Language Models as Knowledge Bases: On Entity Representations, Storage Capacity, and Paraphrased Queries

Benjamin Heinzerling\textsuperscript{1,2} and Kentaro Inui\textsuperscript{2,1}

\textsuperscript{1}RIKEN AIP \& \textsuperscript{2}Tohoku University

benjamin.heinzerling@riken.jp \mid inui@tohoku.ac.jp

Abstract

Pretrained language models have been suggested as a possible alternative or complement to structured knowledge bases. However, this emerging LM-as-KB paradigm has so far only been considered in a very limited setting, which only allows handling 21k entities whose single-token name is found in common LM vocabularies. Furthermore, the main benefit of this paradigm, namely querying the KB using a variety of natural language paraphrases, is underexplored so far. Here, we formulate two basic requirements for treating LMs as KBs: (i) the ability to store a large number of facts involving a large number of entities and (ii) the ability to query stored facts. We explore three entity representations that allow LMs to represent millions of entities and present a detailed case study on paraphrased querying of world knowledge in LMs, thereby providing a proof-of-concept that language models can indeed serve as knowledge bases.

1 Introduction

Language models (LMs) appear to memorize world knowledge facts during training. For example, BERT (Devlin et al., 2019) correctly answers the query “Paris is the capital of [MASK]” with “France”. This observation prompted Petroni et al. (2019) to ask if LMs can serve as an alternative or complement to structured knowledge bases (KBs), thereby introducing the idea of treating LMs as KBs: During training, the LM encounters world knowledge facts expressed in its training data, some of which a stored in some form in the LM’s parameters. After training, some of the stored facts can be recovered from the LM’s parameters by means of a suitable natural language query (Fig. 1). However, this emerging LM-as-KB paradigm is faced with several foundational questions.

First question: KBs contain millions of entities, while the vocabulary size of common LMs usually does not exceed 100k entries. \textbf{How can millions of entities be represented in LMs?} Previous work (Petroni et al., 2019) circumvents this problem by only considering the roughly 21k entities whose canonical name corresponds to a single token in the LM vocabulary, e.g., entities like “France” or “Bert”, but not “United Kingdom” or “Sesame Street”. Hence, this approach cannot handle entities that are not contained in the LM’s vocabulary, and a query like “Bert is a character on [MASK]” is not answerable in this simplified setting.

To answer this first question, we compare three methods for scaling LM-as-KB to millions of entities:

1. Symbolic representation, i.e., extending the LM vocabulary with entries for all entities;
2. Surface form representation, i.e., each entity is represented by their subword-encoded canonical name, which is stored and queried by extending the LM with a sequence decoder for entity names; and
3. Continuous representation, i.e., each entity is represented as an embedding.

We find that, while all three entity representations allow LMs to store millions of world-knowledge
facts involving a large number of entities, each representation comes with different trade-offs: Symbolic representation allows the most accurate storage, but is computationally expensive and requires entity-linked training data. Surface representation is computationally efficient and does not require entity-linked training data, but is less accurate, especially for longer entity names. Continuous representation also requires entity-linked training data, but is computationally more efficient than symbolic representation.

Second question: What is the capacity of LMs for storing world knowledge? Can a LM store, say, all relation triples contained in a knowledge base like Wikidata (Vrandečić and Krötzsch, 2014)? Here we conduct experiments using synthetic data to study the scaling behaviour of current LM architectures. Varying the number of trainable model parameters and recording the number of relation triples memorized at a given accuracy level, we find that, e.g., a Transformer (Vaswani et al., 2017) with 125 million parameters (12 layers of size 768), has the capacity to memorize 1 million Wikidata relation triples with 95 percent accuracy or 5 million relation triples with 79 percent accuracy. Assuming linear scaling, this finding suggests that larger LMs with tens or hundreds of billions of parameters (Raffel et al., 2019; Brown et al., 2020) can be used to store sizable portions, if not all, of a large knowledge base like Wikidata.

Third question: How robustly is world knowledge stored in LMs? Is the LM able to recall a fact even if the query is slightly different than what was memorized during training? For example, if the LM memorized “Barack Obama was born in Hawaii” during training, can it answer queries like “Barack Obama is from [MASK]” or “Where was Barack Obama born? [MASK]”? Here we conduct controlled experiments to measure how well the LM transfers knowledge from memorized statements to query variants, both in a zero-shot setting in which the model is not exposed to the target query variant during training, and a few shot setting, in which the model is finetuned on a small number of statements containing the target query variant. We observe zero-shot transfer in case of highly similar query variants, and see successful few-shot transfer after finetuning with 5 to 100 instances in case of less-similar queries. This ability to handle soft, natural language queries, as opposed to hard, symbolic queries in a language like SQL or SPARQL, is one of the key motivations for using language models as knowledge bases.

Contributions. We formulate two requirements for treating LMs as KBs: (i) the ability to store a large number of facts involving a large number of entities and (ii) the ability to query stored facts. After providing background on world knowledge in language models (§2), we make the following contributions:

- A comparison of entity representations for scaling LM-as-KB to millions of entities (§3);
- Empirical lower bounds on LM capacity for storing world knowledge facts (§4); and
- A controlled study of zero-shot and few-shot knowledge transfer from memorized statements to paraphrased queries (§5).

Terminology. In this work we are interested in storing and retrieving world knowledge facts in and from a language model. World knowledge is knowledge pertaining to entities, such as Barack Obama. A fact is a piece of world knowledge that can be expressed with a concise natural language statement, such as the English sentence Barack Obama was Born in Hawaii, or with a relation triple, such as ⟨Barack Obama, wasBornIn, Hawaii⟩. A relation triple, or relation for short, consists of a head or subject entity (Barack Obama), a predicate (wasBornIn), and a tail or object entity (Hawaii). A knowledge base is a set of relations. Knowledge bases, such as Wikidata, typically contain hundreds or thousands of predicates, millions of entities, and millions or billions of relations.

2 World Knowledge in Language Models

Large pretrained LMs have been the main driver of recent progress in natural language processing (Peters et al., 2018; Howard and Ruder, 2018; Radford et al., 2019; Devlin et al., 2019). While the trend towards larger LMs is likely to continue (Raffel et al., 2019; Kaplan et al., 2020; Brown et al., 2020), it has fundamental limitations: (i) A model trained only on surface forms, i.e., text, lacks grounding in perception and experience and hence cannot learn meaning (Bender and Koller, 2020). (ii) Reporting bias leads to certain knowledge rarely or never being expressed in text. For example, a LM will easily learn that Barack Obama is a former U.S. President, but will less likely learn that he is a male human being, since the latter fact is rarely stated
| Paradigm / Task                  | Input                           | Output                          | Models and objectives                                                                 |
|---------------------------------|---------------------------------|---------------------------------|---------------------------------------------------------------------------------------|
| Language modeling               | Text                            | Text                            | Next word prediction (Shannon, 1948; Elman, 1990; Bengio et al., 2003), masked token prediction (Devlin et al., 2019) |
| LM-as-KB?                       | Text                            | Text / single-token entity name  | Closed-book QA (LAMA probe, Petroni et al., 2019)                                      |
| Sequence-to-sequence            | Text                            | Text                            | Text-to-text transformer (T5, Raffel et al., 2019), closed-book QA (Roberts et al., 2020) |
| Retrieval                       | Text                            | Text, answer span               | Answer-span selection (Chen et al., 2017), retrieval-augmented LM (Guu et al., 2020), open-book QA |
| Entity replacement              | Text, entity mention spans      | Text                            | Detecting replaced entity mentions (Xiong et al., 2019)                                |
| Entity linking (EL)             | Text, entity mention spans      | Target entity                   | AIDA (Hoffart et al., 2011), neural EL (Francis-Landau et al., 2016; Kolitsas et al., 2018) |
| Entity embeddings               | Text, entity mention spans      | Entity embeddings               | Joint embedding of entities and text (Yamada et al., 2016)                              |
| LM with entity embeddings       | Text, linked entity mentions, entity embeddings | Text                            | ERNIE (Zhang et al., 2019), E-BERT (Poerner et al., 2019)                              |
| LM with integrated EL           | Text, entity embeddings         | Text                            | KnowBert (Peters et al., 2019)                                                        |
| LM-as-KB (this work)            | Natural language query          | Target entity                   | Fact memorization, paraphrased queries, closed-book QA                                 |
| Knowledge-aware LM              | Text, knowledge (sub)graph query | Target entity, text             | Neural Knowledge LM (Ahn et al., 2016), Reference-aware LM (Yang et al., 2017), Knowledge graph LM (Logan et al., 2019) SEMPRE (Berant et al., 2013), GNNs for KBQA (Sorokin and Gurevych, 2018) |
| Semantic parsing                | natural language query          | meaning representation, target entity tuple and relation embeddings | Matrix factorization (Riedel et al., 2013)                                               |
| Universal Schema                | relation triples, text patterns | node and edge embeddings        | Link prediction; RESCAL (Nickel et al., 2011), TransE (Bordes et al., 2013), ComplexE (Trouillon et al., 2016), ConvE (Dettmers et al., 2018) |
| Knowledge graph embeddings      | relation triples, text patterns | node and edge embeddings        | DeepWalk (Perozzi et al., 2014), graph neural networks (Kipf and Welling, 2017)           |
| Graph neural networks           | nodes, node features, edges     | node embeddings                 | Storage and retrieval, SQL/SPARQL queries, symbolic reasoning (Coppens et al., 2013)     |
| Knowledge graphs                | nodes, edges                    | nodes, edges                    |                                                                                         |

Table 1: Approaches for using world knowledge in natural language processing, ranging from unstructured, purely text-based approaches (top), over approaches that mix text and structured KBs to varying degrees (middle), to approaches operating on structured KBs (bottom).

explicitly in text. In contrast, this type of knowledge is readily available in knowledge bases. (iii) A large number of rare entities (Hoffart et al., 2014; Derczynski et al., 2017; Ilievski et al., 2018) are, by definition, rarely mentioned, making it difficult for LMs to acquire knowledge about this long tail of entities from text alone.

These limitations have motivated efforts to explicitly¹ equip LMs with world knowledge. Table 1 situates these efforts on a spectrum from purely text-based language modeling to representations of structured graphs. Models based on text generation (Raffel et al., 2019; Roberts et al., 2020) and retrieval (Guu et al., 2020) (denoted with Text in the Output column) have proven most successful in knowledge-intensive tasks. However, we argue that models which reify entities (Logan et al., 2019), i.e., models in which entities are “first-class citizens” that can be directly predicted² (denoted by

¹As opposed to the LM acquiring world knowledge implicitly as a side effect of its training objective.
²As opposed to generating or retrieving a surface form Target entity in the Output column), are a promising research direction, since the direct links into a KB can be seen as a form of grounding. This is one of our main motivations for considering symbolic and continuous entity representations.

3 Entity Representations

We now address our first question: How can millions of entities be represented in a LM? To answer this question, we compare three types of entity representations, namely symbolic, surface form, and continuous representation.

Experimental setup. We evaluate entity representations by measuring how well they allow a LM to store and retrieve world knowledge facts. For example, if the model’s training data contains the statement “Bert is a character on Sesame Street”, the model should be able to memorize this statement and recall the correct object Sesame Street when queried with a query like “Bert is a character on [MASK].” which may or may not correspond to an entity.
**Synthetic data.** It is not a priori clear how many facts a given text from the LM’s training data, say, a Wikipedia article, expresses. Since we want to precisely measure how well a LM can store and retrieve facts, we create synthetic data by generating statements from relation triples and then train the model to memorize these statements in an idealized setting. Using Wikidata as knowledge source, we first define two sets of entities: A smaller set consisting of the top 1 million Wikidata entities according to node outdegree, and a larger set consisting of the roughly 6 million Wikidata entities that have a corresponding entry in the English edition of Wikipedia.

Next, we select the 100 most frequent Wikidata predicates and manually create one statement template for each predicate. For example, for the Wikidata predicate P19 (“place of birth”), we create the template \( S \) was born in \( O \) and generate English statements by filling the \( S \) and \( O \) slots with entities from the sets defined above for which this relation holds.\(^3\) To make queries for an object entity unique given subject and predicate, we arbitrarily select exactly one fact if there are multiple possible objects and discard the other facts. This process yields 5 million statements involving up to 1 million entities, and 10 million statements involving up to 6 million entities. These statements then serve as training instances, i.e., given the query “Barack Obama was born in \([\text{MASK}]\)”, the model should predict Hawaii. As our goal is to store facts in a LM, there is no distinction between training and test data.

**Models and training.** We consider two common LM architectures: LSTMs (Hochreiter and Schmidhuber, 1997) and Transformers (Vaswani et al., 2017). For LSTMs, we compare two model configurations, namely a randomly initialized two-layer LSTM with 256 hidden units per layer (\( \text{LSTM} \ 256 \)) and one with 1024 hidden units per layer (\( \text{LSTM} \ 1024 \)). For Transformers, we compare a pretrained model, namely RoBERTa-base (Liu et al., 2019), and RoBERTa without pretraining, i.e., a randomly initialized Transformer of the same size. For consistent tokenization across all four models, we subword-tokenize all statements with the RoBERTa tokenizer. To store statements with symbolic and continuous representation, we train until the model reaches 99 percent memorization accuracy, i.e., achieves almost perfect overfitting, or stop early if accuracy does not improve for 20 epochs. Further training details are given in Appendix C.

### 3.1 Symbolic Representation

With symbolic representation, each entity is represented as an entry in the LM’s vocabulary. Prediction is done via masked language modeling (Devlin et al., 2019), by encoding the query with the LM, projecting the final hidden state of the \([\text{MASK}]\) token onto the vocabulary and then taking a Softmax over the vocabulary. As the results show (Fig. 2), symbolic representation yields very high memorization accuracies with a vocabulary of 1 million entities. RoBERTa-base without pretraining, i.e., a randomly-initialized Transformer, works best and memorizes 97 percent of 5 million statements correctly.

Unfortunately, the Softmax computation becomes prohibitively slow as the vocabulary size increases (Morin and Bengio, 2005), making symbolic representation with a Softmax over a vocabulary consisting of the full set of 6 million Wikipedia entities impractical. Imposing a hierarchy is a common approach for dealing with large vocabularies, but did not work well in this case (See Appendix E.1).

### 3.2 Surface Form Representation

With surface form representation, each entity is represented by its canonical name.\(^4\) Since this name generally consists of more than one token, we cast memorizing statements and querying facts as a sequence-to-sequence task (Sutskever et al., 2014): Given the source sequence “Bert is a character on \([\text{MASK}]\)”, the model needs to generate the

---

\(^3\) Templates and a sample of the generated statements are shown in Appendices A and B.

\(^4\) We use English Wikidata labels as canonical names.
target sequence “Sesame Street”. To make models memorize statements, we train until perplexity on the training data reaches 1.0 or does not improve for 20 epochs. For evaluation, we generate surface forms of target entities – i.e., the answer to a given query – via a beam search with beam size 10. We measure perfect-match accuracy of the full entity name, i.e., there is no partial credit for partial token matches.

The four models in our comparison are now treated as sequence-to-sequence encoders and extended with a matching decoder of the same size, i.e., LSTM decoders for LSTM encoders (LSTM2LSTM) and randomly initialized Transformers for Transformer encoders (RoBERTa2Transformer and Transformer2Transformer).

Unlike symbolic representation, surface representation is able to handle the entire set of 6 million Wikipedia entities. As with symbolic representation, the randomly initialized Transformer model (Fig. 3, dash-dotted red line) has the highest capacity, memorizing up to 10 million statements with 90 percent accuracy. A pretrained LM as encoder (RoBERTa2Transformer) appears to have a deleterious effect, with much lower accuracies compared to the randomly initialized Transformer2Transformer. While the larger LSTM2LSTM model (1024 hidden units per layer) almost matches the performance of the best Transformer model, the smaller LSTM2LSTM (256 hidden units per layer) has insufficient capacity, memorizing less than 50 percent of 5 million statements correctly.

An analysis of the results produced by the Transformer2Transformer model (Fig. 4) reveals, perhaps unsurprisingly, that statements involving infrequent and long entity mentions are most difficult to memorize. For example, the model fails to memorize most of the entity mentions that occur only in one to ten statements and have a length of 12 or more subword tokens (blue cluster, upper left).

### 3.3 Continuous Representation

With continuous representation, each entity $e_i, i \in [1, N_{entities}]$ is represented by a d-dimensional embedding $y_i \in \mathbb{R}^d$. After encoding the given query with the LM, prediction is performed by projecting the final hidden state corresponding to the [MASK] token onto $\mathbb{R}^d$, thereby obtaining the predicted embedding $\hat{y} \in \mathbb{R}^d$. We use fixed, pretrained entity embeddings and train with cosine loss

$$L = 1 - \cos(\hat{y}, y_i)$$

At test time, the model prediction $\hat{y}$ is mapped to the closest pretrained entity embedding $y_i$ via approximate nearest-neighbor search (Johnson et al., 2017).

**Continuous prediction with fixed, pretrained embeddings.** When training randomly initialized embeddings with a cosine similarity or Euclidean distance objective, a degenerate solution is to make all embeddings the same, e.g., all-zero vectors. To prevent this, it is common practice to use negative samples (Bordes et al., 2013). When training with fixed, pretrained embeddings as supervision signal, negative sampling is not necessary, since the target embeddings are not updated during training and
therefore cannot become degenerate.

**Wikidata embeddings.** We train embeddings for 6 million Wikidata entities using feature-specific autoencoders to encode entity features such as names, aliases, description, entity types, and numeric attributes. This approach follows prior work on multimodal KB embeddings (Pezeshkpour et al., 2018) and learning of KB embeddings with autoencoders (Takahashi et al., 2018). Embedding training is detailed in Appendix D.

**Results.** Fig. 5 shows memorization accuracies achieved with continuous representation. Like surface representation, continuous representation scales to 6 million entities, and we see the same relative order of models, but with overall lower accuracies. RoBERTa without pretraining has the highest capacity for storing world knowledge statements, memorizing 67 percent of 10 million statements, while the small LSTM 256 model has the lowest capacity, memorizing 42 percent. Although far from fully understood, sequence-to-sequence architectures are relatively mature, with highly-optimized toolkits and hyperparameter settings publicly available (Ott et al., 2019). In contrast, prediction of continuous representations is still in an early stage of research (Kumar and Tsvetkov, 2019). Compared to surface form representation, we therefore see the results presented in this subsection as lower bounds for LM capacity with continuous representations.

By design, memorization with continuous representations does not rely on entity names, and hence, in contrast to surface form representation, does not lead to difficulties in handling entities with long names. However, as with surface form representation, infrequent entities are more difficult to memorize than frequent ones. As shown in Fig. 6, most of the memorization errors (blue, left) involve infrequent entities with a median frequency of 3, while most of the correctly memorized statements (orange, right) involve entities that occur more than 100 times.

4 LM Capacity for Storing Facts

We now turn to the question of how model capacity scales with model size (Figure 7). For a 12-layer transformer with layer size 96 or 192 (top subfigure, solid red and dashed green lines), memorization accuracy quickly drops as the number of facts to memorize increases. As expected, larger models are able to memorize more facts, but accuracy drops rather quickly, e.g., to 65 percent of 3 million facts memorized with a layer size of 384 (dotted orange line, 2nd from top).

Assuming a desired memorization accuracy level, e.g., 80 percent, we analyze the maximum number of facts a model of a given size can memorize at that level (Figure 7, bottom). For the model sizes considered here, storage capacity appears to scale linearly, with a model of layer size 384 (55M parameters) able to store one million facts, and a model of layer size 960 (160M parameters) storing up to 7 million facts.

Apart from the number of facts to be stored, we hypothesize that storage capacity depends on two more factors: the number of entities involved and the entropy of their distribution. As expected, a large entity vocabulary makes memorization more difficult (Table 2). The impact of entity vocabulary size is smaller with surface representation (2 percent drop), while for continuous representation, memorization accuracy drops from 85 percent to 79 percent as the vocabulary size increases from 1 to 6 million entities. We also observe an impact of the entity distribution, with an example given in
Figure 7: Scaling of model capacity with model size. The model is a 12-layer Transformer with continuous representation of 6 million entities. The top figure shows the decrease in memorization accuracy as the number of facts to be stored in a model of given size increase. The bottom figure shows the maximum number of facts a model of a given layers size (and parameter count) can memorize with an accuracy of 80 percent.

Appendix. F, but leave a more detailed analysis to future work.

5 Querying Stored Facts

So far, we saw that it is possible to store millions of facts in a LM, by finetuning the model to predict the masked object of simple English statements like *Barack Obama was born in [MASK]*. However, given the large number of model parameters and the effort necessary to train them, mere storage is not a compelling achievement: The underlying relation triples, in this case *⟨Barack Obama, wasBornIn, Hawaii⟩*, can easily be stored more compactly and with 100 percent accuracy in a symbolic knowledge graph.

One of the potential benefits of the LM-as-KB paradigm is the LM’s ability to handle paraphrases. If the LM’s representation of the statement above is sufficiently similar to its representation of queries like *Barack Obama is from [MASK]* or even *Where is Barack Obama from? [MASK]*, it is conceivable that this similarity allows transfer from the memorized statement to these unseen queries. Is this soft querying of facts stored in a LM possible? In this section we conduct a controlled experiment to answer this question, expecting one of the following three outcomes:

1. **Rote memorization.** The model memorizes statements with little or no abstraction, so that even small, meaning-preserving changes to the query prevent the model from recalling the correct object.
2. **Generic association.** The model memorizes pairs of subject and object entities with little or no consideration of the predicate. For example, the model will always predict *Hawaii* whenever the query contains the phrase *Barack Obama*, regardless of context. This pathological behaviour could be especially prevalent if the distribution of object entities co-occurring with a particular subject is dominated by a single object.
3. **Fact memorization.** The model memorizes facts expressed in statements by forming abstractions corresponding to entities and predicates. This would allow retrieving a fact with a variety of queries.

The results presented in previous sections already established that a model of sufficient size is able to perform rote memorization of millions of statements. We now design an experiment to test whether LMs are capable of fact memorization while taking care to distinguish this capability from generic association.

To repeat, our goal is to test if a LM that has memorized a statement like *Barack Obama was born in Hawaii*. can transfer this knowledge to answer a query like *Barack Obama is from [MASK]*. Conveniently, *wasBornIn* relations are among the most frequent in Wikidata and hold for a diverse set of subject and object entities. This diversity of entities makes this predicate a good candidate for our case study, since statements involving a predicate with a less diverse set of possible subject or object entities are easier to memorize.\(^7\)

\(^7\)For example, with the predicate *isA* and relations like

| Representation  | Accuracy |
|-----------------|----------|
| Symbol          | 0.97     |
| Surface         | 0.92     |
| Continuous      | 0.85     |

Table 2: Impact of entity vocabulary size on model capacity. The model is a 12-layer Transformer, hidden layers size 768, memorizing 1 million facts.
Statements and controls. We randomly sample 100k statements generated by the “S was born in O” template. Since the mapping from S (i.e., mentions of persons) to O (i.e., locations) is injective, the model could take the shortcut of memorizing statements via generic association and wrongly answer any query involving entity S, e.g., “Barack Obama is a [MASK]”, with the associated entity O, i.e., Hawai. To prevent this shortcut, we introduce control facts. Given a fact ⟨S, P, O⟩, its control ⟨S, P’, O’⟩ involves the same subject S, but a distinct predicate P’ and object O’. For example, a control for the fact ⟨Albert Einstein, wasBornIn, Ulm⟩ is the fact ⟨Albert Einstein, diedIn, Princeton⟩. We add 100k control statements generated from the template “S died in O” and train RoBERTa-base to memorize all 200k statements with 99 percent accuracy. The combination of statements and control statements prevents the model from relying on generic association: To correctly answer the query “Albert Einstein died in [MASK]”, the model needs to take into account the predicate, since two distinct object entities are associated with Albert Einstein.

Target query variants. Next, we collect target query variants, such as “S is from O” (row labels in Fig. 8). Expecting good transfer for variants that are very similar to the original statement template, we include variants with small changes, such as varying punctuation or prepositions. To include more diverse variants, we select frequent relation patterns, e.g., “S (b. 1970, O)” and “born in O, S is a”, from the “place of birth” and “place of death” portions of the Google-RE corpus, as well as a query in question form. Finally we add irrelevant distractors (“It is true that, S was born in O”) and misleading ones (“S was born in O, but died somewhere else”). From each query variant template, we generate 100k query variants using the same entity pairs to fill the S and O slots as for the original statements. To balance the distribution between statements and control statements when finetuning towards target queries (see next paragraph), we also create a matching number of query variant templates and generate a matching number of control statements.

Transfer results. We evaluate knowledge transfer from memorized statements to query variants using pretrained RoBERTa-base (Fig. 8, left), measuring accuracy over the 100k statements generated with the target query variant template. To measure the effect pretraining has on paraphrasing ability, we compare to RoBERTa-base without pretraining (Fig. 8, right). We consider zero-shot transfer, i.e., without any finetuning towards the target query variant, and a finetuning setting, in which the LM is first trained to memorize all 100k original statements, and then finetuned until it memorizes all statements in the target query format.

In the zero-shot setting (leftmost column), even small variations to the query lead to a drop in fact recall: Adding an ellipsis (4th row) causes the model to answer 95% of queries correctly, a 3% drop from the 98% memorization accuracy of the original statements (first row). Adding an exclamation mark (5th row) has an even larger effect, resulting in a 8% drop. For two paraphrases, namely the relative clausal S, who is from O (7th row) and S is from O, zero-shot transfer works only in about 35% and 20% of cases. The question format (11th row) has an even larger effect, resulting in a 3% drop.

A clear overall trend is visible: Zero-transfer works best for similar statements and worst for dissimilar ones. To quantify this trend, we compute a representation of a statement template by averaging over its 100k mean-pooled, LM-encoded statements, and then measure the Euclidean distance of the original template representation and target query variant representations. Correlating Euclidean distance and accuracy of zero-shot transfer, we obtain a Pearson coefficient of −0.68, indicating a strong negative correlation between distance and knowledge transfer. In other words, transfer tends to work well for paraphrased queries the LMs deems similar to the originally memorized statement. Conversely, transfer fails in case the LM’s representation of a query is too dissimilar to its representation of the original statement.

This trend is also reflected in the finetuning setting, with less-similar variants requiring up to 500 instances until the model achieves 90 percent ac-
curacy (last row), while for more similar variants transfer already works well after finetuning on 5 to 50 target instances.

When using RoBERTa without pretraining to memorize statements, knowledge transfer to query variants is much worse. While transfer still works for the most similar variants (right, top rows), less similar variants require more finetuning instances compared to pretrained RoBERTa (right, middle rows). Transfer does not work for some of the least similar variants, with accuracies as low as 1 to 4 percent even after finetuning with 500 instances (right, bottom rows). Similar results for control statements are presented in Figure 9. We take these results a evidence that pretraining gives LMs the ability to handle paraphrased queries well, and that

| Query variants | accuracy (last row) |
|----------------|---------------------|
| S died in O    | 0.98                |
| S died in O, but they were not born there | 0.96                |
| S died at O    | 0.97                |
| S died in O, but they were born somewhere else | 0.91                |
| S died in O    | 0.88                |
| S, who died in O | 0.80                |
| S died in O, but they were not born there | 0.16                |
| S died at O    | 0.15                |
| It is true that S died in O | 0.19                |
| According to Wikidata, S died in O | 0.06                |
| Where did S die? | 0.01                |
| After their death in O, S was | 0.01                |
| S spent the last days of their life in O | 0.02                |
| S spent the last days of their life in O | 0.01                |

Figure 8: Transfer from memorized statements ($S$ was born in O) to query variants.

| Number of facts for finetuning to query variant |
|-------------------------------------------|
| 0 5 10 25 50 100 250 500 |

| Number of facts for finetuning to query variant |
|-------------------------------------------|
| 0 5 10 25 50 100 250 500 |

Figure 9: Transfer from memorized statements ($S$ died in O) to query variants.
LMs are able to memorize facts beyond mere rote memorization and generic association.

6 Discussion

Limitations. This work is not without limitations. We only consider one knowledge graph, Wikidata, in our experiments. Arguably, as the largest publicly available source of world knowledge, Wikidata is the most promising resource for equipping LMs with such knowledge, but attempts to store different knowledge graphs in a LM might result in different outcomes than those presented here. For example, certain types of graphs, such as randomly uniform graphs, are easier to memorize for a LM, than others, such as scale-free graphs (See Appendix. F).

While we use language like “train a LM to memorize statements” for simplicity throughout this work, what we do in case of pretrained LMs is more akin to adaptive pretraining (Gururangan et al., 2020). It is possible that integrating entity supervision directly into LM pretraining (Févry et al., 2020) allows more efficient fact storage.

Our analysis was entirely focused on entity representations and ignored the question how to represent relation predicates or entire relation triples. Here, incorporating relation learning (Baldini Soares et al., 2019) and learning to represent relation triples in a LM, e.g., from large, fact-aligned corpora (Elsahar et al., 2018), are exciting avenues for future work.

Finally, we formulated the LM-as-KB paradigm in terms of storing and retrieving relation triples. While structured KBs such as Wikidata indeed consist of such triples and hence our experiments showing storage and retrieval of triples LMs are sufficient as a proof-of-concept in principle, structured KBs also allow more complex queries than the ones considered here, such as 1-to-n relations, multihop inference, queries involving numerical ranges, or facts qualified by time and location (Hoffart et al., 2013).

Conclusions and outlook. In this work, we give a positive answer to Petroni et al. (2019)’s question if language models can serve as knowledge bases. We argued that treating LMs as KBs requires representing a large number of entities, storing a large number of facts, and the ability to query a given fact with a variety of queries. We then showed that current LM architectures fulfill these requirements when extended with a component for representing entities. In addition to the ability to handle paraphrased queries, we envision further benefits from the LM-as-KB paradigm. For example, the fact-memorization and paraphrase-finetuning setting introduced in Section 5 allows precise control over which facts a LM learns during training, while it is much less clear which facts are contained in unstructured text. Selecting paraphrases to increase the variety with which a LM can be queried is an interesting problem for future work. For example, selecting maximally dissimilar paraphrases and choosing the number of finetuning instances by similarity may be more efficient than finetuning on large numbers of paraphrases in brute-force fashion.

References

Sungjin Ahn, Heeyoul Choi, Tanel Prnamaa, and Yoshua Bengio. 2016. A neural knowledge language model.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the blanks: Distributional similarity for relation learning. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 2895–2905, Florence, Italy. Association for Computational Linguistics.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In Proceedings of ACL 2020 (to appear).

Joshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. 2003. A neural probabilistic language model. Journal of machine learning research, 3(Feb):1137–1155.

Jon Louis Bentley. 1975. Multidimensional binary search trees used for associative searching. Communications of the ACM, 18(9):509–517.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Durán, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2, NIPS’13, page 27872795, Red Hook, NY, USA. 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.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Sam Coppens, Miel Vander Sande, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. 2013. Reasoning over sparl. In LDOW.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In Proceedings of the 3rd Workshop on Noisy User-generated Text, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Tim Dettmers, Minervini Pasquale, Stenertop Pontus, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In Proceedings of the 32th AAAI Conference on Artificial Intelligence, pages 1811–1818.

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, Minneapolis, Minnesota. Association for Computational Linguistics.

Jeffrey L. Elman. 1990. Finding structure in time. Cognitive Science, 14(2):179–211.

Hady Elsayar, Pavlos Vougiouklis, Arsen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REX: A large scale alignment of natural language with knowledge base triples. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018), Miyazaki, Japan. European Language Resources Association (ELRA).

Thibault Févy, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as experts: Sparse memory access with entity supervision. CoRR, abs/2004.07202.

Matthew Francis-Landau, Greg Durrett, and Dan Klein. 2016. Capturing semantic similarity for entity linking with convolutional neural networks. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 1256–1261, San Diego, California. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL), pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasovi, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don’t stop pretraining: Adapt language models to domains and tasks.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. Neural Comput., 9(8):1735-1780.

Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. 2014. Discovering emerging entities with ambiguous names. In Proceedings of the 23rd international conference on World wide web, WWW 2014, Seoul, South Korea, pages 385–396.

Johannes Hoffart, Fabian M Suchanek, Klaus Berberich, and Gerhard Weikum. 2013. Yago2: A spatially and temporally enhanced knowledge base from wikipedia. Artificial Intelligence, 194:28–61.

Johannes Hoffart, Mohamed Amir Yosef, Ilaria Bordin, 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.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Filip Ilievski, Piek Vossen, and Stefan Schlobach. 2018. Systematic study of long tail phenomena in entity linking. In Proceedings of the 27th International Conference on Computational Linguistics, pages 664–674, Santa Fe, New Mexico, USA. Association for Computational Linguistics.
Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2019. How can we know what language models know? CoRR, abs/1911.12543.

Jeff Johnstone, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. arXiv preprint arXiv:1702.08734.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. CoRR, abs/2001.08361.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In Proceedings of the 22nd Conference on Computational Natural Language Learning, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.

Sachin Kumar and Yulia Tsvetkov. 2019. Von mises-fisher loss for training sequence to sequence models with continuous outputs. In Proc. of ICLR.

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.

Robert Logan, 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, pages 5962–5971, Florence, Italy. Association for Computational Linguistics.

Federico López, Benjamin Heinzerling, and Michael Strube. 2019. Fine-grained entity typing in hyperbolic space. In Proceedings of the 4th Workshop on Representation Learning for NLP (RepL4NLP-2019), pages 169–180, Florence, Italy. Association for Computational Linguistics.

Frederic Morin and Yoshua Bengio. 2005. Hierarchical probabilistic neural network language model. In Aistats, volume 5, pages 246–252.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In ICML, pages 809–816. Omnipress.

Yusuke Oda, Philip Arthur, Graham Neubig, Koichiro Yoshino, and Satoshi Nakamura. 2017. Neural machine translation via binary code prediction. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 850–860, Vancouver, Canada. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In Proceedings of NAACL-HLT 2019: Demonstrations.

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 701–710.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. 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 2227–2237, New Orleans, Louisiana. 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, Hong Kong, China. Association for Computational Linguistics.

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, Hong Kong, China. Association for Computational Linguistics.

Pouya Pezeshkpour, Liyan Chen, and Sameer Singh. 2018. Embedding multimodal relational data for knowledge base completion. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 3208–3218, Brussels, Belgium. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. E-bert: Efficient-yet-effective entity embeddings for bert.

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

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. Deep-type: Multilingual entity linking by neural type system evolution. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 9th AAAI Symposium on Educational Advances in Artificial Intelligence (EAII-18), New Orleans, Louisiana, USA, February 2-7, 2018, pages 5406–5413. AAAI Press.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 74–84, Atlanta, Georgia. Association for Computational Linguistics.

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

Claude E. Shannon. 1948. A mathematical theory of communication. Bell Syst. Tech. J., 27(3):379–423.

Daniil Sorokin and Iryna Gurevych. 2018. Modeling semantics with gated graph neural networks for knowledge base question answering. In Proceedings of the 27th International Conference on Computational Linguistics, pages 3306–3317. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS14, page 31043112, Cambridge, MA, USA. MIT Press.

Ryo Takahashi, Ran Tian, and Kentaro Inui. 2018. Interpretable and compositional relation learning by joint training with an autoencoder. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2148–2159. Melbourne, Australia. Association for Computational Linguistics.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, page 20712080. JMLR.org.

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

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CI Carey, Iñigo Polat, Yu Feng, Eric W. Moore, Jake Vand erPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17:261–272.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. Communications of the ACM, 57(10):78–85.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2019. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning, pages 250–259. Association for Computational Linguistics.

Zichao Yang, Phil Blunsom, Chris Dyer, and Wang Ling. 2017. Reference-aware language models. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 1850–1859, Copenhagen, Denmark. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.
## A Templates for generating English statements from Wikidata relations

| ID  | Template                                                                 | ID  | Template                                |
|-----|-------------------------------------------------------------------------|-----|-----------------------------------------|
| P31 | S is an instance of O                                                   | P413| S plays the position O                  |
| P106| S has the occupation O                                                  | P26 | S is spouse of O                        |
| P17 | S belongs to the country O                                              | P1830| S is owner of O                         |
| P131| S is located in the administrative territorial entity O                 | P1454| S has the legal form O                  |
| P27 | S is citizen of O                                                      | P206 | S is located in or next to body of water O |
| P47 | S shares a border with O                                               | P710 | S is a participant of O                 |
| P19 | S was born in O                                                        | P1441| S is present in the work O              |
| P161| S has the cast member O                                                | P1532| S represents O when playing sport O     |
| P421| S is located in time zone O                                             | P86 | S was composed by O                      |
| P166| S received the award O                                                 | P840 | S is set in the location O              |
| P54 | S is a member of the sports team O                                     | P172 | S belongs to the ethnic group O         |
| P20 | S died in O                                                            | P175 | S is performed by O                     |
| P136| S has the genre O                                                      | P57 | S is directed by O                      |
| P69 | S was educated at O                                                    | P1889| S is different from O                   |
| P1412| S is a language spoken, written or signed in O                         | P162 | S is produced by O                      |
| P190 | S is a twinned administrative body of O                                | P118 | S belongs to the league O               |
| P641 | S participates in the sport O                                           | P58 | S is screenwritten by O                 |
| P150 | S contains the administrative territory O                              | P551 | S has the residence O                   |
| P463 | S is a member of O                                                     | P103 | S has the native language O             |
| P735 | S has the given name O                                                 | P2789| S connects with O                       |
| P1343| S is described by source O                                             | P750 | S has the distributor O                  |
| P361 | S is a part of O                                                       | P725 | S is voiced by O                        |
| P159 | the headquarters of S are located in O                                 | P272 | S is produced by the company O          |
| P1344| S is participant of O                                                  | P112 | S was founded by O                      |
| P495 | S has the country of origin O                                           | P452 | S belongs to the industrial sector O    |
| P39  | S held the position O                                                  | P81 | S is connected to line O                |
| P910 | S has the main category O                                              | P97 | S has noble title O                     |
| P105 | S has the taxon rank O                                                 | P740 | S formed in the location O              |
| P527 | S has the part O                                                       | P360 | S is a list of O                        |
| P108 | S is employed by O                                                     | P793 | S is associated with the significant event O |
| P279 | S is a subclass of O                                                   | P915 | S was filmed at O                       |
| P171 | S has the parent taxon O                                               | P410 | S has military rank O                   |
| P140 | S has the religion O                                                   | P1001| S applies to the jurisdiction of O      |
| P407 | S is in the O language                                                 | P30 | S is located on the continent O         |
| P1411| S has been nominated for O                                             | P749 | S has parent organization O             |
| P102 | S is a member of political party O                                     | P1435| S has heritage designation O           |
| P3373| S is a sibling of O                                                    | P53 | S belongs to the family of O            |
| P1376| S is the capital of O                                                  | P400 | S was developed for the platform O      |
| P509 | S died because of O                                                    | P921 | S has the main subject O                |
| P937 | S works in O                                                           | P37 | S has the official language O           |
| P264 | S was produced by the record label O                                   | P734 | S has the family name O                 |
| P119 | S is buried in O                                                       | P22 | S is the father of O                    |
| P138 | S is named after O                                                     | P137 | S is operated by O                      |

Table 3: Templates used to generate English statements from Wikidata facts.
B  Random sample of English statements generated from Wikidata relations

- The Underfall Yard is followed by English Electric Part One
- Gazi Beg is a child of Farrukh Yassar
- 2011 European Rowing Championships is followed by 2012 European Rowing Championships
- 2009 Yemeni tourist attacks is located in Shibam
- George Best A Tribute is performed by Peter Corry
- Gamecock Media Group is owned by SouthPeak Games
- 201718 Sheffield Wednesday F.C. season is followed by 201819 Sheffield Wednesday F.C. season
- Nennslingen is located in or next to body of water Anlauter
- 201314 Xavier Musketeers men’s basketball team is followed by 201415 Xavier Musketeers men’s basketball team
- Shock to the System is a part of Cyberpunk
- 191819 Ohio Bobcats men’s basketball team follows 191718 Ohio Bobcats men’s basketball team
- Ramya Krishnan has the spouse Krishna Vamsi
- The Cloud Minders follows The Way to Eden
- Curve is followed by Somethingsness
- Austin Road is named after John Gardiner Austin
- Dione juno has the parent taxon Dione
- Spirit Bound Flesh is followed by The Wake
- Sidnei da Silva has the given name Sidnei
- In Memoriam is performed by Living Sacrifice
- Tracks and Traces is followed by Live 1974
- Grumman Gulfstream I is operated by Phoenix Air
- Timeline of Quebec history has the part Timeline of Quebec history (1982-present)
- Edwin C. Johnson held the position of Lieutenant Governor of Colorado
- Here Comes the Summer follows Jimmy Jimmy
- In Custody is screenwritten by Anita Desai
- Bertie Charles Forbes is the father of Malcolm Forbes
- The Mambo Kings has the cast member Helena Carroll
- Carnival of Souls has the cast member Art Ellison
- 199596 Philadelphia Flyers season is followed by 199697 Philadelphia Flyers season
- John Harley is the father of Edward Harley, 5th Earl of Oxford and Earl Mortimer
- Jane Fellowes, Baroness Fellowes has the spouse Robert Fellowes, Baron Fellowes
- Francis of Assisi is buried in Basilica of San Francesco d’Assisi
- 1990 Maharashtra Legislative Assembly election follows 1985 Maharashtra Legislative Assembly election
- Makabana Airport is named after Makabana
- Calvin Booth was born in Reynoldsburg
- The Telltale Head is followed by Life on the Fast Lane
- Alajos Keser is a sibling of Ferenc Keser
- Long An contains the administrative territorial entity Chu Thanh
### Hyperparameter settings

| Entity representation | Architecture | Hyper-param. | Value |
|-----------------------|--------------|--------------|-------|
| Symbolic | LSTM | layers | 2 |
| | | hidden size | 256, 1024 |
| | | dropout | 0.0 |
| | | learning rate | 0.001 |
| | | lr-scheduler | plateau |
| | | optimizer | Adam |
| Transformer | model name | RoBERTa-base |
| | layers | 12 |
| | hidden size | 768 |
| | learning rate | 5e-5 |
| | lr-scheduler | plateau |
| | optimizer | Adam |
| Surface form | LSTM | layers (enc) | 2 |
| | | hidden size (enc) | 256, 1024 |
| | | layers (dec) | 2 |
| | | hidden size (dec) | 256, 1024 |
| | | learning rate | 0.001 |
| | | lr-scheduler | plateau |
| | | optimizer | Adam |
| Transformer | model name (enc) | RoBERTa-base |
| | layers (enc) | 12 |
| | hidden size (enc) | 768 |
| | dropout | 0.0 |
| | model name (dec) | random init. |
| | layers (dec) | 12 |
| | hidden size (dec) | 768 |
| | learning rate | 5e-4 |
| | lr-scheduler | inverse sqrt |
| | optimizer | Adam |
| Continuous | LSTM | layers | 2 |
| | | hidden size | 256, 1024 |
| | | dropout | 0.0 |
| | | learning rate | 0.001 |
| | | lr-scheduler | plateau |
| | | optimizer | Adam |
| | | entity emb. dim | 64 |
| | | entity emb. trainable | no |
| Transformer | model name | RoBERTa-base |
| | layers | 12 |
| | hidden size | 768 |
| | learning rate | 5e-5 |
| | lr-scheduler | plateau |
| | optimizer | Adam |
| | | entity emb. dim | 64 |
| | | entity emb. trainable | no |

Table 4: Hyperparameter settings used in our experiments.
D Embeddings of Wikidata entities

We train the embedding of given Wikidata entity by collecting its features from, encoding each feature to obtain a dense feature representation, and then concatenating feature representations. For textual features, we use RoBERTa-base as encoder and train corresponding decoders in a standard sequence-to-sequence auto-encoding setup. For quantities, we select the 100 most common quantity types to obtain a fixed-sized representation and then follow a standard auto-encoding setup. Similarly we obtain a fixed-size entity type representation by selecting the 1000 most common entity types. The concatenated feature-representations are then compressed to embedding size $d$, using a separate autoencoder. Preliminary experiments with embedding sizes $d \in \{64, 128, 192, 256\}$ showed similar memorization accuracies for all $d$, but faster convergence for smaller sizes. We set $d = 64$ in our main experiments.
E  Things that didn’t work

E.1 Hierarchical entity representation with binary codes

Since imposing a hierarchy is a common method for dealing with large vocabulary sizes (Morin and Bengio, 2005) in general, and large inventories of entities and entity types in particular (Raiman and Raiman, 2018; López et al., 2019), we created a hierarchy of all entities in Wikidata, using a given entity’s position in this hierarchy as training signal. Specifically, we created the entity hierarchy by fitting a KD-tree (Bentley, 1975; Virtanen et al., 2020) with leaf size 1 over pretrained entity embeddings, thereby obtaining a binary partitioning of the embedding space in which each final partition contains exactly one entity embedding. The path from the KD-tree’s root to a leaf can be represented as a binary code, which we use as training signal (Oda et al., 2017). Memorization accuracy of world knowledge facts with object entities represented in the form of these binary codes was substantially lower compared to the three approaches described in the main part of this work.

E.2 Training entity embeddings with negative sampling

Instead of using fixed, pretrained entity embeddings as training signal, we experimented with randomly initialized embeddings that are updated during training, using between 1 and 50 in-batch negative samples, which is a standard method in the knowledge base embedding literature (Bordes et al., 2013) and has been used successfully for entity retrieval (Gillick et al., 2019). However, compared to using fixed, pretrained entity embeddings without negative sampling, we observed lower memorization accuracies and slower convergence in our experiments.

E.3 Updating pretrained entity embeddings during training

Instead of using fixed entity embeddings, we tried updating them during training with in-batch negative sampling. This increased the number of trainable parameters, memory usage, and training time, but did not lead to higher memorization accuracies.

E.4 Continuous representation with Euclidean distance loss

Instead of normalizing entity embeddings to the unit hypersphere and training with cosine loss, we experimented with predicting the original pretrained entity embeddings and using the Euclidean distance as loss. Compared to using spherical entity embeddings as prediction targets, we observed slower convergence and lower memorization accuracies.
Figure 11: Impact of graph type on a model’s ability to memorize the graph. We consider two types of random graphs, namely a uniform (Erdos-Renyi) graph, and a scale-free (Barabasi) graph. We interpret graph edges as relation triples in a knowledge graph and train models to predict the relation object, given subject and predicate, until memorization accuracy reaches 99 percent. For a given number of model parameters, we gradually increase the number of relation triples to memorizes and record the maximum number of relation triples memorized for this number of parameters. We compare an LSTM, as well as a bilinear KB embedding (DistMult). For a given parameter budget, models are able to memorize more triples from an Erdos-Renyi graph (blue) than from a Barabasi graph, indicating that the latter is more difficult to memorize.