Ontology Enrichment for Effective Fine-grained Entity Typing

Siru Ouyang
University of Illinois Urbana-Champaign
Urbana, IL, USA
siruo2@illinois.edu

Jiaxin Huang
Washington University in St. Louis
St. Louis, MO, USA
jiaxinh@wustl.edu

Pranav Pillai
University of Illinois Urbana-Champaign
Urbana, IL, USA
ppillai3@illinois.edu

Yunyi Zhang
University of Illinois Urbana-Champaign
Urbana, IL, USA
yzhan238@illinois.edu

Yu Zhang
University of Illinois Urbana-Champaign
Urbana, IL, USA
yuz9@illinois.edu

Jiawei Han
University of Illinois Urbana-Champaign
Urbana, IL, USA
hanj@illinois.edu

ABSTRACT

Fine-grained entity typing (FET) is the task of identifying specific entity types at a fine-grained level for entity mentions based on their contextual information. Conventional methods for FET require extensive human annotation, which is time-consuming and costly given the massive scale of data. Recent studies have been developing weakly supervised or zero-shot approaches. We study the setting of zero-shot FET where only an ontology is provided. However, most existing ontology structures lack rich supporting information and even contain ambiguous relations, making them ineffective in guiding FET. Recently developed language models, though promising in various few-shot and zero-shot NLP tasks, may face challenges in zero-shot FET due to their lack of interaction with task-specific ontology. In this study, we propose OnEFET, where we (1) enrich each node in the ontology structure with two categories of extra information: *instance* information for training sample augmentation and *topic* information to relate types with contexts, and (2) develop a coarse-to-fine typing algorithm that exploits the enriched information by training an entailment model with contrasting topics and instance-based augmented training samples. Our experiments show that OnEFET achieves high-quality fine-grained entity typing without human annotation, outperforming existing zero-shot methods by a large margin and rivaling supervised methods. OnEFET also enjoys strong transferability to unseen and finer-grained types. Code is available at https://github.com/ozyyshr/OnEFET.

CCS CONCEPTS

• **Information systems → Information systems applications**;
• **Computing methodologies → Natural language processing**.

KEYWORDS

fine-grained entity typing; zero-shot learning; language models; natural language inference

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08.
https://doi.org/3637528.3671857

ACM Reference Format:
Siru Ouyang, Jiaxin Huang, Pranav Pillai, Yunyi Zhang, Yu Zhang, and Jiawei Han. 2024. Ontology Enrichment for Effective Fine-grained Entity Typing. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24), August 25–29, 2024, Barcelona, Spain.* ACM, New York, NY, USA, 10 pages. https://doi.org/3637528.3671857

1 INTRODUCTION

Fine-grained entity typing (FET) aims to identify the nuanced types of entity mentions within their respective contexts. Serving as a foundational task in text mining, FET plays a pivotal role in converting unstructured data into structured ones by means of entity-linking [3, 47], information extraction [16, 59], and knowledge graph construction [36]. Due to its fundamental role and importance, FET has been researched extensively [46].

FET usually requires typing on a confined label space (e.g., a hierarchical structure of ontology including coarse-grained types and fine-grained ones). Hence, it is time-consuming and labor-intensive to acquire annotations for entity mentions given the huge amount of data. To mitigate the label scarcity issue, some studies [4, 7, 20, 55] explored distantly supervised FET to denoise pseudo labels incorporated from external or parametric knowledge bases. Recent studies made attempts in both few-shot [4, 14] and zero-shot [5, 25] settings where the model is provided with the label hierarchy and optional example annotations for each type. Our work focuses on zero-shot FET, where no annotation for training data is given.

An ontology is a structural view of concepts representing relations among them [26, 39, 42]. For zero-shot FET, existing methods typically leverage the ontology structure to accurately characterize the label space [11, 52]. Several studies integrate ontology structure with external knowledge sources such as Wikipedia label descriptions [23, 29, 58]. However, these approaches often require substantial human annotations on the source domain or manually selected representative mentions. The recent development of large language models (LLMs) has shed new light on FET. These models possess remarkable text representation capabilities [9] and can serve as extensive knowledge sources [31, 33]. However, though showcasing incredible ability in general zero-shot tasks [57], their performance on zero-shot FET [34] is not satisfactory as we will elaborate in Section 5.3 later. This is partly because as the ontology layer goes deeper and the entity types become more fine-grained, language models face increasing challenges in distinguishing between these

Table 1: Enrichment of instance and topic information for entity types (left column), and the corresponding generated sentences using the enriched instances (underlined).

Artist	**Instance**: Leonardo Da Vinci **Topic information**: creativity, art history, style, etc. **Generated Context**: The painting depicts Christ on his way to Calvary, surrounded by angels who are carrying him up into heaven. It has been dated between 1475 and 1480. According to art historian Jos Mareda, it shows "the influence of Leonardo da Vinci", but also that "of Giotto".
Cemetery	**Instance**: Cenotaph **Topic information**: tombstones, grave markers, cremation, etc. **Generated Context**: During her visit to Paris, Emily made sure to explore the famous Père Lachaise, where notable figures such as Oscar Wilde and Jim Morrison were laid to rest.
Sports team	**Instance**: The New York Yankees **Topic information**: training, games, rivalries, etc. **Generated Context**: October 5, 2013: In the final game played at Yankee Stadium, the New York Yankees defeat the Oakland Athletics 7-3 behind four home runs from Alex Rodriguez and six RBI from Mark Teixeira.

types due to their subtle semantic differences. The occurrence frequencies of fine-grained types in the pre-trained corpus are also lower. Hence, how to fully *interpret* and *leverage* the structure of the ontology plays a decisive role in zero-shot FET.

In this paper, we present a novel zero-shot FET framework to enrich the ontology structure by incorporating instance and topic information, and design a coarse-to-fine typing algorithm that effectively utilizes the enriched information. Specifically, we first enrich the original ontology structure with two categories of information in the form of words and phrases (shown in Table 1): (i) **instance information** gives concrete demonstrations of a specific type, which are crucial for training sample augmentation and (ii) **topic information** provides type-associated key phrases to distinguish contexts of similar fine-grained types. Based on the automatically-enriched ontology, our coarse-to-fine entity typing algorithm generates pseudo-training sentences containing enriched instances, by incorporating reward and penalty mechanisms into language model decoding. Then we model entity typing as a natural language inference (NLI) [18, 20] task, where we train the entailment model with contrasting topic information to interpret the fine-grained hierarchical semantics. The proposed framework is evaluated on three challenging fine-grained entity typing benchmarks to verify its effectiveness.

To sum up, this study contributes to the state-of-the-art of fine-grained entity typing in the following aspects:

(1) We propose to automatically enrich the existing ontology structure, where two types of information, *instances* and *topics* are leveraged to help describe and distinguish the semantics of fine-grained types.
(2) We design a zero-shot and ontology-guided entity typing framework to make good use of the enriched information. By generating instance-based pseudo-training data and modeling entity typing as an NLI task, we are able to better distinguish fine-grained entity types and encode label dependency.
(3) Empirical studies verify the general effectiveness of our method on fine-grained entity typing. We also conduct comprehensive analyses for interpreting the performance of OnEFET, hoping to shed light on future related research.
(4) We also demonstrate that OnEFET can work on different test sets, so that OnEFET is free to apply to unseen and even more fine-grained types.

2 PROBLEM FORMULATION

Fine-grained entity typing (FET) [22] aims to determine the type of an entity based on its context. The input is a text sequence of length T, $x = \{w_1, ..., \mathbf{m}, ..., w_T\}$, where $\mathbf{m} = \{w_i, ..., w_j\}$ is an entity mention consisting of $(j - i + 1)$ tokens. The output is an entity type label $y \in \mathcal{Y}$ indicating the entity type of $\mathbf{m}$ from a pre-defined set of entity types. In this paper, we focus on the situation where $\mathcal{Y}$ forms a hierarchical structure, i.e., ontology, which consists of both coarse and fine-grained types. For all the datasets used in this paper, the ontology is a multiway tree, where the upper nodes denote coarse-grained types. The edges in the tree represent the relation of "is-a". An example is visualized in the left part of Figure 1. In zero-shot settings, the annotated labels are not available.

3 METHODOLOGY

In this section, we detail the framework of OnEFET, which consists of two parts: (1) ontology enrichment with instance and topic information, and (2) zero-shot coarse-to-fine typing algorithm. We will first automatically enrich the given ontology structure with topic words and instances. The instances are then used to generate pseudo-training samples for the zero-shot setting, where the topics are infused to train a natural language inference (NLI) model. During the inference stage, we iteratively employ the NLI model in a coarse-to-fine manner in the ontology structure until we finally reach the answer. The overall architecture is shown in Figure 1.

3.1 Automatic Ontology Enrichment

Despite the "is-a" relation existing in ontology structures, we argue that enrichment in an ontology using instances and topics could really help distinguish fine-grained types. Examples of enrichment for both instances and topics are displayed in Table 1.

Enrichment for Topic Information. Topic words and phrases are indicatives of the contextual understanding, and help to disambiguate between different entity types that appear similar [1]. For

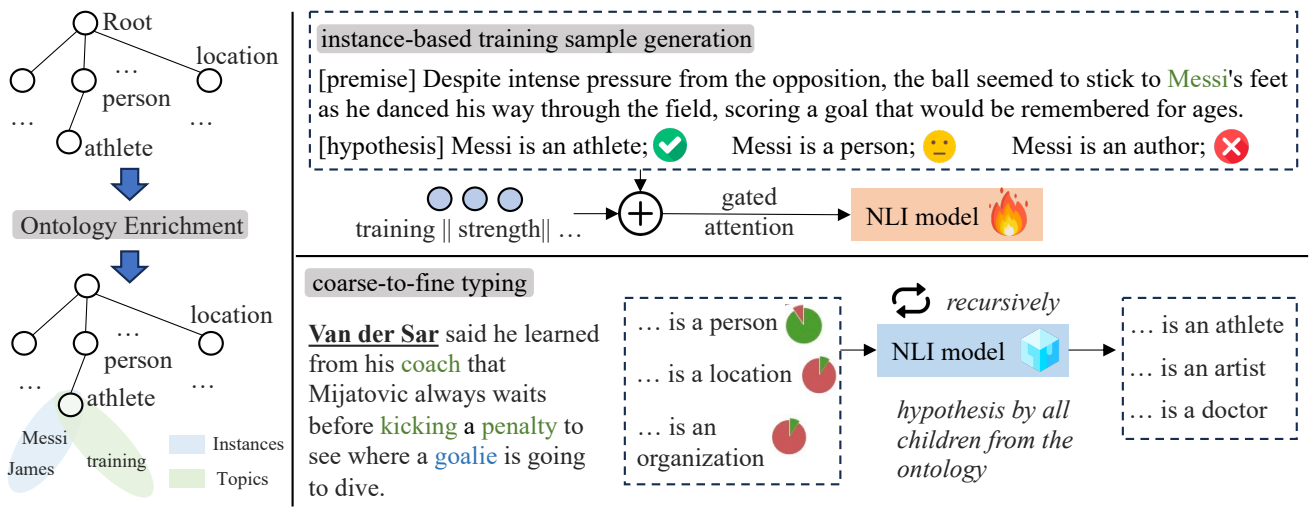

Figure 1: OnEFET framework. The left part is the ontology enrichment for instance and topic information. Then we leverage instances to generate pseudo-training data. Together with topics, we train an NLI model, which will support coarse-to-fine typing during the inference stage.

example, entity types such as biologist and chemist may seem similar, but topic words related to each type such as {*genetics, ecology, viruses, animals*} and {*elements, molecules, equilibrium, catalysts*} can improve the contextual understanding and better distinguish these two types. We employ SeeTopic [53], an unsupervised framework to generate topics using entity types y in the label space $\mathcal{Y}$ as query words. Specifically, SeeTopic[1] takes a text corpus, a set of seeds (query words), and a PLM as input, and outputs the set of topic words/phrases corresponding to each seed in the input. By taking every type node as a seed, we first select the top 20 related documents in a large Wikipedia [2] corpus by implementing Elasticsearch [19]. By only compiling documents related to the query node, we reduce noisy results and can extract relevant topics without intensive memory usage. The retrieved documents serve as the text corpus. We run SeeTopic based on the PLM BERT-base[9], which will return the related topics corresponding to each seed. Importantly, SeeTopic supports out-of-vocabulary search by looking for semantically similar queries via PLM embeddings. This is necessary for automating the process since some entity types are not in the vocabulary of a PLM such as "fraternity_sorority".

Enrichment for Instance Information. Instances are concrete examples of entity types that are in the ontology tree structure. This level of granularity helps detailed classification within a broader categorical structure [45], thus enhancing the understanding of fine-grained entity types. We leverage SECoExpan[3] [54] on the Wikipedia corpus to generate instances, which are divided into the following two stages: (1) using LM-based prompting using question-answering templates [15] for instance seeds curation, and (2) employing SECoExpan over the previously generated seeds to expand instances with the Wikipedia corpus.

For seed generation, we leverage the question-answering framework and the given Wikipedia corpus. Providing the entity type T, we first retrieve related sentences in Wikipedia with T as query words. For every retrieved sentence $\langle S \rangle$ we curate a question-answering template inspired by [15] as follows:

[CLS]What is the instance of $\langle T \rangle$ in this sentence?[SEP]$\langle S \rangle$[SEP]

Here, *[CLS]* is the special classification token, and *[SEP]* is the special token for separation. T denotes the entity type indicated, $\langle S \rangle$ is the retrieved sentence in Wikipedia. An example is shown in the following:

[CLS] What is the instance of $\langle$movie$\rangle$ in this sentence? [SEP] Lepa Shandy was a Nigerian Yoruba movie that was produced by Bayowa and eventually became a very successful project. [SEP]

In the above example, the answer would be "Lepa Sandy". We proceed with this process for n times to get the n total instance seeds for every type T in the ontology structure.

After obtaining instance seeds, we feed them into SECoExpan to automatically mine a sufficient amount of instances for each entity type T based on them.

3.2 Coarse-to-fine Typing

This section presents our ontology-guided zero-shot approach OnEFET based on the restructured ontology. The overall architecture is shown in the right part of Figure 1. Building FET models typically requires training sequence labeling models that recognize entities of specific types under given contexts. First, we construct contextualized training samples for each fine-grained type based on their context-free enriched instances. We then train an entailment model with the enriched information from topic words/phrases to distinguish the fine-grained types. During test time, we make final predictions by inferring with the entailment model in a top-down recursive manner following the ontology structure.

[1] https://github.com/yuzhimanhua/SeeTopic
[2] https://wikipedia.readthedocs.io/en/latest/
[3] https://github.com/yuzhimanhua/SEType

Contextualized Training Data Construction. We propose to construct contextualized training samples [27] for each type by leveraging an LLM to generate a sentence that contains an instance $\boldsymbol{e}$ belonging to that certain type. One intuitive way for generation is to use the instance $\boldsymbol{e}$ as the starting word/phrase. However, such an approach can only create sentences with the instance being at a fixed position, while entities could appear at the beginning, middle, or end of sentences in real practice. Therefore, we propose a rewarding mechanism in LM decoding to generate diverse training samples that contain the instance $\boldsymbol{e}$. Specifically, we use a multiplier to rescale the logits $\boldsymbol{l}_i$ of each token x_i at the output softmax layer:

$$p_{\mathcal{G}}(x_i|\boldsymbol{x}_{<i}) = \frac{\exp(\boldsymbol{l}_i/\omega)}{\sum_{j=1}^{|V|}\exp(\boldsymbol{l}_j/\omega)}, \tag{1}$$

$$\omega = \begin{cases} \tau\alpha & x_i \in \boldsymbol{e} \wedge x_i \notin \boldsymbol{x}_{<i} \\ \tau\beta & x_i \in \boldsymbol{x}_{<i} \\ \tau & \text{else} \end{cases}, \tag{2}$$

where $\omega > 0$ is the sampling temperature ($\omega \to 0$ approximates greedily picking the most probable next token; $\omega \to \infty$ induces a uniform distribution); α and β are scaling hyperparameters of $\tau > 0$: we set $\alpha > 1$ to encourage tokens in the target entity $\boldsymbol{e}$ to have a higher probability of getting generated, and we set $\beta < 1$ to discourage the generation of repetitive tokens already appearing in the sequence $\boldsymbol{x}_{<i}$ to mitigate degenerate repetition, a common issue in text generation. In our experiments, we employ CTRL [17], a 1.6B generative language model for generation and choose "Wikipedia" as the generation style. We select the generated sentences with a higher conditional generation probability than the average [27], and filter out samples that do not contain the given instance.

Following the architecture of NLI [2], we construct training samples for NLI labels with respect to the ontology structure, and contrastively train an entailment model for fine-grained type prediction. Taking the generated sentence $\boldsymbol{x}$ as the "premise", the "hypothesis" is a statement *"[instance] is a [entity type]"*. If the entity type is the original one that generates the instance, the label is set as *Entailment*. For *Contradiction* labels, we choose the fine-grained entity type in other branches. *Neutral* labels are paired with coarse-grained entity types in the same branch to help better distinguish coarse-grained information from fine-grained one.

Training Paradigm. With the generated dataset, we are now able to train an entailment model together with topics to further capture the nuances in fine-grained entity types. Specifically, we build upon the RoBERTa [24] model with the Transformers [41] architecture. The topics are concatenated and sent to the encoder to get embedding H^t. Then a gated attention module is designed to incorporate the topic information (during inference, we use the keywords/phrases in the context as an approximation to topics):

$$\lambda = \sigma(W_\lambda H^t + U_\lambda H^c) \tag{3}$$

where W_λ and U_λ are learnable parameters, and H^c is the context embedding of the input sentence. H^c and H^t are then fused for an effective representation $\tilde{H} = H^c + \lambda H^t$.

One challenge for training the entailment model on the synthetic dataset is that the generated samples may contain noises. Standard supervised training on noisy datasets may raise the risk of overfitting and degraded performance since some easy categories converge faster than hard categories. To improve the noise-robustness of entailment model training for better generalization, we simply apply a noise-robust loss, generalized cross-entropy (GCE) loss [56], which incorporates q-order entropy [10] into the standard cross-entropy loss. Specifically, the calculation of GCE loss for the entailment model $\mathcal{F}$ is as follows:

$$\mathcal{L}_{\text{GCE}} = \sum_{i=1}^{n} \frac{1 - \mathcal{F}_{y_i}(\boldsymbol{x}_i)^q}{q}, \tag{4}$$

where $\mathcal{F}_{y_i}(\boldsymbol{x}_i)$ is the model's predicted probability of sequence $\boldsymbol{x}_i$ belonging to the corresponding entailment label y_i. When $q \to 1$, $\mathcal{L}_{\text{GCE}}$ has better noise-robustness; when $q \to 0$, $\mathcal{L}_{\text{GCE}}$ approximates the standard cross-entropy loss.

Table 2: Detailed statistics of the three fine-grained entity typing datasets.

Datasets	OntoNotes	BBN	FIGER
# of Types	89	46	113
# of Documents	300k	2311	3.1M
# of Entity Mentions	242k	100k	2.7M
# of Train Mentions	223k	84k	2.69M
# of Test Mentions	8963	13766	563
# of total instances	2377	1561	2453

4 EXPERIMENTS

4.1 Experimental Setup

Datasets. In our experiments, we use a wide range of fine-grained entity typing datasets, including OntoNotes [12], BBN [43], and FIGER [22], to verify the effectiveness of our proposed framework. The detailed statistics of three datasets are shown in Table 2. Note that we are directly inferring the test sets in zero-shot settings.

- **OntoNotes.** The OntoNotes dataset is derived from the OntoNotes corpus [44] and 12,017 manual annotations were done by [13] with 3 hierarchical layers of 89 fine-grained types. We follow the dataset split by [40] that retains 8,963 non-pronominal annotations, where the training set is automatically extracted from Freebase API by [38].
- **BBN Pronoun Coreference and Entity Type Corpus (BBN).** This dataset uses 2,311 Wall Street Journal articles and is annotated by [43] with 2 hierarchical layer of 46 types. We follow the split by [38] and also filter out classes with less than 5 annotations in training, validation, and test set, leading to a total of 25 classes.
- **FIGER** The FIGER dataset contains 2M data samples labeled with 113 types. The dev set and the test set include 1,000 and 562 samples respectively. Within its label space, 82 types have a dependency relation with their ancestor or descendant types while the other 30 types are uncategorized free-form words.

Baselines. We compare our framework with baselines from two settings: (i) supervised and distantly-supervised fine-grained methods, and (ii) zero-shot fine-grained methods.

For supervised and distantly supervised fine-grained entity typing, we have the following baselines:

Table 3: Results on the test sets for three benchmark datasets, FIGER, OntoNotes, and BBN. We compare with baselines from two different settings, a fully/distantly-supervised setting, and a zero-shot setting. All results for baselines are taken from original papers. The best results for each setting are highlighted in bold. ChatGPT* indicates results from GPT-3.5-turbo.

Models	FIGER			OntoNotes			BBN		
	Acc.	Micro-F1	Macro-F1	Acc.	Micro-F1	Macro-F1	Acc.	Micro-F1	Macro-F1
Fully / distantly-supervised Setting									
AFET [37]	55.3	66.4	69.3	55.1	64.7	71.1	67.0	73.5	72.7
UFET [6]	-	-	-	59.5	71.8	76.8	-	-	-
BERT-MLMET [8]	-	-	-	67.4	80.4	85.4	-	-	-
LITE [20]	-	**83.3**	**86.7**	-	**80.9**	**86.4**	-	-	-
Zero-Shot Setting									
ProtoZET [25]	29.6	56.4	55.1	28.1	34.5	33.7	25.1	63.1	58.2
OTyper [51]	47.2	67.2	69.1	31.8	36.0	39.1	29.0	48.8	54.4
ZOE [58]	**58.8**	71.3	74.8	50.7	60.8	66.9	61.8	74.9	74.6
DZET [29]	28.5	56.0	55.1	23.1	28.1	27.6	-	-	-
MZET [52]	31.9	57.9	55.5	33.7	43.7	42.3	29.4	68.7	60.6
ChatGPT* [30]	51.7	65.3	58.3	27.7	37.5	32.6	25.1	55.9	50.7
OntoType [18]	49.1	67.4	75.1	65.7	73.4	81.5	-	-	-
OnEFET	56.3	**72.7**	**78.6**	**68.6**	**76.3**	**83.4**	**68.5**	**80.1**	**81.7**

- **AFET** [37] proposes a method for embedding both clean and noisy entity references individually. The technique leverages a defined type hierarchy to formulate loss functions and integrates them into a unified optimization problem to calculate the embeddings of references and type paths.
- **UFET** [6] predicts open types without a pre-defined label structure and is trained using a multi-objective approach that combines supervision from the headwords and prior information from entity linking in Wikipedia.
- **BERT-MLMET** [8] is a model that starts with BERT-base and fine-tunes it using supervision from headwords and entity-type hypernyms extracted from Hearst patterns. The resulting model is used to predict ultra-fine entity types and produce fine-grained entity types by means of a simple type mapping process.
- **LITE** [20] borrows indirect supervision from NLI to perform entity typing. It also involves a type-ranking module to help with generalizing prediction with disjoint type sets.

For zero-shot fine-grained entity typing baselines, we have the following baselines:

- **OTyper** [51] is a neural network trained on a limited training dataset in shallow levels of ontology. This model is then evaluated on the open named entity typing task, which is zero-shot since the fine-grained target types are not known in advance.
- **ZOE** [58] uses a new type taxonomy defined as Boolean functions of Freebase types and determines the type of a given entity reference by linking it to the type-compatible Wikipedia entries.
- **DZET** [29] leverages the type descriptions available from Wikipedia to build a distributed semantic representation of the types and aligns the target entity mention and their corresponding type representations onto the known types.
- **MZET** [52] leverages the semantic meaning and the hierarchical structure into the type representation. With a memory component to model the relationship between the entity mentions and types, the method transfers the knowledge from seen types to label the unseen types.
- **ChatGPT** [30] is a generative large language model aiming for generic AI applications. Pre-trained on the large-scale corpus with reinforcement learning techniques, ChatGPT provides a richer knowledge source for entity typing.
- **OntoType** [18] is an ontology-guided framework that leverages the weak supervision of pre-trained language models and headwords to match the fine-grained types in the ontology.

Experimental details. For each type in the ontology structure, we enrich 30 instances and 5 topics. The number of enriched topics directly adopts the hyperparameter choices from the original topic discovery paper [53], which finds that the quality of top-5 terms is mostly reliable and contains less noise. For the number of instances, we experiment with 10, 20, 30, 40, and 50, with the detailed performance shown Table 4. Setting a number equaling 30 achieves the best results. Potential reasons are (i) we need a sufficient number of pseudo samples in training, and 30 ensures a good number of samples, (ii) too many samples (e.g., 40 or 50 instances) could increase noises (e.g., intrusion between some categories). Also, generating top-40 and top-50 instances per type could take a longer time for instance generation. Note that each type does not always have 30 instances, which is due to the characteristics of each type. For example, "railway" and "time" do not have 30 instances. The total number of instances enriched and the original number of nodes are shown in Table 2. When generating pseudo-training data, we generate 1 contradiction and 1 neutral case for each sample. In this case for neutral, only the direct ancestor is used. When training the entailment model, we set the number of epochs to 10, and the learning rate to $1e-5$. It takes 1 hour to run our method for training a RoBERTa [24] model with 2 Nvidia A6000 GPUs, and around 20 minutes to do inference for OntoNotes. Note that the inference time varies with respect to different numbers of test samples. Results are evaluated based on the metrics described in Appendix A.

Table 4: Experiment results using different number of instances on the test set of OntoNotes dataset.

# instances	10	20	30	40	50
Accuracy	60.9	64.1	68.6	67.4	65.9
Micro-F1	71.0	73.5	76.3	74.9	73.6
Macro-F1	76.4	79.2	83.4	81.8	79.8

4.2 Results

Table 3 presents the performance of all methods on the test set of three benchmark datasets. All the best results are shown in bold. Based on the results, we have the following observations:

(1) OnEFET achieves superior performance on almost all the datasets with dramatic improvements compared with zero-shot baselines. Specifically, OnEFET achieves an absolute improvement of +2.9, +2.9, +1.9 on Accuracy, Micro-F1, and Macro-F1 respectively on the OntoNotes dataset, compared with the previous state-of-the-art zero-shot entity typing model *OntoType*. For the BBN dataset, OnEFET also achieves a consistent performance with a performance boost of +7.7, +5.4, and +6.8 accordingly compared with *ZOE*. For the FIGER dataset, we fall a little behind *ZOE* on Accuracy and Micro-F1 scores. Nevertheless, we achieved a +3.5 gain on the Macro-F1 score. We also observe that OnEFET is competitive compared with supervised or distantly-supervised methods.

(2) We find that the improvement in metric Macro-F1 is much better than that of Micro-F1. Macro-F1 weights every class equally by first calculating P_{Ma} and R_{Ma} and then taking the average over classes. Conversely, Micro-F1 weights every sample equally. The results indicate that OnEFET not only boosts the performance for the majority class labels such as "/person" and "/organization", but also further enhances the typing accuracy on minority class labels such as "/event/war" by a large margin.

(3) We also investigate the performance against a recent LLM, ChatGPT, by evaluating fine-grained entity typing through OpenAI API "gpt-3.5-turbo". Specifically, we use the instruction "*Give the fine-grained entity types of the given entity mentioned in the sentences below. Be concise and you can ONLY use types from this list of possible entity types.[sentence] {sentence} [entity mention] {entity mention}*". We give 3 demonstrations to help with typing. As shown in Table 3, ChatGPT, though deemed promising, proved unsatisfactory in fine-grained entity typing. The reasons are twofold. Firstly, LLMs such as ChatGPT do not have the background of type structure, i.e., ontology, and instead tend to generate free-form entity types, which could be noisy. Also, though we provide strict demonstrations as templates, LLMs tend to generate redundant sentences as output, which limits their accuracy.

(4) We are curious about the phenomenon as to why OnEFET surpasses ZOE by a large margin on OntoNotes and BBN datasets, but falls a little behind on the FIGER dataset. One of the reasons is that there are many nested entity typing problems existing in FIGER dataset. Consider the following example: "Nine of the individuals elected are **UW** faculty members." ZOE directly grounds **UW** in the corresponding Wikipedia entries and will obtain "/organization/educational_institution" almost surely. However, in OnEFET, we leverage prompting predictions as a major source, which may lead to a contextualized understanding of "UW faculty members" and type "UW" with coarse-grained type "/person". We provide a more detailed error analysis in Section 5.3.

Table 5: Ablation studies for different components in OnEFET. The Accuracy and F1 scores are evaluated with the test set of the OntoNotes dataset.

Model	Acc.	Micro-F1	Macro-F1
OnEFET	68.6	76.3	83.4
w/o topics	67.0	74.1	81.9
w/ 3 instances	59.8	70.5	75.6
w/o coarse-to-fine	67.5	75.7	82.1
w/o GCE loss	66.4	74.5	81.6

5 ANALYSIS

5.1 Ablation Study

To dive into the effectiveness of different components in OnEFET, we conduct a comprehensive analysis on the test set of OntoNotes. Table 5 summarizes the results.

Effect of topical enrichment. We explored how performance changes when topics are removed. As shown in Table 5, the performance decreases when we use generated plain text for NLI training, and do not take keywords/phrases into consideration during inference. This result attests to the notion that topical enrichment contributes significantly to understanding and resolving complex semantic relationships in NLI tasks.

Effect of enrichment for instances. We also explored how instance enrichment influences performance. Notably, the training data are constructed from instance information, without which the framework could not be complete as in a zero-shot setting. Therefore, we conducted experiments where only 3 instances were used for training data construction, to approximate the ablation results. The enriched ablation results are shown in Table 5. This indicates that instances also contribute to the final performance.

Effect of inference style. To investigate how coarse-to-fine inference on ontology contributes to the final performance, we conduct another experiment where we treat all the nodes in the ontology at one plain level without any hierarchy. We notice that the performance does not drop as much as we expect, since the traditional NLI model may give more weight to safe answers as coarse-grained types. The potential reason could be our contrastive label design when training the NLI model, which improves the model's ability to capture fine-grained information.

Effect of loss function design. We also tested how the design of GCE loss affects the performance. In this experiment, we replace the GCE loss with traditional cross-entropy loss and everything else remains the same. As shown in Table 5, replacing GCE loss leads to worse performance, and even less accuracy than removing topics. This confirms the necessity of using a noise-tolerant training setting. The reason is that GCE loss can prevent the model from overfitting to false negative labels, thus improving the performance.

5.2 Transferibility Test

To demonstrate OnEFET's ability in entity typing to unseen and even more fine-grained samples, we employ the UFET benchmark

Table 6: Results on the test set of ultra-fine entity typing (UFET) dataset.

Model	P	R	F1
supervised setting			
MLMET [8]	**53.6**	45.3	49.1
LITE [20]	52.4	48.8	**50.6**
zero-shot setting			
direct NLI	1.5	7.1	2.5
direct OnEFET	7.2	17.5	10.2
OnEFET	31.4	**53.1**	39.5

dataset [6] for comparison. The UFET dataset consists of two parts, human-labeled data with 5994 instances split into train/dev/test by 1:1:1, and distant supervision data including 5.2M instances that are automatically labeled by external knowledge bases. There are a total of 10,331 ultra-fine-grained types that are not organized in an ontology. We treat all the types as plain. During inference, we will go over all the types one by one to obtain the NLI label. We focus on the human-labeled data with 1,998 test samples in a zero-shot setting, and follow the original design to evaluate loose macro-averaged precision (P), recall (R), and F1 score. "Direct NLI" refers to the baseline [4] that is pre-trained on the Multi-Genre Natural Language Inference (MNLI) corpus, and then predicted directly without any tuning on the test set. We also introduce the previous state-of-the-art (distantly) supervised models as references. We introduce two ways to apply OnEFET to the UFET dataset: (i) directly employ the trained NLI model with FET datasets to UFET datasets (direct OnEFET); (ii) first generate pseudo training data for each type[5] and then train a new NLI model with these data (OnEFET).

The results are shown in Table 6. We observe that directly employing the previously trained NLI model already achieves competitive performance against the baseline model NLI, which verifies OnEFET's ability to distinguish fine-grained types. There is still a gap between previous supervised state-of-the-art methods due to the difficulty and massive unseen, ultra-fine-grained types. Additionally, we observe a massive performance increase when 5 instances (training samples) are generated for each type in UltraFine. Although we are still around 10 points of F1 behind the state-of-the-art model, we have achieved the best performance of Recall so far. This is due to the characteristics of NLI models, which tend to give more answers that they think are suitable. Despite this, OnEFET still demonstrates its superiority considering that this is actually a 5-shot training in comparison with full fine-tuning of state-of-the-art models.

5.3 Error Analysis of OnEFET and ChatGPT

To shed light on how to further improve OnEFET, we examine the bottleneck of the method by conducting an error analysis here. Specifically, we randomly sampled 100 cases from the OntoNotes test set, and categorized the error cases in the following four types. **Type I: insufficient world knowledge.** Model lacks the commonsense knowledge for the given entity to correctly infer the corresponding type. **Type II: Incorrect fine-grained inference.**

[4]https://huggingface.co/FacebookAI/roberta-large-mnli
[5]Since the label space is large, we generate 5 training samples for each entity type.

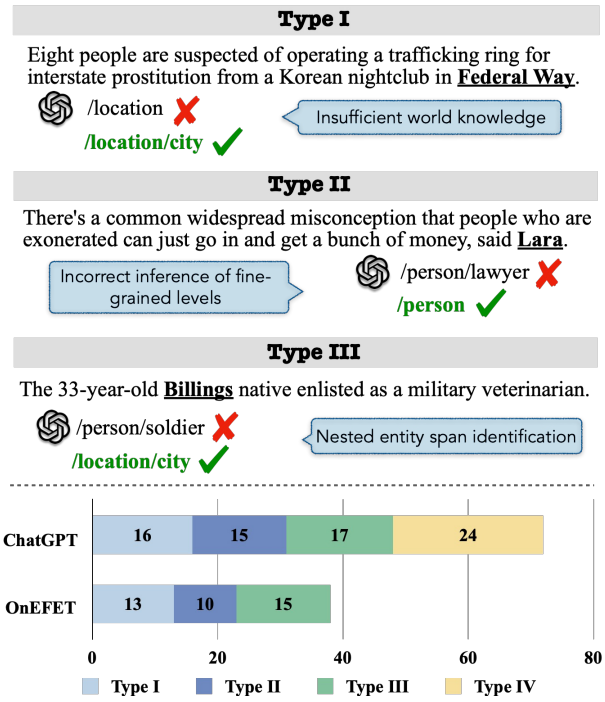

Figure 2: Error types and the corresponding proportions. We manually annotate 100 samples randomly selected from OntoNotes for both ChatGPT and OnEFET.

This error usually results from the understanding of ontology structures, and mistakenly predicts the wrong fine-grained type for an entity. **Type III: Nested entity spans.** Model sometimes assigns mistakenly the types of the entire entity to that of its nested entity. **Type IV: Out-of-vocabulary type prediction.** This type of error happens with ChatGPT, where it generates an out-of-vocabulary entity type even provided with the given ontology structure.

Compared with ChatGPT, OnEFET dramatically reduces the error of incorrect fine-grained inferences, which could be attributed to the coarse-to-fine typing algorithm, and the enrichment of types that help to distinguish the fine-grained semantics. We also observed that even though ChatGPT is loaded with much more comprehensive knowledge bases, OnEFET still wins a little in terms of *Type I* and *Type III*. While direct inference with knowledge bases helps to reduce these two types, OnEFET can delicately infer the correct type with better interpretation of surrounding contexts.

5.4 Case Study

We further conduct a case study compared to some baseline models for an intuitive view as shown in Figure 3. Since ZOE performs the best as to the accuracy in the FIGER dataset, here we randomly grabbed several samples and compare the results generated by OnEFET, ZOE, and the ground truth on the FIGER dataset. We observe that OnEFET performs better than ZOE on these samples. Specifically, OnEFET well interprets the contextualized information. Consider the first example, ZOE maps "Utah" directly to Wikipedia entries, and obtains "location", whereas in the specific context, "Utah" actually means a sports team. For the second and the third

Context & **entity mention**	OnFET predictions	ZOE	Ground truth
The biggest cause for concern for McGuff is the bruised hamstring Regina Rogers suffered against **Utah** last Saturday.	/organization /organization/sports_team	/location	/organization /organization/sports_team
The Fellows Forum, concerned in part with the induction of newly elected fellows, is just one event of the associations annual meeting.	/event	/person /person/politician	/event
"When he left the **Army**, Spencer got a job in Bozeman, where he used acupuncture to save a dog that couldn't walk anymore."	/organization /organization/military	None	/organization /organization/military
After years of wanting to curate such an exhibition, Wieczorek has collaborated with the Henry Art Gallery to feature 26 pieces of **Balth**'s "Videowatercolors."	/person /person/artist /person/artist/painter	/location	/person /person/artist

Figure 3: Case study of OnEFET with baseline model ZOE and ground truth.

example, ZOE even cannot predict the correct coarse-grained type name, and even give invalid answer. In the fourth example, OnEFET takes advantage of the key phrase "Videowatercolors" and infers that "Balth" here refers to a painter, which is even finer than the ground truth annotation.

6 RELATED WORK

Fine-grained Entity Typing with Supervision. The goal of fine-grained entity typing (FET) is to determine the type of entity given a particular context and, optionally, a label hierarchy [22, 50]. Previous studies typically use pre-defined ontology hierarchy and co-occurrence structures estimated from data to enhance the models. To this end, [4, 37, 49] design new loss functions to exploit label hierarchies in the ontology. [28, 40] embed labels into a high-dimension or a new space as representations. [23] explores label dependencies and designs label reasoning mechanisms for effective decoding. There are also studies that exploit the co-occurrence structures including enriching label representations by introducing associated labels [48], requiring latent label representation to reconstruct the co-occurrence structure [21], or limiting the label range for classification during prediction [35]. There are two problems for this research line. First, they neglect the very essential problem inherent in the given ontology, and build blocks to fit in with the potentially problematic structure, which may bring the noise. Second, the experiments are all conducted with direct or indirect supervision, which poses a great challenge and burden for data annotation. To alleviate the data scarcity issue, the setting of zero-shot fine-grained entity typing is introduced.

The most relevant work to us in this research line is LITE [20], which makes use of the indirect supervision from NLI to infer type information. However, LITE still requires supervision for training, which is the first difference from our zero-shot setting. Also, it does not consider the hierarchical structure of ontology as they perform typing, which could limit the performance.

Zero-Shot Fine-Grained Entity Typing. Zero-shot fine-grained entity Typing has been explored widely to alleviate the data scarcity issue. Existing zero-shot FET frameworks infuse multiple sources of information. [52] focuses on capturing the relationship between unseen and seen types in order to produce label representations for the unseen types. [25] enhances label representations by incorporating hierarchical and prototypical information derived from manually selected context-free entities as prototypes. [51] maps mention embeddings to the type embedding space by training a neural model to combine entity and context information. [29, 58] define unseen types by generating type embeddings from Wikipedia descriptions. [5] fuses three external knowledge, contextual, pre-defined hierarchy, and label descriptions. Then three independent modules are trained for the knowledge, whose results are further combined to get the final results. Nonetheless, these zero-shot learning algorithms still require extensive annotations on the source domain or manually selecting high-quality representative mentions.

In zero-shot settings, OntoType [18] stands out as the most closely related work to ours, being an ontology-guided framework. However, OntoType operates on a non-training approach with minimal interaction between the model and the task-specific ontology. Consequently, it struggles with fine-grained type identification, and its headword parsing often falls short in domain-specific tasks.

7 CONCLUSION AND FUTURE WORK

In this paper, we study the problem of zero-shot fine-grained entity typing (FET) via our designed framework OnEFET which leverages the automatically enriched instances and topic information of ontology. We first generate pseudo-training data with instances with reward mechanisms to encourage diversity in the generation. Then we incorporate topic information in self-attention to better align contextualized information. We also use the generalized cross-entropy loss to allow noise in the generated training sentences. Experiments on three benchmark datasets demonstrate that our method outperforms previous zero-shot baselines. Additionally, we also apply OnEFET to unseen and even fine-grained entity types to verify the transferability. For future work, it is of interest to incorporate linguistic clues and contextual information into the hypothesize of NLI models, and to employ a label-binding cross-encoder framework [32] for inference speedup.

ACKNOWLEDGEMENT

Research was supported in part by the Molecule Maker Lab Institute: An AI Research Institutes program supported by NSF under Award No. 2019897, US DARPA KAIROS Program No. FA8750-19-2-1004, INCAS Program No. HR001121C0165, National Science Foundation IIS-19-56151, and the Institute for Geospatial Understanding through an Integrative Discovery Environment (I-GUIDE) by NSF under Award No. 2118329. Any opinions, findings, and conclusions or recommendations expressed herein are those of the authors and do not necessarily represent the views, either expressed or implied, of DARPA or the U.S. Government.

REFERENCES

[1] Yamen Ajjour, Johannes Kiesel, Benno Stein, and Martin Potthast. 2023. Topic Ontologies for Arguments. In *Findings of EACL'23*, Andreas Vlachos and Isabelle Augenstein (Eds.). 1381–1397.

[2] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP'15*. 632–642.

[3] Shuang Chen, Jinpeng Wang, Feng Jiang, and Chin-Yew Lin. 2020. Improving Entity Linking by Modeling Latent Entity Type Information. In *IAAI'20*. 7529–7537.

[4] Tongfei Chen, Yunmo Chen, and Benjamin Van Durme. 2020. Hierarchical Entity Typing via Multi-level Learning to Rank. In *ACL'20*. 8465–8475.

[5] Yi Chen, Haiyun Jiang, Lemao Liu, Shuming Shi, Chuang Fan, Min Yang, and Ruifeng Xu. [n. d.]. An Empirical Study on Multiple Information Sources for Zero-Shot Fine-Grained Entity Typing. In *EMNLP'21*, Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (Eds.). 2668–2678.

[6] Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-Fine Entity Typing. In *ACL'18*. 87–96.

[7] Hongliang Dai, Donghong Du, Xin Li, and Yangqiu Song. 2019. Improving Fine-grained Entity Typing with Entity Linking. In *EMNLP'19*. 6209–6214.

[8] Hongliang Dai, Yangqiu Song, and Haixun Wang. 2021. Ultra-Fine Entity Typing with Weak Supervision from a Masked Language Model. In *ACL'21*. 1790–1799.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL'19*. 4171–4186.

[10] Davide Ferrari. 2008. *Maximum Lq-likelihood estimation*. University of Minnesota.

[11] Yuxia Geng, Jiaoyan Chen, Zhuo Chen, Jeff Z Pan, Zhiquan Ye, Zonggang Yuan, Yantao Jia, and Huajun Chen. 2021. OntoZSL: Ontology-enhanced zero-shot learning. In *WWW'21*. 3325–3336.

[12] Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *arXiv preprint arXiv:1412.1820* (2014).

[13] Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-dependent fine-grained entity type tagging. *arXiv preprint arXiv:1412.1820* (2014).

[14] Jiaxin Huang, Yu Meng, and Jiawei Han. 2022. Few-Shot Fine-Grained Entity Typing with Automatic Label Interpretation and Instance Generation. In *KDD'22*. 605–614.

[15] Yizhu Jiao, Sha Li, Yiqing Xie, Ming Zhong, Heng Ji, and Jiawei Han. 2022. Open-Vocabulary Argument Role Prediction For Event Extraction. In *Findings of EMNLP'22*. 5404–5418.

[16] Yizhu Jiao, Ming Zhong, Sha Li, Ruining Zhao, Siru Ouyang, Heng Ji, and Jiawei Han. 2023. Instruct and Extract: Instruction Tuning for On-Demand Information Extraction. In *EMNLP'23*. 10030–10051.

[17] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858* (2019).

[18] Tanay Komarlu, Minhao Jiang, Xuan Wang, and Jiawei Han. 2023. OntoType: Ontology-Guided Zero-Shot Fine-Grained Entity Typing with Weak Supervision from Pre-Trained Language Models. *arXiv preprint arXiv:2305.12307* (2023).

[19] Oleksii Kononenko, Olga Baysal, Reid Holmes, and Michael W Godfrey. 2014. Mining modern repositories with elasticsearch. In *MSR'14*. 328–331.

[20] Bangzheng Li, Wenpeng Yin, and Muhao Chen. 2022. Ultra-fine entity typing with indirect supervision from natural language inference. *TACL* 10 (2022), 607–622.

[21] Ying Lin and Heng Ji. 2019. An attentive fine-grained entity typing model with latent type representation. In *EMNLP'19*. 6197–6202.

[22] Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *AAAI'12*. 94–100.

[23] Qing Liu, Hongyu Lin, Xinyan Xiao, Xianpei Han, Le Sun, and Hua Wu. 2021. Fine-grained Entity Typing via Label Reasoning. In *EMNLP'21*. 4611–4622.

[24] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[25] Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label Embedding for Zero-shot Fine-grained Named Entity Typing. In *COLING'16*. 171–180.

[26] Yuning Mao, Tong Zhao, Andrey Kan, Chenwei Zhang, Xin Luna Dong, Christos Faloutsos, and Jiawei Han. 2020. Octet: Online catalog taxonomy enrichment with self-supervision. In *KDD'20*. 2247–2257.

[27] Yu Meng, Jiaxin Huang, Yu Zhang, and Jiawei Han. 2022. Generating Training Data with Language Models: Towards Zero-Shot Language Understanding. In *NeurIPS'22*.

[28] Shikhar Murty, Patrick Verga, Luke Vilnis, Irena Radovanovic, and Andrew McCallum. 2018. Hierarchical losses and new resources for fine-grained entity typing and linking. In *ACL'18*. 97–109.

[29] Rasha Obeidat, Xiaoli Fern, Hamed Shahbazi, and Prasad Tadepalli. 2019. Description-based zero-shot fine-grained entity typing. In *NAACL'19*. 807–814.

[30] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. In *NeurIPS'22*. 27730–27744.

[31] Siru Ouyang, Shuohang Wang, Yang Liu, Ming Zhong, Yizhu Jiao, Dan Iter, Reid Pryzant, Chenguang Zhu, Heng Ji, and Jiawei Han. 2023. The Shifted and The Overlooked: A Task-oriented Investigation of User-GPT Interactions. In *EMNLP'23*. 2375–2393.

[32] Shuai Pang, Jianqiang Ma, Zeyu Yan, Yang Zhang, and Jianping Shen. 2020. FASTMATCH: Accelerating the Inference of BERT-based Text Matching. In *COLING'20*. 6459–6469.

[33] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language Models as Knowledge Bases?. In *EMNLP'19*. 2463–2473.

[34] Chengwei Qin, Aston Zhang, Zhuosheng Zhang, Jiaao Chen, Michihiro Yasunaga, and Diyi Yang. 2023. Is ChatGPT a General-Purpose Natural Language Processing Task Solver?. In *EMNLP'23*. 1339–1384.

[35] Maxim Rabinovich and Dan Klein. 2017. Fine-Grained Entity Typing with High-Multiplicity Assignments. In *ACL'17*. 330–334.

[36] Alexander Ratner and Christopher Ré. 2018. Knowledge Base Construction in the Machine-learning Era. *ACM Queue* 16, 3 (2018), 50.

[37] Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016. Afet: Automatic fine-grained entity typing by hierarchical partial-label embedding. In *EMNLP'16*. 1369–1378.

[38] Xiang Ren, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, and Jiawei Han. 2016. Label Noise Reduction in Entity Typing by Heterogeneous Partial-Label Embedding. In *KDD'16*, Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi (Eds.). 1825–1834.

[39] Jiaming Shen, Jinfeng Xiao, Xinwei He, Jingbo Shang, Saurabh Sinha, and Jiawei Han. 2018. Entity set search of scientific literature: An unsupervised ranking approach. In *SIGIR'18*. 565–574.

[40] Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In *EACL'17*. 1271–1280.

[41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS'17*. 5998–6008.

[42] Denny Vrandecic. 2012. Wikidata: a new platform for collaborative data collection. In *WWW'12*, Alain Mille, Fabien Gandon, Jacques Misselis, Michael Rabinovich, and Steffen Staab (Eds.). 1063–1064.

[43] Ralph Weischedel and Ada Brunstein. 2005. BBN pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia* 112 (2005).

[44] Ralph Weischedel, Sameer Pradhan, Lance Ramshaw, Martha Palmer, Nianwen Xue, Mitchell Marcus, Ann Taylor, Craig Greenberg, Eduard Hovy, Robert Belvin, et al. 2011. Ontonotes release 4.0. *LDC2011T03, Philadelphia, Penn.: Linguistic Data Consortium* (2011).

[45] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. 2012. Probase: A probabilistic taxonomy for text understanding. In *SIGMOD'12*. 481–492.

[46] Zilin Xiao, Ming Gong, Paola Cascante-Bonilla, Xingyao Zhang, Jie Wu, and Vicente Ordonez. 2024. Grounding Language Models for Visual Entity Recognition. *arXiv preprint arXiv:2402.18695* (2024).

[47] Zilin Xiao, Ming Gong, Jie Wu, Xingyao Zhang, Linjun Shou, and Daxin Jiang. 2023. Instructed Language Models with Retrievers Are Powerful Entity Linkers. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Houda Bouamor, Juan Pino, and Kalika Bali (Eds.). Association for Computational Linguistics, Singapore, 2267–2282. https://doi.org/10.18653/v1/2023.emnlp-main.139

[48] Wenhan Xiong, Jiawei Wu, Deren Lei, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Imposing Label-Relational Inductive Bias for Extremely Fine-Grained Entity Typing. In *NAACL'19*. 773–784.

[49] Peng Xu and Denilson Barbosa. 2018. Neural Fine-Grained Entity Type Classification with Hierarchy-Aware Loss. In *NAACL'18*. 16–25.
[50] Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. 2012. Hyena: Hierarchical type classification for entity names. In *COLING'12*. 1361–1370.
[51] Zheng Yuan and Doug Downey. 2018. Otyper: A neural architecture for open named entity typing. In *AAAI'18*. 6037–6044.
[52] Tao Zhang, Congying Xia, Chun-Ta Lu, and S Yu Philip. 2020. MZET: Memory Augmented Zero-Shot Fine-grained Named Entity Typing. In *COLING'20*. 77–87.
[53] Yu Zhang, Yu Meng, Xuan Wang, Sheng Wang, and Jiawei Han. 2022. Seed-Guided Topic Discovery with Out-of-Vocabulary Seeds. In *NAACL'22*. 279–290.
[54] Yu Zhang, Yunyi Zhang, Yucheng Jiang, Martin Michalski, Yu Deng, Lucian Popa, ChengXiang Zhai, and Jiawei Han. 2022. Entity Set Co-Expansion in StackOverflow. In *IEEE BigData'22*. 4792–4795.
[55] Yu Zhang, Yunyi Zhang, Yanzhen Shen, Yu Deng, Lucian Popa, Larisa Shwartz, ChengXiang Zhai, and Jiawei Han. 2024. Seed-Guided Fine-Grained Entity Typing in Science and Engineering Domains. In *AAAI'24*. 19606–19614.
[56] Zhilu Zhang and Mert Sabuncu. 2018. Generalized cross entropy loss for training deep neural networks with noisy labels. In *NeurIPS'18*. 8792–8802.
[57] Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting Language Models for Zero-shot Learning by Meta-tuning on Dataset and Prompt Collections. In *Findings of EMNLP'21*. 2856–2878.
[58] Ben Zhou, Daniel Khashabi, Chen-Tse Tsai, and Dan Roth. 2018. Zero-Shot Open Entity Typing as Type-Compatible Grounding. In *EMNLP'18*. 2065–2076.
[59] Sizhe Zhou, Suyu Ge, Jiaming Shen, and Jiawei Han. 2023. Corpus-Based Relation Extraction by Identifying and Refining Relation Patterns. In *ECML/PKDD'23*. 20–38.

A EVALUATION METRICS

Following [22], we evaluate our framework and compare it with baseline models using three metrics: Strict Accuracy (Acc), Micro-F1 (Mi-F1), and Macro-F1 (Ma-F1). Given a set of entity mentions M, ground truths and predicted types are denoted as t_M and $\tilde{t}_M$ respectively. We have the following metrics:

Strict Accuracy. The prediction is considered correct if and only if $t_M = \tilde{t}_M$.

$$Acc = \frac{\sum_{m \in M} \sigma(t_m = \tilde{t}_m)}{|M|}$$

Macro-F1. Macro-F1 is calculated using Macro-Precision (P_{Ma}) and Macro-Recall (R_{Ma}) where

$$P_{\text{Ma}} = \frac{1}{|M|} \sum_{m \in M} \frac{|t_m \cap \tilde{t}_m|}{\tilde{t}_m}$$

$$R_{\text{Ma}} = \frac{1}{|M|} \sum_{m \in M} \frac{|t_m \cap \tilde{t}_m|}{t_m}$$

Micro-F1. Micro-F1 is calculated using Micro-Precision (P_{Mi}) and Micro-Recall (R_{Mi}) where

$$P_{\text{Mi}} = \frac{\sum_{m \in M} |t_m \cap \tilde{t}_m|}{\sum_{m \in M} \tilde{t}_m}$$

$$R_{\text{Mi}} = \frac{\sum_{m \in M} |t_m \cap \tilde{t}_m|}{\sum_{m \in M} t_m}$$