Integrated Embedding Approach for Knowledge Base Completion with CNN

Samuel Chen
Faculty of Information Technology; Melbourne Polytechnic; Preston Victoria 3072; Australia;
e-mail: samuelchen@melbournepolytechnic.edu.au

Shengyi Xie
Experimental Training Center; Fujian Vocational College of Agriculture; Fuzhou 350181; Fujian;
phone: +86 153 5673 2821; e-mail: fjnyxsy@qq.com

Qingqiang Chen
Information Science and Engineering College; Fujian University of Technology; Fuzhou 350118; Fujian;
phone: +86 131 7811 4567; e-mail: 516040610@qq.com

Corresponding author: 516040610@qq.com

In knowledge graph embedding, sophisticated models may suffer from over-fitting and high computational costs. On the contrary, transitional models come with lower complexity but struggle with complex relations while integrating relational attributes or semantic information could help with embedding representation. Convolutional neural networks employed in recent researches are able to model interactions between entities and relations efficiently but may ignore global dependencies. To tackle such problems, a model called Integrated Embedding Approach for Knowledge Base Completion (IEAKBC) is proposed. In this model, embedding representations of entities and relations are put together to constitute a three-column, $k$ dimensional matrix for each triplet. Afterwards, features from different relations are integrated into head and tail entities thus forming fused triplet matrices. Both sets of matrices are used as inputs to a convolutional neural network (CNN) framework. In CNN, kernels go over each row of the matrices for feature extraction. Feature maps are subsequently concatenated and weighted for output scores to discern whether the original triplet holds or not.
Experiments on four benchmark datasets show that our model performs well on complex relations while retaining transitional characteristics. Finally, we apply the model to a personalized search application, verifying its practicality in real-world scenarios.

**KEYWORDS:** Knowledge Representation, Knowledge Base Completion, CNN, Translation Mechanism, Link Prediction.

### 1. Introduction

Knowledge Base (KB) [35] encodes structured facts in the form of triples, each of which is composed of a head entity, a relation and a tail entity, denoted as (h, r, t). In knowledge graph (KG) structure, nodes refer to head entities h and tail entities t, while edges are relations r between nodes. KBs, such as Freebase [5], WordNet [19] and NELL [41], are widely used in semantic searches [77], question answering [7, 23], visual detection [1], etc.

There are a lot of missing facts or incomplete triples in existing KBs, lacking entities or relations [72]. Knowledge Base Completion (KBC) [51] aims to solve this problem by inferring new facts with known information, or by introducing external resources. Embedding methods are among the mainstream research directions for KBC; after entities / relations are embedded into tensor / matrix / vector spaces, local / global patterns and semantic features of facts are extracted with tensor / matrix factorization or vector translation / rotation operations so as to handle downstream tasks such as link prediction and triple classification. Classic embedding models include RESCAL [48-50], NTN [12], TransE [8], DistMult [78], etc.

Relations of triplets could be divided into four categories according to their cardinalities, 1-1, 1-M, M-1 and M-M [8, 42]. The last three kinds of relations are usually referred to as complex relations. Predicting head on 1-1 and 1-M relations and predicting tail 1-1 and M-1 relations are called prediction on “side 1” of triples; predicting head on M-1 and M-M relations and predicting tail on 1-M and M-M relations are called prediction on “side M” of triples, which are obviously more difficult. Relations could also be classified as symmetric, antisymmetric, etc. It is not rare that simple assumptions (e.g., TransE) perform well on relations of 1-1 type. However, it usually goes beyond TransE’s capabilities to appropriately learn complex relationships. Most of subsequent models focus on integrating various role information brought by different relational attributes into entity representations for improvement. Lao et al. [33-34] and Takahashi et al. [59] verify the effectiveness of applying contextual information contained in relationship paths to KBC, demonstrating the value of relational features from another perspective. However, exquisitely designed models with stronger expressivity often come with higher computational costs. In ConvE [15] a parameter-efficient convolutional neural network (CNN) is employed to extract merely local patterns of triplets, yet yielding state-of-the-art (SOTA) performance.

Our initiative is to integrate diverse semantics brought by multiple relational attributes into entity representations with a simple but highly efficient CNN framework while following the idea of TransE, which is, if triplet (h, r, t) holds, the corresponding vector representations of its elements $v_h$, $v_r$, $v_t$ should comply with one rule: $v_h + v_r \approx v_t$. In IEAKBC, firstly entities / relations are embedded into low-dimensional vectors, i.e., each original triplet is represented in the form of a three-column, k dimensional matrix. Then the relation features are integrated into head / tail entities, forming feature-fused matrices of the same size used as another input channel for convolution; afterwards local / global patterns and semantic features are extracted in CNN, and finally the score is computed to verify the validity of the triplet.

We perform link prediction and triple classification tasks on benchmark datasets, making comparison between IEAKBC and several mainstream models.

In addition, IEAKBC is applied to a search personalization problem with the objective that ranking orders of search results could meet users’ personal preferences accurately.

In Section 2 relevant work is discussed, in Section 3 IEAKBC model is proposed, in Section 4 experimental results are presented and analyzed, in Section 5 application of IEAKBC in the personalized search system is discussed, and in Section 6 we conclude and discuss future plans.
2. Related Work

Multiple literature surveys [13, 45-46, 54, 70] have been conducted on knowledge graph embedding models. Such models based on latent features learn low-dimensional representations of nodes and edges in KGs and can be roughly divided into three categories: tensor / matrix factorization based approaches, translation / rotation based methods and models that employ neural networks.

2.1. Tensor / Matrix Factorization Based Approaches

Such approaches utilize semantic similarities. RESCAL, a bilinear model computes the outer product of head / tail entity and relation matrix with Alternating Least Squares (ALS) updating entities and relations alternatively. TATIE [21] adds two way interactions to RESCAL and brings forward an idea that entity representations could be relation specific. Subsequent models expect to acquire a balance between complexity, performance and scalability. DistMult inherits the merits of NTN and TransE to represent relations as diagonal matrices and entities vectors, taking Hadamard product of entity / relationship embeddings as the triplet score. However, since the result is not affected by the order of parameters, DistMult could not identify anti-symmetric relational patterns. ComplEx [64] disables the commutativity by embedding entities / relations into complex spaces. DistMult and ComplEx could be regarded as dimension reduction versions of RESCAL. Other SOTA models include HolE [47], SimplE [28], ANALOGY [38], Tucker [63], etc.

2.2. Translation / Rotation Based Methods

Translation / rotation based methods use distance based score functions. SE [9, 6] takes the idea that if one triplet holds, the head mapping should be close to the tail vector in the relation specific subspace. However, the correlation between entities and relations is weak and costs of optimization on matrix projection are rather high.

TransE, which is designed based on the ideas that hierarchical relationships could be naturally represented by translations and in word embedding studies some 1-1 relationships may be represented by translations in embedding spaces as well, maps the relations as translational vectors and uses a margin-based ranking criterion with a dissimilarity scoring function $d = \|v_h + v_r - v_t\|$, in which $\|\|$ is the L1 or L2 norm for $\cdot$. Stochastic Gradient Descent (SGD) is employed for parameter updating iteratively. Local features of triplets are preserved in the mapping of the same dimension of vectors. TransE featuring effective translations and model simplicity often performs better than SE despite its lower expressivity. However, TransE is inappropriate for learning complex relationship types due to the convergence problem, i.e., entity representations tend to be similar or even identical even though the discrepancy of their semantics could be significant. TransE is widely applied for embedding initialization. For instance, knowledge validity is often constrained by time and Jiang et al. [27] learns the temporal features of relationships with TransE. Lin et al. [36] and Luo et al. [40] further improve the model expressivity by combining semantic information from relation paths with TransE.

Variants of TransE lay stress on embedding mechanisms. TransH [71] maps entities to relation specific hyperplanes to embody their role differences. In TransR [37] the hyperplanes are replaced with relation specific matrices and the semantic spaces of entities / relations are separated to enhance expressivity. STransE [43] further addresses the problem that head / tail entities of different attributes still share the same relational projections. There are more such models [76].

However, higher expressivity is accompanied by increasing number of parameters [80]. TransD [25] decomposes the projection matrices into vectors for simplification while TranSparse [26] introduces sparse matrix to address the imbalance in head / entity proportions.

Related models include TransM [18], ManifoldE [75], FT [20] and TransA [74], improving expressivity by relaxing the constraints imposed on distance based scoring functions. TorusE [17], RotatE [58] and CrossE [81] employ a torus, rotations and composite operations to improve expressivity respectively. TransG [22] and KG2E [24] map entities and relations into to random variables according to their uncertainties.

2.3. Models with Neural Networks

NTN takes the strengths of multiple models, in which head / tail entity representations are concatenated as input and bilinear tensors are employed to replace the
traditional linear transformation layer. NTN holds
the strongest expressivity in almost all models. How-
ever, with high computational costs it is difficult to ap-
propriately train NTN model, over-fitting frequently
spotted especially in sparse or small KGs. ProjE [57]
could be regarded as a simplified NTN.

In recent years, CNN, originally designed for com-
puter vision [65] with less parameters than fully
connected neural networks, is in the spotlight of Nat-
ural Language Processing (NLP) [30, 56]. ConvE is
the first to apply CNN to knowledge representation.
ConvE argues that a padded two-dimensional (2D)
convolution on 2D reshaping of concatenated \(v_h\) and
\(v_r\) could model more interactions and extract more
features than previous models without increasing the
embedding size or number of parameters, thus lower-
ing probability of memory overflow. ConvE is also
considered to be more expressive than HoIE, since it
captures non-linear features. The authors find that fil-
ters with smaller sizes could better extract local pat-
terns but also report predictive accuracy degradation
with one-dimensional (1D) convolution. Presumably,
it is related to the lack of relational attribute integra-
tion. Unintuitively the convolving across reshaped
embeddings promotes such integration [2], while 1D
convolution fails to do so. In addition, ConvE does not
observe the valuable transitional characteristic em-
ployed by TransE and its multiple variants.

There are several studies aiming at simplifying
ConvE with 1D convolution, including HypER [2] and
ConvKB [42], the latter hoping to improve ConvE by
generalizing transitional characteristics, i.e., by cap-
turing global relationships among same dimensional
entries of vector representations of triplet elements.
While there’s obvious improvement on specific KBC
tasks, performance inconsistency is observed, for
which our hypothesis is as discussed above: lack of re-
lationlal integration. While 1D convolution takes care
of the global dependencies, in ConvKB entities and
relations are represented separately, leading to the
loss of various properties of entities pairing with dif-
ferent relations. CapsE [44], a recent study, addresses
this problem by employing a capsule layer to process
the output from the hidden layer. SACN [55] also ex-
tends ConvE by learning relation path representa-
tions while retaining translational features.

Since complex methods are potentially subject to ei-
ther over-fitting or under-fitting, we hope to bring
forward a simple but effective solution that could
achieve trade-offs between accuracy and scalability
for multi-relational domains.

We consider the multiplicative similarity calculation
employed by DistMult and similar models are essen-
tially a way to extract potential semantics; on the
other hand, feature extraction based on transitional
distance is highly efficient and adept at dealing with
sparse datasets while solutions to the convergence
problem adopted by multiple successors to TransE
are in nature also extracting semantic attributes. We
hope to apply this idea to our model.

The CNN framework could extract more non-linear
features than shallow models [15]. However, com-
pared to the 2D kernels with larger receptive fields,
more 1D kernels with smaller sizes may better recog-
nize latent patterns and reduce computational costs.

Comprehensively speaking, our initiative is to adopt
a CNN framework with 1D convolution, succeed the
idea of transitional constraint and combine embed-
dings with relation specific integration so as to ad-
dress the problem of semantic loss, stabilizing model
performance while keeping the computational cost
reasonable in real world scenarios.

3. Integrated Embedding Approach
for Knowledge Base Completion
(IEAKBC)

In KB \(\mathcal{G}\), triplets denoted as \((h, r, t)\) refer to facts con-
taining semantic information, where \(h, t \in \mathcal{E}\), \(r \in \mathcal{R}\)
and \(\mathcal{E}, \mathcal{R}\) refer to a set of entities and relations, respec-
tively. The objective of model design is to find a rea-
sonable score function to measure the plausibility or
implausibility for triplets.

Here we use \(k\) to represent the dimensionality of
embeddings of entities and relations produced by
TransE, so the original triplet could be displayed in
the form of a matrix \(A = [v_h, v_r, v_t] \in \mathbb{R}^{k \times 3}\), and \(A_i \in \mathbb{R}^{1 \times 3}\)
means the \(i\)th row of \(A\). Representations of entities
are independent of their position as well as combina-
tion with different relations so information is propa-
gated among multiple triplets, i.e., global dependency
is retained while local patterns are kept in different
entries of entity and relation representations.

Our core feature is an operation named relational
fusion for incorporating relational features into em-
beddings of head and tail entities so as to retain the integrity of triplets, i.e., to keep the complex pairing condition between entities and relations. The formula is as follows, in which $v'_h$ is the head entity representation after relational fusion, $v_h$ is the original $k$-dimensional vector produced by TransE. $\cdot$ means dot product operation, $w_j$ refers to parameter vectors gained through learning processes in a feedforward neural network, $v_j$ represents the original $k$-dimensional relation vector and $b_j$ is a bias term. For tail entity representation, the process is alike. Set $v'_t = v_t$.

We hope this set of procedures could help to implant relational features into entities so that same entities with variant properties for combination with different relations could be expressed more affluently, covering all kinds of relations with higher expressivity.

$$v'_h = v_h \cdot (w_j \cdot v_j + b_j).$$

After the relational integration, we get $A'$, variation of $A$ as the second input channel with relation attributes embedded into entities, defined as follows.

$$A' = [v'_h, v'_t, v_t] \in \mathbb{R}^{4 \times 3}.$$  

(2)

Given $\omega \in \mathbb{R}^{1 \times 3}$ represents the filters operating on the two channels with multiple kernels allocated to each channel, and what $\omega$ does is to repeatedly go over each row of $A$ and $A'$, two channels of the input to generalize the transitional characteristics from embedding representations, to extract global relationships among the same dimensional entries of triplets, and to analyze the variations from entities with different properties due to changing relations. Feature map $v$ after convolution processes could be denoted as $v = [v_1, v_2, ..., v_k]$, formula listed as follows in which is a non-linear activation function such as sigmoid and $b_j$ is a shared bias term while parameters in kernels are different. Here we do not differentiate between $A_m$ and $A_m^*$ any more, since they are regarded as two channels processed parallel

$$v_j = g(\omega \cdot A_m + b_j).$$

(3)

Let $\omega$ and $\tau$ be the kernels of each channel and the number of such kernels, i.e., $\tau = [\bar{\ell}]$. Therefore, for each channel there will be $\tau$ feature maps that are concatenated into one single vector $\in \mathbb{R}^{2 \times 1}$ respectively and processed by a shared weight vector $w_j$ via a dot product to generate scores for triplets. Formally, our score function $f$ could be defined as follows, with $\ast$ denoting convolutional operations and ‘concat’ the concatenation operation. Finally, scores from different channels are combined together to judge the validity of triplets. The whole process is shown in Figure 1.

**Figure 1**
Process of CNN in IEAKBC
\[ f(h, r, t) = \text{concat}(g([v_h, v_r, v_t] \cdot \Omega)) \cdot w_2. \] (4)

Loss function is defined as follows, in which

\[ l_{(h,r,t)} = \begin{cases} 1, & (h,r,t) \in \mathcal{G} \\ -1, & (h,r,t) \in \mathcal{G}' \end{cases}, \quad \mathcal{G}' \text{ refers to the set of invalid triplets created by replacing entities of valid triplets in } \mathcal{G}, \quad \text{and } \theta \text{ denotes the embeddings and parameters acquired by learning processes.} \]

\[ L = \sum_{(h,r,t) \in \mathcal{G} \cup \mathcal{G}'} \log(1 + \exp(l_{(h,r,t)} \cdot f(h,r,t))) + \frac{\lambda}{2} \| \theta \|^2. \] (5)

The learning process of IEAKBC is displayed in Algorithm 1; space complexity is \( O(n_1 k + n_2 k) \), same as TransE and ConvE while time cost is theoretically within the same order of magnitude of that with TransE.

**Algorithm 1:** Parameter Optimization for IEAKBC

Input: KB \( \mathcal{G} \), entity set \( \mathcal{E} \), relation set \( \mathcal{R} \), embedding dimension \( k \), kernels \( \Omega \), batch size \( b \), weight \( w_2 \), regularizer \( \lambda \), pre-trained embeddings produced by TransE

1. Initialize \((v_h, v_r, v_t)\)
   // Using a truncated normal distribution or using \([0.1, 0.1, 0.1]\)
2. Initialize \( \omega \)
3. \( w_2 \leftarrow \text{uniform}(\frac{-\sqrt{6}}{\sqrt{k \times b + 1}}, \frac{\sqrt{6}}{\sqrt{k \times b + 1}}) \)
4. For \( i = 1, 2, ..., n \), // \( n \) denotes the upper limit of epochs
5. For \( j = 1, 2, ..., \lceil \frac{|\mathcal{G}|}{b} \rceil + 1 \) // \(|\mathcal{G}|\) denotes the number of triplets
6. Sample(\( \mathcal{G}, b \))
7. InvalidBatch \( = \emptyset \)
8. For each triplet in Sample(\( \mathcal{G}, b \))
9. InvalidSample(\( / \) sample corrupted triplets denoted as \((h', r, t')\))
10. InvalidBatch \( \leftarrow \text{InvalidBatch} \cup (h', r, t') \)
11. Batch \( \leftarrow \text{Sample}(\mathcal{G}, b) \cup \text{InvalidBatch} \)
12. For each triplet in \( \text{Batch} \)
13. \( f(h, r, t) = \text{concat}(g([v_h, v_r, v_t] \cdot \Omega)) \cdot w_2 \)
14. Compute \( l_{(h,r,t)} \)
15. Compute gradient of \( \nabla L_{\text{Batch}} \)
16. Update embeddings, \( w_2 \), and \( \Omega \) with regard to \( \nabla L_{\text{Batch}} \)

To summarize, IEAKBC combines global relationships with relational attributes with a CNN framework, aiming at retaining triplet element correlations as much as possible. Therefore, it could be seen as an extension of TransE, a variant of ConvE with 1D convolution.

### 4. Experiments and Analysis

**Tasks:** Two common KBC tasks including link prediction and triple classification are used to measure model performance.

**Datasets:** Four benchmark datasets are used, FB15k-237 and WN18RR for link prediction, WN11 and FB13 for triple classification. All of the four datasets are specially designed for multi-relational tasks and frequently adopted in related studies.

WN18RR and WN11 are extracted from WordNet, a lexical KG in English in which each entity denotes a synset consisting of several words and corresponds to a distinct word sense. Relationships in WordNet are defined as conceptual-semantic and lexical relations. In WN11, in the case of synsets containing multiple words, only the most frequent one is picked.

FB15k-237 and FB13 derive from Freebase, a large KG of general world facts. Triples in FB13 come from the People domain, containing 13 relations, six of which are removed from the test set owing to high predicting difficulty.

Test leakage\(^1\) in FB15k and WN18 discussed by Toutanova et al. [62] and Dettmer et al. [15] leads to the birth of more robust and challenging datasets, FB15k-237 and WN18RR. On the other hand, WN11 and FB13 do not suffer from such problem. WN18RR is denser than FB15k-237 (the average number of entity pairs per relation is bigger). FB15k-237 and WN18RR contain only positive triples while WN11 and FB13 include negative samples.

Details of datasets are shown in Table 1.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>#E</th>
<th>#R</th>
<th>Training Set</th>
<th>Validation Set</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>FB15k-237</td>
<td>14541</td>
<td>237</td>
<td>272115</td>
<td>17535</td>
<td>20466</td>
</tr>
<tr>
<td>WN18RR</td>
<td>40943</td>
<td>11</td>
<td>86835</td>
<td>3034</td>
<td>3134</td>
</tr>
<tr>
<td>WN11</td>
<td>38696</td>
<td>11</td>
<td>112581</td>
<td>2609</td>
<td>10544</td>
</tr>
<tr>
<td>FB13</td>
<td>75043</td>
<td>13</td>
<td>316232</td>
<td>5908</td>
<td>23733</td>
</tr>
</tbody>
</table>

\(^1\) Models with simple rules could achieve high scores on specific datasets with high proportions of inverse relations.
4.1. Link Prediction

**Task:** The goal is to infer a missing head / tail entity given a relation and the other entity, denoted as (?, r, t) or (h, r, ?), results of which are made by ranking the scores of candidate triples produced by various score functions.

**Models:** DistMult, ComplEx, ConvE, TransE and ConvKB are used as baselines for comparison with IEAKBC.

**Evaluation Protocol:** The valid test triple and corrupted triples are ranked together and we hope correct answers can be ranked before incorrect ones. MR (mean rank), MRR (mean reciprocal rank) and Hits@10 (the proportion of correct entities ranked in the top 10, which is commonly used in downstream tasks such as personalized search) are taken as evaluation metrics. Lower MR, higher MRR and Hits@10 indicate better performance. Filtered setting protocol is used, i.e., false-negative triples that already appear either in the training, validation or test set are not taken into account.

**Training Protocol:** Following Bordes et al. [7], entity replacement is performed to create corrupted triplets. The Bernoulli Trick (BT) [71] is employed for all models except ConvE (for ConvE the 1-N strategy as a feature procedure is kept) to create corrupted triples while reducing the probability of generating false-negative samples. Negative sampling is performed at runtime for each batch. Training is of up to 500 / 2000 epochs (for different models) and early stopping is executed if MRR improvement of last 10 epochs is less than 0.01. Models with highest Hits@10 scores on the validation set are implemented on the test set.

**Hyper-parameters Tuning:** Grid search is employed to find optimal hyper-parameters; the parameter pool is listed as follows, some of which are model specific and with annotations. Optimal performance is acquired with hyper-parameters listed in Table 2.

**Embedding Size** ∈ {50, 100, 200, 500, 1000}

**Loss Function** ∈ {MR (margin ranking), CE (cross entropy)}

**Optimizer** ∈ {SGD, AdaGrad [16], Adam [31]}

**Batch Size** ∈ {64, 128, 256, 512, 1024}²

**Initial Learning Rate (SGD)** ∈ {1e⁻⁵, 1e⁻⁴, 5e⁻³, 1e⁻², 1e⁻¹}

**Initial Learning Rate (AdaGrad)** ∈ {0.002, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.4, 0.5}

**Initializer in PyTorch** ∈ {normal, uniform, xavier_normal, xavier_uniform, truncated_normal, [0.1, 0.1, -0.1]}³

<table>
<thead>
<tr>
<th>Hyper-parameters</th>
<th>FB15k-237</th>
<th>WN18RR</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TransE</strong></td>
<td><strong>IEAKBC</strong></td>
<td><strong>TransE</strong></td>
</tr>
<tr>
<td>Embedding Size</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Negative Sampling</td>
<td>BT</td>
<td>BT</td>
</tr>
<tr>
<td>Loss Function</td>
<td>MR</td>
<td>MR</td>
</tr>
<tr>
<td>Optimizer</td>
<td>SGD</td>
<td>Adam</td>
</tr>
<tr>
<td>Batch Size</td>
<td>128</td>
<td>256</td>
</tr>
<tr>
<td>Learning Rate</td>
<td>0.0005</td>
<td>0.0005</td>
</tr>
<tr>
<td>Regularizer</td>
<td>L1</td>
<td>L2</td>
</tr>
<tr>
<td>Initializer</td>
<td>xavier_normal</td>
<td>[0.1, 0.1, -0.1]</td>
</tr>
<tr>
<td>margin γ</td>
<td>1</td>
<td>-</td>
</tr>
<tr>
<td>kernel number</td>
<td>-</td>
<td>100</td>
</tr>
</tbody>
</table>

Due to experimental limitations, the batch size is set to between 64 and 1024 for fear of memory overflow.

³ trun is short for truncated_normal
Initial Learning Rate (Adam) ∈ \{1e^{-6}, 5e^{-6}, 1e^{-5}, 5e^{-5}, 
1e^{-4}, 5e^{-4}, 1e^{-3}, 1e^{-2}\}
Regularizer ∈ \{L1, L2, L3\}\{32\}\}
Margin ∈ \{1, 2, 3, 5, 7, 10\} (for TransE)
Kernel Number τ ∈ \{100, 200, 300, 400, 500\} (for IEAKBC)
Others: Maximum Epoch=500 (for IEAKBC); Maximum Epoch=2000 (for TransE); Regularizer Coefficient=0.001 (for IEAKBC)

4.1.1. Overall Results
Experimental results are shown in Table 3, where [*] means that data are taken from related literature, since previous experiments are carried out on the same datasets. Results of DistMult, ComplEx and ConvE are from Dettmers et al. [15]. Results of ConvKB are from Nguyen et al. [42]. We use our own implementation of TransE and embeddings produced by TransE are employed as initialization embeddings for IEAKBC.

In a nutshell, IEAKBC secures the best Hits@10 score on FB15k-237 and No. 2 at MRR, while achieving the best results at MRR and Hits@10 on WN18RR, No. 2 at MR.

MRR scores of DistMult and ComplEx on WN18RR are rather high, since bilinear models are good at generalizing semantic similarity from denser datasets with multiplicative operations. However, the MRR and Hits@10 scores of these two models on FB15k-237 are not so good, since the number of entity pairs per relation is relatively small in sparse datasets and these models may not learn sufficient information.\(^4\)

On the contrary to bilinear models, TransE excels at handling sparse datasets, achieving better performance at MRR and Hits@10 on FB15k-237. Nevertheless, TransE suffers an obvious degradation at MRR on WN18RR due to the representation restrictions.

Although the neural network mechanism is still a black box, ConvE obtains solid scores at all metrics except at MR on WN18RR and we consider its advantages over DistMult and ComplEx at MRR and Hits@10 on FB15k-237 are due to the 2D convolution extracting richer relation specific features. However, the 2D kernel size (3×3) along with complicated score function may bring about a relatively high computational cost.

On FB15k-237, there is comprehensive improvement at all metrics for ConvKB compared with TransE, since it takes embeddings produced by TransE as input and further generalizes transitional relationships in a CNN framework. However, when it comes to datasets with high proportion of complex relations, ConvKB also suffers a drop at MRR on WN18RR, since it does not solve the problem of entity representations tending to be close.

On FB15k-237, IEAKBC narrowly loses to ConvE, DistMult and ConvKB at MR but obviously beats TransE with an improvement of 78. At MRR which

\(^4\) We also notice that DistMult performs better than ConvKB and IEAKBC at MR on FB15k-237, so one of our plans is to integrate merits of bilinear models into IEAKBC.

\(^5\) The best performance is marked in bold while No.2 in italic font with underlines.

<table>
<thead>
<tr>
<th>Method</th>
<th>FB15k-237</th>
<th>WN18RR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MR</td>
<td>MRR</td>
</tr>
<tr>
<td>DistMult[*]</td>
<td>254</td>
<td>0.241</td>
</tr>
<tr>
<td>ComplEx[*]</td>
<td>339</td>
<td>0.247</td>
</tr>
<tr>
<td>ConvE[*]</td>
<td>246</td>
<td>0.316</td>
</tr>
<tr>
<td>TransE</td>
<td>343</td>
<td>0.297</td>
</tr>
<tr>
<td>ConvKB[*]</td>
<td>257</td>
<td>0.396</td>
</tr>
<tr>
<td>IEAKBC</td>
<td>265</td>
<td>0.374</td>
</tr>
</tbody>
</table>

| Table 3
Results of Link Prediction on FB15k-237 and WN18RR\(^5\) |
is a more stable metric, IEAKBC outperforms all the other models except ConvKB with a wide margin implying the model retains transitional constraint and relational integration could extract more features. IEAKBC’s best performance is at Hits@10 with about 10% increase compared to ConvKB, let alone ConvE and TransE, indicating the probability of applying IEAKBC to tasks such as search personalization.

On WN18RR, the denser dataset, IEAKBC achieves a similar result to ConvKB at MR, steadily outperforming some other models. At MRR, IEAKBC is on par with ConvE, solving the problem of performance slump confronting TransE and ConvKB who rely solely on transitional characteristics, and also better than DistMult and ComplEx indicating the feasibility of extracting latent features by fusing relational attributes instead of computing entity similarity. At Hits@10 IEAKBC achieves the highest score again, slightly overpassing ConvKB. To summarize, IEAKBC proves itself a stable and effective model.

In further experiments afterwards, TransE and ConvKB are selected as baselines.

### 4.1.2 Results for Predicting Head / Tail on FB15k-237

The core initiative of IEAKBC is to enhance the modeling capability for complex relations, so Hits@10 scores for predicting head and tail entities respectively for each relation category on FB15k-237 are computed and shown in Figures 2-3 so as to see whether

![Figure 2](image1.png)

**Figure 2**

Hits@10 on FB15k-237 for Predicting Head

![Figure 3](image2.png)

**Figure 3**

Hits@10 on FB15k-237 for Predicting Tail
the improvement of IEAKBC over TransE and ConvKB is related to its performance when handling complex relations. A common definition for relation classification [8, 42] is adopted.

We can see that on “side 1” prediction all models perform well, since in such task one or more entities point to just one entity, making the latter easy to identify. Gaps get wider on “side M” prediction, in which IEAKBC obtains the laurels on all the four tasks, i.e., M-1 and M-M for predicting head, 1-M and M-M for predicting tail. Since the main difference between IEAKBC and ConvKB lies in fusing relational attributes into entity embeddings, such results indicate that relational integration indeed helps with KBC tasks.

4.1.3. Results with Regard to Each Relation on WN18RR

To confirm our judgement with more evidence, link prediction task is pushed further; Hits@10 and MRR scores with regard to all 11 relations on WN18RR are counted and shown in Figures 4-5 to see where the strength of IEAKBC lies, in which the order of relations is organized according to their proportions of all triplets in ascending order (see the broken line). IEAKBC achieves the highest Hits@10 scores on 7 out of 11 relations (including on par with TransE / ConvKB on 2 relations).

At MRR, it is obvious that IEAKBC maintains previous performance on challenging relations including has_part (1-M), member_meronym (1-M) and hypernym (M-1), while TransE and ConvKB suffer a huge degradation on these relations, of which the latter two constitute a substantial fraction of all test triplets, leading to the decrease of whole MRR scores for TransE and ConvKB. Such performance is in accord with results of predicting tail for 1-M relations and predicting head for M-1 relations on FB15k-237. On M-M complex relations such as similar_to, verb_group, also_see and derivationally_related_form, since symmetric patterns could boost performance to a great extent, all the models achieve similar scores.

Results shown in Figures 4-5 are consistent: IEAKBC works well on prediction for complex relations, evidence that taking into consideration various properties of entities pairing with different relations makes sense.

Figure 4

Hits@10 on WN18RR with Regard to Each Relation

![Figure 4: Hits@10 on WN18RR with Regard to Each Relation](image)
4.2. Triplet Classification

**Task:** The aim is to predict whether or not a given triplet is valid, i.e., this is a **binary classification task.** A threshold for each relation is introduced to help judge the validity of unseen triplets. The triplet is considered valid only if the triplet dissimilarity score is lower than \( q \). According to Chen D et al. [12], \( q \) is achieved by maximizing the micro-averaged classification accuracy on the validation set.

**Models:** ConvE, DistMult, TransE and ConvKB are picked as representatives of their kind for comparison and we also refer to data from previous studies for comparison.

**Evaluation Protocol:** **Classification accuracy** is employed as the metric.

**Training Protocol:** Same as in the link prediction task, negative sampling with Bernoulli Trick (BT) is kept except for ConvE (1-N). Embedding representations initialized by TransE are employed as input to ConvKB and IEAKBC.

**Hyper-parameters Tuning:** We still use grid search to find optimal hyper-parameters, the parameter pool is similar to that in the link prediction task with supplements listed below. Optimal hyper-parameters are listed in Table 4. Due to limited computation resources, we did not combine word embedding approaches [61, 52-53]. Initializer in TensorFlow \( \in \{ \text{normal, uniform, truncated normal, } [0.1, 0.1, -0.1] \} \)

- Embedding Layer Dropout / Feature Map Layer Dropout / Projection Layer Dropout \( \in \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5\} \) (for ConvE)
- Label Smoothing \( \in \{0.05, 0.1, 0.15, 0.2\} \) (for ConvE)
- Kernel Number \( \in \{100, 200, 300, 400, 500\} \) (for ConvE, ConvKB and IEAKBC)
- Maximum Epoch=500 (for ConvE, ConvKB and IEAKBC)
- Maximum Epoch=2000 (for simple models including DistMult and TransE)
- Regularizer Coefficient=0.001 (for ConvKB and IEAKBC)
For ConvE, other optimal hyper-parameters include embedding dropout 0.2, feature map dropout 0.2, projection layer dropout 0.4, label smoothing 0.1.

Table 4
Hyper-parameters for Optimal Performance in Triplet Classification Task

<table>
<thead>
<tr>
<th></th>
<th>WN11</th>
<th>FB13</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedding Size</td>
<td>50</td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>Negative Sampling</td>
<td>BT</td>
<td>1-N</td>
<td>BT</td>
</tr>
<tr>
<td>Loss Function</td>
<td>MR</td>
<td>CE</td>
<td>CE</td>
</tr>
<tr>
<td>Optimizer</td>
<td>SGD</td>
<td>AdaGrad</td>
<td>SGD</td>
</tr>
<tr>
<td>Batch Size</td>
<td>128</td>
<td>128</td>
<td>512</td>
</tr>
<tr>
<td>Learning Rate</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
</tr>
<tr>
<td>Regularizer</td>
<td>L1</td>
<td>L2</td>
<td>L2</td>
</tr>
<tr>
<td>Initializer</td>
<td>normal</td>
<td>xavier_normal</td>
<td>uniform</td>
</tr>
<tr>
<td>margin ( \gamma )</td>
<td>7</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>kernel number</td>
<td>-</td>
<td>100</td>
<td>200</td>
</tr>
</tbody>
</table>

Table 5
Accuracy for Triplet Classification (%)

<table>
<thead>
<tr>
<th>Models</th>
<th>WN11</th>
<th>FB13</th>
<th>Avg.</th>
</tr>
</thead>
<tbody>
<tr>
<td>NTN[*]</td>
<td>70.6</td>
<td>87.2</td>
<td>78.9</td>
</tr>
<tr>
<td>ConvE</td>
<td>83.5</td>
<td>86.2</td>
<td>84.9</td>
</tr>
<tr>
<td>DistMult</td>
<td>84.7</td>
<td>89.8</td>
<td>86.8</td>
</tr>
<tr>
<td>TransE</td>
<td>86.2</td>
<td>87.1</td>
<td>86.7</td>
</tr>
<tr>
<td>TransD[*]</td>
<td>86.4</td>
<td>89.1</td>
<td>87.8</td>
</tr>
<tr>
<td>TransSparse-S[*]</td>
<td>86.4</td>
<td>88.2</td>
<td>87.3</td>
</tr>
<tr>
<td>TransSparse-US[*]</td>
<td>86.8</td>
<td>87.5</td>
<td>87.2</td>
</tr>
<tr>
<td>ConvKB[*]</td>
<td>87.6</td>
<td>88.8</td>
<td>88.2</td>
</tr>
<tr>
<td>ConvKB[our results]</td>
<td>86.3</td>
<td>88.8</td>
<td>87.6</td>
</tr>
<tr>
<td>IEAKBC</td>
<td>85.9</td>
<td><strong>89.4</strong></td>
<td>87.7</td>
</tr>
</tbody>
</table>

Results are shown in Table 5, where [*] means that data are taken from related publications. Results of NTN, TranSparse-S and ConvKB are from Nguyen et al. [42], results of TransD are from Ji et al. [25], and results of TranSparse-US are from Chang et al. [11]. Implementation of DistMult and ConvE are from Dettmers et al. [15]. The implementation of ConvKB is from Nguyen et al. [42]. On the denser dataset FB13, DistMult and IEAKBC beat other competitors, indicating higher generalization capability. IEAKBC achieves higher accuracy than TransE and ConvKB (2.3% and 0.6%, respectively), which is in accord with the results of the link prediction task on WN18RR. On WN11, IEAKBC loses to several models with a narrow margin. Our hypothesis is while simultaneously considering variant relations and transitional characteristics brings benefits, such combination of multiple criteria may lead to confusion in specific scenarios. The performance degradation of ConvE may come from similar cause. Improvement is still under consideration.
On average, ConvKB (the original results) and TransD outperform other models. IEAKBC loses to TransD with a gap of 0.1%, yielding better performance than multiple classic models (In our experiments, IEAKBC performs marginally better than ConvKB).

We notice that ConvE with 2D convolution seems to lag behind on this task, while the average score of DistMult is relatively low. On the other hand, multiple successors to TransE perform well on this task, implying the significance of retaining transitional characteristics.

Results of TransE and ConvKB on FB13 are again used as baselines to discuss the impact of relational integration on model performance for different kinds of relations, as is shown in Figure 6. Proportions of all relations in the test set are shown as a broken line. All relations could be classified as M-1 type except institution and profession belong to M-M type. IEAKBC advances ConvKB on both M-M relations and only loses on 2 out of 5 M-1 relations, religion and nationality, outperforming TransE on all the seven relations, consistent with results in the link prediction task.

5. Application in Personalized Search

**Task:** In this part, we hope to apply IEAKBC to practical scenarios, so the search personalization problem is consulted, since personalized search has become in the spotlight of search engine business in recent years [10, 14, 39, 68, 79]. Related studies exploit user behavior data [73], building user profiles so as to tailor search results accordingly with the idea of relevant items coming first. In such applications, query, relevant user and document are usually put together and constitute triple-like structure (query, user, document) [66], representing lots of M-M relationships and implying the user’s interest on the document under one specific query. To such structure knowledge representation approaches could be applied and evaluated. To be specific, the idea is to re-rank the returned documents with these approaches and expectations are more relevant documents should be ranked higher.

**Dataset and Pre-processing:** The dataset is a subset of a query log repository from a large commercial search engine, which originally includes 1166 randomly chosen users and 489384 triplets, with the time span from July 1st to July 28th, 2012. The same source data has been employed by Nguyen, et al. [42] and Vu, et al. [67, 69]

One search entry includes a masked user ID, a query, top-10 returned URLs (documents) with ranking orders from the search engine, clicking records, dwell time and the relevant-or-not label to documents judged by SAT criteria (either a click with a dwell time of at least 30 seconds or the last result click in a search session is seen as a SAT click. Related document is labeled relevant and all the other documents in the same query are labeled as irrelevant). The rank order of relevant labeled documents is deemed as the ground truth to evaluate the search performance before and after re-ranking.

We try to keep the same setting as Vu et al. [66] as much as possible for comparable results, so we employ the same short-term user profile protocol [3], remove the queries with empty relevant label set, discard domain-related queries such as Twitter or Facebook and construct the validation set and test set with latest log entries while the training set with remaining entries. Statistical information of the dataset is shown in Table 6. All 7796 triplets are valid.

**Embedding Initialization:** The same process used by Vu et al. [66] is followed.

**Evaluation Protocol:** Triplet score is computed and organized in the ascending order to re-rank the top 10 documents returned by the search engine. MRR, Hits@1 and Hits@3 are adopted as performance metrics with higher values indicating better performance. (Hits@10 does not work here for we only use top 10 returned documents).

**Models and Hyper-parameters Tuning:** TransE and ConvKB are employed as baselines. The implementation of ConvKB is from Nguyen et al. [42] and we keep the recommended parameters unchanged. For IEAKBC, we follow the same training protocol and hyper-parameters tuning in the link prediction task. Optimal hyper-parameters are listed in Table 7.

In addition, results from related literature [42] are introduced for comparison marked with [*], including: the original rank returned by the search engine (denoted as SE), CI [60] (a personalized method using clicking records), SP [69] (another personalized method employing session-based user profiles).
is from Nguyen et al. [42] and we keep the recommended parameters returned by the search engine.

Evaluation Protocol:
The same process used by Vu et al. [66] is followed. We try to keep the same setting as Vu et al. [66] as much as possible for comparable results, so we employ the same short-term user profile protocol [3], remove the queries with empty relevant label set, discard domain-related queries such as Twitter or any other social media and all the other documents in the same query are labeled as irrelevant. The rank order of relevant labeled documents is deemed as the ground truth to evaluate the search performance before and after re-ranking.

Table 6
Dataset Statistics for Personalized Search

<table>
<thead>
<tr>
<th>users</th>
<th>distinct queries</th>
<th>SAT clicks</th>
<th>distinct documents</th>
<th>Triplets</th>
<th>Training Set</th>
<th>Validation Set</th>
<th>Test Set</th>
</tr>
</thead>
<tbody>
<tr>
<td>106</td>
<td>5741</td>
<td>7796</td>
<td>31282</td>
<td>7796</td>
<td>5475</td>
<td>1141</td>
<td>1180</td>
</tr>
</tbody>
</table>

Table 7
Optimal Hyper-parameters for Personalized Search

<table>
<thead>
<tr>
<th></th>
<th>TransE</th>
<th>ConvKB</th>
<th>IEAKBC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Embedding Size</td>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Negative Sampling</td>
<td>BT</td>
<td>BT</td>
<td></td>
</tr>
<tr>
<td>Loss</td>
<td>MR</td>
<td>MR</td>
<td>CE</td>
</tr>
<tr>
<td>Optimizer</td>
<td>SGD</td>
<td>Adam</td>
<td>Adam</td>
</tr>
<tr>
<td>Batch Size</td>
<td>128</td>
<td>256</td>
<td>256</td>
</tr>
<tr>
<td>Learning Rate</td>
<td>0.005</td>
<td>0.0005</td>
<td>0.0001</td>
</tr>
<tr>
<td>Regularizer</td>
<td>L1</td>
<td>L2</td>
<td>L1</td>
</tr>
<tr>
<td>Initializer</td>
<td>normal</td>
<td>train</td>
<td>train</td>
</tr>
<tr>
<td>margin γ</td>
<td>5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>kernel number</td>
<td>-</td>
<td>500</td>
<td>200</td>
</tr>
</tbody>
</table>

Table 8
Performance Comparison for Personalized Search

<table>
<thead>
<tr>
<th>Model</th>
<th>MRR</th>
<th>Hits@1(%)</th>
<th>Hits@3(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SE[*]</td>
<td>0.559</td>
<td>38.5</td>
<td>-</td>
</tr>
<tr>
<td>CI[*]</td>
<td>0.597</td>
<td>41.6</td>
<td>-</td>
</tr>
<tr>
<td>SP[*]</td>
<td>0.631</td>
<td>45.2</td>
<td></td>
</tr>
<tr>
<td>TransE</td>
<td>0.648</td>
<td>50.2</td>
<td>82.7</td>
</tr>
<tr>
<td>ConvKB</td>
<td>0.733</td>
<td>58.6</td>
<td>85.5</td>
</tr>
<tr>
<td>IEAKBC</td>
<td>0.749</td>
<td>55.4</td>
<td>88.3</td>
</tr>
</tbody>
</table>

Results are shown in Table 8. All embedding models achieve better performance than traditional approaches CI and SP, implying applications of such models to personalized search tasks. Both ConvKB and IEAKBC perform better than TransE, indicating
better feature extraction capabilities. Compared with a similar model ConvKB, IEAKBC loses at Hits@1, but outperforms ConvKB at MRR and Hits@3. Similar to the accuracy score on WN11, we attribute the Hits@1 score to the relatively complicated architecture of IEAKBC over ConvKB while comprehensive results show that in a wider range relational integration brings more benefits.

6. Conclusion and Future Work

Our aim is to remove the representation limitations haunting TransE by integrating relational attributes into entity representations, extracting richer semantics with the help of highly expressive and efficient CNN framework while retaining the transitional characteristics. We also hope with 1D convolution the ConvE architecture could be simplified without performance degradation. Results of multiple tasks on benchmark datasets validate our model, verifying improvement especially on complex relations. Nevertheless, still space for improvement, and further work may include: combining the strength from bilinear models; drawing support from logic-rules-based reasoning; integrating information from relation path or temporal sequence; initializing embeddings with word vectors; testing model scalability and performance on larger, sparser real world datasets; adapting IEAKBC for other applications like sentiment recognition [29], etc.

Acknowledgement

This paper is sponsored by Fujian Provincial Educational and Scientific Project(FJJJKCG20-402)

References


## Appendix A

### Model Comparison

<table>
<thead>
<tr>
<th>Classification</th>
<th>Model</th>
<th>Score Function</th>
<th>Loss Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Translation</td>
<td>TransE [8]</td>
<td>$|v_h + v_r - v_t|_{L1/L2}$</td>
<td>$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{D}} \sum_{i \in \mathcal{G}} \gamma - d(v_h, v_r, v_t) + d(v_h, v_r, v_t)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\gamma &gt; 0$ is a margin hyper-parameter</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$d(v_h, v_r, v_t) = |v_h + v_r - v_t|_{L1/L2}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$[\cdot]$ denotes the positive part of $\cdot$</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$\mathcal{G}$ is the set of valid triplets while $\mathcal{G}'$ refers to the invalid triplets</td>
</tr>
<tr>
<td>Bilinear</td>
<td>DistMult [78]</td>
<td>$v_h^T W v_t$</td>
<td>$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{D}} \sum_{i \in \mathcal{G}} \max(0, 1 + s(h, r, t) - s(h, r, t))$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$W$ is a diagonal matrix in $\mathbb{R}^{k \times k}$</td>
<td>$s(h, r, t) = v_h^T W v_t$</td>
</tr>
<tr>
<td>Complex Space</td>
<td>ComplEx [64]</td>
<td>$\text{Re}(c_i \hat{c})$</td>
<td>$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{D}} \text{log}(1 + \exp(-\gamma \text{Re}(c_i \hat{c}))) + \lambda |\theta|^2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$c_i, \hat{c} \in \mathbb{C}^k$ are entity representations in complex space</td>
<td>${Y_{rs}}_{h(r,t) \in \mathcal{D}} \in {-1, 1}^H$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$c_i$ is a diagonal matrix in $\mathbb{C}^{k \times k}$</td>
<td>$\lambda$ is a L1/L2 regularizer</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\hat{c}$ is the conjugate of $c_i$</td>
<td>$\theta$ corresponds to the embeddings of entities and relations</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\text{Re}(c)$ denotes the real part of $c$, $c \in \mathbb{C}$</td>
<td>here $\Omega$ denotes a set of observed triples including both true and false facts and $|\cdot|$ denotes the number of all triplets</td>
</tr>
<tr>
<td>Neural Network</td>
<td>ConvE [15]</td>
<td>$f(\text{vec}(f(\text{concat}(v_h, v_r) * \Omega)))Wv_t$</td>
<td>$\mathcal{L} = -\frac{1}{N} \sum_{i \in \mathcal{D}} (t_i \cdot \text{log}(p_i) + (1-t_i) \cdot \text{log}(1-p_i))$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$v_h, v_r$ denote 2D reshaping of $v_h, v_r$ respectively</td>
<td>$p = \text{sigmoid}(f(\text{vec}(f(\text{concat}(v_h, v_r) * \Omega)))Wv_t)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>concat is short for concatenation operation</td>
<td>$t$ is the label vector, elements of which $\in [0, 1]$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$f$ is a non-linear function</td>
<td>$N$ refers to the number of all entities</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$*$ refers to a convolution operation</td>
<td>(in ConvE $v_h, v_r$ may be composed of different dimensions)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\Omega$ denotes a set of kernels</td>
<td>$\text{vec}(\cdot)$ is a reshaping function</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$W$ is a parameter matrix for linear transformation</td>
<td>transforming tensors into vectors</td>
</tr>
<tr>
<td>CNN + Translation</td>
<td>ConvKB [42]</td>
<td>$\text{concat}(\text{ReLU}(\text{concat}(v_h, v_r) * \Omega))) \cdot w$</td>
<td>$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{G}'} \text{log}(1 + \exp(l_{h,r,t} \cdot f(h, r, t))) + \frac{\lambda}{2} |\theta|^2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$w$ is the weight vector</td>
<td>$l_{h,r,t} = \begin{cases} 1, &amp; (h, r, t) \in \mathcal{G}' \ -1, &amp; (h, r, t) \in \mathcal{G}' \end{cases}$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$f(h, r, t)$ is the score function</td>
<td>$f(h, r, t)$ is the score function</td>
</tr>
<tr>
<td></td>
<td>IEAKBC</td>
<td>$\text{concat}(g(\text{concat}(v_h, v_r) * \Omega))) \cdot w$</td>
<td>$\mathcal{L} = \sum_{(h,r,t) \in \mathcal{G}'} \text{log}(1 + \exp(l_{h,r,t} \cdot f(h, r, t))) + \frac{\lambda}{2} |\theta|^2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$\theta$ denotes the embeddings and parameters acquired by learning processes</td>
<td></td>
</tr>
</tbody>
</table>

---

7 We try to unify the symbols with same meanings here.
Appendix B

Implementation of truncated_normal Distribution in PyTorch

Model Comparison:

def truncated_normal_(self, tensor, mean=0, std=1):
    size = tensor.shape
    tmp = tensor.new_empty(size+(4,)).normal()
    valid = (tmp < 2) & (tmp > -2)
    ind = valid.max(-1, keepdim=True)[1]
    tensor.data.copy_(tmp.gather(-1, ind).squeeze(-1))
    tensor.data.mul_(std).add_(mean)
    return tensor

Appendix C

Embedding Initialization in Personalized Search

The same process used by Vu et al. [66] is followed to initialize embeddings for documents, queries and user profiles. First, a LDA topic model [4] from MALLET toolkit is trained with 200 topics only on the relevant documents. Then the model is employed to infer the probability distribution over topics for each document and the topic proportion vector of each document is taken as its embedding with the size $k=200$. In particular, the $z$th element ($z=1,2,\ldots,k$) of the vector embedding for document $d$ is denoted as: $v_{d,z} = P(z|d)$ and $P(z|d)$ is the probability of the topic $z$ given the document $d$.

Each query is also represented by a probability distribution vector over topics. Let $D_q = \{d_1,d_2,\ldots,d_n\}$ be the set of top $n$ ranked documents returned for a query $q$ ($n=10$). The $z$th element of the vector embedding for query $q$ is defined as $v_{q,z} = \sum_{i=1}^{n} \delta^i \lambda^d_{-i}$, where $\delta^i$ is the exponential decay function of $i$ which is the rank of $d_i$ in $D_q$, and $\delta$ is the decay hyper-parameter $(0 < \delta < 1)$. In our experiments we found that when $\delta$ is set to 0.7 model performance is optimal.

After the initialization of query and document embeddings, these representations are fixed during training for TransE, ConvKB and IEAKBC so as to avoid over-fitting. User embeddings are initialized with their clicking records under the temporal weighting scheme proposed by Vu et al. [69] with the idea that most recently clicked documents express more about the users’ current interest [3].

For further details like hyper-parameters tuning please consult Vu et al. [66].

---

http://mallet.cs.umass.edu/