Fine-Grained Entity Typing with High-Multiplicity Assignments

Maxim Rabinovich and **Dan Klein**
Computer Science Division
University of California, Berkeley
{rabinovich,klein}@cs.berkeley.edu

Abstract

As entity type systems become richer and more fine-grained, we expect the number of types assigned to a given entity to increase. However, most fine-grained typing work has focused on datasets that exhibit a low degree of type multiplicity. In this paper, we consider the high-multiplicity regime inherent in data sources such as Wikipedia that have semi-open type systems. We introduce a set-prediction approach to this problem and show that our model outperforms unstructured baselines on a new Wikipedia-based fine-grained typing corpus.

1 Introduction

Motivated by potential applications to information retrieval, coreference resolution, question answering, and other downstream tasks, recent work on entity typing has moved beyond coarse-grained systems towards richer ontologies with much more detailed information, and therefore correspondingly more specific types (Ling and Weld, 2012; Gillick et al., 2014; Yogatama et al., 2015).

As types become more specific, entities will tend to belong to more types (i.e. there will tend to be higher type multiplicity). However, most data used in previous work exhibits an extremely *low* degree of multiplicity.

In this paper, we focus on the high multiplicity case, which we argue naturally arises in large-scale knowledge resources. To illustrate this point, we construct a corpus of entity mentions paired with higher-multiplicity type assignments. Our corpus is based on mentions and categories drawn from Wikipedia, but we generalize and denoise the raw Wikipedia categories to provide more coherent supervision. Table 1 gives examples of type assignments from our dataset.

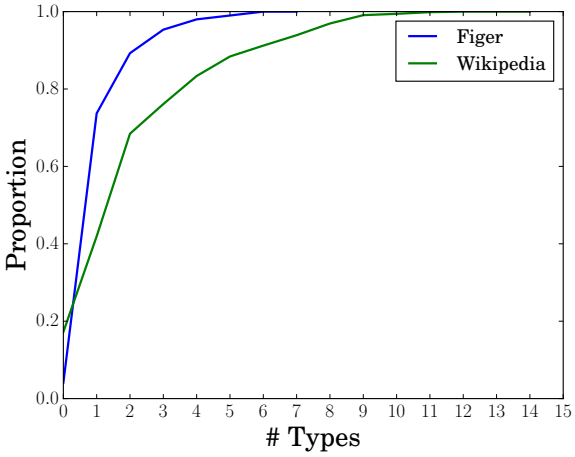

Figure 1: Comparison of type set size CDFs for the our Wikipedia corpus and the prior FIGER corpus (Ling and Weld, 2012). The figure illustrates that our corpus exhibits much greater type assignment multiplicity.

As type multiplicity grows, it is natural to consider type prediction as an inherently set-valued problem and ask questions about how such sets might be modeled. To this end, we develop a structured prediction approach in which the sets of assigned types are predicted as first-class objects, including a preliminary consideration of how to efficiently search over them. The resulting model captures type correlations and ultimately outperforms a strong unstructured baseline.

Related work The fine-grained entity typing problem was first investigated in detail by Ling and Weld (2012). Subsequently, Gillick et al. (2014) introduced a larger evaluation corpus for this task and introduced methods for training predictors based on multiclass classification. Both used the Freebase typing system, coarsened to approximately 100 types, and subsequent work

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 330–334
Vancouver, Canada, July 30 - August 4, 2017.
https://doi.org/10.18653/v1/P17-2052

David Foster Wallace	novelist suicide sportswriter writer alumnus ...
Albert Einstein	physicist agnostic emigrant people pacifist ...
NATO	organization treaty document organisation alliance ...
Federal Reserve	agency authorities banks institution organization ...
Industrial Revolution	concept history evolution revolution past ...
Black Death	concept epidemic pandemic disaster ...

Table 1: With types from a large corpus like Wikipedia, large type set assignments become common.

Entity	Raw Type	Projected Type
David Foster Wallace	Short_story_writers	writer
	Amherst_alumni	alumnus
	Illinois_State_faculty	faculty
	People_from_New_York	people
	Essayists	essayist

Table 2: Example of an entity and its types, before and after projection. The projection operation collapses related types that would be very difficult to learn in their original, highly specific forms.

has mostly followed this lead (Yaghoobzadeh and Schütze, 2016; Yogatama et al., 2015), although types based on WordNet have recently also been investigated (Corro et al., 2015).

Most prior work has focused on unstructured predictors using some form of multiclass logistic regression (Ling and Weld, 2012; Gillick et al., 2014; Shimaoka et al., 2016; Yaghoobzadeh and Schütze, 2016; Yogatama et al., 2015). Some of these approaches implicitly incorporate structure during decoding by enforcing hierarchy constraints (Gillick et al., 2014), while neural approaches can encode correlations in a soft manner via shared hidden layers (Shimaoka et al., 2016; Yaghoobzadeh and Schütze, 2016).

Our work differs from these lines of work in two respects: its use of a corpus exhibiting high type multiplicity with types derived from a semi-open inventory and its use of a fully structured model and decoding procedure, one that can in principle be integrated with neural models if desired. Previously, most results focused on the low-multiplicity Freebase-based FIGER corpus. The only work we are aware of that uses a type system similar to ours used a rule-based system and evaluated on their own newswire- and Twitter-based evaluation corpora (Corro et al., 2015).

2 Model

Our structured prediction framework is based on modeling type assignments as sets. Each entity e is assigned a set of types T^* drawn from the overall set of types $\mathcal{T}$. Our goal is thus to predict, given an input sentence-entity pair, the set of types associated with that entity.

We take the commonly-used linear model approach to this structured prediction problem. Given a featurizer φ that takes an input sentence x and entity e, we seek to learn a weight vector w such that

$$f(x,\ e) = \text{argmax}_T\ w^\top \varphi(x,\ e,\ T) \quad (1)$$

predicts T correctly with high accuracy.

Our approach stands in contrast to prior work, which deployed several techniques, of similar efficacy, to port single-type learning and inference strategies to the multi-type setting (Gillick et al., 2014). Provided type interactions can be neglected, equation (1) can be simplified to

$$f_{\text{single}}(x,\ e) = \left\{t \in \mathcal{T}:\ w^\top \varphi(x,\ e,\ t) \geq r\right\}.$$

This simplification corresponds to expanding each multi-type example triple $(x,\ e,\ T^*)$ into a set of single-type example triples $\left\{(x,\ e,\ t^*)_{t^* \in T^*}\right\}$. Learning can then be done using any technique for multiclass logistic regression, and inference can be carried out by specifying a threshold r and predicting all types that score above that threshold: In prior work, a simple $r = 0$ threshold was used (Ling and Weld, 2012).

In this paper, we focus on the more general specification (1), though in Section 2.2, we explain a simplification that can be used to speed up inference if desired.

2.1 Features

Modeling type assignments as sets in principle opens the door to non-decomposable set features (a simple instance of which would be set size). For reasons of tractability, we assume our features factor along type pairs:

$$\varphi(x,\ e,\ T) = \sum_{t \in T} \varphi(x,\ e,\ t) + \sum_{t,\ t' \in T} \varphi\left(t,\ t'\right) \tag{2}$$

Note that in addition to enforcing factorization over type pairs, the specification (2) requires that any features linking the type assignment to the observed entity mention depend only on a single type at a time. We investigated non-decomposable features, but found they did not lead to improved performance.

We use entity mention features very similar to those in previous work:

1. **Context unigrams and bigrams.** Indicators on all uni- and bigrams within a certain window of the entity mention.

2. **Dependency parse features.** Indicators on the lexical parent of the entity mention head, as well as the corresponding dependency type. Separately, indicators on the lexical children of the entity mention head and their dependency types.

3. **Entity head and non-head tokens.** Indicators on the syntactic head of the entity mention and on its non-head tokens.

4. **Word shape features.** Indicators on the shape of each token in the entity mention.

We combine these features with type-based features to obtain the features our model actually uses:

1. **Conjunction features.** These are simple conjunctions of mention features with indicators on type membership in the predicted set. Using only these features results in an unstructured model.

2. **Type pair features.** These are indicators on pairs of types appearing in the predicted set.

3. **Graph-based features.** As we discuss in Section 3, the type system in our corpus comes with a graph structure. We add indicators on certain patterns occurring within the set–e.g. a parent-child type pair, sibling type pairs, and so on, abstracting away the specific types.

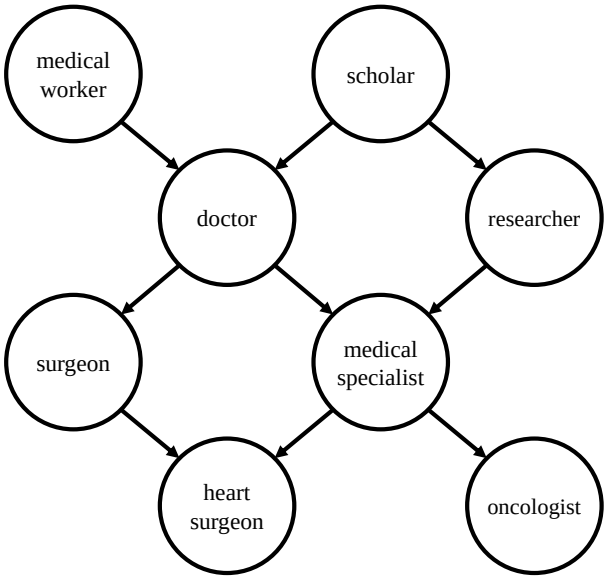

Figure 2: Fragment of the graph underlying our type system.

2.2 Learning and Inference

We train our system using structured max-margin (Tsochantaridis et al., 2005). Optimization is performed via AdaGrad on the primal (Kummerfeld et al., 2015). We use set-F1 as our loss function.

Inference, for both prediction and loss-augmented decoding, poses a greater challenge, as solving the maximization problem (1) exactly requires iterating over all subsets of the type system.

Fortunately, we find a simple greedy algorithm is effective. Our decoder begins by choosing the type that scores highest individually, taking only single-type features into account. It then proceeds by iteratively adding new types into the set until doing so would decrease the score.

At the cost of restricting the permissible type sets slightly, we can speed up the greedy procedure further. Specifically, we can require that the predicted type set T be *connected* in some constraint graph over the types—either the co-occurrence graph, the complete graph, or the graph underlying the type system. If we denote by $\mathcal{C}$ the set of all such connected sets, the corresponding predictor would be

$$f_{\text{conn}}(x,\ e) = \text{argmax}_{T \in \mathcal{C}}\ w^{\top} \varphi(x,\ e,\ T)$$

Level	Features	P	R	F1
Entity	Unstructured	50.0	**67.2**	52.9
	+ Pairs	53.3	64.1	54.3
	+ Graph	**53.9**	63.9	**54.5**
Sentence	Unstructured	42.6	**58.9**	44.4
	+ Pairs	46.5	54.1	**45.6**
	+ Graph	**47.0**	53.6	**45.6**

Table 3: Results on our corpus. All quantities are macro-averaged.

The greedy decoding procedure for this predictor is faster because at each step, it need only consider adding types that are adjacent to some type that has already been included.

3 Corpus

Our corpus construction methodology involves three key stages: mention identification, type system construction, and type assignment.[1] We explain each of these in turn.

Mention identification. We follow prior work on entity linking (Durrett and Klein, 2014) and take all mentions that occur as anchor text. We filter the resulting collection of mentions down to those that pass a heuristic filter that removes mentions of common nouns, as well as spurious sentences representing Wikipedia formatting.

Type system construction. Prior work on fine-grained entity typing has derived its type system from Freebase (Ling and Weld, 2012; Gillick et al., 2014). The resulting ontologies thus inherit the coverage and specificity limitations of Freebase, somewhat exacerbated by manual coarsening.

Motivated by efforts to inject broader coverage, more complex knowledge resources into NLP systems, we instead derive our types from the Wikipedia category and WordNet graphs, in a manner similar to that of Ponzetto and Strube (2007).

Our base type set consists of all Wikipedia categories. By following back-pointers in articles for categories, we derive a base underlying directed graph. To eliminate noise, we filter down to all categories whose syntactic heads can be found in WordNet and keep directed edges only when the head of the parent is a WordNet ancestor of the head of the child. We conclude by projecting each type down to its syntactic head.

[1]Our corpus will be released at http://people.eecs.berkeley.edu/~rabinovich/.

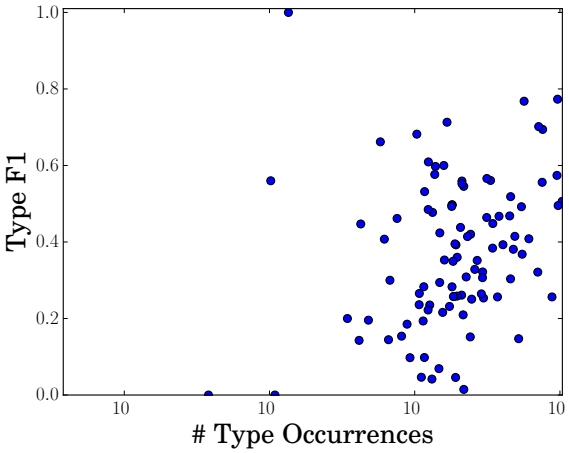

Figure 3: Per-type F1 scores plotted by type frequency in the training corpus.

Type assignment. The type set for an entity is obtained by taking its Wikipedia category assignments, augmenting these with their ancestors in the category graph above, and then projecting these down to their syntactic heads.

4 Experiments

We evaluate our method on the dataset described in Section 3. For these experiments, we restrict to the 100 most frequent types and downsample to 750K mentions. We use a baseline that closely replicates the FIGER system (Ling and Weld, 2012). Within our framework, this can be thought of as a model that sets all type pair features in (2) to zero.

Table 3 summarizes our results. Starting with the baseline, we incrementally add the type pair, graph-based, and set size features discussed in 2.1. Adding type pair features results in an appreciable performance gain, while the graph features bring little benefit—potentially because pairwise correlations suffice to summarize the set structure when the number of types is moderately low.

A concern when studying multiclass problems with large numbers of classes, whether predicting sets or individual labels, is that performance on instances associated with common classes will dominate the performance metric. Figure 3 shows micro-averaged F1 for the binary prediction task associated with predicting the presence or absence of each type, demonstrating that our performance is strong even for many rare types.

5 Conclusion

We have highlighted the issue of multiplicity in fine-grained entity typing. Whereas most prior work has focused on corpora with low multiplicity assignments, we denoised the Wikipedia type system to construct a realistic corpus with high multiplicity type assignments. Using this corpus as a testbed, we showed that an approach based on structured prediction of sets can outperform unstructured baselines when type assignments have high multiplicity. Our approach may therefore be preferable in such contexts.

References

Luciano del Corro, Abdalghani Abujabal, Rainer Gemulla, and Gerhard Weikum. 2015. Finet: Context-aware fine-grained named entity typing. Assoc. for Computational Linguistics.

Greg Durrett and Dan Klein. 2014. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics* 2:477–490.

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

Jonathan K Kummerfeld, Taylor Berg-Kirkpatrick, and Dan Klein. 2015. An empirical analysis of optimization for max-margin NLP. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. volume 273, page 279.

Xiao Ling and Daniel S Weld. 2012. Fine-grained entity recognition. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Simone Paolo Ponzetto and Michael Strube. 2007. Deriving a large scale taxonomy from wikipedia.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2016. An attentive neural architecture for fine-grained entity type classification. *arXiv preprint arXiv:1604.05525* .

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research* 6(Sep):1453–1484.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2016. Corpus-level fine-grained entity typing using contextual information. *arXiv preprint arXiv:1606.07901* .

Dani Yogatama, Dan 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 of the Asian Federation of Natural Language Processing, ACL*.