Textual Entailment for Effective Triple Validation in Object Prediction

Andrés García-Silva, Cristian Berrío, and Jose Manuel Gómez-Pérez

Expert.ai, Language Technology Research Lab,
Poeta Joan Maragall 3, 28020 Madrid, Spain
agarcia@expert.ai, cberrio@expert.ai, jmgomez@expert.ai
https://www.expert.ai

Abstract. Knowledge base population seeks to expand knowledge graphs with facts that are typically extracted from a text corpus. Recently, language models pretrained on large corpora have been shown to contain factual knowledge that can be retrieved using cloze-style strategies. Such approach enables zero-shot recall of facts, showing competitive results in object prediction compared to supervised baselines. However, prompt-based fact retrieval can be brittle and heavily depend on the prompts and context used, which may produce results that are unintended or hallucinatory. We propose to use textual entailment to validate facts extracted from language models through cloze statements. Our results show that triple validation based on textual entailment improves language model predictions in different training regimes. Furthermore, we show that entailment-based triple validation is also effective to validate candidate facts extracted from other sources including existing knowledge graphs and text passages where named entities are recognized.

Keywords: Object Prediction · Knowledge Base Population · Recognizing Textual Entailment

1 Introduction

Knowledge Graphs arrange entities and relationships in a graph structure to represent knowledge [43, 29]. The edges of the graph describe relations between subject and object entities that are encoded as <*subject relation object*> triples. Knowledge graphs have applications in many areas, including search[1], recommendation, and natural language processing [22]. Nowadays the collaboration between editors and bots to curate and extend knowledge graphs has become common [50]. However, this is a complex and never-ending task and as a consequence, knowledge graphs are often incomplete [52, 12].

Knowledge Base Completion KBC [3] aims at predicting relations between existing entities. Similarly, the goal of the Knowledge Base Population KBP task [21] is to expand knowledge graphs with new facts discovered from text corpora. While in recent years a plethora of embeddings-based approaches have emerged

[1] https://blog.google/products/search/introducing-knowledge-graph-things-not/

for KBC [22], KBP research has not progressed at the same speed due to the complexity of the pipelines [21, 14] and the lack of established benchmarks.[2]

Recently, language models have been revealed as promising resources for KBP. Language models trained on large text corpora encode different types of knowledge including syntactic [26], semantic [47], commonsense [57], and factual knowledge [31]. To elicit facts from the internal memory of a language model, researchers typically use cloze statements to make the language model fill in the masked tokens, e.g., John Lennon plays <MASK>. Cloze statements, also known as prompts in this context, enable zero-shot fact retrieval without any fine-tuning [31]. Nevertheless, the knowledge encoded in the language model is limited to the data it has seen during pretraining. Additionally, prompt-based fact retrieval can be brittle [32] and heavily depend on the prompts and context used [9], which may produce results that are unintended or hallucinatory. For example, as a response to the previous prompt, BERT would return `guitar, piano, drums, himself, harmonica`. Lennon played percussion overdubs on some tracks, but he never actually played the drums. Further, while all of the remaining statements are true, "John Lennon plays himself" relates to his acting side, while we are interested in musical instruments. An apparently more specific prompt like John Lennon plays instrument <MASK> returns `here, there, too, himself, onstage`, adding even more noise.

To address such limitations we propose to validate candidate triples using textual entailment [34] against evidence retrieved from the Web. Within KBP, we focus on the object prediction task [42]. Given a subject entity and a relation, the goal is to predict every object that renders a valid triple, where such objects may not have been contained in the knowledge graph yet. In this paper we present our system SATORI (Seek And enTail for Object pRedIction). As shown in Fig. 1, SATORI obtains candidate objects from language models using cloze statements and generates candidate triples. To improve recall SATORI also considers other sources of candidate objects including external knowledge bases and named entities recognized in relevant text passages. A language model fine-tuned on the entailment task is used to validate whether the generated triples can be entailed from passages retrieved from the web. The objects of the triples validated by the model as entailment are the output of the system.

SATORI relies on templates to convert the input subject and relation pair into search engine queries to retrieve text passages, language model prompts to get candidate objects, and hypotheses describing candidate triples. Templates need to be defined only once per relation, and in its most basic form a template can be re-used across all the system components. For example, given the input pair *(John Lennon, PersonIntrument)* and a relation template {X} *plays*, we can submit the query *John Lennon plays* to the search engine, prompt the language model with *John Lennon plays <MASK>* for a candidate object, e.g., *Guitar*, and validate the entailment of the hypothesis *John Lennon plays Guitar* against the premises retrieved through our web search. Using language models as a source

[2] The KBP evaluation track of the TAC [14] is a long running initiative. However, manual system evaluation makes it hard to reproduce evaluation for new systems.

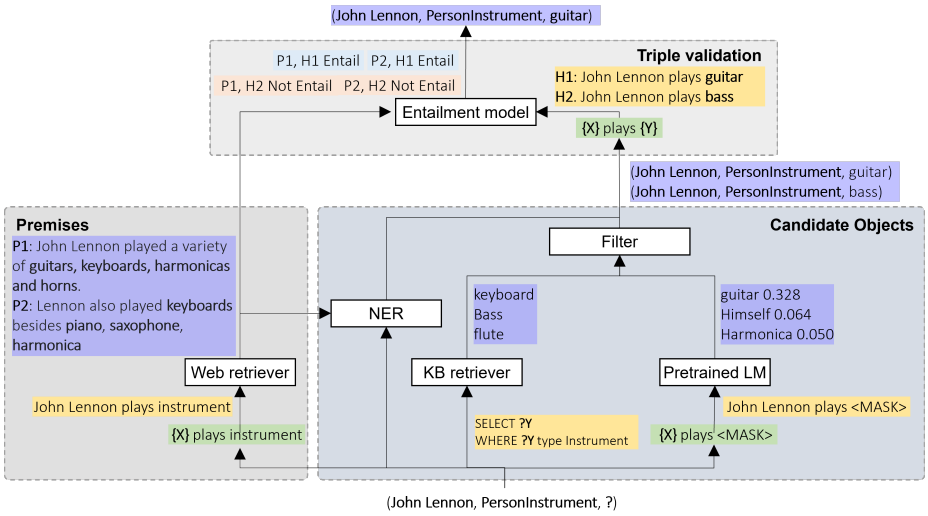


Fig. 1: SATORI architecture exemplified using as input pair *John Lennon* in the subject and *PersonInstrument* in the relation.

of factual knowledge, SATORI could also leverage breakthroughs in prompting and knowledge-based enhancement of language models [32].

Validating candidate triples generated from objects predicted using language models raises the following main research questions:

- Q1: Does candidate triple validation through textual entailment improve object prediction results over prompting a pretrained language model?
- Q2: How do further pretraining the language model and fine-tuning for entailment-based triple validation under different data regimes impact object prediction?
- Q3: How do language models compare to other potential sources of structured knowledge, as well as to methods based on extracting information from text, for the generation of candidate triples?

In this paper, we investigate such research questions and present the following contributions. First, an approach for object prediction including the validation of triples using textual entailment. Second, an experimental study where different configurations of SATORI and baseline systems are evaluated using a gold dataset for object prediction [42]. In such experiments we show that triple validation through textual entailment improves the performance of facts extracted from pretrained language models and also when additional training data is available. Finally, we compare language models with other sources including structured knowledge and unstructured text, which we process using extractive techniques repurposed for object prediction.

2 Related Work

Knowledge base population. A prototypical KBP architecture [4, 21] applies *entity linking* [54] on a corpus to recognize the entities mentioned in the text. Then a *slot filling* process [44] predicts an object, either an entity or a value for each subject entity and named relation, given the text where the entity appears. A key component of slot filling is a *relation extraction* model [1, 18] that identifies the relation between the entity and the candidate object. Thus, in KPB object prediction could complement the slot filling task since both share the goal of getting entities or values for a given subject and predicate. However, while slot filling leverages a large corpus to get the objects, in object prediction we rely on a language model to get objects and neither entity linking, relation extraction nor a text corpus are required.

Similar to our work West et al. [52] extract values from web search snippets although using question-answering (QA). Nevertheless, they experiment only with relations of subject type PERSON, and their approach only predict entities already in the knowledge graph. Unlike West et al., our work is framed in open world KBP [39], where predicted entities are not restricted to existing entities.

Knowledge base completion. KBC [22] mostly refers to *relation prediction* [3, 56] where the goal is to recognize a relation between two existing entities. However, KBC de facto evaluation turns relation prediction into a ranking task where triples with right objects (or subjects) are expected at the top [6, 55]. Such evaluation could lead to confusing KBC goal with object prediction. However in KBC the predicted objects are already part of the knowledge graph, while in object prediction they are not necessarily known.

Prompts and entity-enhanced languages models. To elicit relational knowledge from language models Petroni et al. [31] use prompting. They show that prompting BERT achieves competitive results againts non-neural and supervised alternatives. Prompts can be hand-crafted [31], mined from a large corpus [7] or learned [33, 40]. Li et al. [25] decompose prompts to split the task into multiple steps, and use task-specific pretraining for object prediction. Similar to our work Alivanistos et al. [2] add a fact probing step. However rather than using entailment for the validation, they ask a generative language model whether the generated fact is correct. Other works such as KnowBERT [30], and E-BERT [32] enhance BERT with pretrained entity embeddings, showing that such enhanced versions improve BERT results on LAMA [31].

Textual Entailment. It aims at recognizing, given two text fragments, whether the meaning of one text can be inferred (entailed) from the other [11]. The state of the art for textual entailment [51] is to fine-tune language models on datasets such as SNLI [8] or MNLI [53]. Researchers have reformulated several task as entailment. Wang et al. [51] convert single-sentence and sentence-pair classification tasks into entailment style. Entailment has been also used for zero-shot and few-shot relation extraction [37], answer validation [35], event argument extraction [36], and claim verification in fact checking [16]. Preliminary experiments suggest that a slot filling validation filter using textual entailment

could be useful for KBP systems [5]. To the best of our knowledge we are the first to use textual entailment to validate object prediction results.

Fact checking. Fact checking [16], originally proposed for assessing whether claims made in written or spoken language are true, have been applied also to check the trustworthiness of facts in knowledge graphs. Approaches for truth scoring of facts can be broadly classified into two types [23]: approaches that use unstructured textual data to find supporting evidential sentences for a given statement [13, 46, 48], and approaches that use a knowledge graph to find supporting evidential paths for a given statement [41, 45]. Fact checking, is a step beyond our triple validation approach, where trustworthiness of the triples and the sources is evaluated. Trustworthiness analysis is out of the scope of out work that is defined in the research questions that we pose.

3 Object Prediction

3.1 Task definition

Let $F \subset E \times R \times E$ be the set of facts in a knowledge graph, where E is the set of entities and R the set of relations. Given $s \in E$ and $r \in R$, the goal of the object prediction task is to find every $o \in E \cup E'$ where E' is the complement of E such that the triple (s, r, o) is a valid fact. Valid triples are added to F, and E is expanded when o is not an existing entity in the graph. In object prediction, as in other KBP tasks, the knowledge graph schema is available. The schema defines the relations in the knowledge graph including their range[3]. The range indicates that the objects of a particular relation are instances of a designated class. For example, the range of the *PersonInstrument* relation is the class *MusicalInstrument*.

3.2 SATORI: Seek and entail for object prediction

Recognizing textual entailment (RTE) is central in SATORI to validate triples. RTE is the task to determine whether a hypothesis H is true given a premise P. Typically for an input pair P and H, an RTE model assigns the labels Entailment (true), Contradiction (false) or Neutral [27]. We consider as hypothesis a natural language description of the candidate triple that we are validating, and as premises relevant text retrieved from the web. Thus SATORI has three major steps (see Fig. 1): i) retrieving premises, ii) getting candidate objects to generate new triples, and iii) validating triples using RTE. Along this section we use the input pair $s = JohnLennon$, $r = PersonInstrument$ as an example for the object prediction task.

Retrieving premises. The goal is to query a search engine using the input pair s and r to retrieve relevant text passages (featured snippets). For a relation r we define a search template $t_{search,r}$ as a keyword-based query including a placeholder for the subject. The subject s is replaced in the template $t_{search,r}$ and

[3] See RDF schema in https://www.w3.org/TR/rdf-primer/#properties

the output query is sent to the search engine API. We retrieve the top k featured snippets to use them as premises. For the example input pair and template $t_{search,r}$ = *X plays instrument*, our query is *John Lennon plays instrument*.

Getting candidate objects. Our first source of candidate objects is a pretrained language model. For each relation we define a template $t_{lm,r}$ of a prompt for the language model. The prompt is obtained by replacing the subject s in the template $t_{lm,r}$. For example the template $t_{lm,r}$ = *X plays <MASK>* becomes the prompt *John Lennon plays <MASK>*. In a bidirectional transformer like BERT the output vectors corresponding to the mask tokens are fed into an output softmax layer over the vocabulary. Hence the language model output for a mask token is the words in the vocabulary and the score indicating the probability of being the mask token in the input sequence. We use a per-relation threshold $T_{lm,r}$ on the word score. Predicted words with a score above the threshold are added to the set of candidate objects $objs = \{o_i\}$. Given an input pair (s, r) and each candidate object o_i we create a candidate triple (s, r, o_i)

In addition, we consider existing knowledge graphs as source of candidate objects. We use instances of the classes defined in the relation range as candidate objects. To retrieve such instances we use SPARQL queries. First, we get the classes C_j in the relation range from the schema of the knowledge graph that we are populating. Next, for each class in the relation range we send the following SPARQL query to the knowledge graph from where we want to obtain candidates: *SELECT ?y WHERE ?y rdf:type* C_j.[4] The retrieved entities are added to the set of candidate objects $objs = \{o_i\}$. For example, the relation *personInstrument* expects entities of class *MusicalInstrument* in the range. Thus, the SPARQL query generated for the example input pair is *SELECT ?y WHERE ?y rdf:type MusicalInstrument*.

Both language models and knowledge graphs can generate a high number of non-relevant candidate objects. We use heuristics to filter out unrelated objects. The most basic filter is a stop word list including punctuation marks. Language models are known for assigning high probability to punctuation marks and articles. In addition, we filter out objects which are not explicitly mentioned in the text passages gathered in the premise retrieval stage.

Named Entity Recognition NER can also be used to identify candidate objects for some relations depending on the classes in the relation range. Standard NER recognizes People, Locations, and Organizations[5]. We apply NER on the texts passages retrieved from the web for the input pair. If there is a match between the classes in the relation range and the classes of the recognized entities then we add such entities to the set of candidate objects $objs = \{o_i\}$.

Validating triples. We generate a short text description of a candidate triple (s, r, o_i) to be used as hypothesis to validate against the premises through RTE. We define per-relation templates $t_{h,r}$ to generate hypotheses containing placeholders for the subject (X) and object (Y). Then we replace the subject (X)

[4] While rdf:type is the standard property used to state that a resource is an instance of a class, some knowledge graphs could use other ad-hoc property.

[5] Due to the diverse nature of the MISC category we do not consider it.

and object (Y) placeholders with s and o_i to generate a hypothesis H_l. Given the candidate triple *(John Lennon, PersonInstrument, Guitar)*, and template $t_{h,r}$ = *X plays Y*, the generated hypothesis is H_l = *John Lennon plays guitar*. Next, we use a language model fine-tuned for the RTE task to evaluate whether H_l is entailed on average by the k premises. The objects o_i corresponding to an entailed hypothesis are the final objects for the input tuple (s, r).

Language models fine-tuned for RTE address the task as a multi-class classification of premise and hypothesis pairs into Entailment, Contradiction and Neutral classes. For transformers like BERT the input is a sequence *[CLS] premise [SEP] hypothesis [SEP]*. The [CLS] output vector $C \in R^H$, where H is the hidden size in the transformer, acts as the aggregated representation of the input sequence, and is connected to an output layer for classification $W \in^{K \times H}$, where K is the number of classes. Softmax is the activation function of the output layer and cross-entropy the loss function. In SATORI we focus on the Entailment and Contradiction classes, and softmax is applied only to the corresponding outputs. For each pair of premise and hypothesis the classifier generates scores for each of the classes. We use a per-relation threshold $T_{e,r}$ on the Entailment class score to accept the prediction as valid.

4 Evaluation setup

4.1 Dataset

The datasets used in the TAC KBP [14] evaluation series are tightly coupled with the text corpus used to mine facts: only information extracted from the corpus is considered valid to populate the knowledge graph. We discard such datasets since SATORI's web-based approach does not rely on a particular corpus. In addition, the evaluation of the slot-filling task, which is the most closely related to object prediction, is carried out manually.

Other triple-based datasets like LAMA [31], FB15K-237 [49] and WN18 [6], and their derived versions, were also discarded since they do not guarantee the completeness of objects for subject and relation pairs. Completeness of objects is important to evaluate precision and recall in object prediction. For instance, when predicting objects for relation *PersonInstrument* we want to predict all the instruments and not only some instruments for a given individual. In LAMA triples are randomly sampled, while in FB15K-237 and WN18 only frequent entities and relations are selected.

Moreover, optional relations that do not apply to every subject in the domain play an important role in object prediction evaluation. For example, the optional *PlaceOfDeath* relation only applies to people that have passed away, not to all the instances of people. Thus, to evaluate whether an object prediction system must produce or not an object for a subject and relation pair, the dataset needs to include pairs for which an object is not expected.

Therefore, we resort to the recently introduced LM_KBC22 [42] dataset[6] that includes all expected objects for a given subject and relation pair, is not tied to

[6] https://lm-kbc.github.io/2022/

a particular corpus, and comprises subject-relation pairs for which objects are not expected. The dataset includes 12 relations exemplified by a set of subjects and a complete list of ground-truth objects per subject and relation pair. Five relations also contain subjects without any correct ground truth objects. In this dataset relation domains are diverse, including *Person*, *Organization*, or *Chemical compound*. Similarly the range of relations includes *Country*, *Language*, *Chemical element*, *Profession*, *Musical Instruments*, or *Companies* to name a few classes. The training set includes 100 subjects per relation, and development and test sets each contain 50 subjects per relation.

4.2 Metrics

KBP systems are required to make accurate decisions with good coverage of the predicted entities. Rank-aware metrics customary used in KBC such as Mean Rank, and Mean Recriprocal Rank are of little use in a real KBP setting. They indicate how high in the ranking of predicted objects are the right ones. However in KBP we are interested in making actual predictions. Therefore, we use standard classification metrics that are good indicators of system accuracy and coverage: precision, recall, and F1.

4.3 SATORI configuration

We use the duckduckgo.com search engine to gather premises from the Web for each subject-relation pair. We leverage its python library[7] to send queries to the text search service. This service returns the web page title, url, and featured snippet that we keep as a premise. The duckduckgo library is very convenient since it can be used straightaway in Python programs and does not require registration in a cloud-based platform, unlike with leading search engines. We set k = 3 to gather at least three different premises that we can use to evaluate the hypotheses that we generate.

We evaluate three sources of candidate objects: pretrained language model (LM), knowledge graph (KG) and NER. We use BERT large cased as pretrained LM since it allows us to compare SATORI with the baseline model in the LM-KBC22 dataset. KG and NER are used together since NER recognizes a limited number of entity types and some entity types might not be covered in a KG. To perform NER we use a transformer model fine-tuned on the task[8]. We use NER to get candidate locations for relations *StateSharesBorderState* and *PersonPlaceOfDeath*, and candidate organizations for relations *CompanyParentOrganization* and *PersonEmployer*. We use Wikidata [50] as KG for the rest of the relations in the dataset. Wikidata includes instances of all classes in the ranges of the remaining relations: *Language*, *Country*, *ChemicalElement*, *MusicalInstrument*, *Profession*, and *CauseOfDeath*.

[7] https://pypi.org/project/duckduckgo-search/
[8] https://spacy.io/models/en#en_core_web_trf

To validate candidate triples we test three language models fine-tuned for the entailment task. We choose the models according to their performance on the entailment task and the number of parameters in the model. We use a DeBERTa Extra Large (XL) model [9] with 900M parameters that reports 91.7 accuracy on the MNLI dataset [53]. We also use DeBERTa Extra Small (XS)[10] model with 22M parameters that reports 88.1 accuracy on the same dataset. The final model that we test is a BERT large model[11] with 336M parameters that reports 86.7 accuracy on MNLI.

Templates $t_{lm,r}$, $t_{search,r}$, and $t_{h,r}$ used in the experiments are listed in the paper repository.[12] Particularly, templates $t_{lm,r}$ are reused from the language model baseline that we describe below.

4.4 Baselines

The first baseline that we use is prompting a language model. Such baseline allows to test our hypothesis that triple validation through textual entailment can benefit object prediction from language models. In addition, we are interested in comparing the knowledge in language models with other sources of knowledge including knowledge graphs and text passages related to the input subject and relation pair. Thus, we use as baselines a state of the art extractive question answering model and a relation extraction model repurposed for object prediction.

LM-baseline We reuse the baseline system from the LM_KBC challenge [42], which prompts a BERT-large-cased model. We use the templates to transform triples into prompts made available in the challenge repository.[13] The baseline uses a threshold $T_{lm,r}$ on the likelihood of words to fill in the mask token. Stop words are also filtered out. In addition, following Li et al. [25] we further-pretrain the language model on the masked language modeling objective using training data from the LM_KBC22 dataset. We transform triples in the training dataset into prompts where the objects are masked, and train the model to predict the right objects (see section 4.5 for further pretraining details).

The BERT-based system that scored highest in the LM_KBC22 challenge trains several BERT models (one model per relation) and further pretrains the models with data from Wikidata [25]. While we could have used such approach as a baseline and integrated it in SATORI as source of candidate objects, we decided against it since we think such approach does not scale due to the number of models being trained.

QA-baseline We use an extractive Question Answering (QA) system as baseline. To this end, we transform each subject-relation pair using per-relation templates into a question, where the answer is the expected object. The QA system then attempts to extract the answer from the text passages that we gather

[9] https://huggingface.co/microsoft/deberta-v2-xlarge-mnli
[10] https://huggingface.co/microsoft/deberta-v3-xsmall
[11] https://huggingface.co/boychaboy/MNLI_bert-large-cased
[12] https://github.com/satori2023/Textual-Entailment-for-Effective-Triple-Validation-in-Object-Prediction
[13] https://github.com/lm-kbc/dataset

from the Web as premises in SATORI for the same input pair (see section 3 for the premise retrieval procedure). QA per-relation templates $t_{qa,r}$ have a placeholder (X) for the subject entity and follow a similar basic pattern with minor variations: *What relation* X?*, where relation* is a text fragment derived from the relation label. For example, the template for the *PlaysInstrument* relation is: *What instruments plays X?*. The complete set of QA templates is available in the paper repository.

As QA model we use DeBERTa large[14] fine-tuned on SQUAD 2.0. DeBERTa is the highest scoring single model in the SQuAD 2.0 leaderboard[15] (F1=90.3) that is available on hugginface.co. We slightly post-process DeBERTa's answers containing lists of items to extract each item. For example, for the question *What instruments plays John Lennon?*, DeBERTa extracts the answer *guitar, keyboard, harmonica and horn.* From such list we extract each single instrument as a possible answer. Along with the answer span DeBERTa generates a score that indicates the probability of the answer. We use a per-relation threshold $T_{qa,r}$ on that score to accept or reject an answer. In addition we further fine-tune the QA model using questions and answers derived from the training set and the premise retrieved in SATORI as text passages from where the answers can be extracted (see section 4.5 for further details on QA fine-tuning).

RE-baseline We also test the state of the art relation extraction system REBEL [19] in the object prediction task. REBEL is a sequence to sequence model based on BART[24] that performs end-to-end relation extraction for more than 200 different relation types. REBEL autoregressively generates each triple present in the input text. To use REBEL we need to map the relations it supports with relations in the LM_KBC22 dataset. From the 12 relations in LM_KBC22 we find 10 relations with the same or very similar semantics in REBEL.[16] For the *ChemicalCompoundElement*, one of the remaining relations, we establish a mapping with the broader *has part* relation. The *PersonCauseOfDeath* relation could not be mapped. We use REBEL to generate triples from the text passages previously retrieved as premises for input subject-relation pairs. Finally, we extract the objects from those triples as the predicted objects.

4.5 Pretrained models and training regimes

We evaluate pretrained models off the shelf and once they are further trained using the LM_KBC22 dataset. Since the test set is withheld by the dataset authors, we test on the development set and use a held-out 20% of the training set as a new development set, leaving the remaining 80% as training data.

First we evaluate SATORI and the baselines using available pretrained models (see sections 4.3 and 4.4). To adjust the parameters that control the predictions of the different models for each relation, i.e. $T_{lm,r}$, $T_{e,r}$ in SATORI, $T_{lm,r}$ in the LM_baseline and $T_{qa,r}$ in the QA_baseline, we search for the best thresholds

[14] https://huggingface.co/deepset/deberta-v3-large-squad2
[15] https://rajpurkar.github.io/SQuAD-explorer/
[16] The relation mapping can be found in the paper repository

F1-wise in the range 0.01 to 0.99 with 0.01 increments over the union of training and development sets.

In addition, we train the models that we use in SATORI and the baselines on random samples of 5%, 10% and 20% of the training set corresponding on average to 4, 8, and 16 examples per relation. We also use 100% of the training data. The same percentages are used to sample the development set in each training scenario. Each training scenario is repeated 10 times, and evaluation metrics are averaged. We follow the same strategy to adjust the parameter that we use for pretrained models using the training and development sets. In the following we describe how we train the models.

Language model. The language model used in SATORI and the LM-baseline (BERT large) is further trained on the masked language model objective using the data from the training set in each scenario. We transform triples in the training and development set into prompts using the per-relation templates $t_{lm,r}$ and train the model to fill in the mask tokens with the corresponding objects. For the masked language modeling training objective, we set the hyper-parameters following [25]. That is, we train the model for 10 epochs, use a learning rate of 5e-6, an a batch size of 32. We save checkpoints at the end of every epoch, considering as the best checkpoint the one with the lowest development set loss.

Entailment model. An entailment training instance comprises a premise, hypothesis, and a label indicating Entailment or Contradiction. We create entailment training instances as follows. For each triple in the training set we use the text passages retrieved as premises for the subject and relation in the triple. If the subject and object are mentioned in the text passage we generate a positive entailment example using the passage as premise and the hypothesis that we generate using the corresponding template $t_{h,r}$. To generate negative examples, i.e., contradictions, we replace the object in the positive example with an incorrect object, and use a premise retrieved for the input pair where such object is mentioned. To obtain the incorrect object we prompt the language model using the per-relation template $t_{lm,r}$ where we replace the input subject. We keep objects appearing in any of the premises for the input pair, that are not related to the input subject-relation pair in the training set. If we do not find any object from the language model that satisfies the previous condition, we look for incorrect objects in the training set for the same relation.

To train the entailment model, we reuse the classification scripts available in HuggingFace.[17] We use the default hyper-parameters: 3 epochs, learning rate of 2e-5, and maximum sequence length of 128. Due to hardware limitations we use a batch size of 8, and gradient accumulation steps of 4. Particularly, to fine-tune the DeBERTa xlarge model, we reduce further the batch size to 1 and increase gradient accumulation steps to 32, applying gradient checkpointing, and use "Adafactor" optimizer instead of the default "AdamW".

QA-baseline. A training instance for an extractive QA model includes a question, an answer and a text from where the answer is extracted. To generate

[17] https://github.com/huggingface/transformers/tree/v4.24.0/examples/pytorch/text-classification

training instances we first pre-process the training set and obtain for each subject and relation the complete lists of objects. Next, we get the text passages retrieved as premises for each subject and relation pair. For entries with empty object lists, we use the first retrieved passage and also use the empty gold answer.

If there is only one object for the input subject and relation pair, and there is a text passage where the object appears, we generate a training example with the passage, the question that we generate using the template $t_{qa,r}$ and the object as answer. If there is more than one object for the subject and relation the only way to extract them from a passage using a QA model is if they are arranged as a list of terms in a contiguous text span. Therefore we look for text passages containing every object with the condition that they must be located at most three tokens away from each other. We select the passage with the span containing the highest number of objects, and generate the training example with such passage, the question that we generate using the template $t_{qa,r}$, and the text span where objects appear as answer.

To train the question answering model we leverage HuggingFace scripts[18], and use the default hyper-parameters: batch size of 12, learning rate of 3e-5, 2 training epochs, and a maximum sequence length of 384.

RE-baseline. Training instances for REBEL consist of a short text, followed by the triples that can be extracted from that text. Triples are described in a REBEL-specific format using special tokens. Relations in triples are indicated using their text descriptions. To transform instances from the LB_KBC22 dataset into REBEL training instances we first obtain for each subject and relation the complete lists of objects. Next, we get the text passages retrieved as premises for each subject and relation pair. If we find a text passage containing some of the objects related to subject and relation pair we create a REBEL training instance with such text, and the triples for the subject, relation and objects found in the text. To train the model we use the script[19] provided by the REBEL authors. We use the default training parameters in the script.

5 Evaluation results

Table 1 shows evaluation results in the object prediction task for SATORI and baselines using pretrained models and models further trained on the different training regimes. Note that SATORI is evaluated using different RTE models for triple validation and also using different sources of candidate objects.

5.1 Language model prompting and triple validation

Let us start with Q1 (does candidate triple validation through textual entailment improve object prediction results over pretrained language model prompting?). We compare SATORI using RTE for triple validation and a pretrained language

[18] https://github.com/huggingface/transformers/tree/v4.24.0/examples/pytorch/question-answering

[19] https://github.com/Babelscape/rebel/blob/main/src/train.py

model as source of candidate objects (Table 1: first three rows in the Pretrained column) against the LM_baseline using only a pretrained language model.

Results show that in such scenario triple validation using any RTE model improves the LM_baseline in all evaluation metrics. The largest gain is in precision, with an improvement that ranges from 14 to 15.5 points depending on the RTE model. Recall is also improved, although to a lesser extent, in the range of 1.9 to 3.2. Improvement in precision shows that triple validation using RTE is effective to filter out non-relevant objects predicted by the language model. Since the system is more precise, the threshold limiting the number of objects predicted per relation $T_{lm,r}$ is actually lower in SATORI than in the LM_baseline. A lower $T_{lm,r}$ allows obtaining more candidates from the language model, thus the improvement on recall. Overall, triple validation using RTE improves over the LM_baseline with a margin ranging between 4.7 and 5.4 F1 points and research question Q1 is therefore satisfactorily addressed.

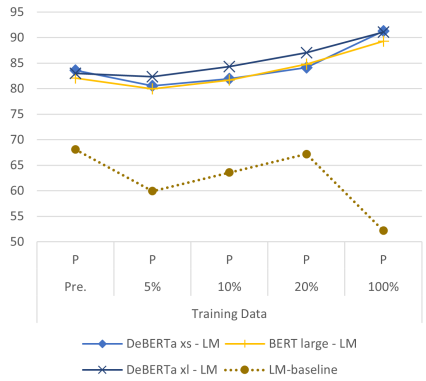
Table 1: Object prediction using pretrained models vs. such models additionally trained in different scenarios. The models include the LM from where we get objects and the entailment model that we use for triple validation. The SATORI versions evaluated include three sources of candidate objects (Language Model LM, Knowledge Graph KG, Named Entities NER) and three entailment models (DeBERTa xs, BERT large, DeBERTa xl).

SATORI		**Pretrained**			**Training Data** 5%			10%			20%			100%		
RTE Model	**Obj. Source**	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
DeBERTa xs	LM	83.6	48.6	47.9	80.5	51.7	47.9	81.9	51.5	48.4	84.1	51.3	50.3	**91.3**	44.7	46.8
BERT large	LM	82.1	49.9	48.6	79.9	52.2	48.1	81.7	51.8	48.7	84.8	51.2	50.6	89.3	46.0	47.1
DeBERTa xl	LM	83.0	49.5	48.5	82.4	51.9	48.5	**84.3**	52.1	50.1	**87.0**	52.0	51.4	91.1	47.8	48.4
DeBERTa xs	KG+NER	67.7	61.8	55.2	67.0	60.9	52.1	69.5	60.1	52.6	71.0	60.8	53.8	73.5	60.4	54.7
BERT large	KG+NER	67.0	62.9	55.2	68.2	61.8	52.2	70.0	61.1	52.9	69.9	63.0	54.9	66.9	64.8	55.1
DeBERTa xl	KG+NER	70.0	62.5	**55.9**	71.4	62.2	**54.9**	72.6	62.3	**55.1**	70.7	63.6	**56.5**	68.3	64.0	54.8
DeBERTa xs	LM+KG+NER	66.6	62.0	54.4	65.4	61.2	51.5	67.1	60.8	51.8	69.1	61.5	53.4	73.3	60.5	54.6
BERT large	LM+KG+NER	64.5	63.2	54.7	65.7	**62.6**	51.5	68.2	61.6	52.7	68.6	63.5	54.8	67.2	**65.1**	55.7
DeBERTa xl	LM+KG+NER	63.5	**64.3**	53.9	69.4	62.6	54.2	70.6	**63.1**	54.8	69.5	**64.2**	**56.5**	70.0	64.3	**56.6**
LM-baseline		68.1	46.7	43.2	59.9	50.8	43.3	63.6	49.0	44.3	67.2	49.6	46.9	52.2	48.1	41.4
QA-baseline		67.1	42.3	40.9	54.1	45.2	38.5	57.9	45.4	40.6	57.6	46.5	41.7	57.4	52.7	47.3
RE-baseline		**85.5**	33.3	31.8	**83.8**	41.8	40.6	81.3	43.0	42.0	78.6	44.7	43.2	70.0	54.9	48.2

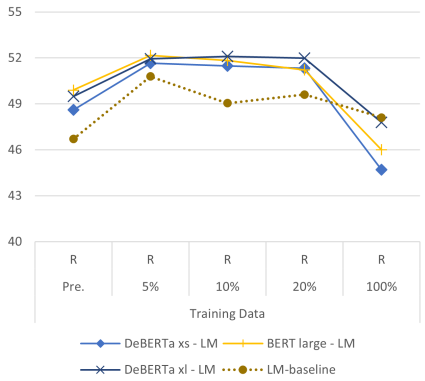
5.2 Additional pretraining and fine-tuning scenarios

To address research question Q2 (how do further pretraining the language model and fine-tuning for entailment-based triple validation under different data regimes impact object prediction?) we analyse evaluation results when models are further trained. In Fig. 2c we can see that triple validation using any RTE model improves F1 across all training scenarios compared to LM_baseline. Interestingly the LM_baseline F1 is lower in full training than the pretrained version and the other low data training regimes. Pending further experimentation, this can

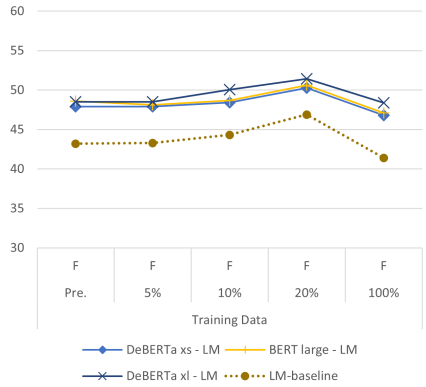
indicate a case of catastrophic forgetting [28, 15, 38], where, upon further pretraining, language models might lose some of the (in this case, factual) knowledge previously acquired during pretraining.



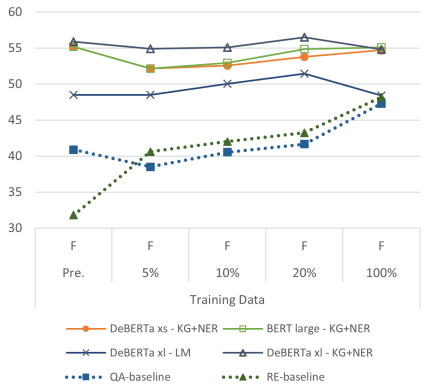
(a) Precision: LM baseline and SATORI using different RTE models and objects from LM.



(b) Recall: LM baseline and SATORI using different RTE models and objects from LM.



(c) F1: LM baseline and SATORI using different RTE models and objects from LM



(d) F1: best SATORI getting objects from LM, vs. using other knowledge sources and the QA and RE baselines.

Fig. 2: Object prediction evaluation on the LM_KBC22 dataset.

The more training data available, the more precise the predicted objects using triple validation are (see Fig. 2a). Triple validation also improves recall, although in a narrower margin, across most training scenarios except in full training (see Fig. 2b). Nevertheless, the use of more training data does not imply an improvement in recall for SATORI. Such recall pattern reflects the higher

precision the system achieves when more training data is used. That is, the triple validation filter becomes more and more strict, thus accepting less predicted objects. In low data regimes (5% of the training data) the LM_baseline precision decreases compared to results achieved with pretrained models. However, the precision using triple validation does not fall at the same rate. This shows that the triple validation component is effective in low data regimes to filter out non relevant predicted objects.

5.3 Language models vs structured knowledge sources and information extraction methods

To answer research question Q3 (how do language models compare to other potential sources of structured knowledge, as well as to methods based on extracting information from text, for the generation of candidate triples?) we first start comparing the evaluation results of LM_baseline and the QA and RE baselines, which in addition leverage text passages. Table 1 shows that in pretrained and most training regimes, except in full training, the QA and RE baselines achieve worse F1 than retrieving objects from the language models. Nevertheless, in a full training scenario the QA and RE baselines achieve an F1 score that is higher than the one obtained with LM_baseline and similar to the one that we obtained with SATORI, using triple validation (see Fig. 2d). Therefore, when enough training data is available extractive methods relying on state-of-the art QA and Relation Extraction are better for object prediction than relying only on LM as source of objects, and comparable to using triple validation to curate candidate objects retrieved from a LM.

Nevertheless, the best performance in object prediction is achieved when we use KG and NER as source of candidate objects along with triple validation (see Fig. 2d). Using KG and NER plus triple validation is the best option for object prediction in low data training regimes (5%, 10% and 20%). Enhancing the list of candidate objects from KG and NER including objects from LM slightly enhances F1 in full training. Manual inspection shows that in full training, the number of candidate objects obtained from the LM is 794, from the KG it is 2338 and from NER it is 3506. While the number of objects from LM is lower due to the thresholds on the LM score and filters, such objects contribute to slightly increase the overall performance when combined with objects from KG and NER. The large number of objects obtained from KG+NER and the evaluation results shows that triple validation is effective to filter out non-relevant objects. In fact, if we remove the triple validation component and use objects from LM, NER and KG in the full training scenario we get 0.276 precision, 0.761 recall, and 0.327 F1. Such training scenario without triple validation is even worse than the LM_baseline in terms of F1.

6 Conclusions

In this paper we posit that object prediction relying on prompting language models can be enhanced through triple validation using textual entailment. Our

experiments show that triple validation provides a boost in object prediction performance using pretrained models and when such models are further fine-tuned under different training regimes. In addition, we compare language models with other sources of knowledge including existing knowledge graphs and relevant text passages where techniques such as named entity recognition NER, question answering QA and relation extraction RE can be applied for object prediction.

We find that in low data regimes, getting objects using language model prompting with or without triple validation is better than using extractive QA and RE models. However in full training the performance of the QA and RE models surpass language model prompting and reach the performance achieved when we validate the triples proposed by the language model. Moreover, using existing knowledge graphs and NER on text passages as source of candidate objects along with triple validation shows the overall best performance when pretrained models and models fine-tuned in low data regimes are evaluated. Adding candidate objects from language models to the ones found on KG and using NER is the best option in full training.

The results presented in this paper are limited to the language model that we use, particularly BERT large, as primary source of candidate objects. Further research is needed to understand whether triple validation will continue having a positive effect in the object prediction task using a language model with a larger number of parameters and if this could balance the current gap between LMs and KGs when it comes to proposing candidate objects that we identified here. As future work, we think Entailment Graphs [17, 10] could be useful to get better per-relation templates or to improve the entailment model used in triple validation. In addition, we plan to assess how triple classification [20] compares to textual entailment for triple validation, and whether fact checking techniques [13] applied to triple validation brings new improvements. Finally, another research avenue is to use triple validation along with the QA and RE models given the good results that we obtained in full training when we use them for object prediction.

Supplemental Material Statement: All necessary resources to reproduce the experiments presented in section 5 are available in Github.[20] The repository includes code, templates and text passages that we use as premises in triple validation, and as input in the QA and RE baselines.

Acknowledgements. We are grateful to the European Commission (EU Horizon 2020 EXCELLENT SCIENCE - Research Infrastructure under grant agreement No. 101017501 RELIANCE) and ESA (Contract No. 4000135254/21/NL/GLC/kk FEPOSI) for the support received to carry out this research.

[20] https://github.com/expertailab/Textual-Entailment-for-Effective-Triple-Validation-in-Object-Prediction

References

1. Adel, H., Schütze, H.: Type-aware convolutional neural networks for slot filling. Journal of Artificial Intelligence Research **66**, 297–339 (2019)
2. Alivanistos, D., Santamaría, S., Cochez, M., Kalo, J., van Krieken, E., Thanapalasingam, T.: Prompting as probing: Using language models for knowledge base construction. In: Singhania, S., Nguyen, T.P., Razniewski, S. (eds.) LM-KBC 2022 Knowledge Base Construction from Pre-trained Language Models 2022. pp. 11–34. CEUR Workshop Proceedings, CEUR-WS.org (2022)
3. Balazevic, I., Allen, C., Hospedales, T.: TuckER: Tensor factorization for knowledge graph completion. 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). pp. 5185–5194. Association for Computational Linguistics, Hong Kong, China (Nov 2019). https://doi.org/10.18653/v1/D19-1522, https://aclanthology.org/D19-1522
4. Balog, K.: Populating Knowledge Bases. In: Balog, K. (ed.) Entity-Oriented Search, pp. 189–222. The Information Retrieval Series, Springer International Publishing (2018). https://doi.org/10.1007/978-3-319-93935-3_6
5. Bentivogli, L., Clark, P., Dagan, I., Giampiccolo, D.: The seventh pascal recognizing textual entailment challenge. Theory and Applications of Categories (2011)
6. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., Yakhnenko, O.: Translating Embeddings for Modeling Multi-relational Data. In: Advances in Neural Information Processing Systems. vol. 26. Curran Associates, Inc. (2013), https://papers.nips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html
7. Bouraoui, Z., Camacho-Collados, J., Schockaert, S.: Inducing relational knowledge from bert. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 34, pp. 7456–7463 (2020)
8. Bowman, S.R., Angeli, G., Potts, C., Manning, C.D.: A large annotated corpus for learning natural language inference. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 632–642. Association for Computational Linguistics, Lisbon, Portugal (Sep 2015). https://doi.org/10.18653/v1/D15-1075, https://aclanthology.org/D15-1075
9. Cao, B., Lin, H., Han, X., Sun, L., Yan, L., Liao, M., Xue, T., Xu, J.: Knowledgeable or educated guess? revisiting language models as knowledge bases. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers). pp. 1860–1874. Association for Computational Linguistics, Online (Aug 2021). https://doi.org/10.18653/v1/2021.acl-long.146, https://aclanthology.org/2021.acl-long.146
10. Chen, Z., Feng, Y., Zhao, D.: Entailment graph learning with textual entailment and soft transitivity. In: Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). pp. 5899–5910. Association for Computational Linguistics, Dublin, Ireland (May 2022). https://doi.org/10.18653/v1/2022.acl-long.406, https://aclanthology.org/2022.acl-long.406
11. Dagan, I., Glickman, O., Magnini, B.: The pascal recognising textual entailment challenge. In: Machine learning challenges workshop. pp. 177–190. Springer (2005)
12. Galárraga, L., Razniewski, S., Amarilli, A., Suchanek, F.M.: Predicting completeness in knowledge bases. In: Proceedings of the

Tenth ACM International Conference on Web Search and Data Mining. p. 375–383. WSDM '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3018661.3018739, https://doi.org/10.1145/3018661.3018739
13. Gerber, D., Esteves, D., Lehmann, J., Bühmann, L., Usbeck, R., Ngonga Ngomo, A.C., Speck, R.: Defacto-temporal and multilingual deep fact validation. Web Semant. **35**(P2), 85–101 (dec 2015). https://doi.org/10.1016/j.websem.2015.08.001, https://doi.org/10.1016/j.websem.2015.08.001
14. Getman, J., Ellis, J., Strassel, S., Song, Z., Tracey, J.: Laying the Groundwork for Knowledge Base Population: Nine Years of Linguistic Resources for TAC KBP. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018). European Language Resources Association (ELRA), Miyazaki, Japan (May 2018), https://aclanthology.org/L18-1245
15. Goodfellow, I.J., Mirza, M., Da, X., Courville, A.C., Bengio, Y.: An empirical investigation of catastrophic forgeting in gradient-based neural networks. CoRR **abs/1312.6211** (2013)
16. Guo, Z., Schlichtkrull, M., Vlachos, A.: A survey on automated fact-checking. Transactions of the Association for Computational Linguistics **10**, 178–206 (2022). https://doi.org/10.1162/tacl_a_00454, https://aclanthology.org/2022.tacl-1.11
17. Hosseini, M.J., Cohen, S.B., Johnson, M., Steedman, M.: Open-domain contextual link prediction and its complementarity with entailment graphs. In: Findings of the Association for Computational Linguistics: EMNLP 2021. pp. 2790–2802. Association for Computational Linguistics, Punta Cana, Dominican Republic (Nov 2021). https://doi.org/10.18653/v1/2021.findings-emnlp.238, https://aclanthology.org/2021.findings-emnlp.238
18. Huang, L., Sil, A., Ji, H., Florian, R.: Improving slot filling performance with attentive neural networks on dependency structures. In: EMNLP (2017)
19. Huguet Cabot, P.L., Navigli, R.: REBEL: Relation extraction by end-to-end language generation. In: Findings of the Association for Computational Linguistics: EMNLP 2021. pp. 2370–2381. Association for Computational Linguistics, Punta Cana, Dominican Republic (Nov 2021). https://doi.org/10.18653/v1/2021.findings-emnlp.204, https://aclanthology.org/2021.findings-emnlp.204
20. Jaradeh, M.Y., Singh, K., Stocker, M., Auer, S.: Triple classification for scholarly knowledge graph completion. In: Proceedings of the 11th on Knowledge Capture Conference. pp. 225–232 (2021)
21. Ji, H., Grishman, R.: Knowledge Base Population: Successful Approaches and Challenges. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies. pp. 1148–1158. Association for Computational Linguistics, Portland, Oregon, USA (Jun 2011), https://aclanthology.org/P11-1115
22. Ji, S., Pan, S., Cambria, E., Marttinen, P., Yu, P.S.: A survey on knowledge graphs: Representation, acquisition, and applications. IEEE Transactions on Neural Networks and Learning Systems **33**(2), 494–514 (2022). https://doi.org/10.1109/TNNLS.2021.3070843
23. Kim, J., Choi, K.s.: Unsupervised fact checking by counter-weighted positive and negative evidential paths in a knowledge graph. In: Proceedings of the 28th International Conference on Computational Linguistics. pp. 1677–1686. International Committee on Computational Linguistics, Barcelona, Spain (Online) (Dec 2020). https://doi.org/10.18653/v1/2020.coling-main.147, https://aclanthology.org/2020.coling-main.147

24. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 7871–7880. Association for Computational Linguistics, Online (Jul 2020). https://doi.org/10.18653/v1/2020.acl-main.703, https://aclanthology.org/2020.acl-main.703
25. Li, T., Huang, W., Papasarantopoulos, N., Vougiouklis, P., Pan, J.Z.: Task-specific pre-training and prompt decomposition for knowledge graph population with language models. ArXiv **abs/2208.12539** (2022)
26. Liu, N.F., Gardner, M., Belinkov, Y., Peters, M.E., Smith, N.A.: Linguistic Knowledge and Transferability of Contextual Representations. 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). pp. 1073–1094. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1112, https://aclanthology.org/N19-1112
27. MacCartney, B., Manning, C.D.: Modeling semantic containment and exclusion in natural language inference. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008). pp. 521–528. Coling 2008 Organizing Committee, Manchester, UK (Aug 2008), https://aclanthology.org/C08-1066
28. McCloskey, M., Cohen, N.J.: Catastrophic interference in connectionist networks: The sequential learning problem. In: Psychology of learning and motivation, vol. 24, pp. 109–165. Elsevier (1989)
29. Miller, G.A.: Wordnet: a lexical database for english. Communications of the ACM **38**(11), 39–41 (1995)
30. Peters, M.E., Neumann, M., Logan, R., Schwartz, R., Joshi, V., Singh, S., Smith, N.A.: 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). pp. 43–54 (2019)
31. Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P., Bakhtin, A., Wu, Y., Miller, A.: 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). pp. 2463–2473. Association for Computational Linguistics, Hong Kong, China (Nov 2019). https://doi.org/10.18653/v1/D19-1250, https://aclanthology.org/D19-1250
32. Poerner, N., Waltinger, U., Schütze, H.: E-bert: Efficient-yet-effective entity embeddings for bert. In: Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 803–818 (2020)
33. Qin, G., Eisner, J.: Learning how to ask: Querying LMs with mixtures of soft prompts. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 5203–5212. Association for Computational Linguistics, Online (Jun 2021). https://doi.org/10.18653/v1/2021.naacl-main.410, https://aclanthology.org/2021.naacl-main.410
34. Richardson, K., Hu, H., Moss, L., Sabharwal, A.: Probing Natural Language Inference Models through Semantic Fragments. Proceedings of the AAAI Conference on Artificial Intelligence **34**(05),

8713–8721 (Apr 2020). https://doi.org/10.1609/aaai.v34i05.6397, https://ojs.aaai.org/index.php/AAAI/article/view/6397
35. Rodrigo, Á., Peñas, A., Verdejo, F.: Overview of the answer validation exercise 2008. In: Peters, C., Deselaers, T., Ferro, N., Gonzalo, J., Jones, G.J.F., Kurimo, M., Mandl, T., Peñas, A., Petras, V. (eds.) Evaluating Systems for Multilingual and Multimodal Information Access. pp. 296–313. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
36. Sainz, O., Gonzalez-Dios, I., Lopez de Lacalle, O., Min, B., Agirre, E.: Textual Entailment for Event Argument Extraction: Zero- and Few-Shot with Multi-Source Learning. In: Findings of the Association for Computational Linguistics: NAACL 2022. pp. 2439–2455. Association for Computational Linguistics, Seattle, United States (Jul 2022). https://doi.org/10.18653/v1/2022.findings-naacl.187, https://aclanthology.org/2022.findings-naacl.187
37. Sainz, O., de Lacalle, O.L., Labaka, G., Barrena, A., Agirre, E.: Label verbalization and entailment for effective zero and few-shot relation extraction. ArXiv **abs/2109.03659** (2021)
38. Serra, J., Suris, D., Miron, M., Karatzoglou, A.: Overcoming catastrophic forgetting with hard attention to the task. In: International Conference on Machine Learning. pp. 4548–4557. PMLR (2018)
39. Shi, B., Weninger, T.: Open-World Knowledge Graph Completion. Proceedings of the AAAI Conference on Artificial Intelligence **32**(1) (Apr 2018). https://doi.org/10.1609/aaai.v32i1.11535, https://ojs.aaai.org/index.php/AAAI/article/view/11535, number: 1
40. Shin, T., Razeghi, Y., Logan IV, R.L., Wallace, E., Singh, S.: AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 4222–4235. Association for Computational Linguistics, Online (Nov 2020). https://doi.org/10.18653/v1/2020.emnlp-main.346, https://aclanthology.org/2020.emnlp-main.346
41. Shiralkar, P., Flammini, A., Menczer, F., Ciampaglia, G.L.: Finding streams in knowledge graphs to support fact checking. In: 2017 IEEE International Conference on Data Mining (ICDM). pp. 859–864 (2017). https://doi.org/10.1109/ICDM.2017.105
42. Singhania, S., Nguyen, T.P., Razniewski, S.: LM-KBC: Knowledge Base Construction from Pre-trained Language Models. In: the Semantic Web Challenge on Knowledge Base Construction from Pre-trained Language Models 2022 co-located with the 21st International Semantic Web Conference (ISWC2022). vol. Vol-3274. Hanghzou, China (Oct 2022), https://ceur-ws.org/Vol-3274/paper1.pdf
43. Speer, R., Chin, J., Havasi, C.: Conceptnet 5.5: An open multilingual graph of general knowledge. In: Thirty-first AAAI conference on artificial intelligence (2017)
44. Surdeanu, M., Ji, H.: Overview of the english slot filling track at the tac2014 knowledge base population evaluation. In: Proc. Text Analysis Conference (TAC2014) (2014)
45. Syed, Z.H., Röder, M., Ngomo, A.C.N.: Unsupervised discovery of corroborative paths for fact validation. In: Ghidini, C., Hartig, O., Maleshkova, M., Svátek, V., Cruz, I., Hogan, A., Song, J., Lefrançois, M., Gandon, F. (eds.) The Semantic Web – ISWC 2019. pp. 630–646. Springer International Publishing, Cham (2019)
46. Syed, Z.H., Röder, M., Ngonga Ngomo, A.C.: Factcheck: Validating rdf triples using textual evidence. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management. p. 1599–1602. CIKM '18, Association for Computing Machinery,

New York, NY, USA (2018). https://doi.org/10.1145/3269206.3269308, https://doi.org/10.1145/3269206.3269308
47. Tenney, I., Xia, P., Chen, B., Wang, A., Poliak, A., McCoy, R.T., Kim, N., Durme, B.V., Bowman, S.R., Das, D., Pavlick, E.: What do you learn from context? probing for sentence structure in contextualized word representations. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=SJzSgnRcKX
48. Thorne, J., Vlachos, A.: Automated fact checking: Task formulations, methods and future directions. In: Proceedings of the 27th International Conference on Computational Linguistics. pp. 3346–3359. Association for Computational Linguistics, Santa Fe, New Mexico, USA (Aug 2018), https://aclanthology.org/C18-1283
49. Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., Gamon, M.: Representing Text for Joint Embedding of Text and Knowledge Bases. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing. pp. 1499–1509. Association for Computational Linguistics, Lisbon, Portugal (Sep 2015). https://doi.org/10.18653/v1/D15-1174, https://aclanthology.org/D15-1174
50. Vrandečić, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Communications of the ACM **57**(10), 78–85 (2014)
51. Wang, S., Fang, H., Khabsa, M., Mao, H., Ma, H.: Entailment as Few-Shot Learner (Apr 2021), http://arxiv.org/abs/2104.14690, arXiv:2104.14690 [cs]
52. West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., Lin, D.: Knowledge base completion via search-based question answering. In: Proceedings of the 23rd international conference on World wide web. pp. 515–526. WWW '14, Association for Computing Machinery, New York, NY, USA (Apr 2014). https://doi.org/10.1145/2566486.2568032, https://doi.org/10.1145/2566486.2568032
53. Williams, A., Nangia, N., Bowman, S.: A broad-coverage challenge corpus for sentence understanding through inference. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). pp. 1112–1122. Association for Computational Linguistics, New Orleans, Louisiana (Jun 2018). https://doi.org/10.18653/v1/N18-1101, https://aclanthology.org/N18-1101
54. Wu, L., Petroni, F., Josifoski, M., Riedel, S., Zettlemoyer, L.: Scalable zero-shot entity linking with dense entity retrieval. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP). pp. 6397–6407 (2020)
55. Yao, L., Mao, C., Luo, Y.: Kg-bert: Bert for knowledge graph completion. arXiv preprint arXiv:1909.03193 (2019)
56. Zha, H., Chen, Z., Yan, X.: Inductive Relation Prediction by BERT. In: Proceedings of the First MiniCon Conference (Feb 2022), https://aaai-022.virtualchair.net/poster_aaai7162
57. Zhou, X., Zhang, Y., Cui, L., Huang, D.: Evaluating commonsense in pre-trained language models. In: AAAI (2020)