# Simultaneously Self-Attending to Text and Entities for Knowledge-Informed Text Representations

**Dung Thai**[1*], **Raghuveer Thirukovalluru**[1*], **Trapit Bansal**[1*], **Andrew McCallum**[1]

[1]UMass Amherst

`{dthai, rthirukovall, tbansal, mccallum}@cs.umass.edu`

## Abstract

Pre-trained language models have emerged as highly successful methods for learning good text representations. However, the amount of structured knowledge retained in such models, and how (if at all) it can be extracted, remains an open question. In this work, we aim at directly learning text representations which leverage structured knowledge about entities mentioned in the text. This can be particularly beneficial for downstream tasks which are knowledge-intensive. Our approach utilizes self-attention between words in the text and knowledge graph (KG) entities mentioned in the text. While existing methods require entity-linked data for pre-training, we train using a mention-span masking objective and a candidate ranking objective – which doesn't require any entity-links and only assumes access to an alias table for retrieving candidates, enabling large-scale pre-training. We show that the proposed model learns knowledge-informed text representations that yield improvements on the downstream tasks over existing methods.

## 1 Introduction

Self-supervised representation learning on large text corpora using language modeling objectives has been shown to yield generalizable representations that improve performance for many downstream tasks. Examples of such approaches include BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019b), XLNET (Yang et al., 2019), GPT-2 (Radford et al., 2019), T5 (Raffel et al., 2019) etc. However, whether such models retain structured knowledge in their representation is still an open question (Petroni et al., 2019; Poerner et al., 2019; Logan et al., 2019; Roberts et al., 2020) which has led to active research on knowledge-informed rep-

resentations (Zhang et al., 2019; Sun et al., 2019; Peters et al., 2019; Soares et al., 2019).

Models that learn knowledge-informed representations can be broadly classified into two categories. The first approach augments language model pre-training with the aim of storing structured knowledge in the model parameters. This is typically done by augmenting the pre-training task, for example by masking entity mentions (Sun et al., 2019) or enforcing representational similarity in sentences containing the same entities (Soares et al., 2019). While this makes minimal assumptions, it requires memorizing all facts encountered during training in the model parameters, necessitating larger models. The second approach directly conditions the representation on structured knowledge, for example fusing mention token representations with the mentioned entity's representation (Peters et al., 2019).

In this paper we consider the latter approach to learning knowledge-informed representations. Conditioning on relevant knowledge removes the burden on the model parameters to memorize all facts, and allows the model to encode novel facts not seen during training. However, existing methods typically assume access to entity-linked data for training (Zhang et al., 2019; Peters et al., 2019), which is scarce and expensive to annotate, preventing large scale pre-training. Moreover, these methods don't allow for bi-directional attention between both the text and the KG when representing text.

We propose a simple approach to incorporate structured knowledge into text representations. This is done using self-attention (Vaswani et al., 2017) to simultaneously attend to tokens in text and candidate KG entities mentioned in the text, in order to learn knowledge-informed representations after multiple layers of self-attention. The model is trained using a combination of a mention-masking objective and a weakly-supervised entity selection objective, which only requires access to an alias

---

table to generate candidate entities and doesn't assume any entity-linked data for training. We show that this objective allows the model to appropriately attend to relevant entities without explicit supervision for the linked entity and learn representations that perform competitively to models trained with entity-linked data.

We make the following contributions: (1) we propose KNowledge-Informed Transformers (KNIT), an approach to learn knowledge-informed text representations which does not require entity-linked data for training, (2) we train KNIT on a large corpora curated from the web with Wikidata as the knowledge graph, (3) we evaluate the approach on multiple tasks of entity typing and entity linking and show that it performs competitively or better than existing methods, yielding large improvements even while using $< 1\%$ of task-specific data for fine-tuning.

## 2  Related Works

BERT (Devlin et al., 2019) proposed a pre-training approach, called masked language modeling (MLM), which requires randomly replacing words in a sentence with a special [MASK] token and predicting the original masked tokens. RoBERTa (Liu et al., 2019b) trained a more robust BERT model on larger data. While MLM has been shown to learn general purpose representations, the amount of factual knowledge stored in such models is limited (Petroni et al., 2019; Poerner et al., 2019). Sun et al. (2019) propose a mention-masking objective which masks mentions of entities in a sentence, as opposed to random words, as a way of incorporating entity information into such models. Zhang et al. (2019) use entity-linked data and infuse representations of the linked entity in the final layer of the model to the representations of the corresponding entity mention. KnowBERT (Peters et al., 2019) learn an integrated entity linker that infuses entity representations into the word embedding input for the model and also relies on entity-linked data for training. K-Bert (Liu et al., 2019a) uses linked triples about entities in a sentence to inject knowledge. KGLM (Logan et al., 2019) proposed a fact-aware language model that selects and copies facts from KG for generation. Recently, Févry et al. (2020) introduced Entity-as-Experts (EAE), which is a masked language model coupled with an entity memory network. EAE learns to predict the entity spans, retrieves relevant entity memories and integrate them back to the Transformer layers. They also assume entity-linked data for training.

## 3  Knowledge-Informed Transformers (KNIT)

In this section, we describe the KNIT model as well as its training procedure. KNIT makes use of the mention-masking objective for training and conditions the encoder on both text as well as mentioned entities but does not assume any entity-linked data for training. Figure 1 shows the overall model.

### 3.1  Text and Entity Encoder

The input consists of a sentence along with *candidate* entities for the sentence. We first run a named entity extraction model on the sentence to extract mentions and then generate candidate entities based on cross-wikis (Ganea and Hofmann, 2017). We use a Wikipedia alias table for generating candidates, taken from Raiman and Raiman (2018). The start and end of mentions are demarcated using special tokens $\langle m \rangle$ and $\langle /m \rangle$. Given the text sequence $\{x_1, \ldots, x_n\}$ and the set of associated candidate entities for the sequence $\{e_1, \ldots, e_m\}$, we first embed the words and entities as vector embeddings. For entities, we use KG pre-trained embeddings (Lerer et al., 2019) and add a projection layer to upscale the entity embedding to the word embedding size. We will use Transformer self-attention (Vaswani et al., 2017) to encode both the text and the entities. Since self-attention has no notion of position in the sequence, it is common to concatenate a position embedding (Devlin et al., 2019) to the word embeddings. We follow this approach for the word embeddings. However, since the entities in the candidate set need to be encoded in a position-independent manner, we don't add any position embeddings to them. This entire sequence, position-dependent word embeddings and position-independent candidates, is passed through multiple layers of self-attention. The end result is contextualized token embeddings conditioned on the entities, $\{\tilde{x}_1, \ldots, \tilde{x}_n\}$, as well as candidate entity embeddings conditioned on the text $\{\tilde{e}_1, \ldots, \tilde{e}_m\}$.

### 3.2  Training

**Mention-masking**  While the approach described above has the potential to learn knowledge-conditioned text representations, it needs a correct pre-training objective to learn to use the extra information from the entities. Since large
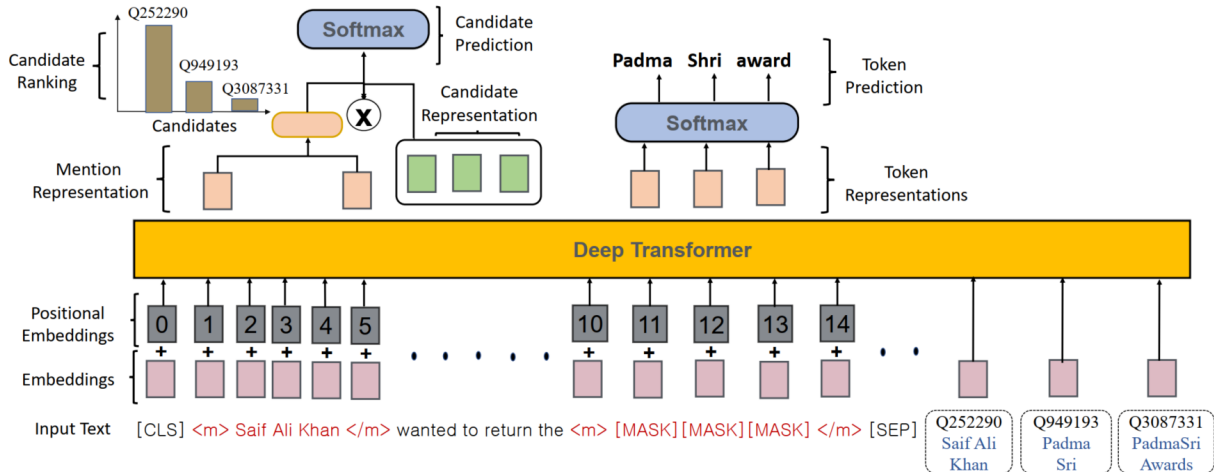
Figure 1: KNIT model with masked mention prediction and candidate ranking

|  | Random Tokens | Mention Tokens |
|---|---|---|
| RoBERTa | 58.8 | 23.1 |
| + MM | 61.6 | 25.7 |
| **KNIT** | 65.1 | 81.3 |

Table 1: Accuracy on predicting random tokens and entity mention tokens. While RoBERTa is highly accurate at predicting random words, it suffers when predicting mention tokens even when it is trained on a mention-masking (MM) objective.

Transformer models (Devlin et al., 2019) have a lot of parameters, they can be highly accurate at predicting random word tokens and thus directly using a MLM objective for training will not work as the model can ignore the entity embeddings. However, we find that, due to lack of factual knowledge, these models are not very good at predicting tokens of entity mentions. Table 1 shows this for RoBERTa (Liu et al., 2019b) model. Thus, mention-masking – predicting tokens of masked entity mentions, provides a better objective to learn to use the candidate entities and learn knowledge-informed representations. Note that in Table 1, even when RoBERTa is trained with mention-masking (+MM) it is unable to provide a high accuracy on predicting mention tokens. Thus including entity embeddings should provide enough context for the model to make correct predictions by using the entities, as reflected by the KNIT score in Table 1.

**Candidate Ranking**   To further enable the model to use the correct entities for a mention, we use a weak entity linking objective that forces the model to rank one of the entities, from the candidate set of a mention, higher than all other entities for the sentence. Consider the $i$-th mention in a sentence with $(m_{i1}, m_{i2})$ as the start and end indices of the mention in the sentence, and a candidate set of entities $C_i$ for this mention. We create a mention representation $\tilde{m}_i$ by concatenating $\tilde{x}_{m_{i1}}$ and $\tilde{x}_{m_{i2}}$. Now, given the representations, we score all entities for the mention $i$: $s_{ij} = W[\tilde{m}_i; \tilde{e}_j]$, where $W$ is a learnable weight matrix. To enforce the model to select one entity from the mention's candidates, we find the highest scoring entity, $\hat{e}_i = \arg\max_{j \in C_i} s_{ij}$, and use that as a target in a cross-entropy loss:

$$\mathcal{L}^{cr} = \text{cross\_entropy}(\text{softmax}(s_{ij}), \mathbb{I}_{\hat{e}_i}) \quad (1)$$

where the softmax is over all entities (not just for mention $i$) in the sentence and $\mathbb{I}_{\hat{e}_i}$ is a one-hot vector with 1 for the entity $\hat{e}_i$ and 0 everywhere else. This objective enforces the model to rank one candidate higher than others candidates for the same mention as well as candidates of other entities. Similar objective has been explored for dealing with noise in entity typing models (Xu and Barbosa, 2018; Abhishek et al., 2017). The overall objective is a combination of bert-style MLM, mention-masking (MM) and candidate ranking:

$$\mathcal{L}^{mlm} + \alpha \mathcal{L}^{mm} + \beta \mathcal{L}^{cr} \quad (2)$$

## 4   Experiments

Implementation details are in Supplementary. Code of our models is available here[1].

---

[1] Source Code: https://github.com/dungtn/KNIT

**Models evaluated:** (1) RoBERTa (Liu et al., 2019b): the model uses the MLM objective for pre-training; (2) RoBERTa + MM: this model uses the mention-masking objective (Sun et al., 2019) in addition to the MLM objective; (3) KNIT: this is the proposed model which uses MLM, mention-masking and candidate ranking for pre-training. We use RoBERTa-base architecture for all models due to lack of computation resources. We compare our method with existing state-of-the-art in knowledge-informed representations: Ernie (Zhang et al., 2019), KnowBERT (Peters et al., 2019) and RELIC (Ling et al., 2020).

| *OpenEntity* | Precision | Recall | F1 |
|---|---|---|---|
| RoBERTa | 76.91 | 73.84 | 75.34 |
| RoBERTa+MM | 74.67 | 74.63 | 74.65 |
| Ernie | 78.40 | 72.90 | 75.56 |
| KnowBert | 78.60 | 73.70 | **76.10** |
| **KNIT** | 76.48 | 75.76 | **76.10** |

| *FIGER* | Precision | Recall | F1 |
|---|---|---|---|
| RoBERTa | 66.89 | 88.12 | 76.05 |
| Ernie | 57.19 | 76.51 | 73.39 |
| **KNIT** | 68.09 | 88.12 | **76.80** |

Table 2: Micro-averaged scores on entity typing tasks.

| | **OpenEnt** (4%) | **FIGER** (0.5%) | **FIGER** (0.05%) |
|---|---|---|---|
| Roberta | $56.98_{\pm 4.71}$ | $69.69_{\pm 0.38}$ | $65.59_{\pm 1.65}$ |
| +MM | $60.16_{\pm 2.44}$ | $69.43_{\pm 0.62}$ | $65.96_{\pm 1.38}$ |
| **KNIT** | $\mathbf{63.97}_{\pm 1.59}$ | $\mathbf{71.37}_{\pm 0.14}$ | $\mathbf{67.40}_{\pm 0.41}$ |

Table 3: F1 score on entity typing when using only a fraction of the task-specific training data ($0.05\% - 4\%$).

## 4.1 Results on Entity Typing

Entity typing is the task of identifying the semantic type of a given mention. We evaluate on two Entity typing datasets - OpenEntity (Choi et al., 2018) and FIGER (Ling et al., 2015). OpenEntity is a crowd-sourced dataset comprising 9 general types and 121 fine-grained types. We follow (Zhang et al., 2019) and evaluate on the nine general entity types. FIGER is a distant supervised dataset comprising over 2M examples and 113 entity types. Experimental results are shown in Table 2. KNIT outperforms RoBERTa(Liu et al., 2019b), Ernie(Zhang et al., 2019), and RoBERTa+MM(Sun et al., 2019) while being comparable to KnowBert (Peters et al., 2019). Note that KNIT performs comparably to the

| *No Fine-tuning* | |
|---|---|
| Wiki Alias Table | 68.78 |
| Our Top Candidate | 70.66 |
| RELIC | 81.90 |
| **KNIT** | 82.71 |
| **KNIT +Wikilinks** | **90.32** |
| *Fine-tune (10% data)* | |
| **KNIT** | 92.04 |
| *Fine-tune (Full data)* | |
| RELIC | 94.90 |
| Févry et al. (2020) | **96.70** |
| Raiman and Raiman (2018) | 94.88 |
| Radhakrishnan et al. (2018) | 93.00 |
| Le and Titov (2018) | 93.07 |
| Ganea and Hofmann (2017) | 92.22 |
| **KNIT** | 92.87 |

Table 4: Entity linking accuracy under various fine-tuning scenarios.

state-of-the-art without utilizing any entity-linked data for pre-training, unlike (Peters et al., 2019).

To further evaluate the effectiveness of KNIT, we consider the scenario where only a fraction of the data is used for task-specific fine-tuning. For this, we sample equal number of examples per type to create the fine-tuning data. The models are fine-tuned using the sampled data but are evaluated on the entire test set. Table.3 shows that KNIT significantly outperforms RoBERTa(Liu et al., 2019b) and RoBERTa+MM in the data constrained cases.

## 4.2 Results on Entity Linking

We demonstrate that our pre-trained model can capture entity linking information. For this, we use the AIDA-CoNLL (Hoffart et al., 2011) dataset and evaluate the linking performance of the model *without any dataset-specific fine-tuning*. We also compare with a model that used wikipedia hyperlinks for supervision during pre-training (KNIT +Wikilinks). As shown in Table 4, KNIT improves upon the candidate ranking by 12.05% and 19.66% when partial entity linking supervision from Wiki linked-text data is available. Even without Wiki-linked data, it outperforms the best pre-trained model that considers mention context (RELIC) by 0.81%. To further explore the entity linking capacity of our model, we fine-tune the model and show that our model has competitive performance, even when using only 10% of the training data. When trained on

the entire dataset, we find RELIC performs better, potentially due to the use of entity-linked data in its pre-training.

## 5 Conclusion

We propose a simple approach to learn knowledge-informed text representations using self-attention between text and mentioned entities. Our approach does not rely on any entity-linked data for training, enabling large-scale pre-training. We show that the method learns better representations than competing approaches and also learns entity-linking without explicit linking supervision. In the future, it will be interesting to explore how such methods can be used to condition the text encoder on structured KG facts about entities.

## Acknowledgments

## References

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

Eunsol Choi, Omer Levy, Yejin Choi, and Luke Zettlemoyer. 2018. Ultra-fine entity typing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 87–96, Melbourne, Australia. Association for Computational Linguistics.

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 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Thibault Févry, Nicholas FitzGerald, Livio Baldini Soares, and Tom Kwiatkowski. 2020. Empirical evaluation of pretraining strategies for supervised entity linking. In *Automated Knowledge Base Construction*.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. *arXiv*.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordino, Hagen Fürstenau, Manfred Pinkal, Marc Spaniol, Bilyana Taneva, Stefan Thater, and Gerhard Weikum. 2011. Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 782–792, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Phong Le and Ivan Titov. 2018. Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1595–1604, Melbourne, Australia. Association for Computational Linguistics.

Adam Lerer, Ledell Wu, Jiajun Shen, Timothee Lacroix, Luca Wehrstedt, Abhijit Bose, and Alex Peysakhovich. 2019. PyTorch-BigGraph: A Large-scale Graph Embedding System. In *Proceedings of the 2nd SysML Conference*, Palo Alto, CA, USA.

Jeffrey Ling, Nicholas FitzGerald, Zifei Shan, Livio Baldini Soares, Thibault Févry, David Weiss, and Tom Kwiatkowski. 2020. Learning cross-context entity representations from text.

Xiao Ling, Sameer Singh, and Daniel S. Weld. 2015. Design challenges for entity linking. *Transactions of the Association for Computational Linguistics*, 3:315–328.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2019a. K-bert: Enabling language representation with knowledge graph. *arXiv preprint arXiv:1909.07606*.

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

Robert L. Logan, IV, Nelson F. Liu, Matthew E. Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife Hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Florence, Italy. Association for Computational Linguistics.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. Bert is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised qa. *arXiv preprint arXiv:1911.03681*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9.

Priya Radhakrishnan, Partha Talukdar, and Vasudeva Varma. 2018. ELDEN: Improved entity linking using densified knowledge graphs. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1844–1853, New Orleans, Louisiana. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: multilingual entity linking by neural type system evolution. *arXiv preprint arXiv:1802.01021*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model?

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. *arXiv preprint arXiv:1906.03158*.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Peng Xu and Denilson Barbosa. 2018. Neural fine-grained entity type classification with hierarchy-aware loss. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 16–25.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. Ernie: Enhanced language representation with informative entities. *arXiv preprint arXiv:1905.07129*.

# A  Appendices

## A.1  Implementation Details(Pretraining)

To train KNIT, we collect 16M sentences from Wikipedia. We also collect 28M sentences from news articles and tag them using the DiffBot Entity Linker [2]. We further reduce the size of the entity vocabulary to 595K and remove examples that have no entity mentions. We limit each context sentence to 512 tokens and no more than 5 mentions per sentence with at least 2 and at most 10 candidate entities per mention span.

We use pre-trained entity embeddings with dimension $d = 200$ from (Lerer et al., 2019) and keep them fixed during the course of KNIT training. We use Adam optimizer with learning rate $1e^{-4}$, polynomial decay scheduler with warm-up, and clip norm 10. We also tune hyper-parameters in Equation (2) and choose $\alpha = 1$ and $\beta = 10$. The code will be made available on github[3].

## A.2  Implementation Details(Entity Typing)

All results in Tables 2-3 are obtained by tuning a few hyperparameters - batch size, learning rate, dropout, attention dropout. Batch size was tuned in

---

[2]www.diffbot.com
[3]Code will be opensourced

| Dataset | Train | Validation | Test |
|---|---|---|---|
| OpenEntity | 1998 | 1,998 | 1,998 |
| Figer | 2,000,000 | 10,000 | 563 |
| OpenEnt(4%) | 82 | 1,998 | 1,998 |
| Figer(0.5%) | 11,300 | 10,000 | 563 |
| Figer(0.05%) | 1,130 | 10,000 | 563 |
| AIDA-CoNLL | 17,830 | 4,623 | 4,292 |

Table 5: Number of examples in Train, Validation and Test split of different datasets

range (16-64). Learning rate was tuned in (0.00001-0.0005). All dropouts were tuned sparsely in the range (0.1-0.3). During finetuning, we restrict the max number of candidates per mention to 10. Unlike pretraining, the entity embeddings were also finetuned during entity typing experiments and the best performing validation set checkpoint was used to generate test set results

Sample dataset creation for experiments of Table 3 were done using random seeds. Three different sample datasets were collected for each of OpenEnt(4%), Figer(0.5%) and Figer(0.05%). Each sample would comprise an equal number of examples per entity type but randomised across the three runs. Numbers reported in Table 3 correspond to mean and standard deviation values of the performance of the three sample dataset trained models on the test set.

### A.2.1 Datasets

The sizes of sample and original datasets are shown in Table 5.