Commonsense Frame Completion and its Probabilistic Evaluation

Anonymous ACL submission

Abstract

Commonsense knowledge is critical to achieving artificial general intelligence. Large language models have demonstrated impressive performance on commonsense tasks, however these tasks are often posed as multiple-choice questions, allowing models to exploit systematic biases (Li et al., 2021). Commonsense is also inherently probabilistic; a plumber could repair a sink in a kitchen or a bathroom, or even a basement, although the former answers are more probable. Existing tasks do not capture the probabilistic nature of common sense. To this end we present commonsense frame completion (CFC), a new generative task which evaluates common sense via multiple open-ended generations. We also propose a method of probabilistic evaluation which strongly correlates with human judgements. Humans drastically outperform strong language model baselines on our dataset, indicating this approach is both a challenging and useful evaluation of machine common sense.

1 Introduction

Commonsense reasoning has become increasingly important for AI models in recent years. In NLP, the recent progress of large language models has demonstrated impressive performance on multiple evaluation benchmarks (Brown et al., 2020; Wang and Komatsuzaki, 2021), including many benchmarks that specifically measure the models' commonsense reasoning ability (Lin et al., 2020c; Sakaguchi et al., 2020; Sap* et al., 2019; Boratko* et al., 2020), with some achieving close to human level performance, leading some to question whether commonsense is solved. A deeper analysis of these models indicates they still make naïve commonsense errors (Lin et al., 2020a), thus the first question which must be addressed is how we can best evaluate commonsense knowledge.

Most existing commonsense evaluations are framed as multiple-choice question answering tasks (Talmor et al., 2019; Sap* et al., 2019; Huang et al., 2019; Bhagavatula et al., 2020). This evaluation requires the model to choose the right answer from a list of candidates, including the correct choice ("positive") and a few incorrect ones ("negatives"). High accuracy in this evaluation is misleading as the candidate answer sets are unrealistically small, and generating hard negatives is challenging (Zellers et al., 2018, 2019). Recent benchmarks attempt to overcome this limitation via generative commonsense evaluation (Lin et al., 2020b), which is more challenging as it can be viewed as multiple-choice question answering with practically unlimited choices.

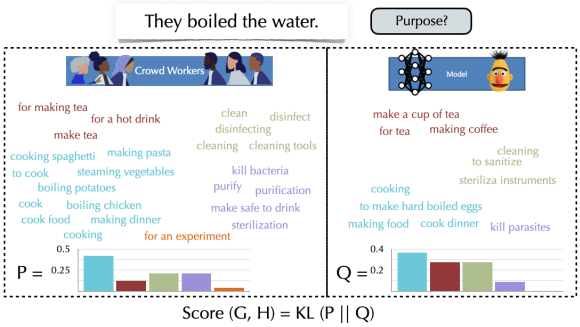


Figure 1: Example from the CFC dataset. Given a short sentence and a slot of interest (in this case, the purpose of boiling water). Human annotators provide ground-truth answer sets G, and model prediction is denoted as answer sets H. Each example in the dataset contains multiple current answers. To evaluate these answers as a probability distribution, we construct a categorical distribution for each answer set, and we calculate KL Divergence between these distributions (details in Section 4)

While generative evaluation avoids the difficulty of generating hard negatives, it does not reflect the fact that there are often multiple correct answers, nor does it incorporate the probabilistic nature of language semantics and commonsense knowledge (Erk, 2022). For example, given a sentence "The plumber is fixing the sink", we can infer using our common sense that the most probable lo-

cations include the kitchen and the bathroom, and with some lower probability perhaps a basement or utility closet. Inspired by the American TV show FAMILY-FEUD, Boratko* et al. (2020) addressed the issue of multiple correct answers by sampling 100 answers from human annotators to prototypical questions, eg. "Name something that people usually do before they leave the house for work," and proposed a rank-based evaluation.

In this work, we take the perspective that *commonsense knowledge is an implicit probability distribution over missing information in a context*. Emphasizing the implicit nature of common sense in a given context enhances the utility of our proposed task for downstream applications, such as home assistants, where the need for common sense is very rarely *explicit*. For example, a home assistant providing cooking directions should only implicitly be aware that "boil the water and add the spaghetti" requires the water to be in a container. Explicitly instructing a human with every minute detail would render the assistant useless, and thus it is paramount that the assistant understand what information can be implicitly inferred from context. Leveraging a probabilistic evaluation also emphasizes the uncertain nature of common sense - for example, the water may be heated on a stove, but it also may be heated using a kettle. This distribution also changes with respect to context - for example, consider how the implicit distribution would change if the instruction was "boil the water and add 4-methoxy-3-buten-2-one".

In this work, we propose the task of commonsense frame completion (CFC), in which models are provided with a context sentence and asked to generate potential values for a missing information or "slot-fillers" for the semantic frame in the sentence, where potential slots include "time", "location", "cause", etc. - see Table 1. We wish to evaluate the proposed slot-fillers probabilistically by comparing them to a large number of ground-truth crowdsourced answers. Having an automatic evaluation is crucial to accelerating the development of strong models, however our setting (probabilistic evaluation of generative text) is novel, and thus we performed a rigorous study of potential contenders. We ultimately define a novel approach which aligns answers and measures the KL divergence between probabilities directly, which we justify on both theoretical and empirical grounds, where we observe a reasonable correlation with human judgements.

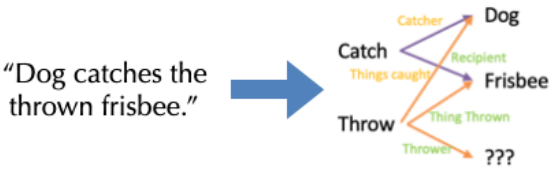


Figure 2: Representing context sentence using semantic representation (AMR) identifies the missing slots.

2 CFC Task Description

Given a direction such as "put the water on the burner to boil," it is *physical* common sense which allows us to know if we need to move other objects out of the way, and *conceptual* common sense which allows us to understand that the water is likely in a kettle and not simply dumped on the burner. In this paper we aim to create a task which evaluates both these aspects of common sense. If we had a way of identifying that the object containing the water is unspecified, we could pose this as a question answering task (i.e. "What is the water contained in?"). Unlike most question answering tasks, however, there is no single correct answer. In this example, the water could be placed in a "kettle", "pot", "cup", or "glass", although the former answers are more probable. This distribution is also *contextual* - consider how the relative probability shifts if we append the phrase "and add the spaghetti", or changes drastically if we append "and add 4-methoxy-3-buten-2-one," in which case the vessel is likely a beaker or test-tube.

It is clearly necessary for any machine learning model which claims to capture common sense to have some sense of the distribution over the implicit information, and moreover it may be absolutely integral to the safety of any model which provides directions to share the same distribution as humans. To assess a model's ability in this regard, we consider the context sentence as a structured semantic frame, identify a missing slot, and ask the model to provide a distribution of potential slot fillers as shown in Figure 2.

3 Dataset Creation and Analysis

In this section we describe the method of creating a dataset amenable to evaluating the task of CFC. The first item to be addressed is where to collect reasonable context sentences which contain some natural element of common sense. CommonGen (Lin et al., 2020c) is a recently released

Missing Slot	Definition	Examples
Arg0	Who/what does the event?	Sentence: putting cheese on the pizza. Arg0? Answers: person, cook
Purpose	What is the goal for doing the event?	Sentence: putting cheese on the pizza. Purpose? Answers: get nutrition, stop being hungry
Instrument	What kind of tools are used to accomplish the event?	Sentence: putting cheese on the pizza. Instrument? Answers: hands, spoon
Time	What is a particular time (time of day, season, etc.) for doing the event?	Sentence: putting cheese on the pizza. Time? Answers: lunch time, dinner time
Location	Where would the event usually happen?	Sentence: putting cheese on the pizza. Location? Answers: kitchen, restaurant

Table 1: Examples for different missing slot types

commonsense dataset which contains many short sentences describing basic information about daily life, and so we use this dataset as the source for potential context sentences.

Given a short sentence, we next need a way of identifying potential missing information. To this end, we perform semantic parsing on the sentence, aligning it with a structured semantic frame, and identify potential missing slots. We use AMR (Banarescu et al., 2013) for semantic parsing based on its ability to provide a rich representation of the sentence with a pre-defined fixed schema for the predicate roles. If a predicate is found, AMR parsing will match it to a schema and fill in the values for any identified slots. Any slots marked with `amr-unkown` indicate potential items of missing information, enabling us to obtain human annotations for the missing slot values.

We uniformly randomly sampled 63,788 sentences from the CommonGen dev dataset, and parsed them using the AMR parser from Cai and Lam (2020), generating 228,170 pairs of context questions with missing slots. From this, we randomly sampled 101 (sentence, missing slot) pairs for crowd workers to annotate, such that we had a balanced distribution of missing slot types, as detailed in Section 3.2. We present the context sentence and missing slot to crowdworkers, who were also provided with training examples and descriptions of the meaning of each slot type (see Table 1). The number of answers is chosen such that the resulting answer distribution is stable (see Section 3.2). Each element of the raw dataset therefore includes a context sentence, missing slot value, and a collection of slot fillers.

3.1 Probability Distribution

In an open-ended task where multiple humans are asked to provide answers as raw strings of text there are a multitude of answers which may essentially capture the same underlying idea. Ultimately we are not interested in the minute variations of the surface form, but rather in capturing the essence of the underlying concept. In the case of the boiling water example, for instance, we may want to treat "kettle" and "teapot" as though they were representative of the same general concept. As originally proposed in Boratko* et al. (2020), we consider *clustering* the responses, converting a set of answer strings into a categorical distribution over answer clusters, where the probability of obtaining an answer from a given cluster is proportional to the number of answer strings contained within it. We explore both manual clustering and automated clustering methods (see Section 4.2).

3.2 Analysis

Number of Answers The number of potential slot fillers might be very large, and we want to ensure we sample enough to approximate the true distribution over answer concepts. An essential question, therefore, is how many samples are enough to approximate the true distribution with reasonable error rate? This is a classic problem in statistics, for which the Neyman-Pearson lemma proves that the uniformly most powerful test is to consider the KL divergence $D_{\text{KL}}(g\|f) = \sum_x g(x) \log \frac{g(x)}{f(x)}$ where g is the empirical distribution and f is the true distribution (Harremoës and Tusnády, 2012). The recent work from Mardia et al. (2020) showed that this can be bounded by the following equation

$$\mathbb{P}(D_{KL}(g_{n,k}\|f) \geq \epsilon) \leq e^{-n\epsilon}\left[\frac{3c_1}{c_2}\sum_{i=0}^{k-2} K_{i-1}(\frac{e\sqrt{n}}{2\pi})^i\right]$$

where c_1 and c_2 are constant values, n is the number of samples, and k is the number of categories in the categorical distribution.

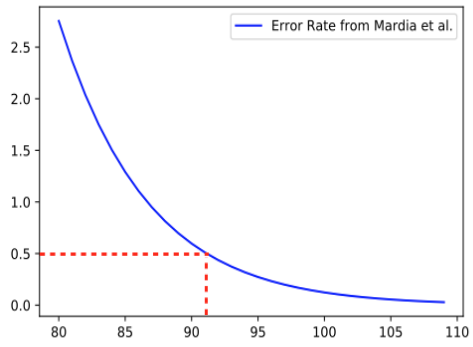


Figure 3: The relationship between the number of examples (x-axis), and the approximation error rate (y-axis).

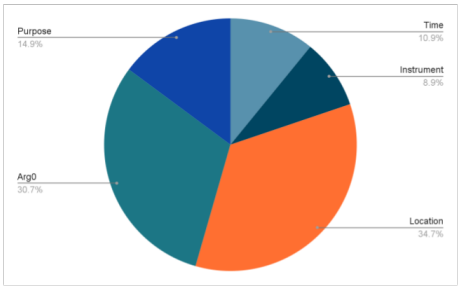


Figure 4: Question type distribution for CFC.

For our setting, we manually clustered 50 questions, and found that the number of categories is not more than 8. To get a bound on the number of answers we should collect, we set $\epsilon = 0.2$, $k = 8$, and solve $e^{-n\epsilon}\left[\frac{3c_1}{c_2}\sum_{i=0}^{k-2} K_{i-1}(\frac{e\sqrt{n}}{2\pi})^i\right]$ for n. Figure 3 shows the value of this bound on the y-axis for increasing numbers of samples n on the x-axis. As we can see from the graph, for samples greater than 90 the error rate is less than 0.5, allowing us to approximate the true answer distribution with 95% confidence if there are fewer than 8 categories in the categorical distribution.

Question Types We collected 101 (context, missing slot) pairs, and obtained 100 slot fillers for each from crowdworkers, resulting in 10,100 annotations overall. The annotators are paid 0.15 per answer, and they are all English speakers who are based in the US. We split the data, creating a dev set with 55 examples and a test set with 46 examples. The distribution of missing slot types are shown in Figure 4. Each question type is associated with a different type of commonsense reasoning, e.g time represents temporal commonsense reasoning. The dataset will be released.

4 Probabilistic Evaluation

In this section, we detail the method of evaluating the CFC task on the provided dataset. As commonsense is inherently probabilistic, a rigorous probabilistic evaluation is required; however the task is presented (both to humans and models) as a generative question answering task. Therefore, we need a way to compare two large sets of answer strings. We will proceed by how human evaluators may go about comparing these sets of answers to determine if they were drawn from similar distributions and then describe the various ways by which this process can be automated.

4.1 Human Evaluation

Our proposed framework for evaluating model prediction is depicted in Figure 5: Given a question, the ground truth answer set $\mathbf{G}$ and the model generated answers $\mathbf{H}$, the goal is to evaluate the similarity between these two answer sets.

For each question:
$G \leftarrow$ ground-truth answers (crowd-sourced)
$H \leftarrow$ evaluation answers (model)

For each human scorer:
Cluster G
Match H to clusters of G
Calculate score
Score(G, H) $\leftarrow$ average of scores

Figure 5: Human Evaluation Process

This is a difficult task even for a humans, particularly if the answer sets are large and diverse, however bearing in mind that we are more interested in *concepts* being captured rather than unique surface forms, a human might choose to cluster the answer strings in $\mathbf{G}$.[1] The expert annotator could then match the answers in $\mathbf{H}$ to the proposed ground-truth clusters in $\mathbf{G}$. At this point we can define categorical probability distributions over the clusters, P_g and P_h, where the probability assigned to a given cluster is equal to the number of answer strings assigned to it.[2] The similarity between $\mathbf{G}$ and $\mathbf{H}$ can be inferred by comparing the KL divergence of the two distributions, $D_{\mathrm{KL}}(\hat{P}_g||\hat{P}_h)$. To ensure evaluation robustness, we propose to repeat the same process with multiple human annotators and average the KL score to remove noise. In the end, the average KL value is the manual assessment

[1] When clustering, a new category "wrong" could be added to the answer set to account for the wrong answers for a question. These will then be discarded prior to model evaluation.

[2] To eliminate zero probabilities, we use Laplace smoothing on all categories before calculating the probabilities, — adding one dummy answer to all categories.

of the quality of the model's answers.

Although this approach yields reliable results, it poses the following challenges: 1. Human experts must cluster the answers in $\mathbf{G}$, which is an expensive, labor-intensive task. 2. Manually matching answers to clusters at evaluation time is infeasible.

4.2 Automatic Evaluation

Due to the disadvantages mentioned above of human evaluation, we aim to design an automatic method that could ease the human evaluation process while achieving a high correlation with human evaluation results.

The high-level approach is: 1. Embed ground-truth answers from $\mathbf{G}$ into a dense vector space. 2. Automatically cluster the embeddings to obtain ground-truth clusters of $\mathbf{G}$. 3. Match elements of $\mathbf{H}$ to clusters of $\mathbf{G}$ by assignment function score.

Each step presents a number of options, which we detail in the following sections. We evaluate the quality of a particular approach by calculating the Spearman correlation of KL divergence using the automatic evaluation compared with that of the manual evaluation across a variety of answer distributions (see Section 4.3 and 4.4).

Embedding We first embed the discrete word tokens in $\mathbf{G}$ and $\mathbf{H}$ as word vectors. We experimented with various word embedding models, both without context (Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2017)) and with context (BERT (Devlin et al., 2019), and RoBERTa (Liu et al., 2019)) We found FastText to perform best, and use it for all future embedding components.

Clustering Given the vector representation of the word answers, we experimented with various clustering algorithms including X-means (Pelleg et al., 2000), G-means (Zhao et al., 2008) and hierarchical agglomerative clustering (HAC) (Murtagh and Legendre, 2014) We used the implementation from pyclustering (Novikov, 2019). The parameters used by these clustering algorithms are treated as hyper-parameters and are tuned based on the correlation score as we discuss in section 4.3 and 4.4. We found HAC to perform best.

Matching Given the predicted answers, we want to match the answers to one or multiple ground truth answer clusters. This was also a requirement for ProtoQA (Boratko* et al., 2020), and we leverage the WordNet matching function which performed best in that setting. As we also have embeddings for our answers, we consider approaches based on embedding-based similarity functions.[3] We train a Gaussian regression model for each cluster in the ground-truth answers. The regression takes one answer representation as input, and output is the label of whether the answer belongs to one particular cluster. If an answer matches with multiple clusters we divide the weight evenly among all matching clusters.

4.3 Evaluator on ProtoQA

In order to validate the automatic evaluator's performance, we compared the automatic evaluator results with the human evaluation results on two generative datasets. We first evaluated the proposed evaluator using ProtoQA.

Sampling A robust automatic evaluation method should align well with human judgment on the best and worst predicted answers, and any in between. To achieve this, we propose three different sampling strategies to generate different answer distributions for each question.

- **Vanilla Sample.** We take random samples from model predictions directly.
- **Diverse Sample.** We take a linear combination of the ground-truth distribution and a uniform distribution to create a new distribution that interpolates between the ideal ground truth answers to random noise:

$$p = \alpha \hat{P}_g + (1 - \alpha)\text{uniform}$$

- **Centered Sample.** Arguably, the most important area to assess the quality of the evaluator is around answers which are likely to be returned from a model. We achieve this by taking a linear combination of the answer distributions of a given baseline model, the ground-truth distribution, and a uniform distribution, with most of the weight assigned to the answers from a baseline model:

$$p = z\hat{P}_h + w_1^{'}\hat{P}_g + w_2^{'}\text{uniform}$$
$$w_1^{'} = \frac{w_1 * (1 - z)}{w_1 + w_2}$$
$$w_2^{'} = \frac{w_2 * (1 - z)}{w_1 + w_2}$$
$$z \sim U(0.5, 1),\ w_1 \sim U(0, 1), w_2 \sim U(0, 1)$$

Clustering	Human	Human	Human	Hierarchical	Hierarchical
Matching	Human	WordNet	Embedding	WordNet	Embedding
Vanila Sample	1	0.351	0.333	0.199	0.148
Diverse Sample	1	0.800	0.890	0.748	0.754
Centered Sample	1	0.752	0.714	0.700	0.593

Table 2: Average Spearman correlation between human evaluation and automatic evaluation under different sampling strategies for ProtoQA dev questions. The top two rows indicate the supervision source: cluster results can be annotated by human or clustering algorithms, and matching could be done via human annotation or automatic similarity functions (wordnet or embedding-based function)

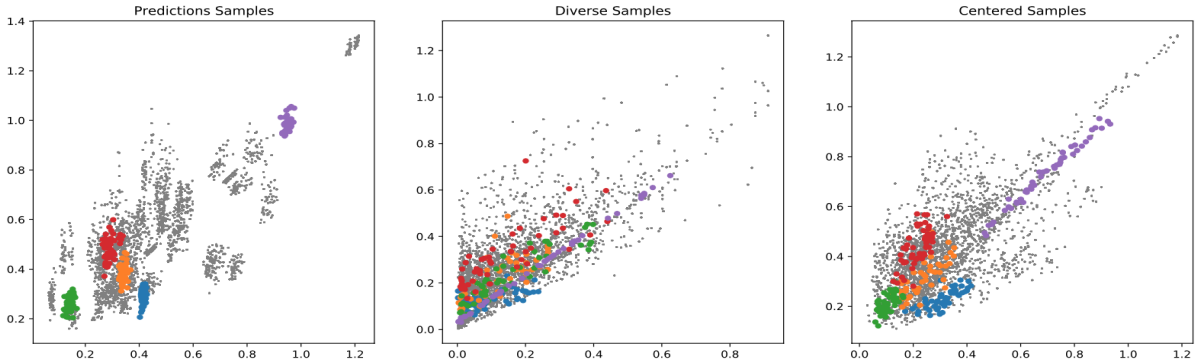


Figure 6: Correlation for sampled questions in ProtoQA with ground-truth clusters. The X-axis is the KL value with human assignment, and the y-axis is the KL value with WordNet assignment. This corresponds to the Human / WordNet column in Table 2. Different questions are annotated with different colors.

The ProtoQA dev set has 100 ground-truth answers and 30 additional human responses that were collected to measure human performance. For each question, in addition to the 130 human responses, we also use the 300 generated answers from the fine-tuned GPT-2 model. All of these answers are annotated by expert annotators with cluster matching to the ground-truth clusters. We use the union of the 30 human responses[4] and the GPT-2 answers as the prediction set, **H**. We sample 50 answer sets for each question from **H** and **G** according to the sampling procedure mentioned above.

We use automatic clustering and matching to get the automatic $D_{\mathrm{KL}}(\hat{P}_g||\hat{P}_h)$. We can also evaluate the KL for manual clustering and matching, as all answers in ProtoQA have been annotated by human experts with clusters and assignments. After getting the human and automatic KL values for various sampled answer sets, we use the Spearman correlation coefficients across questions to measure the alignment between automatic and human evaluation.

Results As we can see from Table 2, the correlation value from the Vanilla sample is fairly low; however, the correlation number for both Diverse sample and Centered Sample strategy are both much higher. Inspecting Figure 6 shows that the Vanilla sample strategy does not provide diverse answer sets. This suggests that our automatic evaluation may struggle to provide fine-grained distinctions, however in reality we predominately care about scoring results from *different* models, which is better represented by the Centered Sample and Diverse Sample approaches.

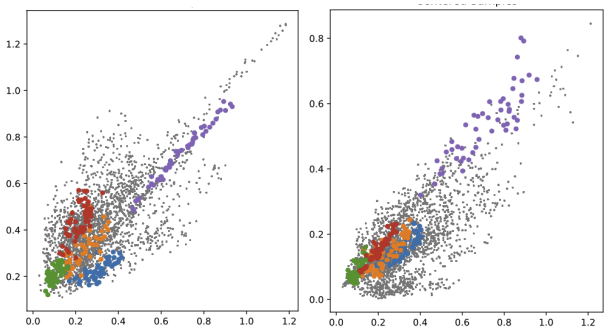
We also note that automating the matching function only yields higher correlation with scores based on human annotations, which is promising as this would only require manual annotation at dataset creation time, not for each evaluation. As we can see from Figure 7, the automatic predicted score is positively correlated with the score based on human-annotations under most conditions.

4.4 Evaluation on CFC

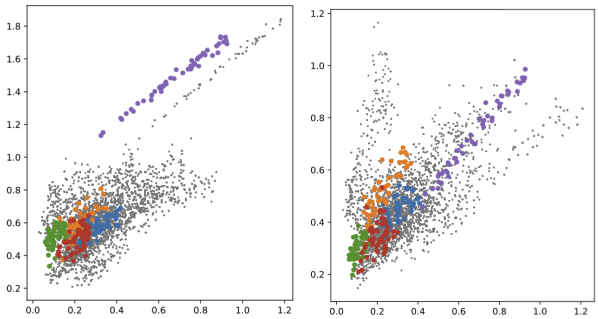
After preliminary experiments on ProtoQA, we verified our proposed evaluator on 55 dev questions

[3]We tried cosine similarity with FastText embeddings, but it is hard to decide the threshold for answers that belong to the "wrong" cluster. We tuned a few values and found that the results are unstable, so we don't report these results here.

[4]we scale up the 30 additional human answers to 300, in order to balance the model predictions and human answers.



(a) Human cluster, WordNet (b) Human cluster, Embedding



(c) Auto cluster, WordNet (d) Auto cluster, Embedding

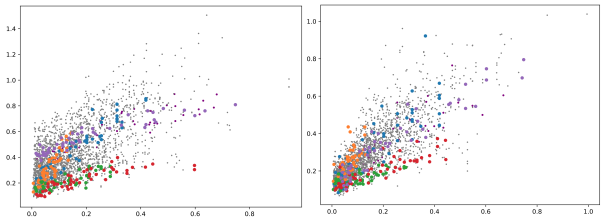
Figure 7: Centered sample correlation plots under different cluster and assignment methods: (a) human and WordNet (b) human and embedding (c) HAC and WordNet (d) HAC and embedding

in CFC. As in ProtoQA, expert annotators clustered the human responses into less than 8 clusters. Based on the results from the ProtoQA, we avoid the need to manually annotate model answers and instead focus on calculating the correlation between automated matching vs. automated matching and clustering. For this reason, we also solely evaluated using Diverse Sample. As shown in Table 3, the average correlation is fairly high (> 0.85).

We fixed the clustering parameters that gave us the best performance on these 55 questions to evaluate model performance on the test set. We also used these parameters to obtain the ground-truth evaluation number using both the WordNet similarity function and FastText similarity function. For WordNet we get a KL value of 0.237, while for FastText we get a KL value of 0.091. The human KL value should be 0 since it is the ground-truth answer set. So we use embedding-based similarity methods to report model performance in Section 5. From Figure 8, we see that the WordNet score function tends to produce a higher KL value compared to Human judgment, which explains the higher KL even for ground-truth answer sets.

Cluster	Human	Hierarchical	Hierarchical
Matching	Human	WordNet	Embedding
Diverse Sample	1	0.865	0.857

Table 3: Average spearman correlation between human and automatic evaluation under Diverse Sample for dev questions in CFC.



(a) Auto cluster, WordNet (b) Auto cluster, Embedding

Figure 8: Diverse sample correlation plots under hierarchical clustering, and different matching methods: (a) human cluster with WordNet matching (b) human cluster with embedding matching

5 Model Performance

5.1 GPT2

Our baseline is a generative language model, as modern language models have improved representational power, and recent evidence has demonstrated their effectiveness in modeling commonsense reasoning tasks (Weir et al., 2020; Tamborrino et al., 2020). We use the Hugging Face PyTorch implementation (Wolf et al., 2019)) of GPT-2 Large and XL (Radford et al., 2019). Our evaluation includes zero-shot and one-shot evaluations, as well as an evaluation after fine-tuning with the ProtoQA training data.

We convert CFC questions to a format "[Q]: context sentence, question, [A]". For the one-shot experiment, we sample one question and one answer from the CFC dev data, then we do the same conversion but pre-pend the converted question-answer pair to the actual question. The assumption is that as part of the prompt provided to the model, the model could get familiar with the task format.

For fine-tuning experiments, we took the ProtoQA pre-trained model[5]. We also trained the GPT-2 Large model with a task format that is similar to our task with the same "[Q]: question. [A]" format using the ProtoQA training data denoted as GPT2-L FT in Table 4. The models are fine-tuned for 3 epochs on an nVidia M40 GPU.

In order to generate different answers for the same prompt, we use Nucleus Sampling (Holtzman

[5] https://github.com/iesl/ProtoQA_GPT2

		GPT2-L	GPT2-XL	ProtoQA FT	GPT2-L FT	Human	GT
Dev	ZS	1.301	1.069	0.631	0.613	0.170	0.091
	FS(1)	0.848	0.740	0.562	0.585		
		GPT2-L	GPT2-XL	ProtoQA FT	GPT2-L FT	Human	GT
Test	ZS	1.197	0.962	0.576	0.612	0.040	0.076
	FS(1)	1.020	0.748	0.623	0.658		

Table 4: Model performance on CFC Data (**lower is better**). ZS means zero-shot, and FS(1) means one-shot prediction. GPT2-L and GPT2-XL is the GPT2 large and XL model respectively, ProtoQA FT is the ProtoQA fine-tuned, while GPT2-L FT is our own fined-tuned model. The GT column represents the KL values with the ground-truth answers.

et al., 2019). We generate 200 sampled answers from the GPT-2 Large model and 100 answers for the GPT-2 XL model for each question and treat them as the model prediction set. We experimented with temperatures from 0.1 to 1.0, and chose the model parameters with the best dev performance, then reported the test performance here.

5.2 Human Performance

In order to get a human performance on this task, we collected 30 additional human responses and evaluated them the same was as a model prediction.

5.3 Discussion

As we can see from Table 4, the model performance and human performance still have a large gap in terms of KL value, while the human performance is very close to ground truth answers. This indicates that the dataset is a challenging dataset for models, while humans could perform very well on this.

Moreover, GPT2-XL performs better despite the fact that the number of sampled answers is much less than the GPT2-large model (100 samples vs. 200 samples). Both of these non-fine-tuned models benefit a lot from zero-shot to one-shot. When the model gets fined-tuned with the ProtoQA training data, the performance improvement is more significant. Nevertheless, all model performances are still far from human-level performance, which leaves us ample space to improve the model.

6 Related Work

Creating commonsense benchmarks to evaluate model performance is a long-standing research topic (Sakaguchi et al., 2020; Lin et al., 2020c; Sap* et al., 2019). However, most benchmarks are created using a multiple-choice selection paradigm, which is simpler to evaluate but misaligned with the real-world use-case of commonsense knowledge, and most egregiously ignores the existence of multiple correct answers. We are not the first ones to gather multiple human answers to facilitate robust evaluations, however. Aydin et al. (2014) and Boratko* et al. (2020) also collected multiple human responses for each question to get aggregated human ground-truth answer sets.

Our work differs from these due to our emphasis on commonsense as *implicit* and *probabilistic*. We don't treat each answer equally; rather, we aim to match the answer distribution given by human responses. For this purpose, we propose a novel probabilistic evaluation for open-ended generation tasks with multiple correct answers. A similar probabilistic evaluation was studied from a language model generation point of view (Pillutla et al., 2021). They proposed a KL-based evaluation to measure language model generations, while our focus is on the implicit answer distribution.

7 Conclusion

In this paper, we assert that commonsense is an implicit probability distribution over missing information, and propose a dataset that aims to evaluate commonsense in this setting via a generative question answering task; moreover, we embrace the probabilistic nature of commonsense knowledge in both the dataset creation and the metric design. We propose a probabilistic automatic evaluation for evaluating answer distributions that is highly correlated to human judgment. Using this metric, we observe that model performance on our new dataset is significantly worse than human performance, indicating that the task is sufficiently challenging. In the future, we aim to further extend the size of the dataset, both in number of instances as well as answer length, which will involve challenging problems on both the dataset creation and probabilistic evaluation front.

Ethics Statement

The dataset aims to capture human commonsense, which is highly related to human bias. And due to the data collection nature of such a dataset, we acknowledge that our collected dataset might be biased toward certain populations, e.g., since all the data annotators are from the US, we may not cover commonsense knowledge for people from different cultural background, which we will try to mitigate in future work.

References

Bahadir Ismail Aydin, Yavuz Selim Yilmaz, Yaliang Li, Qi Li, Jing Gao, and Murat Demirbas. 2014. Crowdsourcing for multiple-choice question answering. In *AAAI*, pages 2946–2953. Citeseer.

Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. 2013. Abstract meaning representation for sembanking. In *LAW@ACL*.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Michael Boratko*, Xiang Lorraine Li*, Tim O'Gorman*, Rajarshi Das*, Dan Le, and Andrew McCallum. 2020. ProtoQA: A question answering dataset for prototypical common-sense reasoning. In *Conference on Empirical Methods in Natural Language Processing, EMNLP*.

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

Deng Cai and Wai Lam. 2020. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1290–1301, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Katrin Erk. 2022. The probabilistic turn in semantics and pragmatics. *Annual Review of Linguistics*, 8(1):101–121.

Peter Harremoës and Gábor Tusnády. 2012. Information divergence is more χ 2-distributed than the χ 2-statistics. In *2012 IEEE International Symposium on Information Theory Proceedings*, pages 533–537. IEEE.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*.

Lifu Huang, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2019. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2391–2401, Hong Kong, China. Association for Computational Linguistics.

Xiang Lorraine Li, Adhi Kuncoro, Cyprien de Masson d'Autume, Phil Blunsom, and Aida Nematzadeh. 2021. A systematic investigation of commonsense understanding in large language models. *arXiv preprint arXiv:2111.00607*.

Bill Yuchen Lin, Seyeon Lee, Rahul Khanna, and Xiang Ren. 2020a. Birds have four legs?! numersense: Probing numerical commonsense knowledge of pre-trained language models. In *Proceedings of EMNLP*.

Bill Yuchen Lin, Minghan Shen, Wangchunshu Zhou, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020b. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Conference on Empirical Methods in Natural Language Processing, EMNLP Findings*.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020c. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

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

Jay Mardia, Jiantao Jiao, Ervin Tánczos, Robert D Nowak, and Tsachy Weissman. 2020. Concentration inequalities for the empirical distribution of discrete

distributions: beyond the method of types. *Information and Inference: A Journal of the IMA*, 9(4):813–850.

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *ICLR*.

Fionn Murtagh and Pierre Legendre. 2014. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification*, 31(3):274–295.

Andrei Novikov. 2019. PyClustering: Data mining library. *Journal of Open Source Software*, 4(36):1230.

Dan Pelleg, Andrew W Moore, et al. 2000. X-means: Extending k-means with efficient estimation of the number of clusters. In *Icml*, volume 1, pages 727–734.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems*, 34:4816–4828.

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

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Winogrande: An adversarial winograd schema challenge at scale. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8732–8740.

Maarten Sap*, Hannah Rashkin*, Derek Chen, Ronan LeBras, and Yejin Choi. 2019. Socialiqa: Commonsense reasoning about social interactions. *Conference on Empirical Methods in Natural Language Processing, EMNLP*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL*.

Alexandre Tamborrino, Nicola Pellicano, Baptiste Pannier, Pascal Voitot, and Louise Naudin. 2020. Pre-training is (almost) all you need: An application to commonsense reasoning.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. https://github.com/kingoflolz/mesh-transformer-jax.

Nathaniel Weir, Adam Poliak, and Benjamin Van Durme. 2020. On the existence of tacit assumptions in contextualized language models.

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

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *EMNLP*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics, ACL*.

Zhonghua Zhao, Shanqing Guo, Qiuliang Xu, and Tao Ban. 2008. G-means: A clustering algorithm for intrusion detection. In *International Conference on Neural Information Processing*, pages 563–570. Springer.

A Appendix

This shows the dataset collection Amazon MTurk screen shot.

Overview

The goal is to collect missing commonsense knowledge in a given sentence or phrase. For example, "the plumber is fixing the sink." A piece of missing knowledge can be "the location of the plumber? (possible answer: bathroom, kitchen, basement)", "the tool the plumber used to fix the sink? (possible answers: hammer, wrenches)" etc. The missing knowledge is not in the given sentence. However, a human can provide reasonable answers to these questions easily.

Instructions

You will be given a short sentence or phrase and a slot indicating the missing information. You can answer with a word or a short phrase. The detailed slot definition and examples are shown below. A few reminders:

- 1. Remember to answer the first question: Is this a valid slot to ask for the given sentence? Otherwise, your answer will be rejected.
- 2. If you answered: "yes" to the first question, your answer string should **not** be part of the context sentence. Otherwise, your answer will be rejected.
- 3. If you answered: "no" to the first question, the alternative slots have to be part of the slot values. ['location', 'time', 'instrument', 'cause', 'arg0', 'parent-event']. Otherwise, your answer will be rejected.
- 4. The sentences may be short phrases or even incomplete because they are taken from image captions. You can answer the question with your own interpretation in this case. Thanks for your time! Contact me if you have any questions about the task.

Slot types & examples

Missing Slot	Definition	Example
Arg0	Who/what does the event?	**Sentence:** putting cheese on the pizza. **Arg0?** **Acceptable Answers (any one of them):** person, cook
Instrument	What kind of tools are used to accomplish the event?	**Sentence:** putting cheese on the pizza. **Instrument?** **Acceptable Answers (any one of them):** hands, spoon
Purpose	What is the goal for doing the event?	**Sentence:** putting cheese on the pizza. **Purpose?** **Acceptable Answers (any one of them):** get nutrition, stop being hungry
Location	Where would the event usually happen?	**Sentence:** putting cheese on the pizza. **Location?** **Acceptable Answers (any one of them):** kitchen, restaurant
Time	What is a particular time (time of day, season, etc.) for doing the event?.	**Sentence:** putting cheese on the pizza. **Time?** **Acceptable Answers (any one of them):** lunch time, dinner time

Click to for definition of valid slot and valid answer.

Sentence: an aircraft receives fuel from cargo aircraft . **Purpose?**

Is this a valid slot to ask for the given sentence?

- ○ Yes
- ○ No

If Yes, enter a word or short phrase as an answer. If No, enter a valid slot:

Submit