Improving Zero-Shot Entity Linking Candidate Generation with Ultra-Fine Entity Type Information

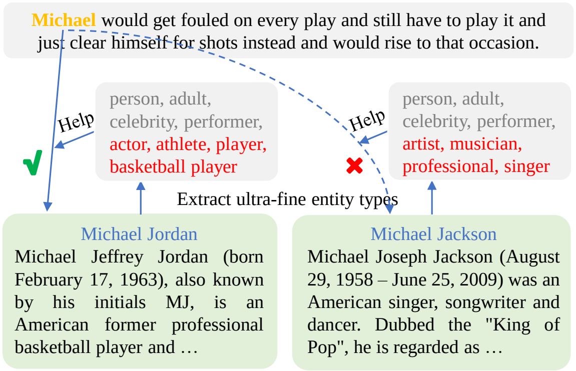
Xuhui Sui[1], **Ying Zhang**[1*], **Kehui Song**[1], **Baohang Zhou**[1],
Guoqing Zhao[2], **Xin Wei**[2], **Xiaojie Yuan**[1]
[1] College of Computer Science, TKLNDST, Nankai University, China
[2] Mashang Consumer Finanace Co, Ltd
{suixuhui,zhangying,songkehui,zhoubaohang}@dbis.nankai.edu.cn
{guoqing.zhao02,xin.wei02}@msxf.com, yuanxj@nankai.edu.cn

Abstract

Entity linking, which aims at aligning ambiguous entity mentions to their referent entities in a knowledge base, plays a key role in multiple natural language processing tasks. Recently, zero-shot entity linking task has become a research hotspot, which links mentions to unseen entities to challenge the generalization ability. For this task, the training set and test set are from different domains, and thus entity linking models tend to be overfitting due to the tendency of memorizing the properties of entities that appear frequently in the training set. We argue that general ultra-fine-grained type information can help the linking models to learn contextual commonality and improve their generalization ability to tackle the overfitting problem. However, in the zero-shot entity linking setting, any type information is not available and entities are only identified by textual descriptions. Thus, we first extract the ultra-fine type information from the entity textual descriptions. Then, we propose a hierarchical multi-task model to improve the high-level zero-shot entity linking candidate generation task by utilizing the entity typing task as an auxiliary low-level task, which introduces extracted ultra-fine type information into the candidate generation task. Experimental results demonstrate the effectiveness of utilizing the ultra-fine entity type information and our proposed method achieves state-of-the-art performance.

1 Introduction

Entity linking (EL) is the task of assigning entity mentions in a text to corresponding entity records in a reference knowledge base. EL plays a key role in the language understanding pipeline, underlying a variety of downstream applications, such as information extraction (Hoffmann et al., 2011; Ji and Nothman, 2016), semantic search (Blanco et al., 2015) and question answering (Berant et al., 2013; Yih et al., 2015; Welbl et al., 2018). In general, EL consists of two phases: candidate generation which generates a set of candidates for each mention from millions of entities, and candidate ranking which retrieves the matched entity for each mention from the set of candidates. As the final results in EL are only generated from candidta sets, the accuracy of the whole EL task is limited by the candidate generation phase. Therefore, in this paper, we focus on the candidate generation phase to set a higher upper bound on EL accuracy.



Michael would get fouled on every play and still have to play it and just clear himself for shots instead and would rise to that occasion.

Help
person, adult, celebrity, performer, actor, athlete, player, basketball player
Help
person, adult, celebrity, performer, artist, musician, professional, singer
Extract ultra-fine entity types
Michael Jordan
Michael Jackson

Michael Jeffrey Jordan (born February 17, 1963), also known by his initials MJ, is an American former professional basketball player and …

Michael Joseph Jackson (August 29, 1958 – June 25, 2009) was an American singer, songwriter and dancer. Dubbed the "King of Pop", he is regarded as …

Figure 1: Examples of entity linking with general ultra-fine-grained entity type information. Different ultra-fine-grained types of the two entities are denoted in red.

Traditional EL approaches usually train models under the setting that linked entities in the test set are available in the training set. However, in many real-world scenarios, labeled data are not easily obtained in multiple domains. Thus, there is a need for EL models to have the capability of generalizing to new domains and new entities. To challenge the generalization ability, a zero-shot entity linking task (Logeswaran et al., 2019) has been proposed, where mentions need to be linked to unseen entities and only the textual information is available. For this task, the training and test sets share different distributions of entities, and thus entity linking models tend to be overfitting due to the tendency of memorizing the properties of entities that appear frequently in the training set.

[*] Corresponding author.

Proceedings of the 29th International Conference on Computational Linguistics, pages 2429–2437
October 12–17, 2022.

We argue that general ultra-fine-grained type information can help the linking models learn contextual commonality and improve their generalization ability to tackle the overfitting problem. If a linking model learns the contextual commonality of *athlete* related entities, it can use similar contextual information to correctly select entities of the same type. Examples of entity linking with general ultra-fine-grained entity type information are shown in Figure 1. A key observation is that the given ultra-fine entity types could have positive effect on the entity linking task. In this example, the type information can help the linking model link the *Michael* in the text to *Michael Jordan* [*actor, athlete, player, basketball player*] instead of *Michael Jackson* [*artist, musician, professional, singer*].

Therefore, in this paper, we try to introduce ultra-fine entity type information into the zero-shot entity linking candidate generation. However, in the zero-shot entity linking setting, any type information is not available and entities are only identified by textual descriptions. Thus, we first extract ultra-fine types from textual descriptions of each entity. In general, more fine-grained entity type information can better help the linking models learn contextual commonality and improve their generalization ability. Thus, we train an entity typing model by utilizing the Ultra-fine Entity Typing dataset (Choi et al., 2018) whose types are more fine-grained rather than other fine-grained entity typing datasets (e.g. FIGER (Ling and Weld, 2012) and OntoNotes (Gillick et al., 2014)), and use the model to extract ultra-fine types from each entity textual description. Then, we propose a hierarchical multi-task model, which jointly models the candidate generation task and ultra-fine entity typing task to introduce the type information extracted by the trained typing model into the candidate generation phase. The ultra-fine entity typing task is utilized as an auxiliary low-level task, providing corresponding type features for the high-level candidate generation task. Our primary motivation is to discover helpful training signals from ultra-fine-grained type information to ensure a more robust zero-shot entity linking candidate generation model.

To summarize, our major contributions are shown as follows:

- To the best of our knowledge, this work is the first to introduce fine-grained type information into zero-shot entity linking task. The fine-grained type information can help the linking models learn contextual commonality and improve their ability to generalize to new domains and unseen entities.
- We first extract ultra-fine entity types for each entity, without depending on additional manually annotated data. Then to take full advantage of extracted type information, we present a hierarchical multi-task model to improve the high-level zero-shot entity linking candidate generation task by utilizing the entity typing task as an auxiliary low-level task.
- Experimental results demonstrate the effectiveness of utilizing the ultra-fine entity type information and our proposed method achieves state-of-the-art performance.

2 Related Work

2.1 Zero-shot Entity Linking

Zero-shot entity linking (Logeswaran et al., 2019) has attracted significant interest from researchers in recent years. In this task, no mentions or entities in the test set have been observed during training and only descriptions of each entity are provided. It consists of two phases: candidate generation (Wu et al., 2020; Ristoski et al., 2021) and candidate ranking (Yao et al., 2020; Tang et al., 2021). In this paper, we focus on the candidate generation phase. (Logeswaran et al., 2019) is the first to formally propose the zero-shot entity linking task and use a traditional IR approach BM25 to generate candidates. BLINK (Wu et al., 2020) uses a bi-encoder architecture to encode mentions and descriptions of entities into dense space to generate candidates, which achieves state-of-the-art results. KG-ZESHEL (Ristoski et al., 2021) utilizes a knowledge graph to extend BLINK. Our work also extends BLINK by introducing auxiliary ultra-fine type information without depending on additional manually annotated data to improve the candidate generation task in zero-shot entity linking.

2.2 Entity Linking with Type Information

Entity typing refers to the act of assigning semantic types to mentions in the text. Fine-grained entity type information has been proven effective in the entity linking process. (Gupta et al., 2017) explores fine-grained entity typing for cross-domain entity linking. (Raiman and Raiman, 2018) proposes a type system to constrain the space in which mentions can be linked. (Onoe and Durrett, 2020)

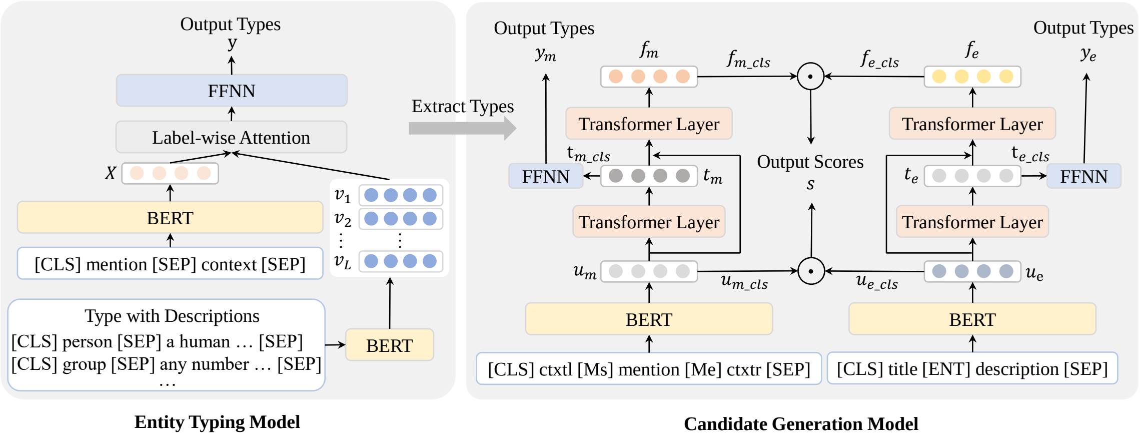


Figure 2: The overall architecture of our proposed model. It consists of two parts: an entity typing model to extract ultra-fine entity types for each entity and a hierarchical multi-task candidate generation model to generate candidate entities with the extracted type information.

converts the cross-domain entity linking task to a very fine-grained entity typing task to generalize across domains effectively. (Hou et al., 2020; Chen et al., 2020) create the semantic embedding for each entity by aggregating entity type embeddings. Inspired by these previous works, to the best of our knowledge, our work is the first to introduce fine-grained type information into zero-shot entity linking task. Also, considering the generalization ability challenge of the zero-shot entity linking task, inspired by (Sanh et al., 2019; Wiatrak and Iso-Sipilä, 2020), our proposed method introduces fine-grained type information in a hierarchical multi-task way to learn contextual commonality and improve the generalization ability.

3 Model

Figure 2 shows our proposed two-stage model, which consists of two parts: the entity typing model and the candidate generation model. In this section, we describe these two models in detail.

3.1 Entity Typing Model

The Entity Typing Model in Figure 2 presents the first part of our proposed model. The goal of this model is to extract ultra-fine entity types from textual description for each entity. Considering off-the-shelf entity typing models (e.g. (Onoe and Durrett, 2019; Onoe et al., 2021)) do not significantly outperform the BERT-based model (Onoe and Durrett, 2019), we simply modify the BERT-based model as our entity typing model.[1] In general, more fine-grained entity type information can better help the linking models learn contextual commonality and improve their generalization ability. Thus, we train the model by utilizing the Ultra-fine Entity Typing dataset (Choi et al., 2018). Then we use the model to extract ultra-fine type information for zero-shot entity linking dataset (Logeswaran et al., 2019). This process does not require any additional manually annotated data.

3.1.1 Label-wise Feature Extraction

Given a mention and its context, we input them to BERT (Devlin et al., 2019) as a sequence pair together with special start and separator tokens ([CLS] mention [SEP] context [SEP]) to extract features. It produces a matrix representation $X = [x_1, x_2, ..., x_n]$ to represent the mention-context pair, where $x_i \in \mathbb{R}^d$ is the word embedding vector for the i-th word, n is the length of the input pair and d is the dimension of hidden states of BERT.

Each entity type has a textual description. Considering that the model training and inference process are on different datasets, we make full use of the type descriptions to improve the generalization ability of the model. To utilize the type description, we input the type and its description to BERT in the form of [CLS] type [SEP] description [SEP]. The embeddings of all the possible labels $C = \{c_1, c_2, ..., c_L\}$ are represented to $V = [v_1, v_2, ..., v_L]$, where $v_i \in \mathbb{R}^d$ is the last hidden layer corresponding to the position of the [CLS] token for the i-th type. Note that the weights of the label embeddings V are fixed after being extracted.

A label-wise attention is utilized to learn individ-

[1] We leave the construction of a more effective entity typing model to future work.

ual representation for each label. The compatibility of label-word pairs is measured as follows:

$$G = VX^T$$

where $G \in \mathbb{R}^{L \times n}$. The attention scores for all the words of mention-context pair with regard to the l-th label are computed via the SoftMax function:

$$a_l = \text{SoftMax}(G_l)$$

where $G_l \in \mathbb{R}^n$ is the compatibility vector of each word and the l-th label. Intuitively, a_l extracts the most relevant information in X about the label l by using attention. Eventually, the label-wise representation is obtained by the weighted aggregation with the attention scores:

$$z_l = \sum_{i=1}^{n} a_{l_i} x_i$$

3.1.2 Multi-label Classification

For the l-th type, the binary prediction $\hat{y}_l$ is computed by: $\hat{y}_l = \text{FFNN}(z_l; \theta_{F_1})$. Each mention is usually associated with a set of types, and a multi-label training objective is required. Thus, the loss is a sum of binary cross-entropy losses over all types over all examples. Finally, we optimize a multi-label binary cross entropy objective:

$$\mathcal{L}_{type} = -\sum_{l=1}^{L} y_l \log \hat{y}_l + (1 - y_l) \log(1 - \hat{y}_l)$$

where y_l takes the value 1 if the l-th type applies to the current mention.

3.2 Candidate Generation Model

After training the Entity Typing Model, we use the model to extract ultra-fine entity types for each entity in the zero-shot entity linking dataset. Our candidate generation model extends BLINK (Wu et al., 2020) bi-encoder by introducing the ultra-fine type information and is shown in Figure 2 Candidate Generation Model, which utilizes a hierarchical multi-task way to jointly learn candidate generation task and ultra-fine entity typing task. The ultra-fine entity typing task is utilized as an auxiliary low-level task at the bottom layer, providing corresponding type features for the high-level candidate generation task at the top layer.

3.2.1 Feature Extraction

Followed (Wu et al., 2020), we use BERT (Devlin et al., 2019) to encode textual input of mentions and entities. The input of each mention $\mathcal{T}_m$ is constructed as follows:

[CLS] ctxtl [Ms] mention [Me] ctxtr [SEP]

where mention, ctxtl, ctxtr are the word-pieces tokens of the mention, context before and after the mention respectively, and [Ms], [Me] are special tokens to tag the mention. The input of each entity $\mathcal{T}_e$ is construct as follows:

[CLS] title [ENT] description [SEP]

where title, description are word-pieces tokens of entity title and description, and [ENT] is a special token to separate the entity title and its description.

Both the context and candidate entity are input to two independent BERT models and are encoded into vectors:

$$u_m = \text{BERT}(\mathcal{T}_m; \theta_{\text{BERT}_1}),\ u_e = \text{BERT}(\mathcal{T}_e; \theta_{\text{BERT}_2})$$

3.2.2 Low-level Entity Typing

The low-level task in our candidate generation model is the ultra-fine entity typing task. We follow (Zhu et al., 2020) to use a binary pairwise relation constraint between mention and each candidate entity. Briefly, a mention and its corresponding entity should share the same type distribution. Thus, the corresponding entity's ground truth types can also be the ground truth types of the mention. We utilize the extracted ultra-fine types from entity descriptions as the target labels of both mention type prediction and entity type prediction tasks. We use two independent transformer layers T as task specific encoders, which takes the extracted features u_m and u_e as input and outputs representations denoted as t_m and t_e:

$$t_m = T(u_m; \theta_{T_1}),\ t_e = T(u_e; \theta_{T_2})$$

Our mention and entity type prediction are similar to the Multi-label Classification in section 3.1.2. For each training pair (m_i, e_i), to predict the l-th type, the binary prediction $\hat{y}^l_{m_i}$ of the mention and $\hat{y}^l_{e_i}$ of the entity are computed respectively by: $\hat{y}^l_{m_i} = \text{FFNN}(t_{m\_cls_i}; \theta_{F_2})$, $\hat{y}^l_{e_i} = \text{FFNN}(t_{e\_cls_i}; \theta_{F_3})$, where $t_{m\_cls_i}$ and $t_{e\_cls_i}$ are the representations corresponding to the position of the [CLS] token of t_{m_i} and t_{e_i} respectively. The losses of mention type prediction and entity type prediction are calculated as follows:

$$\mathcal{L}_{type\_m} = -\sum_{l=1}^{L} y^l \log \hat{y}^l_{m_i} + (1-y^l)\log(1-\hat{y}^l_{m_i})$$

$$\mathcal{L}_{type\_e} = -\sum_{l=1}^{L} y^l \log \hat{y}^l_{e_i} + (1-y^l)\log(1-\hat{y}^l_{e_i})$$

where y^l takes the value 1 if the l-th type is extracted from the description of the entity e_i, and e_i is the corresponding entity of the mention m_i.

3.2.3 High-level Candidate Generation

The high-level task in our candidate generation model is the zero-shot entity linking candidate generation task. It takes the average of the extracted features by BERT u_m, u_e and the low-level task encoder specific output t_m, t_e as the input. We utilizes another two independent transformer layers T as the task-specific encoders:

$$f_m = T(\frac{1}{2}(u_m + t_m); \theta_{T_3}),\ f_e = T(\frac{1}{2}(u_e + t_e); \theta_{T_4})$$

Finally, the score for a given mention m_i and a candidate entity e_i is calculated as the dot product of the corresponding vectors:

$$s(m_i, e_i) = f_{m\_cls_i} \cdot f_{e\_cls_i} + u_{m\_cls_i} \cdot u_{e\_cls_i}$$

where $f_{m\_cls_i}$, $f_{e\_cls_i}$, $u_{m\_cls_i}$ and $u_{e\_cls_i}$ are the representations corresponding to the position of the [CLS] token of f_{m_i}, f_{e_i}, u_{m_i} and u_{e_i} respectively.

Following (Wu et al., 2020), our model is trained on in-batch negatives. Within a batch, the corresponding entity of the mention is the positive sample while other entities in the batch are all negative samples of the mention. Thus, for the candidate scoring, the loss needs to maximize the score of the corresponding entity of the mention in the batch with respect to the other entities of the same batch. To achieve this, for each training pair (m_i, e_i) in a batch of B pairs, the loss is computed as:

$$\mathcal{L}_{(m_i,e_i)} = -s(m_i, e_i) + \log \sum_{j=1}^{B} \exp(s(m_i, e_j))$$

where e_i is the gold entity of the mention m_i.

3.2.4 Hierarchical Multi-task Training

Our model incorporates three objectives, one for candidate scoring and the others for the mention type prediction and the candidate entity type prediction. We jointly optimize these three objectives during our training process. The final loss of the candidate generation model is calculated as follows:

$$\mathcal{L} = \mathcal{L}_{(m_i,e_i)} + \mathcal{L}_{type\_m} + \mathcal{L}_{type\_e}$$

Set	World	Entities	Mentions
Training	American Football	31929	3898
	Doctor Who	40281	8334
	Fallout	16992	3286
	Final Fantasy	14044	6041
	Military	104520	13063
	Pro Wrestling	10133	1392
	StarWars	87056	11824
	World of Warcraft	27677	1437
Validation	Coronation Street	17809	1464
	Muppets	21344	2028
	Ice Hockey	28684	2233
	Elder Scrolls	21712	4275
Test	Forgotten Realms	15603	1200
	Lego	10076	1199
	Star Trek	34430	4227
	YuGiOh	10031	3374

Table 1: Overall statistics of the zero-shot entity linking dataset.

4 Experiments

In this section, we compare our proposed method to other state-of-the-art methods to demonstrate the effectiveness of our model. We first introduce the datasets we used and the implementation details of our model. Then we briefly introduce the baselines and present the overall performance of our model in comparison with others.

4.1 Datasets

We train the entity typing model on the Ultra-Fine Entity Typing dataset (Choi et al., 2018), which has 10331 labels and most of them are defined as free-form text phrases. Each type is marked as one of the three classes: *coarse, fine*, and *ultra-fine*. Note that this classification does not provide explicit hierarchies in the types, and all classes are treated equally during training.

We conduct our experiments mainly on the zero-shot entity linking dataset [2], which is proposed by (Logeswaran et al., 2019) and built using the documents on Wikia [3]. Table 1 shows the overall statistics of the dataset. In this dataset, the entities in the validation and test sets are from different

[2] `https://github.com/lajanugen/zeshel`

[3] `https://www.wikia.com`

Model	Forgotten Realms	Lego	Star Trek	YuGiOh	Macro Recall@64	Micro Recall@64
BM25	83.33	81.23	65.89	60.85	72.83	69.13
BLINK (base)*	90.67	89.99	82.45	71.40	83.63	80.61
BLINK (large)	–	–	–	–	–	82.06
BLINK (large)*	90.92	90.58	84.03	73.30	84.71	82.02
KG-ZESHEL	–	–	–	–	–	82.44
KG-ZESHEL*	91.25	90.40	84.43	73.77	84.96	82.35
Ours (base)	92.08	90.74	83.94	72.00	84.69	81.90
Ours (large)	**92.83**	**91.66**	**85.38**	**74.78**	**86.16**	**83.45**

Table 2: Recall@64 results on the test domains of the zero-shot entity linking dataset. Macro Recall@64 represents the average Recall@64 score of these four test domains. Micro Recall@64 represents the weighted average Recall@64 score of these four domains. * indicates the models are reproduced according to the implementation details in their papers and released codes for a more detailed analysis. We use (base) and (large) to indicate the version of the underlying pre-trained BERT model is BERT-base and BERT-large, respectively. All scores are averaged 5 runs using different random seeds, and our results over all baselines are statistically significant with $p < 0.05$ with the t-test. In the results, the highest values are in bold and the underlined ones are the second highest.

domains compared to the training set, allowing the performance evaluation on entire unseen entities. It uses 8 domains for training, 4 for validation, and 4 for test. The training set has 49,275 labeled mentions while the validation and test sets both have 10,000 unseen mentions.

The samples of this dataset are categorized into 4 categories by (Logeswaran et al., 2019), which are *High Overlap (HO)* whose mention string is identical to its gold entity title, *Multiple Categories (MC)* whose gold entity title is followed by a disambiguation phrase, *Ambiguous substring (AS)* whose mention string is a substring of its gold entity title, and *Low Overlap (LO)* are other mentions. According to the statistics of (Logeswaran et al., 2019), 5% of mentions are categorized as HO, 28% of mentions are MC, 8% of mentions belong to AS, and 59% of mentions are categorized as LO.

4.2 Implementation Details

In our experiments, we use BERT (Devlin et al., 2019) as our base model. The evaluation metric is the recall. We perform our experiments with 5 random seeds and report the average results. And we perform the t-test to demonstrate the statistical significance of our results.

For the entity typing model, the BERT we used to extract mention-context and type description is both the bert-base-uncased (Devlin et al., 2019). We set the maximum sequence length of the input text of mention-context and type description to be 128 and 80, respectively. In this setting, all tokens are covered. The batch size is 32, and the learning rate is 2e-5 with a linear learning rate decay schedule. We use ADAM (Kingma and Ba, 2015) optimization algorithm to optimize our model.

For the candidate generation model, followed (Wu et al., 2020), we use the bert-base-uncased and the bert-large-uncased (Devlin et al., 2019) models respectively. The maximum sequence length of the mention and entity is set to 128. The learning rate is 1e-5 and the batch size is 128. We also train the model by utilizing the ADAM. Our experimental code is available here [4].

4.3 Baselines

For the quantitative evaluation of our proposed model, we use the following state-of-the-art baseline methods for comparison. The first method is BM25 (Robertson and Zaragoza, 2009), which is a traditional IR approach and used by (Logeswaran et al., 2019). The second method is BLINK (Wu et al., 2020), which uses a bi-encoder architecture to encode mentions and entity descriptions into dense space to generate candidates. Our proposed model extends BLINK bi-encoder by introducing ultra-fine entity typing. Comparison results to BLINK could also be regarded as the ablation study to justify the advantage of our proposed model. The last method is KG-ZESHEL (Ristoski et al., 2021), which extends BLINK bi-encoder by utilizing a knowledge graph. Note that we reproduced the BLINK model and the KG-ZESHEL model for a more detailed analysis according to the implementation details in their papers and released codes.

4.4 Overall Performance

The recall@64 results for the candidate generation on the test domains of the zero-shot entity link-

[4] https://github.com/suixuhui/ETZEL

ing dataset are shown in Table 2. We can observe that our proposed model outperforms all baseline models in all test domains and on average. This is consistent with our main claim that our model can improve the performance of the zero-shot entity linking candidate generation task by utilizing the extracted ultra-fine entity type information. Our proposed method utilizes a hierarchical multi-task way to fully mine useful training signals from the low-level ultra-fine entity typing task to help the entity linking candidate generation models learn contextual commonality and improve their generalization ability. This is significant for the zero-shot entity linking setting to generalize the models to new domains and link unseen entities.

We observe that the bi-encoder-based methods perform better than the traditional IR approach BM25, which indicates the effectiveness of our chosen base model. The bi-encoder method achieves state-of-the-art results in the candidate generation task. This approach also allows fast, real-time inference, as the candidate representations can be cached. It can be expected that the BERT-large-based models work better than the BERT-base-based models, due to the larger pre-trained model which encodes more general knowledge. Despite this, we still find that our proposed model with the BERT-base version performs better than the baselines with the BERT-large version in some test domains (e.g., Forgotten Realms and Lego), which further indicates the effectiveness of our proposed method. In general, our proposed method outperforms the baseline approaches for 1.20% and 1.01% on Macro Recall@64 and Micro Recall@64 on the test set, respectively.

5 Analysis

5.1 Top-k Results

The micro recall@k results of the bi-encoder-based models (BLINK, KG-ZESHEL and our proposed model) for the candidate generation task on the test set are shown in Figure 3. We can find that our proposed model consistently outperforms the baseline methods for all k values. The improvement of our model compared to BLINK is pronounced, while the KG-ZESHEL slightly improves the performance of BLINK. This demonstrates the effectiveness of our proposed method. It can also be observed that our proposed model has a relatively significant improvement in the first few candidates (e.g., recall@1, recall@4) over BLINK, allowing for an improvement of more than 3%. This indicates the effectiveness of utilizing extracted ultra-fine entity type information and shows that the ultra-fine entity typing task has strong positive effect on the candidate generation task of zero-shot entity linking. However, as the number of candidates increases, the improvement becomes less and this phenomenon is foreseeable, because there is a tendency for all models to saturate as the number of candidates continues to increase.

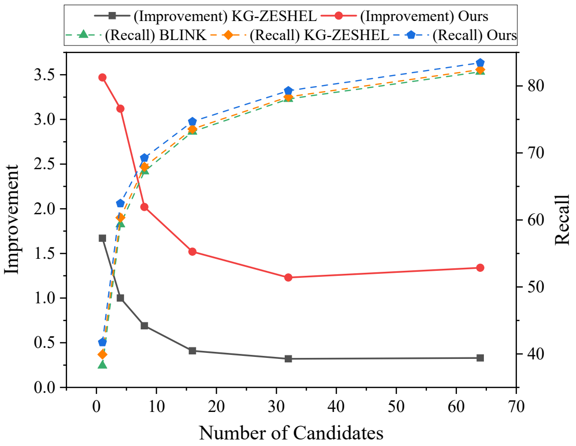


Figure 3: Top-k entity linking recall on test set of zero-shot entity linking dataset. Dashed lines indicate the recall of the three methods and solid lines indicate the relative improvement of our model and KG-ZESHEL compared to the BLINK. The results we choose to report are micro recall@1, recall@4, recall@8, recall@16, recall@32 and recall@64.

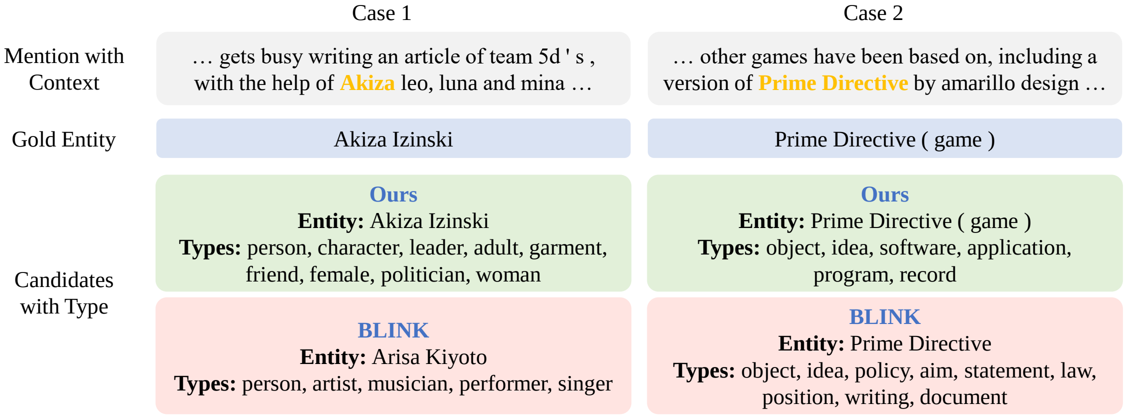
.

Model	HO	MC	AS	LO
BM25	**99.28**	72.54	88.03	54.37
BLINK	98.92	91.04	97.06	74.24
KG-ZESHEL	99.04	91.41	**97.69**	74.52
Ours	99.04	**93.69**	97.27	**75.42**

Table 3: Micro recall@64 scores on the category-specific test subsets including High Overlap (HO), Multiple Categories (MC), Ambiguous Substring (AS) and Low Overlap (LO).

5.2 Results on Category-specific

In addition, we analyze the results of zero-shot entity linking candidate generation models on different categories. There are four categories, and the details about these categories have been described in section 4.1. Table 3 shows the micro recall@64 scores on the category-specific test subsets of the zero-shot entity linking dataset. We find that BM25 outperforms all other methods including our pro-



	Case 1	Case 2
Mention with Context	… gets busy writing an article of team 5d ' s , with the help of **Akiza** leo, luna and mina …	… other games have been based on, including a version of **Prime Directive** by amarillo design …
Gold Entity	Akiza Izinski	Prime Directive (game)
Candidates with Type	**Ours** **Entity:** Akiza Izinski **Types:** person, character, leader, adult, garment, friend, female, politician, woman	**Ours** **Entity:** Prime Directive (game) **Types:** object, idea, software, application, program, record
	BLINK **Entity:** Arisa Kiyoto **Types:** person, artist, musician, performer, singer	**BLINK** **Entity:** Prime Directive **Types:** object, idea, policy, aim, statement, law, position, writing, document

Figure 4: Examples of the compared candidate generation results of our proposed model and the baseline model (BLINK), which are selected from the test set of the zero-shot entity linking dataset. The mentions are denoted in yellow, the text in blue boxes are ground truth entities of the mentions, the candidate entity along with its type extracted by our model and BLINK under the setting of Recall@1 are highlighted in green and pink, respectively.

posed method in HO. This demonstrates the superiority of this traditional IR technique in dealing with cases where the words of the mention string and the entity title are highly overlapping. It can also be observed that KG-ZESHEL performs better than our model in AS, which suggests that using the knowledge graph can be a better choice in some cases. However, the performance of KG-ZESHEL in AS is only slightly higher than our model. Finally, we find that our proposed model improves more in MC and LO. We conjecture that these two categories require more complex reasoning according to the performance of candidate generation models in these two categories is much lower than in the other two categories. This indicates that our proposed model learns more contextual knowledge and has a more powerful reasoning ability to deal with the zero-shot entity linking candidate generation task by utilizing ultra-fine entity type information.

5.3 Case Study

To conduct qualitative analysis, Figure 4 shows two cases from the test set of the zero-shot entity linking dataset. In the case 1, the mention belongs to AS category, whose mention string is a substring of its gold entity title. The BLINK model will point to the entity *Arisa Kiyoto*, while our proposed model will point to the gold entity *Akiza Izinski*. Our model learns the contextual commonality of each type related entities during training by utilizing the ultra-fine entity typing task in a hierarchical multi-task way. At inference time, our model leverages the learned contextual commonality to improve the generalization ability. It will infer that there is some information of gold entity types in the mention with context in case 1 (e.g., *character, leader*, etc. instead of *artist, musician, singer*, etc.). This helps our model point to *Akiza Izinski* instead of *Arisa Kiyoto*. The same is true for case 2, whose gold entity title is followed by a disambiguation phrase and belongs to MC. Our model also infers that there is some information of some types(e.g., *software, application, program*, etc. instead of *policy, law, writing, document*, etc.), which helps our model point to *Prime Directive (game)* instead of *Prime Directive* like BLINK.

6 Conclusion

In this paper, we focus on the zero-shot entity linking task, which links mentions to unseen entities and only the textual information is available. This task challenges the generalization ability and often leads to a tendency of overfitting for entity linking models. To tackle the problem, we introduce the ultra-fine entity type information into the candidate generation phase of this task. Considering only entity description is available, we propose a two-stage model. We first extract ultra-fine entity types from each entity description, without depending on additional manually annotated data. Then we present a hierarchical multi-task model for jointly modeling candidate generation and ultra-fine entity typing, which can help the model to learn contextual commonality of types about the gold entity to improve the generalization ability. The experimental results demonstrate the effectiveness of utilizing the ultra-fine entity type information and our proposed method achieves state-of-the-art performance.

Acknowledgments

This research is supported by the Chinese Scientific and Technical Innovation Project 2030 (2018AAA0102100), the National Natural Science Foundation of China (No. 62272250, U1936206, U1903128).

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, pages 1533–1544.

Roi Blanco, Giuseppe Ottaviano, and Edgar Meij. 2015. Fast and space-efficient entity linking for queries. In *WSDM*, pages 179–188.

Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving entity linking by modeling latent entity type information. In *AAAI*, pages 7529–7537.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *ACL*, pages 87–96.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *CoRR*, abs/1412.1820.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *EMNLP*, pages 2681–2690.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S. Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *ACL*, pages 541–550.

Feng Hou, Ruili Wang, Jun He, and Yi Zhou. 2020. Improving entity linking through semantic reinforced entity embeddings. In *ACL*, pages 6843–6848.

Heng Ji and Joel Nothman. 2016. Overview of TAC-KBP2016 tri-lingual EDL and its impact on end-to-end KBP. In *TAC*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Xiao Ling and Daniel S. Weld. 2012. Fine-grained entity recognition. In *AAAI*.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *ACL*, pages 3449–3460.

Yasumasa Onoe, Michael Boratko, Andrew McCallum, and Greg Durrett. 2021. Modeling fine-grained entity types with box embeddings. In *ACL*, pages 2051–2064.

Yasumasa Onoe and Greg Durrett. 2019. Learning to denoise distantly-labeled data for entity typing. In *NAACL*, pages 2407–2417.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *AAAI*, pages 8576–8583.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *AAAI*, pages 5406–5413. AAAI Press.

Petar Ristoski, Zhizhong Lin, and Qunzhi Zhou. 2021. KG-ZESHEL: knowledge graph-enhanced zero-shot entity linking. In *K-CAP*, pages 49–56.

Stephen E. Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Found. Trends Inf. Retr.*, 3(4):333–389.

Victor Sanh, Thomas Wolf, and Sebastian Ruder. 2019. A hierarchical multi-task approach for learning embeddings from semantic tasks. In *AAAI*, pages 6949–6956.

Hongyin Tang, Xingwu Sun, Beihong Jin, and Fuzheng Zhang. 2021. A bidirectional multi-paragraph reading model for zero-shot entity linking. In *AAAI*, pages 13889–13897.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Trans. Assoc. Comput. Linguistics*, 6:287–302.

Maciej Wiatrak and Juha Iso-Sipilä. 2020. Simple hierarchical multi-task neural end-to-end entity linking for biomedical text. In *EMNLP*, pages 12–17.

Ledell Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *EMNLP*, pages 6397–6407.

Zonghai Yao, Liangliang Cao, and Huapu Pan. 2020. Zero-shot entity linking with efficient long range sequence modeling. In *EMNLP*, pages 2517–2522.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL*, pages 1321–1331.

Ming Zhu, Busra Celikkaya, Parminder Bhatia, and Chandan K. Reddy. 2020. LATTE: latent type modeling for biomedical entity linking. In *AAAI*, pages 9757–9764.