THESIS: KNOWLEDGE BASE COMPLETION WITH EMBEDDINGS OF ENTITIES AND RELATION OPERATORS

A Senior Honors Thesis

Presented to the Computer Science Department

of The Johns Hopkins University

in Partial Fulfillment of the Requirements for the Degree of

Bachelor of Science

by

Corbin L. Rosset

May 2017

THESIS: KNOWLEDGE BASE COMPLETION WITH EMBEDDINGS OF ENTITIES AND RELATION OPERATORS

Corbin L. Rosset

The Johns Hopkins University 2017

Knowledge bases are an effective tool for structuring and accessing large amounts of multi-relational facts and are instrumental in many large-scale information processing systems. However, inferring missing facts becomes a crucial challenge since knowledge bases are often woefully incomplete, especially in broader domains where information is sparse. We consider the task of learning low dimensional embeddings for Knowledge Base Completion and make the following contributions: 1) a novel embedding model, ModelE-X, that uses few parameters yet outperforms many state-of-the-art, more complex algorithms, 2) the realization that the often-unreported metric of relation ranking yields valuable insights into algorithms' behavior and 3) we observe the macro-average of ranking metrics across relations, which treats all relations equally despite their distribution, is a better indicator of generalizability, yet it is also unreported in the literature. We will also discuss algorithms that leverage textual data and long-range path queries.

BIOGRAPHICAL SKETCH

The author, Corbin Rosset, is graduating from the Johns Hopkins University Computer Science Department with a Masters in Science and Engineering and a Bachelors in Science. He is interested in machine learning techniques applied to understanding natural language, and looks forward to researching methods for involving synthesis and inference of knowledge for open-domain question answering systems, and is particularly interested in improving conversational exchanges with “intelligent” agents. This work focuses on the foundations of knowledge representation learning, skills which he hopes will help in his future endeavors. He has been inspired by the faculty, students, and lectures in the Center for Language and Speech Processing and the Computer Science Department as a whole at Johns Hopkins.

To my sister, who is entering college soon

ACKNOWLEDGEMENTS

The author would like to thank his advisor, Dr. Raman Arora for his dedication in teaching and mentoring. Drs. Mark Dredze and Benjamin Van Durme also provided valuable advice and insight. The author would like to thank all the aforementioned for their time and patience.

TABLE OF CONTENTS

Biographical Sketch iii
Dedication iv
Acknowledgements v
Table of Contents vi
List of Tables viii
List of Figures ix

1 Introduction 1
1.1 Knowledge Base Completion 2
1.2 Importance of Knowledge Bases and their completion 5
1.2.1 Factoid question answering 5
1.2.2 Relation extraction 5
1.3 Learning Knowledge Graph Parameters 6
1.3.1 MAP Estimation of Parameters 6
1.3.2 Approximation to MLE/MAP 7
1.3.3 Pairwise Margin Ranking and Negative Sampling 8
1.4 Evaluation Metrics 9

2 Related Work 11
2.1 Latent Feature Models 11
2.1.1 Matrix Factorization 12
2.1.2 Entity-specific Embedding Models 12
2.2 Compositional "Path" Embeddings 14
2.3 Leveraging Text and Relation Extraction Techniques 15
2.3.1 Separate Models for KB and Text embeddings 16
2.3.2 Entity and Text Cooccurance 18
2.3.3 Entity and Text Co-occurance: TEKE 20
2.3.4 CNNs for Textual Relation Extraction 20
2.3.5 Aligning Entity and Word Vectors using Descriptions 22
2.3.6 Fully Joint Learning for Text, Entities, and Relations 23
2.4 Hybrid Techniques 25
2.4.1 Graph Features: NodeFeat and LinkFeat 25
2.4.2 Miscellaneous 26

3 Contribution and Novel Work 27
3.1 A New Embedding Model: ModelE-X 27
3.2 Role of Textual Representations in Latent Factor Models 28
3.3 New Evaluation Approaches 29
3.3.1 Micro Averaging 29
3.3.2 Relation Ranking 30

4 Experimental Results and Discussion **31**
4.1 Data and Experiments 31
4.1.1 Knowledge Base and Text Datasets 31
4.2 Results and Discussion 34

5 A Common Mistake in Practice **39**

LIST OF TABLES

4.1 Some statistics of the major knowledge graph datasets from [32] . . . 31

4.2 We compare our models against several classical and state-of-the-art algorithms using the traditional ”Micro” averages for evaluation metrics. Our ModelE-X is competitive and in some cases surpasses even state of the art algorithms such as STransE and LinkFeat. Each value reports ”raw” / ”filtered” metrics on FB15k test set. Parenthesis indicate the embedding dimension, else it is 100 for all models we implement, which are marked with an asterisk. Values reported in entity columns represent the average of left and right entity ranking metrics; NA means “not applicable”, and “-” means unreported. 33

4.3 Our ModelE-X truly excels when all relations are treated equally, that is, the macro average across relations is taken (the average of relation-specific averages for a particular metric). ModelE-XL is rather difficult to train and regularize since it has quadratically-many parameters in d . 33

4.4 Most prevalent relations in the FB15k test set, which has 16291 triples in total. Clearly, a handful of relationships, all related to either awards or films, comprise a disproportionate amount of the test data. Fortunately, the distribution of relations in the training set is roughly equivalent. 34

4.5 Here we compare two very good models with the same parameters, except the loss function of the model marked with asterisk did not include contrasting positive and negative relations, yet here we evaluate it on relation ranking in the test data to show its weakness in both micro and macro metrics. Contrary to many beliefs, models trained without relation ranking will not automatically learn good relation representations, which in our opinion is a vulnerability. However, with the relation ranking loss, entity metrics do a suffer a bit, showing the trade-offs of these loss functions 35

LIST OF FIGURES

4.1 The distribution of the roughly 16,000 test triples for FB15k is highly non-uniform. Left is a histogram of the test set relations; there are 1,345 relations in FB15k. Right is a histogram of the test set entities, of which there are 14,951 32

4.2 A T-SNE visualization of some of the vector representations learned by a Bilinear model for some of the most common entities in FB15k. . . . 37

4.3 A T-SNE visualization of the head relation vectors learned by ModelE. 38

CHAPTER 1

INTRODUCTION

Knowledge Bases (KBs) represent structured information in a way that is convenient for inference and computation. Internet companies leverage KBs to improve search results on factoid queries, such as Google's Knowledge Vault [7], Microsoft's Satori [32, 24] and are indispensable for tasks like question answering. they help disambiguate entities, and provide references to related sources of information [8, 62, 61, 58].

An important goal in the information retrieval and natural language processing communities is to extract structured facts from unstructured information, which can improve understanding of intent and context surrounding an unstructured query in a search engine, for instance. Modeling facts and events is particularly relevant in modern challenges such as open domain question answering, conversational ”infobots”, and reading comprehension engines, which are required to perform complicated, multi-hop inference over a large set of relevant facts in an evolving context. It is critical to encode this knowledge in compact yet expressive representations, thus, fast and effective knowledge representation algorithms are fundamental for the next generation of artificial general intelligence agents [41, 18, 53].

A knowledge base (KB) over schema of E entities from a set $\mathcal{E}$ and R relations from a set $\mathcal{R}$ is a set of triples $\mathcal{T} = \{(h, r, t)\}$ where $h, t \in \mathcal{E}, r \in \mathcal{R}$, which can be interpreted as a knowledge graph (KG) with edge labels from $\mathcal{R}$ and node labels from $\mathcal{E}$. An embedding for h is denoted $\boldsymbol{e_h} \in \mathbb{R}^d$, which are used interchangeably. Knowledge bases over any useful domain are incomplete, as they are often constructed by hand or semi-automatically [42, 51]. Knowledge Base completion is the well-studied task of determining which triples ought to be in the KB, which is usually treated as a ranking problem; we discuss it in more depth in 1.1

Knowledge base completion is one of many tasks under the umbrella of statistical relation learning, which is loosely defined as representation learning to capture patterns in the relationships that exist between elements of a graph-structured dataset. Other tasks commonly investigated are predicting properties of nodes in the dataset, and clustering nodes. Algorithms for KBC differ depending on what data the KB is constructed from and how. Some algorithms learn based only on the single links expressed in $\mathcal{T}$, some consider paths and deeper structures involving chains of links, and many leverage large textual resources with NLP pipelines including entity linkers and language models.

Recent areas of research have evolved from the overlapping desires of completing existing knowledge graphs and extracting relations or facts from new information (in either batch or online settings), with the ultimate goal being an accurate, never-ending, and self-maintaining knowledge base system[19, 4, 36, 38]. Many researchers have turned to neural approaches to find semantic spaces for knowledge graph entities and relations, sometimes extending this space to include vector based language models as well. This is largely due to the success of embedding models to capture generalizable and abstract concepts from large amounts of noisy data that lie on complex underlying manifolds. There remains much work to be done, however, in designing models to align KB representations with textual mentions of facts and what it means to train such models on KG and textual triples jointly.

1.1 Knowledge Base Completion

As distinguished from algorithms to construct a KB (Construction or Population) from raw text or small seed KBs[4, 38, 37], Knowledge Base Completion seeks to infer missing links in a KBs such as Freebase [1], YAGO [43], and DBPedia [22], which, though enormous, are lacking in coverage [51]. The problem of Construction faces a slightly different set of challenges than Completion, namely, how to handle conflicting information, possibly from multiple sources with varying degrees of credibility [35]. In the setting of Completion, we assume that a KB has already been partially and accurately constructed. We do not assume it is subject to real-time updates, though in practice this is the case since current events change the state of the world frequently and unpredictably. Also, scalability is paramount, as real world KBs must accommodate massive amounts of facts, possibly the entire internet's worth. Some good reviews of KBC are [32, 50]

In many cases, the missing facts can be entailed by the facts that are recorded, and more often, they can be inferred by a probabilistic model conditioned on the existing facts. Here we primarily discuss embeddings or latent feature models for KBC, which learn dense parameterizations of entities and relationship operators and are adept at modeling global patterns in large, noisy KGs, a topic which is reviewed well in [32]. Modern KB schemas define potentially millions of entities and billions of facts; representing these items symbolically makes it challenging to meaningfully compare and operate on entities. Having a fixed d-dimensional embedding of each entity 1) simplifies storage requirements, 2) allows for natural comparisons between entities and 3) a mathematical structure for interpreting relationships as operators over entities, which allows for capturing long-range structural information in the KG [3]. This framework also integrates well with existing techniques for neural language modeling techniques for textual data, where word embeddings can be co-trained with entity embeddings as in [16].

There are also tensor and matrix factorization techniques; the algorithms and limitations of which we describe briefly in 2.1.1 and can be found in more detail in [31, 39, 17]. There is a wide variety of research in using textual resources to refine representations of entities and relations, for instance, algorithms using descriptions of entities, algorithms similar to relation extraction, and algorithms that co-train language models and entity-relation models. We discuss these in depth in 2.3. There is another field of research devoted to deeper understanding of the structure in the existing knowledge graph, which stemmed from compositional extensions of earlier embeddings models; see 2.2.

The framework for nearly all modern automatic KBC algorithms that learn embeddings of entities and relations use the same machinery: in addition to learning said embeddings, each model defines a scoring function $f(h, r, t)$ of a triple over those embeddings, a (pairwise) margin ranking objective[1], and a method of sampling "false" triples under the somewhat unrealistic "closed world" assumption that any triple not in the overall KB is untrue[2]. The following loss function contrasts positive and negative triples (regularization term omitted)

$$\mathcal{L}(ALG) = \sum_{p \in \mathcal{T}, n \notin \mathcal{T}} [\gamma + f_{ALG}(p) - f_{ALG}(n)]_+$$

and is minimized by mini-batch gradient descent over (p, n) pairs where $p = (h, r, t)$ is a sampled positive triple, $n \in \{(h', r, t), (h, r', t), (h, r, t')\}$ is a sampled negative triple with exactly one slot corrupted s.t. $n \notin \mathcal{T}$, and $[\cdot]_+ = max(0, \cdot)$ Usually, whenever p is sampled, the two corrupted-entity versions of it are added to the batch. We extend this by adding the relation-corrupted triple and perform relation ranking during training3.3.2.

[1] [45, 44] instead maximize the conditional log likelihood, but the pairwise ranking loss is more scalable.

[2] However this assumption can be flagrantly violated. A relaxation is the Locally closed world assumption, which asserts that a local subgraph is complete (that is, for a subject-predicate pair, all triples of the form $(e_l, r, \dot{})$ not in the KB are assumed to be false.

1.2 Importance of Knowledge Bases and their completion

1.2.1 Factoid question answering

Not only are KBs useful as “lookup” tools for questions like “Where was Abraham Lincoln Born”, but the facts therein serve as the fundamental units of inference to harder-to-answer queries, such as “How many pages of text can DNA encode?”. If an answer to a factoid question cannot immediately be referenced in a database or knowledge base, such as the previous query, information retrieval techniques are used as a backup: a search engine will try to find sentences across the Internet’s documents that likely contain the answer. However, information is spread across the Internet such that no single sentence likely answers one of these harder “tail queries”. The hope is that storing structured information in the form of a knowledge graph will make inferring an answer a more reliable option if an answer isn’t readily available in on particular sentence on the Internet. For instance, it may be stored in the KB that human DNA holds about 725 MB of data, and it may separately be stored that a page of digitized text contains about 3 KB. If facts are stored in a vector space, then modern neural algorithms for inference with sophisticated attention mechanisms could identify these two numbers as relevant, and will then theoretically be able to do arithmetic operations on these two numbers.

1.2.2 Relation extraction

Relation Extraction, a close cousin of AKBC, is the task of classifying which relationship from a defined schema such as $\mathcal{R}$ applies to a textual instance in which entities have already been recognized, some modern works include [40, 63, 16, 12, 59]. Typically the text between two recognized entities from $\mathcal{E}$ is subject to classification, but broader context may be necessary. It can be treated in a supervised fashion if manual annotations for the schema exist, or as a distant supervision problem if a sufficiently populated knowledge base of the same schema exists [28]. Since manual annotation is expensive and the most relevant relation extraction must take place on vast quantities of data and large schemas, the field has moved toward using the schemas in existing structured databases or knowledge bases as labels and focus on aligning sentences with records in these sources. One drawback, however, is that the pair of entities may be associated with many relations in the knowledge base, and it may not be clear which one is expressed by a particular textual instance (leading to a weak supervision problem). Another drawback is that more than one relation may be expressed in a text segment bookended by entities, a challenge explored in [61].

1.3 Learning Knowledge Graph Parameters

1.3.1 MAP Estimation of Parameters

Whether a triple should exist in a KB can be represented as a Bernoulli random variable, the parameters for which can be found by MAP estimation.

$$\max_{\Theta} \sum_{x \in \mathcal{T} \cup \mathcal{T}'} logBer(y_x | \sigma(f(x; \Theta))) + logp(\Theta | \lambda_1) \tag{1.1}$$

where each triple x (positive or negative) is labeled y_x and scored by f parameterized by Θ. The prior's strength is given by λ_1. This can be reformulated as a regularized loss minimization objective where $\mathcal{L}$ can be defined as $\mathcal{L} = -logBer(y|f(t; \Theta))$ or as the squared error loss $||y - f(t; \Theta)||$:

$$\max_{\Theta} \sum_{t \in \mathcal{S} \cup \mathcal{S}'} \mathcal{L}(\sigma(f(t;\Theta)), y_t) + \lambda_2 reg(\Theta) \tag{1.2}$$

1.3.2 Approximation to MLE/MAP

As the need for scalability becomes more preeminent, the techniques for learning the embeddings and other parameters have relaxed to allow for easier parallelization. We can eliminate the summation over all negative triples to speed up learning while still expressing our desire to score positive triples higher than candidate negative triples. That is, negative triples need only to be ”more incorrect” than the positive ones [32]. Hence we can sample some negative triples to contrast with the positive ones, which also solves another problem with the MAP estimation approach:

The primary impediment of conditional probability models is computing the normalizing denominator term, which usually involves summing over entire vocabularies or dictionaries. An approximation can be found using negative sampling [21]. Toutanova et al [45, 45, 48] have successfully used this technique to approximate the log likelihood of the conditional probability of a missing entity given the other entity and the relationship $p(t|r,h)$:

$$p(t|r,h) = \frac{e^{f(h,r,t)}}{\sum_{h' \in \mathcal{T}'} f(h', r, e_r)} \tag{1.3}$$

This is analogously defined for $p(h|r,t)$ and $p(r|h,t)$. The approximation comes from sampling negative samples (anywhere from 1 to about 200) for the denominator with type constraints optionally enforced.

1.3.3 Pairwise Margin Ranking and Negative Sampling

We can further relax the sampling to contrast each positive fact with a single sampled negative fact, with the constraint that the negative fact is closely related to the positive one, but with one field "corrupted". We can then contrast the positive and negative sample using the large margin loss, which is faster to compute than the approximate likelihood above. Of course, we lose the advantage of working with probabilities.

Specifically, we subsample a training knowledge graph $\mathcal{S}$ from the real knowledge graph, and put the held out triples in a test set. For every positive triple $t \in (e_l, r, e_r)$ in $\mathcal{S}$, create N negative samples by randomly corrupting one of the fields in t such that the corruption does not yield any positive sample. The goal is to learn parameters (the latent representations of constituent entities and relationships) that score positive triples higher than negative ones, for any scoring function f, like TransE, for instance. The following loss function expresses these desires by maximizing a large margin ranking objective.

$$\mathcal{L} = \sum_{t \in \mathcal{S}} \sum_{t' \in \mathcal{S}'} \big[max\big(0, \gamma + f(t) - f(t')\big)\big] \tag{1.4}$$

In 1.4, t is a positive triple (e_l, r, e_r) and t' is a negative triple in one of three forms: (e_l', r, e_r), (e_l, r', e_r), or (e_l, r, e_r') for all (or some subset of) e_l', r', or e_r' which make each respective triple "false" or not in the known KB. Learning is achieved by mini-batch stochastic gradient descent.

Some scoring functions like TransE take on an energy interpretation: true triplets favor lower energy values. TransE and many other models enforce a unit $\ell - 2$ norm constraint on the entity vectors.

Typically, all embeddings are initialized randomly subject to the constraints, and after each iteration all entities are typically renormalized again. For a mini-batch of randomly selected positive triplets, N negative samples are generated, and for each positive-negative pair, an SGD step is taken. [3]

Typically the margin is selected from $\{0.5, 1, 2, 10\}$ and the learning rate for SGD is tuned over $\{0.001, 0.01, 0.1\}$. Distance functions are either $\ell - 1$ or $\ell - 2$ norms. This method of margin ranking can take a long time to converge.

1.4 Evaluation Metrics

Evaluating a KBC system is a nuanced task that depends on the environment the knowledge graph was constructed from and is expected to operate in, and what assumptions/biases exist in those pipelines. It is further complicated in systems that leverage text, as the quality of the AKBC system now depends on the distribution generating the observed text, as well as any errors and noise propagated by textual processing and representation learning.

We evaluate the model on the Link Prediction task, which is to predict, say, the best h given r and t to yield a triple that is most likely to belong the KB[4]. A list of scores is generated by applying $f_{ALG}(h', r, t)\ \forall h' \in \mathcal{E}$, which, upon sorting, hopefully yields a highly-ranked h that makes the triple true. The loss attempts to concentrate mass on observed triples, but it provides only a rough approximation to the ideal list (specifically, it might not score obviously incorrect triples any lower than partially incorrect triples, as long as the positive triple is scored above both). We report several ranking

[3] for TransE, N = 1, but for other methods it is larger

[4] Simiarly we also predict the best r or best t.

metrics: Mean Rank, Median Rank, Mean Reciprocal Rank (MRR), and Hits@10 [15]. In practice not all $h' \in \mathcal{E}$ should be included in the list, only those for which it is known $(h', r, t) \notin \mathcal{T}$ so that the model isn't penalized for ranking one correct answer over another; this is known as the "filtered" (as opposed to "raw") metric.

CHAPTER 2

RELATED WORK

The following sections discuss primarily discuss models that learn latent features for units of a knowledge graph, such as entities, relations, and paths. The representations are learned using statistical techniques from large amounts of data. However, the KBC community is vast and we will briefly mention some other notable techniques in 2.4.2.

2.1 Latent Feature Models

Latent features models for KGs operate under the assumption that the score of a triple depends only on learned latent features of entities and relationships. Patterns in the local and global structure of the KG are intended to be captured by these features. There are wide variety of these models each of which focus on specific kinds of patterns; it is often the case that models can be combined to complement their strengths. The primary drawback of all these methods is interpretability of the learned representations, and the difficulty of enforcing logical and type constraints in a way that can be expressed mathematically and computed efficiently [2].

These algorithms are motivated by the recent success of user-item matrix factorization techniques for single-relational data to find embeddings of users, items, or both that lead to effective recommendation tools. However, unlike user-item matrices, the challenges of multi-relational data are twofold. Firstly, relationships are directed and are often subject to logical or type constraints that may not be captured by traditional recommender systems. Secondly, two distinct entities can exhibit multiple relationships between them (think of the many relationships that can exist between, say, the president of the U.S. and its government).

2.1.1 Matrix Factorization

There are a number of models that approach KBC as a tensor or matrix factorization problem [33, 34, 47, 31] which inspired others to define entity and relation specific embeddings [3, 42, 11].

The first approaches to tackling multi-relational data in the context of knowledge bases was to use tensor factorization methods, where a ”third” dimension was added to account for multiple relationships between entities. It followed naturally that downstream tasks operated on the latent attributes of the constituents of the tensor.

The drawbacks are of course scalability, as these tensors may have enormous dimension.

2.1.2 Entity-specific Embedding Models

Model E

ModelE defines two vectors in $\mathbb{R}^d$ for each relation to allow only certain entities in the head and tail position of a triple. That is, for the score of a triple to be high, both the head and tail entities must align with their respective relation embeddings components $\boldsymbol{r_h}$ and $\boldsymbol{r_t}$ [39]:

$$f_{ModelE}(h, r, t) = \boldsymbol{e_h}^T \boldsymbol{r_h} + \boldsymbol{e_t}^T \boldsymbol{r_t}$$

It has $Ed + 2Rd$ parameters, and aims to give high scores to true triples.

The TransE Family

Inspired by the semantically meaningful translations of word embeddings, the TransE model on a positive triple (h, r, t) learns embeddings such that $\boldsymbol{e_h} + \boldsymbol{r} \approx \boldsymbol{e_t}$ [2]

$$f_{TransE}(h, r, t) = \|\boldsymbol{e_h} + \boldsymbol{r} - \boldsymbol{e_t}\|$$

It has $Ed + R$ parameters and seeks a low score for positive triples.

There are several spinoffs of the TransE model that try to improve on its weaknesses in modeling relations that aren't "one-to-one". TransH is a model that learns a hyperplane for each relation and models translations within it [48], as well as TranR: [26] and TransG [54].

Bilinear Family

For Bilinear (DistMult) and BilinearDiag [60], each relation r is parameterized by $W_r \in \mathbb{R}^{d\times d}$ (which is constrained to be diagonal for BilinearDiag).

$$f_{Bilinear}(h, r, t) = \boldsymbol{e_h}^T W_r \boldsymbol{e_t}$$

Bilinear has $Ed + Rd^2$ parameters, which can be quite slow and prone to overfitting.

It should be noted that Google released trained entity embeddings, which some authors use to initialize their own embeddings. For instance Yang *et al* learn a (nonlinear) mapping from the 1000-dim space of the released Word2Vec entity vectors to the d-dimensional KB entity space in their work [1] [60]. Initialization of a vanilla Dist-Mult model this way provides as almost as much improvement as the compositional path training of a Dist-Mult model, showing that initialization of entity vectors merits careful attention, especially when operating over smaller datasets.

[1] Pre-trained entity vectors with Freebase naming: `https://code.google.com/archive/p/word2vec/`

Structured Embeddings

Structured Embedding [3] finds k dimensional representations of entities and two $k \times k$ matrices for the relationship, L_1 and L_2, such that $f(L_1 e_l, L_2 e_r)$ is small for positive triples for some distance metric f like $\ell - 2$ norm. Teh two different projection matrices account for asymmetry in relationships. If L_2 is fixed to be identity, then SE reduces to TransE. Bordes et al demonstrate that TransE is superior to SE for KBC on FB15k and FB1M datasets. TransE does not perform well on relationships in which 3-way dependencies between $e_l, r, and e_r$ are critical, but these types of relationships are not predominant in large datasets such as FreeBase. [2]

Neural Tensor Model

Neural Tensor Model (NTM) [42] has two k-dimensional vectors as well as a $k \times k$ bilinear operator L for each relationship to allow for extremely expressive connectivity patterns between the entities, and between entity and relationship. It scores triples as:

2.2 Compositional "Path" Embeddings

Knowledge Base Completion learns suitable models for predicting single edges, or triples, in a graph. However, these representations may not be suitable for scoring path queries, or queries that span a path of relations and entities in the graph. These queries arise quite frequently in question answering, as factoid questions such as "Where were Abraham Lincon's parent's born?" refer to two or more relations, namely "parents of" and "born in". This setting requires reasoning over numerous entities and relations along a path, and additionally, the multiple paths that can exist between pairs of entities [20, 29, 13].

A parallel approach to KBC utilizes features in the structure of the graph, particularly paths therein, and train on multi-hop path queries [46], where, for example, the appropriate tail entity is sought after starting at a head entity and traversing a path of relations [13], sometimes with a notion of a path probability [20, 25]. Some models like TransE and Bilinear are naturally compositional and suited for this setting [10], other times heavier compositional models like LSTMs are employed [29].

The discovery of paths in text is closely related to the concept of knowledge graph paths, and [23] show that the words in a textual instance can be used to compute the similarity of dependency paths that connect them. If the textual instance is anchored by two entities, then a system can draw inferences about the relations between them based on the dependency parse of the context surrounding them.

More recent work involving path representation learning with RNNs has evolved to incorporate the entities, not just the relations, along a path for learning. Additionally, sequence to sequence models offer improved performance by incurring loss incrementally along a path, not just at the end [6]

2.3 Leveraging Text and Relation Extraction Techniques

There is another very important community devoted to leveraging textual mentions of KB triples [45, 46, 14, 49, 48, 52]

The intuition motivating automatic relation extraction from text, and the related task of AKBC from text, is that for any pair of closely occurring entities recognized in a document, the context between them should present some kind of information about their relationship, even if it requires some level of inference. “distant supervision”, as it is called, avoids the expensive undertaking of labeling these textual instances with their relationship from the KB schema [28]. While not directly addressed here, a good resource covering leveraging knowledge bases for distant supervision in relation extraction can be found in [52]. It is desirable for AKBC systems to somehow leverage the vast quantities of textual data mined from the Internet, and this intuition has proven fruitful in some of the models discussed here [44, 45].

In order to learn entity and relationship embeddings from text for AKBC, we must have text that is annotated with entities and relationships, which requires both a schema of entities and relations. Errors from the entity linker will propagate, as linking is often confounded by the nuisances of natural language such as polysemy. Labeling a sentence with the relationship from $\mathcal{R}$ that it represents is a trickier problem addressed by relation extraction algorithms. However, distant supervision - labeling text that between two entity anchors with the relation that appears between those entities in a KB triple - is often used to quickly label large quantities of text with (perhaps multiple) relationships [28]

2.3.1 Separate Models for KB and Text embeddings

One of the first and simplest approaches for relation extraction is to extend a KB embedding model with a relationship identifier: learn a model that scores a relationship given a textual mention, and another model to encode the interactions between entities and relationships in the KB [52]

The relationship scoring mechanism $g(m, r)$ works by summing the word embeddings for the words in the textual mention m, and then taking the dot product of that with a relationship embedding r. The relationship that gives the highest dot product value has the highest rank, and is most likely to be expressed by the text. Updating the embeddings of both words and relationships can be done with SGD with some constraints. The same margin objective and training procedure that is described in section **??** can be used to build a relation ranker to find the best relations for a given text.

The KB model is TransE. These two models can be combined to define a relation extractor for KBC in the following way: for every unique pair of entities (e_l, e_r) that appear in the test set, all the corresponding textual mentions $M_{(e_l,e_r)}$ are collected and best candidate relation(s) $\hat{r}$ for them is predicted as:

$$\hat{r} = \underset{r \in \mathcal{R}}{\arg\max} \sum_{m \in M_{(e_l,e_r)}} g(m, r) \tag{2.1}$$

The candidate(s) $\hat{r}$ is/are then re-scored by incorporating the TransE model:

$$f(e_l, \hat{r}, e_r) = \sum_{m \in M_{(e_l,e_r)}} g(m, r) + h(e_l, \hat{r}, e_r) \tag{2.2}$$

Where $h(e_l, r, e_r)$ outputs the rank of r in the list of all relations as computed by the TransE model[2].

The first of the two stages can be perceived to filter only those relations deemed plausible by the textual model, and the final stage selects the relationship that best fits the KB model. The entire relation extractor thus encourages relationships to be consistent with the text and KB, despite that the parameters for the two are not jointly learned.

They train a scoring function (dot product) to measure the similarity between the text relation embedding (bag of words) and the KB relation embedding. And they also

[2] specifically, h outputs whether the rank is greater than some pre-defined threshold

train a linear scoring function (TransE) for positive triple in KB. They train both of them with ranking loss (hinge loss). In relation extraction, they use the first scoring function to find candidate relation and then combine the second scoring function to re-score the candidate relation. They use NYT+FB as training data (52 possible relationships and 121034 training mention where most of them has no relation).

2.3.2 Entity and Text Cooccurance

One of the first works to jointly train representations of entities and relationships with corresponding textual mentions was Universal Schemas [39]

Incorportating textual instances of KB triples began with the seminal paper on Universal Schemas by Riedel et al. They presented a variety of models (F, E, and N) that rely on a factorization of a relationship vs surface-norm matrix. The primary contribution of the work behind Universal Schemas is that the approach of distant supervision can be generalized by taking the union of all data schemas - including structured databases, knowledge base, and surface forms (raw text) - to yield a virtually infinite set of relations. Riedel et al demonstrate that combining textual and structural relations improves the ability to reason about both the structured and unstructured data [39]

Rather than modeling the semantic equivalence between relationships (”and force textual meaning into pre-defined boxes”) they take the approach of modeling implicatures and asymmetry in the data probabilistically: while ”historian-at” may imply ”professor-at”, the converse is not always true and furthermore, the two relations should remain distinct even though they may related enough to be clustered, as some algorithms do.

All of the models they introduce operate on a fact matrix $\mathcal{T}$ of size $|\mathcal{E}|^2$ by $|\mathcal{R}|$ where a cell at index i, j in the training matrix is binary random variable $y_{i,j}$ which takes the value 1 if the fact is true, 0 otherwise. The goal is to regress on the unknown cells at test time using

$$p(y_{i,j} = f_{riedel}(e_l, r, e_r) = p(1|\theta) = sigmoid(\theta) \tag{2.3}$$

Where θ is some latent feature representation of the entites and relations in the facts matrix. We will briefly describe the different models θ represents, while we save how they learn θ for section **??**.

In Model F, the fact matrix is factorized into the product of two matrices $A \in \mathbb{R}^{|\mathcal{E}|^2 \times k}$ and $V \in \mathbb{R}^{k \times |\mathcal{R}|}$. This model is well studied and allows for asymmetry between the subject and object position by skewing the relationship embedding.

In Model N (the ”neighborhood” model), the confidence of a tuple is given by the confidence of other tuples that share the same relation. Theta is a set of weights defined for every pair of relationships: $\theta_{(e_l,r,e_r)} = w_{r,r'} = \sum_{(e_1,r',e_2) \in \mathcal{T} \setminus (e_l,r,e_r)} w_{r,r'}$ which gives rise to a log linear classifier for each relation r.

Their final model, Model E (an abbreviation for Entity), captures the compatibility between entities and the subject/object positions of relations by learning a continuous vector representation of dimension k for each entity type. Two vectors for each relation are learned: r_s for the subject and r_o for the object entities to accommodate relationships that ”fan in” or ”fan out” a multitude of entity types. The score is given as

$$f(e_r, r, e_l) = r_s^T e_l + r_o^T e_r \tag{2.4}$$

The benefit of universal schemas is the ability to reason across myriad target relations that appear across a broad spectrum of natural text, and as an instance of never-ending learning, the universal schemas can be incrementally updated.

2.3.3 Entity and Text Co-occurance: TEKE

Another method which builds on the concept that related entities should exist in similar or overlapping textual contexts. Instead initializes and fine tunes entities/relationships with weighted averages of word embeddings of the words that appear in local neighborhoods surrounding the labeled entities/relationships.

Wang *et al.* build a ”co-occurance network” where nodes are the union of words and entities and edges are counts of co-occurances of the two nodes across all textual instances. They still use TransE model for ranking, but augment the entity/relationship embeddings with ”textual context embeddings” to improve performance on 1-to-N, N-to-1, and N-to-N triple scoring, the primary weakness of family of traditional ”translation” models like TransE, TransH, and TransR [49]

2.3.4 CNNs for Textual Relation Extraction

One of the first works to interpret a sentence or phrase in which two entities are collocated as a natural language expression of the relationship between them. The difficulty is finding a representation of that relatinoship in vector form that can be used in a triple scoring function.

Toutanova *et al.* from Microsoft Research built on the philosophy of Universal

Schemas by including surface forms (rather, the dependency parses) of textual instances into the knowledge graph to admit joint inferences over the Freebase relation schema and text. They employ models E and F (and Dist-Mult) to perform the triple ranking task. Their main contribution was finding k dimensional embeddings of dependency parses of the raw textual mentions using a 1-layer convolutional neural network (CNN), which was motivated by the observation that many synonymous surface forms share common words and dependency structures. The choice of using a CNN stems from a movement recognizing compositional models of text to yield better performance in discriminating the underlying relationship being expressed [45, 44]

$$\mathcal{L}_{\Theta} = \mathcal{L}_{KG}(\mathcal{T}; \theta_E, \theta_R) + \tau \mathcal{L}_{Text}(\mathcal{T}_{Text}; \theta_V, \theta_M) + \lambda ||\Theta||^2 \tag{2.5}$$

$\mathcal{L}_{KG}$ is responsible for the parameters of Θ corresponding to KG entity and relation embeddings, while $\mathcal{L}_{Text}$ is responsible for the vocabulary and relation extraction model, M.

Their model only uses textual mentions at training time to augment the embeddings of entities, as the CNN only takes as input the entity vectors for e_l and e_r and the word vectors of the textual mention. It does not update the KB relationship parameters explicitly, but it asks the model to interpret the extracted textual relation vector as a surrogate for the true relationship over which to translate from the head to tail entity. However, there is no mechanism to enforce that the extracted textual relationship representation should be ”close” to the KB relationship representation, and there’s no reason to believe the CNN would embed the text into the KB relationship space. This leaves the model vulnerable to noise in the raw text and the very common phenomenon that a given textual mention can represent many relationships (even if the head and tail entities are held constant). Alternatives to this will be discussed in section 2.3.6.

During training, the textual instances are subject to same objective as KB triples, but down-weighted by a factor of τ to account for the fact that textual triples are auxiliary. Their experiments show that co-training with text improves the base models, but the improvements are most marked when the text-augmented base models of E and DistMult are combined, suggesting that textual training does not make up for the deficiencies in any one model, but rather lead to broad improvements in models that already have the capacity to be expressive. Their results also show that performance on triples which had textual instances improved for nearly all models, suggesting that entities that engage in textual triples are fine tuned over the base models [45]

2.3.5 Aligning Entity and Word Vectors using Descriptions

Numerous papers investigate refining entity representations with their descriptions (from wikipedia, for instance) [27, 64, 56, 55, 57]

Wang *et al.* propose a new embedding technique that finds a shared space for both entities in a KB and words in a vocabulary, where an entity embedding is trained jointly with KB triples and textual resources, such as the words in the name of the entity itself, and the wikipedia anchors it's referenced in. As long as they appear in some textual form, aligning entities in this fashion provides flexibility to deal with out-of-KB entities [48].

They make the assumption that for two words (or a word and entity) appearing in a given context, there is some hidden variable representing the relation between them. They attempt to learn embeddings of both the word (or entity) and the hidden relationship. The probabilistic model they propose is conceptually similar to skipgram in that if two words co-occur, their inner products should be larger than if they don't co-occur.

For every word $w \in \mathcal{V}$ they wish to estimate a hidden relationship vector $r_{wv} \in \mathbb{R}^d$ between w and every other word v with which it co-occurs in the corpus within some window. As each word co-occurs with too many other words, on average, to justity learning a separate r_{wv} for every v, to reduce the number of parameters, they instead learn a "destination" vector $w' \in \mathbb{R}^d$ such that $w' \approx w + r_{wv}$. Slightly abusing notation, we allow w to be both a word and its word embedding, and in addition w' a word embedding for w is learned as well, resulting in $2|\mathcal{V}|$ parameters. The destination vector w' for word w is meant to satisfy an interpretation of the TransE objective with regard to another co-occuring word v in the following way: $z(w, v) = b - ||w' - v||^2$. That is, for v a word that occurs often with w, w' should be close to v; in general, w' is the region in the vector space containing the other elements that w often accompanies [48].

To align knowledge graph entities with words, each entity v is treated as a word in the vocabulary and given a vector e_v. The context around each Wikipedia anchor link is considered, and the text model is trained on (w, e_v) for all words w that appear near the anchor for entity v. Now, the w' learned for each word is trained to be near the entities which the word co-occurs with in anchor text. They propose a similar alignment model considering the words in the name of the entity as "co-occuring" with it. To train this model, they again use the negative sampling technique to approximate the original probabilistic objective with a ranking objective [48]. Their results improve on AKBC (triplet classification) as well as relation extraction.

2.3.6 Fully Joint Learning for Text, Entities, and Relations

The above works focus on either refining entity embeddings with text or extracting relationships from text. Han *et al.* argue that a truly joint model for learning KB embeddings with text would take advantage of a textual mention to update both the entities and relationship KG embeddings explicitly, rather than just one or the other [14]. They also use the same loss for KB entities as 2.5, but build on the CNN approach by modifying the text objective to force relationship representations extracted by the CNN, r_text to also be close to that of the KB relationship r, which is reminiscent of discriminative models. They define a textual scoring function $f_r(x, r) = ||r - r_{text}||^2$ that scores a textual embedding highly if it is close to the KG relationship r which it is supposed to represent [45]. The overall loss on the textual data set $\mathcal{T}_{Text}$ is defined analogously to the KG loss:

$$\mathcal{L}(\mathcal{T}_{text}) = \sum_{x \in \mathcal{T}_{text}} \sum_{r' \in \mathcal{R}} max\big(0, \gamma + f_r(x, r) - f_r(x, r')\big) \tag{2.6}$$

This objective treats the distantly supervised KB relation r as a label and encourages the CNN to find maximally discriminative relation embeddings for text. They validate this claim by performing a relation classification experiment (without incorporating the bookended entities) to rank the relations that a sentence is predicted to exhibit. They compare the precision-recall curves for relation classification between a CNN that was never exposed to KB embeddings to a CNN trained with KB embeddings and the loss function described above, with the jointly-trained CNN clearly outperforming the other [14]. Although this objective directly informs the relation embedding model to align textual embeddings with their corresponding KB relationship embeddings, it lacks the capacity to update entity embeddings for a textual triple, an important concept in [45]. Lastly, their model appends to each word vector in the sentence another vector encoding the position of the word in the sentence relative to each of the bookended entity anchors.

2.4 Hybrid Techniques

There have been myriad approaches to incorporate the structure of the graph, ”observable features” with latent factor models and even text. This technique of combining objectives and features was coined ”Additive relational effects (ARE)” by Nickel et al [31], and was first introduced for tensor factorization approaches to KBC. We briefly mention some others below. While many publications speculate that latent and observable techniques can be combined to produce better results, very few actually do so and still fewer report significantly improved results without extensive engineering.

2.4.1 Graph Features: NodeFeat and LinkFeat

Features for encoding the structure of a knowledge graph can also be derived from the graph itself in what are known as observed models. These models are extremely powerful and conceptually simple, with the only drawback being that the features must be hand-derived and often have very high dimensionality due to the branching factors that KGs often exhibit. They have proven adept at capturing correlations in the presence of different relation types between pairs of entities in multi-relational data, as well as patterns in relation paths that commonly exist between pairs of entities [44].

Some features Toutanova *et al.* extracted for an arbitrary triple (e_l, r, e_r) include

- Indicators for the entities occuring in the subject and object positions in the triple to capture any biases of where entities prefer to be in subject or object positions in a triple as well as unigram probabilities of entities overall.
- Length one paths between (e_l, e_r): a binary feature which fired for all triples (e_l, r', e_r) for which $r' \neq r$. This captures co-occurence of relationships that

exist between (e_l, e_r), for example ”work-in” and ”live-in” relationships.

- Length-ℓ paths between (e_l, e_r)
- Existence of inverse paths from e_r to e_r: a binary feature which fires for, say, length-1 paths as defined above. This captures correlations between inverse relations.

For observed models, the score of a triple is typically defined by the dot product between the feature vector and a vector of parameters [44].

2.4.2 Miscellaneous

There has been an interest in modeling logical and type constraints, for example, to enforce rules such as a marriage can be only between two people, the employer of a person is a business, etc. [5]

The semantic parsing community is a related field which seeks to map natural language to logical expressions over elements in the KG, that is, paths.

CHAPTER 3

CONTRIBUTION AND NOVEL WORK

3.1 A New Embedding Model: ModelE-X

We introduce a novel vector parameterization of entities and relations inspired by ModelE, which we name ModelE-X (Model-E Extended)[1]. Like ModelE, ModelE-X also defines two relation vectors, but provides three enhancements that improve expressiveness and flexibility:

$$f_{ModelE-X}(h, r, t) = \|\boldsymbol{e_h} \odot \boldsymbol{r_h} - \boldsymbol{e_t} \odot \boldsymbol{r_t}\|$$

where $\odot$ is elementwise vector multiplication (again a true triple should score high).

1. ModelE-X modulates the response of a relation component to its argument at a much finer granularity than a simple dot product due to the element-wise product (inspired by the gates of an LSTM)
2. Whereas ModelE can mistakenly give a high score to a false triple if one of the two dot products is high enough, ModelE-X requires the two response vectors to be similar, which is much less likely to occur spuriously.
3. We allow any choice of dissimilarity metrics, like ℓ_1 or ℓ_2, between response vectors (ℓ_1 works better in practice).

These advantages come without any sacrifice to complexity or runtime ($Ed + 2Rd$ parameters).

[1] We make our code, experiments, and logs available at `www.ANONYMIZEDLINK.com`.

Along a similar vein, we questioned whether an element-wise relation operator was too simple, and perhaps a separate matrix for the head and tail arguments was needed. We introduce "ModelE-XL" (Model-E Extended, Linear)

$$f_{ModelE-XL}(h, r, t) = \|\boldsymbol{W_r^h e_h} - \boldsymbol{W_r^t e_t}\|$$

as the natural generalization of ModelE-X (it reduces to ModelE-X if W_r^h and W_r^t are diagonal). Both ModelE-X and ModelE-XL give high scores to true triples.

3.2 Role of Textual Representations in Latent Factor Models

The community has typically chosen one of two roles for text: using vector representations of text as regularization to the embeddings of KB relations, e.g. $||r_{KB} - r_{text}||$, or using it as a noisy surrogate for the KB relation, e.g. in TranE, $||e_h + r_{text} - e_t||$. Typically, for a model like TransE (or DistMult) the loss of a triple is the sum

$$\texttt{margincost}(e_{head,pos}, e_{head,neg}) + \texttt{margincost}(e_{tail,pos}, e_{tail,neg})$$

where $\texttt{margincost}(e_{head,pos}, e_{head,neg})$ is defined to be $\big[||e_h + r - e_t|| + margin - ||e'_h + r - e_t||\big]_+$ for $e_{head,pos}$ a head entity that makes the triple (e_{head}, r, e_{tail}) true, wheres a negative head entity would make the triple false (non-existent in the KB). It is analogously defined for $\texttt{margincost}(e_{head,pos}, e_{head,neg})$ and even $\texttt{margincost}(r_{pos}, r_{neg})$.

1. Text-as-Regularization: first proposed by Han et al [14], the loss for a textual triple is defined to be only $||r - r_{text}||$ where r is (possibly any of) the KB relation that exists between the annotated head and tail entities.

2. Text-as-Relation: Used in [45], a textual relation extractor (in their case, a CNN) is built to encode text between two linked entities in a corpus into a vector that lives in the same space as the KB relation vectors. They then use the extracted vector in place of the KB relation vector in the hope that the KB entities will help align it, and hence train the extractor to map into the KB relation space.

In this work, we will present results on using a convex combination of Text-as-Regularization and Text-as-Relation during training.

3.3 New Evaluation Approaches

3.3.1 Micro Averaging

Relations are widely and unevenly distributed with respect to how many different kinds of entities they accept into the head and tail argument positions. One informative measure of KBC performance is based on the categories of relationships that arise from the cardinality of head and tail entities: 1-to-1, 1-to-Many, Many-to-1, and Many-to-Many. Bordes *et al.* compute the average number of heads (resp. tails) appearing in the FB15k or FB1M dataset given a pair (r, e_r) (resp. (r, e_l)). If the average number of tails per head (resp. heads per tail) is less than 1.5, then the relationship can be labeled as 1-to-X (resp. X-to-1) depending on what the multiplicity of the other direction is [2].

However, even within these buckets the distribution of frequency of relations may be highly skewed, where a few common relations dominate training and testing sets. Figure 4.1 shows histograms supporting this claim. We propose an extension of bucketing to "micro" averaging, that is, taking the average of averages of a ranking metric for each relation. This treats every relation equally and captures a more realistic notion of how good the learned relation representations are even for new or rarely seen relations, which appear very often in practice. In this way, we argue it is a better notion of generalizability.

3.3.2 Relation Ranking

Virtually no papers mention training with relation ranking loss, as many in the community assume that by training against corrupted entity triples, the relation representations will also be forced to improve. This turns out not to be the case, as our results in Table 4.4 show, where we compare a model which was trained on relation loss versus one that was not (all else being equal).

CHAPTER 4

EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Data and Experiments

4.1.1 Knowledge Base and Text Datasets

TODO: Mention some other data sets like YAGO, etc

1.2 billion facts, 80mil entities, 23k relationship types. Usually the top ≈ 4 million entities appearing in triples are selected.

FB15k: subset of FB entities that are also present in the Wikilinks database (code.google.com/p/wiki-links), and also both entities and relationships must have at least 100 mentions in FB. No redundant relationships (like reversal of another one). This results in 592,213 triplets with 14,951 entities and 1,345 relationships. FB15K also contains 26.2% 1-to-1 relationships, 22.7% 1-to-Many relationships, 28.3% of Many-to-1, and 22.8% of Many-to-Many relationships [2].

KG Dataset	**Entities**	**Relation Types**	**Facts**
Freebase	40 M	35,000	637 M
Freebase 15k	14,951	1,345	600 k
Wikidata	18 M	1,632	66 M
DPpedia (en)	4.6 M	1,367	538 M
YAGO2	9.8 M	114	447 M
Google Knowledge Graph	570 M	35,000	18 B

Table 4.1: Some statistics of the major knowledge graph datasets from [32]

Clueweb is a collection of hundreds of millions of high-quality web pages mined from the internet. Researchers annotated the text on the web pages with links from

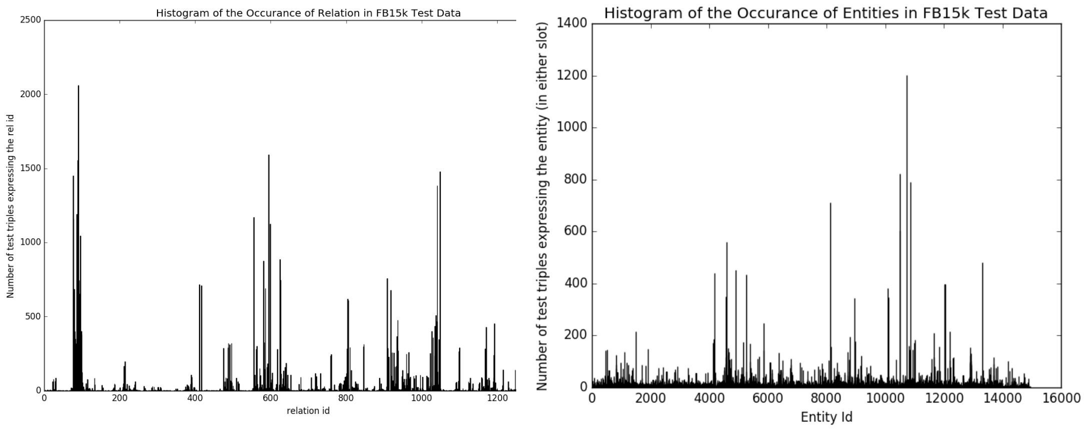

Figure 4.1: The distribution of the roughly 16,000 test triples for FB15k is highly non-uniform. Left is a histogram of the test set relations; there are 1,345 relations in FB15k. Right is a histogram of the test set entities, of which there are 14,951

named entities to their Freebase identifiers[1]. Clueweb annotations, when coupled with Freebase, result in tens of millions of textual instances [9]. For instance, when entities in Clueweb12 are coupled with mentions from FB237 the results is 37 million unique textual instances which cover nearly all of the 14,541 entities in the original FB-237 dataset, and nearly 40 percent of the training triples have textual instances [45].

We use the FB15k dataset with the train, dev, and test splits of [2] for our link prediction task. It contains 14,951 entities, 1,345 relations, 483,142 train triples, 50,000 dev triples, and 59,071 test triples. We re-implemented many canonical algorithms to eliminate sources of error and make comparisons between different algorithms' results more meaningful. We also do this because sometimes there isn't exact consensus about an algorithm's performance in the scientific community.

We tuned the margin γ over $\{0.2, 0.5, 1.0, 1.5, 2.0\}$, we set the dimension d of entity and relation embeddings to be 100 in all the models we implement except Bilinear, which was 50, but we acknowledge that nearly all models improve with more dimen-

[1] http://lemurproject.org/clueweb12/FACC1/

Micro Averages	**Mean Rank**		**Median Rank**		**MRR** (×100)		**Hits@10** (%)	
For Each Method	Rel	Entity	Rel	Entity	Rel	Entity	Rel	Entity
Unstructured	NA	1074 / 979	NA	-	NA	-	NA	4.5 / 6.3
UnstructuredDot*	NA	1172 / 1077	NA	377 / 312	NA	2.9 / 3.83	NA	7.02 / 9.14
ModelE*	**2.01 / 1.66**	460 / 363	1 / 1	84 / 50	79.2 / 89.9	13.8 / 22.1	99.1 / 99.2	24.7 / 34.0
Bilinear* (50)	3.09 / 2.75	182 / 84.4	1 / 1	21 / 7	**83.3 / 94.4**	20.9 / 36.1	**99.5 / 99.5**	38.7 / 56.1
BilinearDiag*	7.057 / 6.74	229 / 129	2 / 2	33 / 17	54.1 / 56.8	15.0 / 23.3	89.9 / 90.8	30.8 / 42.5
TransE*	5.3 / 4.9	800 / 725	2 / 2	79 / 46	54.1 / 58.9	13.6 / 19.6	94.1 / 94.6	24.9 / 32.4
TransR	-	198 / 77	-	-	-	-	-	48.2 / 68.7
TransH	-	212 / 87	-	-	-	-	-	45.7 / 47.1
PTransE	-	207 / 58	-	-	-	-	-	51.4 / 84.6
STransE	-	219 / 69	-	-	-	25.2 / 54.3	-	51.6 / 79.7
Node+LinkFeat	-	-	-	-	-	- / 82.2	-	**- / 87.0**
Our ModelE-X*	2.89 / 2.56	186 / 82.5	1 / 1	16 / 5	77.6 / 86.4	23.6 / 40.6	97.1 / 97.3	43.3 / 62.9
ModelE-X* (200)	2.68 / 2.34	188 / 80	1 / 1	11 / 2	79.7 / 89.4	27.1 / 53.8	98.1 / 98.2	49.8 / 76.6
ModelE-X* (500)	2.37 / 2.04	**168 / 54.6**	1 / 1	9 / 1	79.5 / 88.4	**29.3 / 64.4**	98.4 / 98.5	**53.4 / 83.4**
Our ModelE-XL*	-	223 / 124	-	28 / 11	-	17.8 / 30.2	-	33.6 / 49.2

Table 4.2: We compare our models against several classical and state-of-the-art algorithms using the traditional ”Micro” averages for evaluation metrics. Our ModelE-X is competitive and in some cases surpasses even state of the art algorithms such as STransE and LinkFeat. Each value reports ”raw” / ”filtered” metrics on FB15k test set. Parenthesis indicate the embedding dimension, else it is 100 for all models we implement, which are marked with an asterisk. Values reported in entity columns represent the average of left and right entity ranking metrics; NA means “not applicable”, and “-” means unreported.

Macro Averages	**Mean Rank**		**Median Rank**		**MRR** (×100)		**Hits@10** (%)	
For Each Method	Rel	Entity	Rel	Entity	Rel	Entity	Rel	Entity
ModelE*	**4.72 / 4.08**	151 / 129	**4 / 3**	13.5 / 9	54.7 / 66.6	25.9 / 34.5	93.7 / 94.3	47.9 / 56.5
Bilinear*	14.2 / 13.6	172 / 151	13 / 12	11 / 6	60.4 / 74.5	30.1 / 41.3	92.3 / 92.8	52.2 / 62.0
BilinearDiag*	53.7 / 53.1	284 / 264	50 / 49	41 / 33	14.8 / 15.9	15.9 / 19.8	35.1 / 36.2	30.6 / 35.3
TransE*	61.7 / 61.1	745 / 725	58 / 57	23 / 19	24.3 / 27.6	22.0 / 27.6	59.9 / 61.5	39.3 / 45.1
Our ModelE-X*	37.2 / 36.6	121 / 100	36 / 35	8 / 3.5	30.5 / 36.3	35.4 / 48.4	58.7 / 60.3	59.3 / 70.4
ModelE-X* (200)	38.9 / 38.2	151 / 129	37.5 / 37	6 / 2	38.5 / 46.6	38.7 / 56.9	69.1 / 70.2	64.2 / 78.7
ModelE-X* (500)	24.8 / 24.1	**88.3 / 64.9**	23.7 / 23	**4.5 / 1**	**40.4 / 48.9**	**44.3 / 66.9**	74.4 / 75.1	**68.7 / 84.1**
Our DoubleLinear*	-	466 / 447	-	17.5 / 11.5	-	22.4 / 32.2	-	42.3 / 52.2

Table 4.3: Our ModelE-X truly excels when all relations are treated equally, that is, the macro average across relations is taken (the average of relation-specific averages for a particular metric). ModelE-XL is rather difficult to train and regularize since it has quadratically-many parameters in d

sions[2] We choose between ℓ_1 and ℓ_2 dissimilarity metrics, but in nearly every case ℓ_1 was superior (and faster).

[2] For instance, [44] achieve Hits@10 = 79.7 for Bilinear with $d = 500$ and [60] achieve Hits@10 of 57.7 for $d = 100$, which we corroborate.

Relationships Prevalent in FB15k Test Set	Count
award/award_nomination/award_nominee	2060
film/film/release_date	1591
award/award_nomination/award	1555
people/profession/people_with_this_profession	1478
award/award_category/nominees	1451
people/person/profession	1384
award/award_nominated_work/award_nominations	1190
film/actor/film	1168
award/award_category/nominees	1160
film/film/starring	1123
award/award_winner/awards_won	1045

Table 4.4: Most prevalent relations in the FB15k test set, which has 16291 triples in total. Clearly, a handful of relationships, all related to either awards or films, comprise a disproportionate amount of the test data. Fortunately, the distribution of relations in the training set is roughly equivalent.

We ran mini-batch gradient descent with a batch size of one one-hundredth the size of the training set, with constant step size for up to 500 epochs, with early stopping if MRR did not improve after 30 epochs. We ran the dev set every 10 epochs. We re-normalized all entity embeddings to unit ℓ_2 norm after every minibatch, and we regularized relation-specific parameters with ℓ_2 norm (a regularization coefficient of 0.01 was satisfactory).

4.2 Results and Discussion

The "Micro" partition of Table 1 shows that ModelE-X outperforms or is competitive with state-of-the-art embedding models such as STransE [30] and more sophisticated algorithms that consider expensive path training, such as PTransE [25] and Node+LinkFeat [46], even outperforming them on Mean and Median Rank for entities.

Virtually no publications report relation ranking metrics, which contributes valuable

Micro Averages	**Mean Rank**		**Median Rank**		**MRR** ($\times 100$)		**Hits@10** (%)	
For Each Method	Rel	Entity	Rel	Entity	Rel	Entity	Rel	Entity
ModelE-X (300)	2.55 / 2.22	175 / 63.9	1.0 / 1.0	10 / 2.0	78.3 / 86.9	28.0 / 55.8	97.9 / 98.1	50.9 / 77.7
ModelE-X (300)*	25.6 / 25.3	171 / 60.2	10 / 9	8 / 1	32.8 / 33.2	29.8 / 66.9	50.9 / 51.7	54.5 / 85.5
Macro Averages	**Mean Rank**		**Median Rank**		**MRR** ($\times 100$)		**Hits@10** (%)	
For Each Method	Rel	Entity	Rel	Entity	Rel	Entity	Rel	Entity
ModelE-X (300)	29.3 / 28.7	97.6 / 74.6	28 / 27	5 / 2	36.3 / 43.9	42.4 / 61.9	68.4 / 69.7	66.4 / 80.4
ModelE-X (300)*	110 / 109	96.3 / 73.2	108 / 108	4 / 1	3.1 / 3.1	47.4 / 73.1	4.32 / 4.45	71.6 / 86.9

Table 4.5: Here we compare two very good models with the same parameters, except the loss function of the model marked with asterisk did not include contrasting positive and negative relations, yet here we evaluate it on relation ranking in the test data to show its weakness in both micro and macro metrics. Contrary to many beliefs, models trained without relation ranking will not automatically learn good relation representations, which in our opinion is a vulnerability. However, with the relation ranking loss, entity metrics do a suffer a bit, showing the trade-offs of these loss functions

information. For example, Table 1 suggests that in the usual micro average setting, weaknesses in TransE and BilinearDiag become more readily apparent based on relation metrics rather than entity ranking metrics, since they attain relatively higher Mean Ranks and lower MRRs, while their entity Hits@10 and MRRs do not raise concern These weaknesses become even more apparent for all metrics in the macro case. Relation ranking also reveals that Bilinear and ModelE (and ModelE-X) behave more similarly than previously thought, as they have similar MRR and Hits@10 on relations in both the macro and micro cases.

We note that relation ranking is useful in a number of tasks related to KBC, such as relation extraction from text. In that scenario, a pair of entities close to one another in the text are thus suspected of being linked, but it is uncertain exactly by which relation (or path of relations). We propose that a relation extraction engine could be augmented with a pretrained modelE or modelE-X model rather than training a separate distantly-supervised model [28].

Nearly all publications report the weighted average w.r.t the frequency of relations in the test data (micro average), which might be an over-simplification. It is unclear whether poor ranking performance is a symptom of the test data distribution, or a sign the model truly lacks capacity to capture relevant patterns. The macro average decouples these factors to allow for clearer insight into how the model behaves on the whole schema rather than a potentially biased sample of it. While it is common to report performance on buckets of relation types (1-to-1, 1-to-Many, Many-to-1, Many-to-Many) [2], a few relations may still dominate those buckets. To get the most realistic perspective on how a model will do on new relationships, metrics should be reported irrespective of the distribution of relationships in the dataset.

The macro average also reveals a massive discrepancy in TransE's relation metrics. TransE has been known to struggle with relationships that aren't 1-to-1 [2], but the micro average clearly masks this systemic weakness behind a distribution of relationships TransE can grasp. Interestingly, while most models suffer in the macro average category over the micro, ModelE and ModelE-X actually benefit on relation and entity MRRs and Hits@10 (bolded bottom row), suggesting that these models are adept at general reasoning across an entire schema.

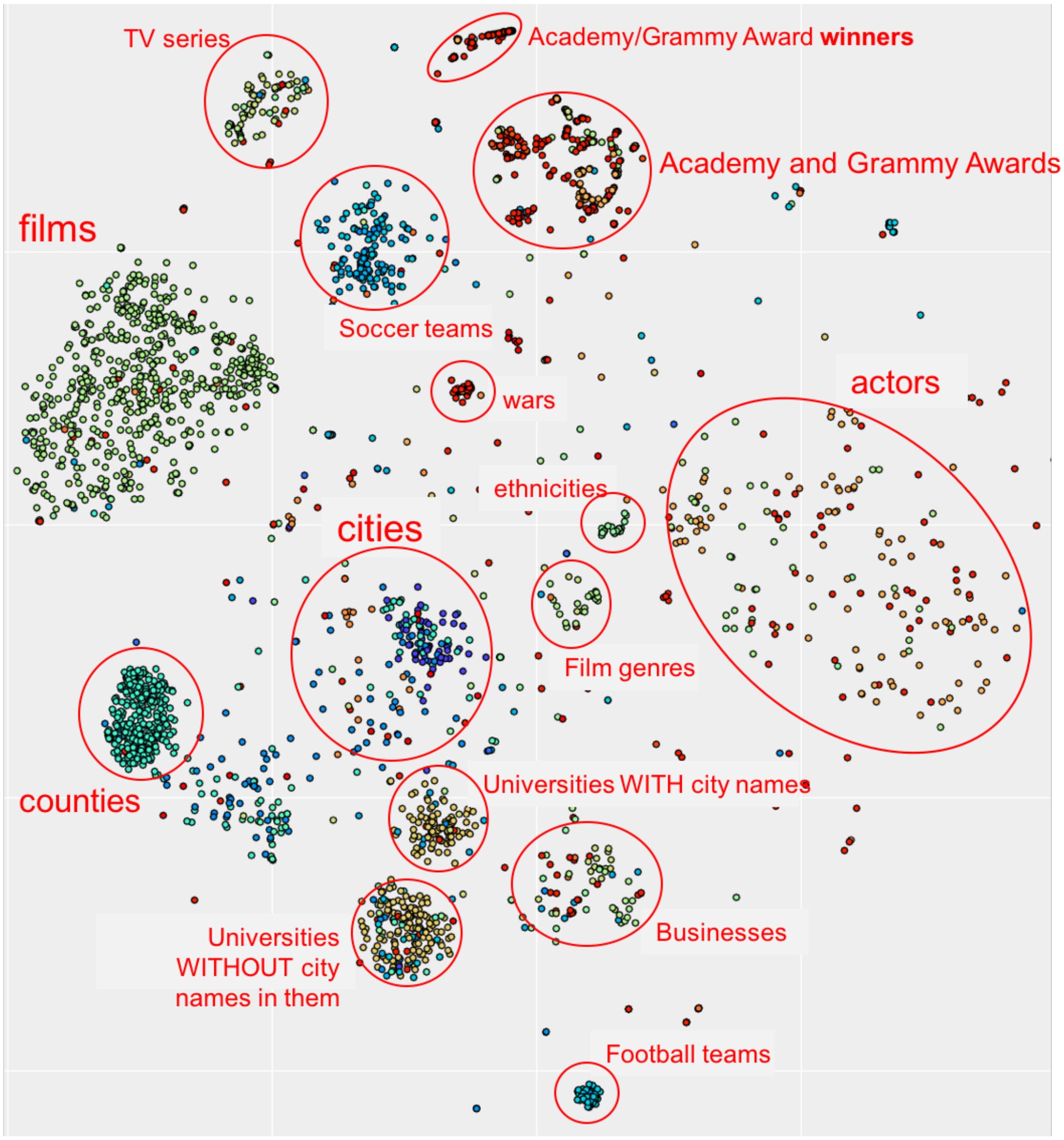

Figure 4.2: A T-SNE visualization of some of the vector representations learned by a Bilinear model for some of the most common entities in FB15k.

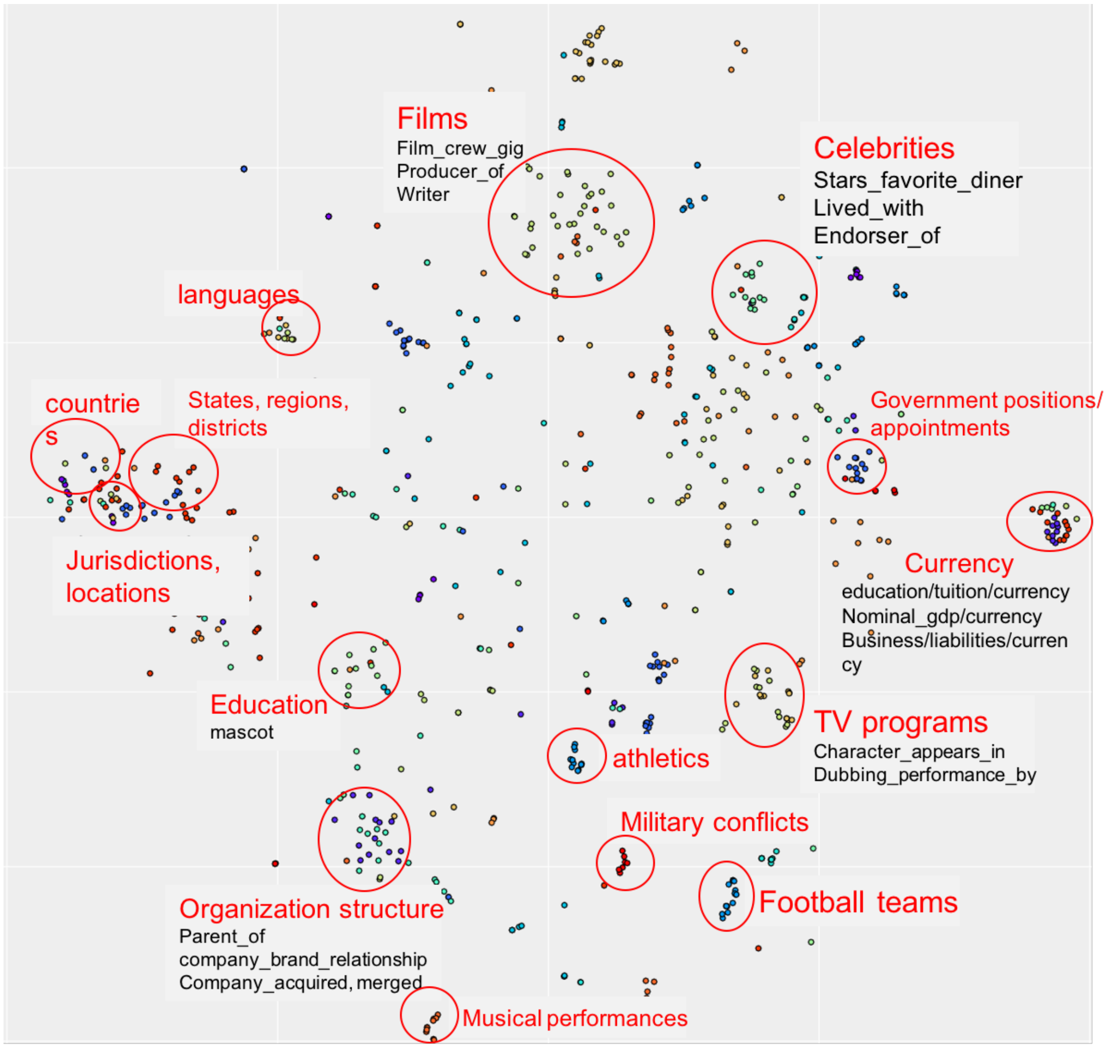

Figure 4.3: A T-SNE visualization of the head relation vectors learned by ModelE.

CHAPTER 5

A COMMON MISTAKE IN PRACTICE

I believe there is a mistake in the code for the original TransE results. The original 2013 NIPS paper [2] cites their organization's website[1] which links to the github implementation `https://github.com/glorotxa/SME`. In line 31 of `model.py`, the definition of `margincost`[2] is $[neg - pos + marge]_+$, when it should be $[pos - neg + marge]_+$. They even state the correct form in their paper. The score function for TransE, $||e_t + r - e_t||$, assigns a positive low score to positive triples (they have "low energy"), but their code's implementation is for a function that gives high scores to a positive triple. What their updates must be doing is as trivial as clustering entities that appear together (to give `pos` a high score, which subsequently forces the relation vectors to zero. The community/authors need to be aware of this, since many many papers (nearly all the ones cited in this document, for instance) simply report these incorrect numbers as truth. I've corrected my code and the entity ranking metrics are indeed worse than they reported. Relation ranking, however, is much better.

[1] `https://everest.hds.utc.fr/doku.php?id=en:codes`

[2] Where `pos` is the score of the positive triple (all scores are nonnegative), and `neg` is the margin cost of a random negative triple. The definition of `margincost`

BIBLIOGRAPHY

[1] Freebase data dumps.

[2] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795, 2013.

[3] Antoine Bordes, Jason Weston, Ronan Collobert, and Yoshua Bengio. Learning structured embeddings of knowledge bases. In *Conference on artificial intelligence*, number EPFL-CONF-192344, 2011.

[4] Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka Jr, and Tom M Mitchell. Toward an architecture for never-ending language learning. In *AAAI*, volume 5, page 3, 2010.

[5] Peter Chen. Entity-relationship modeling: historical events, future trends, and lessons learned. In *Software pioneers*, pages 296–310. Springer, 2002.

[6] Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. Chains of reasoning over entities, relations, and text using recurrent neural networks. *arXiv preprint arXiv:1607.01426*, 2016.

[7] Xin 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 *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610. ACM, 2014.

[8] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th*

ACM SIGKDD international conference on Knowledge discovery and data mining, pages 1156–1165. ACM, 2014.

[9] Evgeniy Gabrilovich, Michael Ringgaard, and Amarnag Subramanya. Facc1: Freebase annotation of clueweb corpora, version 1 (release date 2013-06-26, format version 1, correction level 0). *Note: http://lemurproject.org/clueweb09/FACC1/Cited by*, 5, 2013.

[10] Alberto Garcıa-Durán, Antoine Bordes, and Nicolas Usunier. Composing relationships with translations.

[11] Alberto Garcia-Duran, Antoine Bordes, Nicolas Usunier, and Yves Grandvalet. Combining two and three-way embeddings models for link prediction in knowledge bases. *arXiv preprint arXiv:1506.00999*, 2015.

[12] Matthew R Gormley, Mo Yu, and Mark Dredze. Improved relation extraction with feature-rich compositional embedding models. *arXiv preprint arXiv:1505.02419*, 2015.

[13] Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. *arXiv preprint arXiv:1506.01094*, 2015.

[14] Xu Han, Zhiyuan Liu, and Maosong Sun. Joint representation learning of text and knowledge for knowledge graph completion. *arXiv preprint arXiv:1611.04125*, 2016.

[15] LI Hang. A short introduction to learning to rank. *IEICE TRANSACTIONS on Information and Systems*, 94(10):1854–1862, 2011.

[16] Kazuma Hashimoto, Pontus Stenetorp, Makoto Miwa, and Yoshimasa Tsuruoka. Task-oriented learning of word embeddings for semantic relation classification. *arXiv preprint arXiv:1503.00095*, 2015.

[17] Wenqiang He, Yansong Feng, Lei Zou, and Dongyan Zhao. Knowledge base completion using matrix factorization. In *Asia-Pacific Web Conference*, pages 256–267. Springer, 2015.

[18] Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015.

[19] Alexander Konovalov, Benjamin Strauss, Alan Ritter, and Brendan O'Connor. Learning to extract events from knowledge base revisions. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1007–1014. International World Wide Web Conferences Steering Committee, 2017.

[20] Ni Lao, Tom Mitchell, and William W Cohen. Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539. Association for Computational Linguistics, 2011.

[21] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1188–1196, 2014.

[22] Jens Lehmann, Jörg Schüppel, and Sören Auer. Discovering unknown connections-the dbpedia relationship finder. *CSSW*, 113:99–110, 2007.

[23] Dekang Lin and Patrick Pantel. Dirt@ sbt@ discovery of inference rules from text. In *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 323–328. ACM, 2001.

[24] Thomas Lin, Patrick Pantel, Michael Gamon, Anitha Kannan, and Ariel Fuxman. Active objects: Actions for entity-centric search. In *Proceedings of the 21st international conference on World Wide Web*, pages 589–598. ACM, 2012.

[25] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379*, 2015.

[26] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *AAAI*, pages 2181–2187, 2015.

[27] Teng Long, Ryan Lowe, Jackie Chi Kit Cheung, and Doina Precup. Leveraging lexical resources for learning entity embeddings in multi-relational data. *arXiv preprint arXiv:1605.05416*, 2016.

[28] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009.

[29] Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. Compositional vector space models for knowledge base completion. *arXiv preprint arXiv:1504.06662*, 2015.

[30] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. Stranse: a novel embedding model of entities and relationships in knowledge bases. *arXiv preprint arXiv:1606.08140*, 2016.

[31] Maximilian Nickel, Xueyan Jiang, and Volker Tresp. Reducing the rank in relational factorization models by including observable patterns. In *Advances in Neural Information Processing Systems*, pages 1179–1187, 2014.

[32] Maximilian Nickel, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich. A review of relational machine learning for knowledge graphs. *arXiv preprint arXiv:1503.00759*, 2015.

[33] 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 machine learning (ICML-11)*, pages 809–816, 2011.

[34] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, pages 271–280. ACM, 2012.

[35] Feng Niu, Ce Zhang, Christopher Ré, and Jude Shavlik. Elementary: Large-scale knowledge-base construction via machine learning and statistical inference. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 8(3):42–73, 2012.

[36] Feng Niu, Ce Zhang, Christopher Ré, and Jude W Shavlik. Deepdive: Web-scale knowledge-base construction using statistical learning and inference. *VLDS*, 12:25–28, 2012.

[37] Jay Pujara and Lise Getoor. Building dynamic knowledge graphs. In *Proceedings of the Knowledge Extraction Workshop at NAACL-HLT*, 2014.

[38] Jay Pujara, Ben London, Lise Getoor, and William W Cohen. Online inference for knowledge graph construction. In *Fifth International Workshop on Statistical Relational AI*, 2015.

[39] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M Marlin. Relation extraction with matrix factorization and universal schemas. 2013.

[40] Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. Classifying relations by ranking with convolutional neural networks. *arXiv preprint arXiv:1504.06580*, 2015.

[41] Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. Reasonet: Learning to stop reading in machine comprehension. *arXiv preprint arXiv:1609.05284*, 2016.

[42] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934, 2013.

[43] Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM, 2007.

[44] Kristina Toutanova and Danqi Chen. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, 2015.

[45] Kristina Toutanova, Danqi Chen, Patrick Pantel, Hoifung Poon, Pallavi Choudhury, and Michael Gamon. Representing text for joint embedding of text and knowledge bases. In *EMNLP*, volume 15, pages 1499–1509. Citeseer, 2015.

[46] Kristina Toutanova, Xi Victoria Lin, and Wen-tau Yih. Compositional learning of embeddings for relation paths in knowledge bases and text.

[47] William Yang Wang and William W Cohen. Learning first-order logic embeddings

via matrix factorization. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI 2015), New York, NY, July. AAAI*, 2016.

[48] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph and text jointly embedding. In *EMNLP*, volume 14, pages 1591–1601. Citeseer, 2014.

[49] Zhigang Wang and Juanzi Li. Text-enhanced representation learning for knowledge graph. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 1293–1299. AAAI Press, 2016.

[50] Gerhard Weikum and Martin Theobald. From information to knowledge: harvesting entities and relationships from web sources. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 65–76. ACM, 2010.

[51] Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. Knowledge base completion via search-based question answering. In *Proceedings of the 23rd international conference on World wide web*, pages 515–526. ACM, 2014.

[52] Jason Weston, Antoine Bordes, Oksana Yakhnenko, and Nicolas Usunier. Connecting language and knowledge bases with embedding models for relation extraction. *arXiv preprint arXiv:1307.7973*, 2013.

[53] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory networks. *arXiv preprint arXiv:1410.3916*, 2014.

[54] Han Xiao, Minlie Huang, Yu Hao, and Xiaoyan Zhu. Transg: A generative mixture model for knowledge graph embedding. *arXiv preprint arXiv:1509.05488*, 2015.

[55] Han Xiao, Minlie Huang, and Xiaoyan Zhu. Ssp: Semantic space projection for knowledge graph embedding with text descriptions. *arXiv preprint arXiv:1604.04835*, 2016.

[56] Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In *AAAI*, pages 2659–2665, 2016.

[57] Jiacheng Xu, Kan Chen, Xipeng Qiu, and Xuanjing Huang. Knowledge graph representation with jointly structural and textual encoding. *arXiv preprint arXiv:1611.08661*, 2016.

[58] Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. Question answering on freebase via relation extraction and textual evidence. *arXiv preprint arXiv:1603.00957*, 2016.

[59] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In *EMNLP*, pages 1785–1794, 2015.

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

[61] Xuchen Yao and Benjamin Van Durme. Information extraction over structured data: Question answering with freebase. In *ACL (1)*, pages 956–966. Citeseer, 2014.

[62] Wen-tau Yih and Hao Ma. Question answering with knowledge base, web and beyond. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 1219–1221. ACM, 2016.

[63] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, Jun Zhao, et al. Relation classification via convolutional deep neural network. In *COLING*, pages 2335–2344, 2014.

[64] Dongxu Zhang, Bin Yuan, Dong Wang, and Rong Liu. Joint semantic relevance learning with text data and graph knowledge. *ACL-IJCNLP 2015*, page 32, 2015.