Explanation Graph Generation via Pre-trained Language Models: An Empirical Study with Contrastive Learning

Swarnadeep Saha **Prateek Yadav** **Mohit Bansal**
UNC Chapel Hill
{swarna, prateek, mbansal}@cs.unc.edu

Abstract

Pre-trained sequence-to-sequence language models have led to widespread success in many natural language generation tasks. However, there has been relatively less work on analyzing their ability to generate structured outputs such as graphs. Unlike natural language, graphs have distinct structural and semantic properties in the context of a downstream NLP task, e.g., generating a graph that is connected and acyclic can be attributed to its structural constraints, while the semantics of a graph can refer to how meaningfully an edge represents the relation between two node concepts. In this work, we study pre-trained language models that generate explanation graphs in an end-to-end manner and analyze their ability to learn the structural constraints and semantics of such graphs. We first show that with limited supervision, pre-trained language models often generate graphs that either violate these constraints or are semantically incoherent. Since curating large amount of human-annotated graphs is expensive and tedious, we propose simple yet effective ways of graph perturbations via node and edge edit operations that lead to structurally and semantically positive and negative graphs. Next, we leverage these graphs in different contrastive learning models with Max-Margin and InfoNCE losses. Our methods lead to significant improvements in both structural and semantic accuracy of explanation graphs and also generalize to other similar graph generation tasks. Lastly, we show that human errors are the best negatives for contrastive learning and also that automatically generating more such human-like negative graphs can lead to further improvements.[1]

1 Introduction

Pre-trained sequence-to-sequence language models (PLMs) like BART (Lewis et al., 2020) and

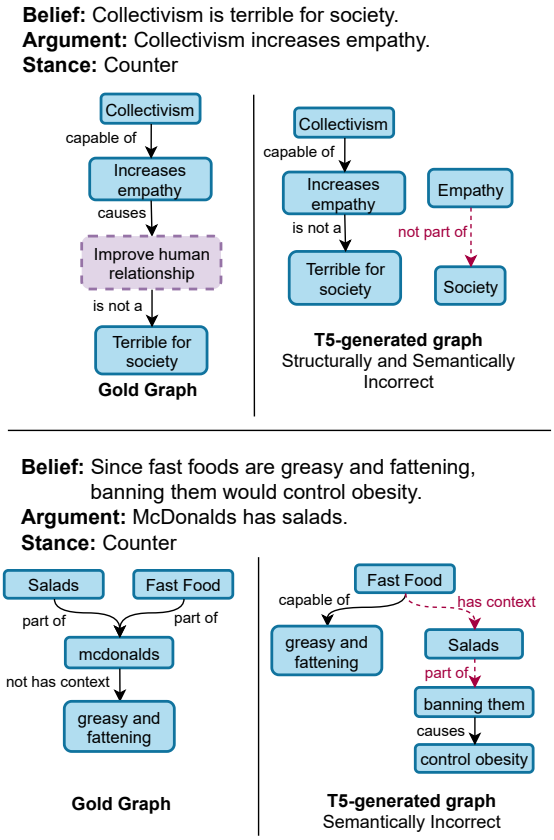

Figure 1: Two representative examples from ExplaGraphs (Saha et al., 2021b) showing the belief, argument, stance, gold explanation graph, and T5-generated explanation graph. The dashed nodes represent commonsense nodes and the dashed edges are incorrect edges. The first generated graph is structurally incorrect and the second graph is semantically incorrect.

T5 (Raffel et al., 2020) have led to significant advances in many natural language generation tasks like text summarization and machine translation. The models are pre-trained on massive amounts of text data with self-supervision, thus enabling them to construct coherent natural language sentences for downstream tasks. This then raises the question whether pre-trained language models, trained on free-form natural language data, can also adapt themselves to generate structured outputs like graphs. Graphs are common in NLP tasks

[1]Our code and models are publicly available at https://github.com/swarnaHub/ExplagraphGen.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 1190 - 1208

that involve representing structured knowledge in the form of knowledge bases (Guarino and Giaretta, 1995), constructing event chains from documents (Chambers and Jurafsky, 2009), or more recent work on encoding reasoning chains, explanations, or deductive proofs (Saha et al., 2020; Tafjord et al., 2021; Dalvi et al., 2021).

Graphs differ from free-form natural language. In the context of NLP, natural language graphs (consisting of textual nodes and edges) can have distinct structural and semantic properties. For example, consider a recently proposed commonsense explanation graph generation task shown in Fig. 1 (Saha et al., 2021b). Each example shows a belief, an argument and an explanation graph explaining how the argument supports or refutes the belief. These explanation graphs encode structured knowledge (augmented with commonsense) and consist of concepts as nodes and relations from ConceptNet (Liu and Singh, 2004) as edges. For example, the second graph encodes the knowledge that "both salads and fast food are part of mcdonalds and hence mcdonalds is not greasy and fattening", thus explicitly refuting the belief. From prior work, the structural constraints enforce the graphs to be connected directed acyclic and the nodes to contain at least two concepts from the belief and two from the argument. The semantic aspect deals with commonsense and evaluates whether each edge expresses coherent relational knowledge and if the whole graph explains the stance.

Following Saha et al. (2021b), we represent graphs as strings composed of concatenated edges and fine-tune T5 to generate graphs in an autoregressive manner. We observe that while moderate amount of supervision enables the model to learn valid graph encodings, the graphs frequently violate task-specific structural constraints (like connectivity). For instance, the first example in Fig. 1 shows a graph generated by T5 that is disconnected and hence structurally incorrect. Moreover, for the fraction of graphs that are structurally correct, the model also makes commonsense mistakes, a type of semantic error, by inferring wrong or incoherent relations between concepts. Both T5-generated graphs shown in Fig. 1 contain incoherent or non-commonsensical edges (marked by dashed arrows) like "fast food; has context; salads". Based on these observations, we study PLMs that generate explanation graphs in an end-to-end manner and analyze their ability to learn the structural constraints as well as the semantics of such graphs.

While a general recipe towards improving the structural and semantic aspects of graph generation can be via large-scale training with more human-annotated graphs, it is prohibitive under most practical scenarios because of the cognitive load associated with a complex data creation task like graph annotation (Dalvi et al., 2021; Saha et al., 2021b). Hence, we propose simple yet effective methods of graph perturbations that perform various kinds of node and edge addition, deletion, and replacement operations to construct structurally and semantically positive (correct) and negative (incorrect) graphs. Overall, we leverage three types of negative graphs (synthetic structural, synthetic semantic, and human-created semantic) and develop multiple contrastive learning models (Hjelm et al., 2018; Chen et al., 2020a; Khosla et al., 2020; Gunel et al., 2020) for effectively distinguishing between correct and incorrect graphs. Our first method is a *Generate-and-Refine* model that first generates an initial graph and further refines it using another T5 model. Next, we propose two improved models – one that uses the negative graphs in a max-margin formulation and another that uses both positive and negative graphs with a InfoNCE (van den Oord et al., 2018) contrastive loss. On two real-world tasks of explanation graph generation and temporal graph generation, with varied node and edge semantics, we observe that our proposed methods and graph perturbation techniques generalize well and lead to improvements in both structural and semantic accuracy of graphs. Further analysis of different types of negative graphs reveal that the human-error graphs are the hardest, most diverse, and hence the best type of negatives to learn from in contrastive learning. Hence, we also develop methods to automatically generate more such human-like semantic negative graphs, which leads to further improvements. We summarize our contributions as follows.

- We present a detailed analysis of graph structure and semantics for end-to-end explanation graph generation via pre-trained language models.
- We propose simple yet effective graph perturbation techniques for constructing positive and negative graphs and use them in different graph contrastive learning models.
- Our methods lead to significant improvements in both structural and semantic accuracy of explanation graphs and also generalize to other similar graph generation tasks.

2 Related Work

Graph Generation from Language Models. Representative works on graph generation from language models include knowledge graph completion models like Comet (Bosselut et al., 2019; Hwang et al., 2021) that fine-tune GPT (Radford et al., 2019; Brown et al., 2020) and BART (Lewis et al., 2020), generation of event influence graphs (Tandon et al., 2019; Madaan et al., 2020), partially ordered scripts (Sakaguchi et al., 2021), temporal graphs (Madaan and Yang, 2021), entailment trees (Dalvi et al., 2021), proof graphs (Saha et al., 2020; Tafjord et al., 2021; Saha et al., 2021a) and commonsense explanation graphs (Saha et al., 2021b). Linguistic tasks like syntactic parsing (Zhou et al., 2020; Mohammadshahi and Henderson, 2021; Kondratyuk and Straka, 2019) and semantic parsing (Chen et al., 2020b; Shin et al., 2021) have also made use of language models. There is also a large body of work on building generative models for learning unconditional graph distributions (You et al., 2018; Simonovsky and Komodakis, 2018; Grover et al., 2019; Liao et al., 2019; Shi* et al., 2020) without any semantics attached to the graphs. Our novelty lies in presenting the first systematic analysis of structure and semantics of graph generation for two downstream NLP tasks using pre-trained language models and improving them via constrastive learning.

Data Augmentation and Contrastive Learning. Data Augmentation for NLP (Hedderich et al., 2020; Feng et al., 2021; Chen et al., 2021) has been a powerful tool in low-data settings, ranging from its early usages with synonym replacement (Kolomiyets et al., 2011; Wang and Yang, 2015) to more recent methods of perturbing hidden representations (Miyato et al., 2016; Shen et al., 2020). Contrastive learning, beyond its historical use in learning robust image representations (Chopra et al., 2005; Hadsell et al., 2006; Gutmann and Hyvärinen, 2010; Hoffer and Ailon, 2015; Hjelm et al., 2018; Chen et al., 2020a; He et al., 2020) has been explored in supervised scenarios (Khosla et al., 2020; Gunel et al., 2020) and for NLP, in training self-supervised language models (Fang et al., 2020), learning sentence representations (Gao et al., 2021), document clustering (Zhang et al., 2021), summarization (Liu and Liu, 2021; Cao and Wang, 2021) and generic text generation (Lee et al., 2020). It has also been used in unconditional graph representation learning (You et al., 2020; Hassani and Khasahmadi, 2020; Zhu et al., 2021). We follow this rich line of work to explore their applicability in supervised graph generation tasks from pre-trained language models in low-resource settings.

Generative Commonsense Reasoning. While traditional commonsense reasoning tasks are discriminative in nature (Zellers et al., 2018; Talmor et al., 2019; Sap et al., 2019; Bisk et al., 2020; Sakaguchi et al., 2020; Talmor et al., 2021), recent focus on generative evaluation have led to the development of tasks and benchmarks that explore unstructured commonsense sentence generation (Lin et al., 2020), event influence graph generation (Madaan et al., 2020), commonsense explanation graph generation (Saha et al., 2021b), etc. We experiment with two graph generation tasks, primarily focusing on ExplaGraphs (Saha et al., 2021b) because of the clear distinction in the underlying structural constraints and the semantic aspect dealing with commonsense.

3 Motivation and Background

Our primary task of interest is a recently proposed commonsense explanation graph generation task called ExplaGraphs (Saha et al., 2021b). In Sec. 6.4, we also experiment with another related task of temporal graph generation (Madaan et al., 2020). In both these tasks, the structural aspect deals with satisfying certain task-specific constraints on the graph (like connectivity) and the semantic aspect deals with the construction of meaningful edges (that adhere to commonsense). Below we discuss ExplaGraphs briefly and analyze pre-trained language models for their ability to generate explanation graphs.

ExplaGraphs (Saha et al., 2021b). In this task, given a belief and an argument, an agent has to perform two sub-tasks – predict the stance (support/counter) and also generate an explanation graph explaining the stance. Explanation graphs are structured explanations that capture explicit reasoning chains between the belief and the argument, thereby making models more interpretable. Formally, an explanation graph is a connected DAG with nodes as concepts and edges as commonsense relations between two concepts (See Fig. 1). The concepts are either part of the belief or the argument (represented with solid boxes) or any external commonsense phrase (represented with dashed boxes). Each edge in the graph forms a coherent sentence and the graph, when read as a whole,

forms reasoning structures explaining why the argument supports or refutes the belief. Saha et al. (2021b) evaluate explanation graphs by defining two accuracy metrics – (1) *Structural Correctness Accuracy (StCA)*: Fraction of graphs that satisfy all structural constraints, and (2) *Semantic Correctness Accuracy (SeCA)*: Fraction of graphs that are both structurally and semantically correct. A graph is considered structurally correct if it satisfies the following constraints: (1) it is connected, (2) it is a DAG, (3) the edge relations belong to a pre-defined list, (4) there are at least two concepts from the belief and two from the argument. If all these constraints are satisfied, the graph is next evaluated for semantic correctness by a model-based metric (Saha et al., 2021b). It works on the principle that an explanation graph is semantically correct if the stance inferred from the belief and the graph matches the gold stance. Refer to Appendix A for a detailed description of all evaluation metrics.

Baseline T5 Model. Following prior work (Saha et al., 2021b), we generate explanation graphs as post-hoc explanations by conditioning on the belief, argument and the predicted stance.[2] The stance prediction model is a fine-tuned RoBERTa model (Liu et al., 2019) which we keep unaltered from prior work and focus on the graph generation sub-task. We generate graphs as linearized strings in an end-to-end manner by leveraging an encoder-decoder pre-trained language model, T5 (Raffel et al., 2020). The input to the model is the concatenated belief, argument and the stance along with a prefix *"Generate an Explanation Graph for"*. The graphs are encoded as concatenated bracketed edges, in which the edges are ordered according to the Depth First Search (DFS) order of the nodes. While we choose T5 because of its superior performance (Saha et al., 2021b), we do not make any model-specific assumptions and graphs can be generated via any encoder-decoder style pre-trained language model (e.g., see Appendix E for results with BART).

Analysis of T5 Baseline. We analyze the quality of the explanation graphs generated by T5 in Table 1. We vary the amount of training data from 500 to 2368 samples (all) and report StCA and SeCA along with other metrics like Graph-BertScore (G-BS) introduced in prior work (Saha et al., 2021b).

[2]These are rationalizing models (Rajani et al., 2019; Hase et al., 2020) that first predict the stance, followed by the graph. While graphs can also be generated first, followed by the stance, we experiment with one model family for this work.

Count	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
500	42.5	20.7	36.3	0.68	20.4
1000	49.2	23.7	42.2	0.63	26.2
1500	50.7	33.2	43.4	0.61	28.2
2368	51.0	34.7	43.9	0.61	29.5

Table 1: Performance of T5-large with varying amount of training data on ExplaGraphs test set.

While the structural accuracy improves with increase in training data, the gain saturates quickly and even after training on the entire data, we find a significant fraction of graphs to violate the structural constraints. We note that a high 91% of T5's generations are valid graph encodings i.e., the generated strings can be parsed into graphical structures (without any post-processing), suggesting that T5 is able to learn the graph encoding from a fairly small amount of supervision. However, it fails to satisfy the various structural constraints – (1) 20% of the graphs are disconnected, (2) 6% of the graphs contain cycles, and (3) 14% of the graphs have less than two concepts from the belief or from the argument. Note that these constraints are not encoded in the model, thus making them fairly hard to learn from limited supervision. On the fraction of structurally correct graphs, the model makes further semantic errors and a lower SeCA of 35% demonstrates that. In Fig. 1, we show examples of structurally incorrect and semantically incorrect graphs generated by T5. Overall, these results indicate that there is a significant scope for improvement both on graph structure and semantics, thus motivating us to develop methods with design choices aimed at improving both aspects.

4 Graph Perturbations

Most prior works that collect human-annotated graphs for a downstream NLP task have found such collection processes to be quite expensive and tedious (Tandon et al., 2019; Dalvi et al., 2021; Saha et al., 2021b). For instance, Saha et al. (2021b) obtained high-quality data only after multiple rounds of refinement and Dalvi et al. (2021) employ trained expert annotators for entailment tree construction. The corresponding datasets are also relatively small in size (2-3k), thus limiting the prospect of large-scale training. Hence, our approach towards improving explanation graph generation is through data augmentation techniques that perturb human-curated graphs to construct positive and negative graphs. As noted earlier, we wish to construct graphs that enable better learning of

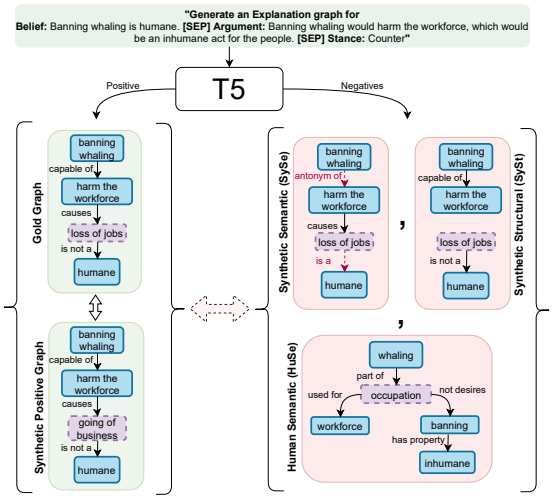

Figure 2: Our T5-based contrastive learning framework for graph generation using positively and three kinds of negatively perturbed graphs.

structural graph constraints and their semantics.

4.1 Positive Graph Perturbations

One simple method to augment existing training data is to create synthetic positive graphs. These graphs should be created such that all the task-specific constraints continue to hold upon perturbations. E.g., removing a node that makes the graph disconnected is a prohibitive action. Hence, we choose nodes (concepts) that are not part of the belief or the argument (also termed as commonsense nodes) and replace them with phrases that are synonymous to the original phrases. To do so, we select words from the concept with POS tags of Adjective, Noun, Adverb, or Verb and replace them with that synonym from Wordnet (Miller, 1995) for which the cosine similarity of their word2vec representations (Mikolov et al., 2013) is the highest.[3] Fig. 2 shows an example of a positive graph perturbation where the node "loss of jobs" is replaced with "going of business". Note that our node replacement operations will always lead to structurally similar graphs. Automatically constructing structurally diverse positive graphs is a challenging problem and we leave that for future work.

4.2 Negative Graph Perturbations

In order to enable the model to learn from explicit hard negatives, we construct three diverse types of graphs – synthetically constructed structural negatives for learning graph constraints and synthetic and human-created semantic negatives to capture a fairly large space of semantically incorrect graphs. Below we discuss the construction of these graphs.

Synthetic & Structurally Negative Graphs (SySt). As shown previously, one common source of errors in the generated explanation graphs is the violation of structural constraints. To enable learning these constraints, we generate four types of negative graphs by performing the following perturbations on each ground-truth graph: (1) removing an edge at random such that the resultant graph becomes disconnected, (2) adding an edge between two randomly chosen nodes such that the resultant graph becomes cyclic, (3) adding and removing one edge at random such that the resultant graph becomes both disconnected and cyclic, (4) removing a node randomly such that the resultant graph contains less than two concepts from the belief or argument. Fig. 2 shows an example of a disconnected graph created as part of the structurally negative graphs.

Synthetic & Semantic Negative Graphs (SySe). We also construct semantically incorrect negative explanation graphs. While the previous category of negative graphs (SySt) captures structural constraints, SySe captures the relational knowledge in graphs. Semantic incorrectness typically arises from inappropriate relations that do not adhere to human commonsense ("loss of jobs; is a; humane"). We create such negative graphs by selecting a random number of edges and then replacing the relations with some other relations. Fig. 2 shows a semantic negative graph in which the relations marked with dashed lines are perturbed.

Human-created & Semantic Negative Graphs (HuSe). The space of semantically incorrect graphs is fairly large and in order to augment our synthetic negative graphs with harder structurally-diverse negatives, we make use of human-created incorrect graphs from prior work (Saha et al., 2021b).[4] Humans make subtle errors, thus making them ideal negative candidates for contrastive learning. ExplaGraphs was constructed via an iterative framework in which the graphs are iteratively refined (up to two times) until they are verified as correct. We treat these refined graphs as negatives. Specifically, in two rounds, if an initial graph $\mathcal{G}_1$

[3] We also tried similar replacement operations with antonyms. However, they often lead to semantically inconsistent graphs. E.g., *A causes B* does not always imply *A not causes not B* or *not A not causes not B*.

[4] Publicly released by Saha et al. (2021b) at https://github.com/swarnaHub/ExplaGraphs/blob/main/data/refinement_graphs_train.tsv.

is refined into graphs $\mathcal{G}_2$ and $\mathcal{G}_3$ successively, then $\mathcal{G}_1$ and $\mathcal{G}_2$ are considered as negative graphs. Unlike SySe which only perturb the relations, these negatives are structurally diverse (see Fig. 2) and capture semantics not just at the level of each edge but for the graph as a whole (e.g., a graph might be refined because it does not explain the stance). Note that human-created graphs can only be semantically incorrect, since their structural correctness is already ensured during construction.

5 Augmentation with Perturbed Graphs

Next we propose different methods of leveraging these positive and negative graphs for explanation graph generation. Our models either use only positive graphs as simple data augmentation, only negative graphs in a max-margin model, or both in a *Generate & Refine* model and a Contrastive model.

5.1 Augmentation with Positive Graphs

In this first simple approach, we augment the training data with the synthetically created positive graphs and retrain the baseline T5 model.

5.2 Max-Margin Graph Generation Model

Our next model leverages the negatively perturbed graphs in a max-margin formulation. During training, given a (belief, argument, stance) context x, a ground truth graph $\mathcal{G}^{(g)}$ and a negative graph $\mathcal{G}^{(n)}$, linearized into a sequence of words $\{y_i^{(g)}\}_{i=1}^{k}$ and $\{y_i^{(n)}\}_{i=1}^{l}$ respectively, we define the loss function $\mathcal{L}$ as a linear combination of the standard cross-entropy loss $\mathcal{L}_{CE}$ and a max-margin loss $\mathcal{L}_{MM}$, defined between a word $y_i^{(g)}$ of the positive graph and a word $y_i^{(n)}$ of the negative graph.

$$\mathcal{L}_{CE} = \sum_i -logP_\theta(y_i^{(g)}|y_{<i}^{(g)}, x)$$

$$\mathcal{L}_{MM} = \sum_i \max(0, logP_\theta(y_i^{(g)}|y_{<i}^{(g)}, x) - \log P_\theta(y_i^{(n)}|y_{<i}^{(n)}, x) + \beta)$$

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha\mathcal{L}_{MM}$$

where α and β (margin) are hyperparameters. As noted earlier, the baseline model often makes commonsense mistakes in distinguishing between positive and negative relations ("causes" vs "not causes") and our relation perturbing negative graphs and the max-margin loss component facilitate learning a better boundary between them.

5.3 Generate & Refine Graph Generation

ExplaGraphs was constructed using a "Refinement" phase wherein the initially constructed graphs that are marked incorrect by human verifiers are further refined by another set of annotators. Here we emulate the graph refinement phase with the help of a model. Specifically, our approach is a 2-stage pipeline – first, an initial graph is generated by the baseline T5 model and second, an *Explanation Graph Refinement* model conditions on the initial graph, along with the belief, argument and the stance to refine the graph. The refiner is also a T5 model fine-tuned with the prefix "Refine the Explanation Graph for" on all positive and negative graphs described in Sec. 4. Note that our approach differs from the actual data collection process in two aspects. Unlike the human-annotated graphs, which are refined only for semantic correctness, the model-generated graphs can be both structurally and semantically incorrect. Second, our approach does not involve a graph verification stage and thus, the refiner model acts on all (correct and incorrect) graphs generated in stage 1 and is thus trained with both correct and incorrect graphs.

5.4 Contrastive Graph Generation Model

Our Contrastive Graph Generation Model (Fig. 2) also leverages both positive and negative graphs but instead of doing so in a 2-stage *Generate & Refine* model, uses a contrastive learning framework (Khosla et al., 2020; Gunel et al., 2020). Given a ground-truth graph $\mathcal{G}^{(g)}$, a positive graph $\mathcal{G}^{(p)}$ and a set of negative graphs $\{\mathcal{G}_i^{(n)}\}_{i=1}^{M}$, contrastive learning aims to learn the graph representations such that the gold graph's representation is close to that of the synthetic positive graph while being distant from those of the negative graphs. Similar to Cao and Wang (2021), we use the last layer of the decoder in T5 as the representation of each token in the graph and obtain the graph representation by averaging over the constituent token representations. Let the graph representations be denoted by $h^{(g)}$, $h^{(p)}$ and $\{h_i^{(n)}\}_{i=1}^{M}$. Given $\mathcal{H}^{(g)} = \{h^{(p)}\} \bigcup \{h_i^{(n)}\}_{i=1}^{M}$, our overall loss combines the cross-entropy loss $\mathcal{L}_{CE}$ and the InfoNCE contrastive loss (van den Oord et al., 2018) $\mathcal{L}_{CL}$ as shown below.

$$\mathcal{L}_{CL} = -\log \frac{\exp(\text{sim}(h^{(g)}, h^{(p)})/\tau)}{\sum_{h_i \in \mathcal{H}^{(g)}} \exp(\text{sim}(h^{(g)}, h_i)/\tau)}$$

$$\mathcal{L} = \mathcal{L}_{CE} + \alpha\mathcal{L}_{CL}$$

	SA↑	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
T5-Base (Saha et al., 2021b)	**87.2**	38.7	19.0	33.6	0.71	20.8
T5-Large	**87.2**	51.0	34.7	43.9	0.61	29.5
Generate & Refine	**87.2**	52.5	37.7	45.3	0.60	30.0
Pos Data Aug	**87.2**	54.5	41.5	46.9	0.58	30.2
Max-Margin	**87.2**	56.7	**43.5**	48.6	0.57	30.5
Contrastive	**87.2**	**60.5**	42.5	**52.1**	**0.52**	**33.1**
Upper Bound	91.0	91.0	83.5	71.1	0.38	46.8

Table 2: Comparison of all models across all metrics on the ExplaGraphs (Saha et al., 2021b) test set. Improvement in SeCA is statistically significant (computed using Bootstrap test (Efron and Tibshirani, 1994)) with $p < 0.005$.

where α and the temperature τ are the hyperparameters and $\mathrm{sim}()$ denotes the cosine similarity function between the graph representations.

6 Experiments

6.1 Impact of Different Models on Graph Structural and Semantic Accuracy

In Table 2, we compare the various modeling techniques described in Sec. 5 and their effect on the structural and semantic correctness of the generated graphs. While our primary metrics of interest are Graph Structural Accuracy (StCA) and Semantic Accuracy (SeCA), following prior work (Saha et al., 2021b), we also report Stance Accuracy (SA), Graph-BertScore (G-BS), Graph Edit Distance (GED) and Edge Accuracy (EA).

Effect of Model Size and Training Data. The T5-Large model uses the same setup as the T5-Base model experimented with in Saha et al. (2021b). We observe that using a larger T5 model improves StCA by 12% and SeCA by 16%. This finding is in line with other commonsense reasoning tasks (Lourie et al., 2021; Elazar et al., 2021) which also show that fine-tuning a larger language model typically leads to better performance. Together with the results reported in Table 1, we conclude that much of the improvement in explanation graph generation comes from increasing the training data and using a larger model. Given its superior performance, we build our proposed models on T5-large.

Results with Generate & Refine Model. The *Generate & Refine* model (Sec. 5.3) improves all metrics; however the gains are small. Note that this model refines all graphs (correct or not) and can lead to already correct graphs becoming incorrect after refinement. In practice, we observe that most graphs do not change much after refinement which we believe stems from the model's inability to distinguish between correct and incorrect graphs.

Effect of Positive Graph Perturbations. On retraining T5 augmented with the positively perturbed graphs (Sec. 5.1), we observe that it obtains significant improvement over T5 and *Generate & Refine* both in structural and semantic accuracy. Note that, by construction, the positive graphs only differ in the commonsense concepts (not part of the belief or argument) while keeping the structure intact. Hence, the model has more supervision about the semantics of the graphs as opposed to the structural constraints. This is reflected in the larger improvement in SeCA. The positive graphs, being structurally correct, also reinforces the model's belief about structural correlation with correct graphs, thus leading to some improvement in StCA as well.

Effect of Negative Graph Perturbations. The *Max-Margin* model (Sec. 5.2) leverages all structurally and semantically incorrect graphs and obtains up to 6% and 9% improvement in StCA and SeCA respectively over the baseline T5 model. The model implicitly learns the structural constraints through relevant supervision and the margin-based loss enables it to learn a better boundary between correct and incorrect graphs. Similarly, the semantically perturbed graphs improves the model's relation prediction capability between concepts. The *Max-Margin* model outperforms the *Pos Data Aug* model because of the former having access to both structural and semantic supervision while the latter is only augmented with structurally similar graphs.

Effect of Positive and Negative Graph Perturbations with Contrastive Learning. The *Contrastive Graph Generation* model (Sec. 5.4) leverages both positive and negative graphs and improves StCA to 60% with comparable SeCA to the *Max-Margin* model. The overall improvements in StCA and SeCA are 9% and 8% respectively compared to T5. We hypothesize that the constrastive model does not lead to further improvement in SeCA because of the structurally similar positive

	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
T5-Large	46.5	31.6	36.8	0.66	26.7
+ SySt	50.2	34.1	40.7	0.64	27.4
+ SySe	50.7	35.1	40.8	0.63	27.3
+ HuSe	49.5	38.4	39.4	0.64	26.1

Table 3: Ablation study showing the effect of different types of negative graphs on ExplaGraphs dev set.

	Valid↑	StCA↑	G-BS↑
T5-Base	88.8	88.7	54.4
Max-Margin	89.1	87.7	55.7
Contrastive	**97.5**	**96.9**	**57.2**

Table 4: Comparison of T5, Max-Margin and Contrastive models for temporal graph generation.

graphs. This can potentially be improved by incorporating more structurally diverse graphs. Finally, our best SeCA is far from perfect and significant future work can be done in improving the graph semantics. Further ablations of negative graphs and human evaluation are done on the *Max-Margin* model, due to its slightly higher SeCA.

6.2 Human Evaluation of Graph Semantics

Automatically evaluating graphs for semantic correctness is challenging. We conduct human evaluation to further validate our findings. We compare the graphs generated by T5 and our *Max-Margin* model on Amazon Mechanical Turk where three annotators choose which graph is better or if they are mostly similar (instructions in Appendix F). For fair comparison, we evaluate only those samples where both models predict the correct stance and the graphs are also structurally correct. In fact, this lets us evaluate the semantic aspect in isolation when both graphs are structurally correct. With majority voting on 150 samples, we observe that our *Max-Margin* model's graphs are preferred 13% more times compared to those of the T5 model (43% vs 30% and statistically significant with $p < 0.05$) while in 22% cases, the graphs are marked similar (remaining have no majority).

6.3 Ablation with Negative Graphs

In Table 3, we show the effect of different types of negative graphs. We compare the results on the ExplaGraphs validation set by leveraging Synthetic Structural (SySt), Synthetic Semantic (SySe) and Human-created Semantic (HuSe) graphs with the *Max-Margin* graph generation model. All types of negatives graphs lead to consistent increase in SeCA. Leveraging human-created negative graphs leads to a bigger gain in SeCA because of the hardness and diversity in these graphs and hence are the best candidates for contrastive learning.

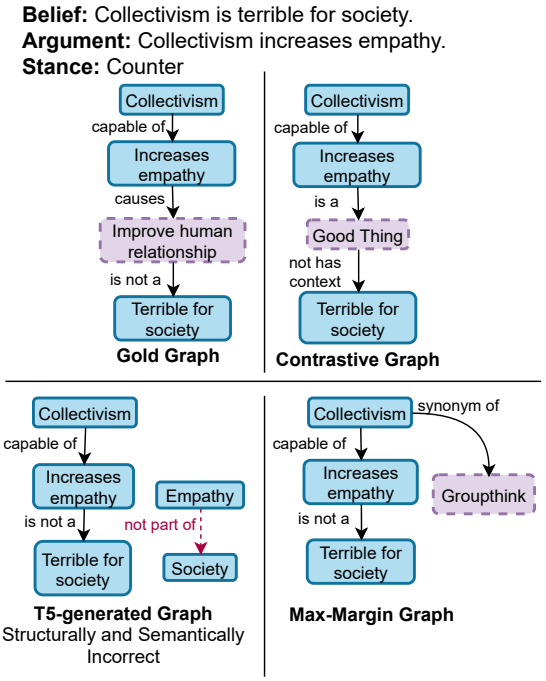

Figure 3: Qualitative analysis of explanation graphs.

6.4 Generalization to Other Graph Generation Tasks

We test the generalizability of constructing structurally and semantically perturbed graphs for contrastive learning by also experimenting on a temporal graph generation task (Madaan and Yang, 2021) that requires constructing a temporal graph from a document. The nodes in the graph are events from the document and the edges are temporal relations between events ("before", "after", etc). Following our overall goal of improving graph generation with limited data, we randomly sample 1.3% of the overall corpus (∼9.5k samples) as the training data such that all graphs are connected DAGs. Similar to ExplaGraphs, we create structurally negative graphs with disconnected and cyclic graphs and semantic negative graphs by perturbating the temporal relations. E.g., if an edge relation is "before", we replace it with "after". We construct positive graphs by replacing edges like "A before B" with "B after A" (more details in Appendix C). In Table 4, we report structural correctness accuracy (StCA) (percentage of connected DAGs) and Graph-BertScore (G-BS) for measuring approximate semantic correctness wrt gold graphs. We observe that our contrastive model not only generates more valid graph encodings but also improves StCA by 8% and G-BS by 3%.

	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
SySt + SySe + HuSe	49.5	38.4	39.4	0.64	26.1
SySt + SySe + HuSe + *HuSe-Gen (IP)*	53.5	38.7	42.1	0.62	28.1
SySt + SySe + HuSe + *HuSe-Gen (AE)*	52.0	40.2	41.3	0.62	28.2

Table 5: Effect of training the Max-Margin model with additional Human-like Semantic Negative Graphs on ExplaGraphs dev set. IP and AE refer to the two thresholding techniques for filtering generated negatives.

6.5 Analysis of Generated Graphs

Fig. 3 shows an example of the graphs generated by different models (more examples in Appendix F). Unlike T5, our models' graphs are both structurally and semantically correct with diverse commonsense nodes ("Groupthink", "Good Thing"). While our models generate more correct graphs, they lack in structural diversity – the Contrastive model generates 77% of linear graphs (i.e., the nodes are in a linear chain) which is comparable to 75% in the T5 model. This can be attributed to our structurally similar positive graphs as the model does not obtain enough supervision to generate diverse graphs. Structural diversity is not a measure of graph correctness; however, like diverse text generation (Vijayakumar et al., 2018), generating diverse graphs is an interesting direction for future work.

6.6 Generating Human-like Semantic Negatives (HuSe-Gen)

In ExplaGraphs, human-created negatives account for 38% of the samples for which the initially constructed graph was incorrect and was refined. Moreover, we see in the previous section that human-error graphs are the best negative candidates for contrastive learning (which is intuitive since tricky and subtle errors made by expert human annotators would make for some of the hardest negatives/distractors for a contrastive learning model to learn from). Hence, in this final section, we further explore whether it is also possible to automatically imitate and generate more of such *harder human-like* incorrect graphs for the remaining samples as well. Our method consists of the following steps.

Human-like Negative Edge Generation. We first fine-tune a T5 model that conditions on the belief, argument and the stance to generate a set of incorrect edges (which is the set of edges that are present in the incorrect graph and not in the refined graph).

Human-like Negative Graph Construction. This generated set of incorrect edges is then added to the correct graph to construct the incorrect graph, such that it is structurally correct and hence representative of human-like erroneous graphs.

Filtering High-quality Negative Graphs. Contrastive models will only benefit from these negatives if the negative edge generation model is accurate and generates edges that are actually incorrect. Hence, we control the quality of the generated incorrect graphs by the following two techniques – (a) *Thresholding via fraction of Acceptable Edges (AE):* We say that a generated incorrect edge is acceptable if it is not part of the correct graph and can be added to the correct graph without violating any structural constraints. We compute the fraction of acceptable edges for every generated negative graph and choose only those graphs with AE above a certain threshold δ. Intuitively, this ensures that a high fraction of the generated edges are actually incorrect and hence when added to the correct graph, will lead to a sufficiently different (human-like) incorrect graph. (b) *Thresholding via Incorrect Probability of a graph (IP):* We use our SeCA metric model (that classifies a graph into support, counter, or incorrect class) to compute the probability of the generated graph being incorrect and choose those graphs that are above a certain threshold γ of incorrect probability.

We set $\delta = 0.4$ and $\gamma = 0.5$ (tuned on the dev set) and train the Max-margin model using these additionally generated human-like negative graphs. As shown in Table 5 both thresholding approaches lead to further improvements over using just the human-created negative graphs. These initial promising results for emulating hard/tricky human errors as strong negatives for contrastive learning will hopefully lead to further future work in this interesting direction.

7 Conclusion

We presented an empirical study of graph structure and semantics for end-to-end explanation graph generation from pre-trained language models and showed that the generated graphs often violate structural constraints or are semantically incorrect. We significantly improve both the structural and semantic accuracy of graph generation by proposing contrastive learning models that leverage simple yet efficient methods of graph perturbations and also generalize to similar graph generation tasks.

Ethical Considerations

From an ethics standpoint, we provide a brief overview and show samples from the datasets that our models are trained on throughout the paper and also in the Appendix. Explanation graph generation improves the interpretability of neural commonsense reasoning systems and could prove to be effective in understanding and debugging such models. Hence we do not foresee any major risks or negative societal impact of our work. However, like any other ML model, the graphs generated by our models may not always be completely accurate and hence should be used with caution for real-world applications.

Acknowledgements

We thank the reviewers for their helpful feedback and the annotators for their time and effort. This work was supported by DARPA MCS Grant N66001-19-2-4031, NSF-CAREER Award 1846185, DARPA YFA17-D17AP00022, ONR Grant N00014-18-1-2871, Microsoft Investigator Fellowship, and Munroe & Rebecca Cobey Fellowship. The views in this article are those of the authors and not the funding agency.

References

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7432–7439.

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.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Shuyang Cao and Lu Wang. 2021. CLIFF: Contrastive learning for improving faithfulness and factuality in abstractive summarization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6633–6649, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Nathanael Chambers and Dan Jurafsky. 2009. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610.

Jiaao Chen, Derek Tam, Colin Raffel, Mohit Bansal, and Diyi Yang. 2021. An empirical survey of data augmentation for limited data learning in nlp. *arXiv preprint arXiv:2106.07499*.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020b. Low-resource domain adaptation for compositional task-oriented semantic parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE.

Bhavana Dalvi, Peter Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7358–7370.

Bradley Efron and Robert J Tibshirani. 1994. *An introduction to the bootstrap*. CRC press.

Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. 2021. Back to square one: Artifact detection, training and commonsense disentanglement in the winograd schema. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10486–10500.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Aditya Grover, Aaron Zweig, and Stefano Ermon. 2019. Graphite: Iterative generative modeling of graphs. In *International conference on machine learning*, pages 2434–2444. PMLR.

Nicola Guarino and Pierdaniele Giaretta. 1995. Ontologies and knowledge bases. *Towards very large knowledge bases*, pages 1–2.

Beliz Gunel, Jingfei Du, Alexis Conneau, and Veselin Stoyanov. 2020. Supervised contrastive learning for pre-trained language model fine-tuning. In *International Conference on Learning Representations*.

Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings.

Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742. IEEE.

Peter Hase, Shiyue Zhang, Harry Xie, and Mohit Bansal. 2020. Leakage-adjusted simulatability: Can models generate non-trivial explanations of their behavior in natural language? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4351–4367.

Kaveh Hassani and Amir Hosein Khasahmadi. 2020. Contrastive multi-view representation learning on graphs. In *International Conference on Machine Learning*, pages 4116–4126. PMLR.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738.

Michael A Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2020. A survey on recent approaches for natural language processing in low-resource scenarios. *arXiv preprint arXiv:2010.12309*.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. 2018. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*.

Elad Hoffer and Nir Ailon. 2015. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, pages 84–92. Springer.

Jena D Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. (comet-) atomic 2020: On symbolic and neural commonsense knowledge graphs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 7, pages 6384–6392.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2011. Model-portability experiments for textual temporal analysis. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, volume 2, pages 271–276. ACL; East Stroudsburg, PA.

Dan Kondratyuk and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. 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 2779–2795.

Eleftherios Koutsofios and Stephen C North. 1996. Drawing graphs with dot.

Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. 2020. Contrastive learning with adversarial perturbations for conditional text generation. In *International Conference on Learning Representations*.

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.

Renjie Liao, Yujia Li, Yang Song, Shenlong Wang, Charlie Nash, William L. Hamilton, David Duvenaud, Raquel Urtasun, and Richard Zemel. 2019. Efficient graph generation with graph recurrent attention networks. In *NeurIPS*.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1823–1840.

Hugo Liu and Push 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*.

Yixin Liu and Pengfei Liu. 2021. Simcls: A simple framework for contrastive learning of abstractive summarization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 1065–1072.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Unicorn on rainbow: A universal commonsense reasoning model on a new multitask benchmark. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 15, pages 13480–13488.

Aman Madaan, Dheeraj Rajagopal, Yiming Yang, Abhilasha Ravichander, Eduard Hovy, and Shrimai Prabhumoye. 2020. Eigen: Event influence generation using pre-trained language models. *arXiv preprint arXiv:2010.11764*.

Aman Madaan and Yiming Yang. 2021. Neural language modeling for contextualized temporal graph generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 864–881.

Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. 2016. Adversarial training methods for semi-supervised text classification. *arXiv preprint arXiv:1605.07725*.

Alireza Mohammadshahi and James Henderson. 2021. Recursive non-autoregressive graph-to-graph transformer for dependency parsing with iterative refinement. *Transactions of the 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.

Swarnadeep Saha, Sayan Ghosh, Shashank Srivastava, and Mohit Bansal. 2020. PRover: Proof generation for interpretable reasoning over rules. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 122–136.

Swarnadeep Saha, Prateek Yadav, and Mohit Bansal. 2021a. multiPRover: Generating multiple proofs for improved interpretability in rule reasoning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3662–3677.

Swarnadeep Saha, Prateek Yadav, Lisa Bauer, and Mohit Bansal. 2021b. ExplaGraphs: An explanation graph generation task for structured commonsense reasoning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7716–7740.

Keisuke Sakaguchi, Chandra Bhagavatula, Ronan Le Bras, Niket Tandon, Peter Clark, and Yejin Choi. 2021. proScript: Partially ordered scripts generation via pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP*.

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

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

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

Chence Shi*, Minkai Xu*, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. 2020. Graphaf: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*.

Richard Shin, Christopher H Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021. Constrained language models yield few-shot semantic parsers. *arXiv preprint arXiv:2104.08768*.

Martin Simonovsky and Nikos Komodakis. 2018. Graphvae: Towards generation of small graphs using variational autoencoders. In *International conference on artificial neural networks*, pages 412–422. Springer.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2021. ProofWriter: Generating implications, proofs, and abductive statements over natural language. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3621–3634, Online. Association for Computational Linguistics.

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.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandra Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2021. Commonsenseqa 2.0: Exposing the limits of ai through gamification. In *NeurIPS 2021 Datasets and Benchmarks Track*.

Niket Tandon, Bhavana Dalvi, Keisuke Sakaguchi, Peter Clark, and Antoine Bosselut. 2019. Wiqa: A dataset for "what if..." reasoning over procedural text. 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 6076–6085.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748.

Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. 2018. Diverse beam search: Decoding diverse solutions from neural sequence models. In *AAAI*.

William Yang Wang and Diyi Yang. 2015. That's so annoying!!!: A lexical and frame-semantic embedding based data augmentation approach to automatic categorization of annoying behaviors using# petpeeve tweets. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2557–2563.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

Jiaxuan You, Rex Ying, Xiang Ren, William Hamilton, and Jure Leskovec. 2018. Graphrnn: Generating realistic graphs with deep auto-regressive models. In *International conference on machine learning*, pages 5708–5717. PMLR.

Yuning You, Tianlong Chen, Yongduo Sui, Ting Chen, Zhangyang Wang, and Yang Shen. 2020. Graph contrastive learning with augmentations. *Advances in Neural Information Processing Systems*, 33:5812–5823.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 93–104.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shang-Wen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew O Arnold, and Bing Xiang. 2021. Supporting clustering with contrastive learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5419–5430.

Tianyi Zhang*, Varsha Kishore*, Felix Wu*, Kilian Q. Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Junru Zhou, Zhuosheng Zhang, Hai Zhao, and Shuailiang Zhang. 2020. LIMIT-BERT : Linguistics informed multi-task BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4450–4461.

Yanqiao Zhu, Yichen Xu, Feng Yu, Qiang Liu, Shu Wu, and Liang Wang. 2021. Graph contrastive learning with adaptive augmentation. In *Proceedings of the Web Conference 2021*, pages 2069–2080.

A Evaluation Metrics for ExplaGraphs

Below we provide brief descriptions of the evaluation metrics used for the ExplaGraphs task. For further details, we refer readers to prior work (Saha et al., 2021b).

Structural Correctness Accuracy of Graphs (StCA). It computes the fraction of graphs where all the structural constraints are satisfied.

Semantic Correctness Accuracy of Graphs (SeCA). SeCA is a model-based metric that computes the fraction of graphs that are both structurally and semantically correct. For computing SeCA, prior work trains a 3-way RoBERTa (Liu

SySt	SySe	HuSe	Total
7522	2368	1336	11226

Table 6: Count of negative graphs in each category.

et al., 2019) classifier that given a belief and a generated explanation graph, infers whether the graph supports the belief, counters the belief or is incorrect (because of incoherent edges). If it predicts support or counter and this stance matches the gold stance, then the graph is considered semantically correct. In essense, SeCA works on the principle that an explanation graph is semantically correct if a stance can be unambiguously inferred from it (by a model in this case or a human) and that stance is the same as the gold stance. Note that SeCA is a reference-free metric (does not use the ground-truth graph) and hence is invariant to structural variations in explanation graphs.

Graph-BertScore (G-BS). Graph-BertScore is an extension of BertScore (Zhang* et al., 2020) for computing the degree of match between the predicted graphs and the ground-truth graphs. It treats a graph as a set of edges and computes the best match between the gold edges and the predicted edges, where the matching score between a pair of edges is given by the BertScore F1.

Graph Edit Distance (GED). GED is the standard Graph Edit Distance for graphs, measuring the number of edit operations (addition, deletion, and replacement of nodes and edges) to transform one graph to the other and further normalized by an appropriate normalizing constant.

Edge Accuracy (EA). The final metric, Edge Accuracy (EA) measures the fraction of edges in the graph that are important. An edge is considered important if removing it from the graph leads to a drop in the gold stance prediction confidence.

B Statistics of Graph Perturbations

We create a total of 11k negative graphs. Table 6 shows the respective counts of the negative graphs belonging to synthetic structural (SySt), synthetic semantic (SySe) and human-created semantic (HuSe) categories.

C Temporal Graph Generation

The task of temporal graph generation requires constructing a temporal graph from a document (see Fig. 4). The nodes in the graph are events from the document (e.g., "Markovic jailed" or "Covering up attempted murder") and the edges are temporal relations between the events (e.g., "Markovic jailed; before; Covering up attempted murder"). The authors consider five temporal relations ("before", "after", "simultaneous", "is included" and "includes") and build an automatically constructed large-scale dataset for the task. Following our overall goal of improving graph generation in limited data settings, we randomly sample 1.3% of the overall corpus ($\sim$ 9.5k samples) as the training corpus such that all graphs are connected DAGs.[5] Following Madaan and Yang (2021), we represent graphs in DOT format (Koutsofios and North, 1996) as shown in Fig. 4. We find that the specifics of the graph representations do not matter much, as long as all the edges are concatenated in one particular ordering (either DFS, BFS or Topological order).

Dataset	Train	Dev	Test
ExplaGraphs	2368	398	400
Temporal (Sampled)	9531	953	949

Table 7: Train, validation and test split sizes of the two datasets. For Temporal Graph Generation, we randomly sample 1.3% of the overall corpus (Madaan and Yang, 2021).

We construct semantic negative graphs by randomly sampling a fraction of the edges and performing the following operations. If an edge relation is one of "before", "after" or "simulatenous", we replace it with any other relation from this set and if the relation is one of "is included" or "includes" we replace it with the other relation. Note that these perturbations will always lead to incorrect graphs because "A before B" implies that "A after B" or "A simultaneous B" do not hold. Finally, we construct positive graphs by randomly sampling a fraction of edges and replacing them using the following rules: (1) "A before B" with "B after A" and viseversa, (2) "A simultaneous B" with "B simultaneous A", (3) "A includes B" with "B is included A". Note that all these operations preserve the temporal meaning of the graph and are done in a way such that the perturbed graph continues to be a connected DAG.

[5] Since the dataset was constructed automatically, we found about 10% of the graphs to be disconnected or cyclic.

	SA↑	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
T5-Base	**86.2**	35.4	15.5	27.7	0.75	19.8
T5-Large	**86.2**	46.5	31.6	36.8	0.66	26.8
Generate & Refine	**86.2**	46.8	34.4	37.2	0.66	27.2
Pos Data Aug	**86.2**	50.0	37.6	39.6	0.64	28.4
Max-margin	**86.2**	49.5	**38.4**	39.4	0.64	26.1
Contrastive	**86.2**	**52.7**	37.9	**41.7**	**0.62**	**29.8**

Table 8: Comparison of our models with baseline T5 models across all metrics on ExplaGraphs dev set.

	SA↑	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
BART-Base	**87.2**	25.7	13.0	22.0	0.81	12.8
BART-Large	**87.2**	34.2	22.2	28.9	0.75	20.0
Contrastive	**87.2**	**40.7**	**26.3**	**31.3**	**0.71**	**22.3**

Table 9: Effect of Contrastive Learning with BART on ExplaGraphs test set.

D Experimental Setup

Table 7 shows the number of train, validation and test samples of the two datasets we experiment with. We build our models on top of the Hugging Face transformers library (Wolf et al., 2020).[6] All models for the ExplaGraphs dataset[7] (Saha et al., 2021b) are trained with a batch size of 8 and an initial learning rate of $3 * 10^{-5}$ for a maximum of 15 epochs. The maximum input and output sequence lengths are both set to 150. For the max-margin graph generation model, we set both the hyperparameters α (mixing ratio) and β (margin) to 1.0 while for the contrastive graph generation model, we set α to 0.1. For the temporal graph generation task[8] (Madaan and Yang, 2021), we train all models with a batch size of 4 and an initial learning rate of $3 * 10^{-5}$ for a maximum of 10 epochs. The maximum input and output sequence lengths are set to 512 and 256 respectively. On this task, the hyperparameters α and β for the max-margin model are again set to 1.0 while for the contrastive graph generation model, we set α to 0.2.

Across all models and tasks, graphs are generated using beam search decoding with a beam size of 4. The batch size and learning rate are manually tuned in the range $\{4, 8, 16\}$ and $\{10^{-5}$, $2 * 10^{-5}$, $3 * 10^{-5}\}$ respectively and the best models are chosen based on the respective validation set performance. Similarly, the mixing ratio hyperparameter α is manually tuned in the range $\{0.1, 0.2, 0.5, 1.0\}$. The random seed is set to 42 in all our experiments. The total number of parameters in our models is similar to T5-Base (220M) or T5-Large (770M) depending on the base architecture. All our experiments are executed on a single A100 Nvidia GPU. Each epoch of the contrastive model has an average runtime of 30 mins for ExplaGraphs and 2.5 hours for Temporal Graph Generation.

[6] https://github.com/huggingface/transformers

[7] https://github.com/swarnaHub/ExplaGraphs

[8] https://github.com/madaan/temporal-graph-gen

	StCA↑	SeCA↑	G-BS↑	GED↓	EA↑
Max-Margin	56.7	43.5	48.6	0.57	30.5
+ Atomic	**58.2**	**45.0**	**49.9**	**0.56**	**30.9**

Table 10: Effect of fine-tuning with additional commonsense knowledge from Atomic.

E Results

Table 8 shows the results of all models on the ExplaGraphs (Saha et al., 2021b) validation set.

Experiments with BART. In Table 9, we show the performance of BART (Lewis et al., 2020) on ExplaGraphs (Saha et al., 2021b) test set. Unsurprisingly, a larger BART model obtains a much higher StCA and SeCA compared to BART-Base. However, we find T5 to perform much better on this task. Applying contrastive learning on top of BART leads to improvements across all metrics, thereby showing our method's generalizability across different pre-trained language models.

Effect of Additional Commonsense Knowledge. In Table 10, we explore the impact of integrating additional commonsense knowledge to our *Max-Margin* model. Specifically, we first fine-tune a

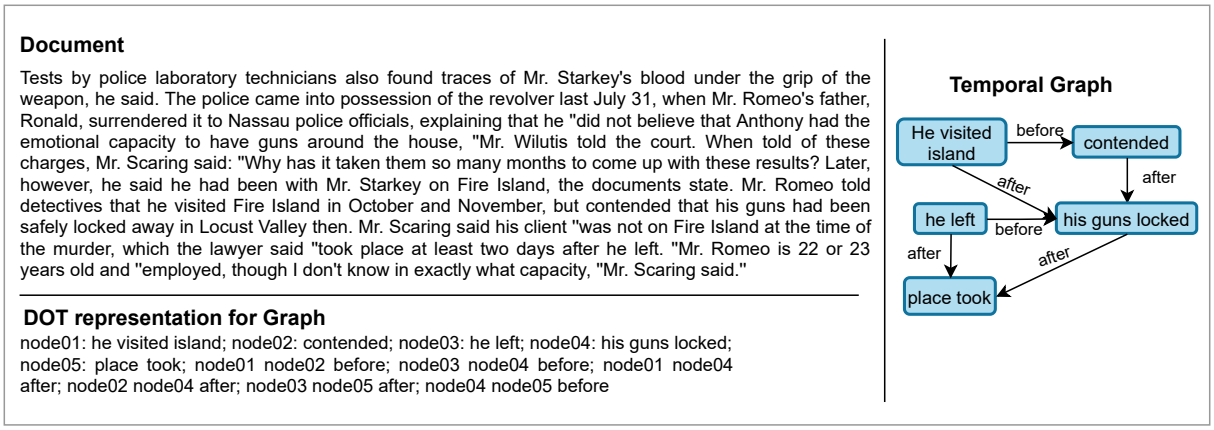

Figure 4: An example of the Temporal Graph Generation Task (Madaan et al., 2020) showing the source document, the target temporal graph and the corresponding DOT representation.

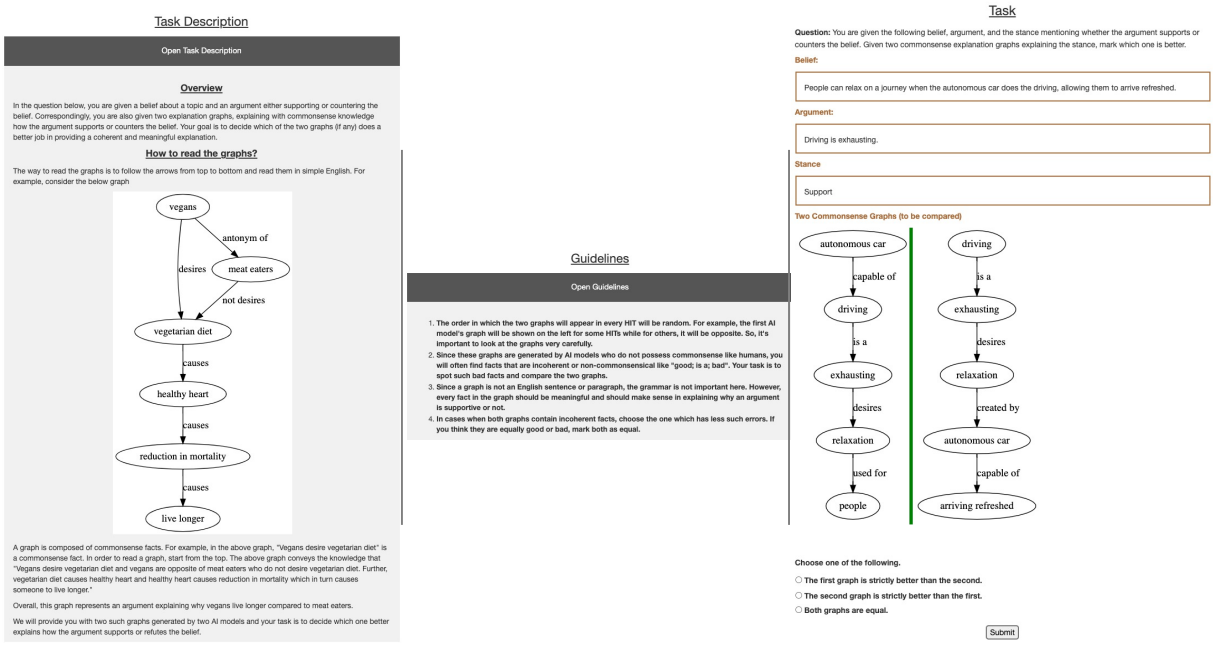

Figure 5: Interface for human evaluation of commonsense explanation graphs.

T5 model on the facts based on ConceptNet relations from ATOMIC-2020 (Hwang et al., 2021), a large-scale commonsense knowledge base. The fine-tuning objective is to predict the target concept given the source concept and the relation. Next, we fine-tune this model further on the end-task of graph generation which leads to small improvements in both StCA and SeCA. This suggests that better methods of inducing commonsense knowledge in these models can potentially lead to bigger gains with more semantically coherent graphs.

F Human Evaluation

In Fig. 5, we show the interface for human verification of commonsense explanation graphs on Amazon Mechanical Turk. We select crowdworkers who are located in the US with a HIT approval rate higher than 96% and at least 1000 HITs approved. Since graph evaluation is a challenging task, we first explain how to read the graphs and also provide clear guidelines for comparing the quality of the two graphs.[9]

G Examples of Generated Explanation Graphs

In Fig. 6, 7, 8 and 9, we show various examples of explanation graphs generated by our models. In Fig. 6 and 7, our proposed models improve upon

[9]The payment for each HIT is 0.25$ at the rate of 12-15$ per hour.

Belief: Since fast foods are greasy and fattening, banning them would control obesity.
Argument: McDonalds has salads.
Stance: Counter

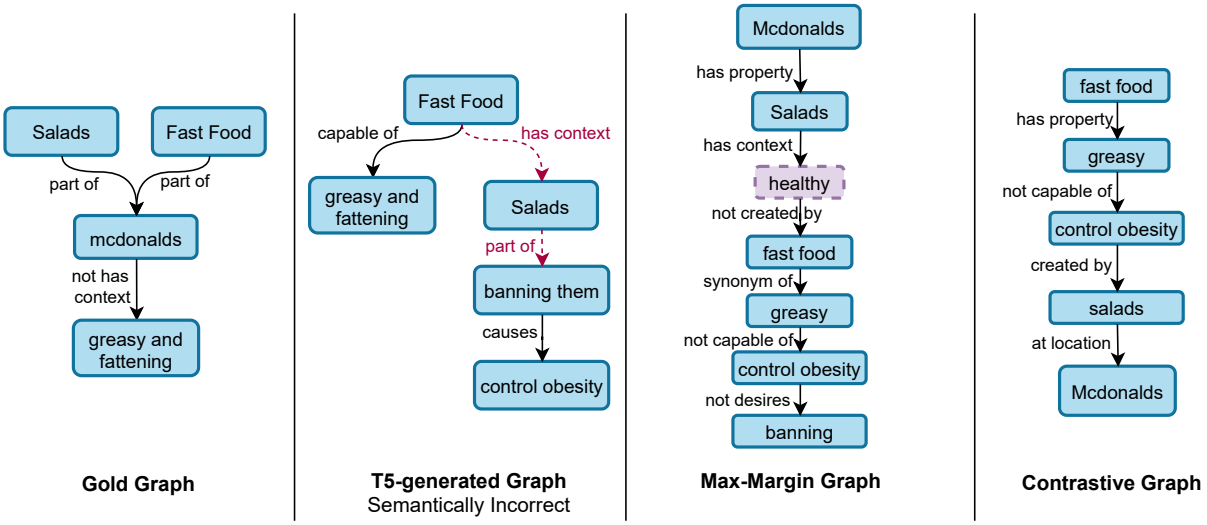

Figure 6: Example of explanation graphs generated by different models. The baseline T5-generated graph is semantically incorrect (incoherent relations marked in dashed red) while our proposed models generate both structurally and semantically correct graphs.

the incorrect semantic relations from the T5 baseline graphs. Fig. 8 shows an example where all generated graphs, while different, are correct. Finally, Fig 9 shows an example where although our proposed models improve the semantic aspect compared to the baseline graph, the generated graphs are disconnected and hence structurally incorrect. Overall, our quantitative results and human evaluation suggest that there is significant room for improvement on the task of commonsense explanation graph generation.

Belief: Homeschooling is not great for children.
Argument: There are plenty of ways for children in homeschooling to socialize.
Stance: counter

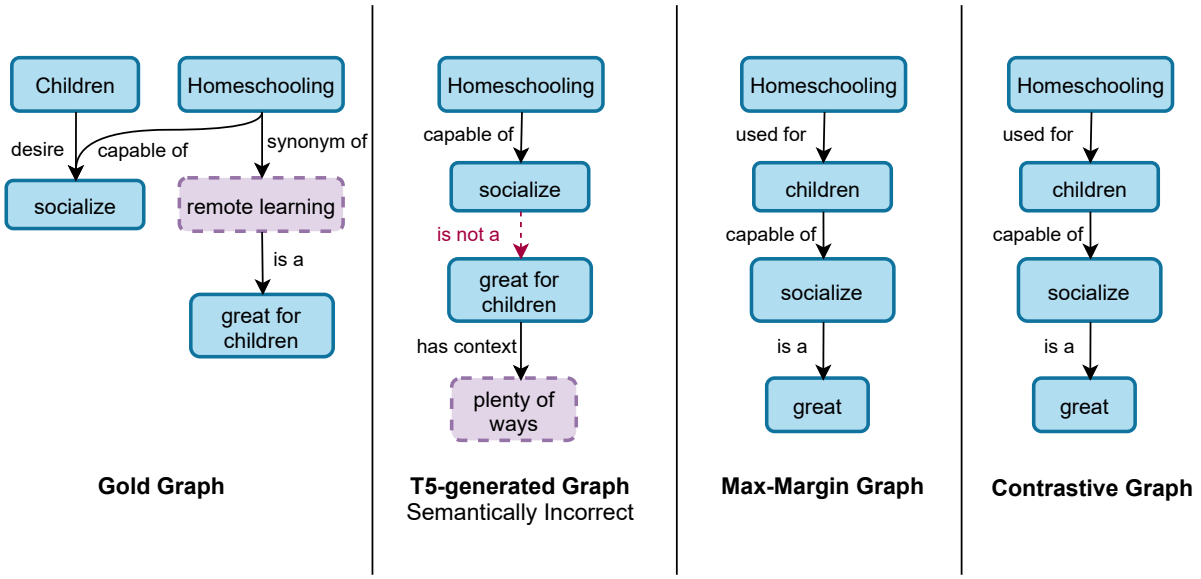

Figure 7: Example of explanation graphs generated by different models. The baseline T5-generated graph is semantically incorrect (incoherent relations marked in dashed red) while our proposed models generate both structurally and semantically correct graphs.

Belief: People can relax on a journey when the autonomous car does the driving, allowing them to arrive refreshed.
Argument: Driving is exhausting.
Stance: support

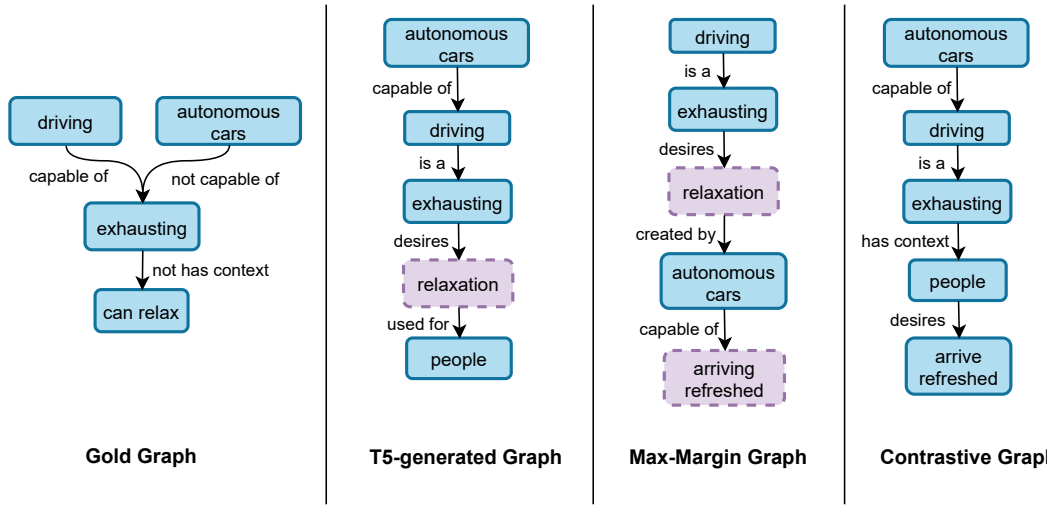

Figure 8: Example of explanation graphs generated by different models. All models generate structurally and semantically correct graphs while the individual nodes and edges differ.

Belief: Autonomous cars are more dangerous than man-driven cars.
Argument: Autonomous cars are not safe for humans.
Stance: support

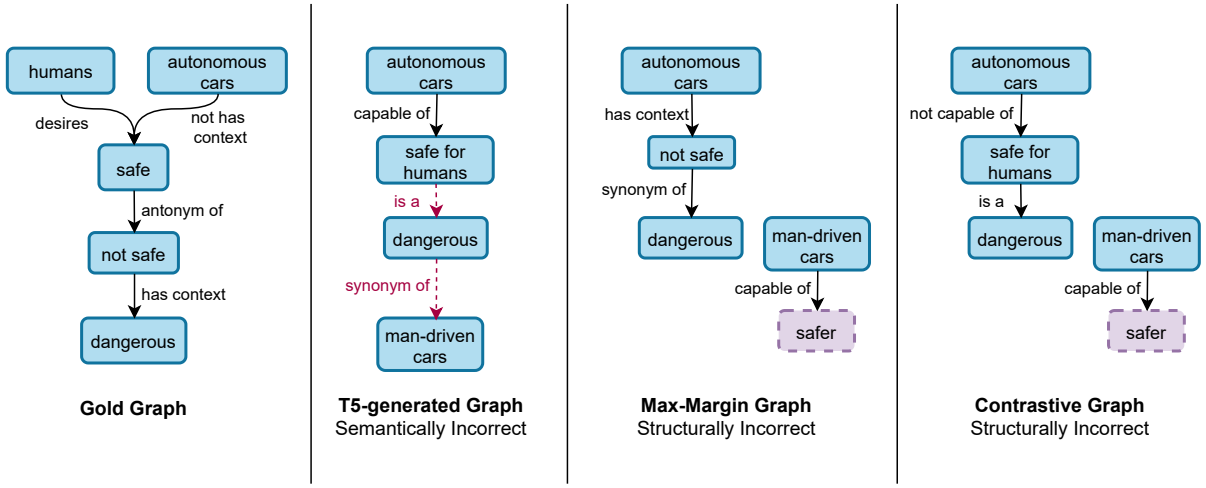

Figure 9: Example of explanation graphs generated by different models. T5 generates a semantically incorrect graph. Our models generate graphs, which while contain meaningful edges, are disconnected and hence are structurally incorrect.