MGAD: Learning Descriptive Representation Distilled from Distributional Semantics for Unseen Entities

Yuanzheng Wang\textsuperscript{1,2}, Xueqi Cheng\textsuperscript{1,2}, Yixing Fan\textsuperscript{1,2}, Xiaofei Zhu\textsuperscript{3}, Huasheng Liang\textsuperscript{4}, Qiang Yan\textsuperscript{4} and Jiafeng Guo\textsuperscript{1,2}

\textsuperscript{1}CAS Key Lab of Network Data Science and Technology, ICT, CAS, Beijing, China  
\textsuperscript{2}University of Chinese Academy of Sciences, Beijing, China  
\textsuperscript{3}College of Computer Science and Engineering, Chongqing University of Technology  
\textsuperscript{4}WeChat, Tencent, Guangzhou, China

{wangyuanzheng19z, cxq, fanyixing, guojiafeng}@ict.ac.cn, 
{zxf@cqut.edu.cn, watsonliang, rolanyan}@tencent.com

Abstract

Entity representation plays a central role in building effective entity retrieval models. Recent works propose to learn entity representations based on entity-centric contexts, which achieve SOTA performances on many tasks. However, these methods lead to poor representations for unseen entities since they rely on a multitude of occurrences for each entity to enable accurate representations. To address this issue, we propose to learn enhanced descriptive representations for unseen entities by distilling knowledge from distributional semantics into descriptive embeddings. Specifically, we infer enhanced embeddings for unseen entities based on descriptions by aligning the descriptive embedding space to the distributional embedding space with different granularities, i.e., element-level, batch-level and space-level alignment. Experimental results on four benchmark datasets show that our approach improves the performance over all baseline methods. In particular, our approach can achieve the effectiveness of the teacher model on all almost entities, and maintain such high performance on unseen entities.

1 Introduction

Entity retrieval, which aims to efficiently filter a small number of candidate entities from a large knowledge base (KB) for a given mention, is a fundamental task for various natural language processing tasks, such as fact extraction and verification [Nooralahzadeh and Øvrelid, 2018]. Recently, embedding-based retrieval methods have achieved great success in entity retrieval due to their effectiveness in capturing the semantics [Wu et al., 2020b; Gillick et al., 2019; Botha et al., 2020]. In essence, these embedding-based methods represent entities and mentions with standalone encoders, in which entity embeddings can be pre-computed and stored offline for efficient retrieval.

A common paradigm for learning entity representations is to take attributes of each entity as input to the encoder, since these attributes consist of concise descriptions with rich semantic information. For example, [Wu et al., 2020b] proposes a BERT-based two-tower model to encode mentions and entity descriptions, and retrieves entities by nearest-neighbor search. We denote these methods as descriptive representation methods. On the other hand, many recent studies propose to learn entity representations based on contexts around its mentions, which is grounded on the distributional hypothesis [Harris, 1954; Firth, 1957]. For example, LUKE [Yamada et al., 2020] takes anchor-texts in Wikipedia as mentions to learn entity embedding, then the embedding can be used in downstream tasks. We denote these methods as distributional representation methods. Distributional representation methods have achieved SOTA performances on several benchmark datasets. This is not surprising since distributional entity embeddings are more consistent with mention embeddings, making them clearly easier to match.

While distributional representation methods achieve better results than descriptive representation methods, the effectiveness of these methods highly relies on large amounts of occurrences for each entity. However, there are often lots of newly emerging entities, such as events and products, which are introduced continually along with the dynamic of KBs. For example, “COVID-19” is a new entity, which does not appear in the training corpus. These new entities often appear with only short descriptions accompanied with them, which we refer to as unseen entities. As a result, distributional representation methods cannot provide reliable representations for these unseen entities since there are no contexts for them in the training corpus.

In this paper, we aim to learn enhanced descriptive representations for unseen entities with inferred distributional semantics by employing a distillation framework. More precisely, we propose to leverage the distillation model to bridge the distributional embedding space and the descriptive embedding space, where the context-based encoder and the description-based encoder are considered as the teacher and the student, respectively. With the distillation model, we can transfer knowledge from the distributional embedding space (teacher) to the descriptive embedding space (student), and obtain the enhanced descriptive representa-
tions for unseen entities. To this end, we propose a novel distillation model referred as Multi-Granularity Alignments based Distillation (MGAD) model. Specifically, we first embed entity description into a vector space and then align the embedded entity from this space to a distributional semantic space. The distillation is implemented with four loss functions. Besides a retrieval loss, we propose three alignment losses with different granularities, i.e., element-level, batch-level and space-level alignment loss. Experiments on four entity linking datasets show that MGAD performs competitively to the teacher model, and outperforms all other baselines.

2 Related Works

In this section, we briefly review studies related to our work, including entity retrieval models, unseen entity representation learning, and lexical semantics alignment.

Entity Retrieval. Most entity retrieval systems consist of two stages, i.e., the candidate generation stage and the candidate ranking stage, to balance the efficiency and effectiveness [Wu et al., 2020b; Ganea and Hofmann, 2017; Onoe et al., 2021]. The candidate generation stage is to efficiently filter a small number of candidate entities from a large-scale KB, while the candidate ranking stage (also known as the disambiguation stage) chooses the most probable entity for each mention among the found candidates. In this paper, we focus on the candidate generation stage.

Researchers proposed different models for entity retrieval in the candidate generation stage. Traditional methods rely on heuristic functions [Le and Titov, 2019; Wu et al., 2020b; Yamada and Shindo, 2019], such as BM25 [Logeswaran et al., 2019] or alias table [Ganea and Hofmann, 2017] to build the model. Recently, neural models achieved great success in entity retrieval [Gillick et al., 2019; Botha et al., 2020; Wu et al., 2020a]. Existing neural models can be categorized into two classes, namely discriminative retrieval models and generative retrieval models. The former represents mentions and entities with dense vectors, and takes either simple interaction functions to evaluate their similarity [Gillick et al., 2019; Botha et al., 2020]. Example models like LUKE [Yamada et al., 2020] and BLINK [Wu et al., 2020b] take BERT or RoBERTa to encode contexts and descriptions to obtain entity embeddings, respectively. On the other hand, the latter retrieves entities by exploiting sequence-to-sequence architecture to generate entity names. For example, GENRE [Cao et al., 2021] takes a transformer architecture with a pre-trained language modeling objective to generate entity names.

Unseen Entity Problem. It is a unique problem for distributional representation methods in learning embeddings for unseen entities since they rely on contexts to learn the embedding. Instead, descriptive representation methods, e.g., BLINK [Wu et al., 2020b] and GENRE [Cao et al., 2021], are able to learn representations for unseen entities since they only rely on entity descriptions and titles. However, our experiments show that descriptive representation methods are less effective than distributional representation methods on seen entities. There are also some methods that employ external resources to learn embeddings for unseen entities. For example, DEEP-ED [Ganea and Hofmann, 2017] and CDTE [Gupta et al., 2017] learn embeddings from entity descriptions or types. ET4EL [Onoe et al., 2021] matches fine-grained entity types without the need for entity embeddings. There are two tasks that are similar but different from our setting. The first is zero-shot entity linking [Logeswaran et al., 2019] which focuses on generalization on unseen domains. The second is unseen-mention entity linking [Onoe et al., 2021] which focuses on generalization on unseen mentions.

Lexical Semantics Alignment. Aligning embeddings from two different semantic spaces has been well studied in word representation learning [Wang et al., 2021; Zock and Schwab, 2008]. Researchers have proposed different models to learn word representations from descriptions, definitions, or contexts. For example, [Hill et al., 2016] encodes word definitions to fit the pretrained distributional word embedding by LSTM or Bag-of-Words model. [Bosc and Vincent, 2018] proposes to learn word embeddings based on dictionary definitions by reconstructing the word as well as the definition. [Bevilacqua et al., 2020] decodes a word with surrounding contexts into its definition by BART. Inspired by these studies, we try to address the distributional representation learning for unseen entities based on embedding alignment.
3 Our Approach

In this section, we will introduce our approach in learning enhanced descriptional representations for unseen entities by transferring knowledge from the distributional semantics based on the distillation framework. Specifically, our work is inspired by [Prokhorov et al., 2019], which employs canonical correlation analysis to align two embedding spaces, to learn semantic representations for unseen words. In this work, we borrow the idea of space alignment to learn embeddings for unseen entities. Instead of learning a linear mapping matrix, we propose a novel multi-granularity embeddings for unseen entities. Instead of learning a linear embedding method is LUKE [Yamada et al., 2020b] to take the pre-trained language model as input.

3.1 Entity Embedding

In this subsection, we will introduce two different embedding methods to learn entity representations, namely description-based and context-based entity embedding.

Description-based Embedding. The description-based embedding method takes only the entity description as input to learn entity embedding. In this work, we follow BLINK [Wu et al., 2020b] to take the pre-trained language model as the entity encoder. Specifically, for any entity $e \in E_{all}$ with $name_e$ and description $desc_e$, the input $I_e$ for entity encoder is constructed as:

$$I_e = [CLS] name_e [SEP] desc_e [SEP].$$

Then the descriptional entity embedding $v_e$ is computed with:

$$v_e = \text{AveragePool}(\text{Enc}_m(I_e)),$$

where $\text{Enc}_m(\cdot)$ is a RoBERTa-like entity encoder.

Context-based Embedding. The context-based embedding methods learn distributional representations from entity-centric contexts for each entity. The state-of-the-art context-based embedding method is LUKE [Yamada et al., 2020], which simultaneously learns contextualized representations of words and entities based on Transformer. In LUKE, for (and only for) any $e \in E_{seen}$, each entity embedding $u_e$ is randomly initialized and pre-trained with multitude of entity-centric contexts. This pre-training task is a variant of masked language model (MLM) task, where the model is trained to predict randomly masked entities in a context. In this work, we take LUKE as the teacher to guide our model to learn distributional representations for unseen entities.

The descriptional and distributional embedding spaces will be aligned by knowledge distillation, which will be introduced in Section 3.3. Before that, in the next section, we first introduce how mentions and entities are matched in retrieval.

3.2 Mention-entity Matching

In this subsection, we will introduce the mention encoder in LUKE and MGAD, and then introduce how to score a mention-entity pair $(m, e)$.

Mention Encoder. Here, we directly employ the mention encoder architecture of LUKE since we focus on the entity representation learning in this work. The mention encoder is a variant of RoBERTa and can encode several mentions simultaneously. Let $I_m$ be the input of $n$ mentions with surrounding contexts:

$$I_m = [CLS] c m_1 c \cdots c m_n c [SEP] [M]_1 \cdots [M]_n,$$

where $c$ is a non-mention word span, $[M]$ is a special entity-mask token and $[M]_i$ corresponds to the word span of the $i$-th mention $m_i$ by position embeddings. Then mention embeddings $\{v_{m_1}, \ldots, v_{m_n}\}$ are computed with:

$$\{v_{m_1}, \ldots, v_{m_n}\} = \text{Enc}_m(I_m) [M]_1 \cdots [M]_n,$$

where $\text{Enc}_m(\cdot)$ denotes the mention encoder, and $\{v_{m_1}, \ldots, v_{m_n}\}$ are the embeddings of $[M]_1 \cdots [M]_n$ in the last self-attention layer. It is worthy to note that this mention encoder benefits from long-distance contextual information and achieves SOTA performance on many entity-related tasks [Yamada et al., 2020].

Scoring Function. Given a mention-entity pair $(m, e)$, $u_e \in R^H$ denotes the entity embedding in LUKE, where $H$ is the entity embedding size. $v_e \in R^D$ denotes the descriptional entity embedding in MGAD, and $v_m \in R^D$ denotes the mention embedding in both model, where $D$ is the shared hidden size of $\text{Enc}_m(\cdot)$ and $\text{Enc}_e(\cdot)$ in Eq. (2) and (4). In MGAD, the matching score $s(m, e)_{MGAD}$ between a mention-entity pair is calculated by the dot-product similarity $v_m \cdot v_e$.

In LUKE, due to the dimension miss-matching between $v_m$ and $u_e$, a transformation $\phi$ is firstly applied on $v_m$:

$$\phi(v_m) = W_2 \cdot \text{LayerNorm} (\text{GELU}(W_1 \cdot v_m + b)),$$

where $\phi(v_m) \in R^H$. $W_1 \in R^{D \times D}$, $b \in R^D$ and $W_2 \in R^{H \times D}$ are learnable parameters. Then the matching score $s(m, e)_{LUKE}$ can be calculated with dot-product $\phi(v_m) \cdot u_e$.

With the precomputed and stored entity embeddings, give a mention embedding $v_m$, both models can retrieve candidate entities by nearest-neighbor search.

3.3 Loss Function

Our distillation-based embedding alignment is implemented with four loss functions. Since we focus on entity retrieval, a retrieval loss $\mathcal{L}_{re}$ is necessary. Furthermore, there are three alignment losses with different alignment granularities, namely element-level alignment loss $\mathcal{L}_{ea}$, batch-level alignment loss $\mathcal{L}_{ba}$ and space-level alignment loss $\mathcal{L}_{sa}$. The final loss $\mathcal{L}$ of our distillation-based alignment is the weighted sum of the four losses:

$$\mathcal{L} = \alpha_1 \cdot \mathcal{L}_{re} + \alpha_2 \cdot \mathcal{L}_{ea} + \alpha_3 \cdot \mathcal{L}_{ba} + \alpha_4 \cdot \mathcal{L}_{sa},$$

where $\alpha_i (i = 1, \ldots, 4)$ are hyper-parameters.

Retrieval Loss. Retrieval loss $\mathcal{L}_{re}$ is for retrieval target itself. Following [Gillick et al., 2019], we use in-batch random negatives to sample positive and negative $(m, e)$ pairs:
for each training batch, there are $n$ randomly sampled mentions and their corresponding $n$ entities (namely gold entities), and for each mention $m$, we construct one positive pair $(m, e)$ with its gold entity $e$ and $n - 1$ negative pairs $(m, \tau)$ with the other entities $\tau$, which can be represented by an one-hot label vector $l^{batch} \in \{0, 1\}^n$. For mention $m$, let $\tau \in R$ be the temperature and $s^{batch}_{m, \tau} \in R^n$ be the scores over $n$ entities, the score distribution $\sigma_{\tau}(s^{batch}_{m, \tau})$ for $m$ over $n$ entities can be defined as follows:

$$\sigma_{\tau}(s^{batch}_{m, \tau}) = \frac{\exp(s^{batch}_{m, \tau}/\tau)}{\sum_j \exp(s^{batch}_{m, j})}.$$  

Let $s^{batch}_{MGAD, m} \in R^n$ be the score of mention $m$ over $n$ in-batch entities from MGAD and $\sigma_{\tau}(s^{batch}_{MGAD, m}) \in R^n$ be the score distribution, then $\mathcal{L}_{re}$ can be defined with CrossEntropy:

$$\mathcal{L}_{re} = \text{CrossEntropy}(l^{batch}, \sigma_{\tau}(s^{batch}_{MGAD, m})).$$  

**Element-level Alignment Loss.** As shown in Figure 2(a), element-level alignment loss $\mathcal{L}_{ea}$ is an “one-to-one” alignment, where the descriptional embedding $\psi_e$ should be similar to the corresponding distributional embedding $u_e$. In practice, because of the dimension miss-matching between $\psi_e$ and $u_e$, a transformation function $\psi(\cdot)$ is first applied on $\psi_e$, where $\psi(\psi_e, u_e) \in R^H$. This is a usual solution for dimension miss-matching in feature-based knowledge distillation [Gou et al., 2020]. $\mathcal{L}_{ea}$ can be defined with the MSE loss:

$$\mathcal{L}_{ea} = \|u_e - \psi(\psi_e)\|^2.$$  

**Batch-level Alignment Loss.** As shown in Figure 2(b), batch-level alignment loss $\mathcal{L}_{ba}$ is an “one-to-batch” alignment, where the matching distribution of a mention over in-batch entities from the student is required to be similar to the teacher. $\mathcal{L}_{ba}$ can be defined with KL-Divergence [Wu et al., 2020b]:

$$\mathcal{L}_{ba} = \text{KL-Divergence}\left(\sigma_{\tau}(s^{batch}_{LUKE, m}) \cdot \sigma_{\tau}(s^{batch}_{MGAD, m})\right),$$

where $\sigma_{\tau}$ converts matching scores into probability distributions, $s^{batch}_{LUKE, m}$ and $s^{batch}_{MGAD, m}$ denote the matching scores of mention $m$ from LUKE and MGAD. $\sigma_{\tau}(s^{batch}_{MGAD, m}) \in R^n$ is the target distribution in KL-Divergence.

**Space-level Alignment Loss.** As shown in Figure 2(c), space-level alignment loss $\mathcal{L}_{sa}$ is an “one-to-all” alignment. In other words, one descriptional embedding $\psi_e$ should be able to select its corresponding $u_e$ from the entire distributional embedding space $S_{dist}$. Essentially, it can be seen as a “generation” task with vocab $E_{seen}$, whose label $l^{max} \in \{0, 1\}^{|E_{seen}|}$ is a one-hot vector. For $\psi_e$, let $s^{gen, e} \in R^{|E_{seen}|}$ be the “generation scores” over $E_{seen}$, $\sigma_e(s^{gen, e}) \in R^{|E_{seen}|}$ be the score distribution, $s^{gen, e}$ can be defined with dot-product score:

$$s^{gen, e}_{m, i} = (\psi_e)^\top \cdot u_e_{i} \in \{1, \cdots, |E_{seen}|\}$$

Then $\mathcal{L}_{sa}$ can be defined with CrossEntropy:

$$\mathcal{L}_{sa} = \text{CrossEntropy}(l^{max}, \sigma_e(s^{gen, e})).$$

## 4 Experiment

In this section, we conduct experiments to demonstrate the effectiveness of our proposed model on benchmark datasets.

### 4.1 Experiment Setup

We first introduce our experiment settings, including datasets, entity collections, baseline methods, training details, and evaluation metrics.

**Datasets.** To evaluate the performance of our model, we choose four widely used entity linking datasets: AIDA [Hoffart et al., 2011], ACE [Ratinov et al., 2011], AQUAINT [Milne and Witten, 2008] and MSNBC [Cucerzan, 2007]. All documents in these datasets are manually annotated news articles. The collection of all gold entities in four datasets is denoted as $E_{gold}$.

**Entity Collections.** As denoted in Section 3, following LUKE [Yamada et al., 2020], $E_{seen}$ is the collection of the most popular 500K entities in Wikipedia, whose distributional embeddings are pretrained by LUKE. Furthermore, $E_{unseen}$ is the collection of all $e \in E_{gold} - E_{seen}$ and 500K+ additional random entities. $E_{all} = E_{seen} \cup E_{unseen}$. Entity names and descriptions come from the Wikipedia dump cleaned by [Wu et al., 2020b].

**Baseline Methods.** We compare with five baselines:

- **Alias:** prior alias table is a heuristic method, in which candidate entities are retrieved by exactly alias matching and sorted by prior probability $P(c|m)$. We directly take the implementation of REL [van Hulst et al., 2020].
- **BM25:** we follow [Wu et al., 2020b] to compare with BM25 where entities are indexed by titles.
- **GENRE** [Cao et al., 2021]: a BART-based state-of-the-art generative model for entity retrieval.
- **BLINK** [Wu et al., 2020b]: a BERT-based two-tower retrieval model which learns description-based embeddings for entities.
- **LUKE** [Yamada et al., 2020]: a variant RoBERTa-based distributional two-tower model, which is our distillation teacher. Note that all experimental results of LUKE with $E_{unseen}$ are empty because of the lack of distributional embeddings for $E_{unseen}$.

**Training Details.** Knowledge distillation aims to learn entity embeddings. Following [Yamada and Shindo, 2019], for further improving the matching ability, we infer the entity embeddings of $E_{seen}$, and finetune the mention encoder of MGAD on AIDA-train with entity embeddings fixed. For a fair comparison, other models are also finetuned in this way. Parameters are shared between $\text{Enc}_e(\cdot)$ in Eq. (2) and $\text{Enc}_m(\cdot)$ in Eq. (4). The initial parameters for $\text{Enc}_e(\cdot)$ and $\text{Enc}_m(\cdot)$ are
Table 1: Performance on the in-domain test set AIDA-testB. AL, BM, GE, BL, MG, LU are the abbreviations for Alias, BM25, GENRE, BLINK, MGAD, LUKE. **Bold** marks the best performance. Significant improvement or degradation with respect to MGAD is indicated (+/-) (p-value ≤ 0.05).

<table>
<thead>
<tr>
<th>Entity Model</th>
<th>MRR</th>
<th>R@1</th>
<th>R@10</th>
<th>R@30</th>
<th>AL</th>
<th>BM</th>
<th>GE</th>
<th>BL</th>
<th>MG</th>
<th>LU (teacher)</th>
<th>AL</th>
<th>BM</th>
<th>GE</th>
<th>BL</th>
<th>MG</th>
</tr>
</thead>
<tbody>
<tr>
<td>micro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>94.92</td>
<td>91.96</td>
<td>94.64</td>
<td>-</td>
<td>-</td>
<td>98.86</td>
<td>94.19</td>
<td>95.51</td>
<td>90.34</td>
<td>99.01</td>
<td>99.65</td>
</tr>
<tr>
<td>R@1</td>
<td>91.44</td>
<td>82.10</td>
<td>87.81</td>
<td>91.50</td>
<td>87.12</td>
<td>85.80</td>
<td>90.93</td>
<td>92.70</td>
<td>92.70</td>
<td>92.70</td>
<td>90.34</td>
<td>99.01</td>
<td>99.65</td>
<td>99.65</td>
<td>99.65</td>
</tr>
<tr>
<td>R@10</td>
<td><strong>99.34</strong></td>
<td><strong>98.05</strong></td>
<td><strong>99.01</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
<td><strong>99.65</strong></td>
</tr>
<tr>
<td>R@30</td>
<td>99.57</td>
<td>89.87</td>
<td>98.86</td>
<td><strong>99.75</strong></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>macro</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>96.47</strong></td>
<td>93.37</td>
<td>95.43</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>90.69</td>
<td>92.52</td>
<td>92.52</td>
<td>92.52</td>
<td>92.52</td>
</tr>
<tr>
<td>R@1</td>
<td><strong>93.85</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
<td><strong>92.74</strong></td>
</tr>
<tr>
<td>R@10</td>
<td><strong>99.65</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
<td><strong>98.56</strong></td>
</tr>
<tr>
<td>R@30</td>
<td>99.78</td>
<td>99.20</td>
<td>99.20</td>
<td><strong>99.84</strong></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Micro-averaged performance on out-of-domain test sets ACE / MSNBC / AQUAINT and entity collection E_{all}. Significant improvement or degradation with respect to MGAD is indicated (+/-) (p-value ≤ 0.05).

<table>
<thead>
<tr>
<th>Dataset Model</th>
<th>MRR</th>
<th>R@1</th>
<th>R@10</th>
<th>R@30</th>
<th>AL</th>
<th>BM</th>
<th>GE</th>
<th>BL</th>
<th>MG</th>
<th>AL</th>
<th>BM</th>
<th>GE</th>
<th>BL</th>
<th>MG</th>
</tr>
</thead>
<tbody>
<tr>
<td>MRR</td>
<td>82.4-</td>
<td>92.9</td>
<td>92.9</td>
<td>92.9</td>
<td>85.9-</td>
<td>88.4-</td>
<td>82.8-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
</tr>
<tr>
<td>R@1</td>
<td>78.9-</td>
<td>88.8</td>
<td>88.8</td>
<td>88.8</td>
<td>73.7-</td>
<td>84.7-</td>
<td>72.7-</td>
<td>95.0-</td>
<td>95.0-</td>
<td>95.0-</td>
<td>95.0-</td>
<td>95.0-</td>
<td>95.0-</td>
<td>95.0-</td>
</tr>
<tr>
<td>R@10</td>
<td>87.2-</td>
<td>97.0</td>
<td>97.0</td>
<td>97.0</td>
<td>84.7-</td>
<td>95.0-</td>
<td>94.7-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
<td>98.4-</td>
</tr>
<tr>
<td>R@30</td>
<td>88.4-</td>
<td>89.7</td>
<td>89.7</td>
<td>89.7</td>
<td>90.5-</td>
<td>97.1-</td>
<td>95.3-</td>
<td>98.9-</td>
<td>98.9-</td>
<td>98.9-</td>
<td>98.9-</td>
<td>98.9-</td>
<td>98.9-</td>
<td>98.9-</td>
</tr>
</tbody>
</table>

Evaluation Metrics. We follow [Wu et al., 2020b] to report R@K (recall at K) where K=1, 10, 30. Moreover, we take MRR as an additional evaluation metric to evaluate the ranking performance. Since one document often contains multiple mentions, each metric is averaged in two strategies: 1) macro average for each document, and 2) micro average for each mention. We perform significant tests using the paired t-test. Differences are considered statistically significant when the p-value is lower than 0.05.

4.2 Overall Results

In this section, we compare MGAD against all baselines over seen entities and unseen entities separately. The main result of our experiments on the in-domain test set AIDA-testB are summarized in Table 1.

Firstly, we can see that the simple Alias model is a strong baseline that performs better than BM25. Recall that the Alias model works only on prepared alias names, which are often hard to obtain for unseen entities. Moreover, we find that neural methods (GENRE, BLINK, MGAD, and LUKE) generally perform better than heuristic methods (Alias and BM25) on top-ranked results (i.e., MRR and R@1), which indicates that entity retrieval can benefit from the semantic modeling ability of neural methods.

Secondly, we observe that LUKE performs consistently better than BLINK on all seen entities, which demonstrates the effectiveness of distributional representations over descriptive representations for seen entities. However, LUKE relies on large amounts of contexts to learn entity embeddings, making it unable to infer unseen entity embeddings.

Finally, we can see that our model MGAD achieves better performance than BLINK on almost all metrics over seen entities as well as unseen entities. Compared with LUKE, MGAD also obtains comparable performance on seen entities. Moreover, we can see that the relative improvement of MGAD over BLINK is greater in MRR and R@1 than in R@10 and R@30 on unseen entities, which shows the ability of MGAD to rank entities in top positions. All these results demonstrate that MGAD can learn better representations for entities on both seen entities and unseen entities.

4.3 Generalization Analysis

We further analyze the generalization ability of our model on 1) out-of-domain datasets and 2) entities with different frequencies. LUKE is excluded in the following experiments because of the lack of E_{unseen} embeddings.

Generalization On Out-of-domain Datasets. To analyze the generalization ability on out-of-domain datasets, we directly test MGAD on three benchmark datasets, i.e., ACE, MSNBC, and AQUAINT, without further finetuning. Results on these datasets are shown in Table 2. We can observe that: 1) the Alias model performs very well, which is consistently better than BM25. Moreover, it obtains relatively comparable performances to neural retrieval models, e.g., GENRE and BLINK. 2) Comparing GENRE and BLINK, we find that BLINK performs significantly better than GENRE when

$Enc_m(\cdot)$ comes from a LUKE checkpoint, which is pre-trained on Wikipedia but not finetuned on AIDA-train. $\psi$ in Eq. (9) and (12) are the same with $\phi$ in Eq. (5). The best weights of four losses are $\alpha_1=0.338$, $\alpha_2=0.002$, $\alpha_3=0.33$, $\alpha_4=0.33$, where the sum of weights equals to 1. $\alpha_2$ is much smaller since the value of $L_{ea}$ is $10^2$ times greater than others. Temperature $\tau = 1$ in Eq. (8) and (12), and $\tau = 2$ in Eq. (10). The optimizer is Adam [Kingma and Ba, 2015] with learning rate $2 \times 10^{-5}$ and weight decay 0.01, linear-warmup learning rate scheduler with first 10% warmup steps. All models are trained to converge with their best context lengths shown in Section 4.4.

Evaluation Metrics. We follow [Wu et al., 2020b] to report R@K (recall at K) where K=1, 10, 30. Moreover, we take MRR as an additional evaluation metric to evaluate the ranking performance. Since one document often contains multiple mentions, each metric is averaged in two strategies: 1) macro average for each document, and 2) micro average for each mention. We perform significant tests using the paired t-test. Differences are considered statistically significant when the p-value is lower than 0.05.
training and testing on AIDA dataset. While on three out-of-domain test sets, the performance of BLINK drops significantly on ACE and AQUAINT, yet the performance of GENRE only drops slightly on ACE but improves significantly on MSNBC and AQUAINT. This demonstrates the robustness of the generative retrieval model. 3) On ACE and MSNBC, MGAD performs best among all baseline methods on all evaluation metrics. On AQUAINT, we observe that GENRE obtains good performance, this may be that entity names in AQUAINT are often copies or synonyms of mentions where GENRE can benefit from this bias.

**Generalization On Entities With Different Frequencies.** To further study the effectiveness of MGAD, we conduct experiments on AIDA to compare the performances on entities with different frequencies. Specifically, We split mentions into five bins according to the frequencies (on Wikipedia) of gold entities. Results are shown in Figure 3. We find some interesting trends where all three models perform relatively better on entities with much higher frequencies (e.g., frequency large than $10^3$) or unseen entities. This is due to the fact that models can learn better representations for frequent entities. For those unseen entities, they are often distinct and show less ambiguity as they appear less frequent. This tells us that we should pay more attention to those entities with intermediate-frequency since they show the most ambiguity. Moreover, we can see that MGAD is consistently better than GENRE and BLINK on entities with different frequencies, this demonstrates the effectiveness of MGAD.

### 4.4 Ablation Study

To further analyze the impact of 1) four losses in distillation, and 2) context lengths in different mention encoders, we conduct ablation studies on AIDA dataset.

**Loss Function.** To study the impact of different loss functions, we drop one loss from $L$ each time to see how the performance varies. All results are summarized in Table 3. Firstly, for seen entities, we can see that drop $L_{re}$ impacts slightly but drop $L_{sa}$ and $L_{sa}$ would significantly reduce the performance. Moreover, the performance reduces the most while dropping $L_{sa}$, which demonstrates the importance of the space-level alignment loss. Secondly, for unseen entities, we observe that the removal of each loss would reduce the performance significantly, and the impact of $L_{ba}$ is the great-

<table>
<thead>
<tr>
<th></th>
<th>$L$</th>
<th>w/o $L_{re}$</th>
<th>w/o $L_{sa}$</th>
<th>w/o $L_{ba}$</th>
<th>w/o $L_{ea}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>seen</td>
<td>94.1</td>
<td>94.0</td>
<td>92.0</td>
<td><strong>94.7</strong></td>
<td>91.1</td>
</tr>
<tr>
<td>unseen</td>
<td><strong>95.5</strong></td>
<td>93.7</td>
<td>93.7</td>
<td>93.1</td>
<td>94.8</td>
</tr>
<tr>
<td>overall</td>
<td>94.3</td>
<td>94.0</td>
<td>92.2</td>
<td><strong>94.5</strong></td>
<td>91.5</td>
</tr>
</tbody>
</table>

Table 3: Micro-MRR of our model with the removal of each loss on dataset AIDA-testB and entity collection $E_{all}$.

**Context Length.** Since our model relies on mention contexts to learn embeddings for seen entities as the bridge between distributional embedding and descriptional embedding, we further study how the context length affects retrieval performances. To analyze the impact, we finetune mention encoders for GENRE, BLINK and MGAD with different context lengths. As shown in Figure 4, we can observe that with the context growing longer, MGAD and GENRE perform better, but BLINK performs worse. This indicates that the mention encoders of MGAD and GENRE can benefit from capturing long-distance contextual information, but that the mention encoder of BLINK lacks such ability and will be noised on the contrary. However, the worst performance of BLINK is still better than GENRE, which is another evidence of the effectiveness of two-tower models.

### 5 Conclusions and Future Work

In this paper, we present a novel knowledge distillation framework to align descriptional entity embedding space with distributional entity embedding space, and take it to infer enhanced embeddings for unseen entities based only on descriptions. The experiments on four entity retrieval benchmarks demonstrate that our model can learn effective representations for unseen entities. For future work, we would like to analyze and address the ambiguity of intermediate-frequent entities in both entity retrieval and re-ranking stages.
Acknowledgments

This work was funded by the National Natural Science Foundation of China (NSFC) under Grants No. 61902381, No. 62141201, and 61872338, the Youth Innovation Promotion Association CAS under Grants No. 2021100, and 20144310, the Young Elite Scientist Sponsorship Program by CAST (No. YESS20200121), the Lenovo-CAS Joint Lab Youth Scientist Project, and the Foundation and Frontier Research Key Program of Chongqing Science and Technology Commission (No. cstc2017jcyjBX0059).

References

[Bevilacqua et al., 2020] Michele Bevilacqua, Marco Maru, and Roberto Navigli. Generationary or “how we went beyond word sense inventories and learned to gloss”. In EMNLP, 2020.


[Ratinov et al., 2011] Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. Local and global algorithms for disambiguation to Wikipedia. In ACL, 2011.


