Zero-Shot Open Entity Typing as Type-Compatible Grounding

Ben Zhou[1], **Daniel Khashabi**[2], **Chen-Tse Tsai**[3], **Dan Roth**[2]
[1]University of Illinois, Urbana-Champaign, [2]University of Pennsylvania, [3]Bloomberg LP
xzhou45@illinois.edu, {danielkh,danroth}@cis.upenn.edu, ctsai54@bloomberg.net

Abstract

The problem of entity-typing has been studied predominantly in supervised learning fashion, mostly with task-specific annotations (for coarse types) and sometimes with distant supervision (for fine types). While such approaches have strong performance within datasets, they often lack the flexibility to transfer across text genres and to generalize to new type taxonomies. In this work we propose a *zero-shot* entity typing approach that requires no annotated data and can flexibly identify newly defined types.

Given a *type taxonomy* defined as Boolean functions of FREEBASE "types", we ground a given mention to a set of *type-compatible* Wikipedia entries and then infer the target mention's types using an inference algorithm that makes use of the types of these entries. We evaluate our system on a broad range of datasets, including standard fine-grained and coarse-grained entity typing datasets, and also a dataset in the biological domain. Our system is shown to be competitive with state-of-the-art supervised NER systems and outperforms them on out-of-domain datasets. We also show that our system significantly outperforms other zero-shot fine typing systems.

1 Introduction

Entity type classification is the task of connecting an entity mention to a given set of semantic types. The commonly used type sets range in size and level of granularity, from a small number of coarse-grained types (Tjong Kim Sang and De Meulder, 2003) to over a hundred fine-grained types (Ling and Weld, 2012). It is understood that semantic typing is a key component in many natural language understanding tasks, including Question Answering (Toral et al., 2005; Li and Roth, 2005) and Textual Entailment (Dagan et al., 2010, 2013). Consequently, the ability to type mentions semantically across domains and text genres, and to use a flexible type hierarchy, is essential for solving many important challenges.

Nevertheless, most commonly used approaches and systems for semantic typing (e.g., CORENLP (Manning et al., 2014), COGCOMPNLP (Khashabi et al., 2018), NLTK (Loper and Bird, 2002), SPACY) are trained in a supervised fashion and rely on high quality, task-specific annotation. Scaling such systems to other domains and to a larger set of entity types faces fundamental restrictions.

Coarse typing systems, which are mostly fully supervised, are known to fit a single dataset very well. However, their performance drops significantly on different text genres and even new data sets. Moreover, adding a new coarse type requires manual annotation and retraining. For fine-typing systems, people have adopted a distant-supervision approach. Nevertheless, the number of types used is small: the distantly-supervised FIGER dataset covers only 113 types, a small fraction of most-conservative estimates of the number of types in the English language (the FREEBASE (Bollacker et al., 2008) and WORDNET (Miller, 1995) hierarchies consist of more than $1k$ and $1.5k$ unique types, respectively). More importantly, adapting these systems, once trained, to new type taxonomies cannot be done flexibly.

As was argued in Roth (2017), there is a need to develop new training paradigms that support scalable semantic processing; specifically, there is a need to scale semantic typing to flexible type taxonomies and to multiple domains.

In this work, we introduce ZOE, a *zero-shot* entity typing system, with *open* type definitions. Given a mention in a sentence and a taxonomy of entity types with their definitions, ZOE identifies a set of types that are appropriate for the mention

Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 2065–2076
Brussels, Belgium, October 31 - November 4, 2018.

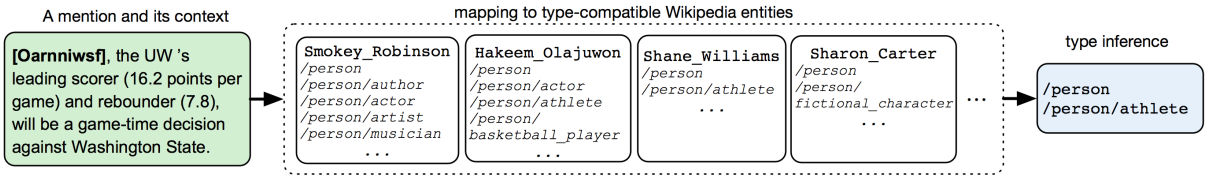

Figure 1: ZOE maps a given mention to its type-compatible entities in Wikipedia and infers a collection of types using this set of entities. While the mention "Oarnniwsf," a football player in the U. of Washington, does not exist in Wikipedia, we ground it to other entities with approximately the same types (§3).

in this context. ZOE does not require any training, and it makes use of existing data resources (e.g., Wikipedia) and tools developed without any task-specific annotation. The key idea is to ground each mention to a set of *type-compatible* Wikipedia entities. The benefit of using a set of Wikipedia titles as an intermediate representation for a mention is that there is much human-curated information in Wikipedia – categories associated with each page, FREEBASE types, and DBpedia types. These were put there independently of the task at hand and can be harnessed for many tasks: in particular, for determining the semantic types of a given mention in its context. In this grounding step, the guiding principle is that type-compatible entities often appear in similar contexts. We rely on contextual signals and, when available, surface forms, to rank Wikipedia titles and choose those that are more compatible with a given mention.

Importantly, our algorithm does not require a given mention to be in Wikipedia; in fact, in many cases (such as nominal mentions) the mentions are not available in Wikipedia. We hypothesize that any entity possible in English corresponds to some type-compatible entities in Wikipedia. We can then rely mostly on the context to reveal a *set* of *compatible titles*, those that are likely to share semantic types with the target mention. The fact that our system is not required to ground to the exact concept is a key difference between our grounding and "standard" Wikification approaches (Mihalcea and Csomai, 2007; Ratinov et al., 2011). As a consequence, while entity linking approaches rely heavily on priors associated with the surface forms and do not consider those that do not link to Wikipedia titles, our system mostly relies on context, regardless of whether the grounding actually exists or not.

Figure 1 shows a high-level visualization of our system. Given a mention, our system grounds it into type-compatible entries in Wikipedia. The target mention "Oarnniwsf," is not in Wikipedia, yet it is grounded to entities with approximately correct types. In addition, while some of the grounded Wikipedia entries are inaccurate in terms of entity types, the resulting aggregated decision is correct.

ZOE is an *open* type system, since it is not restricted to a closed set of types. In our experiments, we build on FREEBASE types as *primitive* types and use them to define types across seven different datasets. Note, however, that our approach is not fundamentally restricted to FREEBASE types; in particular, we allow types to be defined as Boolean formulae over these primitives (considering a type to be a set of entities). Furthermore, we support other primitives, e.g., DBPedia or Wikipedia entries. Consequently, our system can be used across type taxonomies; there is no need to restrict to previously observed types or retrain with annotations of new types. If one wants to use types that are outside our current vocabulary, one only needs to define the target type taxonomy in terms of the primitives used in this work.

In summary, our contributions are as follows:

- We propose a zero-shot open entity typing framework[1] that does not require training on entity-typing-specific supervised data.
- The proposed system outperforms existing zero-shot entity typing systems.
- Our system is competitive with fully-supervised systems in their respective domains across a broad range of coarse- and fine-grained typing datasets, and it outperforms these systems in out-of-domain settings.

2 Related Work

Named Entity Recognition (NER), for which the goal is to discover mention-boundaries in addition to typing, often using a small set of mutu-

[1] https://github.com/CogComp/zoe

ally exclusive types, has a considerable amount of work (Grishman and Sundheim, 1996; Mikheev et al., 1999; Tjong Kim Sang and De Meulder, 2003; Florian et al., 2003; Ratinov and Roth, 2009).

There have been many proposals to scale the systems to support a bigger type space (Fleischman and Hovy, 2002; Sekine et al., 2002). This direction was followed by the introduction of datasets with large label-sets, either manually annotated like BBN (Weischedel and Brunstein, 2005) or distantly supervised like FIGER (Ling and Weld, 2012). With larger datasets available, supervised-learning systems were proposed to learn from the data (Yosef et al., 2012; Abhishek et al., 2017; Shimaoka et al., 2017; Xu and Barbosa, 2018; Choi et al., 2018). Such systems have achieved remarkable success, mostly when restricted to their observed domain and labels.

There is a handful of works aiming to pave the road towards zero-shot typing by addressing ways to extract cheap signals, often to help the supervised algorithms: e.g., by generating gazetteers (Nadeau et al., 2006), or using the anchor texts in Wikipedia (Nothman et al., 2008, 2009). Ren et al. (2016) project labels in high-dimensional space and use label correlations to suppress noise and better model their relations. In our work, we choose not to use the supervised-learning paradigm and instead merely rely on a general entity linking corpus and the signals in Wikipedia. Prior work has already shown the importance of Wikipedia information for NER. Tsai et al. (2016a) use a cross-lingual WIKIFIER to facilitate cross-lingual NER. However, they do not explicitly address the case where the target entity does not exist in Wikipedia.

The zero-shot paradigm for entity typing has only recently been studied. Yogatama et al. (2015) proposed an embedding representation for user-defined features and labels, which facilitates information sharing among labels and reduces the dependence on the labels observed in the training set. The work of Yuan and Downey (2018) can also be seen in the same spirit, i.e., systems that rely on a form of representation of the labels. In a broader sense, such works–including ours–are part of a more general line of work on *zero-shot learning* (Chang et al., 2008; Palatucci et al., 2009; Norouzi et al., 2013; Romera-Paredes and Torr, 2015; Song and Roth, 2014). Our work can be thought of as the continuation of the same research direction.

Approach	Zero-shot?	Use labeled data?
ATTENTIVE (Shimaoka et al., 2017)	No	Yes
AAA (Abhishek et al., 2017)	No	Yes
NFETC-HIER(R) (Xu and Barbosa, 2018)	No	Yes
AFET (Ren et al., 2016)	No	Yes (partial)
PROTOLE (Ma et al., 2016)	**Yes** Prototype Embedding	Yes (partial)
OTYPER (Yuan and Downey, 2018)	**Yes** Word Embedding	Yes (partial)
(Huang et al., 2016)	**Yes** Concept-embedding Clustering	**No**
ZOE (ours)	**Yes** Type-Compatible Concepts	**No**

Table 1: Comparison of recent work on entity typing. Our system does not require any labeled data for entity typing; therefore it works on new datasets without retraining.

A critical step in the design of zero-shot systems is the characterization of the output space. For supervised systems, the output representations are trivial, as they are just indices. For zero-shot systems, the output space is often represented in a high-dimensional space that encodes the semantics of the labels. In OTYPER (Yuan and Downey, 2018), each type embedding is computed by averaging the word embeddings of the words comprising the type. The same idea is also used in PROTOLE (Ma et al., 2016), except that averaging is done only for a few prototypical instances of each type. In our work, we choose to define types using information in Wikipedia. This flexibility allows our system to perform well across several datasets without retraining. On a conceptual level, the work of Lin et al. (2012) and Huang et al. (2016) are close to our approach. The governing idea in these works is to cluster mentions, followed by propagating type information from representative mentions.

Table 1 compares our proposed system with several recently proposed models.

3 Zero-Shot Open Entity Typing

Types are conceptual containers that bind entities together to form a coherent group. Among the entities of the same type, *type-compatibility* creates a network of loosely connected entities:

Definition 1 (Weak Type Compatibility) *Two entities are type-compatible if they share at least one type with respect to a type taxonomy and the contexts in which they appear.*

In our approach, given a mention in a sentence, we aim to discover type-compatible entities in Wikipedia and then infer the mention's types using all the type-compatible entities together. The advantage of using Wikipedia entries is that the rich information associated with them allows us to infer the types more easily. Note that this problem is different from the standard entity linking or Wikification problem in which the goal is to find the corresponding entity in Wikipedia. Wikipedia does not contain all entities in the world, but an entity is likely to have at least one type-compatible entity in Wikipedia.

In order to find the type-compatible entities, we use the context of mentions as a proxy. Defining it formally:

Definition 2 (Context Consistency) *A mention m (in a context sentence s) is* context-consistent *with another well-defined mention m', if m can be replaced by m' in the context s, and the new sentence still makes logical sense.*

Hypothesis 1 *Context consistency is a strong proxy for type compatibility.*

Based on this hypothesis, given a mention m in a sentence s, we find other context-compatible mentions in a Wikified corpus. Since the mentions in the Wikified corpus are linked to the corresponding Wikipedia entries, we can infer m's types by aggregating information associated with these Wikipedia entries.

Figure 2 shows the high-level architecture of our proposed system. The inputs to the system are a mention m in a sentence s, and a type definition T. The output of the system is a set of types $\{t_{\text{Target}}\} \subseteq T$ in the target taxonomy that best represents the given mention. The type definitions characterize the target entity-type space. In our experiments, we choose to use FREEBASE types to define the types across 7 datasets; that is, T is a mapping from the set of FREEBASE types to the set of target types: $T : \{t_{\text{FB}}\} \rightarrow \{t_{\text{Target}}\}$. This definition comprises many atomic definitions; for example, we can define the type `location` as the disjunction of FREEBASE types like `FB.location` and `FB.geography`:

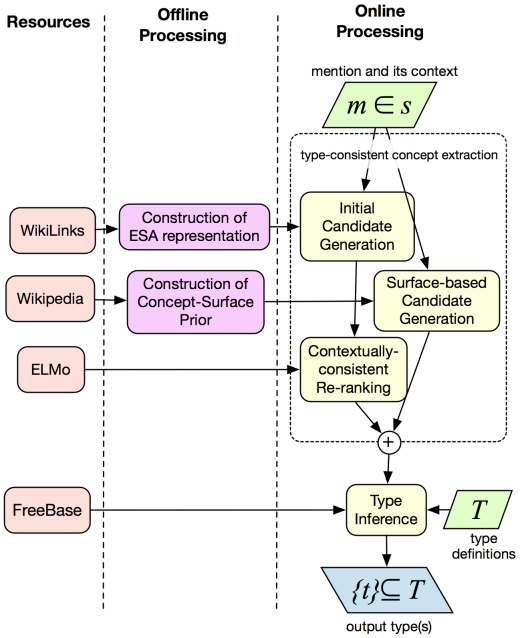

Figure 2: A high-level view of our approach. The inputs to the system are a mention m in a context s, and type definitions T. The output is set of types $\{t\}$ in the type definition. The figure also highlights the input resources, as well as offline and online processes.

```
T (Type Definitions)
- Target.company := FB.business &&
                    !(FB.education || FB.government)
- Target.location := FB.location || FB.geography ||
                     FB.casino || FB.court
- ...
```

The type definitions of a dataset reflect the understanding of a domain expert and the assumptions made in dataset design. Such definitions are often much cheaper to define, than to annotate full-fledged supervised datasets. It is important to emphasize that, to use our system on different datasets, one does not need to retrain it; there is one single system used across different datasets, working with different type definitions.

For notational simplicity, we define a few conventions for the rest of the paper. The notation $t \in T$, simply means t is a member of the image of the map T (i.e., t is a member of the target types). For a fixed concept c, the notation $T(c)$ is the application of $T(.)$ on the FREEBASE types attached to the concept c. For a collection of concepts C, $T(C)$ is defined as $\bigcup_{c \in C} T(c)$. We use $T_{\text{coarse}}(.)$ to refer to the subset of coarse types of $T(.)$, while $T_{\text{fine}}(.)$ defines the fine type subset.

Components in Figure 2 are described in the following sections.

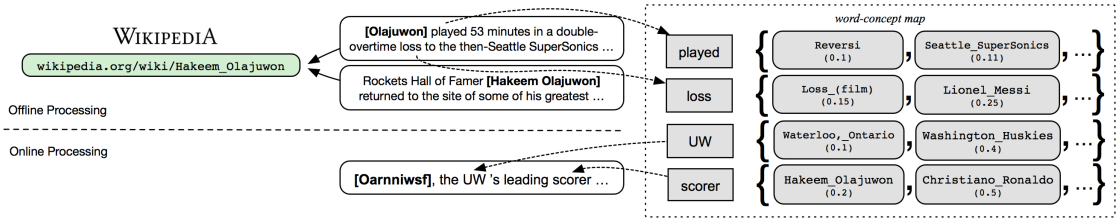

Figure 3: Extraction of topically relevant concepts. Word-concept map is pre-computed using WIKILINKS and used to retrieve the most relevant concepts for a given mention (see §3.1).

3.1 Initial Concept Candidate Generation

Given a mention, the goal of this step is to quickly generate a set of Wikipedia entries based on other words in the sentence. Since there are millions of entries in Wikipedia, it is extremely inefficient to go through all entries for each mention. We adopt ideas from explicit semantic analysis (ESA) (Gabrilovich and Markovitch, 2007), an approach to representing words with a vector of Wikipedia *concepts*, and to providing fast retrieval of the relevant Wikipedia concepts via inverted indexing.

In our construction we use the WIKILINKS (Singh et al., 2012) corpus, which contains a total of 40 million mentions over 3 million concepts. Each mention in WIKILINKS is associated with a Wikipedia concept. To characterize it formally, in the WIKILINKS corpus, for each concept c, there are example sentences $\mathsf{sent}(c) = \{s_i\}$.

Offline computation: The first step is to construct an ESA representation for each word in the WIKILINKS corpus. We create a mapping from each word in the corpus to the relevant concepts associated with it. The result is a map $\mathcal{S}$ from tokens to concepts: $\mathcal{S} : w \rightarrow \{c, \mathsf{score}(c|w)\}$ (see Figure 3), where $\mathsf{score}(c|w)$ denotes the association of the word w with concept c, calculated as the sum of the TF-IDF values of the word w in the sentences describing c:

$$\mathsf{score}(c|w) \triangleq \sum_{s \in \mathsf{sent}(c)} \sum_{w \in s} \text{tf-idf}(w, s).$$

That is, we treat each sentence as a document and compute TF-IDF scores for the words in it.

Online computation: For a given mention m and its sentence context s, we use our offline word-concept map $\mathcal{S}$ to find the concepts associated with each word, and aggregate them to create a single list of weighted concepts; i.e., $\sum_{w \in s} \mathcal{S}(w)$. The resulting concepts are sorted by the corresponding weights, and the top ℓ_{ESA} candidates form a set C_{ESA} which is passed to the next step.

3.2 Context-Consistent Re-Ranking

After quick retrieval of the initial concept candidates, we re-rank concepts in C_{ESA} based on context consistency between the input mention and concept mentions in WIKILINKS.

For this step, assume we have a representation that encodes the sentential information anchored on the mention. We denote this mention-aware context representation as $\mathsf{SentRep}(s|m)$. We define a measure of consistency between a concept c and a mention m in a sentence s:

$$\mathsf{Consistency}(c, s, m) = \mathsf{cosine}(\mathsf{SentRep}(s|m), \mathsf{ConceptRep}(c)), \quad (1)$$

where $\mathsf{ConceptRep}(c)$ is representation of a concept:

$$\mathsf{ConceptRep}(c) \triangleq \text{avg}_s \left(\mathsf{SentRep}(s|c) \middle| s \in \text{WIKILINKS}, c \in s \right),$$

which is the average vector of the representation of all the sentences in WIKILINKS that describe the given concept.

We use pre-trained ELMO (Peters et al., 2018), a state-of-the-art contextual and mention-aware word representation. In order to generate $\mathsf{SentRep}(s|m)$, we run ELMO on sentence s, where the tokens of the mention m are concatenated with "_", and retrieve its ELMO vector as $\mathsf{SentRep}(s|m)$.

According to the consistency measure, we select the top ℓ_{ELMO} concepts for each mention. We call this set of concepts C_{ELMO}.

3.3 Surface-Based Concept Generation

While context often is a key signal for typing, one should not ignore the information included in the surface form of the mentions. If the corresponding concept or entity exists in Wikipedia, many mentions can be accurately grounded with only trivial prior probability $\mathbf{Pr}(\text{concept}|\text{surface})$. The prior distribution is pre-computed by calculating the fre-

quency of the times a certain surface string refers to different concepts within Wikipedia.

In the test time, for a given mention, we use the pre-computed probability distribution to obtain the most likely concept, $c_{\text{surf}} = \arg\max_c \mathbf{Pr}(c|m)$, for the given mention m.

3.4 Type Inference

Our inference algorithm starts with selection of concepts, followed by inference of coarse and fine types. Our approach is outlined in Algorithm 1 and explained below.

Concept inference. To integrate surface-based and context-based concepts, we follow a simple rule: if the prior probability of the surface-based concept (c_{surf}) has confidence below a threshold λ, we ignore it; otherwise we include it among the concepts selected from context (C_{ELMo}), and only choose coarse and fine types from c_{surf}.

To map the selected concepts to the target entity types, we retrieve the FREEBASE-types of each concept and then apply the type definition T (defined just before §3.1). In Algorithm 1, the set of target types of a concept c is denoted as $T(c)$. This is followed by an aggregation step for selection of a coarse type $t_{\text{coarse}} \in T_{\text{coarse}}(.)$, and ends with the selection of a set of fine types $\{t_{\text{fine}}\} \subseteq T_{\text{fine}}(.)$.

Coarse type inference. Our type inference algorithm works in a relatively simple confidence analysis procedure. To this end, we define $\mathsf{Count}(t; C)$ to be the number of occurrences of type t in the collection of concepts C:

$$\mathsf{Count}(t; C) := |\{c : c \in C \text{ and } t \in T(c)\}|.$$

In theory, for a sensible type t, the count of context-consistent concepts that have this type should be higher than that of the initial concept candidates. In other words, $\frac{\mathsf{Count}(t;C_{\text{ELMo}})/\ell_{\text{ELMo}}}{\mathsf{Count}(t;C_{\text{ESA}})/\ell_{\text{ESA}}} > 1$. We select the first concept (in the C_{ELMo} ranking) which has some coarse type that matches this criterion. If there is no such concept, we use the coarse types of the highest scoring concept. To select one of the coarse types of the selected concept, we let each concept of C_{ELMo} vote based on its consistency score. We name this voting-based procedure $\mathsf{SelectCoarse}(c)$, which selects one coarse type from a given concept:

$$\mathsf{SelectCoarse}(c) \triangleq \operatorname*{argmax}_t \sum_{\tilde{c} \in C_{\text{ELMo}}} \sum_{t \in T_{\text{coarse}}(\tilde{c})} \mathsf{Consistency}(\tilde{c}, s, m),$$

Algorithm 1: Type inference algorithm

Input mention m in sentence s, retrieved concepts $C_{\text{ESA}}, C_{\text{ELMo}}, c_{\text{surf}}$, and type definition T
Output Inferred types t_{coarse} and $\{t_{\text{fine}}\}$.
Define, $r(t, t'; C, C') := \frac{\mathsf{Count}(t;C)/|C|}{\mathsf{Count}(t';C')/|C'|}$,
$r(t; C, C') := r(t, t; C, C')$,
$r(t, t'; C,) := r(t, t'; C, C)$.

$\tau_{\text{surf}} \leftarrow \{t | t \in T_{\text{coarse}}(c_{\text{surf}}), r(t; C_{\text{ELMo}}, C_{\text{ESA}}) > 1\}$
if $\boldsymbol{Pr}(c_{surf}|m) \geq \lambda$ **and** $\tau_{surf} \neq \emptyset$ **then**
  $t_{\text{coarse}} \leftarrow \mathsf{SelectCoarse}(c_{\text{surf}})$
  $\tilde{C} \leftarrow \{c_{\text{surf}}\} \cup C_{\text{ELMo}}$
  $\{t_{\text{fine}}\} \leftarrow \left\{ t_f \;\middle|\; \begin{array}{l} t_f \in T_{\text{fine}}(c_{\text{surf}}), \\ \text{compatible w/ } t_{\text{coarse}} \text{ and,} \\ r(t_f, t_{\text{coarse}}; \tilde{C}) \geq \eta_s \end{array} \right\}$
else
  $\tilde{C}_{\text{ELMo}} \leftarrow \left\{ c \;\middle|\; \begin{array}{l} c \in C_{\text{ELMo}}, \exists t \in T_{\text{coarse}}(c) \\ r(t; C_{\text{ELMo}}, C_{\text{ESA}}) > 1 \end{array} \right\}$
  if $\tilde{C}_{\text{ELMo}} = \emptyset$ **then**
    $\tilde{c} \leftarrow \operatorname{argmax}_{c \in C_{\text{ELMo}}} \mathsf{Consistency}(c, s, m)$
  else
    $\tilde{c} \leftarrow \operatorname{argmax}_{c \in \tilde{C}_{\text{ELMo}}} \mathsf{Consistency}(c, s, m)$
  end
  $t_{\text{coarse}} \leftarrow \mathsf{SelectCoarse}(\tilde{c})$
  $\{t_{\text{fine}}\} \leftarrow \left\{ t_f \;\middle|\; \begin{array}{l} t_f \in T_{\text{fine}}(C_{\text{ELMo}}), \\ \text{compatible w/ } t_{\text{coarse}} \text{ and,} \\ r(t_f, t_{\text{coarse}}; C_{\text{ELMo}}) \geq \eta_c \end{array} \right\}$
end

where consistency is defined in Equation (1).

Fine type inference. With the selected coarse type, we take only the fine types that are compatible with the selected coarse type (e.g., the fine type `/people/athlete` and the coarse type `/people` are compatible).

Among the compatible fine types, we further filter the ones that have better support from the context. Therefore, we select the fine types t_f such that $\frac{\mathsf{Count}(t_f;C_{\text{ELMo}})}{\mathsf{Count}(t_c;C_{\text{ELMo}})} \geq \eta$, where t_c is the previously selected coarse type which is compatible with t_f. Intuitively, the fraction filters out the fine-grained candidate types that don't have enough support compared to the selected coarse type.

4 Experiments

Empirically, we study the behavior of our system compared to published results. All the results are reproduced except the ones indicated by $*$, which are directly cited from their corresponding papers.

Datasets. In our experiments, we use a wide range of typing datasets:

- For **coarse** entity typing, we use MUC (Grishman and Sundheim, 1996), CoNLL (Tjong Kim Sang and De Meulder, 2003), and OntoNotes (Hovy et al., 2006).

	Approach	Trained on	FIGER Acc.	FIGER $F1_{ma}$	FIGER $F1_{mi}$	BBN Acc.	BBN $F1_{ma}$	BBN $F1_{mi}$	OntoNotes$_{fine}$ Acc.	OntoNotes$_{fine}$ $F1_{ma}$	OntoNotes$_{fine}$ $F1_{mi}$
Closed Type	AFET* (Ren et al., 2016)	FIGER	53.3	69.3	66.4	-	-	-	-	-	-
	NFETC-HIER(R)*(Xu and Barbosa, 2018)	FIGER	68.9	81.9	79.0	-	-	-	-	-	-
	AFET* (Ren et al., 2016)	BBN	-	-	-	68.3	74.4	74.7	-	-	-
	AAA* (Abhishek et al., 2017)	BBN	-	-	-	73.3	79.1	79.2	-	-	-
	AFET* (Ren et al., 2016)	OntoNotes$_{fine}$	-	-	-	-	-	-	55.1	71.1	64.7
	NFETC-HIER(R)* (Xu and Barbosa, 2018)	OntoNotes$_{fine}$	-	-	-	-	-	-	60.2	76.4	70.2
Open Type	OTYPER (Yuan and Downey, 2018)	FIGER	47.2	69.1	67.2	27	50.3	49.5	31.6	34.5	32.1
		BBN	5.3	11.5	11.5	29	54.4	48.8	2.5	5.1	5.4
		OntoNotes$_{fine}$	0.4	15.6	16.8	23.6	51.1	47.9	31.8	39.1	36
	ELMONN	×	21.5	57.7	53.8	49.3	68.4	66.2	0.5	21.2	21.8
	WIKIFIERTYPER	×	17.2	33.3	46.2	45.8	52.3	66.1	47.8	65.6	58.2
	ZOE (ours)	×	**58.8**	**74.8**	**71.3**	**61.8**	**74.6**	**74.9**	**50.7**	**66.9**	**60.8**

Table 2: Evaluation of fine-grained entity-typing: we compare our system with state-of-the-art systems (§4.1) For each column, the best zero-shot and overall results are **bold-faced** and underlined, respectively. Numbers are $F1$ in percentage. For supervised systems, we report their in-domain performances, since they do not transfer to other datasets with different labels. For OTYPER, cells with gray color indicate *in-domain* evaluation, which is the setting in which it has the best performance. Our system outperforms all the other zero-shot baselines, and achieves competitive results compared to the best supervised systems.

System	Trained on	OntoNotes PER	OntoNotes LOC	OntoNotes ORG	CoNLL PER	CoNLL LOC	CoNLL ORG	MUC PER	MUC LOC	MUC ORG
COGCOMPNLP	OntoNotes	98.4	91.9	97.7	83.7	70.1	68.3	82.5	76.9	86.7
COGCOMPNLP	CoNLL	**94.4**	59.1	**87.8**	95.6	92.9	90.5	**90.8**	90.8	90.9
ZOE (ours)	×	88.4	**70.0**	85.6	**90.1**	**80.1**	**73.9**	87.8	**90.9**	**91.2**

Table 3: Evaluation of coarse entity-typing (§4.2): we compare two supervised entity-typers with our system. For the supervised systems, cells with gray color indicate *in-domain* evaluation. For each column, the best, out-of-domain and overall results are **bold-faced** and underlined, respectively. Numbers are $F1$ in percentage. In most of the out-of-domain settings our system outperforms the supervised system.

- For **fine** typing, we focus on FIGER (Ling and Weld, 2012), BBN (Weischedel and Brunstein, 2005), and OntoNotes$_{fine}$ (Gillick et al., 2014).
- In addition to the news NER, we use the BB3 dataset (Delėger et al., 2016), with contain mentions of *bacteria* or other notions, extracted from sentences of scientific papers.

ZOE's parameters. We use different type definitions for each dataset. In order to design type definitions for each dataset, we follow in the footsteps of Abhishek et al. (2017) and randomly sample 10% of the test set. For the experiments, we exclude the sampled set. For completeness, we have included the type definitions of the major experiments in Appendix D.

The parameters are set universally across different experiments. For parameters that determine the number of extracted concepts, we use $\ell_{\text{ESA}} = 300$ and $\ell_{\text{ELMO}} = 20$, which are based on the upper-bound analysis in Appendix A. For other parameters, we set $\lambda = 0.5$, $\eta_s = 0.8$ and $\eta_c = 0.3$, based on the FIGER dev set. We emphasize that these parameters are universal across our evaluations.

Evaluation metrics. Given a collection of mentions M, denote the set of gold types and predicted types of a mention $m \in M$ as $T_g(m)$ and $T_p(m)$ respectively. We define the following metrics for our evaluations:

- Strict Accuracy (Acc.): $\frac{|\{m|T_g(m)=T_p(m)\}|}{|M|}$.
- Macro F1 ($F1_{\text{ma}}$): Macro Precision is defined as $\frac{1}{|M|}\sum_{m\in M}\frac{|T_p(m)\cap T_g(m)|}{|T_p(m)|}$. With this, the definitions of Macro recall and F1 follow.
- Micro F1 ($F1_{\text{mi}}$): The precision is defined as $\frac{\sum_{m\in M}|T_p(m)\cap T_g(m)|}{\sum_{m\in M}|T_p(m)|}$, and the Micro recall and F1 follow the same pattern.

In the experiment in §4.3, to evaluate systems on unseen types we used modified versions of metrics. Let $G(t)$ be the number of mentions with gold type t, $P(t)$ be the number of mentions predicted to have type t, $C(t)$ be the number of mentions correctly predicted to have type t:

- The precision corresponding to $F1^{\text{type}}_{\text{ma}}$ is defined as $\sum_t \frac{C(t)}{P(t)}\frac{G(t)}{\sum_{t'}G(t')}$; recall follows the same pattern.
- The precision corresponding to $F1^{\text{type}}_{\text{mi}}$ is defined as $\frac{\sum_t C(t)}{\sum_t P(t)}$; recall follows the same pattern.

Approach	$F1^{type}_{ma}$	$F1^{type}_{mi}$
ELMoNN	63.1	53.8
WikifierTyper	53.0	43.9
OTyper (Yuan and Downey, 2018)	50.6	23.4
Zoe (ours)	**71.7**	**71.1**

Table 4: Comparing systems where no labels (types) are seen a priori (§4.3).

System	Bacteria	not-Bacteria	Overall
WikifierTyper	54.8	**86.2**	70.5
ELMoNN	67.6	81.2	74.4
Zoe (ours)	**68.1**	84.2	**76.2**

Table 5: Results of the system classifying mentions to "*bacteria*" or something else (§4.4). Numbers are $F1$ in percentage.

Baselines. To add to the best published results on each dataset, we create two simple and effective baselines. The first baseline, ELMoNN, selects the nearest neighbor types to a given mention, where *mentions* and *types* are represented by ELMo vectors. To create a representation for each type t, we average the representation of the WikiLinks sentences that contain mentions of type t (as explained in §3.2). Our other baseline, WikifierTyper, uses Wikifier (Tsai et al., 2016b) to map the mention to a Wikipedia concept, followed by mapping to FreeBase types, and finally projecting them to the target types, via type definition function $T(.)$. Additionally, to compare with published zero-shot systems, we compare our system to OTyper, a recently published *open*-typing system. Unfortunately, to the best of our knowledge, the systems proposed by Ma et al.; Huang et al. (2016; 2016) are not available online for empirical comparison.

4.1 Fine-Grained Entity Typing

We evaluate our system for *fine-grained* named-entity typing. Table 2 shows the evaluation result for three datasets, FIGER, BBN, and $\text{OntoNotes}_{\text{fine}}$. We report our system's performance, our zero-shot baselines, and two supervised systems (AFET, plus the-state-of-the-art), for each dataset. There is no easy way to transfer the supervised systems across datasets, hence no out-of-domain numbers for such systems. For each dataset, we train OTyper and evaluate on the test sets of all the three datasets. In order to run OTyper on different datasets, we disabled original dataset-specific entity and type features. As a result, among the *open* typing systems, our system has significantly better results. In addition, our system has competitive scores compared to the supervised systems.

4.2 Coarse Entity Typing

In Table 3 we study entity typing for the coarse types on three datasets. We focus on three types that are shared among the datasets: *PER*, *LOC*, *ORG*. In coarse-entity typing, the best available systems are heavily supervised. In this evaluation, we use gold mention spans; i.e., we force the decoding algorithm of the supervised systems to select the best of the three classes for each gold mention. As expected, the supervised systems have strong in-domain performance. However, they suffer a significant drop when evaluated in a different domain. Our system, while not trained on any supervised data, achieves better or comparable performance compared to other supervised baselines in the out-of-domain evaluations.

4.3 Typing of Unseen Types within Domain

We compare the quality of *open* typing, in which the target type(s) have not been seen before. We compare our system to OTyper, which relies on supervised data to create representations for each type; however, it is not restricted to the *observed types*. We follow a similar setting to Yuan and Downey (2018) and split the FIGER test in folds (one fold per type) and do cross-validations. For each fold, mentions of only one type are used for evaluation, and the rest are used for training OTyper. To be able to evaluate on *unseen* types (only for this experiment), we use modified metrics $F1^{type}_{ma}$ and $F1^{type}_{mi}$ that measure *per type* quality of the system (§4). In this experiment, we focus on a within-domain setting, and show the results of transfer across genres in the next experiments. The results are summarized in Table 4. We observe a significant margin between Zoe and other systems, including OTyper.

4.4 Biology Entity Typing

We go beyond the scope of popular entity-typing tasks, and evaluate the quality of our system on a dataset that contains sentences from scientific papers (Delèger et al., 2016), which makes it different from other entity-typing datasets. The mentions refer to either "*bacteria*", or some miscellaneous class (two class typing). As indicated in Ta-

	FIGER			BBN			$\text{OntoNotes}_{\text{fine}}$		
Approach	Acc.	$F1_{\text{ma}}$	$F1_{\text{mi}}$	Acc.	$F1_{\text{ma}}$	$F1_{\text{mi}}$	Acc.	$F1_{\text{ma}}$	$F1_{\text{mi}}$
ZOE (ours)	58.8	74.8	71.3	61.8	74.6	74.9	50.7	66.9	60.8
no surface-based concepts	-8.8	-7.5	-9.2	-12.9	-7.0	-8.6	-1.8	-1.2	-0.1
no context-based concepts	-39.3	-42.1	-25.4	-36.4	-31.0	-13.9	-10.0	-12.3	-7.4

Table 6: Ablation study of different ways in which concepts are generated in our system (§4.5). The first row shows performance of our system on each dataset, followed by the *change* in the performance upon dropping a component. While both signals are crucial, contextual information is playing more important role than the mention-surface signal.

ble 5, our system's overall scores are higher than our baselines.

4.5 Ablation Study

We carry out ablation studies that quantify the contribution of surface information (§3.3) and context information (§3.2). As Table 6 shows, both factors are crucial and complementary for the system. However, the contextual information seems to have a bigger role overall.

We complement our qualitative analysis with the quantitative share of each component. In 69.3%, 54.6%, and 69.7% of mentions, our system uses the context information (and ignores the surface), in FIGER, BBN, and $\text{OntoNotes}_{\text{fine}}$ datasets, respectively, underscoring the importance of contextual information.

4.6 Error Analysis

We provide insights into specific reasons for the mistakes made by the system. For our analysis, we use the erroneous decisions in the FIGER dev set. Two independent annotators label the cause(s) of the mistakes, resulting in 83% agreement between the annotators. The disagreements are later reconciled by an adjudication step.

1. *Incompatible concept, due to context information:* Ambiguous contexts, or short ones, often contribute to the inaccurate mapping to concepts. In our manual annotations, 23.3% of errors are caused, at least partly, by this issue.
2. *Incompatible concept, due to surface information:* Although the prior probability is high, the surface-based concept could be wrong. About 26% of the errors are partly due to the surface signal errors.
3. *Incorrect type, due to type inference:* Even when the system is able to find several type-compatible concepts, it can fail due to inference errors. This could happen if the types attached to the type-compatible concepts are not the majority among other types attached to other concepts. This is the major reason behind 56.6% of errors.
4. *Incorrect type, due to type definition:* Some errors are caused by the inaccurate definition of the type mapping function T. About 23% of the mistakes are partly caused by this issue.

Note that each mistake could be caused by multiple factors; in other words, the above categories are not mutually disjoint events. A slightly more detailed analysis is included in Appendix C.

5 Conclusion

Moving beyond a fully supervised paradigm and scaling entity-typing systems to support bigger type sets is a crucial challenge for NLP. In this work, we have presented ZOE, a zero-shot open entity typing framework. The significance of this work is threefold. First, the proposed system does not require task-specific labeled data. Our system relies on type definitions, which are much cheaper to obtain than annotating thousands of examples. Second, our system outperforms existing state-of-the-art zero-shot systems by a significant margin. Third, we show that without reliance on task-specific supervision, one can achieve relatively robust transfer across datasets.

6 Acknowledgement

The authors would like to thank Jennifer Sheffield, Stephen Mayhew, and Qiang Ning for invaluable suggestions. This work was supported by Contract HR0011-15-2-0025 with the US Defense Advanced Research Projects Agency (DARPA). Approved for Public Release, Distribution Unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. Additionally, this research is supported by grants from Google and the Allen Institute for Artificial Intelligence (allenai.org).

References

Abhishek Abhishek, Ashish Anand, and Amit Awekar. 2017. Fine-grained entity type classification by jointly learning representations and label embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 797–807, Valencia, Spain. Association for Computational Linguistics.

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*.

Ming-Wei Chang, Lev-Arie Ratinov, Dan Roth, and Vivek Srikumar. 2008. Importance of semantic representation: Dataless classification. In *Proceedings of The Conference on Artificial Intelligence (AAAI)*.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke S. Zettlemoyer. 2018. Ultra-fine entity typing. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ido Dagan, Bill Dolan, Bernardo Magnini, and Dan Roth. 2010. Recognizing textual entailment: Rational, evaluation and approaches–erratum. *Natural Language Engineering*, 16(01):105–105.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzoto. 2013. Recognizing textual entailment: Models and applications.

Louise Delėger, Robert Bossy, Estelle Chaix, Mouhamadou Ba, Arnaud Ferrė, Philippe Bessieres, and Claire Nėdellec. 2016. Overview of the bacteria biotope task at bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*, pages 12–22.

Michael Fleischman and Eduard H. Hovy. 2002. Fine grained classification of named entities. In *COLING*.

Radu Florian, Abe Ittycheriah, Hongyan Jing, and Tong Zhang. 2003. Named entity recognition through classifier combination. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 168–171. Association for Computational Linguistics.

Evgeniy Gabrilovich and Shaul Markovitch. 2007. Computing semantic relatedness using wikipedia-based explicit semantic analysis. In *IJcAI*, volume 7, pages 1606–1611.

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

Ralph Grishman and Beth Sundheim. 1996. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, volume 1.

Eduard H. Hovy, Mitchell P. Marcus, Martha Palmer, Lance A. Ramshaw, and Ralph M. Weischedel. 2006. Ontonotes: The 90% solution. In *HLT-NAACL*.

Lifu Huang, Jonathan May, Xiaoman Pan, and Heng Ji. 2016. Building a fine-grained entity typing system overnight for a new x (x= language, domain, genre). *arXiv preprint arXiv:1603.03112*.

Daniel Khashabi, Mark Sammons, Ben Zhou, Tom Redman, Christos Christodoulopoulos, Vivek Srikumar, Nicholas Rizzolo, Lev Ratinov, Guanheng Luo, Quang Do, Chen-Tse Tsai, Subhro Roy, Stephen Mayhew, Zhilli Feng, John Wieting, Xiaodong Yu, Yangqiu Song, Shashank Gupta, Shyam Upadhyay, Naveen Arivazhagan, Qiang Ning, Shaoshi Ling, and Dan Roth. 2018. CogCompNLP: your swiss army knife for nlp. In *Proceedings of International Conference on Language Resources and Evaluation (LREC)*.

Xin Li and Dan Roth. 2005. Learning question classifiers: The role of semantic information. *Journal of Natural Language Engineering*, 11(4).

Thomas Lin, Oren Etzioni, et al. 2012. No noun phrase left behind: detecting and typing unlinkable entities. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 893–903. Association for Computational Linguistics.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Proceedings of The Conference on Artificial Intelligence (AAAI)*.

Edward Loper and Steven Bird. 2002. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Yukun Ma, Erik Cambria, and Sa Gao. 2016. Label embedding for zero-shot fine-grained named entity typing. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 171–180.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60.

Rada Mihalcea and Andras Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242. ACM.

Andrei Mikheev, Marc Moens, and Claire Grover. 1999. Named entity recognition without gazetteers. In *Proceedings of the ninth conference on European chapter of the Association for Computational Linguistics*, pages 1–8. Association for Computational Linguistics.

George Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

David Nadeau, Peter D Turney, and Stan Matwin. 2006. Unsupervised named-entity recognition: Generating gazetteers and resolving ambiguity. In *Conference of the Canadian Society for Computational Studies of Intelligence*, pages 266–277. Springer.

Mohammad Norouzi, Tomas Mikolov, Samy Bengio, Yoram Singer, Jonathon Shlens, Andrea Frome, Gregory S. Corrado, and Jeffrey Dean. 2013. Zero-shot learning by convex combination of semantic embeddings. In *International Conference on Learning Representations (ICLR)*.

Joel Nothman, James R Curran, and Tara Murphy. 2008. Transforming wikipedia into named entity training data. In *Proceedings of the Australasian Language Technology Association Workshop 2008*, pages 124–132.

Joel Nothman, Tara Murphy, and James R Curran. 2009. Analysing wikipedia and gold-standard corpora for ner training. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 612–620. Association for Computational Linguistics.

Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, and Tom M. Mitchell. 2009. Zero-shot learning with semantic output codes. In *Advances in Neural Information Processing Systems (NIPS)*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proc. of the Conference on Computational Natural Language Learning (CoNLL)*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to wikipedia. In *Proc. of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

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 *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*.

Bernardino Romera-Paredes and Philip H. S. Torr. 2015. An embarrassingly simple approach to zero-shot learning. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Dan Roth. 2017. Incidental supervision: Moving beyond supervised learning. In *Proc. of the Conference on Artificial Intelligence (AAAI)*.

Satoshi Sekine, Kiyoshi Sudo, and Chikashi Nobata. 2002. Extended named entity hierarchy. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC)*.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural architectures for fine-grained entity type classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1271–1280.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2012. Wikilinks: A large-scale cross-document coreference corpus labeled via links to wikipedia. *University of Massachusetts, Amherst, Tech. Rep. UM-CS-2012*, 15.

Yangqiu Song and Dan Roth. 2014. On dataless hierarchical text classification. In *Proceedings of The Conference on Artificial Intelligence (AAAI)*.

Erik F Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.

Antonio Toral, Elisa Noguera, Fernando Llopis, and Rafael Munoz. 2005. Improving question answering using named entity recognition. In *International Conference on Application of Natural Language to Information Systems*, pages 181–191. Springer.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016a. Cross-lingual named entity recognition via wikification. In *Proc. of the Conference on Computational Natural Language Learning (CoNLL)*.

Chen-Tse Tsai, Stephen Mayhew, and Dan Roth. 2016b. Cross-lingual named entity recognition via wikification. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 219–228.

Ralph Weischedel and Ada Brunstein. 2005. Bbn pronoun coreference and entity type corpus. *Linguistic Data Consortium, Philadelphia*, 112.

Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, page 9pp.

Dani Yogatama, Daniel Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, volume 2, pages 291–296.

Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, and Gerhard Weikum. 2012. Hyena: Hierarchical type classification for entity names. *Proceedings of COLING 2012: Posters*, pages 1361–1370.

Zheng Yuan and Doug Downey. 2018. Otyper: A neural architecture for open named entity typing. In *Proceedings of The Conference on Artificial Intelligence (AAAI)*.