Mitigating the Effect of Out-of-Vocabulary Entity Pairs in Matrix Factorization for KB Inference

Prachi Jain[*1], **Shikhar Murty**[*1], **Mausam**[1] and **Soumen Chakrabarti**[2]
[1] Indian Institute of Technology Delhi
[2] Indian Institute of Technology Bombay
{p6.jain, shikhar.murty}@gmail.com, mausam@cse.iitd.ac.in, soumen.chakrabarti@gmail.com

Abstract

This paper analyzes the varied performance of Matrix Factorization (MF) on the related tasks of relation extraction and knowledge-base completion, which have been unified recently into a single framework of knowledge-base inference (KBI) [Toutanova *et al.*, 2015]. We first propose a new evaluation protocol that makes comparisons between MF and Tensor Factorization (TF) models fair. We find that this results in a steep drop in MF performance. Our analysis attributes this to the high out-of-vocabulary (OOV) rate of entity pairs in test folds of commonly-used datasets. To alleviate this issue, we propose three extensions to MF. Our best model is a TF-augmented MF model. This hybrid model is robust and obtains strong results across various KBI datasets.

1 Introduction

Inference over knowledge bases (KBs) has received significant attention in the last decade. Most early works focus on adapting probabilistic formalisms such as Markov Logic Networks and Bayesian Logic Programs for inferring new KB facts [Schoenmackers *et al.*, 2008; Niu *et al.*, 2012; Raghavan *et al.*, 2012].These require a set of inference rules, which can be harvested automatically using statistical regularities in KBs [Schoenmackers *et al.*, 2010; Berant *et al.*, 2011; Nakashole *et al.*, 2012; Jain and Mausam, 2016].

Recent research has integrated both rule learning and fact inference into a joint deep learning framework. This eschews explicit representation and learning of inference rules, and instead employs a way to score a (possibly new) KB fact (e_1, r, e_2) directly. Various algorithms differ in their scoring functions, by using different model assumptions.

This line of research can be further subdivided into two broad categories: matrix factorization and tensor factorization. In both cases the models learn one or more embeddings of the relation r, however, they differ in their treatment of entities e_1 and e_2. TF approaches (e.g., E [Riedel *et al.*, 2013], TransE [Bordes *et al.*, 2013], DistMult [Yang *et al.*, 2015], ComplEx [Trouillon *et al.*, 2016], TypedComplEx [Jain *et al.*, 2018], Rescal [Nickel *et al.*, 2011] models) learn separate embeddings for e_1 and e_2, whereas MF methods (e.g., F [Riedel *et al.*, 2013] and extensions [McCallum *et al.*, 2017b]) learn an embedding per entity-pair (e_1, e_2).

MF is considered as the leading technique for Relation Extraction (RE), problems where textual relations are available; whereas TF is the model of choice for Knowledge-Base Completion (KBC). Recent work by Toutanova *et al.* [2015] unified the two tasks of RE and KBC into a single framework, which we call *Knowledge-Base Inference* (KBI). This makes MF applicable to all KBI datasets. They applied MF to one dataset for KBC (FB15K-237), where it could not obtain results competitive with the state of the art.

Contributions: The main goal of this paper is to study MF's effectiveness for the general task of KBI and answer three key questions.

(1) Does MF perform well for KBI? Our extensive evaluation reveals that MF has an unusually varied performance across various KBI datasets, achieving MRR scores as high as 74% but also as low as zero.

(2) What makes MF performance so sensitive? We find that MF's performance can be explained by dataset sparsity, in the form of the fraction of entity-pairs that are outside the training vocabulary (OOV).

(3) Can we improve MF performance to obtain respectable scores in spite of high OOV rates? The original F model has a rather *ad hoc* way of handling OOVs — each OOV entity pair gets a random embedding. We propose three enhancements.

Our first model learns a single OOV vector. This ignores signals from the constituent entities, as it uses the same vector for all OOV entity pairs. Our second model generates entity pair vectors from the constituent entity embeddings, on the fly, using a generator layer that is trained to output "MF-like" embeddings. While the second model has better performance on OOV test examples, it degrades drastically on test facts where the gold entity pair is seen – rendering it far inferior to a TF model like ComplEx. In response, we propose a hybrid TF-MF model (inspired by [Singh *et al.*, 2015]). This model is robust and obtains strong results across all datasets.

We recognize that number of parameters in an MF model is much larger compared to TF models. When a joint model is trained naively (as proposed in [Singh *et al.*, 2015]), this imbalance causes TF models to not train properly. Our final

[*]First two authors contributed equally to the paper

hybrid model allows a more controlled flow of information across the constituent models, achieving much better results.

Additionally, we recognize that special care is needed to handle OOV entity-pairs when *evaluating* MF against TF. Otherwise an MF algorithm may erroneously appear to perform better than it really does, as in the case of F's performance on FB15K-237 [Toutanova *et al.*, 2015]. We describe the first unified KBI evaluation protocol that can meaningfully compare MF and TF approaches for KBI.

We contribute open-source implementations[1] of all models and testing protocols for further research.

2 Background and Experimental Setup

Toutanova *et al.* [2015] proposed a framework that unifies several closely related tasks: knowledge base completion (KBC), link prediction, and relation extraction (RE). We call this the task of Knowledge Base Inference (KBI). In this section, we describe key data sets, existing methods, loss functions, and evaluation protocol and its limitations.

2.1 Datasets

Most KB inference systems have used one or more of four popular KBs for evaluation. These include **WN18** (eighteen Wordnet relations [Bordes *et al.*, 2013]) and three datasets over Freebase (FB) – (1) **FB15K** [Bordes *et al.*, 2013] that has 1,345 relations; (2) **FB15K-237**, which is a subset of FB15K comprising 237 relations that seldom overlap in terms of entity pairs [Toutanova *et al.*, 2015]; (3) **NYT+FB**, which, along with FB triples, also includes dependency path-based textual relations from New York Times between entity mentions that are aligned with entities in Freebase [Riedel *et al.*, 2013].

Our literature search reveals that no model has been tested on all datasets. To the best of our knowledge, no paper reports results of E & F models on WN18 or FB15K, TransE on FB15K-237 or NYT+FB, and DM & ComplEx on NYT+FB. To better understand the strengths and weaknesses of each model, we compare all models on all datasets. We also release their open source implementations for further research.

2.2 Previous Approaches

We are given a KB with a set of entities $\mathcal{E}$ and relations $\mathcal{R}$ which may include canonical and ("open") textual relations. The KB also contains $\mathcal{T}$, a set of known valid tuples $t \in \mathcal{T}$. A tuple $t = \langle e_1, r, e_2 \rangle$ consists of a subject entity $e_1 \in \mathcal{E}$, object entity $e_2 \in \mathcal{E}$, and relation $r \in \mathcal{R}$. We use a shorthand ep_{12} to refer to entity pair (e_1, e_2). The goal in KBI is to predict the validity of a tuple $\langle e_1, r, e_2 \rangle$ not known to be in the KB. Previous approaches fit continuous representations (loosely, "embeddings") to entities and relation, so that the belief in the veracity of $\langle e_1, r, e_2 \rangle$ can be estimated as an algebraic expression (called a scoring function ϕ) involving those embeddings. The scoring functions for the models considered in this work are outlined in Table-1.

Our choice of these models is guided by the fact that these algorithms either form the basis of several recent papers on KB inference or are popular baselines for comparison studies (Toutanova *et al.* [2015], Trouillon *et al.* [2016], Demeester *et al.* [2016], Rocktaschel *et al.* [2015], Verga *et al.* [2017b], Verga *et al.* [2016], Singh *et al.* [2015], Nguyen *et al.* [2016], Xie *et al.* [2017]). Some models learn matrix embeddings instead of vectors [Nickel *et al.*, 2011; Socher *et al.*, 2013]. We don't study these, as they are typically outperformed by the models implemented in this paper.

[1] https://github.com/dair-iitd/KBI

Model (M)	**Scoring function** (ϕ^M)
F [Riedel *et al.*, 2013]	$\vec{r}^{\top} \cdot \overrightarrow{ep_{12}}$
TransE [Bordes *et al.*, 2013]	$-\left\|\vec{e_1} + \vec{r} - \vec{e_2}\right\|_2$
E [Riedel *et al.*, 2013]	$(\vec{e_1}^{\top} \cdot \vec{r_s}) + (\vec{e_2}^{\top} \cdot \vec{r_o})$
DistMult [Yang *et al.*, 2015]	$\vec{r}^{\top} \cdot (\vec{e_1} \bullet \vec{e_2})$
ComplEx [Trouillon *et al.*, 2016]	$\mathrm{Re}\langle \vec{r}^{\top} \cdot (\vec{e_1} \bullet \vec{e_2}^{\star}) \rangle$

Table 1: Scoring functions for MF (row 1) and TF models (row 2 onwards). Larger value implies more confidence in the validity of the triple. '$\cdot$' denotes dot product, '$\bullet$' denotes element-wise multiplication and $\star$ denotes the complex conjugate. Re refers to the real part of the complex valued score returned by the ComplEx model.

Note that some approaches (called tensor factorization or TF) embed each entity individually, whereas others (called matrix factorization or MF) embed *pairs* of entities. F is an MF model, since it uses the $\overrightarrow{ep_{12}}$ embeddings, while the others are TF models.

Loss functions: The models are trained such that tuples observed in the KB have higher scores than unobserved ones. Several loss functions have been proposed; we implement three common ones in this work: log-likelihood based loss [Toutanova *et al.*, 2015], max margin loss [Bordes *et al.*, 2013], and negative loglikelihood of the logistic model [Trouillon *et al.*, 2016]. All three loss functions sample a negative set $Neg(e_1, r)$ for every tuple, computed as $\{\langle e_1, r, e_2' \rangle | e_2' \in \mathcal{E} \wedge \langle e_1, r, e_2' \rangle \notin \mathcal{T}\}$, i.e., tuples formed by uniformly sampling entities that are not apriori known to be valid. Similarly, the set $Neg(r, e_2)$ is sampled.

Note that since MF models operate over entity pairs, they do not need two Neg sets. They use one set where new entity pairs (e_1', e_2') are sampled such that $\langle e_1', r, e_2' \rangle \notin \mathcal{T}$. These negative entity pairs are sampled only from the entity pairs found in $\mathcal{T}$, since embeddings for only those pairs get learned. We tried all the losses with all models on all datasets and report the best scores in the paper.

2.3 Standard Evaluation Protocol

To run the experiments at scale, we follow the most common automatic evaluation protocol for KBI systems. In this method the KB is split into train ($\mathcal{T}_{tr}$) and test tuples ($\mathcal{T}_{ts}$). The system can access only $\mathcal{T}_{tr}$ during training. For each test tuple, $\langle e_1^*, r^*, e_2^* \rangle \in \mathcal{T}_{ts}$, a query $\langle e_1^*, r^*, ? \rangle$ is issued to the trained model M. The model then ranks all entities $e_2 \in \mathcal{E}$ by decreasing $\phi^M(e_1^*, r^*, e_2)$. A higher rank of e_2^* in this list suggests a better performance of the model. Common metrics used to compare algorithms are mean reciprocal rank (MRR) and the percentage of e_2^*s obtained in top 1 (HITS@1) and 10 (HITS@10) results.

The testing procedure is typically run with two modifications. First, it is possible that the test fold is not complete: some of the e_2s ranked higher than e_2^* yield known tuples $\langle e_1^*, r^*, e_2 \rangle$. It is unfair to penalize the model for predicting

	Model	FB15K			FB15K-237			WN18			NYT+FB		
		MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10
1	ComplEx	**66.97**	**55.21**	**85.60**	**37.46**	**27.97**	55.95	**93.84**	**93.32**	**94.54**	69.43	64.84	76.55
2	DistMult	60.82	46.51	84.78	37.21	27.43	**56.12**	80.42	68.5	94.2	62.48	56.40	72.17
3	E	22.86	16.4	35.04	30.87	23.63	45.38	2.74	1.48	5.38	8.83	3.67	19.74
4	TransE	43.11	24.99	71.97	3.57	0.4	1.48	37.15	4.22	84.96	13.57	8.79	39.63
5	F (Old eval)	*33.62*	*22.27*	*60.20*	*28.01*	*13.21*	*64.76*	*82.95*	*71.76*	*98.84*	*89.28*	*83.48*	*97.84*
6	F (KBI eval)	13.35	9.45	17.03	0.0	0.0	0.0	0.14	0.04	0.20	74.34	68.96	80.01
7	MFreq($e_2\|r^*$)	24.91	18.84	36.03	33.05	25.45	47.60	3.10	1.92	5.28	11.42	6.23	20.39
8	MFreq($e_2\|e_1^*$)	8.22	15.61	15.61	0.01	0.0	0.0	0.17	0.38	0.38	**79.34**	**94.93**	**94.93**

Table 2: The first five rows compare 5 models on 4 datasets using the standard evaluation protocol. The 6^{th} row shows F's performance using our proposed KBI evaluation protocol. The last 2 rows report results of 2 most-frequent sanity-check baselines.

these. The *filtered* metrics remove the set $\{e_2|\langle e_1^*, r^*, e_2\rangle \in \mathcal{T}_{tr} \cup \mathcal{T}_{ts}\}$ from the ranked list [Bordes *et al.*, 2013].

The second modification applies primarily to MF models. In MF, an embedding is learned only for entity pairs that appear in $\mathcal{T}_{tr}$. Therefore, it is futile to score every $\langle e_1^*, r^*, e_2\rangle$ over a large range of e_2s, for most of which, $\vec{ep}_{12}$ is not even known. Instead, only those e_2s in a smaller set

$$E_2 = \{e_2|\exists r : \langle e_1^*, r, e_2\rangle \in \mathcal{T}_{tr} \cup \mathcal{T}_{ts}\} \quad (1)$$

are considered as candidates for ranking [Toutanova *et al.*, 2015; McCallum *et al.*, 2017b]. If entity pair (e_1^*, e_2) is not trained then a random vector is assumed for $\overrightarrow{ep_{12}}$.

3 Comparison Under Standard and Sanitized KBI Evaluation Protocol

In this section, we first present a detailed study of previous approaches under standard evaluation protocols, which exposes the limitations of existing evaluation protocols. We then propose a sanitized protocol which evaluates KBI models more accurately. We further add a few baselines overlooked in prior work, and present results adjusted for KBI evaluation.

Implementation details: We implement all algorithms in a common framework written using Keras/Theano [Chollet, 2015]. We use embeddings in $\mathbb{R}^{100}$ throughout, trained using Adagrad. 200 random negative samples are drawn per positive tuple. Margin γ is 1 for max margin methods. Entity and entity-pair vectors are re-normalized to unit norm after each batch update [Yang *et al.*, 2015]. Batch size is 20,000. Models are trained up to 200 epochs, but with early stopping on a validation set to prevent from overfitting. We train each model on each dataset using log-likelihood (LL), max-margin (MM) and logistic (L) loss functions and pick the best loss function (according to dev performance) for every setting. In particular, we find that TransE performs much better with MM loss. LL loss works better or at par in all other models except that MM outperforms LL for DistMult on WN18 dataset. L works best for ComplEx.

We follow the train-dev-test splits used in previous experiments for FB15K, WN18, and FB15K-237. The testsets $\mathcal{T}_{ts}$ are 3–10% random samples from $\mathcal{T}$. For NYT+FB, previous works had experimented on a test fold with only 80 correct tuples [Riedel *et al.*, 2013]. Since such a test set is rather small, and in keeping with our other data sets, we create our own train-test splits by randomly sampling about 2% tuples from $\mathcal{T}$. Only tuples with FB relations are used in the test set similar to previous experiments on this dataset.

Model performance with the Standard protocol: The first five rows of Table 2 report standard protocol performance of all the models across the datasets. E has good performance on FB15K-237, whereas TransE gets good scores on FB15K, however ComplEx emerges the most robust. For TF models on three datasets (FB15K, FB15K-237, WN18) our experiments are able to replicate (or improve upon) most reported results [Yang *et al.*, 2015; Bordes *et al.*, 2013; Toutanova *et al.*, 2015].[2] Since NYT+FB uses a new test fold, and F hasn't been tested on other datasets, those results cannot be directly compared against previous work.

We find that F outperforms ComplEx on NYT-FB dataset by wide margins and does not perform as well as ComplEx on the rest. It appears that a qualitative analysis of ComplEx vs F will shed light on their relative strengths and weaknesses. Our analysis reveals a limitation in the standard evaluation protocol that can inflate F's performance scores for OOV entity pairs.

The problem with the Standard protocol: Recall the second modification from standard protocol. When ranking possible entities e_2 using the score $\phi(e_1^*, r^*, e_2)$ from MF models, the standard protocol considers a subset E_2, instead of all entities in $\mathcal{E}$. This is because many entity pair embeddings (e_1^*, e_2) are not even trained in the model, and hence their scores will be meaningless. We call these **OOV entity pairs**. E_2 contains all entities for which the entity pair (e_1^*, e_2) is trained. Additionally, all such e_2^*s that are gold entities for some test query $\langle e_1^*, r^*, ?\rangle$ are also added to E_2. If these are not trained, a random vector is assumed for them.

Table 3(a) illustrates an extreme case where the gold entity pair (Bill Gates, Medina) is not seen in training, and only one e_2 (Seattle) is seen with e_1^*. Here, MRR for F model will be computed as 0.5 — a gross overestimation. Implicitly, (e_1^*, e_2^*) is getting ranked higher than all other OOV (e_1^*, e_2)s, whereas they should all be equal. In other words, the mere presence of $\mathcal{T}_{ts}$ in Eqn (1) leaks information.

Proposed Sanitized protocol: A correct KBI evaluation protocol must assume all OOV entities at the same rank, and output the average value over all possible rankings for them. In our sanitized protocol, we assume one random OOV entity pair (e_1^*, e_{oov}), identify *all* $e_2 \in \mathcal{E}$ that are OOV, assign them all the same score from the model and compute aggregate

[2] Since we use 100 dimensional embeddings throughout, we obtain slightly lower scores than [Trouillon *et al.*, 2016] for ComplEx, which uses 200 dimensional embeddings.

a.

⟨ Bill Gates, lives in, ?⟩	F (old)	F (new)
(Bill Gates, lives in, Seattle)	5.34	5.34
(Bill Gates, lives in, Medina)	0.04	-1.4
(Bill Gates, lives in, New York)	?	-1.4
⋮ ⋮ ⋮	?	-1.4
Reciprocal rank	0.5	∼0.0

b.

⟨ Tina Fey, lives in, ?⟩	F (old)	F (new)
(Tina Fey, lives in, New York)	2.30	2.30
(Tina Fey, lives in, Seattle)	1.1	1.1
(Tina Fey, lives in, Medina)	?	-2.12
⋮ ⋮ ⋮	?	-2.12
Reciprocal rank	1	1

Table 3: Original F with old evaluation protocol vs. F (trained OOV vector) with KBI evaluation protocol. Gold tuple in bold, and italics means that entity-pair was not seen during training. (a) Bill Gates is seen with one e_2 in training — not the gold answer; (b) Tina Fey is seen with two e_2s including the gold answer.

scores based on all possible rankings of OOV candidates. In Table 3(a), the MRR should be computed as the average of $\frac{1}{2}, \frac{1}{3}, \ldots$, which is very small.

We note that most existing MF models have used test folds in which none of the gold entity pairs are OOV (except FB15k-237). Hence, the results reported in most previous papers are not affected by our proposed fix. Also, if variants of MF models are being compared among themselves, while they may overestimate performance somewhat, the relative ordering of various models may not be affected. On the other hand, OOVs become a central issue when MF models are compared against or combined with TF models, since realistic levels of sparsity are very different in the two models.

Model performance with Sanitized protocol: When the sanitized protocol is used (Table 2 line 6), F's performance on all datasets drops drastically, to the extent that its performance is practically zero on two datasets, and extremely weak on the third. Also, its performance is worse that a simple baseline of MFreq$(e_2|r^*)$ (discussed in next paragraph) in all datasets. However, it continues to have the best numbers (among all trained models) for NYT+FB.

Why such a significant drop? The answer lies in entity pair OOV rates, i.e., the percentage of tuples in the test fold whose entity pairs were not seen while training. Table 4 reports some statistics about the datasets as well as their test sets. We notice that FB15K, FB15K-237 and WN18 all have a very high OOV rate, which is strongly correlated with poor performance of F. NYT+FB has a tiny OOV rate and F performs well on it. Because *single entity* OOVs are infrequent compared to entity pair OOVs, we expect TF methods to shine in large pair OOV regimes. Singh et al. [2015] highlight some theoretical difference between MF and TF models. Our data-driven analysis adds to that understanding. We believe that OOVs, and more generally, data sparsity, offer a more practical insight into differences between two model classes.

Most-frequent baselines: To improve our understanding of the difficulty of each dataset and the quality of each model beyond trivial choices, we introduce two baselines for our task. Given a query, $\langle e_1^*, r^*, ?\rangle$ our first baseline ranks all entities based on the frequency of their occurrence with relation r^*, i.e., it orders each entity e_2 based on the cardinality of the set $\{t|t = \langle e_1, r^*, e_2\rangle \wedge t \in \mathcal{T}_{tr}\}$. A similar baseline orders each entity e_2 based on its frequency of occurence with e_1^*, i.e., based on cardinality of the set $\{t|t = \langle e_1^*, r, e_2\rangle \wedge t \in \mathcal{T}_{tr}\}$. We name these baselines MFreq$(e_2|r^*)$ and MFreq$(e_2|e_1^*)$ respectively.

The last two rows of Table 2 report the performance of these baselines. It is satisfying to see that for FB15K and WN18 datasets, ComplEx outperforms the baselines by large margins. However, for FB15K-237, ComplEx is only marginally better than MFreq$(e_2|r^*)$. A closer analysis reveals that this dataset is constructed so that there is minimal entity-pair overlap between relations. How would any model, then, predict the best e_2 for a query $\langle e_1^*, r^*, ?\rangle$? If entity pairs have not been repeated much, a natural approach may just find the most frequent entities seen with the relation and order based on frequency. We checked some high MRR predictions made by ComplEx and found that often questions, like, what is the language of a specific website, were answered correctly as English. This is likely not because ComplEx figured out the language of each website, but because English was the most frequent one in the dataset.

We also observe that E's performance remains broadly similar to the performance of MFreq$(e_2|r^*)$. We attribute this to E's scoring function, since given e_1^* and r^*, the only term relevant for ranking e_2s is $\vec{e_2}^T \cdot \vec{r_o}$, i.e., the model looks for compatibility with r^* and ignores e_1^* completely.

Finally, for NYT+FB, under the sanitized protocol, MFreq$(e_2|e_1^*)$ beats F model significantly suggesting that while F is the best model on that dataset, it is not good enough. We explore this further in the next section.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	ep OOV (%)
FB15K	14,951	1,345	68.70
FB15K-237	14,541	237	100.00
WN18	40,943	18	99.52
NYT+FB	24,528	4,111	0.75

Table 4: No. of distinct entities, no. of relations and entity pair OOV rate, i.e., percentage of tuples in test set, whose entity pairs (ep) weren't seen while training.

4 Toward a Robust MF Model

The previous section highlights the importance of OOV entity-pairs in the performance of MF models. We now present a series of models to gracefully handle entity pair OOVs within MF.

4.1 MF with a Trained OOV Vector (F$_{vecOOV}$)

A natural extension to F is to explicitly model an OOV entity-pair vector. In particular, we represent a vector ep_{oov} for F and e_{oov} for TF. [3]This modification means that all facts with an OOV entity-pair will have the same score.

OOV vectors can be trained in many ways. We develop two baselines that don't train the vectors explicitly. The first

[3]The choice of parameterizing a single OOV vector is empirically motivated. We experimented with backoff-based parameterization, which learn a distinct OOV vector corresponding to each entity, but we did not observe any improvement likely due to overfitting.

Model	FB15K		WN18		NYT+FB	
	MRR	HITS@10	MRR	HITS@10	MRR	HITS@10
F (random)	13.35	17.03	0.14	0.20	74.34	80.01
F (average)	18.27	24.62	0.13	0.16	71.65	76.80
F (trained)	**20.21**	**27.42**	0.27	0.38	**81.51**	**93.67**
F (generated)	13.51	22.67	**0.80**	**1.22**	0.20	0.10

Table 5: Results on F model after explicitly modeling OOV vectors. OOV training outperforms other baselines, especially for NYT+FB. Results on FB15k-237 not reported, due to 100% OOV rate.

Model	OOV		Non-OOV	
	MRR	HITS	MRR	HITS
F_{vecOOV}	0.01	0	57.33	75.98
F_{genEP}	11.19	21.58	18.59	25.04
DM	55.34	80.13	72.87	94.99
ComplEx	61.22	80.76	79.58	96.22
F+ComplEx (AS)	1.12	1.23	51.37	81.74
F+ComplEx (RAL)	**61.37**	**80.95**	**80.82**	**96.45**

Table 6: Performance segregated by OOV and non-OOV test queries on FB15k. F+C (RAL) performs best on both OOV and non-OOV.

baseline (Table 5 line 1) assigns a *random* value to ep_{oov}. The second one called the *average* baseline computes ep_{oov} as the average of all (e_1, e_2) pairs that occur only once in training (Table 5 line 2).

We also propose a procedure to *train* ep_{oov} (Table 5 line 3). The broad motivation is to score a known tuple higher than a tuple with an OOV. To ensure this, we add ep_{oov} in the Neg set for each train tuple. This encourages the model to learn embeddings such that $\phi^F(r, ep_{12}) > \phi^F(r, ep_{oov})$. Thus, we ensure that the performance of F is maintained when a gold test entity pair is seen during training. Table 3(b) illustrates an example where the correct answer (New York) is seen with Tina Fey and OOV training doesn't displace its position.

To assess the effectiveness of the F_{vecOOV} model, we breakdown its performance on subset of test queries that have OOVs and non-OOV gold entity pairs. This analysis is meaningful only for FB15k, since other datasets have extreme entity-pair OOV rate (see Table 4). Clearly, as Table 6 shows, while F_{vecOOV} has extremely poor performance on OOVs (and thus weak performance overall), it performs decently on non-OOVs. We attribute this to the fact that the F_{vecOOV} is designed with the inductive bias that facts with OOV embeddings are worse than facts with seen entity-pair embeddings. Another shortcoming of this model is that it ignores all information present in constituent entities of an OOV entity pair.

4.2 Generate OOV Entity Pair Vectors (F_{genEP})

To fix the above shortcomings, we propose an enhancement to aid F in performing well on facts with OOV entity pairs. The main insight is to generate an informed OOV entity pair embedding by leveraging the information in the constituent entities. We would like these generated entity pair embeddings to be similar to what MF would have produced had these entity pairs been observed. To this end, we train a *generator* layer that is applied on the concatenation of an entity pair's constituent entities' vectors to produce entity pair embeddings.

We obtain such a generator layer as a by product of training a model with loss function $\mathcal{L}(\vec{ep_{12}}, \vec{r}, \vec{e_1}, \vec{e_2}) = \mathcal{L}^F(\vec{ep_{12}}, \vec{r}) + \mathcal{L}^C(\vec{ep_{12}}, \vec{e_1}, \vec{e_2}))$. Here $\mathcal{L}^F$ is the loss of the vanilla F model and $\mathcal{L}^C = \|ep_{12} - ep^g_{12}\|_2$, is regression loss, which we use to guide the generator layer to generate F like embeddings. We define $ep^g_{12} = f(P \cdot [e_1, e_2])$, where the generator layer $P \in \mathcal{R}^{2d \times d}$ and f can be any activation function. At test time, we concatenate the trained embeddings of constituent entities and use the learnt generator layer to obtain an embedding for an entity pair. Note that use generated entity pair embedding only for OOV entity-pairs, and use the trained embeddings of seen entity pairs from MF model.

At train time, we initialize the model with trained MF entity pair embeddings, MF relation embeddings and DistMult entity embeddings. We do not use any activation function (f) as it led to a slight decrease in performance. Table 6 shows that F_{genOOV} indeed has improved performance on OOV test facts. However, the performance on seen test facts drops significantly, leading to a weak overall performance.

4.3 Using TF to Guide MF

We now extend MF (from Section 4.1) based on the following two observations — (1) TF models like ComplEx perform robustly on both OOV and non-OOV test facts (see Table 6). (2) Singh *et al.* [2015] show that MF and TF models have complimentary strengths. Could we use a TF model (such as ComplEx) to guide MF to perform better on OOV test facts? **Background on TF augmented MF models:** Recall that Singh *et al.* [2015] find that the two models have complimentary strengths. In response, they developed a TF augmented MF model which outperforms all other models on artificial datasets and NYT+FB. Their best model (F+E) uses the scoring function $\phi^{E+F} = \sigma(\phi^E + \phi^F)$, where σ is the sigmoid function. We call this model an **additive score (AS)** model, since the scores (ϕ) of two models are added. Early works of Reidel *et al.* [2013] also experiment with a similar model for NYT+FB. Later, Toutanova *et al.* [2015] implement an AS F+E+DM model and tested it on FB15K-237.

We are motivated to develop an extension to MF that leverages TF to perform gracefully on both OOV and non-OOV test facts. For brevity, we refer to these MF extensions as joint MF-TF models, since they involve a TF model to guide the training of MF. Does an Additive Score model meet this requirement?

Additive loss (AL) joint model: Preliminary investigations (Table 6) reveal that additive score models can suffer substantial loss in performance for both OOV and non-OOV test facts. Table 8 shows drop in performance in the ComplEx component when trained jointly in additive score F+ComplEx model. It clearly shows that ComplEx's performance can reduce drastically due to joint training. A primary reason is that F scores overshadow ComplEx scores, since the scoring function in F involves a product of 2 small numbers and ComplEx involves a product of 3 small numbers.[4] Moreover, the

[4]To calibrate them, we tried standardizing scores obtained from pre-trained models. We also tried to learn a linear function to push ComplEx and F model scores to the same range simultaneously. We also tried sharing of relation parameters to allow information to flow from ComplEx to MF. Unfortunately, none of the approaches were robust across datasets.

	Model	FB15K			FB15K-237			WN18			NYT+FB		
		MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10	MRR	HITS@1	HITS@10
1	F	20.21	16.26	27.42	0.01	0.0	0.0	0.27	0.2	0.38	81.51	74.74	95.67
2	DM	60.82	46.51	84.78	37.21	27.43	56.12	80.42	68.58	94.20	62.48	56.40	72.17
2	ComplEx	66.97	55.21	85.60	37.46	27.97	55.95	93.84	93.32	**94.54**	69.43	64.84	76.55
3	F+E (AS)	26.24	20.59	37.35	29.71	22.15	44.39	1.60	0.07	4.04	82.46	75.99	92.21
4	F+ComplEx (AS)	16.84	11.48	26.42	32.90	24.18	50.25	90.02	88.94	91.54	79.41	72.78	89.70
5	F+E+DM (AS)	29.89	23.42	42.00	33.65	24.04	49.26	22.92	14.54	39.26	81.41	74.37	91.41
6	F+ComplEx (AL)	59.61	49.39	78.77	11.21	4.16	25.96	79.90	68.7	93.82	25.92	20.94	35.56
7	F+ComplEx (RAL)	**67.46**	**56.00**	**85.80**	**37.93**	**28.03**	**57.46**	**93.99**	**93.64**	94.48	**84.21**	**77.25**	**95.63**
8	F+ComplEx (Oracle)	69.02	58.80	84.99	37.46	27.97	55.95	98.71	98.39	99.15	89.79	81.97	96.69

Table 7: Performance of joint models. AL = additive loss. AS = additive score. F+ComplEx combined with regularized additive loss (RAL) is the highest scorer as well as most robust across all datasets.

number of parameters in MF models (vectors for entity pairs) significantly outnumber those in TF models (vectors for entities). This can lead to significant overfitting.

In response, we develop a different class of joint models in which instead of adding the scores (ϕs), we add their loss functions: $\mathcal{L}^{MF+TF} = \mathcal{L}^{MF} + \mathcal{L}^{TF}$. We name these *additive loss* joint models (AL). We expect this to be more resilient to overshadowing, since the joint loss expects each model's individual loss to decrease as much as possible. One may note that AL style of training is equivalent to training the models separately. However, joint training makes other extensions possible, such as regularization.

Regularized additive loss (RAL): We extend the vanilla AL joint model to a *regularized* joint model in which the parameters of MF model are L2-regularized. We expect this regularization to encourage a reduction in overfitting caused due to the large number of MF parameters. Overall, our final joint model has the loss function:

$$\mathcal{L}^{MF+TF}(\theta^{MF}, \theta^{TF}) = \mathcal{L}^{MF}(\theta^{MF}) + \mathcal{L}^{TF}(\theta^{TF}) + \lambda \left\| \theta^{MF} \right\|_2$$

At test time, for a query $\langle e_1^*, r^*, ? \rangle$ an AL model cannot simply add the scores, since some entity-pairs may be OOVs. We develop various backoff cases, reminiscent of traditional backoff in language models [Manning and Schütze, 2001]. For every e_2:

- **Case 1:** $(e_1^*, e_2) \in \mathcal{T}_{tr}$. Score of tuple is $\phi^{TF}(e_1^*, r^*, e_2) + \phi^{MF}(e_1^*, r^*, e_2)$.
- **Case 2:** $(e_1^*, e_2) \notin \mathcal{T}_{tr}$, but e_2 is seen in training. Score of tuple is $\phi^{TF}(e_1^*, r^*, e_2) + \phi^{MF}(e_{oov}, r^*, e_{oov})$.
- **Case 3:** e_2 is not seen in training. Score of tuple is $\phi^{TF}(e_1^*, r^*, e_{oov}) + \phi^{MF}(e_{oov}, r^*, e_{oov})$.

Results: Table 6 shows that RAL F+ComplEx performs well on OOV and non-OOV test queries. Regularization penalty λ is tuned over a small devset from within the training set.

Robustness: RAL F+ComplEx performs well for OOV & Non-OOV test cases, hence is robust to all testing conditions. To validate the hypothesis we evaluate the model on all the datasets in background section. Table 7 compares its performance with individual models as well as other joint models.

We find that different additive score models (rows 3–5) perform well on some datasets, but are not robust across them. For example, in FB15K none of these are able to match up to ComplEx's performance. We attribute this to overfitting by F, which makes the model believe that ϕ^F is predicting the tuple very well. This lets F override TF and reduces the joint model's need to learn the best TF model(s). Note that row 3 and row 5 are the models reported in (Singh *et al.* [2015]) and [Toutanova *et al.*, 2015], respectively.

Rows 6 and 7 report the results of additive loss F+ComplEx models, both without and with regularization. As anticipated, adding the losses improves performance since both models get trained well. Moreover, regularization also helps considerably since now the model is not overwhelmed by too many F parameters. RAL version of F+ComplEx achieves the best scores in all datasets, and is the state-of-the-art among all reported numbers for FB15k-237.

Row 8 of Table 7 also shows the accuracy of an oracle model that, for every test query, post-facto selects the model with the more accurate score (between ComplEx and F). This upper bounds the performance expected from a perfect joint ComplEx+F model, fixing the constituents. We find that the oracle is only 4-5 MRR percentage points better than our best model for two datasets, and the differences are much less for the other two. Overall, it suggests that our proposed joint model obtains a strong robust performance.

5 Other Related Work

The original F model has been extended to incorporate first order logic rules (Rocktaschel *et al.* [2015]; Deemester *et al.* [2016]), to predict for relations not seen at training time [Verga *et al.*, 2016; McCallum *et al.*, 2017a], etc. It has also been extended to generate embedding of a new entity-pair on the fly (Verga *et al.* [2017b]). However their work is different from ours, since, at test time, they expect knowledge of several tuples between the same entity pair. Similarly, other TF models also exist, for example, Parafac [Harshman, 1970], Rescal (Nickel *et al.* [2011]) and NTN [Socher *et al.*, 2013]. These are older models which have been outperformed by models evaluated in this paper. More recent models have also been introduced such as a model using holographic embeddings (Nickel *et al.* [2016]). ComplEx model is shown to be equivalent to the holographic models [Wang *et al.*, 2017]. The joint model formulation can incorporate new models seam-

Dataset	Δ MRR	Δ HITS@10
FB15K-237	-4.09	-5.32
FB15K	-62.18	-75.67
NYT+FB	-69.03	-76.0
WN18	-4.62	-14.36

Table 8: Change in performance of ComplEx initialized with corresponding embeddings extracted from ComplEx+F (AS).

lessly and improve even further. The learned embeddings can use additional information such as typing [Chang *et al.*, 2014; Jain *et al.*, 2018], can be used to mine logical rules [Yang *et al.*, 2015] and induce schemas [Nimishakavi *et al.*, 2016].

6 Discussion and Future Work

We now list two observations that suggest important directions for future research in KB inference.

Dataset Characteristics: Our work subjects datasets to natural sanity checks. First, we introduce two most frequent baselines (Table 2) to understand the nature of the KBs. Second, we compute entity-pair OOV rates (Table 4) as a rough predictor of the relative success of the TF and MF families. Finally, in Table 9, we report the singleton and doubleton percentages (for entity pairs). A singleton is an entity-pair occurring only once in the data ($\mathcal{T}_{tr} \cup \mathcal{T}_{ts}$) and a doubleton is an entity pair that occurs exactly twice. Doubletons have a strong effect in the scenario painted in Table 3.

We find that most datasets have an idiosyncrasy, which raises the question whether they are good representatives for naturally occurring KBs. In particular, WN18 and FB15K-237 have near 100% entity-pair OOV rates, unlikely to be the case in real KBs. In FB15K-237 the best models are not much better than MFreq($e_2|r^*$) baseline. This is because the dataset is artificially constructed to avoid relations with entity-pair overlap. But, this reduces its ability to make many interesting inferences. For NYT+FB, MFreq($e_2|e_1^*$) performance has a strong performance with 95% score on HITS@10. Moreover, learned models are able to improve its MRR by only about four percentage points. Statistics in Table 9 reveal that this could be because the dataset has an unusually high number of entity-pair doubletons: it is the only data set where doubletons by far outnumber singletons. It is unlikely that such a distribution occurs in a naturally occurring dataset. FB15K appears to pass our sanity tests. We believe that focus on better datasets will likely help us in better progress on KB inference.

Path based inference: In KBs, a common type of inference is based on relation paths (or Horn-clauses), e.g., (Michael Jordan, teaches at, Berkeley) and (Berkeley, is located in, California) implies (Michael Jordan, teaches in, California). To assess the ability of inference models to automatically learn such relation paths, we tested them on artificial datasets, where we provided many instances of two-hop paths with relations r_1 and r_2 implying a third relation r_3. We find that *none* of the four models are effective at predicting such relations. A study similar to ours comparing the latest models that train over relation paths [Guu *et al.*, 2015; García-Durán *et al.*, 2015; Toutanova *et al.*, 2016] will benefit our understanding of path-based inference.

Dataset	Singleton Rate	Doubleton Rate
FB15K	83.83%	12.19%
FB15K-237	90.52%	8.64%
WN18	99.80%	0.20%
NYT+FB	8.06%	59.04%

Table 9: The fraction of entity-pairs occurring exactly once and exactly twice. NYT+FB has an unusual distribution.

7 Conclusion

We present the first study on the effectiveness of MF for KBI, which unifies Relation Extraction and Knowledge Base Completion. After replacing the standard evaluation protocol with our sanitized proposal, we find that MF's performance is highly varied — it obtains MRR scores from 0 to 74% on different datasets. We also propose two simple frequency baselines, and are surprised to find that MF's performance is worse than than the better baseline in *all* domains! Further analysis reveals that MF performs poorly at high-OOV rates. We develop a series of extensions aimed at mitigating the effect of OOVs in MF. MF's performance improves by training OOV embeddings. Our most successful model uses ComplEx to augment our improved version of MF via a regularized additive loss. This hybrid model is highly robust and has the best performance on all datasets. Note that a basic MF added to TF isn't robust – only an "OOV trained" MF, when integrated with TF, attains good performance.

We release our code for all models and evaluation protocols for further use by research community.[5]

Acknowledgements

This work is supported by Google language understanding and knowledge discovery focused research grants, a Bloomberg award, an IBM SUR award, and a Visvesvaraya faculty award by Govt. of India to the third author. It is also supported by a TCS Fellowship to the first author. We thank Microsoft Azure sponsorships, and IIT Delhi HPC facility for computational resources. We also acknowledge the insightful comments by Sameer Singh, Dan Weld and Oren Etzioni on an early draft of the work.

References

[Berant *et al.*, 2011] Jonathan Berant, Ido Dagan, and Jacob Goldberger. Global Learning of Typed Entailment Rules. In *ACL-HLT*, pages 610–619, 2011.

[Bordes *et al.*, 2013] 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, *NIPS*, pages 2787–2795, 2013.

[Chang *et al.*, 2014] Kai-Wei Chang, Wen-tau Yih, Bishan Yang, and Christopher Meek. Typed Tensor Decomposition of Knowledge Bases for Relation Extraction. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *EMNLP*, pages 1568–1579. ACL, 2014.

[Chollet, 2015] François Chollet. Keras. https://github.com/fchollet/keras, 2015.

[Demeester *et al.*, 2016] Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. Regularizing relation representations by first-order implications. In Jay Pujara, Tim Rocktäschel, Danqi Chen, and Sameer Singh, editors, *AKBC@NAACL-HLT 2016*, pages 75–80. The Association for Computer Linguistics, 2016.

[5] https://github.com/dair-iitd/KBI

[García-Durán *et al.*, 2015] Alberto García-Durán, Antoine Bordes, and Nicolas Usunier. Composing Relationships with Translations. In *EMNLP*, pages 286–290, 2015.

[Guu *et al.*, 2015] Kelvin Guu, John Miller, and Percy Liang. Traversing Knowledge Graphs in Vector Space. In *EMNLP*, pages 318–327, 2015.

[Harshman, 1970] Richard Harshman. Foundations of the PARAFAC Procedure: Models and Conditions for an "explanatory" Multi-modal Factor Analysis . *UCLA Working Papers in Phonetics*, 1970.

[Jain and Mausam, 2016] Prachi Jain and Mausam. Knowledge-Guided Linguistic Rewrites for Inference Rule Verification. In *NAACL-HLT*, pages 86–92, 2016.

[Jain *et al.*, 2018] Prachi Jain, Pankaj Kumar, Mausam, and Soumen Chakrabarti. Type-sensitive knowledge base inference without explicit type supervision. In *ACL*, 2018.

[Manning and Schütze, 2001] Christopher D. Manning and Hinrich Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 2001.

[McCallum *et al.*, 2017a] Andrew McCallum, Arvind Neelakantan, Rajarshi Das, and David Belanger. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *EACL*, pages 132–141, 2017.

[McCallum *et al.*, 2017b] Andrew McCallum, Arvind Neelakantan, and Patrick Verga. Generalizing to unseen entities and entity pairs with row-less universal schema. In *EACL*, pages 613–622, 2017.

[Nakashole *et al.*, 2012] Ndapandula Nakashole, Gerhard Weikum, and Fabian Suchanek. PATTY: A Taxonomy of Relational Patterns with Semantic Types. In *Joint EMNLP-CoNLL*, pages 1135–1145, 2012.

[Nguyen *et al.*, 2016] Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. STransE: A novel Embedding model of entities and relationships in Knowledge bases. In *NAACL-HLT*, pages 460–466, 2016.

[Nickel *et al.*, 2011] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A Three-Way Model for Collective Learning on Multi-Relational Data. In *ICML*, pages 809–816, 2011.

[Nickel *et al.*, 2016] Maximilian Nickel, Lorenzo Rosasco, and Tomaso A. Poggio. Holographic Embeddings of Knowledge Graphs. In *AAAI*, pages 1955–1961, 2016.

[Nimishakavi *et al.*, 2016] Madhav Nimishakavi, Uday Singh Saini, and Partha P. Talukdar. Relation Schema Induction using Tensor Factorization with Side Information. In *EMNLP*, pages 414–423, 2016.

[Niu *et al.*, 2012] Feng Niu, Ce Zhang, Christopher Ré, and Jude W. Shavlik. Elementary: Large-Scale Knowledge-Base Construction via Machine Learning and Statistical Inference. *IJSWIS*, pages 42–73, 2012.

[Raghavan *et al.*, 2012] Sindhu Raghavan, Raymond J Mooney, and Hyeonseo Ku. Learning to "Read Between the Lines" using Bayesian Logic Programs. In *ACL*, pages 349–358, 2012.

[Riedel *et al.*, 2013] Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. Relation Extraction with Matrix Factorization and Universal Schemas. In *HLT-NAACL*, pages 74–84, 2013.

[Rocktäschel *et al.*, 2015] Tim Rocktäschel, Sameer Singh, and Sebastian Riedel. Injecting Logical Background Knowledge into Embeddings for Relation Extraction. In *NAACL-HLT*, pages 1119–1129, 2015.

[Schoenmackers *et al.*, 2008] Stefan Schoenmackers, Oren Etzioni, and Daniel S Weld. Scaling Textual Inference to the Web. In *EMNLP*, pages 79–88, 2008.

[Schoenmackers *et al.*, 2010] Stefan Schoenmackers, Oren Etzioni, Daniel S Weld, and Jesse Davis. Learning First-Order Horn Clauses from Web Text. In *EMNLP*, pages 1088–1098, 2010.

[Singh *et al.*, 2015] Sameer Singh, Tim Rocktäschel, and Sebastian Riedel. Towards Combined Matrix and Tensor Factorization for Universal Schema Relation Extraction. In *VS@NAACL-HLT*, pages 135–142, 2015.

[Socher *et al.*, 2013] Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Y. Ng. Reasoning With Neural Tensor Networks for Knowledge Base Completion. In *NIPS*, pages 926–934, 2013.

[Toutanova *et al.*, 2015] 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*, pages 1499–1509, 2015.

[Toutanova *et al.*, 2016] Kristina Toutanova, Xi Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. Compositional Learning of Embeddings for Relation Paths in Knowledge Bases and Text. In *ACL*, pages 1434–1444, 2016.

[Trouillon *et al.*, 2016] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex Embeddings for Simple Link Prediction. In *ICML*, pages 2071–2080, 2016.

[Verga *et al.*, 2016] Patrick Verga, David Belanger, Emma Strubell, Benjamin Roth, and Andrew McCallum. Multilingual Relation Extraction using Compositional Universal Schema. In *NAACL-HLT*, pages 886–896, 2016.

[Wang *et al.*, 2017] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743, 2017.

[Xie *et al.*, 2017] Qizhe Xie, Xuezhe Ma, Zihang Dai, and Eduard H. Hovy. An interpretable knowledge transfer model for knowledge base completion. In *ACL*, pages 950–962, 2017.

[Yang *et al.*, 2015] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *ICLR*, 2015.