeddings at each step. For the individual embeddings, the authors utilize the path-inclusive embeddings of PTransE [37], generating two entity embedding sets E_1 and E_2. To build a joint embedding space, the authors propose three methods: a translation-based model, a linear model and a parameter sharing model. The translation-based model, IPTransE, introduces a new alignment relation r that maps $e_s \in E_1$ to $e_t \in E_2$, where

$$E(e_s, e_t) = \left\| e_s + r^{(E_1 \rightarrow E_2)} - e_2 \right\|$$

The linear mapping model replaces this relation with a transformation matrix M such that

$$E(e_s, e_t) = \left\| M^{(E_1 \rightarrow E_2)} e_1 - e_2 \right\|$$

The parameter sharing attempts to make no mapping, instead forcing aligned entities from the seed set $\mathbb{L}$ to have the same embedding representation, such that

$$e_s \equiv e_t, (e_s, e_t) \in \mathbb{L}$$

Once entities are projected into a joint space, each entity embedding in the source space is compared to all entity embeddings in the target space, building an aligned entity where

$$\hat{e_t} = \arg\min(E(e_s, e_t)), E(e_s, \hat{e_t}) < \theta$$

where θ is a hyperparameter controlling the distance. The pair $(e_s, \hat{e_t})$ can then be added to $\mathbb{L}$ and the joint alignments can be updated accordingly. In addition to simply updating the set $\mathbb{L}$, the authors also introduce a soft alignment function

$$R(e_s, e_t) = \sigma(k(\theta - E(e_s, e_t))$$

that tracks the reliability of the new pair, where σ is the softmax function and k is a tunable hyperparameter.

Also leveraging translational-based models for embedding each knowledge graph, semi-supervised entity alignment with degree differences (SEA) [71] adjusts the TransE approach by incorporating information about each entities degree when building the embeddings. The key insight is that entities that are well-connected appear in more triples, and thus have more robust embeddings containing more information than entities that are infrequently occurring in the graph. The more frequently occurring entities thus form hubs in each embedding space, making the alignment maps learned biased toward these points. Adjusting TransE to better reflect the degree distribution of each entity helps to alleviate these issues and build more robust alignment maps. To prevent entities with similar degree from clustering together in the embedding space, the authors use an adversarial network where a generator builds degree-aware embeddings while two discriminators D_1 and D_2 are used to classify entities with high or normal degree and entities with low degree, respectively. By designing the generator to create high-quality embeddings to fool the discriminators, those embeddings will encode the entities degree in such a way that embeddings of various degrees become linearly inseparable, and thus don't occupy a dense area of the embedding space. The adversarial training is done by training the TransE representations with the discriminators fixed, then alternating by training the discriminators with the embeddings fixed, generating two sets of degree-aware KG embeddings θ_i^1 and θ_j^2. To align the entities from the set of pre-labeled seed alignments $\mathbb{L}$, cycle consistent translation matrices are learned by minimizing

$$\sum_{(e_i,e_j)\in\mathbb{L}} \left\|M^1\theta^1_{e_i} - \theta^2_{e_j}\right\| + \left\|M^2\theta^2_{e_j} - \theta^1_{e_i}\right\|$$

where the cycles

$$\theta^1_{e_i} \rightarrow M^1\theta^1_{e_i} \rightarrow M^2M^1\theta^1_{e_i}$$
$$\theta^2_{e_j} \rightarrow M^2\theta^2_{e_j} \rightarrow M^1M^2\theta^2_{e_j}$$

help to improve generalizability to unlabeled instances. By using all the unlabeled entities in generating the degree aware embeddings, the SEA model is able to leverage both labeled and unlabeled entities in building a robust alignment.

By incorporating a bootstrapping approach, the authors of [72] completely abandon the translational-based model and opt instead for a margin-based model designed to directly leverage information contained in the positive and negative samples sets, which are continuously expanded as the model trains. For triples τ, the objective function

$$\mathcal{O}_e = \sum_{\tau\in\mathbb{T}^+}[f(\tau) = \gamma_1] + \mu_1 \sum_{\tau'\ in\mathbb{T}^-}[\gamma_2 - f(\tau')]$$

where $\mathbb{T}^+, \mathbb{T}^-$ refer to the sets of positive and negative triples, respectively. In building $\mathbb{T}^+$ and $\mathbb{T}^-$, the authors introduce an ϵ-truncated negative sampling strategy, emphasizing that corruption of the head or tail entity should be done so in an intelligent way to maximize the signal the model can learn from. In this negative sampling paradigm, the negative candidates are selected from a neighborhood of $s = \lceil(1-\epsilon)N\rceil$ based on the cosine similarity of their embeddings.

After t steps of training, the set $\mathbb{T}^+$ is updated based on the bootstrapping procedure. As these new bootstrapped instances may contain errors, the authors introduce an editing technique to dampen their effect. Prior to new candidates y and y' with a truth label x being added to $\mathbb{T}^+$, they are evaluated through

$$\Delta^{(t)}_{(x,y,y')} = \pi(y|x;\Theta^{(t)}) - \pi(y'|x;\Theta^{(t)})$$

to determine the highest likelihood of the label, preventing uncertainty from leaking into the bootstrapped training set.

4.2.3 Unsupervised Methods

While there is limited research in semi-supervised methods for knowledge graph alignment, fully unsupervised methods are even less common. In their survey of the literature, [8] claim to observe no research articles on unsupervised methods for knowledge graph alignment. In the time between the release of the survey and this publication, we have found an example of authors exploring unsupervised techniques. In [73] an adversarial training paradigm is used to build links between two graphs. The embeddings of each graph are generated using the DeepWalk technique, taking advantage of the structural properties of each individual graph. These are then mapped using a matrix W fed to a discriminator to differentiate between the source and target space. Given the recency of this publication, its current lack of peer review, and experimentation using only social network domains, we leave the other details to the reader but include its mention to highlight an area of growing interest for researchers.

4.3 Sentence-to-Sentence Alignment Techniques

While many of the techniques applied to word-to-word alignment also apply to sentences, there is also a line of research that focuses only on techniques for sentence alignment. Sentence alignment introduces an additional complexity over word alignment due to variability in word ordering and syntactic and morphological differences between languages that challenge the efficacy of traditional mapping based systems. Applications in this space tend to focus on applications to neural machine translation (NMT), however, we believe that these techniques have applications to other research domains.

4.3.1 Supervised Methods

While there are a limited number of publications exploring supervised sentence to sentence embedding alignment, there are applications of existing word-to-word techniques to this domain. In [74], to benchmark their semi-supervised method the authors re-implement several alignment models for the purpose of mapping sentence embeddings for machine translation. Specifically, they use the linear regression model model from [3], the l_2 regularized model of [47], and the inverted softmax model of [58]. Details of each of these approaches are given in Section 4.1.1. In their evaluation of these techniques, the authors find that these simple linear techniques and their extensions outperform the more advanced models from seq2seq [75], fairseq [76] and LASER [27]. As these more advanced models do not perform explicit alignments, we leave these for the reader to explore.

4.3.2 Semi-Supervised Methods

With an eye toward reducing the amount of parallel data necessary, [74] utilize bidirectional GANs for aligning sentence representations. In addition to defining a piece of the loss function for sentence representation pairs (x, y) that are in the training set, the authors also use all available non-parallel data in their approach. The resulting objective function

$$\mathcal{L}_{real} = \mathbb{E}_{x,y}[\log(D_{real}(x,y)] + \mathbb{E}_x[\log(1 - D_{real}(x, G_X(x)))] + \mathbb{E}_y[\log(1 - D_{real}(y, G_Y(y)))]$$

where a real pair (x, y) in the parallel set is contrasted with fake pairs $(x, G_X(x))$ and $(y, G_Y(y))$ created by the respective generators for the source and target space. To further leverage the data available from the non-parallel sentences, the authors additionally introduce two loss functions. The first is designed to minimize the expected value of mismatched pairs, or negative samples $(x', y') \in X \times Y$ such that

$$\mathcal{L}_{mis} = \mathbb{E}_{x',y'}[\log(1 - D_{real}(x', y'))]$$

The second loss function added includes an additional discriminator to distinguish whether the sentence embedding came from the source or target space, defined as

$$\mathcal{L}_{dom} = \mathbb{E}_x[log(D_{dom}(x, G_X(x)))] + \mathbb{E}_y[log(1 - D_{dom}(y, G_Y(y)))]$$

These three loss functions contribute equally to the overall model loss

$$\mathcal{L} = \mathcal{L}_{real} + \mathcal{L}_{mis} + \mathcal{L}_{dom}$$

To represent the sentences in both the source and target space, the authors use FastText word vectors and simply average each word representation to create a sentence embedding. In their ablation study, the authors additionally experiment with TF-IDF weighting, finding that for corpora with longer sentence length the TF-IDF weighting leads to improved accuracy, while shorter sentences do not exhibit similar gains. The intuition behind these findings is that much of the noise in longer sentences, such as stop words and other semantically irrelevant words used for syntactic purposes, can be down-weighted, thus creating a more semantically meaningful sentence representation.

4.3.3 Unsupervised Methods

Due to the added complexities of mapping full sentence representations, there is limited research on completely unsupervised sentence embedding mappings. For that reason, we restrict our evaluation to the methods demonstrated in [77]. To address the issue of lack of parallel corpora for supervised alignment, the authors of [77] introduce the notion of interlingual semantic representations (ISR) for the few or zero-shot cases. ISR attempts to create an intermediate, low-dimensional space that captures word and sentence semantics that can be fine-tuned to any language or downstream task. To accomplish this representation, the authors utilize an adversarial approach, building a sequence of generators and discriminators to encode language into intermediate representations using only a monolingual corpus. A single generator G makes use of an encoding step *enc* to map an input sentence s to ISR, then applies a decoding step *dec* back to a translated form of that sentence $\hat{s}$. The translation is then fed to the discriminator D for the dual task of determining if the sentence translation is real or synthetic, as well as making a classification for the language of that translation.

Simultaneously, the translation $\hat{s}$ is backpropagated through the generator G for calculation of two losses. The first loss, called the ISR loss, minimizes the distance between the ISR of the forward translation and the backward translation, helping to fine tune the intermediate embedding layer. The second loss, the reconstruction loss, measures how closely the original input sentence vector s and the forward-backward representation after two applications of the generator $G(G(s))$ match. This type of cycle-consistency loss has been shown to aid in cross-domain translation, as in [78], and the authors claim that cycle-consistency helps to preserve semantic information flow from the source to intermediate representations. For fixed embedding inputs, the authors use a BERT-as-a-service model for generation, train their neural architecture using the XNLI dataset, and demonstrate through an ablation study the importance of the reconstruction loss, showing that the cycle-consistency constraint does help in the zero-shot learning task. Fully unsupervised sentence translation remains a challenging yet open research problem, and the pace of research publications in this area continues to increase.

5 Benchmark Datasets

In this section, we document some of the most popular datasets used in the alignment literature.

5.1 Cross-language Word Alignment Benchmarks

As introduced in Section 1.1, the task of Bilingual Lexical Induction aims to evaluate the consistency and ability to learn alignments between embeddings of two distinct languages. There are two themes that datasets in this space may be classified as: those that provide aligned source and target text (akin to datasets used for machine translation) and those that provide pre-trained embeddings of the source and target languages along with seed alignments. In regards to the latter, such published datasets typically select a base embedding algorithm (i.e. FastText or GloVe) and generate pre-trained embeddings on several monolingual corpora, each using the same model hyper-parameters to dictate consistency.

A modern and oft-cited toolkit for BLI datasets is the Facebook MUSE dataset [42]. This dataset contains monolingual embeddings and seed dictionaries for 30 languages, as well as bilingual seed dictionary pairs for 110 languages. For languages coupled with monolingual embeddings, all such embeddings were generated using the FastText algorithm trained over a copy of Wikipedia in the respective language. Each set of embeddings has been generated using the Skip-gram model with the embedding dimension set to 300 with no additional parameter tuning. While this paradigm allows for consistency in comparing the embedding spaces, it could be the case that the embeddings may perform better on certain languages, confounding the evaluation of the structural similarities between embedding spaces. For the ground-truth seed dictionary pairs, aligned seeds are provided in sets of 5,000 training pairs and 1,500 testing pairs. This benchmark was further criticized in [56] due to the large presence of proper nouns, which are considered to be only referential and contain limited lexical meaning, in Wikipedia articles, potentially over-inflating the performance metrics of systems tested on this benchmark.

The most popular alternative to the MUSE dataset is that compiled in [47], typically referred to as DINU. This dataset was compiled automatically from Europarl [44], a compilation of European parliament proceedings in 21 languages, typically packaged for evaluation systems into four primary languages: English, Finnish, German and Spanish [56]. Training translation pairs are split by word frequency into five buckets: 1-5K, 5-20K, 20-50K, 50-100K, 100K-200K. Within each bucket, 1,500 testing pairs are selected, as well as non-overlapping sets of training pairs in sizes of 1K, 5K, 10K and 20K.

5.2 Knowledge Graph Entity Alignment Benchmarks

Benchmarking experiments on knowledge graph entity alignment typically required two or more source knowledge graphs with a degree of known overlapping entities. One such way of generating these dataset is to use knowledge graphs describing the same set of triples in multiple languages. The WK31 datasets are an example of this paradigm, containing knowledge graphs focusing on the DBPedia person domain across English, French and German. The WK31 dataset comes in two widely utilized variants based on the number of aligned nodes, namely WK31-15k and WK31-120k. These datasets are additionally evaluated based on the number of *inter-lingual links* (ILL), where the ILL identify the same entity across two pairs of languages. The ILLs are typically used for the testing set for many evaluations and account for a small amount of the overall dataset, representing the challenge of generating a trustworthy seed dataset for supervised methods and motivating the search for many semi- or unsupervised solutions. Experiments on this dataset are reported in [60, 66, 71]. As noted in [8], there is a great deal of variance in the number of triples, entities, relations and seeds described in several publications using the WK31 dataset. Here, we provide a summary of this dataset in Table 2, based on the metrics reported in [66].

Table 2: Seed Alignments of WK31 Datasets

Dataset	Aligned Entities	Aligned Relations
WK31-15k-En-De	2,070	445
WK31-15k-En-Fr	3,116	598
WK31-120k-En-De	9,680	772
WK31-120k-En-Fr	42,378	1,127

Table 3: Overlapping Alignments of DFB Datasets

Dataset	Relations	Entities	OT	Seeds
DFB-1	1,345	14,951	0.5	5,000
DFB-1	1,345	14,951	0.5	500
DFB-1	1,345	14,951	0.1	500

In addition to cross-lingual knowledge graph datasets, several studies have split larger graphs into smaller components, each of which has linking entities that may be used as seeds to re-unify the graph. The advantage of these datasets is that the target graph is entirely known and well defined, allowing for the number of seed entities to be scaled up and down and thus helping to experimentally validate the necessary amount of 'overlap' needed to effectively perform the alignment task. One such group of datasets named DFB-1, DFB-2 and DFB-3, constructed by [70], are constructed by randomly sampling triples from Freebase while specifying an overlapping threshold (OT). A summary of the DFB datasets is provided in 3.

The approach of splitting a larger graph into many subgraphs is also applied to cross-lingual knowledge graphs as well, as is the case with the DBP15k dataset [63]. This dataset uses DBPedia entities in Chinese, English, French and Japanese, creating four sets of ILLs, each containing 15,000 seed pairs, and is experimented with in [63, 69, 64, 65, 72]. A detailed description of the dataset is provided in 4.

6 Summary and Open Questions

To summarize, we conducted a survey of the literature on the task of aligning diverse embedding spaces output by various neural networks for creating low-dimensional representations of data. These methods have been thoroughly studied in the spaces of both natural language and graphs. The majority of these methods aim to learn alignment models between spaces of the same underlying data type, i.e. words-to-words or graphs-to-graphs, typically with the alignment meant to bridge the gap between languages. We find that there is significantly less research in bridging the gap between unlike embedding spaces, for example sentence embeddings and knowledge graphs, which we believe will provide significant gains in the fields of information extraction and data integration. By identifying this gap and outlining existing methodologies we hope this survey provides an entry point for other researchers with a shared goal of aligning embedding spaces of diverse data types.

Table 4: Metrics of DFP Datasets

Dataset	Language	Entities	Relations	Triples	Seeds
DBP15k-ZH-EN	Chinese	66,469	2,830	153,929	15,000
DBP15k-ZH-EN	English	98,125	2,317	237,674	15,000
DBP15k-FR-EN	French	66,858	1,379	192,191	15,000
DBP15k-FR-EN	English	105,889	2,209	278,590	15,000
DBP15k-JA-EN	Japanese	65,744	2,043	164,373	15,000
DBP15k-JA-EN	English	95,680	2,096	233,319	15,000

References

[1] Alexander Ratner, Stephen H. Bach, Henry R. Ehrenberg, Jason Alan Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *CoRR*, abs/1711.10160, 2017.

[2] Ann Irvine and Chris Callison-Burch. A comprehensive analysis of bilingual lexicon induction. *Computational Linguistics*, 43(2):273–310, jun 2017.

[3] Tomas Mikolov, Quoc V. Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *CoRR*, abs/1309.4168, 2013.

[4] Sebastian Ruder. A survey of cross-lingual embedding models. *CoRR*, abs/1706.04902, 2017.

[5] Namyoun Choi, Il-Yeol Song, and Hyoil Han. A survey on ontology mapping. *SIGMOD Rec.*, 35(3):34–41, September 2006.

[6] Xin Luna Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 601–610, 2014. Evgeniy Gabrilovich Wilko Horn Ni Lao Kevin Murphy Thomas Strohmann Shaohua Sun Wei Zhang Geremy Heitz.

[7] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33, 2016.

[8] Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. A benchmarking study of embedding-based entity alignment for knowledge graphs, 2020.

[9] Raphael Hoffmann, Luke S. Zettlemoyer, and Daniel S. Weld. Extreme extraction: Only one hour per relation. *CoRR*, abs/1506.06418, 2015.

[10] Marjorie Freedman, Lance Ramshaw, Elizabeth Boschee, Ryan Gabbard, Gary Kratkiewicz, Nicolas Ward, and Ralph Weischedel. Extreme extraction - machine reading in a week. pages 1437–1446, 01 2011.

[11] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data.

[12] Alisa Smirnova and Philippe Cudré-Mauroux. Relation extraction using distant supervision: A survey. *ACM Computing Surveys (CSUR)*, 51(5):1–35, 2019;2018;.

[13] Benjamin Roth, Tassilo Barth, Michael Wiegand, and Dietrich Klakow. A survey of noise reduction methods for distant supervision. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, AKBC '13, pages 73–78, New York, NY, USA, 2013. ACM.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013.

[15] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[16] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[19] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *CoRR*, abs/1802.05365, 2018.

[20] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[21] Xunjie Zhu and Gerard de Melo. Sentence analogies: Exploring linguistic relationships and regularities in sentence embeddings, 2020.

[22] Nada Almarwani, Hanan Aldarmaki, and Mona Diab. Efficient sentence embedding using discrete cosine transform. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3663–3669, 2019.

[23] Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-thought vectors, 2015.

[24] Lajanugen Logeswaran and Honglak Lee. An efficient framework for learning sentence representations. In *International Conference on Learning Representations*, 2018.

[25] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[26] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[27] Mikel Artetxe and Holger Schwenk. Massively multilingual sentence embeddings for zero-shot cross-lingual transfer and beyond, 2018.

[28] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, PP:1–1, 09 2017.

[29] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013.

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

[31] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, page 2181–2187. AAAI Press, 2015.

[32] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 809–816, Madison, WI, USA, 2011. Omnipress.

[33] Bishan Yang, Wen tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases, 2014.

[34] Théo Trouillon, Christopher R Dance, Éric Gaussier, Johannes Welbl, Sebastian Riedel, and Guillaume Bouchard. Knowledge graph completion via complex tensor factorization. *Journal of Machine Learning Research (JMLR)*, 18(130):1–38, 2017.

[35] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings, 2017.

[36] William L. Hamilton, Rex Ying, and Jure Leskovec. Representation learning on graphs: Methods and applications, 2017.

[37] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases, 2015.

[38] Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space, 2015.

[39] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2016.

[40] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 451–462, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[41] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings, 2018.

[42] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *CoRR*, abs/1710.04087, 2017.

[43] David Alvarez-Melis and Tommi S. Jaakkola. Gromov-wasserstein alignment of word embedding spaces. *CoRR*, abs/1809.00013, 2018.

[44] Philipp Koehn. Europarl: A parallel corpus for statistical machine translation. 5, 11 2004.

[45] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[46] Natalya Noy. Semantic integration: A survey of ontology-based approaches. *SIGMOD Record*, 33:65–70, 12 2004.

[47] Georgiana Dinu, Angeliki Lazaridou, and Marco Baroni. Improving zero-shot learning by mitigating the hubness problem, 2014.

[48] Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, Denver, Colorado, 2015. Association for Computational Linguistics.

[49] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning principled bilingual mappings of word embeddings while preserving monolingual invariance. pages 2289–2294, 01 2016.

[50] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Generalizing and improving bilingual word embedding mappings with a multi-step framework of linear transformations. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 5012–5019, February 2018.

[51] Angeliki Lazaridou, Georgiana Dinu, and Marco Baroni. Hubness and pollution: Delving into cross-space mapping for zero-shot learning. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 270–280, Beijing, China, "jul" 2015. Association for Computational Linguistics.

[52] ChengYue Gong, Di He, Xu Tan, Tao Qin, Liwei Wang, and Tie-Yan Liu. FRAGE: frequency-agnostic word representation. *CoRR*, abs/1809.06858, 2018.

[53] Jiaqi Mu, Suma Bhat, and Pramod Viswanath. All-but-the-top: Simple and effective postprocessing for word representations. 02 2017.

[54] Armand Joulin, Piotr Bojanowski, Tomas Mikolov, Herṽe Jégou, and Edouard Grave. Loss in translation: Learning bilingual word mapping with a retrieval criterion. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

[55] Goran Glavas, Robert Litschko, Sebastian Ruder, and Ivan Vulic. How to (properly) evaluate cross-lingual word embeddings: On strong baselines, comparative analyses, and some misconceptions. *CoRR*, abs/1902.00508, 2019.

[56] Yova Kementchedjhieva, Mareike Hartmann, and Anders Søgaard. Lost in evaluation: Misleading benchmarks for bilingual dictionary induction, 2019.

[57] Ashwinkumar Ganesan, Frank Ferraro, and Tim Oates. Locality preserving loss to align vector spaces, 2020.

[58] Samuel L. Smith, David H. P. Turban, Steven Hamblin, and Nils Y. Hammerla. Offline bilingual word vectors, orthogonal transformations and the inverted softmax, 2017.

[59] Barun Patra, Joel Ruben Antony Moniz, Sarthak Garg, Matthew R. Gormley, and Graham Neubig. Bilingual lexicon induction with semi-supervision in non-isometric embedding spaces. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 184–193, Florence, Italy, 2019. Association for Computational Linguistics.

[60] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multi-lingual knowledge graph embeddings for cross-lingual knowledge alignment. *CoRR*, abs/1611.03954, 2016.

[61] Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. One-shot relational learning for knowledge graphs, 2018.

[62] Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V. Chawla. Few-shot knowledge graph completion, 2019.

[63] Zequn Sun, Wei Hu, and Chengkai Li. Cross-lingual entity alignment via joint attribute-preserving embedding. *CoRR*, abs/1708.05045, 2017.

[64] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. pages 349–357, 01 2018.

[65] Xiaofei Shi and Yanghua Xiao. Modeling multi-mapping relations for precise cross-lingual entity alignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 813–822, Hong Kong, China, November 2019. Association for Computational Linguistics.

[66] Shengnan Li, Xin Li, Rui Ye, Mingzhong Wang, Haiping Su, and Yingzi Ou. Non-translational alignment for multi-relational networks. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 4180–4186. International Joint Conferences on Artificial Intelligence Organization, 7 2018.

[67] Xin Li, Huiting Hong, Lin Liu, and William K. Cheung. A structural representation learning for multi-relational networks, 2018.

[68] Liwei Cai and William Yang Wang. KBGAN: Adversarial learning for knowledge graph embeddings. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1470–1480, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[69] Xixun Lin, Hong Yang, Jia Wu, Chuan Zhou, and Bin Wang. Guiding cross-lingual entity alignment via adversarial knowledge embedding. 11 2019.

[70] Hao Zhu, Ruobing Xie, Zhiyuan Liu, and Maosong Sun. Iterative entity alignment via joint knowledge embeddings. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 4258–4264. AAAI Press, 2017.

[71] Shichao Pei, Lu Yu, Robert Hoehndorf, and Xiangliang Zhang. Semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *The World Wide Web Conference*, WWW '19, page 3130–3136, New York, NY, USA, 2019. Association for Computing Machinery.

[72] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. Bootstrapping entity alignment with knowledge graph embedding. pages 4396–4402, 07 2018.

[73] Chaoqi Chen, Weiping Xie, Tingyang Xu, Yu Rong, Wenbing Huang, Xinghao Ding, Yue Huang, and Junzhou Huang. Unsupervised adversarial graph alignment with graph embedding, 07 2019.

[74] Zuohui Fu, Yikun Xian, Shijie Geng, Yingqiang Ge, Yuting Wang, Xin Dong, Guang Wang, and Gerard de Melo. Absent: Cross-lingual sentence representation mapping with bidirectional gans, 2020.

[75] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014.

[76] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.

[77] Channy Hong, Jaeyeon Lee, and Jungkwon Lee. Unsupervised interlingual semantic representations from sentence embeddings for zero-shot cross-lingual transfer. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:7944–7951, 04 2020.

[78] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.