Can Language Models Take A Hint? Prompting for Controllable Contextualized Commonsense Inference

Anonymous ACL submission

Abstract

Generating commonsense assertions, given a certain story context, is a tough challenge even for modern language models. One of the reasons for this may be that the model has to “guess" what topic or entity in a story to generate an assertion about. Prior work has tackled part of the problem, by providing techniques to align commonsense inferences with stories and training language generation models on these. However, none of the prior work provides means to control the parts of a generated assertion. In this work we present “hinting", a data augmentation technique for improving inference of contextualized commonsense assertions. *Hinting* is a prefix prompting strategy that uses both hard and soft prompts. We demonstrate the effectiveness of *hinting* by showcasing its effect on two contextual commonsense inference frameworks: ParaCOMET (Gabriel et al., 2021a) and GLUCOSE (Mostafazadeh et al., 2020a), for both general and context-specific inference.

1 Introduction

The task of generating commonsense assertions or facts, given a certain story context, while easy for humans, remains challenging for machines (Gabriel et al., 2021b). We define an *assertion* as a tuple that contains a subject, a relation, and an object (e.g., *a dog, is a, animal*), similar to a subject-verb-object triple. As we will see, these tuples can be contextual to a story or generally applicable facts. Automated systems (such as pre-trained transformer-based language models (Devlin et al., 2019; Radford et al., 2019)) struggle with generating these assertions, since there is an implicit assumption that clues for making predictions can always be found explicitly in the text. (Zhang et al., 2021; Liu and Singh, 2004; Da and Kasai, 2019; Davison et al., 2019). This becomes problematic because the model is essentially forced to use knowledge that it may not have seen during pre-training. Additionally, models are forced to guess what to predict about (i.e., what the subject of an assertion is), which may lead to decreased performance.

Previous attempts have tackled the problem of contextual commonsense inference by constructing datasets of stories aligned with assertions (i.e. an assertion is given for a sentence in a story), either through automated or human-annotated ways. One previous attempt to tackle this problem, ParaCOMET (Gabriel et al., 2021b), trained a GPT-2 language model (Radford et al., 2019) to infer the object of a commonsense assertion tuple. Given a story and a sentence identifier token, along with a specified relation, the model has to predict the object of a commonsense assertion.

Another work that tries to approach this is GLUCOSE (Mostafazadeh et al., 2020b). Here a dataset is constructed to consist of stories and human annotations for sentences in the stories. The human annotations provide specific and general commonsense assertions. The authors utilize this dataset to train a T5 (Raffel et al., 2020) model to perform contextualized and generalized story assertion inference. The model takes an input sequence in the form of a story, a relation to predict, and a target sentence, and has to predict both the general and specific assertions that may be present in the target sentence with the given story context.

In both works, the models are expected to do their inference from the story, a target sentence, and relation dimension alone. In the case of ParaCOMET, it is expected to predict an object of a specific assertion, and in the case of GLUCOSE, the model is expected to predict both a specific and a general assertion. In either case, this forces the model to rely on knowledge that it may not have seen and it has to guess what to “talk" about.

Recently, there has been work on exploring *prompting* (Liu et al., 2021), which is essentially finding ways of altering the input to a language model such that it matches templates that it has seen during pre-training. This gives stronger performance in tasks, can help with controllability in the case of text generation, and is more parameter-efficient and data-efficient than fine-tuning, in some cases (Li and Liang, 2021). One novel type of prompting is *prefix prompting* (Li and Liang, 2021; Lester et al., 2021). Prefix prompting consists of modifying a language model's input (i.e. prefix) by adding additional words. These words can be explicit hard prompts (i.e., actual words such as “give a happy review") or they can be soft prompts, embeddings that are input into a model and can be trained to converge on some virtual template or virtual prompt that can help the model.

Prompting holds great potential for improving contextualized commonsense inference. We introduce the idea of a *hint*, a hybrid of hard and soft prompts. We define a *hint* as the part(s) of an assertion that a model has to predict, along with special identifiers for these parts, wrapped within parenthesis characters. Syntactically, a *hint* would take the form of: “([subject symbol,subject], [relation symbol,relation], [object symbol,object])" where the actual content of the *hint* between the parenthesis would be a permutation of all but one of the elements in the target tuple. For example, a *hint* would be “(<|subj|>a dog, <|obj|>an animal)") for predicting the tuple “(a dog, is an, animal)". *Hints* are provided by sampling a binomial distribution for each element in a minibatch, which determines whether to give a *hint* or not. The actual content of the *hint* would then be generated by randomly sampling without replacement up to all but one of the elements in a target tuple. We give a more detailed description of *hinting* in section 3.2 and 3.3.

We hypothesize that this scheme of *hinting* strikes a balance between the model recalling information that it may have seen, with information that it may not have seen, that may only be present in the target tuple. Additionally, by providing and training on the combination of hard and soft prompts, a generative language model can be guided to “talk" about a certain subject, object, or relation. We note that the approach was designed to be simple to implement, and to give control when generating text. In the following sections, we give some background on prompting, controllability, and the task of contextual commonsense inference. We follow this by a set of experiments to show the effects of *hinting* within the framing of ParaCOMET and GLUCOSE, and finally, analyze the results, and present future directions for this work. Concretely, our contributions are:

- A hybrid prefix prompting technique called *hinting* that provides a partial assertion to augment data for contextual commonsense inference, and
- Demonstrating that *hinting* does in fact improve the performance for contextual commonsense inference as measured by automated metrics.

2 Related Work

2.1 Prompting

Recently, there has been a shift in paradigm in Natural Language Processing from pre-training and fine-tuning a model, to pre-training, prompting, and predicting (Liu et al., 2021). One primary reason for this shift is the creation of ever-larger language models, which have become computationally expensive to fine-tune. Prompting can be described as converting a pre-trained language model input sequence into another sequence that resembles what the language model has seen during pre-training. Overall, most prompting research is focused on formulating the task as a *cloze* (fill-in-the-blanks) task. However, we consider the task of language generation, an open-ended formulation.

Recall that prefix prompting modifies the input to a language model, by adding either a hard prompt (additional words to the input sequence)(Shin et al., 2020) or a soft prompt (i.e., adding trainable vectors that represent, but are not equivalent to, additional words) (Li and Liang, 2021; Lester et al., 2021; Liu et al., 2021).

Unlike classic prefix prompting, *hinting* uses both hard and soft prompts. The soft prompts are in the form of symbols that represent the different parts of the assertion (i.e., subject, relation type, and object), and the hard prompts are in the form of the actual parts of the assertion that are selected to be appended as part of the *hint*. Our work is similar to KnowPrompt(Chen et al., 2021), except that they use a masked language model and soft prompts for relationship extraction. AutoPrompt (Shin et al., 2020) is also similar, but finds a set of “trigger" words that give the best performance on a *cloze*-related task, whereas we provide specific structured input for the model to guide text generation.

2.2 Controllable Generation

Controllable generation can be described as ways to control a language model's text generation given some kind of guidance. One work that tries to implement controllable generation is CTRL (Keskar et al., 2019). The authors supply control signals during pre-training of a language model. This approach is intended to provide a generally applicable language model. A body of work in controllable generation has focused on how it can be used for summarization. Representative work that uses techniques similar to ours is GSum (Dou et al., 2021). In contrast to GSum, our method is model independent, allows for the source document to interact with the guidance signal, and contains soft prompts in the form of trainable embeddings that represent the parts of a tuple. The GSum system gives interesting insight into the fact that highlighted sentences, and the provision of triples, does in fact help with the factual correctness of abstractive summarization.

2.3 Discourse-aware/Contextual commonsense inference

Commonsense inference is the task of generating a commonsense assertion. Discourse-aware/contextual commonsense inference is the task of, given a certain narrative or discourse, infering commonsense assertions that are coherent within the narrative(Gabriel et al., 2021a). This task is particularly hard because commonsense knowledge may not be explicitly stated in text (Liu and Singh, 2004) and the model needs to keep track of entities and their states either explicitly or implicitly. Research into the knowledge that pre-trained language models learn has yielded good results in that they do contain various types of factual knowledge, as well as some commonsense knowledge(Da and Kasai, 2019; Petroni et al., 2019; Davison et al., 2019). The amount of commonsense knowledge in these models can be improved by supplementing sparsely covered subject areas with structured knowledge sources such as ConceptNet (Speer et al., 2017; Davison et al., 2019).

Knowing that these pre-trained language models may contain some commonsense information has led to the development of knowledge models such as COMET(Bosselut et al., 2019). This line of research has been extended from the sentence-by-sentence level in COMET, to the paragraph-level in ParaCOMET (Gabriel et al., 2021a). Contemporaneously, GLUCOSE Mostafazadeh et al. (2020a) builds a dataset of commonsense assertions that are contextualized to a set of stories, and generalized (e.g., *John is a human* is generalized to *Someone_A is a human*).

In this work, we explore the effect of *hinting*, in both the ParaCOMET setting for discourse-aware commonsense inference, and the GLUCOSE setting, which includes generalization.

3 Modeling

3.1 Task

Our general task is the following. We are given a story S composed of n sentences, $S = \{S_1, S_2, \ldots, S_n\}$, a target sentence from that story, S_t, where $S_t \in S$, and a dimension/relation type R. Given all this, we want to generate a tuple in the form of $(subject, R, object)$ that represents an assertion, present or implied, in S_t given the context S, and the relation type R.

In our initial ParaCOMET experiments, we represent S_t with a unique token rather than repeating the sentence. Additionally, we only generate the $object$ of the tuple. In our second set of experiments, the T5 GLUCOSE experiments, we represent S_t by marking it with $*$ on the left and right of the sentence. Additionally, we generate two tuples, one that is the context-specific tuple, and the other is the general tuple. We explain the experiments in detail shortly.

3.2 Hinting

The mechanism we present in this work, called *hinting*, is a kind of mixed/hybrid prompting for generative language models. Prompting is essentially supplying additional text (i.e. prompts) to a language model to aid/guide it in a specific task. In our case, we opt to give a "hint", as to what the assertion that we want to predict contains, at the end of our input text. This can be seen as a hybrid of prompting the generative model with hard prompts of parts of what it should predict along with soft prompts of symbols that represent those parts. We structure it this way such that, after training, whenever a *hint* is given, the model can be guided to generate knowledge about the *hint's* content.

To balance the model's reliance on the context, its knowledge, and the *hint*, we determine whether to supply the *hint* by sampling a binomial distribution. Thus, we can control the frequency of when to supply a *hint* (perhaps a lower frequency for easier

Model Type	Example Input	Example Output
ParaCOMET	A dog is man's best friend. Usually people go to walk dogs. When walking dogs, people get free exercise in addition to a breath of fresh air. <|sent1|><|xWant|>	to get exercise
ParaCOMET **with Hint**	A dog is man's best friend. Usually people go to walk dogs. When walking dogs, people get free exercise in addition to a breath of fresh air. <|sent1|><|xWant|>**(<|subj|>A person is at the park)**	to see other dogs
GLUCOSE	1: A dog is man's best friend. *Usually people go to walk dogs.* When walking dogs, people get free exercise in addition to a breath of fresh air.	Person has a dog >Causes>Person to walk dogs ** Someone_A has Something_A (that is a dog) >Causes> Someone_A to walk Something_A (that is a dog)
GLUCOSE **with Hint**	1: A dog is man's best friend. *Usually people go to walk dogs.* When walking dogs, people get free exercise in addition to a breath of fresh air. **(<|specific|><|subj|>A person is at the park)**	A person is at the park >Causes>A person sees other dogs ** Someone_A is Somewhere_A (that is a park) >Causes> Someone_A to see Something_A (that is a dog)

Table 1: Example of inputs for experiment set 1 (rows 1 and 2) and experiment set 2 (rows 3 and 4). In rows 2,4 we can see an example of what a *hint* could do. In row 2 we can see that giving the subject of "A person is at the park" guides the model to a generation of wanting to see other dogs. In row 4 we can see a similar example but with generalization. The *hint* is **bolded** in the examples.

tasks. and a higher frequency for harder tasks). Additionally, the content of the *hint* is determined by random sampling of permutations of components, up to a maximum of all but one component. Since our task is to predict the tuple, we do not want to make the model overly reliant on *hints* for the answer. This approach is relatively naive, however. In the future we can explore the freedom to construct more complex ways of supplying *hints*. In the section 4 we give details on how we use *hints* for our ParaCOMET and GLUCOSE experiments.

3.3 An example of Hinting

A simple example of *hinting* is the following:

Story: *A dog is man's best friend. Usually people go to walk dogs. When walking dogs, people get free exercise in addition to a breath of fresh air.*

Target sentence: *Usually people go to walk dogs.*

Target assertion: *(subject: People, relation: are capable of, object: walking a dog.)*

A *hint* can be any permutation of the target assertion, except the complete assertion, along with some symbol that indicates which part it is:

Possible Hints: *(<|subj|> People), (<|subj|> People, <|rel|> capable of), (<|subj|> People, <|obj|> walking a dog), (<|rel|> Capable of, <|obj|> walking a dog), (<|obj|> Walking a dog), (<|rel|> Capable of)*

Putting everything altogether, a *hint* for the given story, target sentence and target assertion, yields the following:

Hint: *(<|subj|> People, <|rel|> capable of)*

3.4 Models

For our first set of experiments, we utilize the ParaCOMET (Gabriel et al., 2021b) framing with a BART(Lewis et al., 2020) model and a T5(Raffel et al., 2020) model to observe the effects of *hinting* in a sequence-to-sequence formulation. We use the off the shelf pretrained "base" version of these models for efficiency in the computations. We also utilize the Huggingface Transformers (Wolf et al., 2019) library implementation of these models. For our second set of experiments we utilize the T5 model in our experiments. In the next section we give details on the setups that we utilize for each set of experiments.

4 Experimental Setup

4.1 Experiment Description

We run two sets of experiments to show the effectiveness of *hinting*. The first is utilizing the original ParaCOMET problem framing and adding *hints* to this framing. This framing consists of given a story S composed of n sentences, $S = \{S_1, S_2, \dots, S_n\}$, a relation type R, and a target sentence token (i.e. $< |sent0| >, < |sent1| >, \dots, < |sent(n-1)| >$), we must predict the object of a triple, utilizing the sentence as a subject and the supplied relation R. Within this framework, after the relation R, we add our *hint* between parenthesis (i.e. "([*hint*])"). In this framing, our *hint* can be composed of: a subject symbol along with the target sentence (i.e., the subject of the triple), a relation symbol along with the relation type or dimension R, or an object symbol along with the object of the triple. Using the example in section 3.3, a possible *hint* in this set of experiments would be: "(<|subj|> Usually people go to walk dogs, <|rel|> Capable of)".

We utilize the same formulation as the original work only with a BART and T5 model. We note that the T5 and the BART model utilize a sequence-to-sequence(Sutskever et al., 2014) formulation, which requires encoding a source sequence, and decoding it into a target sequence whereas Para-COMET originally used a cross-entropy loss for the concatenation of the story, target sentence, relation, and object, and a GPT-2 model (Radford et al., 2019). For the T5 model, we add the prefix "source:" before the story S, and the prefix "hint:" for placing our *hints*, to be consistent with the prefix training that the T5 model has. For simplicity, we construct the same "heuristic""-based dataset that ParaCOMET which utilizes a heuristic matching technique to align ATOMIC(Sap et al., 2019) triples to story sentences.

For our second set of experiments, we utilize the formulation utilized in GLUCOSE(Mostafazadeh et al., 2020b). The formulation utilizes the T5 model in a sequence-to-sequence formulation once more. In this formulation, the source text is composed of a prefix of a dimension to predict $D \in 1, 2, \dots 10$[1], followed by the story S with the marked target sentence. The target sentence, S_t, is marked with $*$ before and after the sentence. An example input is: "1: The first sentence. *The target sentence. * The third sentence.". This task is slightly different from the ParaCOMET one, in that in addition to predicting a context specific triple, the model has to predict a generalized triple. In this task we have to infer a general and context specific subject, object and a relation. For our *hints* we provide up to five out of these six things, along with a symbol that represents whether it is the subject, object, or relation, and another symbol that represents whether it is part of the general or specific assertion. We add our *hint* after the story S, utilizing the prefix "hint:" and supplying the *hint* between parenthesis. In this set of experiments, an example of our *hint* can be, given the example in section 3.3: "(<|specific|> <|obj|> walking a dog, <|general|> <|obj|> walking something that is an animal)".

[1]The definition for each dimension number is given in the GLUCOSE work

4.2 Experiment Configuration

We run the first set of experiments for 3 epochs on the dataset's training data and evaluation data. We utilize a max source sequence length for the BART and T5 models of 256, and a max target length of 128. Additionally, we use the ADAM (Kingma and Ba, 2015) optimizer with a learning rate of 2e-5, and a linear warm-up of 0.2 percent of the total iterations. We utilize the same system to generate the training data and testing data that ParaCOMET uses. We also utilize a batch size of 4 for training. In this set of experiments we generate our aggregation scores from the final score at the end of every epoch.

We run the second set of experiments for 1 epochs on the data. Additionally, we utilize a linear warm up of 3000 steps. We utilize the ADAM optimizer with a learning rate of 1e-5, a train batch size 4, and a max source length of 1024 and max target length of 384. We utilize the training and testing data given by the GLUCOSE work. In this set of experiments, we calculate the scores on the test set every 2000 iterations and these are the scores we utilize to report our aggregation.

In both experiments we report the scores given by SacreBLEU(Post, 2018), ROUGE (Lin, 2004), and METEOR(Banerjee and Lavie, 2005) using the datasets library (Lhoest et al., 2021) metrics system to get a sense of how good the commonsense inferences are, and we run both experiments with 4 different variations or seeds each.

5 Results and Analysis

5.1 Experiments Set 1: ParaCOMET formulation with *hints*

The aggregated results for this set of experiments can be found in table 2. We can see here that on average, *hinting* does tend to improve the score even if slightly. It seems that providing a *hint* is beneficial and not detrimental for contextual commonsense inference. We also notice that *hinting* has very similar average and median values, which may mean that performance is more consistent. This can be expected as *hinting* is a mechanism to make the model focus on generating inferences about whatever is provided on the *hint*, which may eliminate spurious generations and make the overall outputs more consistent. Given the way that this task is framed, a possibility that could explain the relative similarity of the performances, is that *hinting* *only* adds the object of the triple as additional pos-

	BLEU		METEOR		ROUGE-1		ROUGE-2		ROUGE-L		ROUGE-L-SUM	
Model	**No Hint**	**Hint**	**No Hint**	**Hint**	**No Hint**	**Hint**	**No Hint**	**Hint**	**No Hint**	**Hint**	**No Hint**	**Hint**
BART (Average)	41.803	**41.834**	**58.933**	58.907	61.127	**61.136**	50.059	**50.184**	61.059	**61.079**	61.065	**61.084**
BART (Median)	41.818	**41.824**	58.914	**58.929**	**61.199**	61.142	50.083	**50.191**	**61.138**	61.084	**61.136**	61.087
T5 (Average)	37.857	**37.897**	56.263	**56.329**	55.313	**55.429**	43.718	**43.787**	55.274	**55.384**	55.275	**55.387**
T5 (Median)	37.824	**38.036**	56.321	**56.298**	55.367	**55.488**	43.712	**43.839**	55.323	**55.431**	55.330	**55.436**

Table 2: Median and average scores for the first set of experiments utilizing the ParaCOMET formulation with and without *hinting*. The largest scores are **bolded**.

Model	**BLEU**		**METEOR**		**ROUGE-1**		**ROUGE-2**		**ROUGE-L**		**ROUGE-L-SUM**	
	No Hint	Hint	No Hint	Hint	Hint	Hint	No Hint	Hint	No Hint	Hint	No Hint	Hint
T5 (Avg. of Max)	40.242	**44.367**	**40.550**	**43.069**	53.881	**56.209**	34.702	**36.817**	50.321	**52.61**	50.329	**52.617**
T5 (Avg.)	30.237	**32.147**	33.953	**34.761**	**46.925**	**47.562**	28.334	**28.791**	43.32	**44.08**	**43.330**	**44.083**
T5 (Median)	33.578	**36.002**	36.163	**36.787**	49.787	**50.496**	30.864	**31.479**	46.149	**47.257**	46.143	**47.293**

Table 3: Results for the GLUCOSE formulation set of experiments. We include the average, median, and the average of the maximum values per run. The largest scores are **bolded**.

sible data that the model may see during training; the subject and the relation can be repeated with *hinting*.

5.2 Experiments Set 2: GLUCOSE formulation with *hints*

The aggregated results for this set of experiments can be found in table 3. For this particular formulation we notice that *hinting* does tend to improve the performance when evaluating the average of the maximum scores per variation, the average, and median of the scores. This suggests that *hinting* is indeed beneficial for the task of contextualized commonsense inference, especially when faced with the harder task of generating both a general and context dependent assertion. We believe that this improvement is because *hinting* gives the model the clues it may need to decide on what to focus or attend to, to generate useful inferences.

In our tests we noticed that the performance of the system tends to degrade over time (at least according to the automated metrics). Because of this, in addition to the median and average of our scores, we report the average of the maximum scores in our variations.

5.3 Why *hint*?

From the results in both sets of experiments, we can see that *hinting* tends to increase the performance of contextualized commonsense inference. However, in some cases, the performance increase is minimal. This brings the question of: Why *hint* at all? The primary reason is for controllability in the generation. By supplying these *hints*, we are teaching the model pay attention and talk about a certain subject, relation, or object. This in turn, after training, can be leveraged by a user or downstream application to guide the model to generate assertions from parts that they supply (i.e., if they supply a subject, it will generate assertions of the subject). Although this is not very clear within the ParaCOMET formulation, it becomes clearer in the GLUCOSE formulation of the problem. We give an illustrative example of the usefulness of *hinting* in table 1. We can see that by giving a model the *hint*, the model could be capable of inferring about information that may not be present in the story. Although the effectiveness of this is not explicitly tested in this work, we do note that this behavior is useful in downstream tasks such as story understanding and contextual knowledge graph generation.

Lastly, *hinting* was designed to be simple to implement, and is model independent. Given all of these, there is very little work that has to be done to incorporate *hinting* in the contextual commonsense inference task.

5.4 Is *hinting* optimal?

This work was a proof of concept for this technique. We acknowledge there is a large body of research on the area of prompting. The way the *hinting* mechanism was designed however, leaves much space to explore alternate mechanisms such as AutoPrompt(Shin et al., 2020), or including additional soft prompts such as those in Li and Liang (2021). Because of the naivety of the approach, we do not think it is an optimal approach, and there is a large body of research that points to manual templating of prompts being less effective than learned prompts (Liu et al., 2021). However, from our tests, it does not degrade performance, and seems to only improve it.

6 Future Work

When designing the *hinting* system certain aspects were formulated to leave space for improvements. One area that can be improved upon is finding a smarter way of selecting when to *hint*, and finding a smarter way of selecting what to *hint*. In addition to this, more soft prompts could be added to the *hint* such that they would learn an optimal virtual template or soft-prompt that could improve even further performance on this task.

Another area that is open for future work is providing deeper ablation studies to determine what parts of the *hint* are more effective and when. This work is more a proof-of-concept that *hinting*, or more broadly prompting, is applicable towards this task of contextual commonsense inference. Furthermore, given that models trained with *hinting* for contextual commonsense inference can be guided by the information supplied in *hints*, such models can be utilized in a variety of downstream applications such as story understanding and contextual knowledge graph generation.

7 Conclusion

In this work we presented *hinting*, a simple hybrid prompting mechanism that consists of appending parts of a target tuple into an input sequence for the task of contextual commonsense inference. We showed that *hinting* tends to improve performance in automated metrics, and when it does not improve performance, it is comparable to not utilizing *hinting*. With this, we open the doors for exploring prompting within the realm of contextual commonsense inference.

References

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, Ann Arbor, Michigan. Association for Computational Linguistics.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Çelikyilmaz, 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 (ACL)*.

Xiang Chen, Xin Xie, Ningyu Zhang, Jiahuan Yan, Shumin Deng, Chuanqi Tan, Fei Huang, Luo Si, and Huajun Chen. 2021. Adaprompt: Adaptive prompt-based finetuning for relation extraction. *CoRR*, abs/2104.07650.

Jeff Da and Jungo Kasai. 2019. Cracking the contextual commonsense code: Understanding commonsense reasoning aptitude of deep contextual representations. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 1–12, Hong Kong, China. Association for Computational Linguistics.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Zi-Yi Dou, Pengfei Liu, Hiroaki Hayashi, Zhengbao Jiang, and Graham Neubig. 2021. GSum: A general framework for guided neural abstractive summarization. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4830–4842, Online. Association for Computational Linguistics.

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021a. Paragraph-level commonsense transformers with recurrent memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12857–12865.

Saadia Gabriel, Chandra Bhagavatula, Vered Shwartz, Ronan Le Bras, Maxwell Forbes, and Yejin Choi. 2021b. Paragraph-level commonsense transformers with recurrent memory. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(14):12857–12865.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Quentin Lhoest, Albert Villanova del Moral, Patrick von Platen, Thomas Wolf, Yacine Jernite, Abhishek Thakur, Lewis Tunstall, Suraj Patil, Mariama Drame, Julien Chaumond, Julien Plu, Joe Davison, Simon Brandeis, Victor Sanh, Teven Le Scao, Kevin Canwen Xu, Nicolas Patry, Steven Liu, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Nathan Raw, Sylvain Lesage, Anton Lozhkov, Matthew Carrigan, Théo Matussière, Leandro von Werra, Lysandre Debut, Stas Bekman, and Clément Delangue. 2021. huggingface/datasets: 1.13.2.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Hugo Liu and Push Singh. 2004. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. 2020a. Glucose: Generalized and contextualized story explanations. *arXiv preprint arXiv:2009.07758*.

Nasrin Mostafazadeh, Aditya Kalyanpur, Lori Moon, David Buchanan, Lauren Berkowitz, Or Biran, and Jennifer Chu-Carroll. 2020b. GLUCOSE: GeneraLized and COntextualized story explanations. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4569–4586, Online. Association for Computational Linguistics.

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

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium. Association for Computational Linguistics.

Alec Radford, Jeff 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(140):1–67.

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

Taylor Shin, Yasaman Razeghi, Robert L Logan IV, Eric Wallace, and Sameer Singh. 2020. Autoprompt: Eliciting knowledge from language models with automatically generated prompts. *arXiv preprint arXiv:2010.15980*.

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*.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Zhihan Zhang, Xiubo Geng, Tao Qin, Yunfang Wu, and Daxin Jiang. 2021. Knowledge-aware procedural text understanding with multi-stage training. In *WWW '21: The Web Conference 2021, Ljubljana, Slovenia, April 19–23, 2021*.