Instilling Type Knowledge in Language Models via Multi-Task QA

Shuyang Li
UC San Diego
shl008@ucsd.edu

Mukund Sridhar
Amazon Alexa AI
harakere@amazon.com

Chandana Satya Prakash
Amazon Alexa AI
chanprak@amazon.com

Jin Cao
Amazon Alexa AI
jincao@amazon.com

Wael Hamza
Amazon Alexa AI
waelhamz@amazon.com

Julian McAuley
UC San Diego
jmcauley@ucsd.edu

Abstract

Understanding human language often necessitates understanding entities and their place in a taxonomy of knowledge—their types. Previous methods to learn entity types rely on training classifiers on datasets with coarse, noisy, and incomplete labels. We introduce a method to instill fine-grained type knowledge in language models with text-to-text pre-training on type-centric questions leveraging knowledge base documents and knowledge graphs. We create the WikiWiki dataset: entities and passages from 10M Wikipedia articles linked to the Wikidata knowledge graph with 41K types. Models trained on WikiWiki achieve state-of-the-art performance in zero-shot dialog state tracking benchmarks, accurately infer entity types in Wikipedia articles, and can discover new types deemed useful by human judges.

1 Introduction

Entities can be categorized by their types, which indicate where they belong in a taxonomy of knowledge. For example, Venus is a planet and thus also an astronomical body. Much like how knowledge acquisition in cognitive development progresses from recognizing concrete objects to gradually understanding their relations to one another (Lucariello et al., 1992), we aim to extend language models’ existing rough understanding of entities (Heinzerling and Inui, 2021) to the types that govern how entities are related. Instilling type knowledge in multi-purpose models can improve performance in tasks like entity linking (Onoe and Durrett, 2020), question-answering (Févry et al., 2020a), and semantic parsing (Thirukovalluru et al., 2021).

While language models can memorize some facts (Petroni et al., 2019), they frequently hallucinate false information (Logan IV et al., 2019; Shuster et al., 2021). Current attempts to learn to infer types for entities are hampered by 1) the difficulty of collecting diverse, large-scale typing datasets; and 2) how existing corpora assume independence between types (Choi et al., 2018), while in reality types sit at levels of granularity that are useful in different settings: a pizza store may care whether a user likes Cheese Pizza; a restaurant recommender might care whether the user wants Pizza; finally, a general dialog agent might only care whether the user wants Food.

We address both issues by proposing a simple and effective approach for pre-training generative language models to answer questions about entities, types, and surface forms (mentions) in a large public knowledge graph (KG) consisting of Wikipedia articles and Wikidata nodes. We leverage high quality type labels in a large corpus of knowledge-rich text and an ordered, hierarchical type ontology.

To summarize our main contributions: 1) We create the new WikiWiki dataset comprising 10M Wikipedia articles linked to nodes from Wikidata; 2) We propose a pre-training scheme for generative language models using type-centric question-answering based on WikiWiki; 3) We achieve state-of-the-art (SOTA) performance in zero-shot domain adaptation for dialog state tracking using our type-instilled models, with average per-domain gains of 14.9% (49.4% relative) joint accuracy; and 4) We show that our models can precisely infer types for seen and unseen entities in new articles from WikiWiki, and propose novel types that hu-
Yamada et al. (2020) explicitly denote entity tokens with a learned input embedding. Specific entity embeddings have also been learned jointly by using knowledge graphs as auxiliary inputs during language model pre-training (Sun et al., 2020a; Févry et al., 2020b; Zhang et al., 2021). Another line of work aims to model specific factual statements from knowledge bases (Wang et al., 2021) for reading comprehension (Lu et al., 2021) and trivia QA (Agarwal et al., 2021). We propose text-to-text pre-training on knowledge recovery tasks to instill type-awareness. Our models learn type knowledge that transfers to the type-adjacent downstream task of dialog state tracking and can infer unseen types.

3 Type-Centric Multitask Modeling

WikiWiki Corpus To train an entity- and type-aware language model, we build the WikiWiki dataset by combining Wikipedia articles with the Wikidata KG (Vrandecic, 2012). Wikipedia articles have been used to enrich corpora for dialog (Dinan et al., 2019), coreference resolution (Singh et al., 2012), and QA (Liu et al., 2020). KGs have been used for entity typing and relation extraction (Sakor et al., 2020). Yao et al. (2019) use Wikipedia pages as context for relation triples mined from Wikidata.

We link articles, entities, and types as in Figure 1: like Wu et al. (2020b), we take Wikipedia hyperlinks as links between entities (target page) and their mentions (link text); we link pages to Wikidata nodes via ID; and for each node we extract types $T$ from Wikidata where $t \in T$ is an instance/subclass of the node (discarding entities with no types). To address sparsity of hyperlinks,
we follow Yao et al. (2019) and use spaCy to identify additional entities. We sample 10M articles for training, with two disjoint 5K-article splits for evaluation, containing seen and unseen (New Ent) entities respectively (Table 1). The ontology of Wikidata types forms a directed acyclic graph with 41K type nodes applying to 2.2M entities. Previous entity typing datasets rely on annotations from small groups of crowd-workers and include a small type ontology in the hundreds (Ling and Weld, 2012) and/or sacrifice label accuracy (Choi et al., 2018). We instead rely on the cumulative, cross-checked annotations from tens of thousands of active Wikidata users.

Entities in Wikidata on average are assigned 1.28 types; for entities with multiple types, not all types are necessarily relevant to a context. For example, take the following passage: “Obama was elected to the Illinois Senate in 1996, succeeding Democratic State Senator Alice Palmer from Illinois’s 13th District, which, at that time, spanned Chicago South Side neighborhoods from Hyde Park–Kenwood south to South Shore and west to Chicago Lawn.”

While Wikidata entities may have 5+ types, many are not directly relevant to a context. For example, while Barack Obama has types including Politician, Jurist, Political Writer, Community Organizer, and Podcaster, the latter is not relevant to the context. To teach our models to infer types relevant to the context at hand, in pre-training data we take only types that are shared between Barack Obama and other entities in the document (e.g. Alice Palmer—Politician). We have made the WikiWiki dataset publicly available on Github.²

### Pre-training Tasks
To instill type-centric knowledge from WikiWiki, we train our models to answer four types of knowledge-based questions conditioned on a passage from Wikipedia (examples in Table 2). In entity/type discovery, the model is tasked to recover all surface forms (mentions) that reference an entity, along with their types—this is analogous to simultaneous entity recognition and typing. Entity typing consists of assigning types to an entity of interest. For entity recognition, we follow Cao et al. (2021) by training our model to respond with an entity’s full name and type when queried with a surface form. In slot filling we ask our model to return all entities mentioned in the passage belonging to a certain type. For multi-type entities, we use a subset of relevant types given other entities in the context (Appendix A). We treat QA as a universal format for diverse NLU tasks (McCann et al., 2018), and adopt the framework of Raffel et al. (2020) to treat each of our tasks as text-to-text generative modeling. We create 50M questions for pre-training.

### Model Architecture
We use an encoder-decoder (Sutskever et al., 2014) model initialized from BART—a Transformer (Vaswani et al., 2017) language model pre-trained via de-noising auto-encoding (Lewis et al., 2020a). Our model generates an answer a as a text sequence given a document D of length $t_D$ and question q. The document is encoded via the encoder—consisting of $l$ Transformer layers of hidden dimensionality $h$, each applying 16-headed self-attention—to produce $z := \text{Enc}(D) \in \mathbb{R}^{t_D \times h}$.

We train the model to perform QA via conditional language modeling. Instead of concatenating the question with the context in encoder input (Lin et al., 2021), the decoder generates a sequence consisting of the question and answer: $x = [q; a]$. We can thus cache the document encoding at inference to answer multiple questions. At training time we perform next-token prediction, calculating cross-entropy loss by maximizing the log likelihood of the question and answer conditioned on the document: $P(q, a|D) = \prod_{t} P(x_t|x_{<t}, D)$. We assess the impact of our pre-training on Base ($l=12$, $h=768$) and Large ($l=24$, $h=1024$) models.

### 4 Experiments
We demonstrate the effectiveness of our pre-training on two tasks that require type understand-
Table 4: Zero-shot domain adaptation JGA (%) on MultiWOZ 2.1 test set on the (R)estaurant, (H)otel, (A)traction, (T)rain, and Ta(X)i domains. We achieve SOTA results on all domains by significant margins.

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th>R</th>
<th>H</th>
<th>A</th>
<th>T</th>
<th>X</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRADE</td>
<td>90M</td>
<td>12.6</td>
<td>14.2</td>
<td>20.1</td>
<td>22.4</td>
<td>59.2</td>
</tr>
<tr>
<td>MA-DST</td>
<td>90M</td>
<td>13.6</td>
<td>16.3</td>
<td>22.5</td>
<td>22.8</td>
<td>59.3</td>
</tr>
<tr>
<td>SUMBT</td>
<td>355M</td>
<td>16.5</td>
<td>19.8</td>
<td>22.6</td>
<td>22.5</td>
<td>59.5</td>
</tr>
<tr>
<td>GPT2-DST</td>
<td>355M</td>
<td>26.2</td>
<td>24.4</td>
<td>31.3</td>
<td>29.1</td>
<td>59.6</td>
</tr>
<tr>
<td>BART</td>
<td>139M</td>
<td>27.9</td>
<td>31.9</td>
<td>38.4</td>
<td>34.3</td>
<td>70.5</td>
</tr>
<tr>
<td>Ours (Base)</td>
<td>139M</td>
<td>40.4</td>
<td>36.5</td>
<td>39.8</td>
<td>36.1</td>
<td>70.9</td>
</tr>
<tr>
<td>Ours (Large)</td>
<td>406M</td>
<td>46.7</td>
<td>38.8</td>
<td>49.8</td>
<td>37.7</td>
<td>72.1</td>
</tr>
</tbody>
</table>

Zero-Shot DST

The goal of Dialog State Tracking (DST) is to infer user intent and goals from conversations by filling in belief slots (Lemon et al., 2006; Wang and Lemon, 2013). In many real-world settings, DST models must be able to predict new slot values (i.e. new entities that are not present in the training corpus) and new slot types (e.g. requirements for applications in new domains). This problem setting is known as zero-shot DST (Table 3). We follow the zero-shot setting in Campana et al. (2020): train a model on multi-domain DST data and evaluate on a held-out domain. We measure domain generalization performance via joint goal accuracy (JGA): the percent of turns in which a model successfully predicts values for all slots in the target domain. We use the MultiWOZ 2.1 benchmark (Eric et al., 2019), evaluating zero-shot JGA for the Restaurant, Hotel, Attraction, Train, and Taxi domains. At each turn, we ask the model a question about the preference for each slot. We compare against recent systems that can perform zero-shot DST: TRADE (Wu et al., 2019), MA-DST (Kumar et al., 2020), SUMBT (Lee et al., 2019), and GPT2-DST (Li et al., 2021). Our method is complementary to systems for creating synthetic in-domain dialogs (Kim et al., 2021).

As seen in Table 4, our type-centric pre-training allows a model to answer questions about unseen slots. BART-base itself achieves SOTA JGA across all domains, and our pre-training significantly increases the gain to 10.6% absolute / 34.8% relative JGA—despite only using one-third of the parameters. Our Large model achieves 14.9% absolute and 49.4% relative gain in JGA compared to previous SOTA. The most significant gains come in the Hotel and Restaurant domains, which contain the most categorical slots that resemble types (e.g. cuisine, hotel type). In Table 5 we compare our models against same-size BART models at different levels of training data availability to demonstrate the additive utility of our method. Our method is particularly helpful with less fine-tuning data (low-data regimes), with average gains of 39% for small models and 4.8% for large models at 20% data availability. Gains are magnified for smaller models, affirming that our method can effectively instill type knowledge in lightweight language models.

Table 5: Relative gain (%) in JGA for models trained on WikiWiki vs standard BART pre-training. Our method helps more in low-data regimes and for smaller models.

<table>
<thead>
<tr>
<th>Model</th>
<th># Params</th>
<th>100%</th>
<th>50%</th>
<th>20%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base (139M)</td>
<td>13.7</td>
<td>14.7</td>
<td>39.0</td>
<td></td>
</tr>
<tr>
<td>Large (406M)</td>
<td>0.9</td>
<td>1.6</td>
<td>4.8</td>
<td></td>
</tr>
</tbody>
</table>
tors to judge the accuracy of 443 type labels from 200 randomly sampled contexts. We confirm that WikiWiki is a high-quality benchmark for entity typing, with 85% type precision assessed by human judges (compared to 68% for UltraFine).

We found that multi-label classifiers built on RoBERTa (Liu et al., 2019) that perform well on UltraFine require significant hyper-parameter tuning to output non-trivial predictions to classify our large and sparse (41K) type ontology. To perform entity typing with our model, we generate comma-delimited text sequences of types (Yang et al., 2018). This allows our models to infer and generate novel types while classifiers remain restricted to the training ontology. We confirm that our pre-training helps models better infer types for both seen (+14.3 F1) and unseen entities (+16.7 F1) in new contexts compared to classifiers (Table 6).

To investigate if our model can discover novel types, we perform another human evaluation over 557 such predictions from 300 contexts, with inter-annotator agreement of $\kappa = 0.4086$. Our model accurately extrapolates its type knowledge beyond the training ontology—we observe 73.3% precision when inferring new types (compared to 74.5% precision for seen types), demonstrating that our pre-training enables models to reason about types beyond simple memorization. Our model discovers complex and specific scientific types, correctly proposing that anorthosite (an aluminum silicate rock) is a metallurgical rock$^3$ and that speckled tortoises are monotrophs.$^4$ This reflects the robust taxonomy of types in scientific disciplines. Our model also proposes granular categories of events, and is judged to correctly type the 2015 Tour of Taiwan as an instance of the Tour de Taiwan cycling race. In the future, we seek methods to automatically assess the factual accuracy of new types.

Table 6: P/R/F1 of pred. vs. gold types on WikiWiki Test (seen) and Test New Ent (unseen entities) splits.

<table>
<thead>
<tr>
<th>Entities</th>
<th>Model</th>
<th>Precision</th>
<th>Recall</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seen</td>
<td>RoBERTa</td>
<td>62.35</td>
<td>59.38</td>
<td>60.82</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>78.13</td>
<td>72.39</td>
<td>75.15</td>
</tr>
<tr>
<td>Unseen</td>
<td>RoBERTa</td>
<td>48.88</td>
<td>47.96</td>
<td>48.41</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>66.65</td>
<td>63.71</td>
<td>65.14</td>
</tr>
</tbody>
</table>

5 Conclusion

In this paper, we 1) propose a text-to-text pre-training scheme to instill type knowledge in language models via QA and 2) release the WikiWiki dataset built from Wikipedia articles and the Wikidata KG. We show that WikiWiki is larger-scale and more accurate than existing fine-grained type recognition datasets. We demonstrate that our type-centric pre-training framework allows us to train language models that can better generalize to unseen domains, entities, and types—which in turn lead to improved model performance on downstream tasks like dialog state tracking (achieving SOTA results on zero-shot DST with average gains of 14.9% joint accuracy). Our models can extrapolate type knowledge and infer novel types that humans judge to be useful and precise. As the body of human knowledge grows, we see an opportunity to use life-long learning (Parisi et al., 2019) on news and publications to expand and model the taxonomy of knowledge.

Acknowledgements

We would like to thank Stephen Rawls, Ryan Gabbard, and anonymous reviewers for providing valuable feedback on this work. We also thank Nicolas Guénon des Mesnards and Victor Soto for their help setting up MTurk for human evaluations. Work was performed during first author’s internship at Amazon Alexa AI. Findings and observations are of the authors only, and do not necessarily reflect the views of Amazon or UCSD.

References


Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In ICLR.


Hongliang Dai, Yangqiu Song, and Haixun Wang. 2021. Ultra-fine entity typing with weak supervision

---

$^3$rocks containing metallic compounds and properties

$^4$has diet comprising one type of food (Herrera, 1976)


Oliver Lemon, Kallirroi Georgila, James Henderson, and Matthew N. Stuttle. 2006. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: Generic slot-filling in the TALK in-car system. In EACL.


A Data

We use the June 2021 Wikidata database file from https://www.wikidata.org/wiki/Wikidata:Database_download for raw KG data. We use English Wikipedia article HTML crawled from the same time period. While Wikidata contains multilingual definitions and labels for each node, in this paper we use only English entity and type names.

Wikipedia data was collected under the original terms of release which allow free usage of such materials for non-commercial purposes. We will release WikiWiki under the same license.

When creating questions for pre-training tasks, if a question has multiple answers (e.g. multiple chemists in Table 2), the answers are a comma- and and-delimited sequence, in order of appearance in the context. For the entity typing question, we use the order that types appear in the Wikidata page.

B Experimental Settings

We train all of our models on a node with eight Nvidia V100 GPUs (comprising 256 GB total VRAM) and 768 GB of RAM. We optimize using Deepspeed Stage 1 (Pudipeddi et al., 2020) using FP16 and the Lamb optimizer (You et al., 2020). Experimental results, where applicable, are reported as median of 3 experiments.

Hyperparameters For pre-training, we use a learning rate of 1e-4 with a linear warm-up for the first 10% of training iterations, using an effective batch size of 960. Our models were trained on a single pass of our pre-training dataset of 50M questions, totaling 52K steps. We fine-tune models using the same learning rate schedule, using an effective batch size of 2560 and early stopping for a maximum of 10 epochs based on validation loss. We aim to establish the general ability of our pre-training scheme to instill type awareness, and thus fix hyperparameters for generative language models trained with our method without hyperparameter tuning.

As mentioned in Section 4, the RoBERTa-based classifier for entity typing on WikiWiki required significantly more hyperparameter tuning; we performed a hyperparameter sweep on batch size (512 to 2048), learning rate (1e-3 to 1e-5), optimizer

Table 7: Zero-shot domain adaptation JGA (%) on MultiWOZ 2.1 test set on the (R)estaurant, (H)otel, (A)ttraction, (T)rain, and Ta(X)i domains. Compared to GPT2-DST (Li et al., 2021) augmented with out-of-domain DST data (+SGD), our Base model outperforms the augmented model in 3/5 domains and our Large model outperforms it in 4/5 domains.

(Adam vs. Lamb), and whether to freeze the encoder. We achieved best performance (as in Table 6) with a learning rate of 1e-4, the Adam optimizer, an effective batch size of 960, and with gradual unfreezing (Howard and Ruder, 2018) over 5K steps. We found gradual unfreezing to be critical for model performance, with fully frozen and fully unfrozen RoBERTa models achieving entity typing F1 scores of ≤10.0.

C Dialog State Tracking Notes

As discussed in Section 4, our method is orthogonal to and can thus be used simultaneously with techniques for creating synthetic in-domain training data for DST (Campagna et al., 2020; Kim et al., 2021). For slot queries, we use templated questions of the form: What [domain] [slot] is the user interested in?

We compare our models against SOTA models for zero-shot DST on MultiWOZ 2.1. We affirm the observations of Lin et al. (2021) that while T5-DST achieves strong DST performance on the 2.0 version of the dataset, performance degrades on the 2.1 benchmark.

Li et al. (2021) also present results for GPT2-DST when training is augmented with additional DST data from a wider pool of domains—the Schema-Guided Dialog dataset (Rastogi et al., 2020). In the interest of fairness, we do not compare this setting in Table 4 as our models do not have access to any conversational data in pre-training and—like the other baseline models—cannot access additional DST data in fine-tuning. Despite the lack of exposure to conversational data, in Table 7 we show that our Small and Large models out-perform GPT2-DST + SGD in 3/5 domains (with absolute per-domain gain of 5.5% and relative gain of 18.3%) and 4/5 domains (with absolute gains of 9.7% and relative gains of 30.6%), respectively. We additionally present zero-shot DST performance (JGA) on the MultiWOZ 2.1 validation set in Table 8.

D Human Evaluation Details

We perform our evaluation using the Amazon Mechanical Turk platform. To ensure high quality annotations, we recruit only crowd workers with Master qualification—indicating a history of high quality accepted work—and who are native English speakers. Crowd-workers remained anonymous outside of their qualifications and we did not collect any additional demographic information. Workers were informed that their type accuracy judgements were to be used in an academic research setting, with an option to opt-out and reject the task.

As both gold types and predicted types could be complex and require domain knowledge, evaluators were instructed to search any relevant additional material (textbooks, sites, papers) to ensure they made a high confidence judgment of type accuracy. Based on the average time spent evaluating each article, our pay rate worked out to above Federal minimum wage in the United States.

In Figure 2 we display the example instructions given to a human evaluator for assessing the accuracy of a type for an entity referenced in a context. In Figure 3 we show sample instructions given to a human evaluator to choose which of two types (predicted or gold label in random order) is more suitable / applies more accurately to the referenced entity.

E Ethics

As with all models capable of generating arbitrary text sequences, models trained with our framework and tasks run the risk of outputting toxic or offensive text (Gehman et al., 2020). However, our training aims to instill type knowledge for type-
and concept-reliant downstream tasks. As such, we expect that our pre-training does not heighten the risk of offensive outputs compared to other general-purpose pre-training schemes on wide internet corpora.

The primary risk of instilling models with type knowledge lies in the potential for misinformation (Weidinger et al., 2021). For example, if our model is used to extend existing taxonomies, it runs the risk of hallucinating false types. We observe in Table 6 that while our model achieves high typing precision and recall for seen and unseen types in new documents, we are not at the point where it can be used in isolation to discover and add knowledge to existing knowledge graphs. In parallel with developing better methods for verifying type ontologies and assignments, it is important to incorporate domain experts or crowd-source verification when language models are used to discover facts or type relationships in new documents.

We also advocate for more careful inspection of racial, gender, and socioeconomic biases in existing type ontologies, as it is possible for type-aware models to propagate such biases (e.g., associating people with certain patterns of names with specific occupations).