Knowledge-driven Self-supervision for Zero-shot Commonsense Question Answering

Kaixin Ma¹, Filip Ilievski², Jonathan Francis¹,³, Yonatan Bisk¹, Eric Nyberg¹, Alessandro Oltramari³

¹Language Technologies Institute, School of Computer Science, Carnegie Mellon University
²Information Sciences Institute, Viterbi School of Engineering, University of Southern California
³Human-Machine Collaboration, Bosch Research Pittsburgh

Abstract

Recent developments in pre-trained neural language modeling have led to leaps in accuracy on commonsense question-answering benchmarks. However, there is increasing concern that models overfit to specific tasks without learning to utilize external knowledge or perform general semantic reasoning. In contrast, zero-shot evaluations have shown promise as a more robust measure of a model’s general reasoning abilities. In this paper, we propose a novel neuro-symbolic framework for zero-shot question answering across commonsense tasks. Guided by a set of hypotheses, the framework studies how to transform various pre-existing knowledge resources into a form that is most effective for pre-training models. We vary the set of language models, training regimes, knowledge sources, and data generation strategies, and measure their impact across tasks. Extending on prior work, we devise and compare four constrained distractor-sampling strategies. We provide empirical results across five commonsense question-answering tasks with data generated from five external knowledge resources. We show that, while an individual knowledge graph is better suited for specific tasks, a global knowledge graph brings consistent gains across different tasks. In addition, both preserving the structure of the task as well as generating fair and informative questions help language models learn more effectively.

Introduction

Common sense is key to efficient communication in everyday situations, as it enables natural language understanding through contextual reasoning. Machine question answering (QA) benchmarks, like SocialIQA (Sap et al. 2019b) and PhysicalIQA (Bisk et al. 2020), are effective behavioral tests of commonsense reasoning in machines, each focusing on different capabilities. Answering a question in SocialIQA might require the knowledge that readers typically prefer heroes over villains in fantasy novels; whereas, in PhysicalIQA, the knowledge that metal stools can break windows, because windows are made of glass and metal is a more enduring material than glass. Although such tasks had been traditionally difficult for machines, recent developments in pre-trained neural language modeling have led to leaps in accuracy—closing the gap between human and machine performance to single-digit percentage points. However, due to increasing concern that large-capacity neural systems are modeling individual datasets, rather than learning how to perform logical reasoning or to utilize external knowledge effectively (Mitra et al. 2019), focus is shifting to alternative training and evaluation strategies. In particular, zero-shot evaluation shows promise as an efficient measure of model generalisability across tasks (Shwartz et al. 2020; Li et al. 2020). Here, models are trained and validated on task A, and tested on a different task B, without access to B’s training data or labels. This leads state-of-the-art models from individual tasks to falter, sometimes by as much as a 50% decrease in performance (Shwartz et al. 2020).

Repositories of commonsense knowledge, like ConceptNet (Speer, Chin, and Havasi 2017) and ATOMIC (Sap et al. 2019a), can be beneficial for commonsense QA, especially when little or no training data is available. Enriching the training data with ConceptNet and ATOMIC has been shown (Ma et al. 2019; Mitra et al. 2019) to improve accuracy on datasets derived from these graphs: CommonSenseQA (Talmor et al. 2019) and SocialIQA. Knowledge bases (KBs) can be used to generate question-answer pairs and distractors automatically, in order to test a model’s reasoning ability (Petroni et al. 2019; Richardson and Sabharwal 2019) or provide additional supervision (Ye et al. 2019; Yang et al. 2020). While KBs have been shown to help in a zero-shot setting recently (Banerjee and Baral 2020), no comprehensive study exists on the relation between various knowledge, its usage method, and neural models across zero-shot commonsense tasks. Moreover, while adversarial filtering techniques (Bras et al. 2020) improve the quality of a manually created question set, their impact on automatically generated questions from a variety of KBs has not been investigated yet.

In this paper, (1) we compile a set of hypotheses and design a novel neuro-symbolic framework that investigates the dependency between knowledge sources, question generation techniques, language model (LM) variants, and tasks. Our framework leverages a wide range of KBs, covering visual, social, and concept-based knowledge, to pre-train LMs for zero-shot evaluation on multiple-choice commonsense QA tasks.
sense QA tasks. (2) Recognizing that the aspect of question generation is especially understood, we expand on prior work to devise and test four distractor-sampling strategies for effective question generation. We analyze their impact on model performance across tasks, conditioned on model class and (pre-)training regime, and show that generating questions that are simultaneously fair and informative is difficult but beneficial for LM pre-training. (3) We determine that combination of knowledge graphs (KGs), data construction/training, and architectures that is most effective and can utilize appropriately rich contexts across five tasks. We observe that diversifying knowledge generally improves performance, under the condition of it being aligned with the task, and that preserving the structure of the task is desired. (4) We make our code and resulting datasets available to the community to facilitate scientific research in the area of knowledge-driven self-supervision for commonsense QA.

Related Work

Knowledge Injection

Strong performance on standard multiple-choice QA benchmarks, like SocialiQA and PhysicalIQA, has been achieved by fine-tuning a task-specific prediction layer, placed atop pre-trained LMs, such as BERT (Devlin et al. 2019), RoBERTa (Liu et al. 2019), and GPT (Radford et al. 2019). As shown by Ma et al. (2019) and Mitra et al. (2019), combining neural methods with structured background knowledge from ConceptNet, WordNet (Miller 1995), and ATOMIC works well for commonsense datasets that have been partially derived from these resources, such as SocialiQA and CommonsenseQA. Here, the structured knowledge, formalized as lexicalized task-targeted evidence paths, is injected into an LM, either via an attention mechanism (Bauer, Wang, and Bansal 2018) or through an auxiliary training objective (Xia, Wu, and Yan 2019). Graph and relation networks can also be used to score answer candidates, by inferring the graph structure with information from LMs (Lin et al. 2019; Zhong et al. 2019). Finally, complete KGs can be incorporated directly in training by introducing additional modeling objectives, to teach a model about general commonsense regardless of the task at hand (Peters et al. 2019; Levine et al. 2020; Liu et al. 2020; Zhang et al. 2019; Talmor et al. 2020). This line of work resembles our approach of including background knowledge in a general, task-agnostic way; however, it still relies on the task training data and has generally not been tested in a zero-shot regime.

Generating Commonsense Questions and Answers

Richardson and Sabharwal (2019) use the direct links in WordNet to generate question-answer pairs and leverage the resulting dataset to evaluate language models. Similarly, Petroni et al. (2019) prompt the skills of language models by sentences instead of questions, generated from sources like ConceptNet and SQuAD (Rajpurkar et al. 2016). Previous works have generated synthetic QA sets to facilitate model training, albeit mostly in addition to existing training data. Ye et al. (2019) proposed an ‘align-mask-select’ method to generate questions using ConceptNet and Wikipedia. Kocijan et al. (2019) constructed a large set of pronoun resolution questions using Wikipedia sentences. Yang et al. (2020) proposed to generate QA pair and distractors using generative models. Regarding zero-shot evaluation, the Self-Talk model of (Shwartz et al. 2020) generates clarification prompts based on a template prefix, which are leveraged to elicit knowledge from another language model. The additional knowledge is used jointly with original context and question to score each answer candidate. Given a task context, one can use COMET (Bosselut and Choi 2019), a generative model trained on commonsense KGs, to generate background knowledge statements, and to compute scores for each answer candidate based on the context, question, and generated knowledge. Banerjee and Baral (2020) pre-train the language model with three objective functions which aim to complete a knowledge triple given two of its elements. These representation learning functions are used jointly to compute the distance for each answer candidate. The ambition of our paper is to provide a comprehensive framework for these prior efforts on zero-shot learning with KGs. We cover a wider set of KGs, question generation techniques, and tasks, which allows us to systematically investigate the effect of using different KGs, generation methods, and techniques across tasks.

Zero-Shot QA Framework

Given a natural language question $Q$, and $n$ possible answers $A_1, \ldots, A_n$, the task of a system (human or machine) is to select the most probable single answer $A$. We refer to the remaining $n-1$ possible answers: $D_1, \ldots, D_{n-1}$ as distractors. In a zero-shot QA evaluation mode, the system has no access to the task training or development data.

We assume a setup where the system is pre-trained once and then applied across different tasks in a zero-shot manner. Our zero-shot evaluation framework covers variants of addressing this task by pre-training an LM on an artificial QA set, created from KG data. We investigate the dependency between different tasks, sources of knowledge, question generation strategies, LM techniques, and training regimes. We describe each of these dimensions, in turn.

Synthetic QA Generation

We generate questions, answers, and distractor options from five KGs: ATOMIC, ConceptNet, WordNet, VisualGenome (Krishna et al. 2017), and Wikidata (Vrandečić and Krötzsch 2014), found in the Commonsense Knowledge Graph (CSKG) (Ilievski et al. 2020). Notably, ATOMIC differs from the other knowledge sources in two ways: 1) its relations have a different focus than those of the other sources; and 2) its nodes have longer labels and are formalized as templates. Due to these considerations, we prepare two sets of question-answer sets: one based on ATOMIC and one based on the remaining four knowledge sources. Figure 1 illustrates our question generation pipeline.

Data partitions ATOMIC expresses pre- and post-states for events and their participants with nine relations. Its head
nodes are events, whereas the tail nodes are either events or attributes. Its nodes have two particularities: 1) irrelevant parts of the node text are replaced with blanks (‘_’); and 2) references to fictional agents are indicated with special tokens (e.g., PersonX). We follow the SocialiQA’s ATOMIC train/dev/test splits, to ensure that the facts of the dev and test partitions are excluded in training.

Our second partition, CWWV, covers three other KGs in CSKG that express commonsense facts between concepts: ConceptNet, WordNet, and Wikidata. We use them jointly to generate questions, and we enrich them with additional distractors from VisualGenome. Treating these four sources as a single one is enabled by their CSKG mapping to a single set of relations, defined by ConceptNet. We focus on 14 semantic relations that are grounded on strong psycholinguistic and pragmatic evidence (Murphy 2003), like /r/Causes and /r/HasPrerequisite. Since there is no pre-defined train/dev/test split for CWWV, we randomly sample 5% of generated questions as development set, while the other 95% are used for training.

Generating questions and answers If a triple \((h, r, t)\) has an associated sentence, we directly employ it for question generation; otherwise, we generate a sentence in a lexicalization step, using a set of pre-defined templates. Next, we generate the question \(Q\) by removing the tail of the sentence, and extract this tail as the correct answer, \(A\). Here, we ensure that there is no token overlap between the head and the correct answer. For ATOMIC, we: 1) compare the keyword tokens instead of all tokens, in order to avoid stopwords; and 2) the agent templates (e.g., ‘PersonX’) are replaced with randomly sampled gender-neutral names from a pre-defined set. For CWWV, we filter out questions where either the head or the tail are not common concepts or they are named entities. We use corpus frequency as a proxy for commonness\(^1\) while named entities are filtered by removing all concepts whose labels start with a capital letter.

Generating negative samples (distractors) We seek to generate distractor options that satisfy two criteria: informativeness and fairness. Namely, a good distractor has semantic relatedness with the context (informative), while being relatively easy to discriminate from the correct answer (fair). We create the pool of distractors \(D\) for every sample as follows: 1. The distractor candidates are the tails of knowledge triples \((h', r', t')\) with the same relation \(r' = r\), randomly sampled from the KGs. This would ensure that the distractors can fill the same semantic role as the correct answer. 2. The head \(h'\) of the sampled triples does not have non-stop word overlap with \(h\). The distractor tail \(t'\) is not part of the correct answer set, i.e., there exist no triples, \((h, r, t')\). Considering the example in Figure 1 the triple \((\text{gaining weight}, \text{CausesDesire}, \text{change appearance})\) will be filtered out by rule (1), \((\text{losing weight}, \text{UsedFor}, \text{feeling better})\) will be ruled out by both (2) and (3), and \((\text{relaxing}, \text{UsedFor}, \text{feeling better})\) will be ruled out by (3). Here, we replace any references to fictional ATOMIC agents in the distractors with the same names used in the question. We then randomly select two distractors \((D_1, D_2)\) from \(D\). We refer to this distractor pooling strategy as random, and propose three alternative strategies in the next Section.

Example questions with each partition are shown in Table 1. For ATOMIC, this procedure generates 535K QA pairs for training and 60K for development. For CWWV, the training set contains 157K and the dev set has 8K QA pairs.

Distractor Sampling

Existing data generation procedures are likely to introduce annotation artifacts in datasets [Zellers et al. 2019; Sakaguchi et al. 2019]. Models may exploit these artifacts to achieve spuriously strong performance during training, at the expense of degradation in robustness. To generate more challenging QA pairs from KGs and to alleviate potential biases in our synthetic sets, we test two other distractor sampling strategies in addition to the random strategy: 1) we select distractors that are as similar as possible to the answer, while being under a certain threshold (adv-answer); and 2) we select distractors that are as similar as possible to the question, while being under a certain threshold (adv-question). Here we define similarity of two nodes to be their proximity in the embedding space, measured by cosine similarity. The intuition is that, by generating more challenging QA pairs for the models, we could achieve better generalization across tasks. We use the RoBERTa sentence embedding model (Reimers and Gurevych 2020) to compute embeddings for all KG nodes. For these two strategies, we set an upper bound on the similarity score to avoid unfair distractors, i.e., paraphrases of the correct answer.

Sample filtering Besides these distractor sampling strategies, we test another condition (3), where we select the dis-

---

1. https://pypi.org/project/wordfreq/ (Accessed 9 Sept. 2020)

Table 1: Generated questions from ATOMIC (top) and CWWV (bottom). (*) denotes the correct answer.

| Question: Losing weight is for? | A: Being healthier. (*) | A2: Embedded software. |
|--------------------------------|-------------------------|------------------------|
| Question: Robin takes the fifth. As a result, Robin wanted to go to the cinema. | A1: Go to the cinema. | A2: Withhold information. (*); A3: Hear what they think. |

| Question: Buying things in store. | A1: Embedded software. | A2: Being healthier. (*) |
|----------------------------------|------------------------|-------------------------|
| Question: Having a birthday.      | A1: Having a birthday. | A2: Buying things at store. |
| Question: Drawing a picture.      | A1: Drawing a picture. | A2: Embedded software. |
| Question: Embedded software.      | A1: Embedded software. | A2: Buying things at store. |
Language Models

We consider 2 types of language models: auto-regressive language models and masked language models (MLM). Specifically, we use GPT-2 and RoBERTa to select the best answer candidate. Given a context C, a question Q, and a list of answer options (A_1, A_2, ...), we concatenate C and Q with each answer option to build input sequences (T_1, T_2, ...). We also use templates to convert a sequence T into a natural language sentence following (Shwartz et al. 2020). For example, we transform the sequence: [C]: What will X want to do next? [A_1] into: [C], as a result, X want to [A_1]. The score S for the resulting sequence using an auto-regressive LM is computed as follows:

\[
S_{LM} (T) = - \frac{1}{n} \sum_{i=1}^{n} \log P (t_i | t_1 \ldots t_{i-1})
\]  

where n is the number of tokens in the sequence and P is the conditional probability provided by the LM. To evaluate MLMs, we mask out one token at a time and compute its loss (Zhou et al. 2020). We repeat this process for every token in the sequence. The final MLM score is:

\[
S_{MLM} (T) = - \frac{1}{n} \sum_{i=1}^{n} \log P (t_i | t_1 \ldots t_{i-1}, t_{t+1} \ldots)
\]  

The predicted option is the one with the lowest score.

LM Fine-tuning In the typical model architecture for fine-tuning LM for multiple-choice tasks, a linear layer is added on top of the LM encoder to predict the answer. The model inputs are separated by a model-specific delimiter. However, as this architecture introduces randomly initialized parameters, it may not be able to fully utilize the pre-trained weights (Lamborino et al. 2020). Instead, we re-use the GPT-2 and RoBERTa with LM head for fine-tuning. By keeping the model intact, we can reuse the same converting templates and scoring functions. To train the model, given the scores computed for each answer candidate S_1, S_2, ... S_m, we use the marginal ranking (MR) loss defined as:

\[
\mathcal{L} = \frac{1}{m} \sum_{i=1}^{m} \max (0, \eta - S_y + S_i)
\]  

Here, \(\eta\) represents the margin and \(y\) is the index of the correct answer. For a MLM model, the computation cost for the scoring function scales quadratically with the input length. To make the training more efficient, we only mask out non-stop tokens in the head and tail nodes.

Training Regimes In order to disentangle the contribution of the KGs from the structure of the QA pairs, we consider different training methods for augmentation of language models with KGs. Specifically, we compare marginal ranking (MR) training with masked language modeling (MLM) training. For MLM, we directly concatenate the question and the correct answer in our synthetic QA set and then train RoBERTa on these sentences using the MLM objective.

Tasks We select commonsense tasks based on two criteria. Firstly, we strive to cover a diverse set of tasks, both in terms of their format (question answering, pronoun resolution, natural language inference), as well as their type of knowledge (e.g., social or physical knowledge). Secondly, we prefer larger task datasets that are manually constructed. For this reason, we do not include datasets like COPA (Gordon, Kozareva, and Roemmele 2012), or HellaSwag (Zellers et al. 2019). We opt for the following five task datasets:

1. Abductive NLI (aNLI) (Bhagavatula et al. 2019) is posed as a natural language inference task. Given the beginning and the ending of a story, the task is to choose the more plausible hypothesis out of two options. The dataset consists of nearly 170k entries.

2. CommonsenseQA (CSQA) (Talmor et al. 2019) evaluates a broad range of common sense aspects. Each entry contains a question and 5 answer candidates. The questions are crowdsourced based on a subgraph from ConceptNet. The answer candidates combine ConceptNet nodes with additional crowdsourced distractors.

3. PhysicalIQA (PIQA) (Bisk et al. 2020) is a two-choice question answering dataset which focuses on physical reasoning. Given a question, the system (or human) is asked to pick the more plausible out of two possible continuations.

4. SocialIQA (SIQA) (Sap et al. 2019b) is a question-answering dataset which requires reasoning about social interactions. Each entry contains a context, a question, and 3 answer candidates. The context is derived from the ATOMIC knowledge graph, the questions are generated based on nine templates (corresponding to the relations in ATOMIC), and the answers are crowdsourced.

5. WinoGrande (WG) (Sakaguchi et al. 2019) contains 44 thousand pronoun resolution problems. Each entry consists of a context description with an emphasized pronoun, and two options are offered as its possible references.
Table 2: Zero-shot evaluation results with different combinations of models and knowledge sources, across five commonsense tasks. CSKG represent the combination of ATOMIC and CWWV. We run our experiments three times with different seeds and report average accuracy with 95% confidence interval.

| Model                       | KG   | aNLI | CSQA | PIQA | SIQA | WG  |
|-----------------------------|------|------|------|------|------|-----|
| Majority                    | -    | 50.8 | 20.9 | 50.5 | 33.6 | 50.4|
| GPT2-L                      | -    | 56.5 | 41.4 | 68.9 | 44.6 | 53.2|
| RoBERTa-L                   | -    | 65.5 | 45.0 | 67.6 | 47.3 | 57.5|
| Self-talk (Shwartz et al. 2020) | -    | -    | 32.4 | 70.2 | 46.2 | 54.7|
| COMET-CGA (Bosselut and Choi 2019) | ATOMIC | -    | -    | -    | 49.6 | -   |
| SMLM (Banerjee and Baral 2020) | ATOMIC | -    | -    | 48.5 | -    | -   |
| GPT2-L (MR)                 | ATOMIC | 50.2±(0.3) | 48.0±(0.9) | 67.5±(0.7) | 53.5±(0.4) | 54.7±(0.6) |
| GPT2-L (MR)                 | CWWV | 58.3±(0.4) | 46.2±(1.0) | 68.6±(0.7) | 48.0±(0.7) | 52.8±(0.9) |
| GPT2-L (MR)                 | CSKG | 50.0±(0.5) | 48.6±(1.0) | 68.6±(0.9) | 53.3±(0.5) | 54.1±(0.5) |
| RoBERTa-L (MR)              | ATOMIC | 70.8±(1.2) | 64.2±(0.7) | 72.1±(0.5) | 63.1±(1.5) | 59.6±(0.3) |
| RoBERTa-L (MR)              | CWWV | 70.0±(0.3) | 67.9±(0.8) | 72.0±(0.7) | 54.8±(1.2) | 59.4±(0.5) |
| RoBERTa-L (MR)              | CSKG | 70.5±(0.2) | 67.4±(0.8) | 72.4±(0.4) | 63.2±(0.7) | 60.9±(0.8) |
| RoBERTa-L (supervised)      | -    | 85.6 | 78.5 | 79.2 | 76.6 | 79.3|
| Human                       | -    | 91.4 | 88.9 | 94.9 | 86.9 | 94.1|

**Experimental Setup**

**Baselines**

We compare our results with the following baselines. Majority answers each question with the most frequent option in the entire dataset. ‘Vanilla’ versions of the language models are used in order to understand the impact of further tuning. Here we directly use the LMs to score the QA pairs without any finetuning. We also show the results of other unsupervised systems that leverage KGs: Self-talk, COMET-CGA, and SMLM. To indicate the upper bound of this work, we include results of a supervised fine-tuned RoBERTa system and of human evaluation.

**Implementation**

For the LM baselines, we directly load the weights from the Transformers library (Wolf et al. 2019) and evaluate on the downstream tasks. The finetuned LMs are trained for a single epoch on our synthetic QA set. For Adv-filter, we train the models for 5 epochs to compensate for less training data. We use our synthetic dev set to select the best model. We describe other hyper-parameters used and computing infrastructure in the appendix.

**Hypotheses**

Based on individual prior findings and understanding of different components of our framework, we put forward a set of hypotheses which will be validated in our experiments:

H1 *RoBERTa would have better performance than GPT-2.* This is in line with prior findings that RoBERTa has the advantage of bi-directional context (Zhou et al. 2020).

H2 *Pre-training a language model with artificially created question-answer sets enhances zero-shot performance.* This is also supported in previous study about unsupervised QA (Li et al. 2020).

H3 *The impact of more knowledge depends on the alignment between KGs and the task,* partial evidence for which is provided by (Ma et al. 2019) [Mitra et al. 2019].

H4 *Adding diverse knowledge (from different KGs) improves performance.* This is the initial motivation behind the creation of CSKG (Ilievski et al. 2020), but has not been investigated in detail.

H5 *When selecting negative samples for a question, it helps to use an adversarial strategy that ensures the question is not trivial for a language model.* H5 is inspired by adversarial filtering, which has not been investigated in detail for automatically-generated questions and across KGs.

H6 *Preserving the task structure when generating synthetic data leads to better accuracy.* This is implicitly assumed in prior data augmentation work (Kocijan et al. 2019).

H7 *The automatically created questions are notably easier for humans than they are for machines - a general assumption made by commonsense task creators and typically correct for any existing, human-generated benchmark.*

**Results**

We evaluate various combinations of: knowledge sources, question generation strategies, LMs, training regimes, and tasks. We use accuracy as a metric. All our experiments are performed in a zero-shot setting, i.e., the models do not leverage the official training data of the task. We report results on the dev sets of these tasks, as the official test sets are not publicly available. We note that, since we did not use the tasks’ dev sets for hyperparameter tuning or checkpoint selection, the dev sets can be used effectively as test sets.

**Main Results**

Table 2 shows that GPT-2 and RoBERTa outperform the majority baseline by a large margin on all tasks, indicating that...
the LMs have already learned relevant knowledge during pretraining. Despite being a smaller model, RoBERTa outperforms GPT-2 on 4 out of 5 tasks without pretraining, and on all tasks when pretraining over different synthetic QA sets. This shows the advantage of leveraging bi-directional context, and confirms our hypothesis H1. As expected (H2), training RoBERTa on our ATOMIC or CWWV synthetic sets brings notable performance gain on all 5 tasks. We observe that models trained on ATOMIC sets have a large advantage on SIQA compared to models trained on CWWV, while CWWV brings advantage on the CSQA task. This is not surprising as these two tasks are derived from ConceptNet and ATOMIC, respectively. The difference between ATOMIC and CWWV on the remaining three tasks is relatively small. This supports our hypothesis H3: knowledge alignment is crucial for obtaining better performance.

Training on the combined question set (CSKG) is mostly able to retain the best of both partitions. Training on CSKG leads to best performance on three out of five tasks, showing that a global commonsense resource is able to bring consistent gain across different tasks. This supports our hypothesis H4: adding more diverse knowledge is beneficial for language models. Finally, even with this knowledge, we recognize that there is still a large gap between our model’s accuracy and that of the supervised RoBERTa model.

Comparison of QA Generation Strategies

Table 3 shows the results with different sampling strategies, thus addressing H5. The best performing adversarial algorithm, Adv-answer, yields comparable accuracy to the random strategy, revealing that distractors sampled with a more sophisticated strategy are not necessarily more informative for the LMs. Adv-question and Adv-filter typically lead to declines in accuracy. Considering Adv-question, this could be due to the similarity of the distractors to the question, which might guide the model to learn to pick the most dissimilar candidate as the correct answer, which is an artifact of our question generation and cannot be expected to work well for downstream tasks. Our manual inspection of the remaining questions preferred by Adv-filter indicates that many questions are unfair, as some distractors are also correct answers, which is a consequence of the incompleteness of the KGs. Adv-filter prioritizes these questions as they are “difficult” for LMs, however, training on them might teach the LM incorrect knowledge and harm downstream accuracy.

Comparison of Training Regimes

Table 4 presents results with two different training regimes. In comparison to the baseline without additional training, MLM training on ATOMIC only improves on the SIQA task, and harms on the rest. With CWWV, it brings large gain on CSQA and small improvements on SIQA and WG. At the same time, marginal ranking training on either question set consistently outperforms MLM training by a large margin, suggesting that preserving the task structure is beneficial in addition to the question content and validating H6.

Difficulty of the Synthetic QA Sets

Ideally, the generated question-answer pairs should be challenging for the models but easy for humans to solve (H7). Here, we probe this hypothesis by assessing the difficulty of our synthetic QA sets. A total of five researchers were asked to first provide the correct answer, then rate the question difficulty. For the latter, the annotator chose between easy, moderate, hard, or non-sensical - as a guideline, nonsensical questions have unfair distractors and cannot be easily understood. Following this procedure, we obtained three judgments for each question.

The inter-annotator agreement on selecting the correct answer is 0.62 using Fleiss Kappa score, which is substantial agreement. The Krippendorf alpha (Krippendorf 2004) for rating question difficulty is 0.35, which is fair agreement. The results of the baseline LMs and human performance (Table 5) show that the ATOMIC subset presents a harder challenge for both models, as well as for humans. Overall, the results support our hypothesis H7: the synthetic questions are relatively easy for humans to solve and much harder for models. However, the annotation pointed to several directions for improving the synthetic QA sets. A number of questions generated from ATOMIC are ungrammatical, which makes them harder to understand, while some questions from CWWV were rated as unfair. For example, all answer options for the question A person can be valid: (a) cost several thousand dollars (b) expressing boredom (c) check snow level. As discussed earlier, this is due to the incompleteness of our KGs, and the current lack of understanding on how to generate fair, yet informative, distractors.
Discussions
Towards a Commonsense Service
The overarching pursuit of this paper is to understand whether generating artificial QA sets with KGs improves the zero-shot QA performance of LMs. We observe a consistent leap in accuracy across tasks, LMs, knowledge sources, and question generation strategies. While these accuracies are notably above supervised LM accuracies, they might further improve by architectural improvements of the LMs, knowledge sources with wider coverage and stronger semantics, and well-tuned scoring functions. In addition, despite its complexity and diversity, commonsense knowledge (unlike knowledge on entities and events, which changes rapidly) is largely static and evolves slowly over time, thus making the dataset-specific finetuning unnecessary in theory. A natural question arises: can we build a sufficiently reliable, general commonsense service, by pretraining a LM on a rich set of questions covering a wide spectrum of knowledge types?

Impact of Knowledge
In general, we observed that using knowledge from a wider set of sources is beneficial. However, on aNLI and CommonSenseQA, the best accuracy was obtained with a subset of all questions. This could be due to the kinds of knowledge covered: 1. aNLI focuses on expectations of agents and events, for which ATOMIC is directly useful, whereas the other KGs might be delusive; 2. CommonSenseQA mostly requires knowledge about properties of objects (e.g., function or appearance) which is the focus of ConceptNet, Wikidata, and VisualGenome, but not of ATOMIC. This indicates a tension between H3 and H4: while more knowledge often helps, it might not be the case when the task and the knowledge are not well-aligned. Our current understanding of the dimensions of commonsense knowledge in knowledge sources and benchmarks is limited, and would benefit from further study.

Generating Fair and Informative Questions
An alternative explanation for this result is indicated by our human evaluation: not all automatically generated questions are fair and a subset has more than one correct answer. Besides being a disadvantage of automatic question generation, this finding points to a more substantial challenge: generating fair and informative multiple-choice questions is not well-understood today. Adversarial strategies generate more plausible candidates than random sampling, making the task less fair; however, fully relying on random sampling would result in generating distractors that are trivially discernible from the correct option. Striking a balance between fairness and informativeness, then, seems to be a core requirement for multiple-choice question generation: our empirical evidence suggests that such balance could be achieved by a mixed approach, where part of distractors is generated randomly, and part by adopting suitable adversarial strategies.

Conclusions
While zero-shot QA is gaining focus, no study so far has investigated systematically the dependency between the task, the technique, and any additional knowledge used. To address this gap, this paper proposed a framework for zero-shot question answering by pretraining language models on artificial data created based on knowledge graphs. We studied the interplay between five knowledge sources, four data generation strategies, two language models, two training regimes, and five tasks. We put forward seven hypotheses, which guided our experiments. We observed that language models benefit from pretraining on artificially created QA sets. Similar to prior work, we observed that the best performance is obtained by the RoBERTa model. Furthermore, combining knowledge sources leads to best performance, indicating that diverse relevant knowledge is desired for pretraining. However, this is conditioned on a good alignment between knowledge and the task. The training regime had a role too: marginal ranking is superior to vanilla language modeling, as it preserves better the structure of the task. Our analysis and human evaluation indicated that the generated questions were typically easier for humans than for language models, which is appropriate for commonsense questions.

Yet, the human evaluation revealed that a notable portion of the questions is nonsensical or difficult, hinting that automatically generating high-quality, informative commonsense questions is non-trivial and should be revised in subsequent work. Future work should also investigate the impact of this approach on knowledge injection systems (Ma et al., 2019) and graph relational networks (Lin et al., 2019). It should also consider: (1) other, less structured knowledge sources, like WikiHow; (2) different distractor sampling strategies, e.g., based on iterative sampling (Niu et al., 2020); and (3) additional LM scoring functions, e.g., based on scoring sequences of tokens (Tamborrino et al., 2020).

Table 4: Comparison between MLM and MR training.

| RoBERTa-L | Train | aNLI | CSQA | PIQA | SIQA | WG |
|-----------|-------|------|------|------|------|----|
| baseline  | -     | 65.5 | 45.0 | 67.6 | 47.3 | 57.5 |
| + ATOMIC  | MLM   | 62.9 | 43.8 | 65.8 | 53.9 | 55.5 |
| + ATOMIC  | MR    | 70.8 | 64.2 | 72.1 | 63.1 | 59.6 |
| + CWWV    | MLM   | 65.3 | 57.3 | 67.2 | 49.3 | 59.4 |
| + CWWV    | MR    | 70.0 | 67.9 | 72.0 | 54.8 | 59.4 |

Table 5: LM and human accuracy on our synthetic QA sets.

| Model     | ATOMIC | CWWV |
|-----------|--------|------|
| GPT2-L    | 43.2   | 69.5 |
| RoBERTa-L | 45.9   | 64.5 |
| Human     | 78.0   | 80.7 |

Footnote:
1 For example, scoring sequences of tokens by a language model might improve the performance of LMs (Tamborrino et al., 2020).

Discussion
Towards a Commonsense Service
The overarching pursuit of this paper is to understand whether generating artificial QA sets with KGs improves the zero-shot QA performance of LMs. We observe a consistent leap in accuracy across tasks, LMs, knowledge sources, and question generation strategies. While these accuracies are notably above supervised LM accuracies, they might further improve by architectural improvements of the LMs, knowledge sources with wider coverage and stronger semantics, and well-tuned scoring functions. In addition, despite its complexity and diversity, commonsense knowledge (unlike knowledge on entities and events, which changes rapidly) is largely static and evolves slowly over time, thus making the dataset-specific finetuning unnecessary in theory. A natural question arises: can we build a sufficiently reliable, general commonsense service, by pretraining a LM on a rich set of questions covering a wide spectrum of knowledge types?

Impact of Knowledge
In general, we observed that using knowledge from a wider set of sources is beneficial. However, on aNLI and CommonSenseQA, the best accuracy was obtained with a subset of all questions. This could be due to the kinds of knowledge covered: 1. aNLI focuses on expectations of agents and events, for which ATOMIC is directly useful, whereas the other KGs might be delusive; 2. CommonSenseQA mostly requires knowledge about properties of objects (e.g., function or appearance) which is the focus of ConceptNet, Wikidata, and VisualGenome, but not of ATOMIC. This indicates a tension between H3 and H4: while more knowledge often helps, it might not be the case when the task and the knowledge are not well-aligned. Our current understanding of the dimensions of commonsense knowledge in knowledge sources and benchmarks is limited, and would benefit from further study.

Generating Fair and Informative Questions
An alternative explanation for this result is indicated by our human evaluation: not all automatically generated questions are fair and a subset has more than one correct answer. Besides being a disadvantage of automatic question generation, this finding points to a more substantial challenge: generating fair and informative multiple-choice questions is not well-understood today. Adversarial strategies generate more plausible candidates than random sampling, making the task less fair; however, fully relying on random sampling would result in generating distractors that are trivially discernible from the correct option. Striking a balance between fairness and informativeness, then, seems to be a core requirement for multiple-choice question generation: our empirical evidence suggests that such balance could be achieved by a mixed approach, where part of distractors is generated randomly, and part by adopting suitable adversarial strategies.

Conclusions
While zero-shot QA is gaining focus, no study so far has investigated systematically the dependency between the task, the technique, and any additional knowledge used. To address this gap, this paper proposed a framework for zero-shot question answering by pretraining language models on artificial data created based on knowledge graphs. We studied the interplay between five knowledge sources, four data generation strategies, two language models, two training regimes, and five tasks. We put forward seven hypotheses, which guided our experiments. We observed that language models benefit from pretraining on artificially created QA sets. Similar to prior work, we observed that the best performance is obtained by the RoBERTa model. Furthermore, combining knowledge sources leads to best performance, indicating that diverse relevant knowledge is desired for pretraining. However, this is conditioned on a good alignment between knowledge and the task. The training regime had a role too: marginal ranking is superior to vanilla language modeling, as it preserved better the structure of the task. Our analysis and human evaluation indicated that the generated questions were typically easier for humans than for language models, which is appropriate for commonsense questions.

Yet, the human evaluation revealed that a notable portion of the questions is nonsensical or difficult, hinting that automatically generating high-quality, informative commonsense questions is non-trivial and should be revised in subsequent work. Future work should also investigate the impact of this approach on knowledge injection systems (Ma et al., 2019) and graph relational networks (Lin et al., 2019). It should also consider: (1) other, less structured knowledge sources, like WikiHow; (2) different distractor sampling strategies, e.g., based on iterative sampling (Niu et al., 2020); and (3) additional LM scoring functions, e.g., based on scoring sequences of tokens (Tamborrino et al., 2020).

Footnote:
1 For example, scoring sequences of tokens by a language model might improve the performance of LMs (Tamborrino et al., 2020).
Sap, M.; Rashkin, H.; Chen, D.; Le Bras, R.; and Choi, Y. 2019b. Social IQa: Commonsense Reasoning about Social Interactions. In Proc. of EMNLP-IJCNLP, 4463–4473.

Shwartz, V.; West, P.; Bras, R. L.; Bhagavatula, C.; and Choi, Y. 2020. Unsupervised Commonsense Question Answering with Self-Talk. ArXiv abs/2004.05483.

Speer, R.; Chin, J.; and Havasi, C. 2017. ConceptNet 5.5: An Open Multilingual Graph of General Knowledge. In Proc. of AAAI, AAAI’17, 4444–4451.

Talmor, A.; Herzig, J.; Lourie, N.; and Berant, J. 2019. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In Proc. of NAACL, 4149–4158.

Talmor, A.; Tafjord, O.; Clark, P.; Goldberg, Y.; and Berant, J. 2020. Teaching Pre-Trained Models to Systematically Reason Over Implicit Knowledge. arXiv preprint arXiv:2006.06609.

Tamborrino, A.; Pellicanò, N.; Pannier, B.; Voitot, P.; and Naudin, L. 2020. Pre-training Is (Almost) All You Need: An Application to Commonsense Reasoning. In Proc. of ACL, 3878–3887.

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

Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; and Brew, J. 2019. HuggingFace’s Transformers: State-of-the-art Natural Language Processing. ArXiv abs/1910.03771.

Xia, J.; Wu, C.; and Yan, M. 2019. Incorporating Relation Knowledge into Commonsense Reading Comprehension with Multi-Task Learning. In Proc. of CIKM, CIKM ’19, 2393–2396. ISBN 9781450369763.

Yang, Y.; Malaviya, C.; Fernandez, J.; Swayamdipta, S.; Bras, R. L.; Wang, J.; Bhagavatula, C.; Choi, Y.; and Downey, D. 2020. G-DAUG: Generative Data Augmentation for Commonsense Reasoning. ArXiv abs/2004.11546.

Ye, Z.-X.; Chen, Q.; Wang, W.; and Ling, Z.-H. 2019. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models. arXiv preprint arXiv:1908.06725.

Zellers, R.; Holtzman, A.; Bisk, Y.; Farhadi, A.; and Choi, Y. 2019. HellaSwag: Can a Machine Really Finish Your Sentence? In Proc. of ACL, 4791–4800.

Zhang, Z.; Han, X.; Liu, Z.; Jiang, X.; Sun, M.; and Liu, Q. 2019. ERNIE: Enhanced Language Representation with Informative Entities. In Proc. of ACL, 1441–1451.

Zhong, W.; Tang, D.; Duan, N.; Zhou, M.; Wang, J.; and Yin, J. 2019. Improving Question Answering by Commonsense-Based Pre-training.

Zhou, X.; Zhang, Y.; Cui, L.; and Huang, D. 2020. Evaluating Commonsense in Pre-trained Language Models. In AAAI.
Hyperparameters

Model training
In all experiments that involve training, we used learning rate $1e^{-5}$, batch size 32, max sequence length 128, weight decay 0.01, adam epsilon $1e^{-6}$, $\beta_1 = 0.9$, $\beta_2 = 0.98$ and warm-up proportion 0.05, margin 1.0. For MLM training with RoBERTa, We only mask non-stop words in head or tail and we set the masking probability as 0.5 for ATOMIC and 0.3 for $\text{CWWV}$. Tuned parameters
We also tested smaller margins for training (0.2, 0.5) and we observe that margin 1.0 works slightly better.

Distractor sampling
For both Adv-answer and Adv-question, we set the distractor similarity upper bound to be 0.6 for $\text{CWWV}$ and 0.4 for ATOMIC. We observe that with higher upper bound, the likelihood of creating unfair questions increases significantly.

AFLite
In our experiments, we set $N=64$, $\tau=0.75$, $k_1=1/50$ of $|\text{Trn}|$, $k_2=1/50$ of $|\text{Dev}|$ and $O = 1/5$ of $|\text{Trn}|$.

Algorithm 1: AFLite

| Algorithm 1: AFLite |
|-------------------|
| **Input:** $\text{Trn}$, $\text{Dev}$, ensemble size $N$, cutoff sizes $k_1, k_2$, threshold $\tau$, target size $O$ |
| **while** $|\text{Trn}| > O$ **do** |
|  **for** $s$ in $\text{Trn} + \text{Dev}$ **do** |
|  | Initialize Prediction $P(s) = \emptyset$ |
|  **end** |
|  **for** $i=1, \ldots, N$ **do** |
|  | Random Partition $\text{Trn}$ into $U$, $V$ s.t. $|U| = O$ |
|  | Train a linear classifier $F$ on $U$ |
|  | **for** $s$ in $V + \text{Dev}$ **do** |
|  |  | Add $F(s)$ in $P(s)$ |
|  **end** |
|  **end** |
|  **for** $s$ in $\text{Trn} + \text{Dev}$ **do** |
|  | $\text{Acc}(s) = \frac{|p \in P(s) \text{ s.t. } p = y|}{|P(s)|}$ |
|  **end** |
| Select top-$k_1$ samples $S_1$ in $\text{Trn}$ s.t. $\text{Acc}(s) > \tau$ |
| Select top-$k_2$ samples $S_2$ in $\text{Dev}$ s.t. $\text{Acc}(s) > \tau$ |
| $\text{Trn} = \text{Trn} - S_1$ |
| $\text{Dev} = \text{Dev} - S_2$ |
| **if** $|S_1| < k_1$ **then** break; |
| **Output:** filtered sets $\text{Trn}'$ and $\text{Dev}'$ |

Computing resources
We run our experiments on servers with Intel(R) Core(TM) i7-7820X CPU @ 3.60GHz (1 CPU, 8 physical cores per CPU, total 16 logical CPU units) and with 125GB RAM. For GPUs, We used Nvidia RTX 2080Ti and Nvidia Titan RTX. For libraries, we used Pytorch 1.2.0, transformers 3.0.2 and sentence-transformer 0.3.4.