Modeling Fine-Grained Entity Types with Box Embeddings

Yasumasa Onoe[♠], **Michael Boratko**[♢], **Andrew McCallum**[♢♣], **Greg Durrett**[♠]
[♠]The University of Texas at Austin
[♢]University of Massachusetts Amherst
[♣]Google Research
{yasumasa,gdurrett}@cs.utexas.edu
{mboratko,mccallum}@cs.umass.edu
mccallum@google.com

Abstract

Neural entity typing models typically represent fine-grained entity types as vectors in a high-dimensional space, but such spaces are not well-suited to modeling these types' complex interdependencies. We study the ability of *box embeddings*, which embed concepts as d-dimensional hyperrectangles, to capture hierarchies of types even when these relationships are not defined explicitly in the ontology. Our model represents both types and entity mentions as boxes. Each mention and its context are fed into a BERT-based model to embed that mention in our box space; essentially, this model leverages typological clues present in the surface text to hypothesize a type representation for the mention. Box containment can then be used to derive both the posterior probability of a mention exhibiting a given type and the conditional probability relations between types themselves. We compare our approach with a vector-based typing model and observe state-of-the-art performance on several entity typing benchmarks. In addition to competitive typing performance, our box-based model shows better performance in prediction consistency (predicting a supertype and a subtype together) and confidence (i.e., calibration), demonstrating that the box-based model captures the latent type hierarchies better than the vector-based model does.[1]

1 Introduction

The development of named entity recognition and entity typing has been characterized by a growth in the size and complexity of type sets: from 4 (Tjong Kim Sang and De Meulder, 2003) to 17 (Hovy et al., 2006) to hundreds (Weischedel and Brunstein, 2005; Ling and Weld, 2012) or thousands (Choi et al., 2018). These types follow some kind of hierarchical structure (Weischedel and Brunstein, 2005; Ling and Weld, 2012; Gillick et al., 2014; Murty et al., 2018), so effective models for these tasks frequently engage with this hierarchy explicitly. Prior systems incorporate this structure via hierarchical losses (Murty et al., 2018; Xu and Barbosa, 2018; Chen et al., 2020) or by embedding types into a high-dimensional Euclidean or hyperbolic space (Yogatama et al., 2015; López and Strube, 2020). However, the former approach requires prior knowledge of the type hierarchy, which is unsuitable for a recent class of large type sets where the hierarchy is not explicit (Choi et al., 2018; Onoe and Durrett, 2020a). The latter approaches, while leveraging the inductive bias of hyperbolic space to represent trees, lack a probabilistic interpretation of the embedding and do not naturally capture all of the complex type relationships beyond strict containment.

In this paper, we describe an approach that represents entity types with *box embeddings* in a high-dimensional space (Vilnis et al., 2018). We build an entity typing model that jointly embeds each entity mention and entity types into the same box space to determine the relation between them. Volumes of boxes correspond to probabilities and taking intersections of boxes corresponds to computing joint distributions, which allows us to model mention-type relations (what types does this mention exhibit?) and type-type relations (what is the type hierarchy?). Concretely, we can compute the conditional probability of a type given the entity mention with straightforward volume calculations, allowing us to construct a probabilistic type classification model.

Compared to embedding types as points in Euclidean space (Ren et al., 2016a), the box space is expressive and suitable for representing entity types due to its geometric properties. Boxes can nest, overlap, or be completely disjoint to capture

[1]The code is available at https://github.com/yasumasaonoe/Box4Types.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 2051–2064
August 1–6, 2021.

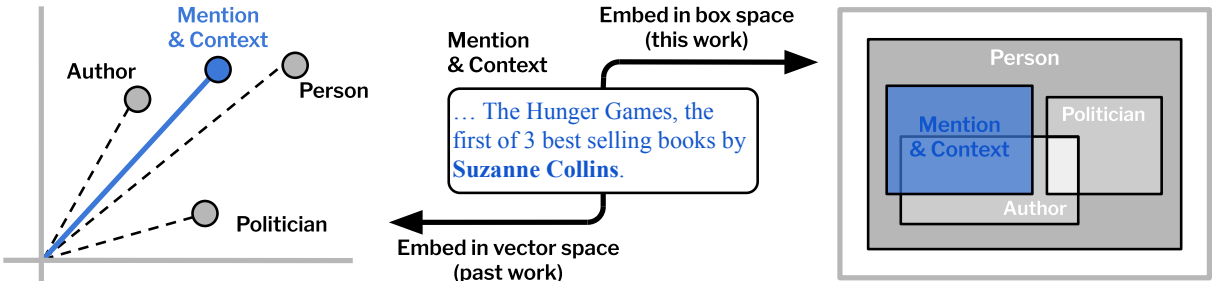


Figure 1: A mention (**Suzanne Collins**) and three entity types are embedded into a vector space (left) and a box space (right). The box space can more richly represent hierarchical interactions between types and uncertainty about the properties of the mention.

subtype, correlation, or disjunction relations, properties which are not explicitly manifested in Euclidean space. The nature of the box computation also allows these complex relations to be represented in a lower-dimensional space than needed by vector-based models.

In our experiments, we focus on comparing our box-based model against a vector-based baseline. We evaluate on four entity typing benchmarks: Ultra-fine Entity Typing (Choi et al., 2018), OntoNotes (Gillick et al., 2014), BBN (Weischedel and Brunstein, 2005), and FIGER (Ling and Weld, 2012). To understand the behavior of box embeddings, we further analyze the model outputs in terms of consistency (predicting coherent supertypes and subtypes together), robustness (sensitivity against label noise), and calibration (i.e., model confidence). Lastly, we compare entity representations obtained by the box-based and vector-based models. Our box-based model outperforms the vector-based model on two benchmarks, Ultra-fine Entity Typing and OntoNotes, achieving state-of-the-art-performance. In our other experiments, the box-based model also performs better at predicting supertypes and subtypes consistently and being robust against label noise, indicating that our approach is capable of capturing the latent hierarchical structure in entity types.

2 Motivation

When predicting class labels like entity types that exhibit a hierarchical structure, we naturally want our model's output layer to be sensitive to this structure. Previous work (Ren et al., 2016a; Shimaoka et al., 2017; Choi et al., 2018; Onoe and Durrett, 2019, inter alia) has fundamentally treated types as vectors, as shown in the left half of Figure 1. As is standard in multiclass or multi-label classification, the output layer of these models typically involves taking a dot product between a mention embedding and each possible type. A type could be more general and predicted on more examples by having higher norm,[2] but it is hard for these representations to capture that a coarse type like `Person` will have many mutually orthogonal subtypes.

By contrast, box embeddings naturally represent these kinds of hierarchies as shown in the right half of Figure 1. A box that is completely contained in another box is a strict subtype of that box: any entity exhibiting the inner type will exhibit the outer one as well. Overlapping boxes like `Politician` and `Author` represent types that are not related in the type hierarchy but which are not mutually exclusive. The geometric structure of boxes enables complex interactions with only a moderate number of dimensions (Dasgupta et al., 2020). Vilnis et al. (2018) also define a probability measure over the box space, endowing it with probabilistic semantics. If the boxes are restricted to a unit hypercube, for example, the volumes of type boxes represent priors on types and intersections capture joint probabilities, which can then be used to derive conditional probabilities.

Critically, box embeddings have previously been trained *explicitly* to reproduce a given hierarchy such as WordNet. A central question of this work is whether box embeddings can be extended to model the hierarchies and type relationships that are **implicit** in entity typing data: we **do not** assume access to explicit knowledge of a hierarchy during training. While some datasets such as OntoNotes have orderly ontologies, recent work on entity typing has often focused on noisy type sets from crowdworkers (Choi et al., 2018) or derived from Wikipedia (Onoe and Durrett, 2020a). We show that box embeddings can learn these structures organically; in fact, they are not restricted to only tree structures, but enable a natural Venn-diagram style of representation for concepts, as

[2] We do not actually observe this in our vector-based model.

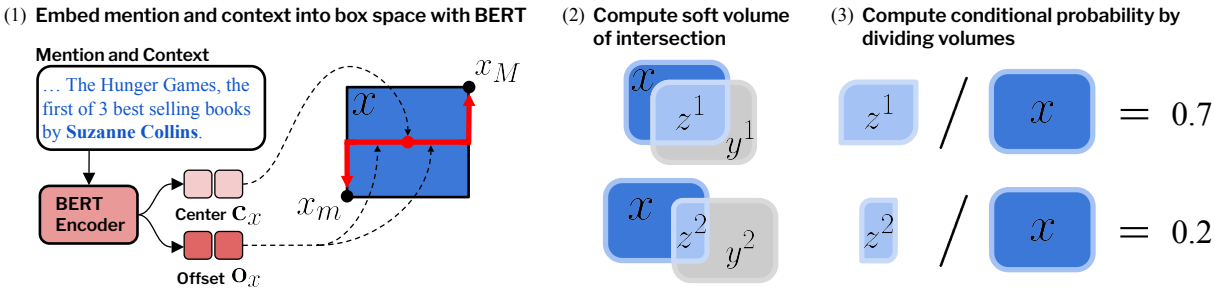


Figure 2: Box-based entity typing model. The mention and context (left) are embedded into the box space and probabilities for each type are computed with a soft volume computation.

with `Politician` and `Author` in Figure 1.

3 Type Modeling with Boxes

3.1 Background: Box Embeddings

Our box embeddings represent entity types as n-dimensional hyperrectangles. A box x is characterized by two points (x_m, x_M), where $x_m, x_M \in \mathbb{R}^d$ are the minimum and the maximum corners of the box x and $x_{m,i} \leq x_{M,i}$ for each coordinate $i \in \{1, ..., d\}$. The volume of the box x is computed as $\text{Vol}(x) = \prod_i (x_{M,i} - x_{m,i})$. If we normalize the volume of the box space to be 1, we can interpret the volume of each box as the marginal probability of a mention exhibiting the given entity type. Furthermore, the intersection volume between two boxes, x and y, is defined as $\text{Vol}(x \cap y) = \prod_i \max\left(\min(x_{M,i}, y_{M,i}) - \max(x_{m,i}, y_{m,i}), 0\right)$ and can be seen as the joint probability of entity types x and y. Thus, we can obtain the conditional probability $P(y \mid x) = \frac{\text{Vol}(x \cap y)}{\text{Vol}(x)}$.

Soft boxes Computing conditional probabilities based on hard intersection poses some practical difficulties in the context of machine learning: sparse gradients caused by disjoint or completely contained boxes prevent gradient-based optimization methods from working effectively. To ensure that gradients always flow for disjoint boxes, Li et al. (2019) relax the hard edges of the boxes using Gaussian convolution. We follow the more recent approach of Dasgupta et al. (2020), who further improve training of box embeddings using max and min Gumbel distributions (i.e., Gumbel boxes) to represent the min and max coordinates of a box.

3.2 Box-based Multi-label Type Classifier

Let s denote a sequence of context words and m denote an entity mention span in s. Given the input tuple (m, s), the output of the entity typing model is an arbitrary number of predicted types $\{t_0, t_1, ...\} \in \mathcal{T}$, where t_k is an entity type belonging to a type inventory $\mathcal{T}$. Because we do not assume an explicit type hierarchy, we treat entity typing as a multi-label classification problem, or $|\mathcal{T}|$ independent binary classification problems for each mention.

Section 3.3 will describe how to use a BERT-based model to predict a mention and context box[3] x from (m, s). For now, we assume x is given and we are computing the probability of that mention exhibiting the kth entity type, with type box y^k. Each type $t^k \in \mathcal{T}$ has a dedicated box y^k, which is parameterized by a center vector $\mathbf{c}_y^k \in \mathbb{R}^d$ and an offset vector $\mathbf{o}_y^k \in \mathbb{R}^d$. The minimum and maximum corners of a box y^k are computed as $y_m^k = \sigma(\mathbf{c}_y^k - \text{softplus}(\mathbf{o}_y^k))$ and $y_M^k = \sigma(\mathbf{c}_y^k + \text{softplus}(\mathbf{o}_y^k))$ respectively, so that parameters $\mathbf{c} \in \mathbb{R}^d$ and $\mathbf{o} \in \mathbb{R}^d$ yield a valid box with nonzero volume.

The conditional probability of the type t^k given the mention and context (m, s) is calculated as

$$p_\theta(t^k \mid m, s) = \frac{\text{Vol}(z^k)}{\text{Vol}(x)} = \frac{\text{Vol}(x \cap y^k)}{\text{Vol}(x)},$$

where z^k is the intersection between x and y^k ((2) and (3) in Figure 2). Our final type predictions are based on thresholding these probabilities; i.e., predict the type if $p > 0.5$.

As mentioned in Section 3.1, we use the Gumbel box approach of Dasgupta et al. (2020), in which the box coordinates are interpreted as the location parameter of a Gumbel max (resp. min) distribution with variance β. In this approach, the intersection

[3] We could represent mentions as points instead of boxes; however, representing them as boxes enables the size of a mention box to naturally reflect epistemic uncertainty about a mention's types given limited information.

box coordinates become

$$z_m^k = \beta \ln \left(e^{\frac{x_m}{\beta}} + e^{\frac{y_m^k}{\beta}} \right),$$

$$z_M^k = -\beta \ln \left(e^{-\frac{x_M}{\beta}} + e^{-\frac{y_M^k}{\beta}} \right).$$

Following Dasgupta et al. (2020), we approximate the expected volume of a Gumbel box using a softplus function:

$$\mathrm{Vol}(x) \approx \prod_i \mathrm{softplus}\left(\frac{x_{M,i} - x_{m,i}}{\beta} - 2\gamma \right),$$

where i is an index of each coordinate and $\gamma \approx 0.5772$ is the Euler–Mascheroni constant,[4] and $\mathrm{softplus}(x) = \frac{1}{t}\log(1 + \exp(xt))$, with t as an inverse temperature value.

3.3 Mention and Context Encoder

We format the context words s and the mention span m as $\mathbf{x} =$ `[CLS]` m `[SEP]` s `[SEP]` and chunk into WordPiece tokens (Wu et al., 2016). Using pre-trained BERT[5] (Devlin et al., 2019), we encode the whole sequence into a single vector by taking the hidden vector at the `[CLS]` token. A highway layer (Srivastava et al., 2015) projects down the hidden vector $\mathbf{h}^{\texttt{[CLS]}} \in \mathbb{R}^\ell$ to the $\mathbb{R}^{2d}$ space, where ℓ is the hidden dimension of the encoder (BERT), and d is the dimension of the box space. This highway layer transforms representations in a vector space to the box space without impeding the gradient flow. We further split the hidden vector $\bar{\mathbf{h}} \in \mathbb{R}^{2d}$ into two vectors: the center point of the box $\mathbf{c}_x \in \mathbb{R}^d$ and the offset from the maximum and minimum corners $\mathbf{o}_x \in \mathbb{R}^d$. The minimum and maximum corners of the mention and context box are computed as $x_m = \sigma(\mathbf{c}_x - \textsc{Softplus}(\mathbf{o}_x))$ and $x_M = \sigma(\mathbf{c}_x + \textsc{Softplus}(\mathbf{o}_x))$, where σ is an element-wise sigmoid function, and Softplus is an element-wise softplus function as defined in Section 3.2 ((1) in Figure 2). The output of the softplus is guaranteed to be positive, guaranteeing that the boxes have volume greater than zero.

3.4 Learning

The goal of training is to find a set of parameters θ that minimizes the sum of binary cross-entropy losses over all types over all examples in our training dataset $\mathcal{D}$:

$$\mathcal{L} = -\sum_{(m,s,\mathbf{t})\in\mathcal{D}} \sum_k t^k_{\text{gold}} \cdot \log p_\theta(t^k \mid m, s) + (1 - t^k_{\text{gold}}) \cdot \log(1 - p_\theta(t^k \mid m, s)),$$

where $t^k_{\text{gold}} \in \{0, 1\}$ is the gold label for the type t^k. We optimize this objective using gradient-based optimization algorithms such as Adam (Kingma and Ba, 2015).[6]

4 Experimental Setup

Our focus here is to shed light on the difference between type hierarchies learned by the box-based model and the vector-based model. To this end, we first evaluate those two models on standard entity typing datasets. Then, we test models' *consistency*, *robustness*, and *calibration*, and evaluate the predicted types as entity representations on a downstream task (coreference resolution). See Appendix A for hyperparameters.

4.1 Baseline

Our chief comparison is between box-based and vector-based modeling of entity types. As our main baseline for all experiments, we use a **vector-based** version of our entity typing model. We use the same mention and context encoder followed by a highway layer, but this baseline has vector-based type embeddings (i.e., a $|\mathcal{T}| \times d'$ matrix), and type predictions are given by a dot product between the type embeddings and the mention and context representation followed by element-wise logistic regression. This model is identical to that of Onoe and Durrett (2020b) except for the additional highway layer.

4.2 Evaluation and Datasets

Entity Typing We evaluate our approach on the Ultra-Fine Entity Typing (UFET) dataset (Choi et al., 2018) with the standard splits (2k for each of train, dev, and test). In addition to the manually annotated training examples, we use the denoised distantly annotated training examples from Onoe and Durrett (2019).[7] This dataset contains 10,331 entity types, and each type is marked as one of the three classes: *coarse*, *fine*, and *ultra-fine*. Note

[4] From Dasgupta et al. (2020), the Euler-Mascheroni constant appears due to the interpretation of $x_{m,i}, x_{M,i}$ as the *location* parameters of Gumbel distributions.

[5] We use BERT-large uncased (whole word masking) in our experiments.

[6] With large type sets, most types are highly skewed towards the negative class (>99% negative for many fine-grained types). While past work such as Choi et al. (2018) has used modified training objectives to handle this class imbalance, we did not find any modification to be necessary.

[7] This consists of 727k training examples derived from the distantly labeled UFET data.

that this classification **does not provide explicit hierarchies** in the types, and all classes are treated equally during training.

Additionally, we test our box-based model on three other entity typing benchmarks that have relatively simpler entity type inventories with **known hierarchies**, namely OntoNotes (Gillick et al., 2014), BBN (Weischedel and Brunstein, 2005) , and FIGER (Ling and Weld, 2012). See Appendix B for more details on these datasets.

Consistency A model that captures hierarchical structure should be aware of the relationships between supertypes and subtypes. When a model predicts a subtype, we want it to predict the corresponding supertype together, even when this is not explicitly enforced as a constraint or consistently demonstrated in the data, such as in the UFET dataset. That is, when a model predicts `artist`, `person` should also be predicted. To check this ability, we analyze the model predictions on the UFET dev set. We select 30 subtypes from the UFET type inventory and annotate corresponding supertypes for them in cases where these relationships are clear, based on their cooccurrence in the UFET training set and human intuition. Based on the 30 pairs, we compute accuracy of predicting supertypes and subtypes together. Table 10 in Appendix C lists the 30 pairs.

Robustness Entity typing datasets with very large ontologies like UFET are noisy; does our box-based model's notion of hierarchy do a better job of handling intrinsic noise in a dataset? To test this in a controlled fashion, we synthetically create noisy labels by randomly dropping the gold labels with probability $\frac{1}{3}$.[8] We derive two noisy training sets from the UFET training set: 1) adding noise to the *coarse* types and 2) adding noise to *fine* & *ultra-fine* types. We train on these noised datasets and evaluate on the standard UFET dev set.

Calibration Desai and Durrett (2020) study calibration of pre-trained Transformers such as BERT and RoBERTa (Liu et al., 2019) on natural language inference, paraphrase detection, and commonsense reasoning. In a similar manner, we investigate if our box-based entity typing model is calibrated: do the probabilities assigned to types by the model match the empirical likelihoods of those types? Since models may naturally have different scales for their logits depending on how long they are trained, we post-hoc calibrate each of our models using temperature scaling (Guo et al., 2017) and a shift parameter. We report the total error (e.g., the sum of the errors between the mean confidence and the empirical accuracy) on the UFET dev set and the OntoNotes dev set.

[8] If this causes the gold type set to be empty, we retain the original gold type(s); however, this case is rare.

Model	P	R	F1
Box	52.8	**38.8**	**44.8**
Vector	**53.0**	36.3	43.1
Choi et al. (2018)	47.1	24.2	32.0
Label GCN (Xiong et al., 2019)	50.3	29.2	36.9
ELMo (Onoe and Durrett, 2019)	51.5	33.0	40.2
BERT-base (Onoe and Durrett, 2019)	51.6	33.0	40.2

Table 1: Macro-averaged P/R/F1 on the test set for the ultra-fine entity typing task of Choi et al. (2018).

Entity Representations We are interested in the usefulness of the trained entity typing models in a downstream task. Following Onoe and Durrett (2020b), we evaluate entity representation given by the box-based and vector-based models on the Coreference Arc Prediction (CAP) task (Chen et al., 2019) derived from PreCo (Chen et al., 2018). This task is a binary classification problem, requiring to judge if two mention spans (either in one sentence or two sentences) are the same entity or not. As in Onoe and Durrett (2020b), we obtain type predictions (a vector of probabilities associated with types) for each span and use it as an entity representation. The final prediction of coreference for a pair of mentions is given by the cosine similarity between the entity type probability vectors with a threshold 0.5. The original data split provides 8k examples for each of the training, dev, and test sets. We report accuracy on the CAP test set.

5 Results and Discussion

5.1 Entity Typing

Here we report entity typing performance on Ultra-Fine Entity Typing (UFET), OntoNotes, FIGER, and BBN. For each dataset, we select the best model from 5 runs with different random seeds based on the development performance.

UFET Table 1 shows the macro-precision, recall, and F1 scores on the UFET test set. Our box-based model outperforms the vector-based model and state-of-the-art systems in terms of macro-

Model	Total			Coarse			Fine			Ultra-Fine		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Box	52.9	**39.1**	**45.0**	71.2	**82.5**	**76.4**	50.9	**55.2**	**53.0**	**45.4**	**24.5**	**31.9**
Vector	**53.3**	36.7	43.5	**71.7**	79.9	75.6	**51.9**	48.5	50.2	43.7	22.7	29.8
Choi et al. (2018)	48.1	23.2	31.3	60.3	61.6	61.0	40.4	38.4	39.4	42.8	8.8	14.6
Label GCN (Xiong et al., 2019)	49.3	28.1	35.8	66.2	68.8	67.5	43.9	40.7	42.2	42.4	14.2	21.3
ELMo (Onoe and Durrett, 2019)	50.7	33.1	40.1	66.9	80.7	73.2	41.7	46.2	43.8	45.6	17.4	25.2
HY XLarge (López and Strube, 2020)	43.4	34.2	38.2	61.4	73.9	67.1	35.7	46.6	40.4	36.5	19.9	25.7

Table 2: Macro-averaged P/R/F1 on the dev set for the entity typing task of Choi et al. (2018) comparing various systems. Our box-based model outperforms models from past work as well as our vector-based baseline.

F1.[9] Compared to the vector-based model, the box-based model improves primarily in macro-recall compared to macro-precision. Choi et al. (2018) is a LSTM-based model using GloVe (Pennington et al., 2014). On top of this model, Xiong et al. (2019) add a graph convolution layer to model type dependencies. Onoe and Durrett (2019) use ELMo (Peters et al., 2018) and apply denoising to fix label inconsistency in the distantly annotated data.

Note that past work on this dataset has used BERT-base (Onoe and Durrett, 2019). Work on other datasets has used ELMo and observed that BERT-based models have surprisingly underperformed (Lin and Ji, 2019). Some of the gain from our vector-based model can be attributed to our use of BERT-Large; however, our box model still achieves stronger performance than the corresponding vector-based version which uses the same pre-trained model.

Table 2 breaks down the performance into the *coarse*, *fine*, and *ultra-fine* classes. Our box-based model consistently outperforms the vector-based model in macro-recall and F1 across the three classes. The largest gap in macro-recall is in the *fine* class, leading to the largest gap in macro-F1 within the three classes.

We also list the numbers from prior work in Table 2. HY XLarge (López and Strube, 2020), a hyperbolic model designed to learn hierarchical structure in entity types, exceeds the performance of the models with similar sizes such as Choi et al. (2018) and Xiong et al. (2019) especially in macro-recall. In the *ultra-fine* class, both our box-based model and HY XLarge achieve higher macro-F1 compared to their vector-based counterparts.

One possible reason for the higher recall of our model is a stronger ability to model dependencies between types. Instead of failing to predict a highly correlated type, the model may be more likely to predict a complete, coherent set of types.

Other datasets Table 3 compares macro-F1 and micro-F1 on the OntoNotes, BBN, and FIGER test sets.[10] On OntoNotes, our box-based model achieves better performance than the vector-based model. Zhang et al. (2018) use document-level information, Chen et al. (2020) apply a hierarchical ranking loss that assumes prior knowledge of type hierarchies, and Lin and Ji (2019) propose an ELMo-based model with an attention layer over mention spans and train their model on the augmented data from Choi et al. (2018). Among the models trained only on the original OntoNotes training set, the box-based model achieves the highest macro-F1 and micro-F1.

The state-of-the-art system on BBN, the system of Chen et al. (2020) in the "undefined" setting, uses explicit knowledge of the type hierarchy. This is particularly relevant on the BBN dataset, where the training data is noisy and features training points with obviously conflicting labels like `person` and `organization`, which appear systematically in the data. To simulate constraints like the ones they use, we use three simple rules to modify our models' prediction: (1) dropping `person` if `organization` exists, (2) dropping `location` if `gpe` exists, and (3) replacing `facility` by `fac`, since both versions of this tag appear in the training set but only `fac` in the dev and test set. Our box-based model and the vector-based model perform similarly and both achieve results comparable with recent systems.

On FIGER, our box-based model shows lower performance compared to the vector-based model, though both are approaching comparable results

[9] We omit the test number of López and Strube (2020), since they report results broken down into coarse, fine, and ultra-fine types instead of an aggregated F1 value. However, based on the development results, their approach substantially underperforms the past work of Onoe and Durrett (2019) regardless.

[10] Note that our hyperparameters are optimized for macro F1 on OntoNotes.

Model	OntoNotes		BBN		FIGER	
	Ma-F1	Mi-F1	Ma-F1	Mi-F1	Ma-F1	Mi-F1
Box	77.3	70.9	78.7*	78.0*	79.4	75.0
Vector	76.2	68.9	78.3*	78.0*	81.6	77.0
Zhang et al. (2018)	72.1	66.5	75.7	75.1	78.7	75.5
Chen et al. (2020) (exclusive)	72.4	67.2	63.2	61.0	82.6	80.8
Chen et al. (2020) (undefined)	73.0	68.1	79.7	80.5	80.5	78.1
Lin and Ji (2019)	82.9†	77.3†	79.3	78.1	83.0	79.8

Table 3: Macro-averaged F1 and Micro-averaged F1 on the test set for the entity typing task of OntoNotes, BBN, FIGER. †: Not directly comparable since large-scale augmented data is used. *: We fix the predictions using simple rules post-hoc.

Model	BBN	FIGER
	Dev Ma-F1	Dev Ma-F1
Box	92.4	94.3
Vector	92.3	94.7

Table 4: Macro-averaged F1 on the dev set of BBN and FIGER. These dev sets are drawn from the same distributions as their training sets.

with state-of-the-art systems. We notice that some of the test examples have inconsistent labels (e.g., `/organization/sports_team` is present, but its supertype `/organization` is missing), penalizing models that predict the supertype correctly. In addition, FIGER, like BBN, has systematic shifts between training and test distributions. We hypothesize that our model's hyperparameters (tuned on OntoNotes only) are suboptimal. The high dev performance shown in Table 4 implies that our model optimized on held-out training examples may not capture these specific shifts as well as other models whose inductive biases are better suited to this unusually mislabeled data.

5.2 Consistency

One factor we can investigate is whether our model is able to predict type relations in a sensible, consistent fashion *independent of the ground truth for a particular example*. For this evaluation, we investigate our model's predictions on the UFET dev set. We count the number of occurrences for each subtype in 30 supertype/subtype pairs (see Table 10 in Appendix C). Then, for each subtype, we count how many times its corresponding supertype is also predicted. Although these supertype-subtype relations are not strictly defined in the training data, we believe they should nevertheless be exhibited by models' predictions. Accuracy is given by the ratio between those counts, indicating how often the supertype was correctly picked up.

Table 5 lists the total and per-supertype accuracy on the supertype/subtype pairs. We report the number of subtypes grouped by their supertypes to show their frequency (the "Count" column in Table 5). Our box-based model achieves better accuracy compared to the vector-based model on all supertypes. The gaps are particularly large on `place` and `organization`. Note that some of the UFET training examples have inconsistent labels (e.g., a subtype `team` can be a supertype `organization` or `group`), and this ambiguity potentially confuses a model during training. Even in those tricky cases, the box-based model shows reasonable performance. The geometry of the box space itself gives some evidence as to why this consistency would arise (see Section 5.6 for visualization of box edges).

5.3 Robustness

Table 6 analyzes models' sensitivity to the label noise. We list the UFET dev performance by models trained on the noised UFET training set. When the *coarse* types are noised (i.e., omitting some supertypes), the vector-based model loses 4.8 points of macro-F1 while our box-based model only loses 1.5 points. A similar trend can be seen when the *fine* and *ultra-fine* types are noised (i.e., omitting some subtypes). In both cases, the vector-based model shows lower recall compared to the same model trained on the clean data, while our box-based model is more robust. We also note that the vector-based model tends to overfit to the training data quickly. We hypothesize that the use of boxes works as a form of regularization, since moving boxes may be harder than moving points in a space, thus being less impacted by noisy labels.

5.4 Calibration

Following Nguyen and O'Connor (2015), we split model confidence (output probability) for each typing decision of each example into 10 bins (e.g., 0-0.1, 0.1-0.2 etc.). For each bin, we compute mean confidence and empirical accuracy. We show the total calibration error (lower is better) as well as the scaling and shifting constants in Table 7. As the results on UFET and OntoNotes show, both box-based and vector-based entity typing models can be

Supertype	Box		Vector	
	Count	Acc.	Count	Acc.
person	982	**99.7**	745	98.6
location	470	**86.1**	450	84.4
place	49	**95.9**	29	68.9
organization	496	**84.6**	407	77.8
Total	1,997	**92.7**	1,631	89.0

Table 5: Consistency: accuracy evaluated on the 30 supertype & subtypes pairs. The "Count" column shows the number of subtypes found in the predictions. The accuracy is the frequency of predicting the corresponding supertype when the subtype is exhibited.

Training Data	Model	P	R	F1	Δ in F1
Noised Coarse	Box	51.0	**37.9**	**43.5**	-1.5
	Vector	**51.5**	31.0	38.7	-4.8
Noised Fine & Ultra-fine	Box	53.0	**37.2**	**43.7**	-1.3
	Vector	**58.6**	30.6	40.2	-3.3

Table 6: Entity typing results of the UFET dev set. Models are trained on the noised UFET training set. The "Δ in F1" column shows the performance drop from the model trained on the original UFET training set (not noised).

reasonably well calibrated after applying temperature scaling and shifting. However, the box-based model achieves slightly lower total error.

5.5 Entity Representation for Coreference

This experiment evaluates if model outputs are immediately useful in a downstream task. For this task, we use the box-based and vector-based entity typing models trained on the UFET training set (i.e., we do not train models on the CAP training set). Table 8 shows the test accuracy on the CAP data. Our box-based model achieves slightly higher accuracy than the vector-based model, indicating that "out-of-the-box" entity representations obtained by the box-based model contains more useful features for the CAP task.[11]

5.6 Box Edges

To analyze how semantically related type boxes are located relative to one another in the box space, we plot the edges of the person and actor boxes along the 109 dimensions one by one. Figure 3 shows how those two boxes overlap each other in the high-dimensional box space. The upper plot

[11] Our results are not directly comparable to those of Onoe and Durrett (2020b); we train on the training set of UFET dataset, and they train on examples from the train, dev, and test sets.

Model	Scale / Shift	Total Error
UFET		
Box	0.5 / -1.1	**0.1119**
Vector	0.2 / -1.1	0.3279
OntoNotes		
Box	0.9 / -0.3	**0.1358**
Vector	0.7 / -0.4	0.1568

Table 7: Total calibration error on UFET and OntroNotes. We scale and shift logits post-hoc.

Model	Test Acc.
Box	**78.1**
Vector	77.3
Random	50.0

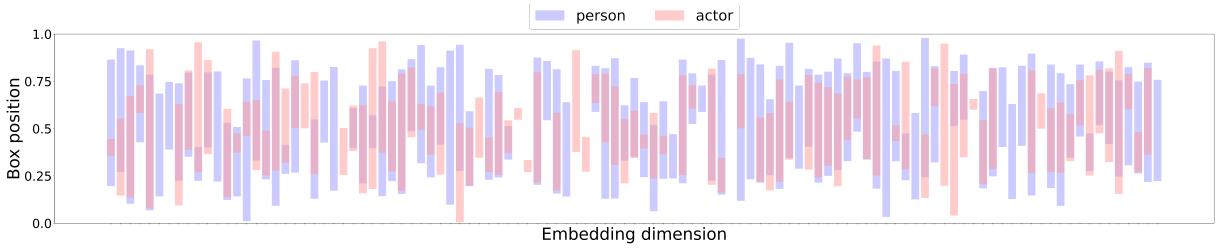
Table 8: Accuracy on the CAP test set (Chen et al., 2019). This is a binary classification task.

in Figure 3 compares the person box and the actor box learned on the UFET data. We can see that the edges of person contain the edges of actor in many dimensions but not all, meaning that the person box overlaps with the actor box but doesn't contain it perfectly as we might expect.

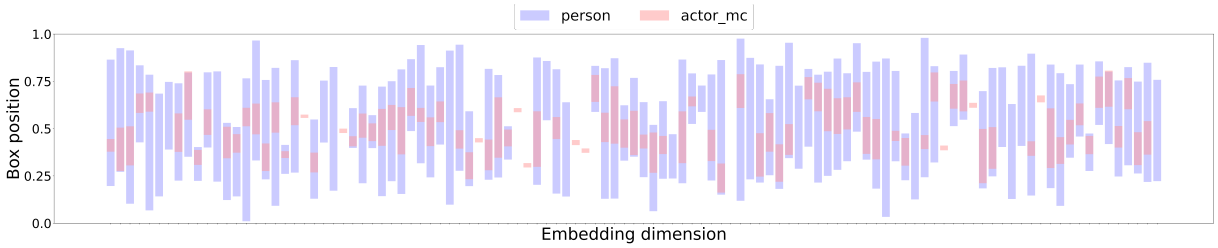
However, we can additionally investigate whether the actor box is *effectively* contained in the person for parts of the space actually used by the mention boxes. The lower plot in Figure 3 compares the person box and the minimum bounding box of the intersections between the actor and the mention and context boxes obtained using the UFET dev examples where the actor type is predicted. This minimum bounding box approximates the effective region within the actor box. Now the edges of actor are contained in the edges of person in the most of dimensions, indicating that the person box almost contains this "effective" actor box.

6 Related Work

Embeddings Embedding concepts/words into a high-dimensional vector space (Hinton, 1986) has a long history and has been an essential part of neural networks for language (Bengio et al., 2003; Collobert et al., 2011). There is similarly a long history of rethinking the semantics of these embedding spaces, such as treating words as regions using sparse count-based vectors (Erk, 2009a,b) or dense distributed vectors (Vilnis and McCallum, 2015). Order embeddings (Vendrov et al., 2016) or their probabilistic version (POE) (Lai and Hockenmaier, 2017) are one technique suited for hierarchical modeling. However, OE can only handle binary entailment decisions, and POE cannot model negative correlations between types, a critical limitation in its use as a probabilistic model; these shortcomings directly led to the development of box embeddings. Hyperbolic embeddings (Nickel and Kiela,



(a)



(b)

Figure 3: Edges of (a) the `person` box vs the `actor` box and (b) the `person` box vs the minimum bounding box of the intersections between mention & context boxes and the `actor` box.

2017; López and Strube, 2020) can also model hierarchical relationships as can hyperbolic entailment cones (Ganea et al., 2018); however, these approaches lack a probabilistic interpretation.

Recent work on knowledge base completion (Abboud et al., 2020) and reasoning over knowledge graphs (Ren et al., 2020) embeds relations or queries using box embeddings, but entities are still represented as vectors. In contrast, our model embed both entity mentions and types as boxes.

Entity typing Entity typing and named entity recognition (Tjong Kim Sang and De Meulder, 2003) are old problems in NLP. Recent work has focused chiefly on predicted fine-grained entity types (Ling and Weld, 2012; Gillick et al., 2014; Choi et al., 2018), as these convey significantly more information for downstream tasks. As a result, there is a challenge of scaling to large type inventories, which has inspired work on type embeddings (Ren et al., 2016a,b).

Entity typing information has been used across a range of NLP tasks, including models for entity linking and coreference (Durrett and Klein, 2014). Typing has been shown to be useful for cross-domain entity linking specifically (Gupta et al., 2017; Onoe and Durrett, 2020a). It has also recently been applied to coreference resolution (Onoe and Durrett, 2020b; Khosla and Rose, 2020) and text generation (Dong et al., 2020), suggesting that it can be a useful intermediate layer even in pre-trained neural models.

7 Conclusion

In this paper, we investigated a box-based model for fine-grained entity typing. By representing entity types in a box embedding space and projecting entity mentions into the same space, we can naturally capture the hierarchy of and correlations between entity types. Our experiments showed several benefits of box embeddings over the equivalent vector-based model, including typing performance, calibration, and robustness to noise.

Acknowledgments

Thanks to the members of the UT TAUR lab, Pengxiang Cheng, and Eunsol Choi for helpful discussion; Tongfei Chen and Ying Lin for providing the details of experiments. This work was also partially supported by NSF Grant IIS-1814522, NSF Grant SHF-1762299, and based on research in part supported by the Air Force Research Laboratory (AFRL), DARPA, for the KAIROS program under agreement number FA8750-19-2-1003, as well as University of Southern California subcontract no. 123875727 under Office of Naval Research prime contract no. N660011924032. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL, DARPA, or the U.S. Government.

References

Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. 2020. BoxE: A Box Embedding Model for Knowledge Base Completion. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A Neural Probabilistic Language Model. 3:1137–1155.

Lukas Biewald. 2020. Experiment Tracking with Weights and Biases. Software available from wandb.com.

Hong Chen, Zhenhua Fan, Hao Lu, Alan Yuille, and Shu Rong. 2018. PreCo: A large-scale dataset in preschool vocabulary for coreference resolution. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Mingda Chen, Zewei Chu, Yang Chen, Karl Stratos, and Kevin Gimpel. 2019. EntEval: A holistic evaluation benchmark for entity representations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Tongfei Chen, Yunmo Chen, and Benjamin Van Durme. 2020. Hierarchical Entity Typing via Multi-level Learning to Rank. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-Fine Entity Typing. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

Shib Sankar Dasgupta, Michael Boratko, Dongxu Zhang, Luke Vilnis, Xiang Lorraine Li, and Andrew McCallum. 2020. Improving Local Identifiability in Probabilistic Box Embeddings. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Shrey Desai and Greg Durrett. 2020. Calibration of Pre-trained Transformers. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Xiangyu Dong, Wenhao Yu, Chenguang Zhu, and Meng Jiang. 2020. Injecting Entity Types into Entity-Guided Text Generation. *ArXiv*, abs/2009.13401.

Greg Durrett and Dan Klein. 2014. A Joint Model for Entity Analysis: Coreference, Typing, and Linking. *Transactions of the Association for Computational Linguistics (TACL)*, 2:477–490.

Katrin Erk. 2009a. Representing words as regions in vector space. In *Proceedings of the Conference on Computational Natural Language Learning (CoNLL)*.

Katrin Erk. 2009b. Supporting inferences in semantic space: representing words as regions. In *Proceedings of the International Conference on Computational Semantics (IWCS)*.

Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. 2018. Hyperbolic Entailment Cones for Learning Hierarchical Embeddings. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh. 2014. Context-Dependent Fine-Grained Entity Type Tagging. *CoRR*, abs/1412.1820.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity Linking via Joint Encoding of Types, Descriptions, and Context. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Geoffrey E Hinton. 1986. Learning distributed representations of concepts. In *Proceedings of the eighth annual conference of the cognitive science society*.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. OntoNotes: The 90% Solution. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Sopan Khosla and Carolyn Rose. 2020. Using Type Information to Improve Entity Coreference Resolution. In *Proceedings of the First Workshop on Computational Approaches to Discourse*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representations (ICLR)*.

Alice Lai and Julia Hockenmaier. 2017. Learning to Predict Denotational Probabilities For Modeling Entailment. In *Proceedings of the Conference of the*

European Chapter of the Association for Computational Linguistics (EACL).

Lisha Li, Kevin G. Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. 2017. Hyperband: Bandit-Based Configuration Evaluation for Hyperparameter Optimization. In *International Conference on Learning Representations (ICLR)*.

Xiang Li, Luke Vilnis, Dongxu Zhang, Michael Boratko, , and Andrew McCallum. 2019. Smoothing the Geometry of Probabilistic box Embeddings. In *International Conference on Learning Representations (ICLR)*.

Ying Lin and Heng Ji. 2019. An Attentive Fine-Grained Entity Typing Model with Latent Type Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Xiao Ling and Daniel S. Weld. 2012. Fine-Grained Entity Recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

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*, abs/1907.11692.

Federico López and Michael Strube. 2020. A Fully Hyperbolic Neural Model for Hierarchical Multi-Class Classification. In *Findings of the Association for Computational Linguistics: EMNLP*.

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 *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Khanh Nguyen and Brendan O'Connor. 2015. Posterior Calibration and Exploratory Analysis for Natural Language Processing Models. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Maximilian Nickel and Douwe Kiela. 2017. Poincaré Embeddings for Learning Hierarchical Representations. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*.

Yasumasa Onoe and Greg Durrett. 2019. Learning to Denoise Distantly-Labeled Data for Entity Typing. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Yasumasa Onoe and Greg Durrett. 2020a. Fine-Grained Entity Typing for Domain Independent Entity Linking. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Yasumasa Onoe and Greg Durrett. 2020b. Interpretable Entity Representations through Large-Scale Typing. In *Findings of the Association for Computational Linguistics: EMNLP*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Hongyu Ren, Weihua Hu, and Jure Leskovec. 2020. Query2box: Reasoning over Knowledge Graphs in Vector Space using Box Embeddings. In *International Conference on Learning Representations (ICLR)*.

Xiang Ren, Wenqi He, Meng Qu, Lifu Huang, Heng Ji, and Jiawei Han. 2016a. AFET: Automatic Fine-Grained Entity Typing by Hierarchical Partial-Label Embedding. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Xiang Ren, Wenqi He, Meng Qu, Clare R. Voss, Heng Ji, and Jiawei Han. 2016b. Label Noise Reduction in Entity Typing by Heterogeneous Partial-Label Embedding. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Sonse Shimaoka, Pontus Stenetorp, Kentaro Inui, and Sebastian Riedel. 2017. Neural Architectures for Fine-grained Entity Type Classification. In *Proceedings of the Conference of the European Chapter of the Association for Computational Linguistics (EACL)*.

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway Networks. *ArXiv*, abs/1505.00387.

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 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2016. Order-embeddings of Images and Languagen. In *International Conference on Learning Representations (ICLR)*.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic Embedding of Knowledge Graphs with Box Lattice Measures. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Luke Vilnis and Andrew McCallum. 2015. Word Representations via Gaussian Embedding. In *International Conference on Learning Representations (ICLR)*.

Ralph Weischedel and Ada Brunstein. 2005. BBN pronoun coreference and entity type corpus. Linguistic Data Consortium.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Gregory S. Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *ArXiv*, abs/1609.08144.

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 *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Peng Xu and Denilson Barbosa. 2018. Neural Fine-Grained Entity Type Classification with Hierarchy-Aware Loss. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*.

Dani Yogatama, Daniel Gillick, and Nevena Lazic. 2015. Embedding methods for fine grained entity type classification. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Sheng Zhang, Kevin Duh, and Benjamin Van Durme. 2018. Fine-grained Entity Typing through Increased Discourse Context and Adaptive Classification Thresholds. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics (*SEM)*.

Appendix A: Hyperparameter Search

We use Bayesian hyperparameter tuning and the Hyperband stopping criteria (Li et al., 2017) implemented in the Weights & Biases software (Biewald, 2020). We use Adam (Kingma and Ba, 2015) for all experiments. We perform hyperparameter search on OntoNotes due to its fast convergence. This finds a lower dimension for the box-based model compared to the vector-based model (109-d $\times$ 2 vs 307-d), resulting fewer parameters in the box-based model. When we train the box-based model on the UFET dataset, we sample 1,000 negatives (i.e., wrong types) to speed up convergence; this is not effective in the vector-based model, so we do not do this there.

We use the same hyperparameters for the other three datasets. We train all models using NVIDIA V100 GPU with batch size 128. We implement our models using HuggingFace's Transformers library (Wolf et al., 2020).

Table 9 shows hyperparameters of the box-based and vector-based models as well as their ranges to search. For Adam, we use $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for training.

Model	Hyperparameter	Range	Selected
Box	Batch Size	{16, 32, 64, 128}	128
	lr (BERT)	-	2e-5
	lr (Other)	[0.0001, 0.01]	0.00372
	Box Dimension	[50, 250]	109
	Gumbel Temp.	[0.0001, 0.01]*	0.00036
	Softplus Temp.†	[0.1, 10]*	1.2471
Vector	Batch size	{16, 32, 64, 128}	128
	lr (BERT)	-	2e-5
	lr (Other)	[0.0001, 0.01]	0.00539
	Vector Dimension	[100, 500]	307

Table 9: Hyperparameters and their ranges. *: we use a log uniform distribution. †: Pytorch implementation of a softplus function takes inverse β.

Appendix B: Entity Typing Benchmarks

OntoNotes (Gillick et al., 2014) has 89 types with a 3-level hierarchy (e.g., `/location/geography/mountain`). We use the same splits (250k train / 2k dev / 9k test) provided by (Shimaoka et al., 2017). FIGER (Ling and Weld, 2012), derived from Wikipedia, uses 113 types with a 2-level hierarchy (e.g., `/person/musician`). We use the same splits (2M train / 10k dev / 563 test) as (Shimaoka et al., 2017). BBN (Weischedel and Brunstein, 2005) is based on the one million word Penn Treebank corpus from Wall Street Journal articles. We use the same splits (84k train / 2k dev / 14k test) as Ren et al. (2016b); Chen et al. (2020).

Appendix C: Supertype/subtype pairs

Table 10 shows the supertype/subtype pairs we manually annotated for our consistency test.

Supertype	Subtype
person	politician
person	athlete
person	leader
person	official
person	spokesperson
person	musician
person	actor
person	professional
person	male
person	female
location	country
location	city
location	area
location	region
location	position
location	space
location	district
location	territory
place	structure
place	building
organization	company
organization	institution
organization	government
organization	agency
organization	team
organization	administration
organization	military
organization	association
organization	social_group
organization	committee

Table 10: 30 supertype and subtype pairs used for the consistency test.

Appendix D: Box Edges

Similar to Figure 3, we plot the semantically unrelated type boxes `food` and `building` in Figure 4. These boxes are largely misaligned as expected, and the minimum bounding box of the intersections between the `building` and the mention and context boxes is also off from the `food` box.

Appendix E: Reliability Plot

Figure 5 visualizes the alignment between confidence and empirical accuracy on the UFET and OntoNotes dev sets.

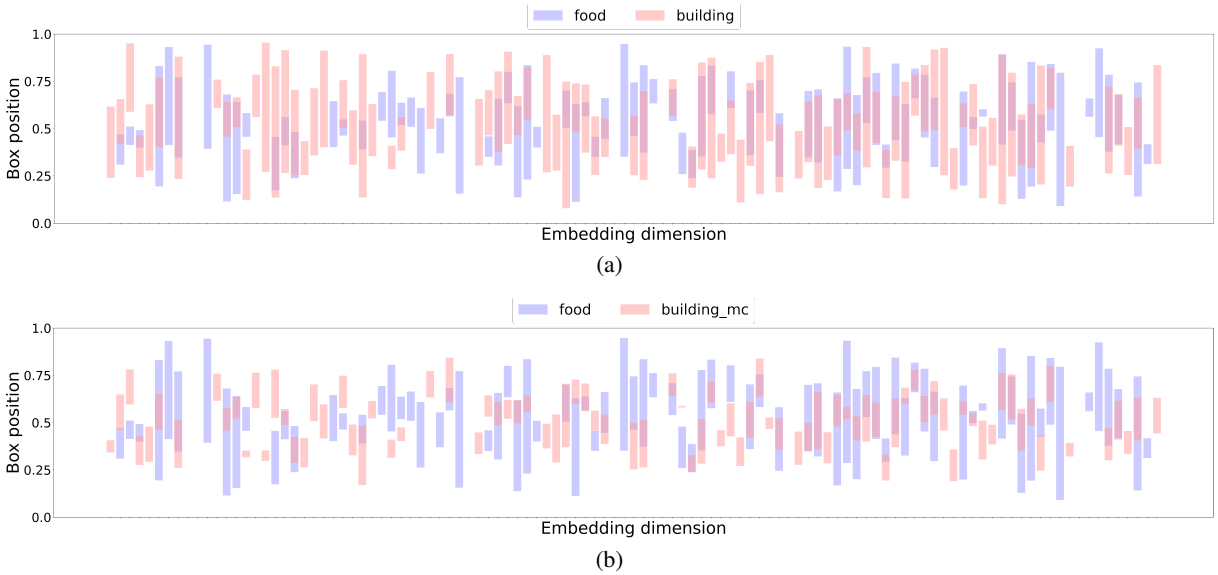


Figure 4: Edges of (a) the `food` box vs the `building` box and (b) the `food` box vs the minimum bounding box of the intersections between mention & context boxes and the `building` box.

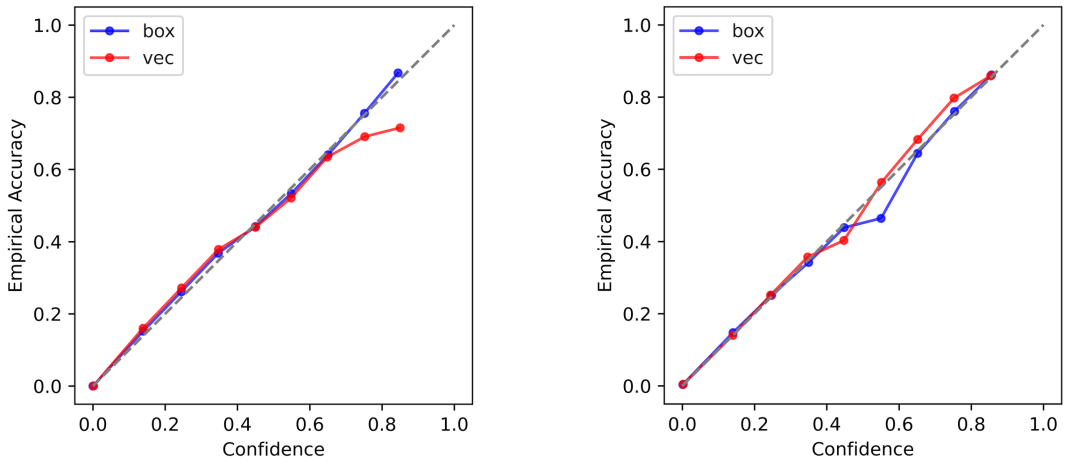


Figure 5: Reliability Plots on (a) UFET and (b) OntoNotes.