To Copy Rather Than Memorize: A Vertical Learning Paradigm for Knowledge Graph Completion

Rui Li\textsuperscript{1*}, Xu Chen\textsuperscript{2†}, Chaozhuo Li\textsuperscript{3*†}, Yanming Shen\textsuperscript{1}, Jianan Zhao\textsuperscript{4}, Yujing Wang\textsuperscript{2}, Weihao Han\textsuperscript{2}, Hao Sun\textsuperscript{5}, Weihao Han\textsuperscript{5}, Qi Zhang\textsuperscript{5}, Xing Xie\textsuperscript{3}

\textsuperscript{1}Dalian University of Technology, \textsuperscript{2}Renmin University of China, \textsuperscript{3}Microsoft Research Asia, \textsuperscript{4}Université de Montréal, \textsuperscript{5}Microsoft

\texttt{xu.chen@ruc.edu.cn, cli@microsoft.com}

Abstract

Embedding models have shown great power in knowledge graph completion (KGC) task. By learning structural constraints for each training triple, these methods implicitly memorize intrinsic relation rules to infer missing links. However, this paper points out that the multi-hop relation rules are hard to be reliably memorized due to the inherent deficiencies of such implicit memorization strategy, making embedding models underperform in predicting links between distant entity pairs. To alleviate this problem, we present Vertical Learning Paradigm (VLP), which extends embedding models by allowing to explicitly copy target information from related factual triples for more accurate prediction. Rather than solely relying on the implicit memory, VLP directly provides additional cues to improve the generalization ability of embedding models, especially making the distant link prediction significantly easier. Moreover, we also propose a novel relative distance based negative sampling technique (ReD) for more effective optimization. Experiments demonstrate the validity and generality of our proposals on two standard benchmarks. Our code is available at https://github.com/rui9812/VLP.

1 Introduction

Knowledge graphs (KGs) structurally represent human knowledge as a collection of factual triples. Each triple \((h, r, t)\) represents that there is a relation \(r\) between head entity \(h\) and tail entity \(t\). With the massive human knowledge, KGs facilitate a myriad of downstream applications (Xiong et al., 2017). However, real-world KGs such as Freebase (Bollacker et al., 2008) are far from complete (Bordes et al., 2013). This motivates substantial research on the knowledge graph completion (KGC) task, i.e., automatically inferring missing triples.

As an effective solution for KGC, embedding model learns representations of entities and relations with pre-designed relation operations. For example, TransE (Bordes et al., 2013) represents relations as translations between head and tail entities. RESCAL (Nickel et al., 2011), DistMult (Yang et al., 2015) and ComplEx (Trouillon et al., 2016) model the three-way interactions in each triple. RotatE (Sun et al., 2019), QuatE (Zhang et al., 2019) and DualE (Cao et al., 2021) represent relations as rotations in different dimensions. Rot-Pro (Song et al., 2021) further introduces the orthogonal projection for each relation.

Essentially, embedding models learn structural constraints for every factual triple during the training period. For example, for each training triple \((h, r, t)\), TransE constrains that the head embedding \(h\) plus the relation embedding \(r\) equals the tail embedding \(t\). Such single-triple constraints empower embedding models to implicitly perceive (i.e., memorize) the high-order entity connections and intrinsic relation rules (Sun et al., 2019). As shown in Figure 1, by imposing the structural constraints (e.g., \(h + r = t\) in TransE) on the five training triples, embedding models can memorize the entity connection \((x, r_1 \land r_2, z)\) and the relation rule \(r_1 \land r_2 \rightarrow r\). In this way, the missing link...
explicit

This learning process can be viewed as an
are summarized as follows: our contributions
both general techniques and can be widely applied
false negative samples. Note that VLP and ReD are
mative negative samples and reduce the toxicity of
technique (ReD), which can generate more infor-
effectively optimize the models, we further propose
memorization
strategy of HLP, making it signifi-
canonicalmente

</p>

We give an affirmative answer by presenting Ver-
fapon areplac

We conduct in-depth experiments on two stan-
dard benchmarks, demonstrating the validity
and generality of the proposed techniques.

2 Preliminaries

To elicit our proposal from a general paradigm per-
describe how the models learn to pre-
set up of KGC task. Afterwards, we sum-

2.1 Problem Setup

Given the entity set $\mathcal{E}$ and relation set $\mathcal{R}$, a knowl-
edge graph can be formally defined as a collection

of factual triples $\mathcal{D} = \{(h, r, t)\}$, in which

head/tail entities $h, t \in \mathcal{E}$ and relation $r \in \mathcal{R}$. KGC

task aims to infer new links by answering a query

$(h, r, ?)$ or $(?, r, t)$. As an effective tool for this

task, embedding model learns representations of

entities and relations to measure each candidate’s

plausibility with a pre-designed score function.

2.2 Generalized Score Function

Based on a series of previous works (Nickel et al.,
2011; Bordes et al., 2013; Wang et al., 2014; Lin
et al., 2015; Yang et al., 2015; Trouillon et al., 2016;
Sun et al., 2019; Gao et al., 2020; Song et al., 2021),
we summarize a generalized score function (GSF)

of embedding models. To facilitate presentation,

we only describe the query case of $(h, r, ?)$, while

$(?, r, t)$ can be similarly conducted.

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Model & Score Function & \multicolumn{3}{c|}{\(g(W_r, h + b_r, W_r, t)\)} & Space \\
\hline
RESCAL (Nickel et al., 2011) & \(h^\top W_r t\) & 0 & 0 & 0 & \(q^\top k\) & \(\mathbb{R}\) \\
TransE (Bordes et al., 2013) & \(-\|h + r - t\|\) & 0 & 0 & 0 & \(q^\top k\) & \(\mathbb{R}\) \\
TransR (Lin et al., 2015) & \(-\|W_r h + r - W_r t\|\) & 0 & 0 & 0 & \(q^\top k\) & \(\mathbb{R}\) \\
DistMult (Yang et al., 2015) & \(h\top \text{diag}(r) t\) & \(\text{diag}(r)\) & 0 & 0 & \(q^\top k\) & \(\mathbb{R}\) \\
ComplEx (Trouillon et al., 2016) & \(\text{Re}(h\top \text{diag}(r) t)\) & \(\text{diag}(r)\) & 0 & 0 & \(q^\top k\) & \(\mathbb{C}\) \\
RotatE (Sun et al., 2019) & \(-\|h \circ r - t\|\) & \(\text{diag}(r)\) & 0 & 0 & \(q^\top k\) & \(\mathbb{C}\) \\
\hline
\end{tabular}
\caption{The score functions and GSF settings of several models, where $\circ$ denotes the Hadamard product.}
\end{table}
Given a query \((h, r, ?)\) and a candidate answer \(t\), GSF first maps the head embedding \(h \in \mathbb{X}^{d_e}\) to the query embedding \(q \in \mathbb{X}^{d_k}\) with a relation-specific linear transformation:

\[
q = W_{r,1}h + b_r,
\]

(1)

where \(X \in \{\mathbb{R}, \mathbb{C}\}\) is the embedding space, \(d_e\) and \(d_k\) are the embedding dimensions of entities and relations, \(W_{r,1} \in \mathbb{X}^{d_k \times d_e}\) and \(b_r \in \mathbb{X}^{d_k}\) denote the relation-specific projection matrix and bias vector.

Then, GSF uses another linear function to generate the answer embedding \(k \in \mathbb{X}^{d_k}\) from the tail embedding \(t \in \mathbb{X}^{d_e}\):

\[
k = W_{r,2}t,
\]

(2)

where \(W_{r,2} \in \mathbb{X}^{d_k \times d_e}\) denotes the relation transformation matrix for tail projections.

Finally, the plausibility score of the triple \((h, r, t)\) is calculated by a similarity function \(g\):

\[
\text{score} = g(q, k).
\]

(3)

By combining the above three steps, we formally define the generalized score function \(f_g\) as follows:

\[
f_g(h, r, t) = g(W_{r,1}h + b_r, W_{r,2}t).
\]

(4)

With different choices of \(W_{r,1}, b_r, W_{r,2}, g\), GSF can be instantiated as specific score functions of existing models. Table 1 exhibits several popular methods and their corresponding GSF settings.

2.3 Horizontal Learning Paradigm

With the pre-defined score functions, embedding models commonly follow the horizontal learning paradigm, which constructs the single-edge constraints to implicitly memorize high-order entity connections and intrinsic relation rules.

Take RotatE to process the triples in Figure 1 as an example. By imposing the rotation constraints on three triples \((a, r_1, b), (b, r_2, c)\) and \((a, r, c)\), RotatE is able to perceive a two-hop entity connection and further induce a two-hop relation rule:

\[
\begin{align*}
    b &= a \circ r_1, \\
    c &= b \circ r_2 \Rightarrow r &= r_1 \circ r_2, \\
    c &= a \circ r \\
\end{align*}
\]

(5)

Similarly, the high-order connection can also be captured by constraining \((x, r_1, y)\) and \((y, r_2, z)\):

\[
\begin{align*}
    y &= x \circ r_1, \\
    z &= y \circ r_2 \Rightarrow z &= x \circ r_1 \circ r_2.
\end{align*}
\]

(6)

Finally, by combining Equation (5) and (6), RotatE is capable of inferring the missing link \((x, r, z)\).

3 Motivation

The motive of our work originates from an observation that embedding models underperform in predicting links between distant entity pairs (refer to Appendix A for more details). Since the effectiveness of embedding models is largely determined by the ability to learn intrinsic relation rules (Sun et al., 2019; Song et al., 2021; Li et al., 2022), such inferior performance reveals that the models are hard to memorize the multi-hop relation rules. We attribute this deficiency to the multi-hop bias accumulation and high-demanding memory capacity in the implicit memorization strategy of HLP.

Multi-hop Bias Accumulation The HLP-based embedding models implicitly perceive the multi-hop relation rules by constraining each training edge as shown in Section 2.3. Nevertheless, the single-edge constraints inevitably have biases during the optimization, which will accumulate with the increase of relation hops. This bias accumulation makes the memorized relation rules unreliable, leading to the deficient generalization ability for link prediction between distant entities. Concretely, considering the single-edge biases, the rule learning process in Equation (5) can be rewritten as:

\[
\begin{align*}
    b &= a \circ r_1 \circ \epsilon_1, \\
    c &= b \circ r_2 \circ \epsilon_2 \Rightarrow r &= r_1 \circ r_2 \circ \epsilon_{abc}, \\
    c &= a \circ r \circ \epsilon_0
\end{align*}
\]

(7)

where \(\epsilon_{abc} = \epsilon_0^{-1} \circ \epsilon_1 \circ \epsilon_2\) is the cumulative bias. Note that \(\epsilon_{abc}\) is triple-dependent, which makes it intractable for other queries, e.g., \((x, r, ?)\) in Figure 1, to rely on this rule for prediction.

High-demanding Memory Capacity The HLP-based models essentially learn the general rules from the relation paths between head and tail entities. With the increase of path length, the quantity of different paths (or rules) expands exponentially (Wang et al., 2021). This requires intensive memory to memorize the whole crucial relation rules. However, the modeling capacity of embedding models is insufficient to meet this requirement. Since these models constrain basic edges to form long-range paths following the bottom-up design of HLP, they are more inclined to memorize the low-order rules and forget the high-order rules.

Design Goal We seek to develop a general technique to alleviate the "Hard to Memorize" problem of existing embedding models.
A straight-forward strategy is to directly extract and process the enclosing subgraph between head and tail entities (Teru et al., 2020), which can avoid the multi-hop bias accumulation. However, such a sophisticated procedure needs to be executed once for each candidate triple, which brings enormous training and test time costs. For example, GraIL (Teru et al., 2020) takes about 1 month to infer on the full FB15k-237 test set (Zhu et al., 2021). Moreover, the enclosing subgraph extraction is also constrained by the path length, severely harming the performance of link prediction.

Therefore, this paper aims to propose a general framework which can: (1) alleviate the deficiency of HLP; (2) enjoy the merits of validity and generality with tractable computational costs.

4 Methodology

4.1 Vertical Learning Paradigm

Inspired by the notion that “to copy is easier than to memorize” (Khandelwal et al., 2020), we propose a vertical learning paradigm for KGC task. Different from the implicit memorization strategy of HLP, VLP provides embedding models with the ability to reference related triples as cues for prediction, which can be viewed as an explicit copy strategy.

More concretely, we present the overall pipeline of VLP in Figure 2. Given a query \( (h, r, ?) \), the procedure of predicting tail \( t \) can be divided into reference query selection, reference graph construction and reference answer aggregation.

Reference Query Selection For the input query \( q = (h, r, ?) \), the VLP-based models first select \( N \) entity-relation pairs \( \{h_i, r\}_i=1^N \) in the training graph as the reference queries \( \{q_i\}_i=1^N \), which can provide relevant semantics for prediction. For example, to answer \( \text{(Jill Biden, lives_in, ?)} \), we can reference the answer-known query \( \text{(Joe Biden, lives_in, ?)} \) for target information, since Joe Biden and Jill Biden are highly related. One intuitive way for the reference selection is to choose the top-\( k \) entities in terms of the cosine similarity between \( h \) and all entities involved in relation \( r \) during the optimization. Nevertheless, this approach incurs high computational costs and is intractable. Numerically, the time complexity of such similarity calculation is \( O(n_r d_e) \), where \( n_r \) is the number of \( r \)-involved entities and \( n_e \approx |E| \gg d_e \) in the worst case.

In this work, inspired by the small world principle (Newman, 2001; Liben-Nowell and Kleinberg, 2007), in which related individuals are connected by short chains (e.g., Joe Biden and Jill Biden are directly connected by the marriage relationship), we introduce the graph distance based approach for efficient reference query selection. Specifically, we select \( N \) \( r \)-involved entities \( \{h\}_{i=1}^N \) closest to \( h \) in terms of their relative graph distance (i.e., the shortest path length on the training graph). The corresponding ground-truth targets \( t_i \) of the reference queries \( q_i = (h_i, r, ?) \) are referred as reference answers. In this way, VLP-based models can pre-retrieve \( N \) related references for every input query, thus incurring no additional computational cost for training and inference.

Reference Graph Construction After the efficient reference retrieval, we construct an edge-attribute reference graph to integrate the selected \( N \) reference queries and their corresponding answers with the input query. As shown in Figure 2, the input query \( q \) is regarded as the central node, and the reference answers \( t_i \) are treated as the \( N \) neighbors. VLP-based models aims to leverage the explicit reference answers for prediction. However, since there is no guarantee that \( t_i \) is the same as the target tail \( t \), it is unreasonable to directly copy \( t_i \) without any modification. For example, to answer \( \text{(England, capital_is, ?)} \), we cannot directly copy the answer of \( \text{(France, capital_is, ?)} \).

Therefore, we introduce the query similarity \( s_{q,q_i} \) as the edge attribute between \( q \) and \( t_i \). By considering the query differences, VLP-based models are able to adaptively copy the reference answers. For example, to answer the input \( \text{(England, capital_is, ?)} \), we can adjust the target information from Paris in terms of the difference between \( \text{(France, capital_is, ?)} \) and the input query.
Reference Answer Aggregation  With the constructed reference graph, VLP-based models learn to explicitly gather target information from neighbor answers for prediction. Specifically, based on the generalized functions summarized in Section 2.2, the central node embedding \( q \) and neighbor node embedding \( k_i \) can be defined as:

\[
q = W_{r,1} h + b_r, \\
k_i = W_{r,2} t_i.
\]

The edge embedding \( s_{q,q_i} \) (i.e., query similarity embedding) can be further defined as:

\[
s_{q,q_i} = q - q_i = W_{r,1}(h - h_i).
\]

Then, combining the neighbor nodes and edge attributes, VLP-based models aggregate the reference answers to generate the final embedding \( t' \):

\[
t' = \sigma(W_{agg}[t_N, q]), \\
t_N = \frac{1}{N} \sum_{i=1}^{N} (W_{node} k_i + W_{edge} s_{q,q_i}),
\]

where \( \sigma(\cdot) \) is a nonlinear activation function (e.g., tanh), \([\cdot, \cdot]\) is the concatenate operation, \( W_{agg} \), \( W_{node} \) and \( W_{edge} \) are shared projection matrices. The output \( t' \) should be close to the target tail embedding \( t \) in the latent space, whose score can be revealed by the cosine similarity:

\[
f_c(h, r, t) = \frac{t^\top t'}{||t|| ||t'||}.
\]

We highlight that the VLP’s aggregating strategy in Equation (10) differs from GNN-based methods (Vashishth et al., 2020; Bansal et al., 2019; Shang et al., 2019; Schlichtkrull et al., 2018). For each query \((h, r, ?)\), regardless of whether the reference query is a neighbor of \( h \) in the training graph, VLP-based models can directly attend to the reference answer throughout the entire training set.

Score Function  For each triple \((h, r, t)\) in the test sets, to alleviate the deficiency of HLP and predict more accurately, we integrate the vertical score \( f_c \) with the horizontal score \( f_g \) to form the final score function \( f \) with a weight hyper-parameter \( \lambda \):

\[
f(h, r, t) = f_c(h, r, t) + \lambda f_g(h, r, t).
\]

Note that VLP can be widely applied to various embedding models, since the reference aggregation is designed on the generalized score function.

Complexity Analysis  Compared with the vanilla embedding models, the VLP-based models only bring a few additional parameters, i.e., the shared aggregation matrices in Equation (10). Therefore, the VLP-based models have the same space complexity as the HLP-based models, i.e., \( O(|\mathcal{E}| d_e) \). In the aspect of time cost for processing single triple, the time complexity of vanilla embedding models is \( O(d_e d_e) \), derived from the generalized score function in Equation (4). The VLP-based models require the same computation for each reference, which produces the time complexity of \( O(N d_e d_e) \). Such computation is tractable since a small \( N \) (no more than 8) is enough for VLP-based models to achieve high performance in the experiments.

4.2 Optimization  During training, we jointly optimize \( f_c \) and \( f_g \) by a two-component loss function with coefficient \( \alpha \):

\[
L = L_1 + \alpha L_2.
\]

For the former one, we use the cross-entropy between predictions and labels as training loss:

\[
L_1 = - \sum_{i=1}^{|\mathcal{E}|} y_i \log p_i,
\]

where \( p_i \) and \( y_i \) are the \( i \)-th components of \( \mathbf{p} \) and \( \mathbf{y} \), respectively; \( \mathbf{p} \in \mathbb{R}^{|\mathcal{E}|} \) is calculated by applying the softmax function to the "1-to-All" (Lacroix et al., 2018a) results of \( f_c \); \( \mathbf{y} \in \mathbb{R}^{|\mathcal{E}|} \) is the one-hot vector that indicates the position of true label.

For the latter one, negative sampling has been proved quite effective in extensive works (Song et al., 2021; Sun et al., 2019). Formally, for a positive triple \((h, r, t)\), we first sample a set of entities \( \{t'_i\}_{i=1}^l \) (or \( \{h'_i\}_{i=1}^l \) based on the pre-sampling weights \( p_0 \) to construct negative triples \((h, r, t'_i)\) (or \((h'_i, r, t)\)). With these samples, a negative sampling loss is designed to optimize embedding models:

\[
L_2 = - \sum_{i=1}^l p_1(h'_i, r, t'_i) \log \sigma(-f(h'_i, r, t'_i) - \gamma) - \log \sigma(\gamma + f(h, r, t)),
\]

where \( \gamma \) is a pre-defined margin, \( \sigma \) is the sigmoid function, \( l \) denotes the number of negative samples, \((h'_i, r, t'_i)\) is a negative sample against \((h, r, t)\). Importantly, \( p_1(h'_i, r, t'_i) \) is the post-sampling weight, which determines the proportion of \((h'_i, r, t'_i)\) in the current optimization.
As shown in Figure 3, recent works (Song et al., 2021; Chao et al., 2021; Gao et al., 2020; Sun et al., 2019) utilize the self-adversarial technique (Self-Adv), in which the pre-sampling weights follow a uniform distribution and the post-sampling weights increase with the negative scores. Differently, in this work, we propose a new approach named ReD based on the relative distance, which can draw more informative negative samples and reduce the toxicity of false negative samples.

For the pre-sampling weights, considering the deficiency of embedding models as described in Section 3, the distant entities are usually hard to be predicted as the target answer. It reveals a rational priori, i.e., distant entities are more likely to form easy (meaningless) negative triples. This inspires us to sample more hard (informative) negative triples based on the relative graph distance $d_g$. As shown in Figure 3, the pre-sampling weight in ReD decreases with the increase of graph distance between head and tail entities. Formally, for a training query $(h, r, ?)$, we pre-sample entities $t'$ to construct negatives from the following distribution:

$$p_0(h, r, t') = \frac{\exp - \alpha_0 d_g(h, t')}{\sum_{i=1}^{[\mathcal{E}]} \exp - \alpha_0 d_g(h, t'_i)}, \quad (16)$$

where $\alpha_0$ is the pre-sampling temperature, $d_g(\cdot, \cdot)$ outputs the relative graph distance between two entities. Note that the calculation of $d_g(\cdot, \cdot)$ is a one-time preprocessing step, which will not bring additional training overhead.

For the post-sampling weights, Self-Adv assigns greater weights to high scoring negative triples in Equation (15), which makes the optimization focus more on hard negatives. However, this monotonically increasing strategy ignores the issue of false negatives, since triples with higher scores are more likely to be correct. A more rational posteriori is that the easy negatives are underscored and the false negatives are overscored. In this work, we use the relative latent distance between the positive and negative samples to determine whether the negative score is too low or too high. Specifically, ReD defines the post-sampling weights as a distribution that first rises and then falls as the negative score increases. As shown in Figure 3, if the negative score is significantly greater than (or less than) the positive score, this negative sample is more likely to be false (or easy), and thus be assigned a small weight in the Equation 15. Formally, based on the positive score $c = f_p(h, r, t)$ and negative score $n_i = f_g(h_i', r, t'_i)$, the post-sampling weight in ReD is defined as:

$$p_i(h_i', r, t'_i) = \frac{\exp w(h_i', r, t'_i)}{\sum_j \exp w(h_j', r, t'_j)},$$

$$w(h_i', r, t'_i) = \begin{cases} \alpha_1 n_i, & n_i \leq c + \tau \\ \alpha_1 c - \alpha_2 n_i, & n_i > c + \tau, \end{cases}$$

$$m_i = n_i - c - \tau, \quad (17)$$

where $\alpha_1$ and $\alpha_2$ are the post-sampling temperatures. By combining the sampling weights in Equation (16) and (17), ReD is able to generate and process higher quality negatives for optimization.

## 5 Experiment

### 5.1 Experimental Setup

**Datasets** We evaluate our proposal on two widely-used benchmarks: WN18RR (Dettmers et al., 2018) and FB15k-237 (Toutanova and Chen, 2015). More details can be found in Appendix B.

**Baselines** To verify the effectiveness and generalizability of our proposal, we combine the proposed techniques with three representative embedding models DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016) and RotatE (Sun et al., 2019). For performance comparison, we select a series of embedding models as baselines in Table 2.

**Implementation Details** We fine-tune the hyper-parameters with the grid search on the validation sets. Please see Appendix C for more details.

### 5.2 Main Results

The experimental results are reported in Table 2. Compared to DistMult, ComplEx and RotatE, all three VLP-based versions achieve consistent and significant improvements on both datasets. For example, on WN18RR and FB15k-237 datasets, RotatE-VLP outperforms RotatE with 2.2% and
Table 2: Link prediction results on WN18RR and FB15k-237. Best results are in **bold** and second best results are underlined. \[1\]: Results are taken from (Nguyen et al., 2018). \[2\]: we re-evaluate DistMult and ComplEx based on the open source codes from (Sun et al., 2019), achieving better results than those reported in the original papers.

Table 3: MRR on each distance split of WN18RR.

Table 4: MRR on each relation of WN18RR.

Table 5: Performance on Distance Splits Table 3 reports the performance of three VLP-based models on the distance splits defined in Appendix A. One can observe that: (1) the VLP-based embedding models outperform the vanilla models across all the distance splits; (2) the VLP models achieve greater relative improvement on the split with larger \(d_{ht}\). For example, as \(d_{ht}\) increases from 1 to 4, RotatE-VLP achieves 0.5%, 4.3%, 20.6% and 22.0% relative improvements over RotatE on the MRR metric, respectively. This reveals that the explicit vertical contexts can significantly alleviate the limitations of memory strategy in the embedding models.

**Performance on Each Relation** To verify the modeling capacity of our proposal from a fine-grained perspective, we explore the performance of VLP-based models on each relation of WN18RR following (Zhang et al., 2019). As shown in Table 4, compared to RotatE and QuatE, RotatE-VLP surpasses them on all the 11 relation types, confirming that the explicit reference aggregation brings superior modeling capacity.

**Performance on Mapping Properties** Table 5 exhibits the performance of our proposal on different relation mapping properties (Sun et al., 2019) in FB15k-237. We observe that RotatE-VLP consistently outperforms RotatE across all RMP types. Such advanced performance owes to the powerful modeling capability of the explicit copy strategy.
Table 5: MRR on mapping properties in FB15k-237.

5.4 Impact of Reference Quantity
VLP aggregates target information from \( N \) references pre-selected before training. We investigate the impact of \( N \) on the performance (MRR) of VLP-based models. Figure 4 shows the results on WN18RR dataset. As expected, all three VLP-based models with more vertical references achieve better performance than the ones with fewer references, since the aggregation of sufficient references brings the superior modeling capacity. Moreover, we can observe that the models can achieve high performance with \( N \) less than 10, making the computation tractable as discussed in Section 4.1.

5.5 Ablation Study of ReD
To explore the effectiveness of the proposed ReD, we conduct ablation studies on the pre-sampling and post-sampling parts of the three VLP-based models. Table 6 shows the detailed results. We can observe that the removal of any part reduces the performance, which demonstrates that ReD makes the model focus more on meaningful negative samples for more effective optimization. Moreover, we also integrate ReD with original embedding models to verify the generality of this technique. Please refer to Appendix D for more results.

6 Related Work
Embedding models can be roughly categorized into distance based models and semantic matching models (Chao et al., 2021). Distance based models use the Euclidean distance to measure the plausibility of each triple. A series of work is conducted along this line, such as TransE (Bordes et al., 2013), TransH (Wang et al., 2014), TransR (Lin et al., 2015), RotatE (Sun et al., 2019), PairRE (Chao et al., 2021), Rot-Pro (Song et al., 2021), ReflectE (Zhang et al., 2022) and so on. TransE and RotatE are the most representative distance-based models, which represent relations as translations and rotations, respectively. Semantic matching models utilize multiplicative functions to score each triple, including RESCAL (Nickel et al., 2011), DistMult (Yang et al., 2015), ComplEx (Trouillon et al., 2016), QuatE (Zhang et al., 2019), DualE (Cao et al., 2021) and so on. Typically, RESCAL (Nickel et al., 2011) defines each relation as the tensor decomposition matrix. DistMult (Yang et al., 2015) simplifies the relation matrices to be diagonal for preventing overfitting. However, existing embedding models essentially follow the horizontal learning paradigm, underperforming in predicting links between distant entities.

Moreover, some advanced techniques are proposed to improve embedding models, such as graph encoders (Schlichtkrull et al., 2018; Shang et al., 2019; Vashishth et al., 2020; Wang et al., 2022) and regularizers (Lacroix et al., 2018b). Note that our proposals are orthogonal to these techniques, and one can integrate them for better performance.

7 Conclusion
In this paper, we present a novel learning paradigm named VLP for KGC task. VLP can be viewed as an explicit copy strategy, which allows embedding models to consult related triples for explicit references, making it much easier to predict distant links. Moreover, we also propose ReD, a new negative sampling technique for more effective optimization. The in-depth experiments on two datasets demonstrate the validity and generality of our proposals.
Limitations

Although our proposal enjoys the advantages of validity and generality, there are still two major limitations. First, VLP cannot directly generalize to the inductive setting, since VLP is defined based on the score functions of transductive embedding models. One potential direction is to design an inductive reference selector for emerging entities. Second, how to efficiently select more helpful references for prediction is still an open challenge. We expect future studies to mitigate these issues.

Acknowledgements

This work was supported in part by the National Natural Science Foundation of China under Grant 62276044, and also Sponsored by CAAI-Huawei MindSpore Open Fund.

References


Zongsheng Cao, Qianqian Xu, Zhiyong Yang, Xiaochun Cao, and Qingming Huang. 2021. Dual quaternion knowledge graph embeddings. In Thirty-Fifth AAAI Conference on Artificial Intelligence, pages 6894–6902.


Table 7 summarizes the detailed statistics of two benchmark datasets. WN18RR (Dettmers et al., 2018) and FB15k-237 are tested on each split. Representative embedding models (DistMult, ComEx and RotatE) are tested on each split.

![Image](image-url)

**Figure 5** summarizes the detailed MRR results and split ratios on the two datasets. We can observe that all three embedding models achieve promising results in link prediction between close entities, while the performance drops significantly in the prediction between distant entities. For example, on the split where \(d_{ht} = 1\) in WN18RR, RotatE achieves excellent performance (MRR of 0.986), while on the split where \(d_{ht} = 2\), the performance of RotatE decreases by about 62% (MRR of 0.375).

### B Datasets

Table 7 summarizes the detailed statistics of two benchmark datasets. WN18RR (Dettmers et al., 2018) and FB15k-237 are tested on each split.

- **WN18RR**: A benchmark dataset for knowledge graph completion, consisting of 18 entities and 41 relations. It is divided into 5 training splits and 1 test split.
- **FB15k-237**: Another benchmark dataset for knowledge graph completion, consisting of 15,127 entities and 23,728 relations. It is divided into 5 training splits and 1 test split.
Figure 5: The MRR results of three popular embedding models (DistMult, ComplEx and RotatE) tested on each distance split in WN18RR and FB15k-237 datasets. The blue diamond marks denote the ratio of each test split.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>WN18RR</th>
<th>FB15k-237</th>
</tr>
</thead>
<tbody>
<tr>
<td>#entity</td>
<td>40,943</td>
<td>14,541</td>
</tr>
<tr>
<td>#relation</td>
<td>11</td>
<td>237</td>
</tr>
<tr>
<td>#training</td>
<td>86,835</td>
<td>272,115</td>
</tr>
<tr>
<td>#validation</td>
<td>3,034</td>
<td>17,535</td>
</tr>
<tr>
<td>#test</td>
<td>3,134</td>
<td>20,466</td>
</tr>
</tbody>
</table>

Table 7: Statistics of five standard benchmarks.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>Search Space</th>
</tr>
</thead>
<tbody>
<tr>
<td>$b$</td>
<td>{256, 512, 1024}</td>
</tr>
<tr>
<td>$d$</td>
<td>{500, 1000}</td>
</tr>
<tr>
<td>$\alpha_0, \alpha_1, \alpha_2$</td>
<td>{0.1, 0.5, 1.0, 1.5}</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>{0.1, 0.3, 0.5, 0.7, 0.9}</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>{4, 6, 8, 11, 15}</td>
</tr>
</tbody>
</table>

Table 8: Hyperparameter search space.

Dataset WN18RR FB15k-237

<table>
<thead>
<tr>
<th>Model</th>
<th>MRR H@10</th>
<th>MRR H@10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DistMult-Adv</td>
<td>0.439</td>
<td>0.534</td>
</tr>
<tr>
<td>DistMult-ReD</td>
<td>0.445</td>
<td>0.539</td>
</tr>
<tr>
<td>ComplEx-Adv</td>
<td>0.466</td>
<td>0.552</td>
</tr>
<tr>
<td>ComplEx-ReD</td>
<td>0.470</td>
<td>0.554</td>
</tr>
<tr>
<td>RotatE-Adv</td>
<td>0.476</td>
<td>0.571</td>
</tr>
<tr>
<td>RotatE-ReD</td>
<td>0.478</td>
<td>0.572</td>
</tr>
</tbody>
</table>

Table 9: Results of DistMult, ComplEx and RotatE with different negative sampling techniques. X-Adv denotes the embedding model X combined with Self-Adv.

We use Adam (Kingma and Ba, 2015) as the optimizer and fine-tune the hyperparameters on the validation dataset. The hyperparameters are tuned by the grid search, including batch size $b$, embedding dimension $d$, negative sampling temperatures $\{\alpha_i\}_{i=0}^2$, loss weight $\lambda$ and fixed margin $\gamma$. The hyper-parameter search space is shown in Table 8.

To verify the generality of the proposed negative sampling technique ReD, we integrate ReD with three representative embedding models (i.e., DistMult, ComplEx and RotatE) for KGC task. As shown in Table 9, compared to Self-Adv, the embedding models combined with ReD achieve better performance on both datasets, since ReD guarantees more informative negative samples from both pre-sampling and post-sampling stages.

2018) and FB15k-237 (Toutanova and Chen, 2015) datasets are subsets of WN18 (Bordes et al., 2013) and FB15k (Bordes et al., 2013) respectively with inverse relations removed. WN18 is extracted from WordNet (Miller, 1995), a database featuring lexical relations between words. FB15k is extracted from Freebase (Bollacker et al., 2008), a large-scale KG containing general knowledge facts.

C Implementation Details

We use Adam (Kingma and Ba, 2015) as the optimizer and fine-tune the hyperparameters on the validation dataset. The hyperparameters are tuned by the grid search, including batch size $b$, embedding dimension $d$, negative sampling temperatures $\{\alpha_i\}_{i=0}^2$, loss weight $\lambda$ and fixed margin $\gamma$. The hyper-parameter search space is shown in Table 8.

D Embedding Models with ReD

To verify the generality of the proposed negative sampling technique ReD, we integrate ReD with