GRADA: Graph Generative Data Augmentation for Commonsense Reasoning

Adyasha Maharana **Mohit Bansal**
Department of Computer Science
University of North Carolina at Chapel Hill
{adyasha,mbansal}@cs.unc.edu

Abstract

Recent advances in commonsense reasoning have been fueled by the availability of large-scale human annotated datasets. Manual annotation of such datasets, many of which are based on existing knowledge bases, is expensive and not scalable. Moreover, it is challenging to build augmentation data for commonsense reasoning because the synthetic questions need to adhere to real-world scenarios. Hence, we present GRADA, a graph-generative data augmentation framework to synthesize factual data samples from knowledge graphs for commonsense reasoning datasets. First, we train a graph-to-text model for conditional generation of questions from graph entities and relations. Then, we train a generator with GAN loss to generate distractors for synthetic questions. Our approach improves performance for SocialIQA, CODAH, HellaSwag and CommonsenseQA, and works well for generative tasks like ProtoQA. We show improvement in robustness to semantic adversaries after training with GRADA and provide human evaluation of the quality of synthetic datasets in terms of factuality and answerability. Our work provides evidence and encourages future research into graph-based generative data augmentation. [1]

1 Introduction

Recent work has seen the emergence of several datasets for improving commonsense reasoning of language models through tasks like question answering (QA) (Sap et al., 2019b; Talmor et al., 2019; Bisk et al., 2020) and natural language inference (Bhagavatula et al., 2020; Zellers et al., 2019; Sakaguchi et al., 2020). Some of these datasets are based on existing knowledge graphs that represent different aspects of commonsense through entities and relations. For example, annotators for SocialIQA (Sap et al., 2019b) were shown an event from the inferential knowledge graph ATOMIC (Sap et al., 2019a) and instructed to turn it into a sentence by adding names, filling placeholders and adding context, etc. For multiple-choice QA datasets, annotators are also instructed to write distractor choices for each question. These useful datasets are collected through a time-taking and money-intensive crowdsourcing process which is hard to scale. Large pretrained models like GPT2 (Radford et al., 2018) can be finetuned to generate sentences from narrow data distributions, and it has recently been leveraged to augment datasets for text classification (Anaby-Tavor et al., 2020) and question answering (Puri et al., 2020; Yang et al., 2020). However, it is challenging to generate augmentation data for commonsense reasoning because the generated questions and answers (referred to as "synthetic" in rest of the paper) need to depict plausible real-world scenarios accurately. Hence, we develop GRADA, a graph-based generative data augmentation framework to generate

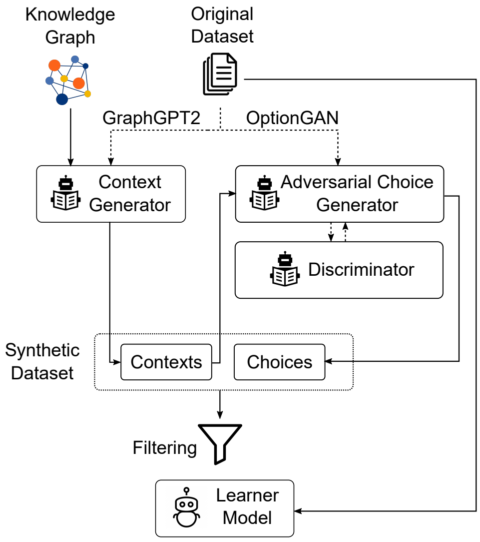

Figure 1: GRADA framework: The original dataset is used to train GraphGPT2, a graph-to-text question generator and OptionGAN, a distractor generator. The synthetic dataset is subjected to filtering and used to train the model in combniation with the original dataset.

[1]Code and synthetic data files are available at https://github.com/adymaharana/GraDA.

synthetic samples from existing knowledge graphs that encode information about the real world.

Each sample in commonsense reasoning datasets comprises a question which describes a real-world scenario and can be mapped to a set of predefined entities and relations from knowledge bases like ConceptNet and ATOMIC. For instance, the question "Besides a mattress, name something people sleep on." from the ProtoQA dataset (Boratko et al., 2020) can be mapped to the single-hop path (mattress, *RelatedTo*, people) using ConceptNet. If a pretrained language model is trained to conditionally generate questions from such input paths, we can expect it to generate sensible questions when it is provided new paths with similar relations. The model will likely generalize to unseen entity nodes and generate questions containing unique commonsense knowledge. Following this intuition, we finetune GPT2 (Radford et al., 2019) to generate questions which explicitly depict the entities and relations in input path. When trained on the aforementioned example (alongside other similar examples) and provided with the new path (mattress, *RelatedTo*, soft), our model generates "Besides a mattress, name something that's soft.", which is a valid question for probing real-world commonsense. Usually, these paths contain multiple nodes with several hops and hence are referred to as graphs in rest of the paper. In order to represent the graph, we explore both (a) encoding of linearized graph and (b) augmentation of linear encodings with structure-aware encoding of graph, and find that the latter improves the transfer of semantic knowledge from graph to text.

Synthetic questions need to be accompanied by synthetic answers and distractor choices (for multiple-choice datasets), which are similarly generated by finetuning GPT2 for conditional generation of answers/distractors from the question. However, Yang et al. (2020) report that human annotators find it hard to pick a unique/unambiguous answer in more than 50% of the synthetic dataset generated in this manner. Therefore, we explore an alternative where we finetune the generative model within a GAN framework (Nie et al., 2019a) where it is continuously challenged by a discriminator model to generate unique distractors that can fool the discriminator (see OptionGAN, Figure 1). The synthetic questions and answers thus generated are assembled into synthetic samples which are then used in a two-stage training pipeline (Mitra et al., 2019). Additionally, since the generative pipeline is only an approximate imitation of the human annotation process, we are left with several ambiguous and inaccurate samples in the synthetic pool. Hence, we retain the most informative data samples from the synthetic pool by using Question Answering Probability (Zhang and Bansal, 2019) to measure accuracy by answerability. Our contributions can be summarized as follows:

- We present a generative framework consisting of (i) a graph-to-text model to convert knowledge graphs to questions, (ii) a model finetuned with GAN loss to generate distractors for commonsense reasoning QA datasets, and (iii) combined with a filter for selecting the most informative samples from synthetic datasets.
- We improve performance on commonsense reasoning datasets, and perform ablation analysis to show the impact of various modules in our framework as well as human evaluation of synthetic dataset quality.

2 Related Work

Explicit reasoning over knowledge graphs has been a popular approach for improving commonsense understanding of QA models. Bauer et al. (2018); Lin et al. (2019); De Cao et al. (2019); Feng et al. (2020) and Lv et al. (2020) extract relevant multi-hop relational commonsense from knowledge graphs and show significant improvements over models that operate solely on text. Devlin et al. (2019); Yang et al. (2019); Ye et al. (2019) expand the rich latent knowledge of large pretrained models by finetuning on similar corpora (Havasi et al., 2010) before finetuning on the target dataset. Mitra et al. (2019) convert external resources (Koupaee and Wang, 2018) to QA samples for data augmentation. Yang et al. (2020) generate randomly initialized samples from finetuned GPT2 as augmentation data for target datasets. We ground the generated samples to real-world facts by providing knowledge graphs as input to the model.

There has been a surge of efforts in neural graph-to-text modeling in the recent years. Marcheggiani and Perez-Beltrachini (2018) encode input graphs using a graph convolutional encoder (Kipf and Welling, 2017). Koncel-Kedziorski et al. (2019) propose the model GraphWriter which improves on the graph attention networks presented in Velickovic et al. (2018) by replacing self-attention encoder with Transformer blocks (Vaswani et al., 2017). Several recent works have shown that pretrained generative models can be finetuned with or without structure-aware graph encoding to improve graph-to-text generation (Mager et al., 2020; Ribeiro et al., 2020; Hoyle et al., 2020; He et al., 2020; Ke et al., 2021). Query or question generation has also been shown to benefit from knowledge graphs in Shen et al. (2022); Bi et al. (2020). We combine the structure-aware encoding capabilities of graph-to-text models with the rich contextual knowledge of pretrained models in GraphGPT2 and generate rich real-world scenarios from sparse sub-graphs (Shen et al., 2022; Chen et al., 2020; Kumar et al., 2019).

Good distractors are necessary for a task model to learn the right reasoning towards answering multiple-choice datasets. To this end, Liang et al. (2018) rank distractors using feature-based ensemble methods. Offerijns et al. (2020); Yang et al. (2020) finetune GPT2 to generate distractors. Chung et al. (2020) approach distractor generation as a coverage problem and select distractors for maximizing sample difficulty. Cai and Wang (2018) use adversarial training to sample high quality negative training examples for knowledge graph embeddings. In a similar line of work, we use generative adversarial networks (GANs) (Goodfellow et al., 2014) with the Gumbel-Softmax relaxation (Kusner and Hernández-Lobato, 2016; Nie et al., 2019b) and train a generator with GAN loss to imitate the creation of human-authored tricky, incorrect answer options. Most NLP applications use REINFORCE (Sutton et al., 2000) algorithm and its variants (Yu et al., 2017; Cai and Wang, 2018; Qin et al., 2018; Zhang et al., 2018) to circumvent the discrete sampling issue for text-based GANs.

3 Methods

In this section, we describe the various modules in the GRADA framework.

3.1 Graph-to-Text Generation

In the first module of our pipeline, we generate synthetic questions by using knowledge graphs as input. Given a dataset of input graphs (g_i), we finetune GPT2 with cross-entropy loss for conditional generation of questions (q_i) from the graphs i.e., $L_q = \sum_{i=1}^{N} log\, p(q_i|f(g_i))$, where $f(.)$ is the function for encoding the graph and $p(.)$ represents the probabilities. We explore linearized graph encoding as well as structure-aware encoding of graph.

Linearized Graph Input. Graph linearization is a simple way to use graphs like text when finetuning GPT2. We adopt depth-first-search to linearize the input graphs and preserve edge information to some extent by augmenting GPT2 vocabulary with special tokens for edges. GPT2 is finetuned for conditional generation of target question from this linearized graph input.

Using linearized graphs with pretrained language models (PTLMs) surpasses graph-based architectures at data-to-text generation by a large margin (Ribeiro et al., 2020). However, Mager et al. (2020) show that omitting the edge information from linearized graphs notably degrades performance, implying that graph structure is beneficial for generation. Hence, we propose GraphGPT2.

GraphGPT2 for Structure-aware Graph Input. Instead of linearizing the input graph, we encode the graph using a Transformer-based graph encoder $f_s(.)$ which preserves the graph structure by performing masked self-attention over edges and nodes. We use the Transformer-based graph encoder from Graph Writer (Koncel-Kedziorski et al., 2019) for structure-preserving encoding of graphs. First, we convert the input graphs g_i into unlabeled connected bipartite graphs $G_i = (v_i, e_i)$, where v_i is the list of entities, relations and global vertex, and e_i is the adjacency matrix describing the directed edges (Beck et al., 2018). The global vertex is connected to all entity vertices and promotes global context modelling by allowing information flow between all parts of the graph. Next, v_i is projected to a dense, continuous embedding space V_i and is sent as input to the graph encoder (see Figure 2). The encoder is composed of L stacked Transformer blocks; each Transformer block consists of a N-headed self-attention layer followed by normalization and a two-layer feed-forward network. The resulting encodings i.e. $f_s(g_i)$, are referred to as graph contextualized vertex encodings. These encodings are prepended to the embedded representation of linearized graph in the form of past key values, and sent as input to the decoder. The decoder i.e., pretrained GPT2, is finetuned to generate a coherent question from the combined embeddings. The graph encoder is initialized with GPT2 embeddings to force continuity in word representation across modules. Figure 2 shows the integration of graph contextualized encodings with

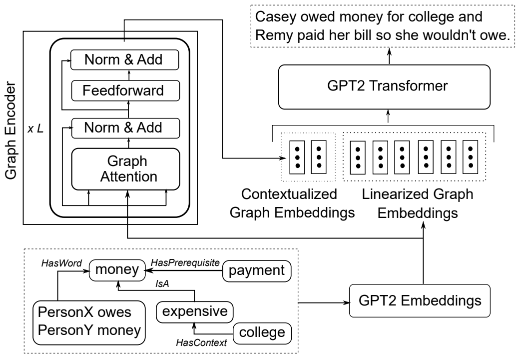

Figure 2: GraphGPT2: The Graph Encoder is composed of L Transformer blocks and its output is concatenated with GPT2 embeddings for input to GPT2.

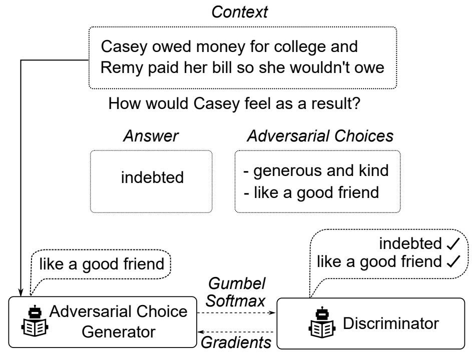

Figure 3: Training process for OptionGAN.

GPT2 in GraphGPT2. The combined generative model is finetuned end-to-end for maximizing the conditional log-likelihood of target question q_i i.e. $L_q = \sum_{i=1}^{N} log\, p(q_i \mid [f_l(g_i); f_s(g_i)])$, where $f_l(.)$ represents the linearized graph embeddings.

During inference, both of the above models are provided with graphs that do not appear in training dataset to generate synthetic questions containing new knowledge. See Sec. 4.1 for details on creation of training and inference datasets.

3.2 Answer & Distractor Generation

We finetune a GPT2 model for conditional generation of answers from questions i.e., $L_a = \sum_{i=1}^{N} log\, p(a_i|q_i)$. However, as we discussed in Sec. 1, a similar method for conditional generation of distractors does not guarantee good distractors. Hence, we finetune GPT2 within a GAN framework to generate maximally adversarial distractors, in a bid to imitate the best human annotator.

OptionGAN for Adversarial Choices. We train a model to generate distractors (in the multiple-choice QA task) for the synthetic questions obtained from GraphGPT2 (see Figure 1) using a generator-discriminator adversarial framework. The discriminator D is a sequential classification model that takes the question q_i, concatenated with the ground truth correct answer a_i i.e., $[q_i; a_i]$ or the distractor $\hat{d}_i$ generated by generator G i.e., $[q_i; \hat{d}_i]$ as input and classifies the pair as correct or otherwise. While training, the generator runs the risk of learning to generate correct answers instead of distractors, since it's goal is to be able to fool the discriminator into classifying the question-distractor pair $[q_i; \hat{d}_i]$ as correct. To prevent this, we heavily bias the model by first pretraining it to generate only distractors using the conditional cross-entropy loss and then continue with adversarial training from the saved weights. Mathematically, we pretrain the generator G with the loss $L_g = \sum_{i=1}^{N} log\, p(d_i|q_i)$, where q_i, d_i are question and distractor, respectively. We use the question as input instead of the knowledge sub-graph, since most generated questions contain additional semantics from the latent knowledge of the pretrained generative model which is not present in the original sub-graph. Then, the pretrained generator is finetuned within an adversarial framework to produce distractors that successfully fool the discriminator, so that we get adversarial options that are as tricky as human-annotated options (see Figure 3). We use the Gumbel-Softmax relaxation (Nie et al., 2019a) while sampling from generator to allow flow of gradients through the discriminator model i.e. $z = softmax(\frac{1}{\tau}(h+g))$, where h, g and τ are the logits generated from G, Gumbel distribution sample and temperature respectively. The temperature is annealed using an exponential function during training. Following RelGAN (Nie et al., 2019a), we use the Relativistic standard GAN loss for the adversarial training i.e. $\min_{G} \max_{D} log\, sigmoid(D([q_i; a_i]) - D([q_i; \hat{d}_i]))$. Generator G is trained to minimize the loss while discriminator D is trained to maximize the loss. In practice, we use GPT2 for both roles i.e., generator as well as discriminator.

3.3 Filtering and Selection of Samples

Inspite of the careful construction of synthetic samples using knowledge graphs, the pool of synthetic samples can be noisy and may consist of incoherent text, incorrect question-answer pairs or out-of-distribution samples. Hence, we use Question Answering Probability (QAP) (Zhang and Bansal, 2019) to measure accuracy of synthetic samples.

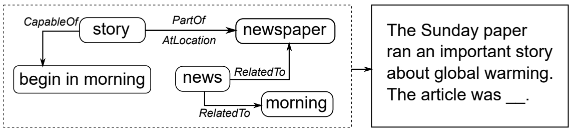

Figure 4: Example of synthetic context generated from GraphGPT2 for the CODAH dataset.

The QAP score (μ) is the prediction probability of the true class by a model with parameters θ which has been trained on the original dataset i.e. $\mu_i = p_\theta(a_i|x_i)$. Samples with low prediction probabilities for the correct choices are either annotated incorrectly or are especially difficult instances for the model. We define a low and high threshold for the QAP filter and samples lying within this range are retained in the dataset.

See supplementary for a comparison of QAP with two other methods for filtering i.e. Energy (Liu et al., 2020) and Model Confidence & Variability (Swayamdipta et al., 2020).

4 Experimental Setup

4.1 Datasets

SocialIQA (Sap et al., 2019b) and CommonsenseQA (Talmor et al., 2019) are annotated using knowledge graphs, making them a suitable choice for testing our approach. SocialIQA is a question answering dataset based on ATOMIC (Sap et al., 2019a), containing 33,410/1954/2224 samples in training, development and test set, resp. CommonsenseQA (CQA) is a similarly crowd-sourced dataset based on ConceptNet (Speer et al., 2017) containing an official split of 9741/1221/1241 samples. Following Yang et al. (2020), we also test our method on HellaSwag-2K (Zellers et al., 2019) and CODAH (Chen et al., 2019) for low-resource scenario. HellaSwag-2K is created by sampling 2000/1000/1000 examples from HellaSWAG training and validation sets. We test our approach on the CoDAH folds (2.8k samples) released by Yang et al. (2020) for comparison. Apart from these four MCQ datasets, we also experiment with the generative QA dataset ProtoQA (9762/52/102) (Boratko et al., 2020) and find that our approach works especially well with it. See Appendix for details.

Data Preparation. To prepare graph-to-text datasets for training GraphGPT2, we map the questions to multi-hop paths in ConceptNet (Bauer et al., 2018). We use Spacy[2] to tag the questions with part-of-speech and extract verbs and nouns as concepts, retaining those that appear in ConceptNet as entities and the connecting relations (see example in Fig. 4).[3] We remove inverse relations from the set of triples. The graphs extracted in this manner are acyclic and can be linearized with a depth-first search. For SocialIQA, we map the questions to a combination of ATOMIC and ConceptNet. ATOMIC events contain nouns and verbs which are representative of the social scenario being described in the event and are further extended in the context by SocialIQA annotators. We tokenize and stem the events and contexts to extract these representative words, and compute the percentage of overlapping words in the context with respect to each event. The event with maximum overlap with context is selected as the corresponding ATOMIC subject. The ATOMIC relation is selected from the predefined map of ATOMIC relations to SocialIQA questions. This way, we recover the ATOMIC alignments of nearly 20,000 samples from training set of SocialIQA (88% acc.).

Generation of Synthetic Data. In order to prepare synthetic datasets, we create a dataset of unseen input graphs by mutating the graphs from training sets of graph-to-text datasets. One or two entities are replaced by a randomly selected entity (or relation-entity pair) with similar adjacency to other entities in the input graph, to create a mutated graph. The maximum sequence length of graph contextualized embeddings is set to 64, while that of GPT2 is set to 128. The synthetic dataset size (pre-filtering) is 100k/50k/10k/10k/50k for SocialIQA, CQA, HellaSwag-2K, Codah, and ProtoQA respectively. For generation of synthetic data for SocialIQA, we use the set of tuples from ATOMIC that do not appear in the original dataset. To prepare the synthetic dataset for CommonsenseQA, we select two adversarial choices from ConceptNet and two choices generated by OptionGAN. For ProtoQA, we find accurate answers by generating 30 sets of answers for each synthetic question, ranking the answer choices by frequency and retaining the ones that appear at least 5 times in the 30 sets. See example of synthetic context generation in Fig. 4.

Evaluation. To evaluate graph-to-text generation, we define an ORACLE score which measures the semantic relevance of synthetic question when

[2]https://spacy.io/

[3]We use the question concept present in CQA annotations as additional concept for the questions.

paired with the original answer options. We replace the original question in validation set samples with the synthetic question and re-evaluate models on this modified dataset. In addition, we adopt the following NLG metrics: BLEU-4 (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), CIDEr[4] (Vedantam et al., 2015) and BERTScore (F1 score) (Zhang et al., 2020). Models trained on the synthetic and original commmonsense reasoning datasets are evaluated using their respective task-specific accuracies (see Appendix). For ProtoQA, we report the accuracy in top-k answers where $k = 1, 3, 5$. We also perform human evaluation of the samples generated using GraphGPT2 and OptionGAN.

5 Results & Analysis

First, we present results from the complete GRADA framework followed by results from ablation experiments. Then, we discuss evaluation of the various generative models in GRADA using automated metrics as well as human annotators. Finally, we evaluate the robustness of models trained with and without GRADA to semantic adversaries and discuss upper bounds of our data augmentation pipeline. See Appendix for visualization of the quality of the synthetic datasets.

5.1 Data Augmentation Results

Results from the best GRADA model are presented in Table 1.[5] The baseline row represents results from the same task models used for GRADA but trained without any data augmentation i.e. T5-3B for ProtoQA and RoBERTa for all other datasets. We see 1-2% improvements over baseline across all multiple-choice datasets using GRADA. For the best GRADA models (selected using validation results), synthetic samples are generated from structured GraphGPT2 and OptionGAN, and filtered using QAP.[6] GRADA results in large improvements for ProtoQA i.e. 4-6% higher values on the Max Answers 1/3/5 metrics (see Appendix), suggesting the effectiveness of our approach for similar generative tasks. We see 0.3%, 0.3% and 0.26% improvement with GRADA over G-DAUG for CQA, Codah and HellaSwag-2K respectively. Our approach also performs similar to the Option Comparison Network in HyKAS (Ma et al., 2019) for CQA (row 3 in Table 1). Our approach is orthogonal to HyKAS, KG-Fusion as their instance-level approach retrieves information for each sample while GRADA augments knowledge on a global level.

Ablation results from the GRADA framework on validation sets are presented in Table 2. The first row of Table 2 presents results from baseline task models i.e., trained without data augmentation. Next, we compare results from two-stage training and see upto 1.7% ($p<0.05$ for all datasets) improvements (row 1 vs. 4 in Table 2) with the addition of synthetic data without filtering.[7] Using structured GraphGPT2 leads to 0.47% (p=0.043), 0.39% (p=0.078), 1.46% (p=0.12)[8] improvements over linearized GraphGPT2 for SocialIQA, CQA, ProtoQA and diminishing improvements for the smaller datasets. We see consistent but modest improvements which are not significant, from addition of distractors generated from OptionGAN. Even though improvements with OptionGAN are marginal, it is necessary for the completeness of the pipeline for synthetic generation. Next, adding filter to denoise the synthetic pool unequivocally improves results by large margins for all datasets except CQA. Filtering by QAP (row 5 in Table 2) provides additional benefit (p=0.069 and p=0.093 for SocialIQA and CQA, $p<0.05$ for other datasets) to downstream task models over unfiltered synthetic data augmentation (row 4).[9] See examples of high and low quality synthetic data samples filtered using QAP in Table 7. Smaller datasets benefit the most from GRADA.

Single-hop vs. Multi-hop Paths. Additionally, we finetune GraphGPT2 with sub-graphs made of single-hop paths only to generate the context. We perform data augmentation using the synthetic questions generated through this approach and compare to the GRADA results on validation sets. See results in Table 4. We observe 0.92%, 0.08%, 1.48% and 1.05% drops in performance for validation sets of SocialIQA, CQA, CODAH and HellaSwag respectively. The larger drops for smaller datasets suggest that multi-hop paths are effective in low-resource scenarios.

[4]https://github.com/Maluuba/nlg-eval

[5]It should be noted that the state-of-the-art UnifiedQA has 30x parameters in RoBERTa$_{\text{LARGE}}$

[6]ProtoQA is not a multiple-choice dataset, so OptionGAN is not used and we use sample perplexity as the only filter.

[7]Statistical significance is computed with 100K samples using bootstrap (Noreen, 1989; Tibshirani and Efron, 1993).

[8]p-values are larger for improvements on ProtoQA validation set which has only 52 samples.

[9]We also ran experiments with MLM pretraining (ATOMIC for SocialIQA and OMCS corpus for the rest) before finetuning on target dataset and saw <1% improvements.

Method	SocialIQA	CQA	Codah	HellaSwag-2K	ProtoQA
UnifiedQA-11B (Khashabi et al., 2020)	81.45	79.1	-	-	41.49 / 24.95 / 21.77
RoBERTa + KG Fusion (Mitra et al., 2019)	78.00	-	-	-	-
RoBERTa + HyKAS (Ma et al., 2019)	-	73.2			
BACKTRANSLATION (Yang et al., 2020)		70.2	81.8		-
G-DAUG (Yang et al., 2020)	-	72.6	84.3	75.70	-
Baseline* (No Augmentation)	76.74	72.1	82.3	73.40	35.77 / 43.81 / 49.88
GRADA	77.85	72.9	84.7	75.96	42.02 / 48.90 / 54.23

Table 1: Results on test sets of commonsense datasets and comparative results from other approaches taken from leaderboards. *We use T5-3B for ProtoQA baseline and GRADA results and RoBERTa for all other datasets.

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	77.78	77.23	84.48	75.10	41.1
Synthetic Data Augmentation					
Linearized	78.21	77.55	86.07	76.40	45.63
+ Structured	78.68	77.94	86.13	76.70	46.09
+ OptionGAN	78.82	78.02	86.19	76.70	-
Filtering					
QAP*	79.12	78.06	86.81	77.60	50.34

Table 2: Ablation results on validation set of commonsense reasoning datasets. *We use sample perplexity for filtering ProtoQA samples.

Dataset	Original	GraphGPT2	
		Linearized	*Structured*
SocialIQA	75.92	55.18	57.34
CQA	77.23	57.63	58.71
CODAH	82.19	46.23	46.78
HellaSWAG-2K	76.58	41.35	41.74
ProtoQA	41.10	28.21	23.47

Table 3: ORACLE scores for question generation. Original represents the performance of baseline task models on original dataset. The columns GPT2 and GraphGPT2 represent similar evaluation with synthetic questions generated from linearized graphs and structure-aware graph encoder respectively.

Generalization to Unseen Concepts. We looked for %overlap of entity nodes and single-hop paths (subject– relation– object) between the multi-hop KGs spanning the questions of correctly answered samples after GraDA training and the questions of synthetic data, and observed 5-60% entity overlap and <20% path overlap. This suggests GRADA also promotes reasoning capabilities of the downstream models for unseen concepts.

5.2 Generative Model Evaluation Results

ORACLE scores for the two variations of GraphGPT2 are presented in Table 3. The scores in first column refer to the validation set performance of baseline models on original datasets. These models are re-evaluated on the questions generated by GraphGPT2 (as described in Sec. 4.1). The largest improvement i.e. 2.16% (p=0.068) is observed for SocialIQA, which may be attributed to

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	77.78	77.23	84.48	75.10	41.1
GraDA (single-hop)	78.70	77.31	85.96	76.05	45.67
GraDA (multi-hop)	79.12	78.06	86.81	77.60	50.34

Table 4: Results on validation set of commonsense reasoning datasets using single-hop vs. multi-hop graphs for GRADA pipeline.

Dataset	Question	Answer	Distractors
SocialIQA	96.1%	86.0%	50.0%
CommonsenseQA	100.0%	97.2%	25.0%
HellaSwag-2K	92.0%	88.1%	25.8%
CODAH	90.3	83.4%	30.6%
ProtoQA	97.2%	75.0%	-

Table 5: Results from human evaluation of generated questions, answers and distractors.

its large dataset size. We see diminishing improvements for low-resource scenarios i.e. Codah and HellaSwag-2K. We observe a similar trend when the synthetic questions are evaluated using NLG metrics (see Appendix). More importantly, since phrase-matching metrics are not ideal for NLG evaluation (Novikova et al., 2017), we also perform human evaluation to judge the quality of generation for SocialIQA and CQA as we see significant improvements from structured GraphGPT2 vs. linearized GraphGPT2. We ask annotators on Amazon Mechanical Turk[10] (AMT) to select the sentence which is more representative of the information encoded in input graph, for 100 samples from validation set. Questions generated from GraphGPT2 are preferred 46% and 53% of the times for SocialIQA and CQA resp., compared to those from linearized inputs only, showing that the addition of graph encoder improves integration of knowledge in generated text.

We perform human evaluation (AMT) of answerability of the generated questions/answers/distractors on 50 randomly selected samples from the filtered augmentation

[10]Located in United States, HIT Approval Rate>98%, Number of HITs Approved>10K, $15 per hour (approx.).

G-Daug (Yang et al., 2020)	GRADA	
	Knowledge-Graph Tuple	**Generated Data**
A human enjoys putting rubber on furniture. They should do this before .. front of the mirror.	*S*: PersonX provides __ for PersonY's children	Taylor provided meals for Kendall's children and they all enjoyed it greatly.
There was a large, cold bite of ice on my where?	*R*: xIntent	Why did Taylor do this?
He hated flying, the controls were what?	*O*: To be helpful	[A] to be a bad friend [B] to be helpful [C] to be rude
What is a square leg made of made out of?	*S*: weasel *R*: AtLocation	The man was a weasel, he was part of a powerful what?
What country does a cow go to make a milk run?	*O*: mafia organization	[A] out of doors [b] terrarium [c] mafia organization [D] farmyard [E] backyard

Table 6: Comparison of randomly generated synthetic data from G-Daug (Yang et al., 2020) (left) and knowledge-grounded synthetic data generated using GRADA (right). (S=Subject, R=Relation, O=Object)

High-quality synthetic samples	
SIQA	Riley provided help to the community through his many charity events over the years. How would Others feel as a result? [A] selfish [B] appreciative [C] bored
CQA	When a child is upset by something, what may they do? [A] fall down [B] wish to fly [C] start crying [D] play tag [E] boy or girl
PQA	Name something you worry you're still doing when you're not supposed to. drinking, smoking, sleeping, working, using cell phone
Low-quality synthetic samples	
SIQA	Tracy raised her arm to her face to cover her eyes during the scary movie. What does Tracy need to dobefore this? [A] scared [B] be scared of the movie [C] to have a fundraiser
CQA	What will you do if you want to go public? [A] prepare for worst [B] tell family first [C] own private company [D] telegram [E] charming
PQA	Name a family tradition that has deep roots in the dialect of suzh. cooking, caroling, knitting, hunting, fishing

Table 7: High and low quality synthetic samples generated through GRADA for SIQA, CQA, ProtoQA (PQA) and ranked using QAP scores (and perplexity for PQA). Labels are marked in green.

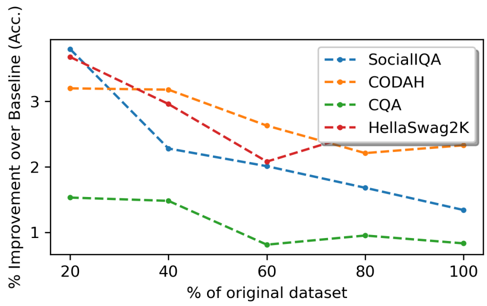

Figure 5: % improvement in accuracy over baseline with different % of original dataset. Baseline is RoBERTa finetuned on the same % of original dataset.

data (see Table 5). Annotators were provided with the question, answer and distractors, and asked to evaluate a) if the question can be answered in a few words (b) if the question can be answered by the given answer and (c) if the distractors are wrong answers for the question. More than 90% of the questions were judged as answerable, 75-90% of the answers were judged as correct answers for the respective questions. The quality of distractors ranged from 50% for SocialIQA to 20-30% for smaller datasets. However, the overall quality of distractors is high enough to benefit data augmentation. See examples in Table 7. We also perform human evaluation for the factuality of samples generated using our method GraDA and GDaug (Yang et al., 2020). We picked a randomly sampled set of 100 synthetic QA pairs from G-Daug for the datasets CQA, Codah and HellaSWAG-2K. For a fair comparison, we collected 100 synthetic pairs from GraDA for the same datasets. We asked an annotator to evaluate if each of the synthetic QA pair adheres to a plausible real-world scenario, and found that 56% G-Daug samples were judged as factual as compared to 68% of the GraDA samples (see examples in Table 6).

5.3 Upper Bounds

We ran experiments for augmentation with 20%, 40%, 60%, 80% and 100% training data from the original set (see Fig. 5). The improvement margins from the augmentation dataset is upto 4% at 20% of the original SocialIQA dataset. We see similar trends for CODAH, HellaSwag and ProtoQA, while the improvements for CQA were <1.5%.

5.4 Robustness Evaluation

We expect that data augmentation exposes the task model to diverse language and improves its robustness to semantic adversaries in addition to boosting its performance on the target task. To evaluate this, we use the TextFooler system (Jin et al., 2020; Yang et al., 2020; Wei and Zou, 2019) to generate adversarial text by computing word importance ranking and replacing the most influential words

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	21.7/10.3	14.9/12.5	31.3/16.1	19.4/10.6	5.1/16.2
GRADA	22.4/10.8	15.8/12.9	34.8/18.2	20.5/11.5	6.3/16.8

Table 8: Robustness Evaluation. Failure rate / perturbation ratio (higher is better) from TextFooler experiments are shown on development sets.

with their synonym in the vector space. Overall, GRADA benefits the robustness of task models and improves their failure rate by 1-3% (see Table 8).

6 Conclusion

We present GRADA, a graph-based data augmentation framework for commonsense reasoning QA datasets. We train a graph-to-text question generator and GAN-based adversarial choice generator for creating synthetic data samples, which are used to augment the original datasets. GRADA promotes factuality in synthetic samples and improves results on five downstream datasets.

7 Ethical Considerations

The usage of pretrained generative models in any downstream application requires careful consideration of the real-world impact of generated text. In our approach, we provide concrete inputs for grounding the generated text to specific entities and relations which encode real-world facts, thereby reducing the possibility of propagating unintended stereotypical and social biases embedded within the pretrained models. However, since these entities and relations are derived from existing knowledge bases like ConceptNet (Speer et al., 2017), there is potential for transfer of bias present in these resources to the generated texts. Additionally, the graph-to-text generative models in GRADA pose the same risk as other data-to-text generative models (Ribeiro et al., 2020; Hoyle et al., 2020; Mager et al., 2020) i.e. the models can be made to generate incorrect facts by providing incorrect data as input. Therefore, we recommend restricting the use of GRADA to low-risk, unbiased graphs inputs.

Acknowledgments

We thank the reviewers for their useful feedback. This work was supported by DARPA MCS Grant N66001-19-2-403, ONR Grant N00014-18-1-2871, and NSF-CAREER Award 1846185. The views are those of the authors and not of the funding agency.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2020. Not enough data? deep learning to the rescue! In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *Proceedings of the 2018 Conference on EMNLP*, pages 4220–4230.

Daniel Beck, Gholamreza Haffari, and Trevor Cohn. 2018. Graph-to-sequence learning using gated graph neural networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 273–283.

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 *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Sheng Bi, Xiya Cheng, Yuan-Fang Li, Yongzhen Wang, and Guilin Qi. 2020. Knowledge-enriched, type-constrained and grammar-guided question generation over knowledge bases. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2776–2786.

Yonatan Bisk, Rowan Zellers, Ronan Le Bras, Jianfeng Gao, and Yejin Choi. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3027–3035.

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.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779.

Liwei Cai and William Yang Wang. 2018. Kbgan: Adversarial learning for knowledge graph embeddings. In *Proceedings of NAACL-HLT*, pages 1470–1480.

Michael Chen, Mike D'Arcy, Alisa Liu, Jared Fernandez, and Doug Downey. 2019. Codah: An adversarially authored question-answer dataset for common sense. In *Proceedings of the 3rd Workshop on Evaluating Vector Space Representations for NLP*.

Yu Chen, Lingfei Wu, and Mohammed J Zaki. 2020. Toward subgraph guided knowledge graph question generation with graph neural networks. *arXiv preprint arXiv:2004.06015*.

Ho-Lam Chung, Ying-Hong Chan, and Yao-Chung Fan. 2020. A bert-based distractor generation scheme with multi-tasking and negative answer training strategies. *arXiv preprint arXiv:2010.05384*.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2019. Question answering by reasoning across documents with graph convolutional networks. In *Proceedings of NAACL-HLT*, pages 2306–2317.

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.

Yanlin Feng, Xinyue Chen, Bill Yuchen Lin, Peifeng Wang, Jun Yan, and Xiang Ren. 2020. Scalable multi-hop relational reasoning for knowledge-aware question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1295–1309.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Catherine Havasi, Robert Speer, Kenneth Arnold, Henry Lieberman, Jason Alonso, and Jesse Moeller. 2010. Open mind common sense: Crowd-sourcing for common sense. In *Workshops at the Twenty-Fourth AAAI Conference on Artificial Intelligence*.

Bin He, Di Zhou, Jinghui Xiao, Xin Jiang, Qun Liu, Nicholas Jing Yuan, and Tong Xu. 2020. Integrating graph contextualized knowledge into pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2281–2290.

Alexander Hoyle, Ana Marasović, and Noah Smith. 2020. Promoting graph awareness in linearized graph-to-text generation. *arXiv preprint arXiv:2012.15793*.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. 2020. Is bert really robust? natural language attack on text classification and entailment. In *AAAI*.

Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. 2021. Jointgt: Graph-text joint representation learning for text generation from knowledge graphs. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2526–2538.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1896–1907.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*. OpenReview.net.

Rik Koncel-Kedziorski, Dhanush Bekal, Yi Luan, Mirella Lapata, and Hannaneh Hajishirzi. 2019. Text generation from knowledge graphs with graph transformers. In *Proceedings of the 2019 Conference of the NAACL: HLT, Volume 1 (Long and Short Papers)*, pages 2284–2293.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Vishwajeet Kumar, Yuncheng Hua, Ganesh Ramakrishnan, Guilin Qi, Lianli Gao, and Yuan-Fang Li. 2019. Difficulty-controllable multi-hop question generation from knowledge graphs. In *International Semantic Web Conference*, pages 382–398. Springer.

Matt J Kusner and José Miguel Hernández-Lobato. 2016. GANs for sequences of discrete elements with the gumbel-softmax distribution. *arXiv preprint arXiv:1611.04051*.

Chen Liang, Xiao Yang, Neisarg Dave, Drew Wham, Bart Pursel, and C Lee Giles. 2018. Distractor generation for multiple choice questions using learning to rank. In *Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications*, pages 284–290.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for 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 2822–2832.

Weitang Liu, Xiaoyun Wang, John Owens, and Yixuan Li. 2020. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 33.

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*.

Shangwen Lv, Daya Guo, Jingjing Xu, Duyu Tang, Nan Duan, Ming Gong, Linjun Shou, Daxin Jiang, Guihong Cao, and Songlin Hu. 2020. Graph-based reasoning over heterogeneous external knowledge for commonsense question answering. In *Proceedings of the Thirty-Fourth Conference on Association for the Advancement of Artificial Intelligence (AAAI)*.

Kaixin Ma, Jonathan Francis, Quanyang Lu, Eric Nyberg, and Alessandro Oltramari. 2019. Towards generalizable neuro-symbolic systems for commonsense question answering. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 22–32.

Manuel Mager, Ramón Fernandez Astudillo, Tahira Naseem, Md Arafat Sultan, Young-Suk Lee, Radu Florian, and Salim Roukos. 2020. Gpt-too: A language-model-first approach for amr-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1846–1852.

Diego Marcheggiani and Laura Perez-Beltrachini. 2018. Deep graph convolutional encoders for structured data to text generation. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 1–9.

Arindam Mitra, Pratyay Banerjee, Kuntal Kumar Pal, Swaroop Mishra, and Chitta Baral. 2019. Exploring ways to incorporate additional knowledge to improve natural language commonsense question answering. *arXiv preprint arXiv:1909.08855*.

Weili Nie, Nina Narodytska, and Ankit Patel. 2019a. Relgan: Relational generative adversarial networks for text generation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Weili Nie, Nina Narodytska, and Ankit Patel. 2019b. Relgan: Relational generative adversarial networks for text generation. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Eric W Noreen. 1989. *Computer-intensive methods for testing hypotheses*. Wiley New York.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. 2017. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark. Association for Computational Linguistics.

Jeroen Offerijns, Suzan Verberne, and Tessa Verhoef. 2020. Better distractions: Transformer-based distractor generation and multiple choice question filtering. *arXiv preprint arXiv:2010.09598*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mostofa Patwary, Mohammad Shoeybi, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. *arXiv preprint arXiv:2002.09599*.

Pengda Qin, Weiran Xu, and William Yang Wang. 2018. Dsgan: Generative adversarial training for distant supervision relation extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 496–505.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *OpenAI Technical Report*.

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

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*, 21:1–67.

Leonardo FR Ribeiro, Martin Schmitt, Hinrich Schütze, and Iryna Gurevych. 2020. Investigating pretrained language models for graph-to-text generation. *arXiv preprint arXiv:2007.08426*.

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.

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 EMNLP and the 9th IJCNLP*, pages 4453–4463.

Xinyao Shen, Jiangjie Chen, Jiaze Chen, Chun Zeng, and Yanghua Xiao. 2022. Diversified query generation guided by knowledge graph. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 897–907.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. 2012. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. 2000. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Swabha Swayamdipta, Roy Schwartz, Nicholas Lourie, Yizhong Wang, Hannaneh Hajishirzi, Noah A Smith, and Yejin Choi. 2020. Dataset cartography: Mapping and diagnosing datasets with training dynamics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9275–9293.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826.

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 NAACL: HLT, Volume 1 (Long and Short Papers)*, pages 4149–4158.

Robert J Tibshirani and Bradley Efron. 1993. An introduction to the bootstrap. *Monographs on statistics and applied probability*, 57:1–436.

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

Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Jason Wei and Kai Zou. 2019. Eda: Easy data augmentation techniques for boosting performance on text classification tasks. 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 6383–6389.

Yiben Yang, Chaitanya Malaviya, Jared Fernandez, Swabha Swayamdipta, Ronan Le Bras, Ji-Ping Wang, Chandra Bhagavatula, Yejin Choi, and Doug Downey. 2020. G-daug: Generative data augmentation for commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1008–1025.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5754–5764.

Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. 2019. Align, mask and select: A simple method for incorporating commonsense knowledge into language representation models. *arXiv preprint arXiv:1908.06725*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. 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 2495–2509.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Zhirui Zhang, Shujie Liu, Mu Li, Ming Zhou, and Enhong Chen. 2018. Bidirectional generative adversarial networks for neural machine translation. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 190–199. Association for Computational Linguistics.

A Experiment Setup

Datasets: Social IQA (Sap et al., 2019b) and CommonsenseQA (Talmor et al., 2019) are popular datasets based on knowledge graphs, making them a suitable choice for testing our approach. Social IQA is a multiple-choice question answering dataset. Each sample consists of a context, question and three multiple choices. CommonsenseQA is also a multiple-choice QA dataset, wherein each sample consists of a context and five multiple choices. Of those 5 choices, three are taken from ConceptNet and the other two are authored by annotators. We only use the human-authored incorrect choices to train our adversarial choice generator OptionGAN. The ATOMIC knowledge graph contains 24K base events and 877K tuples describing a variety of social scenarios. We use the 710K training split introduced in Bosselut et al. (2019) to randomly sample 100K tuples as the seed subgraphs for generation of synthetic data dataset for Social IQA. For CommonsenseQA, we use the entire ConceptNet knowledge graph, subject to pruning as outlined in Talmor et al. (2019), to sample seed tuples for synthetic dataset generation. For SocialIQA, CQA, Codah and HellaSwag-2K, we use simple accuracy for model evaluation.

ProtoQA (Boratko et al., 2020) is a generative QA dataset which is evaluated by 7 different metrics[11]. We report the first 3 metrics i.e. Max Answers 1/3/5. For tables showing only one number for ProtoQA, such as the ablation table in main text, we report the Max Answer 1 metric. In order to train T5-3B for ProtoQA, we concatenate the ranked choices for each question and finetune the model for conditional generation of this concatenated string from the input question.

All of the above datasets are being for their intended purposes i.e. research only, in our work. All of these datasets are in the English language.

Data Preparation: To prepare graph-to-text datasets for training GraphGPT2, we map the questions to multi-hop paths in ConceptNet (Bauer et al., 2018). We use Spacy[12] to tag the questions with part-of-speech and extract verbs and nouns as concepts, retaining those that appear in ConceptNet as entities[13]. For SocialIQA, we map the questions to a combination of ATOMIC and ConceptNet. ATOMIC events contain nouns and verbs which are representative of the social scenario being described in the event and are further extended in the context by Social IQA annotators (see Table 6). We tokenize and stem the events and contexts to extract these representative words, and compute the percentage of overlapping words in the context with respect to each event. The event with maximum overlap with context is selected as the corresponding ATOMIC subject. The ATOMIC relation is selected from the predefined map of ATOMIC relations to Social IQA questions. This way, we recover the ATOMIC alignments of 20,000 samples from training set of SocialIQA with 88% accuracy.

Synthetic Data Generation. In order to prepare synthetic datasets, we create a dataset of unseen input graphs by mutating the graphs from training sets of graph-to-text datasets. One or two entities are replaced by a randomly selected entity (or relation-entity pair) with similar adjacency to other entities in the input graph, to create a mutated graph. The synthetic dataset size (pre-filtering) is 100k/50k/10k/10k/50k for SocialIQA, CQA, HellaSwag-2K, Codah, and ProtoQA respectively. For generation of synthetic data, we use the set of tuples from ATOMIC and ConceptNet that do not appear in SocialIQA and CommonsenseQA datasets respectively. To prepare the synthetic dataset for CommonsenseQA, we select two adversarial choices from ConceptNet and two choices generated by OptionGAN. For ProtoQA, we find accurate answers by generating 30 samples of answers for each synthetic question, ranking the answer choices by frequency and retaining the ones that appear atleast 5 times in the 30 samples. After this, the synthetic question and answer (concatenation of high-frequency answer choices) is subjected to filtering. Due to lack of option for supplementary in this submission, we have included a sample of the generated synthetic examples in Table 9.

A.1 Filtering and Selection of Samples

Inspite of the careful construction of synthetic samples using knowledge graphs, the pool of synthetic samples can be noisy and may consist of incoherent text, incorrect question-answer pairs or out-of-distribution samples. Hence, we compare the effect of three different methods to filter samples on downstream task performance.

Question Answering Probability (QAP). The QAP score (μ) (Zhang and Bansal, 2019) is the prediction probability of the true class by a model with parameters θ which has been trained on the original dataset i.e. $\mu_i = p_\theta(y_i^*|x_i)$. Samples with low prediction probabilities for the correct choices are either annotated incorrectly or are especially difficult instances for the model. We define a low and high threshold for the QAP filter and samples lying within this range are retained in the dataset.

Model Confidence and Variability. Swayamdipta et al. (2020) propose the model confidence ($\hat{\mu}_i$) and variability ($\hat{\sigma}_i$) measures to identify

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

[12] https://spacy.io/

[13] We use the question concept present in CQA annotations as additional concept for the questions.

HellaSWAG-2K	
Question	*Answer*
A close up of a gymnast is shown. a gymnast balances on beam as she sweeps __	(a) over obstacles. (b) around with other gymnast. (c) performs a front squat and a flip, and crosses her arms. (d) performing multiple back and forth moves.
"We then search for a car by its model and make. Once we get the car model __	(a) we determine what the tires are for. (b) we either buy a new or recycle it. If we want to recycle the car, we simply (c) click the buy now button. The seller will then provide a description of the car and (d) we'll add it to the computer so we can make a list of the different models we'll
A man in black robes is walking into a bar. He __	(a) is telling several anecdotes about how he has been following other people around and talking to them. (b) speaks to a group of workers and they all rise and raise their arms in the air. (c) starts singing into the microphone. (D) begins a beat down on a man standing behind him.
CODAH	
Question	*Answer*
I am feeling very hungry. I think that __	(a) I will have dinner. (b) I will drink some milk. (c) I will sleep a lot. (d) I will play catch with my grandpa.
A man with no body hair was peacefully wallowing in the sea of ocean. The man then __	(a) was surrounded by a flock of birds. (b) hung from the ceiling and sang (c) began to carpet the beach. (d) watched a movie with his headphones on.
A man excitedly planned a surprise party for his friend. He __	(a) got a shotgun. (b) put up a giant neon sign with his own hand painted on it. (c) decided to end his life in front of his friend. (d) planned to brew a cup of coffee and play chess.
ProtoQA	
Question	*Answer*
Name something you worry you're still doing when you're not supposed to.	drinking, smoking, sleeping, working, using cell phone
Besides milk, name a popular product in the dairy market.	cheese, ice cream, yogurt, butter
Name something you can disagree about.	religion, politics, parenting, weight, money
If you sent a postcard from china what would be pictured on the front?	great wall, temple, dragon
Name something a knight needs for a good day's work.	horse, armour, sword, lance, shield

Table 9: Examples of synthetic samples generated for HellaSWAG-2K, CODAH and ProtoQA datasets from the GRADA pipeline. Correct answers for multiple-choice questions are marked in green.

the effect of data samples on the model's generalization error. Specifically, $\hat{\mu_i} = \frac{1}{E}\sum_{e=1}^{E} p_\theta(y_i^*|x_i)$ and $\hat{\sigma_i} = \sqrt{\frac{\sum_{e=1}^{E}(p_\theta(y_i^*|x_i)-\hat{\mu_i})^2}{E}}$, where E is training epochs. They find that ambiguous samples i.e., high variability and mid-range confidence, contribute the most to test performance on downstream task. Following this, we define low and high thresholds for both confidence and variability in order to find the most informative samples.

Energy. Liu et al. (2020) show that the energy score can be reliably used for distinguishing between in- and out-of-distribution (OOD) samples, as compared to the traditional approach of using the softmax scores. We introduce an energy threshold to select samples which are out-of-distribution i.e. $E_i = -log\sum_j^C e^{p_\theta(y_i^j|x)}$ where C is the number of choices in the QA sample, and measure the effect of using OOD samples as augmentation data.

A.2 Training Details

Baselines: We use pretrained RoBERTa$_{\text{LARGE}}$ (Liu et al., 2019) for multiple-choice datasets and T5-3B (Raffel et al., 2020) for ProtoQA as the task models. The baseline task model is finetuned on original datasets with no data augmentation, and is used as scoring model for filtering. We use GPT2$_{\text{MEDIUM}}$ for GraphGPT2, GPT2$_{\text{SMALL}}$ as the pretrained generator and discriminator for OptionGAN. For GRADA, the model is first finetuned on synthetic samples using label smoothing (Szegedy et al., 2016) and then on original dataset. We refer the reader to Koncel-Kedziorski et al. (2019) for full implementation details of the Graph Encoder in GraphGPT2.

OptionGAN: It is tricky to train GAN models, especially with discrete data like text. We follow the training method in Nie et al. (2019a) to finetune the adversarial choice generator in a minimax

Parameter	Bounds
Filter Parameters	
QAP/Model Confidence Lower Threshold	[0.0, 0.49]
QAP/Model Confidence Higher Threshold	[0.51, 1.0]
Energy Lower Threhsold	[0.0, 1.0]
Energy Higher Threshold	[0.0, 1.0]
Model Variability Lower Threshold	[0.0, 0.5]
Model Variability Higher Threshold	[0.0, 0.5]
Training Parameters	
Learning Rate	[1, 10]*1e-6
Batch Size (*inc*)	[4, 8, 16]
Total Train Epochs	[3, 5]

Table 10: Optimization bounds for grid-search based tuning of training hyperparameters.

Method	BLEU4	METEOR	CIDEr	BERTScore
Social IQA				
GPT2	14.58	26.41	132.84	89.12
GraphGPT2	15.37	26.95	135.91	91.83
CommonsenseQA				
GPT2	1.71	12.78	30.89	85.76
GraphGPT2	1.90	13.64	33.76	87.34

Table 11: Comparison of performance for GPT2 and GraphGPT2 on development sets.

game with discriminator. In addition to the training parameters mentioned in Table 17, we restrict the number of training iterations to 5000, and perform one gradient descent step on generator for every 5 gradient descent steps on discriminator.

Training & Hyperparameter Tuning. After generation of synthetic examples, we perform two-stage training of the task models. In the first phase, the model is finetuned on synthetic data only, In the second phase, the model is finetuned on the original dataset. The model trained in first phase is subject to bayesian optimization (Snoek et al., 2012) of filter parameters.

A.3 Human Evaluation

Generative source of the sentences are omitted when presented to annotators. The input graphs are seed tuples from ATOMIC and ConceptNet for samples from the development sets of Social IQA and CommonsenseQA respectively. The annotators can pick both the sentences if either of them are equally relevant in their subjective opinion. We allow for a single hit for each assignment in Amazon Mechanical Turk.

Dataset	Wins	Loses	Tie
SocialIQA	46%	37%	17%
CommonsenseQA	53%	31%	16%

Table 12: Results from comparative human evaluation of generated questions. Wins and Loses refer to the %times synthetic question generated from structured graph input was chosen over linearized graph.

B Results

B.1 Generative Model Evaluation

As shown in Table 11, we see small improvements for BLEU-4 and METEOR, but larger improvements in other metrics from GraphGPT2 i.e., 3.07% (p=0.027), 2.87% (p=0.035) in CIDEr, and 2.71% (p=0.042), 1.58% (p=0.056) in BERTScore for Social IQA and CQA, resp. The phrase-matching metric scores are low for CQA, which may be attributed to its small sample size. However, BERTScore for CQA lies between 85-88%, showing that the model manages to convey similar meaning as human-annotated context albeit with different words.

More importantly, since phrase-matching metrics are not ideal for NLG evaluation (Novikova et al., 2017), we also perform human evaluation to judge the quality of generation for SocialIQA and CommonsenseQA as we see significant improvements from structured GraphGPT2 vs. linearized GraphGPT2. We ask annotators on Amazon Mechanical Turk[14] to select the sentence which is more representative of the information encoded in input graph, for 100 samples from validation set. Results are shown in Table 12. Samples generated from structured input are selected significantly more times than those from linearized inputs, for both SocialIQA and CQA, showing that addition of a graph encoder improves representation of knowledge in generated sample.

Additionally, we perform human evaluation of the samples generated using GraphGPT2 and OptionGAN. We randomly select 50 samples from the filtered augmentation datasets for each of the five datasets, and ask 2 annotators to answer 3 yes/no questions about the quality of the question, answer and distractors respectively. We present results from the survey in Table 5. More than 90% of the questions in each dataset were judged as answerable, showing the effectiveness of GraphGPT2 as well as the QAP-based filtering method. Simi-

[14]Located in United States, HIT Approval Rate>98%, Number of HITs Approved>10K.

Method	SIQA	CQA	CDH	H2K	PQA
Baseline	77.78	77.23	84.48	75.10	41.1
Filtering					
QAP*	79.12	78.06	86.81	77.60	50.34
Confidence	79.05	77.83	86.59	77.40	-
Energy	78.76	77.79	86.38	77.10	-

Table 13: Ablation results on validation set of commonsense reasoning datasets using various filtering methods. *We use sample perplexity for filtering ProtoQA samples.

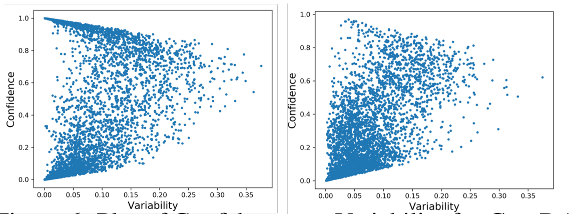

Figure 6: Plot of Confidence vs. Variability for GRADA synthetic samples for CQA (left) and H2K (right).

larly, 75-90% of the answers were judged as correct answers for the respective questions. The quality of distractors were relatively lower, ranging from 50% for larger datasets like SocialIQA to 20-30% for rest of the datasets. The inter-annotator agreement was also low (<0.6) for distractor judgements, suggesting the general difficulty of both tasks: distractor generation and measurement of distractor quality. However, the overall quality of distractors in our datasets is high enough to benefit data augmentation.

For both human evaluation annotation tasks, it was made clear in the instructions that the data is being collected for research purposes only.

B.2 Comparison of Filtering Methods

Table 13 demonstrates the effect of using various methods of filtering i.e. QAP, Energy and Model Confidence/Variability. Results are shown on the validation sets the commonsense reasoning datasets. We see largest improvements with using QAP as the filter. Similar improvements are seen with the confidence/variability scores; however, it requires scores from multiple finetuned models from various training checkpoints.

B.3 Robustness Evaluation

We expect that data augmentation exposes the task model to diverse language and improves its robustness to semantic adversaries in addition to boosting its performance on the target task. To evaluate this, we use the TextFooler system (Jin et al., 2020; Yang et al., 2020; Wei and Zou, 2019) to generate adversarial text by computing word importance ranking and replacing the most influential words with their synonym in the vector space. Failure rate is the %examples for which TextFooler fails to change the original model prediction, and average perturbation ratio is the average % of words replaced when TextFooler succeeds at changing the prediction. We use our best GRADA models in comparison with baseline models (Table 8). Overall, GRADA positively impacts the robustness of task models to TextFooler and improves the failure rate by >3% for Codah and upto 1% for all other datasets. We observe similar trends for the perturbation ratios too. This shows that GRADA improves semantic robustness of the models. It is also worthwhile noting that generative task models like T5-3B for ProtoQA are especially prone to adversarial attacks like TextFooler with a mere 5-6% failure rate and there needs to be more research towards improving their robustness.

B.4 Cartography Quality Evaluation

We use dataset cartography Swayamdipta et al. (2020) to visualize the quality of our synthetic datasets. Samples in top left of figure are easy, while samples towards bottom and right of the figure are difficult and ambiguous respectively. We can observe from the figure that the synthetic dataset for CQA (left) has a higher % of easy samples than HellaSwag-2K, suggesting that the quality of synthetic samples generated by GRADA improves with original dataset size. Moreover, when applying QAP filtering, using the entire synthetic dataset yields largest improvements for CQA whereas for HellaSwag-2K (right), the lower cutoff for QAP is 0.3 which filters out most of the samples present in bottom part of the plot. This suggests that in low-resource scenarios, it is important to remove inaccurate samples, while larger datasets benefit from ambiguous and inaccurate samples.

Best Parameters	Social IQA	CQA	Codah	HellaSwag-2K	ProtoQA
QAP Lower Threshold	0.49	0.32	0.43	0.49	0.27
QAP Higher Threshold	1.0	1.0	1.0	1.0	1.0

Table 14: Best Filter Hyperparameters.

	Social IQA			CommonsenseQA		
Hyperparameter	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**
Learning Rate	5e-6	4e-6	3e-6	1e-5	5e-6	1e-5
Epochs	3	1	3	5	1	5
Max Gradient Norm	1.0	1.0	1.0	None	None	None
Weight Decay	0.01	0.01	0.01	0.01	0.01	0.01
Batch Size	8	8	8	16	16	16
Max Length	128	128	128	70	70	70
Warmup Ratio	0.0	0.0	0.0	0.06	0.06	0.0
LR Decay	Linear	Linear	Linear	Linear	Linear	Linear
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Hardware	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti
Single GPU Hours	5 hrs	1.5 hrs	5 hrs	2 hrs	0.5 hrs	2 hrs

Table 15: Training hyperparameters for baseline and two-stage GRADA training of SocialIQA and CQA

	CODAH			HellaSwag-2K		
Hyperparameter	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**	**Baseline**	**GRADA Phase 1**	**GRADA Phase 2**
Learning Rate	1e-5	4e-6	3e-6	5e-5	5e-6	1e-5
Epochs	5	1	5	5	1	5
Max Gradient Norm	1.0	1.0	1.0	None	None	None
Weight Decay	0.01	0.01	0.01	0.01	0.01	0.01
Batch Size	16	8	16	8	8	8
Max Length	90	90	90	128	128	128
Warmup Ratio	0.06	0.06	0.06	0.06	0.06	0.06
LR Decay	Linear	Linear	Linear	Linear	Linear	Linear
Optimizer	AdamW	AdamW	AdamW	AdamW	AdamW	AdamW
Hardware	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti	RTX 2080Ti
Single GPU Hours	2 hrs*	1 hr* hrs	2 hrs*	0.5 hr	0.2 hr	0.5 hr

Table 16: Training hyperparameters for baseline and two-stage GraDA training of RoBERTa models for HellaSwag-2K and CODAH. *values reported for five-fold training

		OptionGAN		
Hyperparameter	**GraphGPT2**	**Generator**	**Discriminator**	**GAN**
Learning Rate	4e-5	1e-5	1e-5	1e-6
Epochs	5	5	3	-
Max Gradient Norm	1.0	1.0	1.0	None
Weight Decay	0.0	0.01	0.01	0.01
Batch Size	8	8	8	4
Max Length	128	128	128	128
Warmup Ratio	0.0	0.0	0.0	0.06
LR Decay	Linear	Linear	Linear	Linear
Optimizer	AdamW	AdamW	AdamW	AdamW

Table 17: Training hyperparameters for GraphGPT2, Generator, Discriminator and OptionGAN