MLMLM: Link Prediction with Mean Likelihood Masked Language Model

Louis Clouatre[*†], **Philippe Trempe**[*], **Amal Zouaq**[*], **Sarath Chandar**[* † ▽]
[*] Polytechnique Montréal
[†] Mila - Quebec AI Institute
[▽] Canada CIFAR AI Chair
{philippe.trempe,amal.zouaq,sarath.chandar}@polymtl.ca
{clouatrl,sarath.chandar}@mila.quebec.ca

Abstract

Knowledge Bases (KBs) are easy to query, verifiable, and interpretable. They however scale with man-hours and high-quality data. Masked Language Models (MLMs), such as BERT, scale with computing power as well as unstructured raw text data. The knowledge contained within these models is however not directly interpretable. We propose to perform link prediction with MLMs to address both the KBs scalability issues and the MLMs interpretability issues. By committing the knowledge embedded in MLMs to a KB, it becomes interpretable. To do that we introduce MLMLM, **M**ean **L**ikelihood **M**asked **L**anguage **M**odel, an approach comparing the mean likelihood of generating the different entities to perform link prediction in a tractable manner. We obtain State of the Art (SotA) results on the WN18RR dataset and SotA results on the Precision@1 metric on the WikidataM5 inductive and transductive setting. We also obtain convincing results on link prediction on previously unseen entities, making MLMLM a suitable approach to introducing new entities to a KB.

1 Introduction

1.1 Context

KBs have many desirable properties. They are easy to query, verifiable, and perhaps most importantly interpretable by humans. They however have one critical shortcoming, they are expensive to build, making them harder to scale. Indeed, modern KBs scale with high-quality data, manual labor, or a mix of both. Approaches that scale with available computation and the massive amounts of unstructured data that are being created and accumulated have proven invaluable in the recent deep learning boom.

Large pretrained MLMs (Devlin et al., 2018; Liu et al., 2019) have been shown to scale well with large amounts of unstructured text data as well as with computing power. They also have displayed some interesting emergent abilities, such as the ability to perform zero-shot question answering (Radford et al., 2019; Brown et al., 2020). This ability implies that the model parameters contain a large amount of factual knowledge that it can leverage to answer a wide variety of questions. However, that knowledge is hardly interpretable by humans, as it is hidden within the millions to billions of parameters of the language model.

By using MLMs to completes KBs, we can address both the issue of scalability of KBs and the issue of the interpretability of MLMs by committing knowledge of the latter to an interpretable format in the former. The MLM can learn new knowledge from the large amount of unstructured textual data that keeps being added to the World Wide Web and then be used to continually complete and update the KB. This has the very desirable effect of making the link prediction approach scale with both computational power and a large quantity of unstructured data, both of which show no sign of slowing down.

1.2 Problem Definition

Given an entity and a relation, we want to train an MLM to generate all entities completing the KB triple.

Several technical challenges had to be addressed to achieve proper link prediction with pretrained MLMs. The first one is tractability. It is well known that inference in the task of Link Prediction is extremely costly, to the point where validation and test sets are purposefully kept small and most datasets will shy away from containing millions of different entities (Wang et al., 2019). While smaller Link Prediction datasets (Dettmers et al., 2017; Toutanova and Chen, 2015) are limited to a few thousand entities, a dataset more representative of full sized KBs (Wang et al., 2019) would contain upwards of a million potential entities. A

Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4321–4331

model that would be used on such a dataset could not realistically require an inference step through an MLM for every potential entity completing a triplet. It is necessary to enable link prediction with as little inference to the model as possible, as performing inference on pretrained models is expensive. Otherwise, it could result in a model with no practical purposes, even if it obtained better leaderboard scores.

The second challenge has to do with the inference outputs format of the MLMs. The length of the output needs to be known at inference time (Devlin et al., 2018; Vaswani et al., 2017), making it hard to sample entities of varying lengths from it. The example "Horses like _" could be completed with the word "carrot" and "long runs", but those two answers require varying length of masked inputs to be filled. Work like Petroni et al. (2019) is limited to single token outputs, which is useful to probe the model for the presence of embedded knowledge, but is not usable in practice for tasks such as link prediction, as the missing entities will have variable lengths. Solutions have to be able to sample an MLM for entities of varying lengths to have practical applications.

Finally, the use of MLMs opens the door to performing link prediction on entities that have not been previously seen by the model or the KB. This permits the addition of new entities to a KB on top of the link prediction capacities. Some capability of such an approach with MLMs was previously demonstrated (Petroni et al., 2019) and other works have approached the task by generating and comparing entities embedding (Daza et al., 2020; Gupta et al., 2017; Wang et al., 2019) for different KB tasks. By generating entity embeddings from text, they permit apt representation for previously unseen entities that can then be compared to other, previously seen entities. Unlike previous approaches we forgo the entity embedding step and let the model directly output the entity. We show that our approach yields strong results with unseen entities of arbitrary lengths in this task and should be explored further.

1.3 Contribution

Our main contributions are summarized here:

- We propose MLMLM, a mean likelihood method to compare the likelihood of different text of different token lengths sampled from an MLM.
- We demonstrate the tractability of our approach, requiring only one inference step through the model to perform link prediction on any numbers of possible entities, something which was not previously possible with an MLM.
- We achieve SotA results on the WN18RR benchmark and the best Precision@1 on both the inductive and transductive setting of the WikidataM5 dataset.
- We demonstrate that our approach can generalize to previously unseen entities on all benchmarks.

2 Background and Related Work

2.1 Masked Language Models

Pretrained MLMs, popularized by BERT (Devlin et al., 2018), have seen tremendous success when applied to Natural Language Understanding (NLU) problems. They are pretrained on massive amount of unsupervised text data. Those models incorporate enormous amounts of language knowledge and world knowledge within their weights. This lets them be further tuned on challenging NLU tasks with great success. Being based on the transformer (Vaswani et al., 2017) encoder architecture, the output length of the model is equal to the input length. This makes it challenging to sample text of arbitrary length when using MLMs without knowing the length of the desired sample in advance.

2.2 Sampling from MLM

Sampling single words from an MLM is trivial. By adding a mask to the input, we can sample likelihoods for the whole vocabulary. This feature is used by several pieces of work to complete sentences, answer questions and more (Guu et al., 2020; Lewis et al., 2020; Petroni et al., 2019). Work to generate and evaluate multi-token spans from MLM has also yielded interesting results (Wang and Cho, 2019; Salazar et al., 2020). We are however unaware of any other approach to sampling and evaluating multi-token spans of variable length, which is necessary to properly accomplish the task of link-prediction in a single pass through the model.

2.3 A Re-evaluation of Knowledge Graph Completion Methods

Recently, Sun et al. (2020) has found that many of the SotA approaches to link prediction have used an inappropriate evaluation protocol. They have shown that the evaluation protocol typically used in the link prediction approaches assigns a perfect score to a constant output, by putting the correct entities on top during a tiebreaker. In essence, under this evaluation protocol, assigning a likelihood of 0 to all entities would yield a perfect reranking score, since the tiebreaker would put the target entity as the first prediction. This was shown to yield very inflated scores for many neural network based link prediction approaches (Nathani et al., 2019; Vu et al., 2019; Nguyen et al., 2017), as several of them output a large number of tied scores for the various entities. Entity-embedding based approaches (Balažević et al., 2019; Sun et al., 2019; Dettmers et al., 2018) do not suffer from this issue. While we have found that our approach does not suffer from this issue despite not being an entity-embedding approach, we will use the random evaluation protocol proposed by Sun et al. (2020) for all evaluations and compare against approaches that used a similar protocol to ensure the validity of the comparisons. This protocol is similar to the filtered setting (Bordes et al., 2013), with the difference that the rank among entities with tied scores is randomly assigned.

2.4 KG-BERT

KG-BERT (Yao et al., 2019) is an approach to KB tasks based on MLM. It successfully demonstrates the potential of leveraging these models' internal knowledge on KB tasks. They train a BERT model to classify whether an individual triple fed to the model is correct or not. In essence, they feed every single possible (h, r, ?) or (?, r, t) triple in a string format to the model to obtain all scores to be reranked. This can result in millions of inference steps on the MLM for a single triple completion depending on the size of the KB. KG-BERT has many advantages over our proposed approach. It uses the target entity in the input, thus giving the model more information to use at inference time. The problem that it solves is much simpler, reducing it to a simple sequence classification problem. With a similar setup, it is even likely that KG-BERT would yield better results than our proposed approach since it has access to more information and has a more straightforward training setup. Unfortunately, KG-BERT is not tractable on any reasonably large KB (see Figure 1). For a KB containing millions of entities, KG-BERT would require millions of inference step through the MLM model for every triple completion. In contrast, our approach requires only one inference step through the MLM model for every triple completion, by generating all logits required to obtain the likelihood of any potential entity at once. Modern KBs can contain millions of entities (Wang et al., 2019), which would translate in KG-BERT requiring hours to complete a single triplet on a GPU.

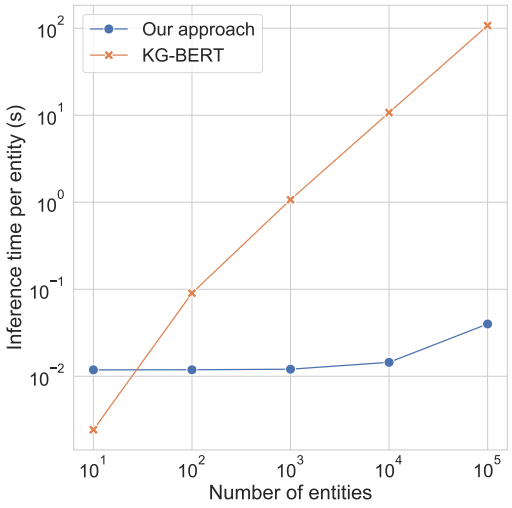

Figure 1: Approach Inference Time. This figure shows the per-entity inference time based on the total number of entities to be re-ranked, of MLMLM and KG-BERT, the most comparable approach.

3 Methodology

3.1 Overview

Our system performs link prediction. It uses MLM to generate all possible logits of all tokens required to generate all entities, and mean likelihood sampling to rerank all possible entities to complete the triple and perform link prediction. It can also be used to sample likelihoods for previously unseen entities.

Figure 2 shows a toy example of our inference setup. Our model outputs logits for the whole vocabulary of the RoBERTa-Large model. The outputs are simply the logits for all words and subwords from that vocabulary. This vocabulary is expressive enough to generate any English text.

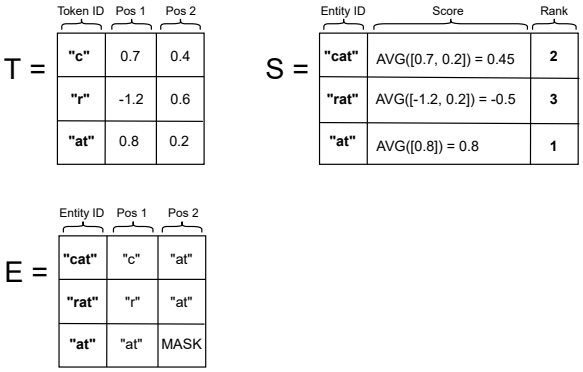

Figure 2: Ranking Example. The figure contains a minimal example of the ranking system. It represents the system in a KB containing 3 subwords in its vocabulary ['c', 'r', 'at'], 3 entities to rank ['cat', 'rat', 'at'] and a maximum entity length of 2.

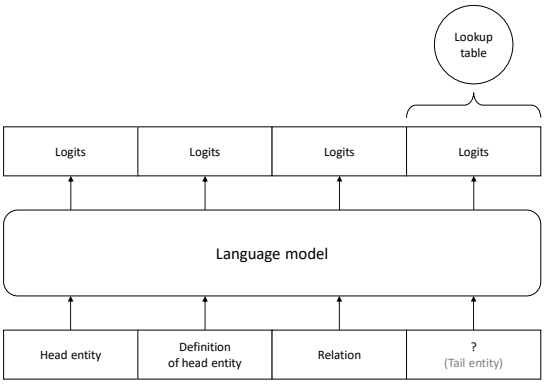

Figure 3: Lookup Table Generation For Tail Entity Prediction. The figure shows how the lookup table for tail entity prediction is generated. A string representation of the head entity and the relation are fed to the masked language model which outputs logits that represent the likelihood of finding each token at each possible position of the tail entity.

We decide in advance on a maximum length n for the maximum length of English text to generate. With a vocabulary of roughly 50,000 logits, the model would output a vector of logits of shape [50,000 x n]. Knowing in advance the tokens that would build the string for all possible entities, we can obtain a score for the likelihood of those entities completing a triplet by averaging the logits of those tokens. To evaluate the likelihood of the entity "brown dog", we would average the logits for the word "brown" on the first column of the matrix with the logits of the word "dog" on the second column of the matrix, ignoring all other columns of our logit matrix. Even if the model would never have encountered the entity "brown dog" it could still produce a score for said entity.

3.2 Data Pre-processing

The data pre-processing pipeline takes a link prediction dataset and transforms it into a generic format usable by the model. It is required that both the entity and relations have string representations. For every entity in the dataset, we extract an entity string, which uniquely identifies the entity, and a definition string, which is a textual description of the given entity. For every relation, we extract a relation string, which uniquely identifies and describes the relation.

We tokenize all strings through the pretrained RoBERTa tokenizer (Sennrich et al., 2016) and further transform the entity string by adding padding to match the longest tokenized entity within the dataset. Concretely, in a dataset where the longest entity has a length of 4 token ids, the entity string "dog" would be padded to have the representation "dog _ _ _" and the entity string "cat and dog" would have the representation "cat and dog _" where "_" is the padding token. The purpose of this padding is to standardise the masked representation of all entities, therefore letting the model treat all entities in the same manner.

3.3 Model

Our approach uses the RoBERTa-Large model (Liu et al., 2019) for all experiments. We finetune the pretrained model on the link prediction datasets to generate the logits of the unknown entities. As our approach does a single call to the model to rerank all possible entities, it is acceptable to use the larger model for better performance. Figure 3 shows the inference process for tail entity prediction. Similarly, the head entity prediction takes as input the head entity mask, the relation, the tail entity and the tail entity definition. We use the relation string, the known entity string and the entity definition of the known entity string to make the model generate the logits representing the unknown entity string.

3.4 Ranking System

The ranking system pictured in Figure 4 performs link prediction on a given triple. The MLM outputs logits for all possible token ids and positions for the missing entity to complete the triple. This forms the lookup table T. The link prediction dataset contains a list of all possible entities. The token ids forming those entities make up E. We obtain

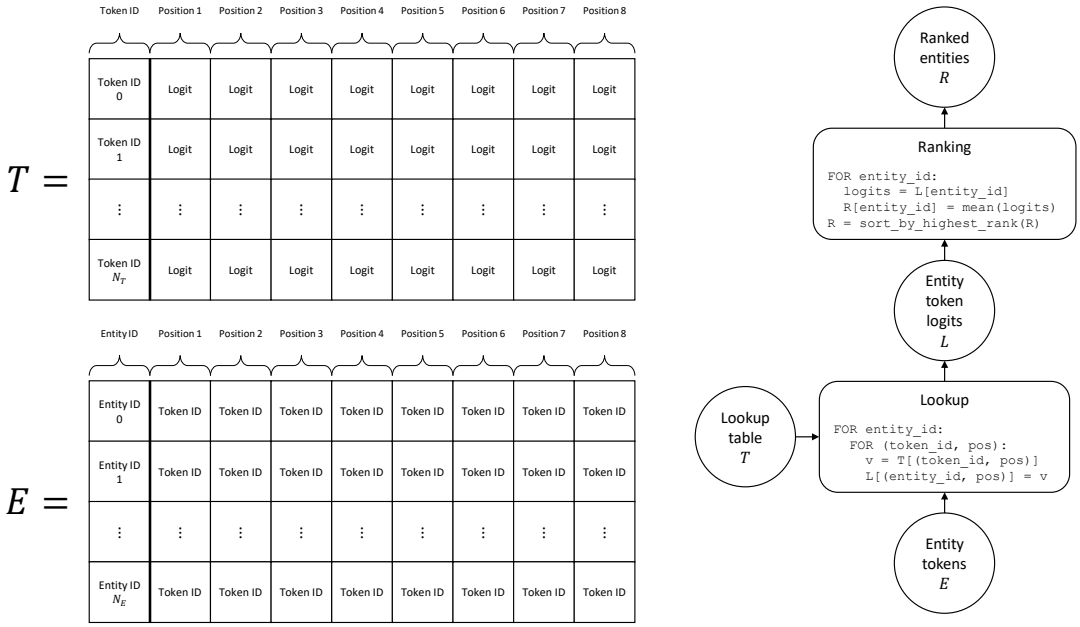

Figure 4: Ranking System. The figure details the inner workings of the ranking system which uses the lookup table generated by the masked language model to compute the score associated with each possible entity. The scored entities are then ranked by highest score.

the entity token logits L by matching all token ids in E with their corresponding values in T. L represents how likely every token of the entity is to be generated by the MLM at that specific position. The mean likelihood[1] of each entity is computed by averaging L over *non-padded* token logits.[2] This value is used to determine the ranking of the entity. It provides a proper comparison between entities of different lengths.

Concretely, in our previous "cat and dog _" example, we average the output logits for the "cat and dog" token ids and positions while ignoring the final padded logit. This averaging is done on all entities in the dataset completing the triple, yielding the average likelihood assigned by the model to all entities.

Entities are then sorted by highest rank using the randomized setting (Sun et al., 2020). For equal scores the tie-breaking is done randomly, to produce the ordered list of ranked entities R. We use the filtered setting (Bordes et al., 2013) for evaluation and remove corrupted triples from the list of ranked entities, corrupted triples being all other known correct triples.

[1]Because the length of non-padded tokens is variable, using the mean of the logits is the chosen comparison metric for re-ranking.

[2]By far, the token the model sees most is the padding token. Counting it would most likely yield a heavy skew towards shorter entities with more padding.

4 Experimentation

4.1 Datasets

The two datasets used are WN18RR and WikidataM5 (Dettmers et al., 2017; Fellbaum, 1998; Bollacker et al., 2008; Wang et al., 2019), a commonly used link prediction benchmark and a new, large scale, link prediction benchmark. Summary stats are shown in Table 1.

Table 1: Datasets statistics.

	WN18RR	WikidataM5
#Entities	40,943	4,594,585
#Relations	11	822
#Training	86,835	20,614,279
#Validation	3034	5163
#Test	3134	5133
Mean in-degree	2.12	1
Median in-degree	4.49	0

WN18RR is a dataset composed of WordNet synsets. We use the cleaned synset as the entity string. The synset "dog.n.01" would have a string representation of "dog noun 1" which should be more interpretable by the model while remaining a unique identifier. The entity definition is the definition of the entity given by WordNet. The relation string is a cleaned representation of the relation

string. The relation "_member_of_domain_usage" would be represented with the string "member of domain usage". Full examples of inputs and outputs are shown in Listing 1 and Listing 2.

WikidataM5 is composed of triples based on Wikidata and the English Wikipedia with aligned descriptions for each entity. We use the entity string and definitions provided in the benchmark.

4.2 Metrics

We use the Mean Reciprocal Rank (MRR) metric to validate our model and select the best model. For all experiments, we also report the Mean Rank (MR), the Mean Precision at 1 (MP@1), the Mean Precision at 3 (MP@3), and the Mean Precision at 10 (MP@10).

4.3 Training

The training setup is a modified MLM training, where we let the model generate the missing entity. The previously mentioned padding lets us deal with the generation of entities of varying sizes. The input fed to the model for tail entity prediction, depicted in Figure 3, consists of the concatenated token ids of the head entity, the head entity definition, the relation and the tail entity mask. The model will then generate, in the place of the mask, the missing entity. The input fed to the model for head entity prediction is similar. An example of the input for head entity prediction is found in Listing 1 and an example for tail entity prediction is found in Listing 2.

We use the categorical cross-entropy loss to train the language model. The loss only depends on the non-padded token of the generated entity, ignoring all other outputs. The target is the actual entity completing the triple, aligned with the mask in the input. We retain the model with the best validation MRR. All experiments are run for 5 random seeds and the mean and standard deviation of the results are reported.

For all experiments, we use the hyperparameters and training setup described in Liu et al. (2019), with a total of 25 epochs for the WN18RR dataset and 1 epoch for the WikidataM5 dataset.

4.4 Unseen Entities

An alternative version of the dataset is made to test the generalization capacity of our methodology to unseen entities. For WN18RR we start by randomly sampling 5% of the entities for the validation entities and 5% of the entities for the testing entities. Our training set consists of all triples not containing any of the validation or testing entities. Our validation set consists of all triples containing the validation entities. Finally, our test set consists of all triples containing the test entities, but not containing any of the validation entities. The training is done in the same fashion. The validation and testing are only done on entities present in the validation or test entity list, but are still reranked against all other possible entities. While WN18RR would only have 2047 test entities in this setting, the reranking would still involve $40{,}943$ entities. If the tail entity is the one present in the test entity list, we will complete the link $(h, r, ?)$ and not the link $(?, r, t)$. The reported results are therefore only on the performance of previously unseen entities in the KB, compared to all other possible entities. The validation and test set are rebuilt for every random seed, to evaluate our approach on a wider array of unseen entities.

WikidataM5 has an inductive setting which closely resembles our unseen entities setting. The main difference is that the reranking step in the validation and test only compares with other previously unseen entities. While there is upwards of 4 million training entities, the validation and testing would only consider roughly 7000 entities. For the purpose of proper comparison, the M5 results are however done within their specific setting.

5 Results and Analysis

5.1 WN18RR

We achieve SotA results on the WN18RR dataset on all tested metrics with the exception of MR, as shown in Table 2. The WN18RR dataset is sparse in terms of the KG in-degree connections, see Table 1. Sparseness lends itself naturally to leveraging a pretrained model. The amount of information that can be extracted from the dataset on any given entity is then limited, which makes outside information all the more valuable.

We can observe that the MR metric is relatively much weaker for our model, compared to its other metrics. This implies that while the model will often rerank the correct entities to the top, it will also sometimes forget certain entities completely, given them ranks in the thousands. This could be explained by an issue of disambiguation in the name of the entity. While approaches using entity embeddings (Balažević et al., 2019; Sun et al., 2019; Dettmers et al., 2018) will have no issue separating

Table 2: WN18RR Results

Approach	MRR ↑	MR ↓	MP@1 ↑	MP@3 ↑	MP@10 ↑
ConvE	44.4	4950	—	—	50.3
RotatE	47.3	3343	—	—	57.1
TuckER	46.1	6324	—	—	51.6
ConvKB	24.9	3433	—	—	52.4
CapsE	41.5	718	—	—	55.9
KBAT	41.2	1921	—	—	55.4
KG-BERT	—	**97**	—	—	52.4
MLMLM	**50.17** ± **0.18**	1603 ± 26.8184	43.91 ± 0.20	54.18 ± 0.28	**61.10** ± **0.20**

The results are reported as <mean> ± <standard deviation>. Results for other models are taken from Sun et al. (2020); Yao et al. (2019).

Listing 1: Example of an error of the model on WN18RR. Shown are the top 3 ranked entities by the model with the score assigned to them. The correct answer, `matchmaker noun 1`, was ranked 14,108 by the system.

```
Prompt : <s><mask><mask><mask><mask><mask><mask><mask><mask>hypernym mediator noun 1
     a negotiator who acts as a link between parties</s><pad><pad><pad><pad>
Correct answer : matchmaker noun 1<pad><pad><pad><pad>   Answer rank 14108
Rank 1   Score 32.0242   : interpreter noun 2<pad><pad><pad>
Rank 2   Score 32.0103   : harmonizer noun 1<pad><pad><pad>
Rank 3   Score 31.8889   : diplomat noun 1<pad><pad><pad>
```

Table 3: WikidataM5 Results

Approach	MRR ↑	MR ↓	MP@1 ↑	MP@3 ↑	MP@10 ↑
TransE	**25.3**	109370	17.0	**31.1**	**39.2**
DKRL	16.0	31566	12.0	18.1	22.9
KEPLER-Wiki	15.4	**14454**	0.105	0.174	0.244
KEPLER-Cond	21.0	20267	17.3	22.4	27.7
MLMLM	22.3	488161	**20.1**	23.2	26.4

A single seed was ran for the WikidataM5 experiments because of the size of the dataset. Results for other models are taken from Wang et al. (2019).

Table 4: WN18RR Unseen Entities Result

Approach	MRR ↑	MR ↓	MP@1 ↑	MP@3 ↑	MP@10 ↑
Random baseline	0.03 ± 0.007	20541.91 ± 87.88	0.002 ± 0.004	0.002 ± 0.004	0.026 ± 0.008
Non-finetuned RoBERTa	2.73 ± 0.05	10130.35 ± 187.61	1.54 ± 0.07	2.95 ± 0.11	4.92 ± 0.19
MLMLM	18.42 ± 2.66	3761.50 ± 255.4437	14.16 ± 0.81	21.75 ± 1.19	29.39 ± 0.88

The results are reported as <mean> ± <standard deviation>.

the synsets `dog.n.01` and `dog.n.03` as meaning respectively "a member of the genus Canis [...]" and "informal term for a man", our model will have to discern between those two meanings only by the digit appended to the name. It is probable that the model is often confused about whether it should generate `dog noun 1` or `dog noun 3`, having only the final digit to differentiate both of

Listing 2: Example of a disambiguation error of the model on WN18RR. Shown are the top 3 ranked entities by the model with the score assigned to them. The correct answer, `aid noun 3`, was ranked second by the system, after `aid noun 1`.

```
Prompt : <s>grant noun 1 any monetary aid hypernym<mask><mask><mask><mask><mask><
    mask><mask><mask></s><pad><pad><pad><pad>
Correct answer : aid noun 3<pad><pad><pad><pad><pad>     Answer rank 2
Rank 1   Score 33.7597   : aid noun 1<pad><pad><pad><pad><pad>
Rank 2   Score 33.5948   : aid noun 3<pad><pad><pad><pad><pad>
Rank 3   Score 32.7605   : aid noun 2<pad><pad><pad><pad><pad>
```

Table 5: WikidataM5 Inductive setting Results

Approach	MRR ↑	MR ↓	MP@1 ↑	MP@3 ↑	MP@10 ↑
DKRL	23.1	78	5.9	32.0	54.6
KEPLER-Cond	**40.2**	**28**	22.2	**51.4**	**73.0**
MLMLM	28.4	932	**22.6**	28.5	34.8

The results are reported as <mean> ± <standard deviation>.

them. An example of such an error is shown in Listing 2, where the model confuses `aid.n.01` and `aid.n.03`. Follow up work on better representations for entity names could yield stronger results.

Our model generally has a much easier time predicting the tail entity than the head entity. It has an MRR of 60.15 on tail entities and an MRR of 40.09 on head entities. By observing the instances where our model gives the worst rank to the correct answer, we can understand why. A large number of those cases are hypernyms on the head entity. An example of a hypernym relationship would be: "animal is an hypernym of dog, since all dogs are animals." Correctly ranking all possibilities for "X is an hypernym of dog." is more straightforward for the model than correctly ranking all possibilities for "Animal is an hypernym of Y.". An example of such failure is shown in Listing 1, where we look for the hypernym of the term `mediator`. It is clear that the model understands the concept and outputs plausible answers in its top 3. A large amount of the model's severe failure cases are similar to this one, where the model will output a plausible hypernym of the tail entity, while completely missing the targeted hypernym. This seems to be the likely cause for the weak MR of the approach.

5.2 WikidataM5

The results on WikidataM5 are shown in Table 3. MLMLM boasts the best Precision@1 metric by a fair margin. Once again, we observe the weakness in the MR metric. The implications are that there are many possible correct entities that are given no weights by the approach.

5.3 Unknown Entities Experiments

We demonstrate the capacity of our approach to generalize to unknown entities. Results for the WN18RR datasets are shown in Table 4.

For baselines, we use a random baseline, reranking the entities randomly, as well as a non-finetuned RoBERTa-large model, that simply generates the entity tokens without being finetuned on the dataset first. We can notice that while our approach outperforms a non-finetuned benchmark, the non-finetuned RoBERTa model still far outperforms the random baseline, supporting some of the findings of Petroni et al. (2019) in the capacity of MLM to perform unsupervised link prediction.

The M5 inductive settings are reported in Table 5. We obtain the best Precision@1 metric. The weakness in MR is once again visible, supporting the intuition that while the model might generate the correct entity with high conviction, it will often not give positive score to all plausible entities, yielding a much worst average rank.

We believe that leveraging MLMs could eventually lead to automatically populating KBs with new entities, as new knowledge and new facts are created and added to the web.

5.4 Limitations

MLMLM comes with several limitations. Our approach to padding limits the size of an unknown

entity to the size of the longest known entity. While it is unlikely to be limiting in practice, it is still a weakness of our approach to sampling. The model size can be very prohibitive and specialized hardware such as GPUs is required to run it in a timely fashion. The approach however remains tractable as it can provide likelihoods for all possible entities in a single inference call. Compared to entity-embedding based methods, our approach needs additional information in the form of meaningful string representations for both entities and relations. The lack of entity disambiguation is also a limiting factor that does not affect other approaches. Finally, our approach is liable to *forgetting* some entities, leading to comparatively much worst MR than prior approaches.

6 Conclusion

We have developed a methodology for training masked language models to perform link prediction. By leveraging the natural language understanding abilities of these models and the factual knowledge embedded within their weights, we have achieved a tractable approach to link prediction that yields state of the art results on a standard benchmark and the best Precision@1 on another competitive benchmark. We have also demonstrated the ability of our model to perform link prediction of previously unseen entities, making our approach suitable to introduce new entities to knowledge bases. More generally, we have introduced an approach to sampling text from a masked language model of varying lengths, which can have a wider use case.

Acknowledgements

This research was supported by Apogée Canada, Canada First Research Excellence Fund program and École Polytechnique Startup Fund PIED. SC is supported by a Canada CIFAR AI Chair and an NSERC Discovery Grant.

References

Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

D. Daza, Michael Cochez, and Paul T. Groth. 2020. Inductive entity representations from text via link prediction. *ArXiv*, abs/2010.03496.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2017. Convolutional 2d knowledge graph embeddings. In *AAAI*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Christiane Fellbaum, editor. 1998. *WordNet: an electronic lexical database*. MIT Press.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2681–2690, Copenhagen, Denmark. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Z. Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, F. Petroni, V. Karpukhin, Naman Goyal, Heinrich Kuttler, M. Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *ArXiv*, abs/2005.11401.

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.

Deepak Nathani, Jatin Chauhan, Charu Sharma, and Manohar Kaul. 2019. Learning attention-based embeddings for relation prediction in knowledge graphs. *arXiv preprint arXiv:1906.01195*.

Dai Quoc Nguyen, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2017. A novel embedding model for knowledge base completion based on convolutional neural network. *arXiv preprint arXiv:1712.02121*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *EMNLP/IJCNLP*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Julian Salazar, Davis Liang, Toan Q. Nguyen, and K. Kirchhoff. 2020. Masked language model scoring. In *ACL*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Zhiqing Sun, Shikhar Vashishth, Soumya Sanyal, Partha Talukdar, and Yiming Yang. 2020. A Re-evaluation of Knowledge Graph Completion Methods. *arXiv e-prints, accepted at ACL 2020*, page arXiv:1911.03903.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *CoRR*, abs/1706.03762.

Thanh Vu, Tu Dinh Nguyen, Dat Quoc Nguyen, Dinh Phung, et al. 2019. A capsule network-based embedding model for knowledge graph completion and search personalization. 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 2180–2189.

Alex Wang and Kyunghyun Cho. 2019. BERT has a mouth, and it must speak: BERT as a markov random field language model. *CoRR*, abs/1902.04094.

Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhiyuan Liu, Juanzi Li, and Jian Tang. 2019. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *CoRR*, abs/1911.06136.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Liang Yao, Chengsheng Mao, and Yuan Luo. 2019. Kg-bert: Bert for knowledge graph completion. *ArXiv*, abs/1909.03193.