Masked Language Models Know Which are Popular: A Simple Ranking Strategy for Commonsense Question Answering

Xuan Luo[1,4], **Chuang Fan**[1,4], **Yice Zhang**[1,4], **Wanguo Jiang**[3]
Bing Qin[1], **Ruifeng Xu**[1,2,4*]
[1] Harbin Insitute of Technology [2] Peng Cheng Laboratory [3] Merchants Securities Co., LTD.
[4] Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies
gracexluo@hotmail.com fanchuanghit@gmail.com zhangyc_hit@163.com
jiangwanguo@cmschina.com.cn qinb@ir.hit.edu.cn xuruifeng@hit.edu.cn

Abstract

We propose a simple ranking strategy to solve a generative commonsense question answering (QA) problem. Compared with multiple-choice QA, it is challenging because the answers to a question are not unique and they are supposed to be popular and diverse. Our strategy exploits the dataset itself and negative samples that we collect from WordNet to train a ranker that picks out the most popular answers for commonsense questions. The effectiveness of our strategy is verified on different pre-trained masked language models (MLMs) in a pipeline framework, where an MLM reranks the generated answers. Further, we explore an end-to-end framework where MLMs are utilized to guide the generation of generative language models (GLMs). Taking advantage of reinforcement learning, we apply policy gradient to train a GLM with the rewards fed back by an MLM. Empirical results on `ProtoQA` dataset demonstrate that MLMs can acquire the ability to distinguish the popular answers and improve the typical answer generation of GLMs as well.

1 Introduction

Commonsense reasoning has been making progress over recent years (Rajani et al., 2019; Tamborrino et al., 2020; Lin et al., 2021; Liang et al., 2022, 2021), arising from the advent and wide application of pre-trained language models (PLMs). Most current commonsense reasoning studies focus on *multiple-choice* question answering (QA), such as `CommonsenseQA` (Talmor et al., 2019) and `Social IQa` (Sap et al., 2019b), for which a well-designed model is required to determine which of the candidate choices can best answer the question. However, such *multiple-choice* QA models may not be helpful in practical scenarios where candidate answers are not provided (e.g, answering a question asked in a search engine or during a conversation).

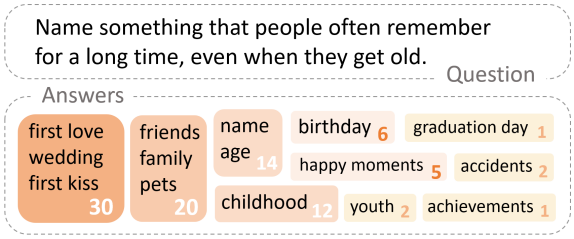

Figure 1: An example from ProtoQA dataset. The reasonable answers are collected and categorized into clusters with the numbers indicating their typicality.

Towards this problem, Boratko et al. (2020) present a novel question/answer dataset `ProtoQA` for generative QA, in which several plausible answers are generated as a ranked list, rather than selected from candidates. For example, as shown in Figure 1, given a question "*Name something that people often remember for a long time, even when they get old*", a QA model is expected to generate commonsensical and typical answers which cover commonest clusters as many as possible. In this case, a combination of "*first love*", "*friends*", and "*name*" would receive the highest score when three answers are allowed. In this setting, generative language models (GLMs) are apt to generate plausible answers (Ma et al., 2021; Chang and McCallum, 2022). However, in our preliminary experiments, we observe that GLMs such as GPT-2 (Radford et al., 2019), T5 (Raffel et al., 2020), and BART (Lewis et al., 2020) have difficulty distinguishing the most typical answers from the rare ones. Meanwhile, Zhou et al. (2020) find that masked language models (MLMs) such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) which utilize bidirectional contexts are more capable of learning commonsense knowledge than unidirectional LMs (UniLMs) such as GPT-2. Based on this observa-

* Corresponding Authors

Findings of the Association for Computational Linguistics: EMNLP 2022, pages 3200 - 3213
December 7-11, 2022

tion, we pose a question: *Whether MLMs can be utilized to promote the typicality of answers generated by GLMs?*

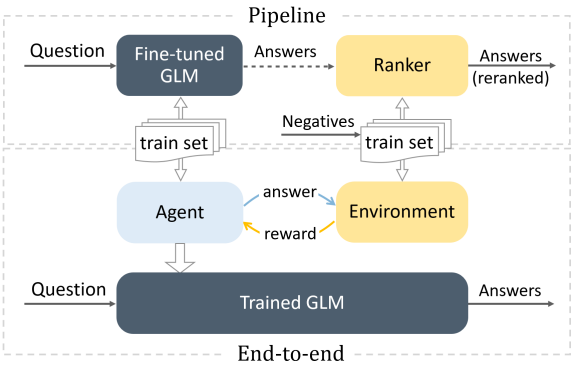

Figure 2: The pipeline framework and the end-to-end framework. In pipeline, the ranker reranks the generated answers to yield the final results. In end-to-end, the trained agent (GLM) directly yields the final results; during training phase, the agent passes the generated answers to the environment and receives the feedback rewards to update its parameters.

To this end, we propose a simple ranking strategy for MLMs to model the typicality of answers. In this strategy, an MLM is trained with the dataset purely without extra knowledge. To further increase the discrimination of the MLM, it is trained with the original answers as positive samples and negative samples gleaned from WordNet (Miller, 1994). After training, it serves as a ranker to find out the most popular answers among the generations by a fine-tuned GLM.

On top of that, we attempt to take advantage of MLMs' discrimination to improve the generation probability of typical answers by GLMs. Inspired by reinforcement learning (RL, Kaelbling et al., 1996), we construct a network with two PLMs: an agent (GLM) and an environment (MLM). We apply policy gradient (Sutton et al., 1999) to train the agent in three steps. First, the agent samples one answer for each question. Second, the environment, which is trained ahead with our ranking strategy, calculates a reward for every generated answer. Third, the agent updates its parameters according to both ground truth answers and the rewards from the environment. During inference, the final answers are generated by the trained agent without post-processing. The pipeline and the end-to-end frameworks are illustrated in Figure 2.

We design a series of experiments on `ProtoQA` to comprehensively examine our proposed ranking strategy. We develop our trials from UniLM GPT-2 to T5 and BART, which are sequence-to-sequence GLMs. For MLMs, we investigate BERT, RoBERTa, and DeBERTaV3 (He et al., 2021). The effectiveness of our strategy is evidenced by the experimental results: a leap (over 11 points) in the pipeline framework and a modest improvement (around 3 points) in the end-to-end framework.

Our research reveals that MLMs can learn to tell which answers are more popular, with little knowledge or even without knowledge. Moreover, they can guide the training of GLMs by providing higher rewards for the popular answers, and consequently the trained GLMs gain higher generation probabilities of typical answers.

2 Preliminary

We introduce the concept of reinforcement learning and the essential of policy gradient in this section.

Reinforcement Learning. Along with supervised learning and unsupervised learning, reinforcement learning (RL) is one of the three basic machine learning methods. It is composed of five elements: agent, environment, state, action, and reward. The agent takes actions within the environment and its states changes accordingly. Reversely, the environment feedbacks a reward to the agent.

Policy Gradient. In RL, the actor does not know whether an action is correct or not, it can only judge the quality of the action by the rewards. If an action gets more rewards, then the actor increases the probability of its occurrence; if fewer rewards, the probability decreases. Given a neural network with parameters θ and its state-action sequence τ, the expected reward of this network is the sum of the product of the likelihood of each sequence $p_\theta(\tau)$ and its corresponding reward $R(\tau)$. The objective function is $\max_\theta \bar{R}_\theta$.

$$\bar{\mathcal{R}}_\theta = \sum_\tau \mathcal{R}(\tau) p_\theta(\tau) \quad (1)$$

In our task, an answer may not be absolutely right or wrong, so a reward $\mathcal{R}(\cdot)$ is introduced to indicate the typicality of the answer. In our end-to-end framework, the agent, environment, and action are a GLM, an MLM, and answers generated by the GLM, respectively. The GLM is trained to receive a maximum expected reward.

3 Methodology

In this section, we elaborate on our simple ranking strategy for MLMs (i.e. the ranker or environment). Then, we detail the implementation of

policy gradient applied for the training of GLMs in the end-to-end framework.

3.1 Ranking Strategy

We train MLMs with (*question*||*answer*, *typicality*) pairs to model the distribution of typical answers.

We denote a question with s words as $q = \{q_1, q_2, \ldots, q_s\}$ and its original answer set with descending typicality as $\tilde{A}^q = \{(\tilde{A}^q_1, c^q_1), \ldots, (\tilde{A}^q_k, c^q_k)\}$, where each answer $\tilde{A}^q_i$ is composed of u words $\{a^q_{i1}, a^q_{i2}, \ldots, a^q_{iu}\}$ and c is the typicality. We set the typicality of negative answers to zero. Then, the negative set is $\bar{A}^q = \{(\bar{A}^q_1, 0), \ldots, (\bar{A}^q_n, 0)\}$ and the compound answer set is $A^q = \tilde{A}^q \cup \bar{A}^q$. We assume that the frequency can depict the distribution of typical answers, where $freq^q_i = c^q_i / \sum_{j=1}^{k} c^q_j$.

The typicality predicted by the ranker is obtained as follows:

$$h_{q,A^q_i} = \mathrm{MLM}(q||A^q_i)_{[\mathrm{CLS}]} \quad (2)$$

$$score_{q,A^q_i} = W h_{q,A^q_i} + b \quad (3)$$

where $q||A^q_i$ is concatenated as [CLS], q, [SEP], A^q_i, [SEP]. The scores are converted to an estimated probability of being typical by the $\mathrm{softmax}$ function so that negative values (scores) can be assigned to negative samples. Then, we compute the Kullback-Leibler divergence between the probability and the target distribution.

$$\sigma(q, A^q_i) = \mathrm{softmax}(score_{q,A^q_i}/t) \quad (4)$$

$$\mathcal{L}_{kl}(q) = \sum_{i=1}^{k} freq^q_i \log \frac{freq^q_i}{\sigma(q, A^q_i)} \quad (5)$$

where t is the temperature hyperparameter.

The above formulas only determine the relative numeric relationship between positive and negative answers. To ensure answers' class labels (positive or negative), we consider binary cross entropy ($\mathcal{L}_{bce}$) to constrain their value ranges. We use the least typical positive answer $\tilde{A}^q_k$ and one negative answer $\bar{A}^q_1$ to calculate the loss:

$$\mathcal{L}_{bce}(q) = -\log \mathrm{sigmoid}(score_{q,\tilde{A}^q_k}) - \log(1 - \mathrm{sigmoid}(score_{q,\bar{A}^q_1})) \quad (6)$$

The parameters of the ranker are updated with objective function $\min \sum_q (\mathcal{L}_{kl}(q) + \mathcal{L}_{bce}(q))$.

This strategy is also applicable without negative answers and it becomes a knowledge-free ranking strategy, where the objective function is $\min \sum_q \mathcal{L}_{kl}(q)$.

In the pipeline framework, a GLM needs a standard fine-tuning. After training, the fine-tuned GLM generates an answer set for all input questions. Then, the ranker estimates the typicality of every *question*||*answer* with Eq 2 - Eq 3. The score is finally converted to range (0, 1) with $\mathrm{sigmoid}$ function. The most popular answers are regarded as the final results.

3.2 GLM's Training with Policy Gradient

Inspired by reinforcement learning, we set an agent and an environment in our end-to-end framework. The agent is a GLM which is responsible for answer generation. The environment is an MLM discerning how typical the generated answers are. The parameters of the environment are well-trained with the ranking strategy (Section 3.1) and fixed during the training of the agent.

Specifically, given a question q, the GLM samples several answers $\hat{A}^q$. Then the MLM calculates the reward indicating the typicality for each answer $\hat{A}^q_i$ and feeds it back to the agent. We optimize the GLM by maximizing the overall expected reward. Technically, this can be formulated as:

$$\mathcal{L}_1(q) = -\sum_i \mathcal{R}(q, \hat{A}^q_i) \cdot \log P(\hat{A}^q_i|q) \quad (7)$$

$$P(\hat{A}^q_i|q) = \sum_j P_{\mathrm{GLM}}(\hat{A}^q_{i,t=j}|q, \hat{A}^q_{i,t\leq j}) \quad (8)$$

where $\mathcal{R}$ denotes the reward yielded by the environment. It is a normalized value of the score (Eq 3) by the $\mathrm{sigmoid}$ function:

$$\mathcal{R}(q, \hat{A}^q_i) = \mathrm{sigmoid}(score_{q,\hat{A}^q_i}) \quad (9)$$

In addition, we utilize ground truth answers $\tilde{A}^q_i$ to supervise the training of the GLM and the cross entropy loss is defined as follow:

$$\mathcal{L}_2(q) = -\sum_i \log P(\tilde{A}^q_i|q) \quad (10)$$

where $P(\tilde{A}^q_i|q)$ is computed as Eq 8.

As a whole, the objective function of the GLM is $\min \sum_q (\alpha \cdot \mathcal{L}_1(q) + \beta \cdot \mathcal{L}_2(q))$, where α and β are hyperparameters.

4 Experiments

4.1 Dataset

We evaluate our methods on a generative commonsense QA dataset: `ProtoQA` (Boratko et al.,

2020)[1][2], instead of *multiple-choice* benchmarks. It consists of around 9k commonsense reasoning questions over prototypical situations. The dataset splits used in our experiments follow the partition of Boratko et al.'s. (8782, 52, 102 pieces of questions for train, dev, and test). The average number of answers to each question in *train* set is 5.

The answers for test set of ProtoQA are possessed by AllenAI Leaderboards[3] and not public. As a result, the traps of test data leakage and parameters overfitting are eliminated from our results.

4.2 Negative Samples Preparation

From preliminary experiments, we have found that MLMs embrace the ability to tell whether a sentence is grounded in commonsense or not. To refine the discriminative ability of the ranker / environment, we construct negative samples with the following three strategies. The gleaned negatives are displayed in Table 1.

Synset. For each answer in `ProtoQA`, from its "brothers" (hyponyms of its hypernym), we chose one furthest "brother" as the negative according to the jcn-similarity (Jiang and Conrath, 1997) between synsets in WordNet (Miller, 1994).

Definition. For each question in `ProtoQA`, we collect a bunch of negatives among the "brothers" and "father" (hypernym) of every answer according to their definitions in WordNet. The word whose definition embedding has less than 0.5 cosine similarity with those of all answers to the question is regarded as a negative sample. The sentence embedding of definition is obtained by *bert-base-nli-mean-tokens* (Reimers and Gurevych, 2019).

Echo. We have observed that the GLMs would answer the questions with words in question stems, which are definitely not the expected answers in most cases. So we select words (nouns, verbs, and adjectives) appearing in the questions and their antonyms (only for adjectives) as the negatives.

4.3 Baselines

Due to the generative requirement of this task, old-fashioned classifier models are not applicable. We compare our methods with all the baselines reported by Boratko et al. (2020): *Human*, *QA Model*, *GPT-2*, and *GPT-2 FT*. We also report the results by fine-tuning T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) with the original dataset. In addition, there are two studies report only on dev set (Ma et al., 2021; Chang and McCallum, 2022), and we compare our results on dev set with theirs in Appendix D.

Question	At the beach, name something that might protect you from sun.	
Answer	sunscreen, sun block, umbrella, ...	
Strategy	**Negatives**	**Average**
`Synset`	cold cream	5
`Definition`	lanolin, nard	20
`Echo`	beach, protect, sun	5

Table 1: The statistics and instances of negative samples under different strategies. **Negatives** are chosen for the answer *sunscreen* and from the question stem. **Average** denotes the average number of negatives for each question in train set.

The MLMs we explore are DeBERTaV3 (He et al., 2021), RoBERTa (Liu et al., 2019), and BERT (Devlin et al., 2019), which are trained with our ranking strategy.

The version of models used in our experiments are *gpt2-large*, *t5-base*, *bart-large*, *deberta-v3-large*, *roberta-large*, and *bert-large-uncased*.

4.4 Evaluation

We follow the metrics proposed for ProtoQA by Boratko et al. (2020): Max Answers @ k and Max Incorrect @ k.[4] Employing the Hungarian matching algorithm (Kuhn, 1955; Munkres, 1957), the metrics compute the optimal matching between the answers and the clusters based on the reward matrix, where the rewards are equal to the size of clusters.[5] The scores of WordNet Similarity are given by AllenAI Leaderboards.

4.5 Parameters

The experimental results are mainly produced by the following parameters. AdamW is the optimizer for all models.

GLMs. We fine-tune the GPT-2 model with a batch size of 8, gradient accumulation batch of 1, and the others following the parameters for the best performing model by Boratko et al. (2020). GLMs are trained for 1 epoch, with 1e-5 and 1e-3 learning rates for BART and T5 respectively. For a fair comparison, we follow their generation settings.

[1] https://github.com/iesl/protoqa-data

[2] As far as we know, this is the only dataset for multi-answer generative commonsense QA.

[3] https://leaderboard.allenai.org/protoqa/submissions/public

[4] Max Answers @ k limits the total number of answers allowed to up to k answers. Max Incorrect @ k allows unlimited answers, but stops after k unmatched answers.

[5] https://github.com/iesl/protoqa-evaluator

Models	Max Answers (%)				Max Incorrect (%)			△ (%)		
	@ 1	@ 3	@ 5	@ 10	@ 1	@ 3	@ 5	Max	Min	Ave
Human*	78.4	76.8	76.0	77.0	59.0	74.0	77.9			
QA Model*	3.4	6.4	9.1	15.7	1.4	5.3	8.4			
GPT-2*	6.2	18.5	23.0	30.5	4.3	17.9	24.2			
+ rerank	41.9	39.5	39.6	42.7	25.4	35.1	39.0	35.7	12.2	19.8
GPT-2 FT*	36.4	44.4	46.4	53.5	26.1	41.7	48.2			
+ rerank	53.8	**56.4**	56.4	**62.2**	**39.3**	**53.2**	**57.6**	17.4	8.7	11.7
GPT-2 RL	41.9	45.5	49.4	55.6	27.7	45.8	51.0	5.5	1.1	2.9
+ rerank	**55.1**	**56.4**	**56.7**	60.2	39.2	50.9	57.0	18.7	6.7	11.2

Table 2: The main results on test set. Rows with * are reported by Boratko et al. (2020). The rest are our methods. The △ column are max, min, and average increments over the metrics, compared with the first row within the section. All scores are evaluated by AllenAI Leaderboards.

Methods	Max Answers (%)				Max Incorrect (%)			△ (%)		
	@1	@3	@5	@10	@1	@3	@5	Max	Min	Ave
GPT-2 FT*	36.4	44.4	46.4	53.5	26.1	41.7	48.2			
+filter (pipeline)	38.6	45.4	**50.0**	52.5	26.9	44.3	49.1	3.4	−1.0	1.4
GPT-2 RL (Binary)	**42.4**	**46.4**	49.0	**55.9**	**28.4**	**46.8**	**51.4**	6.0	2.0	3.3
GPT-2*	6.2	18.5	23.0	30.5	4.3	17.9	24.2			
GPT-2 RL w/o $\mathcal{L}_2$	22.0	29.8	30.7	36.2	15.5	27.5	31.7	15.8	5.7	9.8

Table 3: The results of ablation study on test set. The underlined scores are lower than the baseline *GPT-2 FT**.

Rankers. We train the Rankers with 15 negatives, which is composed of up to 3 from `Echo` and 12 from the union of `Synset` and `Definition`. Learning rate and epoch are set to 5e-5 and 5.

Agents. The environment is actually a ranker trained with the above parameters. The agent is trained with a weight decay of 1e-5. The epochs and coefficients (α, β) in the overall loss function vary with the GLM and MLM combinations. They are listed in Appendix B.

5 Results

In this section, we report results in both pipeline and end-to-end framework. Ablation study, further analysis of the two frameworks, and case study are located in Section 5.2, 5.3, 5.4, and 5.5. Results on *dev* set are listed in Appendix D and E. In addition, we test our method on multiple-choice dataset CommonsenseQA(Talmor et al., 2019) in Section 5.6.

5.1 Main Results

The main and best results are listed in Table 2. The *GPT-2 RL* model applies policy gradient to train the agent with rewards calculated by the environment. In the pipeline framework, we rerank the sampled answers generated by GLMs, denoted with "+*rerank*". We report reranked results of *GPT-2* (vanilla), *GPT-2 FT* (fine-tuned with train set), and *GPT-2 RL*. It should be noted that the ranker in pipeline is DeBERTaV3 trained for 1 epoch with train set plus 10 negatives per question.

The ranker raises the scores of fine-tuned models more than 11 points on average, manifest than those of *GPT-2 RL*. It indicates that the typicality of answers is beneficial to the training of rankers; nevertheless, the environment in end-to-end provides a relatively weak influence for the training of the agent. The biggest leap is the reranked results of vanilla *GPT-2*, especially Max Answers @ 1 (41.9, even better than *GPT-2 FT*'s 36.4). It corroborates that GPT-2 is an implicit source of world knowledge and thus, even without fine-tuning, the answers which are more popular can be excavated by the ranker. From the fact that the increases of *GPT-2 FT+rerank* and *GPT-2 RL+rerank* are almost on a par, we conjecture that policy gradient method increases the probability of typical answers and does not have deleterious effects on the overall diversity of generated answers.

5.2 Ablation Study

We conduct three ablation experiments (Table 3), and the corresponding findings are in bold.

A filter or binary ranker still works. For the pipeline framework, we degrade our softmax model to a filter, which filters *GPT-2 FT*'s outputs ranked by the occurrence. The results in the second line shows that it can cross out less popular answers to increase the typicality. Moreover, we train a hard label ranker, where 1 is assigned to the ground truth and 0 to the negatives. Results of reranking with

the binary ranker lying in the third line proves that our ranking strategy also works for a binary ranker.

The ranking strategy and policy gradient are reliable. For the end-to-end framework, we disregard the ground truth and eliminate $\mathcal{L}_2$ (Eq 10) in the objective function. The results in the last line are much better than *GPT-2*'s. It indicates that the reward is helpful to a vanilla PLM and therefore verifies the effectiveness of both our ranking strategy and applying policy gradient. Nevertheless, the result is worse than *GPT-2 FT* since the baseline was fine-tuned with ground truth. Therefore, the ground truth answers are still more valuable for PLM's training than pure reward.

5.3 Analysis on Ranking Strategy

The effectiveness of our simple ranking strategy is evaluated from two aspects: we experiment with 1) different numbers of epochs and the negatives on GPT-2; 2) different GLMs and MLMs.

5.3.1 Epochs and Negative Samples

To investigate the effect of the number of epochs and the negatives, we train the ranker with different (# epochs, # negatives) combinations and test them in the pipeline framework. We experiment with (1, 10), (1, 15), (5, 10), (5, 15), and (10, 5). Figure 3 depicts the scores and increments on each metric. Rankers of different combinations all significantly surpass the baseline on all metrics. Among them, there is a clear trend over the metrics, except for the Max Answers @ 1.

From the specific metric perspective, such as Max Answers @ 1, more negatives or epochs may be favorable. For instance, (5, 15) which has more negatives than (5, 10) achieves better score, and so does (1, 15) versus (1, 10). Surprisingly, (10, 5) trained with most epochs receives the best score.

From the overall perspective, the ranker learns the most important weights in early epochs, especially the first epoch. The rankers trained with one epoch gain greatest average increase (top three in the rightmost bar chart). On one hand, the increase falls as the training epoch rises, still there being an apparent improvement in the worst case (8.3 points of average increase). On the other hand, the appropriate number of negatives is correlative to the number of epochs. For small epochs, too many negatives may distract the attention to the positive answers. For moderate epochs, which may lead to overfitting, the shortage of negatives would intensify the imbalance. This explains why (1, 10) outperforms (1, 15) while (5, 15) exceeds (5, 10).

To verify the effect of negative samples, we compare the one-epoch rankers trained with or without negatives.[6] The solid line represents the ranker trained without negatives and it generally lies under (1, 10) and (1, 15). Despite the fact that its overall increment is slightly inferior to (1, 10) and (1, 15), it exceeds the other combinations. So the negatives are not a decisive factor to our ranking strategy, but they are valuable to boosting the performance. Therefore, **the negatives are the icing on the cake and our ranking strategy can be knowledge-free, without negatives.**

The wide range of epochs and negatives demonstrate the robustness of our ranking strategy.

5.3.2 Choices of MLMs

Reranked results of different GLM+MLM[7] combinations are displayed in Table 4. In Table 5 are the average scores and the standard deviations of the same MLM against different GLMs. Among the MLMs, there are large differences between their average scores but each model has a small variance. It indicates that, although MLMs have their own upper bound, their discrimination to answers generated by different GLMs are fairly stable. Taking both mean and standard deviation into consideration, DeBERTaV3 is the best ranker, followed by RoBERTa. The discriminative ability of BERT is insufficient, compared to the other two MLMs, and it even holds back the scores of *GPT-2 FT*.

The consistent reranked results of different GLMs further demonstrate that they can obtain the ability to answer commonsense questions after simple fine-tuning (sampled answers can cover typical ones for the rankers to choose from and to achieve similar reranked results). A suitable MLM can be utilized as a post-processer to effectively improve the typicality of the generations.

5.4 Analysis on Policy Gradient

In end-to-end mode, the typicality of generated answers by the agents trained with policy gradient are shown in Table 6. The majority of models improve the typicality of the first answer most (Max Answers @1), which also occurs in the pipeline.

However, the relative strength among MLMs is obscure. For GPT-2, DeBERTaV3 makes improvements in all metrics; RoBERTa has more merits than faults; BERT is deleterious for the whole,

[6] Because the rankers trained for one epoch perform best.

[7] The corresponding parameters are listed in Appendix A.

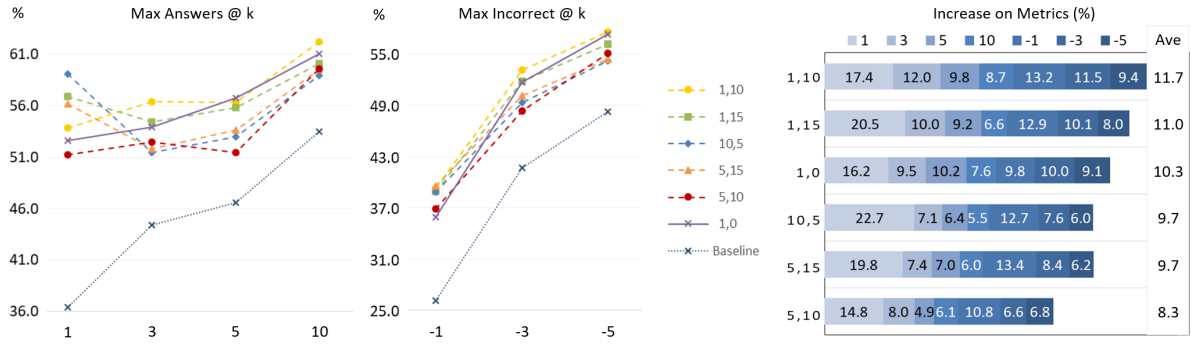

Figure 3: Effects of the number of negatives and epochs. Dotted line with crosses is the baseline *GPT-2 FT**; solid line with crosses is the ranker trained without negatives; dashed lines with various markers are trained with different (# epochs, # negatives) combinations. On the right side, the values in blue bars are the increases on each metric compared with the baseline, along with the average increase over all metrics.

GLM	Ranker	Max Answers (%)				Max Incorrect (%)			△ (%)		
		@ 1	@ 3	@ 5	@ 10	@ 1	@ 3	@ 5	Max	Min	Ave
GPT-2	FT*	36.4	44.4	46.4	53.5	26.1	41.7	48.2			
	FT	36.0	44.8	48.6	53.1	25.7	43.5	49.6			
	DeBERTaV3	**53.8**	**56.4**	56.4	**62.2**	**39.3**	**53.2**	57.6	17.4	8.7	11.7
	RoBERTa	49.7	46.0	49.0	56.2	31.5	44.6	51.1	13.3	1.6	4.4
	BERT	32.2	42.6	44.6	54.7	21.8	40.8	48.0	1.2	−4.3	−1.8
T5	FT	21.9	32.4	37.8	47.1	15.5	30.8	39.6			
	DeBERTaV3	**53.8**	55.2	**57.3**	61.5	**39.3**	52.9	**58.5**	**31.9**	**14.4**	**21.9**
	RoBERTa	47.2	48.1	49.6	54.9	28.6	44.7	49.9	25.3	7.8	14.0
	BERT	28.5	32.2	39.5	44.1	17.8	33.1	40.4	6.6	−3.0	1.5
BART	FT	26.4	31.7	37.9	47.2	16.6	31.1	38.5			
	DeBERTaV3	**53.8**	52.0	53.6	59.7	35.6	51.4	55.8	27.4	12.5	18.9
	RoBERTa	48.8	47.3	48.9	53.3	31.9	45.0	49.8	22.4	6.1	13.7
	BERT	28.5	31.3	39.3	47.8	16.9	32.4	40.0	2.1	−0.4	1.0

Table 4: The results of the pipeline framework on test set. FT denotes a fine-tuned GLM without reranking. The underlined scores are lower than the first row within the section. We also report our fine-tuned GPT-2 (in the second line) to show that it is our ranking strategy which contributes to the increase, rather than fine-tuning tricks.

Ranker	AVE/ STDE	Max Answers (%)				Max Incorrect (%)			△ (%)		
		@ 1	@ 3	@ 5	@ 10	@ 1	@ 3	@ 5	Max	Min	Ave
DeBERTaV3	AVE	**53.8**	**54.5**	**55.8**	**61.1**	**38.1**	**52.5**	**57.3**	**25.6**	**11.9**	**17.5**
RoBERTa		48.6	47.1	49.2	54.8	30.7	44.8	50.3	20.3	5.2	10.7
BERT		29.7	35.4	41.2	48.9	18.8	35.4	42.8	3.3	−2.6	0.2
DeBERTaV3	STDE	**0.0**	2.3	1.9	**1.3**	2.1	1.0	1.4	7.5	2.9	5.2
RoBERTa		1.3	**1.1**	**0.4**	1.5	**1.8**	**0.2**	**0.7**	6.3	3.2	5.5
BERT		2.1	6.3	3.0	5.4	2.6	4.6	4.5	**2.9**	**2.0**	**1.7**

Table 5: The discrimination of different MLMs. The values are derived from Table 4. *AVE* and *STDE* are the average and the standard deviation.

which may be the result of error propagation from the environment to the agent through the supervision signal. For T5, RoBERTa is the most helpful environment followed by DeBERTaV3 and BERT. For BART, RoBERTa is still the best environment; BERT improves most on average but has one metric lagged behind; DeBERTaV3 hardly plays a role.

We conjecture that the typicality of answers is indigestible to GLMs, so their generation probability of popular answers increase modestly and similarly after RL, no matter which MLM serves as the environment. Due to the more elaborate selection of hyperparameters, the larger time consumption (Appendix B and C), and the smaller gains of end-to-end models than reranking, the pipeline framework is much more economic.

5.5 Case Study

Since the answers of test set are not public, we gather different models' outputs of a question in dev set (Figure 4), taking GPT-2 as an example. *GPT-2 FT* generates 4 typical answers which covers 3 clusters. All models in the pipeline pick out at

Agent	Environment	Max Answers (%)				Max Incorrect (%)			△ (%)		
		@ 1	@ 3	@ 5	@ 10	@ 1	@ 3	@ 5	Max	Min	Ave
GPT-2	FT*	36.4	44.4	46.4	53.5	26.1	41.7	48.2			
	DeBERTaV3	**41.9**	**45.5**	**49.4**	**55.6**	**27.7**	**45.8**	**51.0**	**5.5**	**1.1**	**2.9**
	RoBERTa	41.0	45.2	48.7	53.4	25.2	43.6	47.6	4.6	−0.9	1.1
	BERT	40.4	39.9	46.9	54.6	26.6	39.5	48.1	4.0	−4.5	−0.1
T5	FT	21.9	32.4	37.8	47.1	15.5	30.8	39.6			
	DeBERTaV3	25.5	32.5	38.5	49.2	**17.5**	32.1	40.6	3.6	0.1	1.5
	RoBERTa	**28.2**	**35.4**	**40.1**	**50.9**	17.0	**34.5**	**41.8**	**6.3**	**1.5**	**3.3**
	BERT	26.8	34.3	36.1	45.6	17.1	31.1	38.6	4.9	−1.7	0.6
BART	FT	26.4	31.7	37.9	47.2	16.6	31.1	38.5			
	DeBERTaV3	28.5	31.9	38.1	47.0	17.9	30.8	38.6	2.1	−0.3	0.5
	RoBERTa	26.7	**34.4**	38.8	47.8	**18.3**	**33.3**	39.8	2.7	**0.3**	1.4
	BERT	**30.7**	31.5	**38.9**	**48.2**	17.5	32.5	**41.8**	**4.3**	−0.2	**1.7**

Table 6: The results of the end-to-end framework on test set. FT denotes a fine-tuned GLM without RL.

Question : Name something that people often remember for a long time, even when they get old.
Answer : {"first kiss", ... }^{0}, {"parents", "family", "pets", ... }^{1}, {"names", "age", ... }^{2}, {"memories of ...", ... }^{3}, {"birthday"}^{4}, ... , {"youth"}^{7}, ...

	GPT-2 FT	3	name^2, faces, looks, age^2, family^1, memories^3, appearance, health, job, clothes
Pipeline	+DeBERTaV3	6	children^3, birthday^4, faces, family^1, names^2, pets^1, loved one^0, names children^2, youth^7, favorite tv show
	+RoBERTa	6	name^2, birthday^4, family^1, looks, youth^7, age^2, faces, children^3, pets^1, loved one^0
	+BERT	5	names^2, time were young, things do, memories^3, children^3, birthday^4, loved one^0, health, appearance, pets^1
End-to-end	+DeBERTaV3	4	name^2, children^3, face, health, family^1, birthday^4, looks, parents^1, job, appearance
	+RoBERTa	5	name^2, birthday^4, person^0, name person^0, good memory^3, ate, first kiss^0, favorite tv show, favorite movie, friend^1
	+BERT	5	name^2, person^0, name person^0, birthday^4, good memory^3, first kiss^0, good meal, friend^1, loved^0, person's name

Figure 4: An example of answer generation in the pipeline and the end-to-end frameworks. The first row is a question and its answers in clusters. The following are generations of *GPT-2 FT*, reranked outputs of *GPT-2 FT*, and generations of *GPT-2 RL*. The third column is the number of unique clusters covered by the generations. Typical answers are in green with superscripts indicating the clusters they belong to. Full data is listed in Appendix F.

least 6 popular answers, two more than the baseline, and they also cover more unique clusters. So do the end-to-end models, but they cover fewer clusters on average than the pipeline models. The priority of typical answers among models in pipeline varies wider than those in end-to-end. In addition, the order of answers generated by end-to-end models are similar to the baseline (*+DeBERTaV3* vs *GPT-2 FT*) as well as to each other (*+RoBERTa* vs *+BERT*). These are consistent with previous experiment results: 1) pipeline models perform better than end-to-end models; 2) the increase of pipeline models differ significantly while end-to-end models have similar improvements; 3) Parameters of GLMs are slightly steered with policy gradient.

5.6 Results on CommonsenseQA

Although the number of answers varies with the question in this generative task, our method can be easily adapted for multiple-choice QA. [8] We assign 1 to the typicality value of the true answer and 0 to that of wrong answers. The multiple-choice ranker is trained with the same process as Section 3.1. During inference, for each question, the ranker calculates the probability of each (question, answer) pair. The highest one among all candidates is the predicted answer.

DeBERTaV3 is trained with learning rate of 1e-5 for 3 epochs. The accuracy on *dev* and *test* set of CommonsenseQA(Talmor et al., 2019) are 83.8 and 77.4 respectively.[9] We ranked 9 among the single models and the Top-15 single models on the CommonsenseQA leaderboard[10] are listed in Appendix G.

6 Related Work

Prior works evaluate LMs against commonsense benchmarks to probe the commonsense knowledge learned by LMs. Recent studies have shown that PLMs are implicit sources of world knowledge (Davison et al., 2019; Petroni et al., 2019). Et-

[8] Most existing methods are tailored to multiple-choice QA, but they are not adaptive to our answer generation task.

[9] We use official random split.

[10] https://www.tau-nlp.sites.tau.ac.il/csqa-leaderboard

tinger (2020) finds that BERT struggles with challenging commonsense inferences but it is robust on within-category distinctions. Meanwhile, Zhou et al. (2020) finds that MLMs are better at learning commonsense knowledge than unidirectional LMs. Based on these, we assume that PLMs can generate commonsensical answers and we utilize ProtoQA (Boratko et al., 2020) to explore the discriminative ability of MLMs. ProtoQA has been used by Ma et al. (2021) to study the effect of fine-tuning and prompt methods (Li and Liang, 2021; Shin et al., 2020) on PLM's learning process and by Chang and McCallum (2022) to compare the quality of the distributions generated by different LMs for answering ambiguous questions.

Goodfellow et al. (2014) proposed a training method for generative models. To make it feasible to discrete probabilistic models, Yu et al. (2017) proposed SeqGAN, an extended GAN with a reinforcement learning-based generator (Sutton and Barto, 2018), to solve the sequence generation problem. Inspired by SeqGAN, we apply policy gradient methods (Sutton et al., 1999) to optimize the agent with a reward function to guide the policy, where an MLM serves as the environment.

7 Conclusion

In this work, we propose a simple ranking strategy for masked language models (MLMs) to find out typical answers from the generation of generative language models (GLMs). It is a knowledge-free strategy when training without the negatives. In addition, we apply policy gradient to the training of GLMs to improve the generation of GLMs in end-to-end mode. Comprehensive experiments demonstrate the effectiveness of our proposed strategy and the discriminative ability of MLMs.

Limitations

Our research investigates the discriminative ability of MLMs in the pipeline framework and the end-to-end framework and these two paradigms both increase the typicality of generated answers. However, there still exists a large gap between performance of well-designed models and human's. In addition, reasoning of neural models lacks transparency and interpretability. We argue that it would be more beneficial to investigate reasoners that can utilize commonsense knowledge bases such as ConceptNet (Liu and Singh, 2004) and ATOMIC (Sap et al., 2019a). Knowledge-aware models can incorporate external knowledge as relational inductive biases in an explicit way. This would enhance their reasoning capability and improve the transparency of model behaviors for more interpretable predictions.

Ethics Statement

We experiment on ProtoQA, a public commonsense question answering dataset published by Boratko et al. (2020). Our data splits are consistent with Boratko et al.'s. We collect negative samples from WordNet (Miller, 1994), a public knowledge base. Except for the negative samples, there is no other additional data collection process.

Acknowledgements

This work was partially supported by the National Natural Science Foundation of China 62006062, 62176076, and 61976073, Shenzhen Foundational Research Funding JCYJ20200109113441941, JCYJ20210324115614039, The Major Key Project of PCL2021A06, Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies 2022B1212010005, and Joint Lab of HITSZ and China Merchants Securities.

References

Michael Boratko, Xiang Li, Tim O'Gorman, Rajarshi Das, Dan Le, and Andrew McCallum. 2020. ProtoQA: A question answering dataset for prototypical common-sense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1122–1136, Online. Association for Computational Linguistics.

Haw-Shiuan Chang and Andrew McCallum. 2022. Softmax bottleneck makes language models unable to represent multi-mode word distributions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8048–8073, Dublin, Ireland. Association for Computational Linguistics.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. 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 1173–1178, Hong Kong, China. 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 *Proceedings of the 2019 Conference of*

the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *NIPS*.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pretraining with gradient-disentangled embedding sharing. *arXiv preprint arXiv:2111.09543*.

Jay J. Jiang and David W. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *ROCLING/IJCLCLP*.

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research*, 4:237–285.

Harold W Kuhn. 1955. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. 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*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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)*, pages 4582–4597, Online. Association for Computational Linguistics.

Bin Liang, Hang Su, Lin Gui, Erik Cambria, and Ruifeng Xu. 2022. Aspect-based sentiment analysis via affective knowledge enhanced graph convolutional networks. *Knowledge-Based Systems*, 235:107643.

Bin Liang, Rongdi Yin, Jiachen Du, Lin Gui, Yulan He, Min Yang, and Ruifeng Xu. 2021. Embedding refinement framework for targeted aspect-based sentiment analysis. *IEEE Transactions on Affective Computing*.

Bill Yuchen Lin, Seyeon Lee, Xiaoyang Qiao, and Xiang Ren. 2021. Common sense beyond English: Evaluating and improving multilingual language models for commonsense reasoning. 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)*, pages 1274–1287, Online. Association for Computational Linguistics.

H. Liu and P. Singh. 2004. Conceptnet — a practical commonsense reasoning tool-kit. *BT Technology Journal*, 22(4):211–226.

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 preprint arXiv:1907.11692*.

Kaixin Ma, Filip Ilievski, Jonathan Francis, Satoru Ozaki, Eric Nyberg, and Alessandro Oltramari. 2021. Exploring strategies for generalizable commonsense reasoning with pre-trained models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5474–5483, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

George A. Miller. 1994. WordNet: A lexical database for English. In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*.

James Munkres. 1957. Algorithms for the assignment and transportation problems. *Journal of the society for industrial and applied mathematics*, 5(1):32–38.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

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.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nazneen Fatema Rajani, Bryan McCann, Caiming Xiong, and Richard Socher. 2019. Explain yourself! leveraging language models for commonsense reasoning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*,

pages 4932–4942, Florence, Italy. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019a. Atomic: An atlas of machine commonsense for if-then reasoning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3027–3035.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019b. Social IQa: Commonsense reasoning about social interactions. 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 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. 1999. Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, NIPS'99, page 1057–1063, Cambridge, MA, USA. MIT Press.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Alexandre Tamborrino, Nicola Pellicanò, Baptiste Pannier, Pascal Voitot, and Louise Naudin. 2020. Pre-training is (almost) all you need: An application to commonsense reasoning. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3878–3887, Online. Association for Computational Linguistics.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. 2020. Evaluating commonsense in pre-trained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9733–9740.

A Parameters of Pipeline Models

The rankers are trained with the following parameters in Table 7.

GLM	Ranker	Epoch	Negative
GPT2	DeBERTaV3	1	10
	RoBERTa BERT	5	15
T5	DeBERTaV3	1	10
	RoBERTa BERT	5	15
BART	DeBERTaV3 RoBERTa BERT	5	15

Table 7: The parameters of the rankers.

B Parameters of End-to-end Models

The agents are trained with the following parameters in Table 8.

G	D	Epoch	α	β
GPT-2	DeBERTaV3	2	1	8
	RoBERTa			4
	BERT	3		
T5	DeBERTaV3	2	4	0.5
	RoBERTa BERT			1
BART	DeBERTaV3	2	4	1
	RoBERTa	3		
	BERT	2		

Table 8: The parameters of the agents.

C Time Consumption and Model Size

The statistics of time consumption and the amount of parameters are listed in Table 9. In reinforcement learning, since only the parameters of the agents are updated, the amounts of parameters are the same as single models'.

D Results of Pipeline Models on Dev Set

We report the results of GPT-2, BART, and T5 on dev set in Table 10. Ma et al. (2021) experimented with GPT-2 and BART; Chang and McCallum (2022) experimented with GPT-2 only.

Mode	Model	Time	Parameter	Epoch
Single	GPT-2	30 min	774M	1
	T5	20 min	223M	
	BART	20 min	406M	
	DeBERTaV3	15 min	435M	5
	RoBERTa		355M	
	BERT		335M	
RL	GPT-2	1.5 hr	774M	2
	T5	6 hr	223M	
	BART	8 hr	406M	

Table 9: Time consumption on average per epoch and the number of parameters.

In addition, we report the reranked result of different GLM+MLM combinations on dev set in Table 11.

E Results of End-to-end Models on Dev Set

We report the results of GPT-2, BART, and T5 on dev set in Table 12.

F Metadata of the Example in Figure 4

Table 13 and Table 14 are the raw answers and clustered answers of dev question r2q15, respectively.

G CommonsenseQA Leaderboard

Top-15 single models are listed in Table 15.

Model	Strategy	Max Answers (%)				Max Incorrect (%)			△ (%)
		@1	@3	@5	@10	@1	@3	@5	Ave
GPT-2	softmax	34.1	35.2	37.8	45.0	18.3	30.7	38.5	
	MoS	33.9	36.0	37.7	44.9	18.3	31.7	38.2	
	MFS w/o Multi-partition	**34.3**	**36.7**	38.1	45.2	19.4	32.0	38.6	
	MFS	34.1	**36.7**	**38.6**	**45.4**	**19.7**	**32.1**	**39.7**	
	(vanilla)	28.2	27.1	27.2	30.7	14.4	21.1	27.5	
	Autoprompt	25.5	30.9	35.1	42.4	14.7	28.7	35.5	
	Prefix-tuning	42.7	**51.5**	52.2	60.8	28.8	47.6	56.9	
	Finetune	**49.3**	50.3	**53.0**	**63.0**	**31.9**	**49.9**	**57.9**	
	rerank (Ours)	**55.8**	**52.2**	**56.2**	61.7	**36.3**	**51.3**	55.6	1.8
BART	(vanilla)	20.9	29.8	32.2	37.5	15.1	27.3	32.2	
	Autoprompt	28.2	33.8	37.2	44.6	16.6	31.1	38.9	
	Prefix-tuning	45.5	51.0	54.8	**62.9**	32.7	51.4	58.7	
	Finetune	**53.6**	**54.3**	**56.3**	62.6	**35.6**	**53.9**	**59.5**	
	rerank (Ours)	**63.1**	**55.2**	**58.9**	**63.3**	**38.1**	52.3	59.3	2.0
T5	rerank (Ours)	**47.4**	**51.0**	**55.3**	**62.3**	**34.1**	**50.5**	**59.0**	

Table 10: The main results of the pipeline framework on dev set. The ranker is DeBERTaV3. The bold figures are the best scores in these two related works and the scores that is higher than baselines in our method. The average increment is the difference between our scores and the best scores among other models.

GLM	Ranker	Max Answers (%)				Max Incorrect (%)			△ (%)
		@1	@3	@5	@10	@1	@3	@5	Ave
GPT-2	FT	41.9	43.2	47.6	53.6	21.7	38.5	49.1	
	DeBERTaV3	55.8	52.2	56.2	61.7	36.3	51.3	55.6	10.5
	RoBERTa	42.2	46.5	47.7	57.4	26.2	39.4	50.3	2.0
	BERT	37.5	40.6	43.0	52.5	22.2	38.0	45.2	-2.4
T5	FT	24.3	31.8	42.2	50.8	13.1	32.7	42.3	
	DeBERTaV3	47.4	51.0	55.3	62.3	34.1	50.5	59.0	17.5
	RoBERTa	34.8	39.7	45.7	51.8	20.4	39.5	46.3	5.9
	BERT	36.2	36.8	38.1	48.1	21.0	32.2	41.7	2.4
BART	FT	25.8	33.1	39.8	49.3	15.1	32.5	40.2	
	DeBERTaV3	**63.1**	**55.2**	**58.9**	**63.3**	**38.1**	**52.3**	**59.3**	22.1
	RoBERTa	41.8	42.7	48.4	53.0	21.8	38.6	46.9	8.2
	BERT	27.5	34.8	39.4	50.0	15.6	33.7	39.8	0.7

Table 11: The results of the pipeline framework on dev set. FT denotes a fine-tuned GLM without reranking.

Agent	Environment	Max Answers (%)				Max Incorrect (%)			△ (%)
		@1	@3	@5	@10	@1	@3	@5	Ave
GPT-2	FT	41.9	43.2	47.6	53.6	21.7	38.5	49.1	
	DeBERTaV3	**48.8**	47.3	51.0	**57.6**	**30.4**	43.1	**52.4**	5.0
	RoBERTa	37.5	47.3	**52.7**	55.3	27.8	**45.9**	51.8	3.2
	BERT	38.1	**48.1**	50.5	56.2	27.0	44.9	49.5	2.7
T5	FT	24.3	31.8	42.2	50.8	13.1	32.7	42.3	
	DeBERTaV3	27.3	33.0	40.8	50.1	14.4	32.8	42.5	0.5
	RoBERTa	25.9	34.7	38.6	53.1	15.7	31.8	46.0	1.2
	BERT	24.6	34.0	41.7	52.5	16.0	34.0	41.9	1.1
BART	FT	25.8	33.1	39.8	49.3	15.1	32.5	40.2	
	DeBERTaV3	30.2	34.8	37.5	49.1	17.2	33.2	39.5	0.8
	RoBERTa	31.0	37.2	38.3	47.6	16.5	32.4	37.1	0.6
	BERT	29.2	35.0	37.4	47.0	16.2	34.2	40.0	0.5

Table 12: The results of the end-to-end framework on dev set. FT denotes a fine-tuned GLM without RL.

Data split	dev.crowdsourced
ID	r2q15
Question	Name something that people often remember for a long time, even when they get old.
Answers	"birthday": 6, "a kiss": 1, "accidents": 1, "achievements": 1, "age": 3, "childhood": 9, "children trauma": 1, "eat": 1, "family": 1, "family members": 1, "first kiss": 6, "first love": 9, "first time": 1, "first time something happened": 1, "friends": 4, "graduation day": 1, "grandmother": 1, "happy moments": 1, "best childhood moments": 1, "travels": 1, "parents": 5, "lose virginity": 1, "memories of childness": 1, "mother": 1, "name": 6, "name of family": 1, "no": 1, "one love": 1, "pets": 1, "relations": 1, "sons": 3, "bad experiences": 1, "heat": 1, "the first person they fell in love": 1, "love of his life": 2, "names": 2, "love": 3, "the old times": 1, "toys": 1, "travel": 2, "wedding": 4, "when a son is born": 1, "wife": 1, "your age": 2, "happy moment": 1, "anniversaries": 1, "youth": 2

Table 13: Raw answers of dev question r2q15. The answer in clusters are listed in Table 14.

No.	Count	Answers
0	30	"the first person they fell in love", "first kiss", "love", "wedding", "one love", "a kiss", "first time something happened", "first love", "first time", "love of his life", "lose virginity"
1	20	"mother", "parents", "family members", "when a son is born", "family", "grandmother", "pets", "wife", "sons", "friends", "relations"
2	14	"name of family", "names", "age", "your age", "name"
3	12	"childhood", "best childhood moments", "memories of childness", "children trauma"
4	6	"birthday"
5	5	"happy moment", "travels", "happy moments", "travel"
6	2	"bad experiences", "accidents"
7	2	"youth"
8	1	"achievements"
9	1	"graduation day"
10	1	"anniversaries"
11	1	"heat"

Table 14: Answer clusters of dev question r2q15.

Rank	Model	Affiliation	Accuracy	Accuracy with ConceptNet
	Human		88.9	
1	CPACE	Anonymous		87.4
2	KEAR	Microsoft Azure Cognitive Services Research		86.1
3	ALBERT+DESC-KCR	Microsoft Cognitive Services Research		80.7
4	ALBERT+KD	HIT-SCIR-QA		80.3
5	Albert + KCR	ITNLP (Harbin Institute of Technology)		79.5
6	UnifiedQA	Allen Institute for AI	79.1	
7	Albert+Headhunter	SUDA-NLP & I2R	78.4	
8	T5	Allen Institute for AI	78.1	
9	**DeBERTaV3 (Ours)**	**HITSZ-HLT**	**77.4**	
10	ALBERT+HGN	USC MOWGLI / INK Lab		77.3
11	TeGBERT	Anonymous		76.8
12	QA-GNN	Stanford University		76.1
13	PEAR	Sogang University and Gachon University		76.1
14	Albert + PathGenerator	USC MOWGLI / INK Lab		75.6
15	ACP Graph + ELECTRA	NLP & AI, Korea University		75.4

Table 15: Top-15 single models on CommonsenseQA Leaderboard.