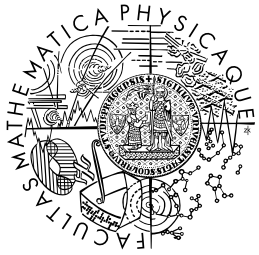

FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Bc. Kristína Szabová

Neural Concept-to-text Generation with Knowledge Graphs

Institute of Formal and Applied Linguistics

Supervisor: Mgr. et Mgr. Ondřej Dušek, Ph.D.

Study program: Computer Science - Language Technologies and Computational Linguistics

Prague 2023

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

First and foremost, I would like to thank my supervisor Ondřej Dušek for his excellent advice and endless patience. A big thank you also belongs to my family (especially my sister Viktória, who wanted to be mentioned by name), and my boyfriend, who have supported me during the whole course of my studies.

Title: Neural Concept-to-text Generation with Knowledge Graphs

Author: Bc. Kristína Szabová

Institute: Institute of Formal and Applied Linguistics

Supervisor: Mgr. et Mgr. Ondřej Dušek, Ph.D., Institute of Formal and Applied Linguistics

Abstract: Modern language models are strong at generating grammatically correct, natural language. However, they still struggle with commonsense reasoning — a task involving making inferences about common everyday situations without explicitly stated information. Prior research into the topic has shown that providing additional information from external sources helps language models generate better outputs. In this thesis, we explore methods of extracting information from knowledge graphs and using it as additional input for a pre-trained generative language model. We do this by either extracting a subgraph relevant to the context or by using graph neural networks to predict which information is relevant. Moreover, we experiment with a post-editing approach and with a model trained in a multi-task setup (generation and consistency classification). Our methods are evaluated on the CommonGen benchmark for generative commonsense reasoning using both automatic metrics and a detailed error analysis on a small sample of outputs. We show that the methods improve over a simple language model fine-tuning baseline, although they do not set a new state of the art.

Keywords: natural language generation, concept to text generation, knowledge graph, natural language processing

Contents

Introduction **2**

1 Motivation **4**
1.1 Generative commonsense reasoning 4
1.2 Related work 7
1.2.1 Fine-tuned off-the-shelf language models 7
1.2.2 Knowledge-graph-based methods 8
1.2.3 Retrieve-and-generate models 9
1.2.4 Other methods 10

2 Background **12**
2.1 Text generation 12
2.1.1 Language models 12
2.1.2 Encoder-decoder models 14
2.1.3 Transformer models 15
2.1.4 Pre-trained language models 18
2.1.5 Large language models 20
2.2 Knowledge bases 21
2.3 Graph neural networks 22
2.3.1 Graph 22
2.3.2 GNNs and message passing 23
2.3.3 GNN tasks 23

3 Experiments **25**
3.1 Baseline 25
3.2 Enhancing inputs using simple graph algorithms 26
3.3 Information extraction using GNNs 28
3.4 Post-editing 29
3.5 Composite loss 30
3.6 Evaluation 31

4 Results **32**
4.1 Automatic evaluation 32
4.2 Manual evaluation 36

Conclusion **52**

References **54**

A Selected outputs of the experiments **64**

Introduction

> *"A man provided with paper, pencil, and rubber, and subject to strict discipline, is in effect a universal machine."*
>
> Alan Turing

The opening quote inspires one to imagine a man sitting behind a desk, surrounded by heaps of paper, holding a pencil in one hand and rubber in the other one; bags under his eyes but a slight, tired smile, for he has just made a groundbreaking discovery. Most people easily deduce that the rubber in question refers to a small rectangular object used for erasing pencil marks. It is unlikely that anyone — especially those familiar with said stationery — would think of a car tire in this situation. This is in spite of the fact that the quote does not explicitly mention erasing errors.

While such a deduction may seem trivial, it actually requires years of lived experience. Humans are able to reason about the world because they have been observing it all their life, often without the facts being explicitly mentioned. They have acquired an immense amount of *commonsense knowledge* that they use constantly to infer unstated information.

Even though it may come naturally to humans, commonsense reasoning remains a challenging task for computers. Commonsense reasoning involves critical aspects of language, such as the capacity to resolve ambiguity[1] or co-references.[2] Modern language models — while excellent at grammar, language understanding, and a wide range of other tasks — still often struggle to make inferences if they require a substantial amount of world knowledge.

Specifically, generative commonsense reasoning is an integral part of natural language generation (NLG) tasks. Contemporary NLG is done using language models (LMs), such as GPT [Radford et al., 2019] or BART [Lewis et al., 2020]. NLG requires the language model to be able to understand commonsense relations in the generated text. Failure to do so leads to unrealistic, often nonsensical texts. Conversely, language models strong at commonsense reasoning produce natural, contextually appropriate generations; they can infer implicit information which is useful, e.g., in conversational agents; they are able to detect and prevent inconsistencies and contradictions. At the end of the day, commonsense reasoning is vital if we want to deploy a system in the real world.

The CommonGen benchmark dataset [Lin et al., 2020] has been developed to test language models' commonsense capabilities. The benchmark's objective is to generate a commonsense sentence from a set of 3 to 5 words, also called *concepts*. The examples range from easy concept sets that frequently occur together to difficult ones where the complex relationship between the concepts is not obvious and a model must possess strong commonsense reasoning capabilities

[1] Homonyms can lead to completely different interpretations of a text. Consider the sentence: *A man shoots his girlfriend.* The meaning is drastically different depending on whether the shooting is done with a camera or with a gun.

[2] The Winograd Schema Challenge [Levesque et al., 2012] is a famous commonsense reasoning challenge. The task is to disambiguate references for pairs of twin sentences that differ in one word; this word determines what the pronoun refers to. Solving the challenge requires commonsense reasoning.

to uncover it.

Current research in generative commonsense reasoning generally explores various ways of adding relevant information to the model on input. The prevalent approach is the retrieve-and-generate approach in many variations [Fan et al., 2020, Li et al., 2021, Yu et al., 2022]. Retrieve-and-generate models involve extracting a sentence relevant to the concepts from an external source and using it as a basis for generating a prediction. The external source is "the entire Internet" — a large multi-domain collection of unstructured documents. The individual approaches differ in both the retrieval and the generating methods used.

In this thesis, we explore using knowledge graphs to find relevant information to feed to the model. Unlike retrieve-and-generate models, we use a structured source to extract knowledge. Some researchers claim that knowledge graphs do not contain enough commonsense knowledge to be especially useful for the task, which is, perhaps, why this direction has been relatively under-investigated so far. However, we believe that their structured nature presents a promising opportunity.

In our approach, we fine-tune pre-trained language models and provide additional information on input. We propose several input enhancements with knowledge extracted from a knowledge graph. First, we extract different types of subgraphs from the knowledge graph containing some or all of the input concepts. The linearized subgraphs are used as an additional input to a language model. Second, we train a graph neural network (GNN) model to predict whether two concepts can occur in the same sentence. The predicted concepts are added to the input on top of the original input concept set. Moreover, we propose to refine these models' predictions by using a model trained to add missing concepts to a sentence. Finally, we explore a model trained to perform two tasks, classification and language generation, simultaneously.

Some of our methods lead to significant improvements over a simple language model fine-tuning approach. They manage to produce sentences that reproduce commonsense relations well, although they still struggle with concept sets with more complex relationships. While we do not achieve a new state of the art on the CommonGen benchmark, we show that using knowledge graphs is a promising direction of research that deserves further exploration.

This thesis is structured into four main parts. In Chapter 1, we define the task of generative commonsense reasoning and show the research that has been done in this area. Chapter 2 provides a background for the concepts and techniques used throughout this thesis, such as language modeling and knowledge graphs. In Chapter 3, we provide a description of the experiments we perform with the objective of improving at commonsense language generation over a fine-tuning baseline approach. Finally, we discuss the results of our experiments in Chapter 4, where we present both an automatic evaluation and a manual analysis. We conclude with a short summary of our results and potential suggestions for future work on this task.

1. Motivation

The objective of this thesis is to explore methods aimed at addressing commonsense reasoning. Commonsense reasoning continues to pose significant challenges for computers. Extracting implicit information from text or images requires computer programs to possess or be able to acquire a profound understanding of the world. While humans accumulate world knowledge from their day-to-day experience, computer programs lack the inherent ability to do so and therefore must be provided with the necessary information through alternative means.

In this chapter, we describe the subject in detail. In Section 1.1, we formulate the task of commonsense reasoning; specifically, the task of generative commonsense reasoning. The section contains a description of the CommonGen benchmark which we use to evaluate our models, as well as the metrics used for the evaluation. Section 1.2 provides an overview of methods used to advance the state of the art in generative commonsense reasoning. We show a variety of approaches to injecting commonsense information into a language model, including retrieval from external sources and different pre-training regimes.

1.1 Generative commonsense reasoning

According to Lin et al. [2020], the ability to understand and use concepts from our surrounding environment, i.e., commonsense reasoning, is a significant milestone in human development.

Multiple challenges have been designed to test computers' capacity to perform commonsense reasoning. Some of them are designed as discriminative tasks: the model is trained to choose the most likely answer from a set of options. Such discriminative challenges include Commonsense QA (Talmor et al. [2019]), Social IQA (Sap et al. [2019]), and HellaSwag (Zellers et al. [2019]).

Commonsense reasoning is hard. Davis and Marcus [2015] identify five key challenges of the commonsense reasoning task: poor understanding of the domains involved, their logical complexity, the involvement of plausible reasoning (making conclusions that are likely but not guaranteed to be correct from partial information), the *long-tail* problem (a small number of examples is very frequent in a corpus, while a much larger number only occurs once), and choosing the proper level of abstraction.

Another issue with machine commonsense reasoning lies in the fact that due to the extent of the domain that commonsense reasoning involves (in essence, the entire human experience), it has proved difficult to design benchmarks that measure computers' performance on the task. Each such benchmark focuses on a narrow subdomain only, thus a model performing well on one benchmark does not necessarily perform well on others.

In this work, we focus on commonsense reasoning from the generative perspective. In simple words, the task of generative commonsense reasoning (commonsense text generation) consists of generating a sentence from a set of keywords

(concepts). This sentence should describe a common real-life scenario. For example, consider the set of concepts from Lin et al. [2020]:

dog, frisbee, catch, throw,

and their corresponding two sentences:

*The **dog catches** the **frisbee** when the boy **throws** it.* (1.1)

*A **dog throws** a **frisbee** and a **dog catches** it.* (1.2)

We can tell that sentence 1.1 is consistent with our everyday experience, while sentence 1.2 violates it. Therefore, our goal, given these concepts, is to generate a sentence such as sentence 1.1.

The formal definition of generative commonsense reasoning, as given by Lin et al. [2020], is the following. The task is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that maps a set of k concepts (a concept set) $x = \{c_1, c_2, \ldots, c_k\} \in \mathcal{X}, c_i \in \mathcal{C}\ \forall i$, where $\mathcal{C}$ denotes the concept vocabulary and $\mathcal{X}$ denotes the space of all possible concept sets, to a grammatical sentence $y \in \mathcal{Y}$ that describes and ordinary real-life scenario and uses all concepts from x.

This thesis explores techniques aiming to improve the performance of baseline systems on the generative commonsense reasoning task. We use the CommonGen benchmark dataset [Lin et al., 2020][1] to train and test our systems.

Each individual data point in the CommonGen dataset consists of a concept set (consisting of three to five concepts), a concept-set index (an integer that identifies the concept set), and a target sentence. Multiple target sentences can be associated with each concept set.

The concept sets in the dataset were sourced from various image- and video-captioning datasets, such as Flickr30k [Young et al., 2014], MSCOCO [Lin et al., 2014], and LSMDC [Rohrbach et al., 2015]. After part-of-speech-tagging the captions, the authors selected concept sets consisting of 3-5 words based on their frequency and scene diversity. This procedure ensures that the concept sets in the dataset are not completely arbitrary (there exist common situations that can be described using all of the concepts), and, at the same time, they reflect the natural distribution of concept sets in the real world.

The original captions were then discarded. The target sentences were crowd-sourced from *Amazon Mechanical Turk* with manual quality control.

Table 1.1 shows the basic statistics of the dataset. We can see that the validation and test partitions have a larger average number of references, allowing more freedom in text generation. In addition, the test set does not contain any concept pairs present in the train set, meaning that a well-performing model must possess the ability to generalize to unseen pairs.

The dataset authors identify two key challenges of CommonGen. The first one is **relational reasoning**. To achieve good performance, the models must recall relevant relational commonsense facts about the concepts, including spatial relations, object properties, temporal event knowledge, social conventions, etc. It is possible that these facts are not included in any existing knowledge base. The second challenge is **compositional generalization**: the models must be able to reason about sets of concepts even if they have never co-occurred before.

[1]The dataset is available at `https://huggingface.co/datasets/common_gen`.

Statistics	train	dev	test
# Concept sets	**32,651**	**993**	**1,497**
- size = 3	25,020	493	-
- size = 4	4,240	250	747
- size = 5	3,391	250	750
# Sentences	67,389	4,018	7,644
per concept set	2.06	4.04	5.11
Average length	10.54	11.55	13.28
# Unique concepts	4,697	766	1,248
# Unique concept pairs	59,125	3,926	8,777
# Unique concept triples	50,713	3,766	9,920
# Unseen concepts	-	6.53%	8.96%
# Unseen concept pairs	-	96.31%	100%
# Unseen concept triples	-	99.60%	100%

Table 1.1: Basic statistics of the CommonGen dataset.

CommonGen evaluation

Models tested on the CommonGen benchmark are evaluated using a range of automatic metrics which we describe below.

BLEU The BLEU metric [Papineni et al., 2002] is one of the most widely used automatic metrics in natural language processing. Its simplicity makes it fast (therefore, suitable for testing many iterations of a model) and easily interpretable. The BLEU metric is based on computing the overlap between a prediction and one or more references. First, we define the n-gram precision for a given n over a test set:

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n\text{-}gram \in C} \text{count}_{clip}(n\text{-}gram, C)}{\sum_{C' \in \{Candidates\}} \sum_{n\text{-}gram' \in C'} \text{count}(n\text{-}gram', C')} \tag{1.3}$$

where $\text{count}_{clip}(n\text{-}gram, C)$ is the maximum number of times the n-gram occurs in a single C's reference sentence and $\text{count}(n\text{-}gram, C')$ is the number of times the n-gram occurs in C'. In other words, p_n is the proportion of correctly predicted n-grams out of all predicted n-grams.

We also define the *brevity penalty* BP that penalizes short sentences and thus discourages the model from generating very short outputs containing very probable words only. Let c be the length of the candidate sentence and r the length of the reference. Then,

$$\text{BP} = \begin{cases} 1 & \text{if } c > r \\ e^{(1-r)/c} & \text{if } c \leq r \end{cases}. \tag{1.4}$$

The BLEU score of the corpus is then computed as

$$\text{BLEU} = \text{BP} \cdot \exp(\sum_{n=1}^{N} w_n \log p_n) \tag{1.5}$$

Essentially, the BLEU metric computes a weighted average of n-gram precisions, balancing the prediction's adequacy (meaning the correct choice of words,

corresponding to 1-grams) and fluency (captured by longer n-grams). The standard maximum n-gram length is 4.

CIDEr The CIDEr metric [Vedantam et al., 2015] was originally developed for image captioning. It measures the similarity of the candidate to the *human consensus*. The computation involves calculating the overlap of n-grams of length 1 to 4 between the candidate and each reference sentence. Each n-gram of length n is assigned a Term Frequency Inverse Document Frequency (TF-IDF) [Robertson, 2004] score with respect to the candidate and a TF-IDF score with respect to the reference. The scores are aggregated in two vectors (corresponding to the candidate- and reference-related scores, respectively) for which we calculate their cosine similarity. The final CIDEr score is computed by taking a weighted average of the cosine similarities between vectors corresponding to different values of n.

SPICE Another metric originating from the field of computer vision is the SPICE metric [Anderson et al., 2016]. The metric eliminates the sensitivity to n-grams, which limited previous major metrics. Instead, it focuses on the *semantic propositional context* of the candidate and reference sentences.

The metric compares a candidate-reference pair by transforming both of them into a *scene graph* by parsing the sentence's syntactic dependencies and transforming it according to a set of rules. The score is calculated as the F-score over the triples corresponding to the relations in the resulting graph.

It is possible to use other standard NLP metrics to evaluate systems, such as ROUGE [Lin, 2004], developed for text summarization evaluation, or METEOR [Banerjee and Lavie, 2005], originally designed for machine translation. However, the dataset authors consider CIDEr and SPICE to be the most appropriate for the tasks since they focus on the associations between the concepts. In this work, we focus on BLEU, CIDEr, SPICE, and *coverage*, which is the proportion of input concepts that appear in the predictions.

1.2 Related work

Commonsense reasoning has long been of great interest among researchers who have developed a variety of methods to tackle the task. At the point of writing this thesis, systems still struggle to provide consistently good results, although recent advancements have led to significant improvements.

We provided examples of commonsense reasoning benchmarks in Section 1.1. In this section, we focus specifically on approaches to generative commonsense reasoning evaluated on the CommonGen benchmark [Lin et al., 2020]. Some concepts used in this section are described in more detail in Chapter 2. The results of the systems we describe in this section are shown in Table 1.2.

1.2.1 Fine-tuned off-the-shelf language models

Lin et al. [2020] provide a baseline performance score for the CommonGen dataset by fine-tuning standard transformer-based language models: GPT-2 [Radford et al., 2019], BERT-Gen [Mitzalis et al., 2021], UniLM [Dong et al., 2019],

Model	BLEU-4	CIDEr	SPICE
GPT-2	23.73	12.19	23.57
BERT-Gen	23.47	12.61	24.82
UniLM	30.73	14.89	27.43
BART-base	29.01	13.98	28.00
T5-base	18.54	9.40	19.87
T5-large	31.96	15.13	28.86
CoNT + T5-base [An et al., 2023]	31.96	15.12	28.86
EKI-BART [Fan et al., 2020]	35.95	17.0	29.58
KG-BART [Liu et al., 2020]	33.87	16.93	29.63
SAPPHIRE (T5-large) [Feng et al., 2021a]	37.12	16.90	29.75
VisCTG (BART-large) [Feng et al., 2021b]	36.94	17.20	29.97
NeuroLogic A*esque [Lu et al., 2022]	39.60	17.29	30.13
RE-T5 [Wang et al., 2021]	40.86	17.66	31.08
Imagine-and-Verbalize [Wang et al., 2022]	40.57	17.72	31.29
PU-GEN (T5-large) [Seo et al., 2022]	38.23	18.04	31.68
KGR$\hat{4}$ (BART) [Liu et al., 2022]	42.82	18.38	33.56
KFCNet [Li et al., 2021]	43.62	18.85	33.91
RACo [Yu et al., 2022]	43.62	19.14	34.03
DKRM [He et al., 2022]	44.33	19.54	34.59
Human performance (upper bound)	46.49	37.64	52.43

Table 1.2: Methods for generative commonsense reasoning measured on the CommonGen benchmark. Scores measured on the test set and reported by Lin et al. [2020].

UniLM-v2 [Bao et al., 2020], BART [Lewis et al., 2020], and T5 [Raffel et al., 2019]. Additionally, the authors provide scores obtained by humans which can be considered the upper bound.

1.2.2 Knowledge-graph-based methods

Some works on commonsense generative reasoning incorporate knowledge graphs. Knowledge graphs such as ConceptNet [Speer et al., 2017] contain world knowledge in the form of relations between common concepts. The advantage of knowledge graphs is that they have a well-defined schema, making it easy to use them programmatically.

However, knowledge graphs also lack a substantial amount of knowledge. Since common sense covers an immense number of topics, it is to be expected that a human-constructed knowledge base does not contain every possible link. This limitation suggests that using knowledge graphs in commonsense-reasoning models may not lead to optimal results.

Liu et al. [2020] use knowledge graphs and graph attention to enhance language models' commonsense reasoning. Their framework, Knowledge Graph-Augmented BART (KG-BART), works in two steps: knowledge-graph grounding and graph-based encoder-decoder modeling.

In the first step, the authors construct two graphs. The concept-reasoning graph $\mathcal{G}^R$ is established by matching the input concepts to the entities in Con-

ceptNet. The concept-expanding graph $\mathcal{G}^E$ is then constructed by ranking neighboring nodes in $\mathcal{G}^R$ according to a word-similarity score.

$\mathcal{G}^E$ is used in the second step in a graph-augmented encoder and decoder. The authors use a graph-informed attention mechanism to incorporate the graph-structure information into token embeddings, as well as the decoding process. The underlying encoder-decoder structure to which the graph attention is attached is a fine-tuned BART language model [Lewis et al., 2020].

In general, knowledge graphs have rarely been used in generative commonsense reasoning research, possibly due to the limitations outlined above. Nevertheless, we believe that they have a potential to improve a language model's performance. This thesis attempts to find novel approaches to exploit the knowledge contained in them, as described in Chapter 3.

1.2.3 Retrieve-and-generate models

Some of the best-performing commonsense-reasoning systems are based on the retrieve-and-generate framework. While they differ in the exact details, the general process is the same. First, the system retrieves knowledge related to the input concepts from an external source. The source generally consists of unstructured documents collected from the internet. The retrieved information is then used to construct prototype sentences. By *prototype sentence* we mean any sentence that somehow resembles the desired output sentence but is not necessarily the final output. The prototypes, along with the input concepts, are then passed into a language model (generally, encoder-decoder models such as BART [Lewis et al., 2020] or T5 [Raffel et al., 2019], but decoder-only models are possible too) which is trained to generate the reference sentences.

The prototype's function in the framework is to compensate for the lack of world knowledge in the language model. As noted by Yu et al. [2022], retrieval-augmented methods work well because of their flexibility in accessing world knowledge. Unlike, for example, knowledge-graph-augmented methods, which are limited by the relation schema, retrieval-augmented allow access to almost unlimited sources of knowledge, which is why they tend to perform better (see Table 1.2).

Fan et al. [2020] propose the Enhanced Knowledge Injection BART (EKI-BART). This retrieve-and-edit framework uses several image- and video-captioning datasets as sources of external knowledge to retrieve a single prototype sentence by matching the input concepts. The prototype is then edited with the help of the language model to generate the final sentence.

As an example, consider the concepts

front, guitar, sit.

An external source might provide the prototype

A singer performed the song standing in front of the audiences while playing guitar.

The prototype lacks the concept *sit*, which is why it cannot be used as the final output. However, it already contains the remaining concepts and thus provides some necessary context.

The authors then construct the input to the language model by concatenating the concepts and the prototype. The language model (in this case, BART [Lewis et al., 2020]) is then trained to generate, for example, the sentence

A singer sitting in front of the audiences while playing guitar.

Similar to Fan et al. [2020], Wang et al. [2021] propose a retrieve-and-generate method with a trainable retriever. Their method, Retrieval-Enhanced T5 (RE-T5), involves training a retriever model to predict the score of the prototype sentences extracted from an external source. The concatenation of the concepts and the best-ranked prototypes is then used as input to T5 [Raffel et al., 2019].

[Liu et al., 2022] propose a four-stage Knowledge-enhanced Commonsense Generation framework. The four stages are Retrieve, Retrospect, Refine, and Rethink. In the first stage (Retrieve), prototype sentences are retrieved by a rough mapping from an external source. Similarly to Wang et al. [2021], a scorer is trained to select the most plausible prototypes. In the second stage (Retrospect), a BART [Lewis et al., 2020] is trained to either copy or edit the prototypes to generate candidate sentences. The candidate sentences are refined using another BART model to remove potential errors in the third stage (Refine). In the last stage (Rethink), the authors employ the scorer trained in the first stage to select the best sentence within the generated candidates.

The Knowledge Filtering and Contrastive Learning Network (KFCNet) [Li et al., 2021] first employs a BERT-based [Devlin et al., 2019] model to rank prototype sentences retrieved from an external source. Then, it uses *contrastive learning* in both the encoder and the decoder. It picks an anchor sample from which it constructs both positive and negative samples. The model learns to pull the anchor close to the positive examples and push it away from the negative examples in the embedding space.

He et al. [2022] further improve the retriever by distilling the knowledge from a metric (e.g., BLEU [Papineni et al., 2002]). They claim that previous approaches' retrievers often do not provide the most relevant prototypes as they use sparse methods depending only on the concepts for the retrieval. They train the retriever to pick sentences that achieve a high score with respect to the reference sentences, thus improving the quality of the prototypes.

Yu et al. [2022] construct a unified framework for Retrieval-Augmented Commonsense Reasoning (RACo) to solve various commonsense tasks. They collect over 20 million documents from various sources containing commonsense knowledge and propose strategies for efficient document retrieval from the corpus in order to boost retrieval-based models' performance. Such strategies include using two models to encode the query and the document separately and training them to minimize the distance between the encodings.

1.2.4 Other methods

Other methods have been proposed to tackle the commonsense generative challenge.

An et al. [2023] propose to use contrastive learning to improve the performance of a T5 model [Raffel et al., 2019]. In addition to positive examples, the model is also provided with negative examples. It is then trained to predict the

likelihood of the generated sentence, as well as the distance between the input to the model and the provided sentence. During inference, the two objectives are then combined to select the best sentence generated by the model.

The SAPPHIRE framework (Set Augmentation and Post-hoc Phrase Infilling and Recombination), proposed by Feng et al. [2021a], uses two augmentation methods: keyword-based augmentation that selects new keywords based on the similarity of their embeddings to the original concept set, and attention-based augmentation where the concept set is augmented by those keywords that have been most attended to in the human references.

Feng et al. [2021b] introduce the Visually Grounded Concept-to-Text Generation (VisCTG) framework enhanced with visual information. They motivate this approach by the fact that images generally depict common, plausible situations. It involves obtaining images for each concept set and captioning them using a pre-trained captioning model. The concepts and captions are then concatenated to form the input of a language model which is then trained to produce the final sentences.

Lu et al. [2022] note that in common open-ended text-generation approaches, the cost (i.e., probability) of generating the next word only depends on previously-generated text; whereas constrained text generation would benefit from estimating future cost as well. Taking inspiration from the A* algorithm [Russell and Norvig, 2010, Chap. 3], they develop the NeuroLogic A*esque heuristic that combines decoding algorithms with future cost estimation. Specifically, the heuristic relies on lexical constraint satisfaction which guides the selection of candidates in the next decoding step.

Wang et al. [2022] design the Imagine-and-Verbalize framework based on constructing a scene knowledge graph (SKG) from the concepts (the imagination module) and generating a sentence based on the SKG (the verbalization module). The SKG construction is formulated as an autoregressive sequence-generation task conditioned on the context and the concepts with the objective being to construct a plausible scene involving the concepts. The linearized SKG is then fed into a language model which produces the final sentence.

Seo et al. [2022] PU-GEN, incorporate both scene and relational knowledge into the generative process. To enhance the model with scene knowledge, they collect human-generated image captions. For each concept set, they then retrieve those captions that contain at least two of the concepts. Relational knowledge is retrieved from Wikipedia articles. A language model is then pre-trained to generate information the retrieved information from concepts. This method extends the commonsense knowledge contained in the language model. Finally, the model is fine-tuned on the CommonGen dataset.

In conclusion, most methods for generative commonsense reasoning attempt to inject world knowledge into the model by providing additional information on input. We follow this approach as well in Chapter 3. However, instead of using external knowledge sources as do most models in this section, we explore the opportunities provided by knowledge graphs.

2. Background

In Chapter 1, we introduced the generative commonsense reasoning task along with its recent approaches. The primary objective of this thesis is to explore new approaches to address this task. Before we discuss our experiments in Chapter 3, it is essential to acquire an understanding of the techniques employed. This chapter aims to establish a background to our methods.

First, in Section 2.1, we explain text generation — an essential pre-requisite for generative commonsense reasoning — and related concepts. The most important one is language modeling which underlies most of the current state-of-the-art for language generation. Next, we introduce knowledge bases and, specifically, knowledge graphs in Section 2.2, which we use to inject external information into the language models. Finally, in Section 2.3, we describe graph neural networks which we use to capture higher-order relationships between nodes in ConceptNet.

2.1 Text generation

Text generation constitutes the basis of a large portion of today's state-of-the-art systems in machine translation [Kocmi et al., 2022], speech recognition [Interspeech, 2022], conversational dialogue [Ji et al., 2022], and others. In text generation, the system usually generates text based on some other text or another medium: in machine translation, it is the text in the source language; in speech recognition, it might be the speech signal.

This section starts by introducing the notion of a *language model*. Then, we describe how language modeling is useful in text generation, presenting approaches from N-gram models to modern-day large neural language models.

2.1.1 Language models

The fundamental component of many modern state-of-the-art text-generating systems is a language model. The role of language models is to predict the next word in a sequence of words; they are usually trained to produce natural-sounding, grammatically correct text. In addition to language technologies, language models have found their application even in seemingly unrelated fields such as biotechnology.

Formally, according to Jurafsky and Martin [2023, Chap. 3], a language model is a probability distribution over sequences of words, $P(w_1, w_2, \ldots, w_n), w_i \in \mathcal{V}\ \forall i$, where $\mathcal{V}$ denotes the vocabulary. We can equivalently reformulate the definition as a *next-word* prediction task by leveraging the chain rule of probabilities:

$$\begin{aligned} P(w_{1:n}) &= P(w_1)P(w_2|w_1)P(w_3|w_{1:2})\ldots P(w_n|w_{1:n-1}) \\ &= \prod_{k=1}^{n} P(w_k|w_{1:k-1}). \end{aligned} \tag{2.1}$$

Each factor except for the first one is in the form $P(w|h)$, i.e., the probability that the word w follows the history h. Therefore, a language model can be trained to model the probability of a word following a specific history.

N-gram language models. In general, h may be of unlimited length. Historically, modeling the probability distribution over sequences of an arbitrary length has been a difficult task. Instead, language models used to exploit the Markov assumption

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-1}) \tag{2.2}$$

which can be generalized to

$$P(w_n|w_{1:n-1}) \approx P(w_n|w_{n-k+1:n-1}) \tag{2.3}$$

for some $k \in \mathcal{N}$.

In other words, the Markov assumption says that the probability distribution over sequences of words of an arbitrary length can be approximated by a probability distribution over sequences of words whose length is bounded from above by an integer k.

A class of language models that use the Markov assumption is the class of N-gram language models. These models are trained to estimate the probability of N-grams, i.e., sequences of words of length N. For example, a bigram language model ($N = 2$) estimates the probability $P(w_n|w_{n-1})$, while a trigram language model ($N = 3$) estimates the probability $P(w_n|w_{n-2}, w_{n-1})$. The probability of a sequence of words using an N-gram language model is then computed as

$$P(w_{1:n}) \approx \prod_{k=1}^{n} P(w_k|w_{k-N+1:k-1}) \tag{2.4}$$

where the first N words' probabilities are estimated by padding the history with special *beginning-of-string* tokens.

The N-gram probabilities are calculated using the *maximum-likelihood estimation* according to the following formula:

$$P(w_N|w_{1:N-1}) = \frac{C(w_{1:N})}{\sum_w C(w_{1:N-1}w)} \tag{2.5}$$

where $C(w_{1:N})$ and $C(w_{1:N-1}w)$ denote the number of times the word sequences $w_{1:N}$ and $w_{1:N-1}w$ appeared in the training data, respectively.

Additional techniques can be used in probability estimation, such as smoothing or backoff to avoid zero-probabilities.

RNN language models. While N-gram language models achieve relatively high performance while being simple to implement, they suffer from major limitations stemming from the fact that they only see a very limited part of history.

Models from the family of recurrent neural networks (RNN) counteract this limitation. RNN models work by processing the input sequence of an arbitrary length one output at a time, storing an intermediate representation of the already processed part of the sequence. Formally, the process for sequences of words is as follows:

1. The words are converted into word-embedding vectors [Mikolov et al., 2013]. Word embeddings capture the meanings of words in a latent space of a dimension much smaller than the size of the vocabulary, thus making it easier to work with them than, e.g., one-hot vector representations.

2. The first word embedding e_1 and a pre-initialized hidden-state vector h_0 are passed through an RNN cell. The cell can be a simple neural network, such as a perceptron, or it can have a more complicated architecture, such as *long short-term memory* LSTM [Hochreiter and Schmidhuber, 1997].

3. The RNN cell outputs two vectors: an output y_1 for the current time step and a hidden state h_1.

4. The hidden state h_1 and the second word embedding e_2 are passed through the same RNN cell (i.e., it has the same weights). The cell produces a new output y_2 and a new hidden state h_2.

5. The process is repeated until the whole input sequence is consumed.

6. The final hidden state, theoretically, captures the information from the whole sequence and can be used for further tasks.

RNNs model language by learning the probability distribution over the vocabulary given a history. Each output y_t is a vector whose dimension is equal to the size of the vocabulary; each component of the vector determines the probability of the following word being the word corresponding to the component's index. Mathematically, the probability of the word sequence $P(w_{1:n})$ can be expressed as

$$\begin{aligned} P(w_{1:n}) &= \prod_{t=1}^{n} P(w_t|w_{1:t-1}) \\ &= \prod_{t=1}^{n} \mathbf{y_t}[w_t]. \end{aligned} \tag{2.6}$$

2.1.2 Encoder-decoder models

In many natural language processing tasks, the expected generated text usually varies depending on some input, such as text in the source language in machine translation or dialogue history in dialogue systems. For such tasks, it is convenient to use an encoder-decoder model.

The encoder-decoder model based on RNNs [Sutskever et al., 2014] consists of two main components: the encoder that processes the input and outputs its representation, and the decoder that generates text based on this input representation. The encoder-decoder model is often used in tasks with inputs or outputs of variable length.

The encoder processes the input sequence as shown in Section 2.1.1. After consuming the whole sequence, it produces the final hidden state, also called the *context vector*. The decoder, which is another RNN, produces an output sequence as follows:

1. The first input to the RNN is the context vector and a pre-initialized hidden state.

2. The inputs are passed through an RNN cell, which produces a hidden vector and an output modeling the distribution of the vocabulary.

3. The output vector determines which word will be the next word in the output sequence. One can use *greedy decoding*, meaning that at each time step, the word with the highest probability is selected, or a more complex algorithm, such as *beam search* [Zarrieß et al., 2021].

4. The RNN cell receives the selected word (its embedding) and the previous hidden state as input in the next time step.

5. The process is repeated until the decoder produces an end-of-string token or the sequence reaches a predetermined length.

The process of decoding in which the next word in the sequence is selected based on the words that the decoder output previously is called *autoregressive* decoding.

2.1.3 Transformer models

RNN attention-based encoder-decoder systems outperformed previous state-of-the-art machine translation approaches [Bahdanau et al., 2016] [Luong et al., 2015]. However, RNNs have a significant disadvantage because the input must be processed one token at a time, rendering the computation impossible to parallelize.

A novel architecture based solely on *attention mechanisms*, the Transformer architecture [Vaswani et al., 2017], has been proposed. An attention mechanism enables the model to focus on specific parts of the sequence. While previous encoder-decoder RNN models already used attention [Luong et al., 2015], Transformer's major improvement over them is the fact that it processes the entire input sequence at once instead of sequentially. The most important consequence is that computations can be parallelized, enabling larger input sizes during training and more training data.

The architecture models global dependencies between words by introducing the attention mechanism into the network. Attention performs well at capturing long-distance dependencies in the decoder because the number of computation steps required to model the relationship between the generated words is constant regardless of the words' position. This is a significant improvement over RNN architectures where modeling the relationship between words at distance n requires $\mathcal{O}(n)$ steps [Vaswani et al., 2017].

Figure 2.1 shows the architecture of a Transformer model. We provide an explanation of the model based on Alammar [2018]. The Transformer model consists of two major components: the encoder and the decoder. Both are composed of an identical number of encoder and decoder layers (6 in the original paper). The role of the encoder and the decoder is similar to the encoder and decoder from previously-described RNN networks: the former captures the information from the input, while the latter autoregressively generates the output based on this information.

Each encoder layer further consists of a self-attention layer and a feedforward network. Additionally, the architecture employs residual connections around each sublayer, followed by layer normalization.

A decoder layer is almost identical to an encoder layer with the difference being an additional encoder-decoder attention sublayer between the self-attention and

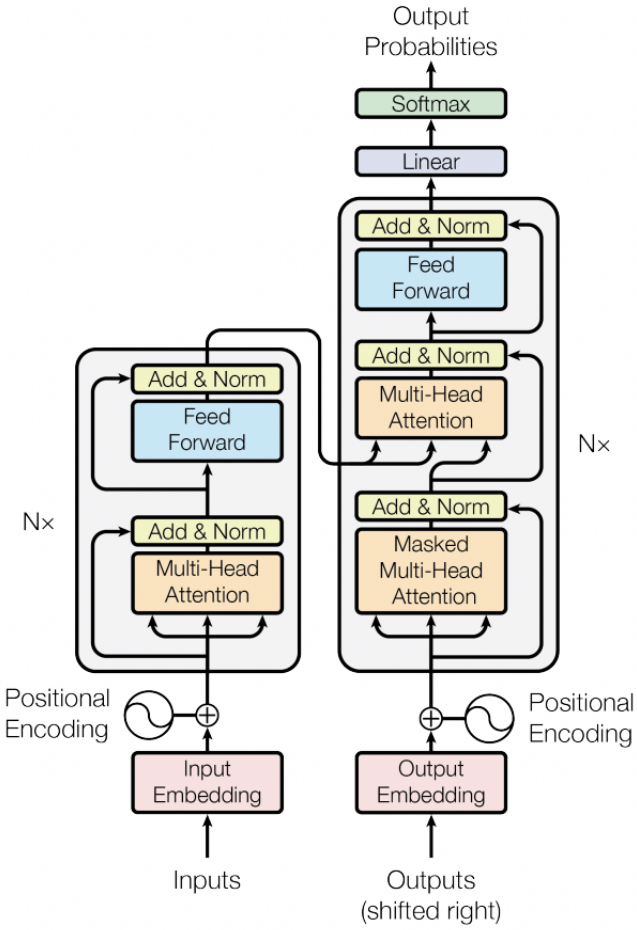

Figure 2.1: The Transformer model architecture [Vaswani et al., 2017].

the feedforward sublayer. The self-attention sublayer is modified in such a way that the decoder can only attend to previous positions. This ensures that each newly-generated token depends on previous tokens only.

A sequence is processed as follows:

1. The words from the input sequence are converted into word embeddings. Additionally, they are summed with positional encoding vectors in order to distinguish the same word at different positions.

2. The first encoder layer receives the word representations as input. The entire input sequence at once is passed through all sublayers. The layer produces a sequence of the same length as the input sequence.

3. The other encoder layers' input is the previous layer's output. They also process the whole sequence at once.

4. The decoder works the same way as the encoder. The input to the decoder (either the target output sequence during training or a single end-of-string token during inference) is converted into word embeddings and summed with positional encodings.

5. The sequence passes through each decoder layer.

6. The encoder layers' outputs are also passed to the decoder layers where they are used in the encoder-decoder attention sublayers.

7. The output of the top decoder layer is further processed by a linear layer to generate a logit vector whose size corresponds to the size of the model's vocabulary.

8. Finally, we apply the softmax function to obtain normalized probabilities from the logits; these are used to decide on the next token.

The decoding works slightly differently depending on whether we are doing training or inference. During training, we pass the entire target output sentence through the decoder at once, masking certain tokens so that the decoder can only attend to the left context of each token. During inference, we start with a single end-of-string token. After it passes through the decoder, the next token is generated. The process must be repeated until we generate the entire sequence.

Attention is the most important part of the model. Transformer uses two types of attention, self-attention and encoder-decoder attention, that only differ in what is being attended to. The authors describe attention as mapping a query and a set of key-value pairs to an output. The queries, keys, and values are computed by multiplying a vector $\mathbf{X}$ by a weight matrix:

$$\begin{aligned} \mathbf{Q} &= \mathbf{X} \times \mathbf{W^Q} \\ \mathbf{K} &= \mathbf{X} \times \mathbf{W^K} \\ \mathbf{V} &= \mathbf{X} \times \mathbf{W^V} \end{aligned} \tag{2.7}$$

where $\mathbf{W^Q}$, $\mathbf{W^K}$, and $\mathbf{W^V}$ are trainable parameters. In self-attention, $\mathbf{K}$, $\mathbf{V}$, and $\mathbf{Q}$ are derived from the previous layer's outputs, whereas in encoder-decoder attention, $\mathbf{K}$ and $\mathbf{V}$ are constructed from the encoder's output.

Attention is then computed as

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^\top}{\sqrt{d_k}})V. \tag{2.8}$$

More descriptively, attention is a weighted sum of the values where weights are determined by a combination of the query and the corresponding key. The scaling factor $\sqrt{d_k}$ (d_k denotes the key vectors' dimension) is used for reasons of numerical stability.

The Transformer model actually uses the so-called multi-head attention. The intuition behind it is that each head can focus on different aspects of the input [Voita et al., 2019]. We train multiple sets of parameters $\mathbf{W_i^Q}$, $\mathbf{W_i^K}$, and $\mathbf{W_i^V}$ for each head. We then calculate $\mathbf{Q_i}$, $\mathbf{K_i}$, and $\mathbf{V_i}$ as in Equation (2.7). The attention for each head is then computed according to Equation (2.8) using the head's respective parameters. The outputs of each head are then concatenated and projected into the desired space.

2.1.4 Pre-trained language models

After the Transformer was first introduced, a range of architectures and pre-trained models based on it were released. While the original Transformer was developed for and tested on the machine translation task, these new models target a wider variety of tasks.

Pre-trained language models can be used for downstream tasks by adding an additional task-specific layer on top of the base model. The model is then fine-tuned on task-specific data either by training the top layer only, training the entire model, or anything in-between. With some exceptions, these pre-trained models are publicly available.[1] Depending on the task, we distinguish encoder-only, decoder-only, and encoder-decoder Transformer-based models.

Encoder-only models. For tasks such as text classification, it is not strictly necessary to have a decoder as all necessary information from the input should be captured by the encoder. This motivates the development of encoder-only Transformer-based models, such as BERT [Devlin et al., 2019] or RoBERTa [Zhuang et al., 2021].

BERT, which stands for Bidirectional Encoder Representations from Transformers, is an encoder-only language representation model. BERT processes the input sequence bidirectionally, meaning that each token representation models information from both its left and right context.

BERT's architecture is identical to the encoder from the original Transformer. The input sequence to BERT always starts with a special `[CLS]` token whose representation is used for downstream sentence-level tasks. If the input sequence consists of two or more sentences (e.g., in question-answering tasks), they are separated by a special `[SEP]` token.

BERT is pre-trained on two unsupervised tasks. The first one is **masked language modeling (MLM)**. It is similar to language modeling in that it predicts an unknown word from the context. However, while standard language modeling predicts the word from previous tokens only, MLM predicts a masked word from

[1] The pre-trained models are available, for example, on `https://huggingface.co/models`.

both previous and following tokens. This form of pre-training enables the model to learn token representations from both the left and right contexts.

The second pre-training objective is **next-sentence prediction**. The model is given two sentences `A` and `B` and it is trained to predict whether `B` follows `A`. This way, the model is trained to capture the relationship between the two sentences in the `[CLS]` token representation.

Fine-tuning BERT is done by either passing the `[CLS]` token representation through a feedforward network for sentence-level (classification) tasks or by passing each token representation through a feedforward network for token-level tasks.

BERT outperformed the state-of-the-art in a broad set of tasks including paraphrasing, natural language inference, question answering, and others.

Decoder-only models. Some tasks can be modeled as generating a plausible continuation to a text. In this case, the input plays the role of a *prompt* to a model. The most famous in this class is the family of GPT models [Radford et al., 2018, 2019, Brown et al., 2020].

Generative Pretrained Transformer, or **GPT** [Radford et al., 2018], was developed for language understanding tasks. The authors only used the decoder from the original Transformer. Each decoder layer consists of a self-attention and a feedforward sublayer; the encoder-decoder attention sublayer is missing as there is no encoder.

The authors first trained a base GPT model on a **language modeling** objective. They fine-tuned the model for the individual tasks by taking the representation of the last token from the top transformer layer and passing it through a feedforward network. As all of the benchmarks are classification tasks, the training objective is the following:

$$\mathcal{L}(x, y) = \log P(y|x) \tag{2.9}$$

for all example-label pairs (x, y).

GPT achieved state-of-the-art results in a number of tasks including natural language inference [Bowman et al., 2015], question answering [Lai et al., 2017], and text similarity [Dolan and Brockett, 2005].

Encoder-decoder models. These models are suitable for tasks that require a conversion between the input and output sequences (e.g., between languages). In addition to the original Transformer, this family of models consists of BART [Lewis et al., 2020], T5 [Raffel et al., 2019], or LLaMa [Touvron et al., 2023].

BART is a denoising autoencoder for pre-training sequence-to-sequence models. Its name comes from the fact that it combines Bidirectional and Autoregressive Transformers.

BART's architecture is identical to the original Transformer. The model is trained by optimizing a *reconstruction loss* on documents corrupted by an arbitrary noising function. The noising functions used by the authors are token masking, token deletion, text infilling (replacing a span of text with a single `[MASK]` token, thus forcing the model to learn to predict the missing span length), sentence permutation, and document rotation.

The model performs especially well at generation tasks, outperforming previous state-of-the-art models at summarization [Narayan et al., 2018], dialogue [Dinan et al., 2020], and abstractive question answering [Fan et al., 2019]. It matches the results of state-of-the-art models in discriminative tasks. Additionally, the authors showed that the model also performed well at machine translation.

The authors noted that the major limitation of the model is its tendency to hallucinate.

2.1.5 Large language models

In recent years, the language-model research paradigm has shifted from the pre-training and fine-tuning approach to prompting. Radford et al. [2019] showed that language models perform well in zero-shot settings since many tasks, such as translation, occur naturally in corpora on which the models are trained.

This gives rise to the idea of large language models (LLMs). Their architecture is largely similar to previous models such as GPT, except it is scaled many-fold to a size of billions of parameters and trained on very large amounts of data. LLMs, such as GPT3 [Brown et al., 2020], can be used as-is without any further task-specific fine-tuning. As LLMs are typically decoder-only models, we can accomplish specific tasks by using the input as a prefix and letting the model generate a continuation. This method is called *prompting*.

Generally, scaling the models leads to better performance [**?**Chowdhery et al., 2022]. Additionally, new abilities arise in LLMs that are not present in smaller models. This phenomenon is referred to as *emergence* and includes abilities such as chain-of-thought reasoning, or the ability to answer questions truthfully, among others [Wei et al., 2022]. However, increasing the parameter count yields diminishing returns [Chowdhery et al., 2022]. Other techniques have thus been developed to advance the state of the art.

Hoffmann et al. [2022] found that contemporary large language models are significantly undertrained. They proposed to double the amount of training data with every doubling of the number of parameters. Their *Chinchilla* model, trained on an adequate number of tokens, matches the performance of larger models and even outperforms them at some tasks.

While LLMs perform well at language modeling, they are not necessarily aligned with the user. This manifests as generating untruthful, toxic, or generally unhelpful outputs. Such behavior can be partially rectified by training an instruction-following model, such as InstructGPT [Ouyang et al., 2022]. InstructGPT is trained by fine-tuning a pre-trained GPT-3 model. First, a team of human writers provides demonstrations of the desired outputs given a specific prompt. The base GPT-3 model is fine-tuned on this dataset. Second, human labelers rank different model outputs for the same input. A reward model is trained to predict human preference. Finally, InstructGPT is fine-tuned using a reinforcement-learning algorithm with the reward model used as a reward function.

Fine-tuning GPT-3 in this manner showed promising results. InstructGPT's outputs were significantly preferred by human labelers. The model also improved in truthfulness and toxicity over GPT-3. The instruction-following paradigm has since been widely adopted in publicly available LLMs, such as ChatGPT [OpenAI,

2022].

Models' instruction-following capabilities can be further improved by using the Self-Instruct framework [Wang et al., 2023a]. Self-Instruct uses a bootstrapping algorithm to generate new instructions or tasks for instruction-following models. Thanks to this process, models can learn to follow more diverse instructions, as human-written instructions often suffer from the fact that they tend to be common NLP tasks, as noted by the authors.

LLMs generally require too much computing power to be run locally even for inference only and are usually only accessible via online APIs. However, techniques such as mixed-precision training and inference [Raschka, 2023] enable the models to consume fewer resources with minimal impact on performance. Such techniques may lead to smaller-scale equally-powerful models in the future. Still, LLMs currently require significantly more computational resources than models such as BERT and BART.

2.2 Knowledge bases

Knowledge bases, in general, are sources of information about a topic, a service, a product, etc [Atlassian]. Typically, they are stored in a machine-readable form that facilitates easy retrieval of information contained in them; this is necessary because they commonly constitute components in software systems that need external information to provide accurate information to users.

If a knowledge base is constructed as a set of objects connected by (possibly different types of) relations, we refer to it as a *knowledge graph*.

DBPedia[2] [Lehmann et al., 2015] is a well-known example of a knowledge base. It is a community-built multidomain ontology that started by collecting infoboxes from Wikipedia.[3] Its entries are organized in hierarchies and can be described using over 3000 properties.

Another large knowledge base is Wikidata[4] which stores structured data from Wikipedia and related projects. The information contained in it is multilingual and community-sourced. Similarly to DBPedia, its entities are connected by different types of relations.

In NLP applications, knowledge bases often represent relationships between words. One of the most widely-known knowledge bases in natural language processing is WordNet [Princeton University, 2010]. WordNet groups words together based on their meanings. The authors define the concept of a *synset* - specific meaning of each word. A synset consists of a set of words that shared the same meaning, i.e. synonyms. It is these synsets that are linked together in the network. In addition to implicit synonymy, the relations between synsets include antonymy, meronymy, hyponymy, and hyperonymy.

While WordNet is a very extensive work in terms of the number of words it covers, the narrow set of relations, which are mostly oriented towards linguistics, is a limiting factor. ConceptNet [Speer et al., 2017] is a multilingual knowledge graph that represents world knowledge. It consists of a set of approximately

[2]https://www.dbpedia.org/

[3]https://www.wikipedia.org/

[4]https://www.wikidata.org/wiki/Wikidata:Main$_{P}age$

relation	example
RelatedTo	(language, RelatedTo, English)
IsA	(French, IsA, language)
HasPrerequisite	(talking, HasPrerequisite, language)
UsedFor	(language, UsedFor, communication)
HasProperty	(language, HasProperty, spoken or written)

Table 2.1: Examples of relations in ConceptNet.

8 million nodes corresponding to *concepts*, and over 21 million directed edges representing the relations between them. Both nodes and edges may contain additional metadata, such as the original source of the information.

The set of relations extends WordNet's set of relations to 36 types. Some of them are shown in Table 2.1. While most relations are asymmetric, some, such as *RelatedTo*, are designed to be symmetric.

ConceptNet is available online.[5] Programmatically, it can be accessed via an online API, or through a locally-run copy.

2.3 Graph neural networks

Some deep-learning approaches assume data that is structured in a certain way. For example, convolutional neural networks (CNNs) are often used in practice for data structured in a grid, while RNNs work with sequences, i.e., data structured as a directed path. Graph neural networks (GNNs) can be thought of as a generalization of these and other techniques.

GNNs are networks capable of modeling data structured as general graphs. Social groups, computer programs, or biological molecules are examples of graph-like entities whose structure is vital in defining their characteristics. GNNs are an integral part of some of the most influential advancements in recent years, such as the AlphaFold model for protein structure prediction [Jumper et al., 2021], a task which had previously found very little success.

In this section, we introduce GNNs and their applications. First, we define a *graph* and related terminology. Then, we present the fundamental idea of GNNs, *message passing*. Finally, we describe the possibilities of modeling entities with GNNs.

2.3.1 Graph

A *graph* is the pair (V, E), where V is the set of nodes and E is the set of edges [Matoušek and Nešetřil, 2022]. If $E \subseteq \binom{V}{2}$, we say that the edges are *undirected* and the graph is *undirected*. If $E \subseteq V \times V$, then we say that the edges are *directed* and the graph is *directed*.

We can represent a graph in an *adjacency matrix*. The adjacency matrix $\mathbf{A}$ of a graph $G = (V, E)$ is a matrix of size $|V| \times |V|$ whose entries are ones and zeros. $\mathbf{A}[u, v] = 1$ if $(u, v) \in E$, or 0 otherwise. The adjacency matrix of a directed graph is symmetrical, that of an undirected graph does not need to be.

[5] https://conceptnet.io/

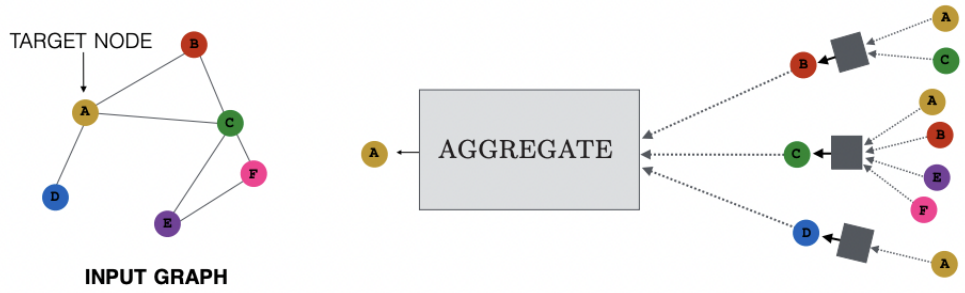

Figure 2.2: Overview of message passing [Hamilton, 2020, p. 49].

Since adjacency matrices are sparse, they are stored as lists of edges in computer memory.

2.3.2 GNNs and message passing

The fundamental idea of GNNs is *message passing*. This idea is implicitly present in CNNs and RNNs as well. As its name suggests, message passing ensures that information is propagated between nodes that are connected by a path.

According to Hamilton [2020], message passing iteratively updates each node's representation based on its neighboring nodes' representations. The following two steps are performed for every node during a single iteration of the algorithm:

1. Aggregate information from the node's neighboring nodes.
2. Update the node's representation by combining its current representation and the aggregated information.

Formally, the update to a node v at time step t can be expressed as

$$h_v^t = \text{Update}(h_v^{t-1}, \text{Aggregate}(\{h_u^{t-1}\ \forall u \in \mathcal{N}(v)\})) \tag{2.10}$$

where h_v^t denotes node v's representation at time t and $\mathcal{N}(v)$ denotes the neighborhood of node v. The exact details of the Update and Aggregate functions' implementation are not important for this work.

Figure 2.2 shows how information is propagated to node A in two-iteration message passing. The left side of the figure shows the structure of the graph. The right side shows the "unrolled" graph with the paths through which the message is passed. We see that some nodes' information reaches the target node multiple times.

2.3.3 GNN tasks

In practice, GNNs are used for three main purposes: node classification, graph classification, and edge prediction.

Node classification. Node classification involves assigning labels to nodes in a graph. The prediction is based on node features, as well as the graph's structure. Popular benchmarks include scientific paper classification [McCallum et al., 2000, Giles et al., 1998]. These benchmarks consist of a network of scientific papers, each of which is represented using language features (e.g., word vectors or TF-IDF vectors [Robertson, 2004]), where and edge connects two papers if one of them cites the other. The task is to predict the topic of the paper. Other applications include social media analysis (e.g., identifying bot accounts) or fraud detection in banking. Usually, such graphs only include a small number of positive examples.

Kipf and Welling [2017] proposed the graph convolutional network (GCN) model for node classification. The GCN model uses localized aggregations of node features that propagagate across the graph. GCN is a specific implementation of message passing, where the Aggregate function is a convolution.

Graph classification. Some tasks, such as molecule classification [Wang et al., 2023b], require the model to make a prediction for the entire graph. Graph classification is implemented by adding a special node with edges connecting it to all other nodes. This node represents the whole graph. Training and inference is then done in the same way as for node classification, except labels are only predicted for this node.

Relation prediction. The capabilities of GNNs can also be leveraged in tasks that require relation prediction, such as recommender systems [Ying et al., 2018] or knowledge graph completion [Schlichtkrull et al., 2017]. Relation prediction - predicting whether an edge is present in the graph or not - is how we use GNNs in this thesis (see Section 3.3). During training, the model optimizes a pairwise node embedding loss function, such as the inner product $z_i^\top z_j$ for the node embeddings z_i and z_j for which we want to predict the presence of an edge [Hamilton, 2020].

3. Experiments

The goal of this thesis is to explore improvements to generative commonsense reasoning in pre-trained language models. We investigate multiple approaches, with our main focus being using external databases as knowledge sources. We motivate our approach by recalling that multiple papers in Section 1.2 showed that adding information on input, including concept-set augmentation, leads to considerable performance improvement. Although some works (notably, Yu et al. [2022]) claim that knowledge graphs do not contain sufficient knowledge to lead to significant performance gain, we nevertheless believe that exploring the approach is worthwhile. While we do not expect to surpass the current state of the art, we expect that our experiments will lead to improvements over a baseline. Our results will show which directions are worth exploring further.

In this chapter, we describe the experiments we performed. We divide the experiments into several groups based on the nature of the augmentations added to the models. First, in Section 3.1, we describe a baseline model - a simple fine-tuning of a language model - to which we compare the other experiments. In Section 3.2 and Section 3.3, we describe experiments with textual additions to the input. These two methods are based on enriching the model with information from an external source, namely ConceptNet [Speer et al., 2017], using several different methods. While the former is based on extracting relevant subgraphs from ConceptNet, the latter processes ConceptNet with a GNN.

In Section 3.4, we describe a post-editing approach. The method described is partially inspired by the approach proposed by Liu et al. [2022]; specifically, its refining stage, where an initial generation is passed through another model to fix potential errors.

Finally, we propose a modification of the model's learning objective in Section 3.5. This approach is influenced by the technique of multi-task learning, where two tasks being learned at the same time can potentially improve the model's performance on both.

At the end of the chapter, we describe the process of evaluating the experiments in Section 3.6.

3.1 Baseline

We compare all our experiments to a baseline model's performance. To obtain our baseline, we fine-tune a Hugging Face implementation of BART-base [Lewis et al., 2020] with a language-modeling head[1] on the CommonGen dataset. During fine-tuning, all parameters are set to trainable. On input, we use concatenated input concepts separated by a space. The target output is the reference sentence. As the CommonGen dataset contains multiple reference sentences for each concept set, we train the model on multiple instances of the same input with different target outputs.

[1] `https://huggingface.co/docs/transformers/model_doc/bart#transformers.BartForConditionalGeneration`

3.2 Enhancing inputs using simple graph algorithms

We propose to use a knowledge graph as a source of external knowledge. ConceptNet [Speer et al., 2017] is a human-created knowledge graph that contains over 21 million commonsense relations of different types connecting over 8 million concepts, as described in Section 2.2. We are of the opinion that even though the knowledge covered by ConceptNet does not include every possible piece of commonsense information, the source is rich enough to lead to considerable improvements over the baseline if used well.

In this section, we explore approaches to extracting useful information from ConceptNet by using simple queries and standard graph algorithms (as opposed to using GNNs, which is described in Section 3.3).

In ConceptNet, the individual concepts, represented as nodes, are connected to other concepts by different types of semantically meaningful relations. Instead of injecting the graph structure directly into the language model, we transform it so that it resembles natural language. For each relation, we define a template which is then filled in with the specific concepts. We work with 17 templates in total.

As an example, consider the relation type `MadeOf`. The template for the relation (`A, MadeOf, B`) is

`A` *is made of* `B`.

Therefore, the relation (`window, MadeOf, glass`) will be transformed to the sentence

Window is made of glass.,

even though it is not grammatically correct.

We train several models that differ in what specific information is obtained from ConceptNet. The models are trained on input-output pairs corresponding to data points in the CommonGen dataset. For each input concept set, we query ConceptNet for information related to the concepts. The input to the language model then consists of space-separated concepts, followed by a special `[SEP]` token, followed by one or more sentences created as described above from the extracted information. The model is trained to generate a reference sentence for the specific concept set.

Below, we describe the different input enhancements, i.e., approaches to extracting information from ConceptNet and feeding it to the model as additional input.

Basic input enhancement. In this input enhancement, we select one of the input concepts at random. We further randomly choose one edge leading from this concept in the ConceptNet graph. This enhancement can be thought of as another baseline since it allows us to compare enhancements that add more information and see whether the additional information provides any value.

All-concept input enhancement. Here, we select one random relation in ConceptNet for *each* input concept. Since we do not restrict the relation selection, this enhancement will lead to a lot of irrelevant information on the input. We investigate this approach mostly in order to compare it to the *fully-connected* enhancement. The latter adds a similar amount of additional information, but the information is selected purposefully; therefore, we expect it to yield better results.

Pairwise input enhancement. We try to find an edge in ConceptNet that contains two of the input concepts. If no such edge exists, the input is not enriched at all. If multiple such edges exist, we pick one of them at random. We expect this enhancement to outperform the *basic* enhancement as the additional knowledge should be relevant to the context.

Fully connected input enhancement. Here, we attempt to connect all nodes to each other through the shortest path that exists between them in ConceptNet and feed all information on the path to the model.[2] This is the maximum size of added information investigated in this thesis. We expect that amount of extra knowledge steers the language model to generate contextually-appropriate sentences that also align with common sense. Algorithm 1 shows how we obtain the relations for a specific concept set. Each extracted relation is added to the input using templates as described above.

Algorithm 1 Fully connected information extraction

```
edges ← ∅
for each unordered pair c1, c2 in concept set do
    path ← shortest path from c1 and c2
    edges ← edges ∪ all edges in path
end for
return edges
```

Spanning-tree-like input enhancement. While the *fully-connected* enhancement enriches the model with a large amount of information, it only connects the concepts to each other. Perhaps, a language model can perform better if we add information about the new nodes in the *fully-connected* enhancement. We thus experiment with taking the spanning tree induced by vertices of the graph in the *fully-connected* enhancement. We describe how we extract the information in Algorithm 2.

Table 3.1 shows what the input to a language model looks like for the concept set *drive, snow, car* when using different enhancements.

[2]We call this approach fully-connected despite the fact we actually do not use the whole subgraph induced by vertices on the shortest paths.

Algorithm 2 Spanning-tree-like information extraction

vertices $\leftarrow \emptyset$
for each unordered pair c_1, c_2 in concept set **do**
  path $\leftarrow$ shortest path from c_1 and c_2
  vertices $\leftarrow$ *vertices* $\cup$ all vertices on *path*
end for
$G \leftarrow$ minimum spanning tree of the subgraph induced by *vertices*
return all edges in G

enhancement	**input to the LM**
baseline	drive snow car
basic	drive snow car `[CLS]` If you want to drive then you should go to a golf course
all-concept	drive snow car `[CLS]` drive is a type of return `[SEP]` The effect of snow is shovelling. `[SEP]` A volvo is a type of car
pairwise	drive snow car `[CLS]` car would make you want to drive
fully-c.	drive snow car `[CLS]` drive is related to person person does not desire rain rain is the opposite of snow `[SEP]` drive is related to car `[SEP]` snow is used for children `[SEP]` children can be found at car
spanning-tree-like	drive snow car `[CLS]` snow can be found at roof `[SEP]` roof is a part of car `[SEP]` children can be found at car `[SEP]` car causes drive `[SEP]` Something that might happen while ride is climb `[SEP]` ocean can be found at water `[SEP]` land can be found at ocean `[SEP]` Something that might happen while drive is ride `[SEP]` water can be found at snow

Table 3.1: Different enhancements on the concept set *drive, snow, car*.

3.3 Information extraction using GNNs

Approaches in Section 3.2 exploit the concepts' close neighborhood in ConceptNet. While these methods can reveal substantial information about the relationship between two concepts, a lot of knowledge about the nodes is embedded further away in the graph. It might thus be advantageous to use techniques that can aggregate information from the graph beyond simple paths between nodes.

We propose using graph neural networks (GNNs, Section 2.3) to capture more complex relationships between nodes. Our approach is inspired by the notion of recommender systems in social media where GNNs are employed to recommend entities to other entities of the same type, such as suggesting accounts in social media [Wu et al., 2022]. Here, we learn to "recommend" a concept to another concept to be included in the same sentence.

We devise a multi-step method that involves pre-training a GNN "recommender" model and using its results to fine-tune a BART model for sentence generation. The steps are the following:

1. First, we collect all pairs of words that appear together in at least one reference sentence in the training set of CommonGen. We only consider verbs, nouns, and adjectives.

2. We define a new type of undirected edge in ConceptNet representing the co-occurrence of two concepts in the same sentence. We connect all pairs of nodes in ConceptNet corresponding to the words identified in the first step by this edge type.

3. We train a GNN model to predict the existence of a co-occurrence edge between two arbitrary nodes. We sample negative examples randomly, which is a standard approach in recommender systems [Wu et al., 2022].

4. For each concept that appears in CommonGen, we use the GNN model to predict other concepts that might co-occur in the same sentence. We make predictions for all words at a distance of at most 2 steps from the origin node in ConceptNet[3] since it would be too computationally expensive to make predictions for the entire ConceptNet graph.

5. We fine-tune BART to generate reference sentences from an input that consists of the input concept set and the concepts predicted in Step 4. The predicted concepts are appended to the input as individual words since it is not clear what the relationship between the original concepts and their predicted co-occurring concepts is and, therefore, we cannot use templates as in Section 3.2.

3.4 Post-editing

The post-editing method has two components: a base model that produces an initial generation, and an error-correcting model that refines the initial generation. Both models are trained separately and are only combined at the inference stage. This approach builds upon previously-discussed methods. Its main objective is to correct omission errors.

The training data for the error-correcting model is synthesized from the reference sentences from the training set of CommonGen. We obtain a dependency tree of the sentence and remove a subtree that contains exactly one input concept. A subtree can only be deleted if its root is a non-core or nominal dependent.[4] This way, the resulting sentence remains semantically and grammatically correct. If there are multiple candidates for removal, we include all the resulting sentences in the training data. If no subtree meets the conditions, we do not add any sentence to the training data.

As an example, consider the concepts

ride, shirt, wear, bike, helmet

and a reference sentence

A boy puts on his helmet and rides his bike to the store to find a new shirt to wear.

The corresponding dependency tree is shown in Figure 3.1.

We can remove the subtree corresponding to the phrase *to wear*, whose reoot is *wear*, since it contains exactly one concept (*wear*) and the root is a nominal dependent (adnominal clause). We cannot remove the subtree corresponding to the phrase *his helmet*, whose root is *helmet*, because the root is a core argument (object), nor can we remove the subtree corresponding to the phrase *to find a new shirt to wear*, whose root is *find* since it contains multiple concepts (*shirt, wear*), even though the root is a non-core dependent. The only candidate sentence is thus

[3]We determined the distance limit empirically.

[4]We use the list of non-core and nominal dependents at `https://universaldependencies.org/u/dep/index.html`.

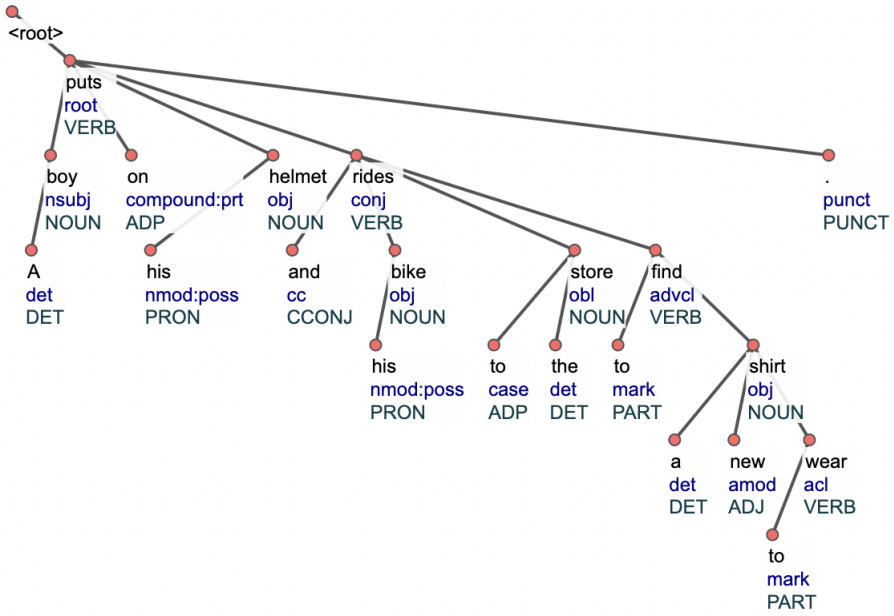

Figure 3.1: A dependency tree of an example sentence. Generated by UDPipe [Straka, 2018].

A boy puts on his helmet and rides his bike to the store to find a new shirt.

In the training data, we store the concept set, the modified sentence, and the reference sentence which will, again, be the target for a language model. Since some initial generations by the base model may already be sufficient, we also include data points where both the input and the target are unmodified reference sentences. The ratio of modified to unmodified inputs is approximately 2:1.

Once we have the training data, we use it to fine-tune a BART model. It gets the concatenated concepts and the potentially synthetic sentence as input and is trained to generate the original sentence.

During inference, we first generate a sentence using either the baseline model from Section 3.1, or one of the two best-performing models from Sections 3.2 and 3.3. Then, we pass the output and the concept set to the fine-tuned error-correcting model which produces the final sentence. We expect that the error-correcting model learns to recognize when a concept is missing and add it to a suitable position.

3.5 Composite loss

The final method we explore is training a model to perform two tasks at once. The first task is to generate a commonsense sentence from a set of concepts same as the baseline from Section 3.1. The second task is learning to classify sentences depending on whether they contain all input concepts.

The training data for this model is synthesized in the same way as the training data for the error-correcting model from Section 3.4; however, the synthesized sentences are used as targets, not inputs. In addition to that, we include a classification target – a boolean feature that determines whether the sentence

includes all concepts or not. The value is determined by matching the lemmas in the sentence to the concept set lemmas.

Specifically, the model gets two inputs (the concept set and the potentially modified sentence) and learns to produce two outputs (the original sentence and the completeness). Its architecture consists of two heads (one for generation, one for classification) built on top of a base BART model. The sum of the losses from both heads is propagated through the entire model. The generation loss is set to 0 for sentences marked as incomplete, i.e., the model is not trained to generate incomplete sentences.

Our objective is to investigate whether the inclusion of an additional classification task leads to performance gains on the generation task.

3.6 Evaluation

In this section, we explain the evaluation process for our experiments. We test our models on the validation set of CommonGen since the test set is not available publicly. It is worth mentioning that the validation set differs from the test set, especially in the overlap of pairs of concept sets present in the train set and other characteristics shown in Table 1.1.

First, we evaluate our experiments using automatic metrics. The metrics we use are described in Section 1.1. We report BLEU [Papineni et al., 2002], ROUGE [Lin, 2004], METEOR [Banerjee and Lavie, 2005], CIDEr Vedantam et al. [2015], and SPICE [Anderson et al., 2016] for the entire validation set. We use the testing suite provided by the CommonGen authors.[5] Unlike other works, we also split the validation set by concept set size and report BLEU, CIDEr, and SPICE separately for each subset. This enables us to see whether a method yields significantly better or worse results for a specific input complexity.

Additionally, we measure the *coverage* of the concepts by the model, i.e., the proportion of input concepts that appear in the generated sentence. We allow the input concept in any word form. We determine whether a concept is present by lemmatizing the generated output.

Since it is possible that random network initialization affects the model, we repeat each experiment 5 times. The reported metrics are the arithmetic average of the 5 measurements. Additionally, we perform a two-sample t-test to determine whether the measured results are significantly different from the baseline.

Second, we perform a manual performance analysis. After obtaining the baseline model's results, we identify its key weaknesses. We select specific inputs and compare the quality of the other models' generations on these inputs. Our main focus is commonsense correctness. Additionally, we also compare examples on which the baseline performed well to ensure that our approaches do not introduce new errors.

[5] The code and instructions for the CommonGen test suite are available at `https://github.com/INK-USC/CommonGen/tree/master/evaluation/Traditional`.

4. Results

This chapter presents the results of our experiments with setups described in Chapter 3. The purpose of this chapter is to determine whether the approaches we designed improve upon a basic fine-tuning approach and if so, by how much.

Section 4.1 shows the results of an automatic evaluation with all commonly used metrics for this task. We report detailed results broken down by input length in order to better identify the specific strengths and weaknesses of the respective approaches. In Section 4.2, we select specific input examples and observe how each approach affects the outputs. We focus on examples that are particularly challenging for the basic fine-tuning approach.

Overall, we find that graph information extraction described in Section 3.2, specifically, the *all-concept* and *fully-connected* enhancements, yield the best results. Moreover, when combined with post-editing (Section 3.4), they produce even better outputs as measured by the automatic metrics and confirmed by manual evaluation.

4.1 Automatic evaluation

In this section, we summarize the results of the automatic evaluation of the experiments. We report the BLEU [Papineni et al., 2002], CIDEr [Vedantam et al., 2015], SPICE [Anderson et al., 2016], ROUGE [Lin, 2004], and METEOR [Banerjee and Lavie, 2005] scores; all these metrics are described in Section 1.1. BLEU, CIDEr, and SPICE are also computed separately for each input concept set length. Additionally, we report the concept *coverage* (the percentage of concepts included in the outputs).

As explained in Section 3.6, we repeated each experiment five times. In addition to the overall mean scores from all runs, we also report the t-test value for each set of experiments by comparing its results to the baseline (on the set of 5 results). The t-test assumptions (independent, normally distributed random variables) are fulfilled in our case. We set the significance level to 0.05.

Baseline

model	B	C	S	R	M	c
baseline	26.88	14.09	28.82	35.42	38.84	87.66

Table 4.1: Automatic evaluation of the baseline model on the entire validation dataset. B = BLEU, C = CIDEr, S = SPICE, R = ROUGE, M = METEOR, c = coverage.

model	(3)B	(3)C	(3)S	(4)B	(4)C	(4)S	(5)B	(5)C	(5)S
baseline	25.38	14.34	28.72	30.6	15.11	29.64	25.64	13.48	28.14

Table 4.2: Automatic evaluation of the baseline model split by concept set size. B = BLEU, C = CIDEr, S = SPICE, R = ROUGE, M = METEOR, c = coverage. Metric initial preceded by a number indicates that it was computed on concept sets of that size.

Tables 4.1 and 4.2 show the metrics computed for the outputs of the **baseline** model (Section 3.1). One thing to notice is that the model performs best according to all metrics on concept sets of size 4. Another interesting result is the relatively low concept coverage of 87.66% which could certainly be improved.

Enhancing input using simple graph algorithms

model	B	C	S	R	M	c
baseline	26.88	14.09	28.82	35.42	38.84	87.66
basic	27.1	14.42*	29.1	35.52	39.28	88.84
all-concept	28.64†	**15.36**†	30.6†	36.48†	40.82†	**92.62**†
pairwise	26.76	14.03	28.88	35.48	39.16	88.66
fully-c.	**28.92**†	15.35†	**30.68**†	**36.68**†	**40.88**†	92.26†
spanning-tree	28.5*	15.26†	30.26*	36.44†	40.62*	91.44*

Table 4.3: Automatic evaluation of models with simple graph algorithm enhancements on the entire validation set compared to the baseline. B = BLEU, C = CIDEr, S = SPICE, R = ROUGE, M = METEOR, c = coverage. A star (*) marks those results that are significant ($p < 0.05$). A cross (†) marks results with $p < 0.001$. Bold type is used for the best-performing model according to the respective metrics; underline for the second best.

model	(3)B	(3)C	(3)S	(4)B	(4)C	(4)S	(5)B	(5)C	(5)S
baseline	25.38	14.34	28.72	30.6	15.11	29.64	25.64	13.48	28.14
basic	25.72	14.79*	29.12	30.08	15.21	29.84	26.3	13.76*	28.34
all-concept	27.58*	15.67†	**30.92**†	32.3*	16.36†	31.34*	**26.94***	**14.54***	29.24*
pairwise	25.62	14.4	28.94	29.78	14.94	29.46	25.58	13.33	28.16
fully-c.	27.78†	15.62†	30.84†	**32.96***	**16.56**†	**31.52**†	26.86*	14.42†	**29.56***
spanning-tree	**27.8***	**15.7**†	30.62*	31.46	16.1*	30.38	**26.94***	14.37*	29.3*

Table 4.4: Automatic evaluation of the model with the composite loss split by concept set size compared to the baseline. See Table 4.3 for metric and symbol explanations.

Tables 4.3 and 4.4 show the results of the experiments described in Section 3.2. The models with the *all-concept* and *fully-connected* enhancements perform best, followed by the model with the *spanning-tree-like* enhancement. These three achieve higher scores in every metric than any other model. Based on the t-test p-values, this improvement is statistically significant. The only exception is the *spanning-tree-like* enhancement, which does not have a significant effect on concept sets of size 4. That being said, neither input enhancement leads to a better performance than the state-of-the-art systems (cf. Section 1.2).

The three enhancements share the property that they add a substantial amount of information to the input. We hypothesized that the *fully-connected* enhancement would perform better than the *all-concept* one due to the information being relevant for all the concepts. However, this does not seem to be the case. While the *fully-connected* enhancement was better than the *all-concept* one in more cases than vice versa, the scores for each metric are similar. On the other hand, the *spanning-tree-like* enhancement, whose amount of added information is the largest among the three, was worse than both of them on average. It seems that the input concepts' immediate neighbors in ConceptNet help the language model generate better sentences, even though they are, in theory, less relevant for the context than more distant nodes.

On the other hand, the *basic* and *pairwise* enhancements do not lead to significant improvements. While they generally achieve higher scores than the baseline, they do not do so by a large margin. Note that the *basic* enhancement performs slightly better than the *pairwise* enhancement in all metrics. Considering the fact that the *pairwise* enhancement either adds information relevant to at least two input concepts or it does not add any information at all, and information was added in about 85% cases, while the *basic* enhancement adds information that is possibly relevant to only one of the concepts, this is surprising. It suggests that *some* information, however irrelevant it may seem, drives the model to generate better sentences than possibly no information at all.

Enhancing input using GNNs

model	B	C	S	R	M	c
baseline	26.88	14.09	28.82	35.42	38.84	87.66
gnn	**27.54***	**14.66***	**29.58***	**35.92***	**39.58***	**89.58***

Table 4.5: Automatic evaluation of the model with the GNN enhancement on the entire validation set compared to the baseline. See Table 4.3 for metric and symbol explanations.

model	(3)B	(3)C	(3)S	(4)B	(4)C	(4)S	(5)B	(5)C	(5)S
baseline	25.38	14.34	28.72	30.6	15.11	29.64	25.64	13.48	28.14
gnn	**26.06**	**14.95***	**29.76***	**31.22**	**15.66***	**29.98**	**26.3***	**13.92***	**28.96***

Table 4.6: Automatic evaluation of the model with GNN enhancement split by concept set size compared to the baseline. See Table 4.3 for metric and symbol explanations.

Tables 4.5 and 4.6 show the scores achieved by the model with the *GNN* enhancement. We see that the enhancement had a significant positive effect except for the concept sets of size 4. While the improvement is not as substantial as for the *all-concept*, *fully-connected*, and *spanning-tree-like* enhancements, we can confidently say that the *GNN* enhancement manages to steer the language model to generate better outputs.

Post-editing

model	B	C	S	R	M	c
baseline	**26.88**	14.09	28.82	**35.42**	38.84	87.66
baseline + post	26.26	**14.41**	**29.26**	35.24	**39.72***	**92.66***
all-concept	**28.64**	15.36	30.6	**36.48**	40.82	92.62
all-concept + post	28.22	**15.56**	**30.94**	36.36	**41.48**	**96.3***
fully-c.	**28.92**	15.35	30.68	**36.68**	40.88	92.26
fully-c. + post	28.46	**15.56**	**31.08**	36.56	**41.56***	**96.08***

Table 4.7: Automatic evaluation of models with post-editing on the entire validation set compared to their non-post-edited counterparts.See Table 4.3 for metric and symbol explanations.

Shown in Tables 4.7 and 4.8 are the scores of the two-stage models with post-editing (Section 3.4). For comparison, we also show the scores of the respective

model	(3)B	(3)C	(3)S	(4)B	(4)C	(4)S	(5)B	(5)C	(5)S
baseline	**25.38**	**14.34**	28.72	**30.6**	15.11	29.64	**25.64**	13.48	28.14
baseline + post	24.86	14.29	**28.84**	29.62*	**15.38**	**30.12**	25.28	**14.59***	**29.2***
all-concept	**27.58**	**15.67**	30.92	**32.3**	16.36	31.34	**26.94**	14.54	29.24
all-concept + post	27.28	15.61	**30.94**	31.62	**16.55**	**31.78**	26.64	**15.31***	**30.14***
fully-c.	**27.78**	**15.62**	30.84	**32.96**	16.56	31.52	**26.86**	14.42	29.56
fully-c. + post	27.46	15.61	**30.96**	32.14	**16.69**	**31.92**	26.66	**15.2***	**30.38***

Table 4.8: Automatic evaluation of models with post-editing split by concept set size compared to their non-post-edited counterparts. See Table 4.3 for metric and symbol explanations.

model	edited predictions
baseline + post-editing	24.97%
all-concept + post-editing	16.8%
fully-c. + post-editing	17.24%

Table 4.9: Proportion of the base model's predictions edited by the error-correcting model.

models' outputs without post-editing. We see that most of the time, there was no statistically significant difference between the base model and its post-edited version. Notable exceptions are CIDEr and SPICE measured on concept sets of length 5 and *coverage*. Clearly, the post-editing managed to introduce missing concepts into the base predictions, which was our main objective here. While this did not have an effect on input concept sets of sizes 3 and 4, it led to a significant improvement for the longer inputs of length 5 in the most relevant metrics (CIDEr and SPICE). The interpretation of this improvement could be that it is easier for the error-correcting model to add the missing concepts to base predictions in a meaningful way when there is already enough context.

The error-correcting model did not affect all base predictions equally. As we can see in Table 4.9, predictions made by the models with the *all-concept* and *fully-connected* enhancements were considerably less likely to be changed. This may be due to the fact that their base predictions are already of higher quality than those made by the baseline, as evidenced by Tables 4.3 and 4.4.

Compound loss

model	B	C	S	R	M	c
baseline	26.88	14.09	28.82	35.42	38.84	87.66
composite loss	6.12	3.27	10	20.7	12.8	12.17

Table 4.10: Automatic evaluation of the model with the composite loss on the entire validation set compared to the baseline. See Table 4.3 for metric explanations.

model	(3)B	(3)C	(3)S	(4)B	(4)C	(4)S	(5)B	(5)C	(5)S
baseline	25.38	14.34	28.72	30.6	15.11	29.64	25.64	13.48	28.14
composite loss	6.76	3.95	10.78	6.38	3.39	9.96	4.82	1.15	8.5

Table 4.11: Automatic evaluation of the model with the composite loss split by concept set size compared to the baseline. See Table 4.3 for metric explanations.

As Tables 4.10 and 4.11 show, using a loss function composed of a classification and a generation component leads to severe performance degradation on the

concepts	model	outputs
field, look, stand	baseline	A woman stands in a field looking at the camera.
kid, room, dance	baseline	The kids are dancing in the living room.

Table 4.12: Examples of baseline model's satisfactory outputs for concept sets of size 3. For full outputs of all five runs, see Appendix A (Table A.1).

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage.
score, goal, team, player	baseline	soccer player scoring a goal for his team

Table 4.13: Examples of baseline model's satisfactory outputs for concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.2).

generation task. The model not only fails to generate outputs with proper semantic content, which is measured by SPICE, but it also produces ungrammatical sentences, as indicated by the BLEU scores.

We hypothesize that the additional loss makes the model overfit on the easier task (i.e., the classification component), leading to degraded performance on generation. Our experiments with different settings of the classification loss weight did not resolve the issue; we believe additional regularization may be necessary.

4.2 Manual evaluation

Although evaluating the experiments automatically gives us some insight into the models' performance, it only does so on a high level. Automatically computed metrics can tell us that a model performed better or worse overall but they do not say anything about the origin of the differences [van Miltenburg et al., 2021]. In order to understand how exactly a specific enhancement affects the model's behavior, we must analyze its output manually.

Another reason to analyze the results manually is that automatic metrics compare the outputs to a relatively small set of references. However, there are potentially infinitely many correct sentences for each concept set, many of which are quite distant from the references. In other words, automatic metrics do not necessarily correlate with human judgement [Novikova et al., 2017]. This can cause models to obtain low scores even if they generate good sentences. Manual analysis helps identify such instances.

In this section, we analyze the sentences generated by each model to see how they differ from the baseline model. Since it is infeasible to evaluate every test example, we select 9 specific informative *"difficult"* examples for the baseline model and compare the other models' outputs for them. To ensure that our approaches do not fail at examples that the baseline model was able to handle, we also select a sample of 6 *"easy"* examples where the baseline performed well.

Baseline

First, let us examine the outputs generated by the baseline models. In general, they produced good sentences in most cases, as evidenced by Tables 4.1 and 4.2.

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps down a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face. A woman uses a brush to apply foundation to her face.

Table 4.14: Examples of baseline model's satisfactory outputs for concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.3).

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch.
walk, street, leash	baseline	a woman walks along the street on a leash A woman walks down the street on a leash.
picture, camel, take	baseline	an image taken by a camel A camel takes a photo.

Table 4.15: Examples of baseline model's unsatisfactory outputs for concept sets of size 3. For full output listings of all five runs, see Appendix A (Table A.4).

Looking at the specific generations, Tables 4.12, 4.13 and 4.14 show examples where the baseline model generated sentences that are correct in both the commonsense and grammatical aspects. The models were able to correctly identify the relationships between the concepts: for example, in the generated sentences in Table 4.13, one *jumps* a *ramp* and *wears* a *shirt*, instead of the other way round.

However, even these good outputs exhibit a problem that is prevalent among all generated outputs. Namely, the models have a tendency to omit a concept, sometimes replacing it with a closely related word. An example of this error can be seen in Table 4.14: for the concept set *brush, makeup, face, use, apply*, one of the models failed to use the word *makeup* and generated the word *foundation* instead. Skipping concepts is mostly present in the outputs for the more complex concept sets of lengths 4 and 5.

The baseline models made specific types of errors depending on the length of the input concept set. For concept sets of size 3, they seemingly learned to treat the concepts as a triple of a verb, subject, and object (regardless of their order) and generate the simplest sentence structure containing these parts of speech. While this approach often works, some concept sets do not fit into the pattern.

Table 4.15 shows examples where such a sentence structure is inadequate. In the first concept set (*cat, pet, couch*), the models are either unable to reason that they need another word as the petter or the subject of the petting (*A cat is petting a couch*), or they fill in one of the words from input (*cat petting a cat on*

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair with toys hands sitting on a toy
spray, water, hose, dog	baseline	A dog sprinkles water on a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.

Table 4.16: Examples of baseline model's unsatisfactory outputs for concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.5).

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A man holding a knife with an orange peeling it.
use, gutter, stand, roof, clean	baseline	A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof.

Table 4.17: Examples of baseline model's unsatisfactory outputs for concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.6).

the couch). Similarly, for concept set *picture, camel, take*, some models generated a sentence where the camel is the picture taker, instead of being the subject of the picture. Given concept set *walk, street, leash*, all models correctly identified that one walks on the street and attempted to connect the word *leash* with a new word. However, they did not manage to construct the sentence so that it makes sense.

The outputs for concept sets of length 4 (Table A.5) are the easiest ones for the models according to Tables 4.1 and 4.2. However, there are still some errors made. Similarly to concept sets of length 3, the model seems to have learned to generate sentences of a very similar structure: a subject, a verb, and two objects. Since four concepts provide more opportunities for different combinations, the models most often produced commonsense sentences. However, in some instances, the models failed to make the right connection.

For the concept set *sit, toy, hand, chair*, the models generated sentences with one of the concepts (*hand*) as the subject, while a completely different word should be used instead. The output for the concept set *spray, water, hose, dog* suffers from the same problem of an inappropriate subject, suggesting that the models do not have the knowledge of the common trope of a dog being sprayed with a hose. In the last example (*cut, pumpkin, knife, top*), the models seem to have been confused by the word *top* which they all used in an incorrect context.

Finally, the outputs for the 5-concept inputs (Table 4.17) indicate that the models are often unable to connect all the words in a proper manner, leading to certain concepts being left out as in the outputs for the concept set *peel, hand, knife, orange, hold*, or generating a sentence with all of them in a way that does not make sense (*use, gutter, stand, roof, clean*).

The first example given (*game, paper, kid, scissor, rock*) is an example of missing world knowledge in the language model. While most people would probably immediately think of the game *rock, paper, scissors*, as evidenced by the human-generated references shown in Appendix A (Table A.6), the models did not generate any output containing this phrase.

Enhancing input using simple graph algorithms

Shown in Tables 4.18, 4.19 and 4.20 are the outputs of the models introduced in Section 3.2 for "easy" inputs on which the baseline performs well (cf. Tables 4.12, 4.13 and 4.14). The models occasionally omit concepts even in examples where the baseline did not, but these are isolated incidents that are likely due to random chance. In the vast majority of cases, all of the input-enhanced models produce a sentence very similar or identical to that of the baseline.

concepts	model	outputs
field, look, stand	baseline	A woman stands in a field looking at the camera.
	basic	A man stands in a field looking at the camera.
	all-concept	Two giraffes standing in a field looking towards the camera.
	pairwise	A man is standing in a field looking at the camera.
	fully-c.	A man stands in a field looking at the camera.
	spanning-tree	A man stands in a field looking at the camera
kid, room, dance	baseline	The kids are dancing in the living room.
	basic	The kid is dancing in the living room.
	all-concept	A kid is dancing in a room.
	pairwise	The kids are dancing in the living room.
	fully-c.	A kid is dancing in a room.
	spanning-tree	The kids are dancing in the living room.

Table 4.18: Examples of baseline model's satisfactory outputs for concept sets of size 3. For full output listings of all five runs, see Appendix A (Table A.1).

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage.
	basic	A crowd of people are dancing in front of a stage.
	all-concept	A crowd dancing in front of a stage.
	pairwise	A crowd of people are dancing in front of a stage.
	fully-c.	The crowd is dancing in front of the stage.
	spanning-tree	A man is dancing in front of a crowd on stage.
score, goal, team, player	baseline	soccer player scoring a goal for his team
	basic	players score a goal against sports team
	all-concept	football player celebrates scoring a goal against sports team
	pairwise	soccer player scoring a goal for his team
	fully-c.	football player celebrates after scoring a goal against sports team
	spanning-tree	football player score a goal for football team

Table 4.19: Examples of graph-enhanced models' outputs for "easy" concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.8).

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps down a ramp.
	basic	A skateboarder wearing a red shirt jumps off a ramp.
	all-concept	A skateboarder wearing a red shirt is jumping down a ramp.
	pairwise	A skateboarder wearing a red shirt jumps off a ramp.
	fully-c.	A skateboarder in a red shirt is doing a jump on a ramp.
	spanning-tree	A skateboarder wearing a red shirt jumps off a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face.
	basic	use a brush to apply makeup to your face
	all-concept	use a brush to apply makeup to your face
	pairwise	use a brush to apply makeup to your face.
	fully-c.	use a brush to apply makeup to the face
	spanning-tree	A woman uses a brush to apply makeup to her face.

Table 4.20: Examples of graph-enhanced models' outputs for "easy" concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.9).

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch.
	basic	A cat is petting a couch. A cat petting a cat on a couch.
	all-concept	A cat is petting a cat on a couch. A cat is petting a couch.
	pairwise	A cat is petting a couch. cat petting a cat on a couch
	fully-c.	A cat is petting a cat. A man petting a cat on a couch.
	spanning-tree	A cat petting a cat on a couch. A man petting a cat on a couch.
walk, street, leash	baseline	a woman walks along the street on a leash A woman walks down the street on a leash.
	basic	a woman walks down the street on a leash Two people walking on a leash on a street.
	all-concept	a dog walks on a leash on a street A man walks down a street with a leash.
	pairwise	a dog walks on a leash on a street A woman walks on a leash in the street.
	fully-c.	A dog walks down the street on a leash. A dog walks on a leash on a street.
	spanning-tree	a dog walks on a leash in the street A dog walking on a leash in a street.
picture, camel, take	baseline	an image taken by a camel A camel takes a photo.
	basic	A man takes a picture of a camel. A woman takes a picture of a camel.
	all-concept	A man is taking a picture of a camel. A camel taking a picture.
	pairwise	A camel takes a photo. a man taking a photo of a camel
	fully-c.	c camel taking a picture A man takes a picture of a camel.
	spanning-tree	c camel taking a picture A man takes a picture of a camel.

Table 4.21: Examples of graph-enhanced models' outputs for "difficult" concept sets of size 3. For full output listings of all five runs, see Appendix A (Table A.10).

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair with toys hands sitting on a toy
	basic	hands sitting on a chair with toys hands sitting on a chair with a toy
	all-concept	hands sitting on a chair with toys hands sitting on a chair with a toy
	pairwise	hands sitting on a chair with toys hands sitting on a chair with a toy
	fully-c.	hands sitting on a chair with toys A man sits on a chair with a toy in his hand.
	spanning-tree	hands sit on a chair and play with toys A hand sits on a chair next to a toy.
spray, water, hose, dog	baseline	A dog sprinkles water on a hose.
	basic	a dog sprinkles water on a hose A dog isspraying water on a hose.
	all-concept	A dog is spraying water on a hose. A dog is being sprayed with water from a hose.
	pairwise	A dog sprinkles water on a hose.
	fully-c.	A dog is spraying water on a hose. A dog is being sprayed with water. A dog uses a hose to spray water on the ground.
	spanning-tree	A dog is using a hose to spray water. A dog is being spray with water from a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.
	basic	A man cutting a pumpkin on top of a knife. A man cutting a pumpkin with a knife on top.
	all-concept	A man cutting a pumpkin with a knife on top.
	pairwise	A man cutting a pumpkin with a knife on top.
	fully-c.	A man cutting pumpkins with a knife on top.
	spanning-tree	A man cutting a pumpkin with a knife on top.

Table 4.22: Examples of graph-enhanced models' outputs for "difficult" concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.11).

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors.
	basic	A kid is playing a game with scissors on a rock. A kid is playing a game of scissors with paper.
	all-concept	A kid is playing a video game with scissors and paper on a rock. A group of kids playing a game with scissors and paper.
	pairwise	A kid is playing a game with scissors and paper. A group of kids playing a game with scissors and paper.
	fully-c.	A kid is playing a game of scissors on paper with rocks. A kid is playing a game with scissors on paper.
	spanning-tree	A kid is playing a game of scissors on paper. A group of kids play a game with paper and scissors.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A man holding a knife with an orange peeling it.
	basic	hands holding a knife with oranges A man holding a knife with an orange on it.
	all-concept	A man holding a knife with oranges and oranges on it. A hand holds a peel of an orange with a knife.
	pairwise	A hand holding a knife with oranges in it. A hand holds a large knife with an orange on it.
	fully-c.	A man holding a knife with oranges in his hand. A man holding a knife with an orange peel on it.
	spanning-tree	A man holding a knife with oranges and bananas in it. A hand is holding an orange peeled with a knife.
use, gutter, stand, roof, clean	baseline	A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof.
	basic	A man uses a gutter to clean a roof. A man is standing in a gutter to clean the roof.
	all-concept	A man uses a gutter to clean a roof. A man uses a gutter to clean his roof while standing.
	pairwise	use a gutter to clean the roof A man uses a gutter to clean a man standing on a roof.
	fully-c.	A man standing next to a gutter that is being used to clean the roof. A man is standing on a roof and cleaning a gutter.
	spanning-tree	A man uses a gutter to clean the roof. A man is standing in a gutter to use to clean the roof.

Table 4.23: Examples of graph-enhanced models' outputs for "difficult" concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.12).

concepts	model	outputs
field, look, stand	baseline	A woman stands in a field looking at the camera.
	gnn	A man standing in a field looking at the camera.
kid, room, dance	baseline	The kids are dancing in the living room.
	gnn	The kids are dancing in the living room.

Table 4.24: Examples of GNN-enhanced model's outputs for "easy" concept sets of size 3. For full output listings of all five runs, see Appendix A (Table A.13).

The models' outputs for the "difficult" examples are shown in Tables 4.21, 4.22 and 4.23. The weaker models based on metrics in Section 4.1 (the *basic* and *pairwise* enhancements) often generated similar outputs to the baseline, including the same errors. Interestingly, the model with the *basic* enhancement generated a valid commonsense sentence for the concept set *picture, camel, take* in all five cases (see Table A.10). However, the slightly higher scores for the *basic* and *pairwise* models compared to the baseline probably originate from similar one-off successes; it does not seem that the improvements are systematic.

Conversely, there is a clear trend of improvements in the models with the *spanning-tree-like*, *all-concept*, and especially *fully-connected* enhancements. The *fully-connected* enhancement managed to drive the model to generate commonsense sentences for all selected 3-concept concept sets (Table 4.21), two concept sets of size 4 (*sit, toy, hand, chair* and *spray, water, hose, dog*, Table 4.22) and two concept sets of 5 concepts (*game, paper, kid, scissor, rock* and *use, gutter, stand, roof, clean*, Table 4.23), although these improvements were not consistent across all instances within a set of experiments. It was the only enhancement that led to a commonsense output for a particularly challenging input, producing the sentence *A man sits on a chair with a toy in his hand.* for the input *sit, toy, hand, chair*.

The *all-concept* and *spanning-tree-like* enhancements showed some of the same improvements, although to a lesser extent. From the analyzed sample, we can conclude that these three enhancements do, in fact, add previously absent information to the language model and lead to observable improvements.

However, some examples remain hard for all of these models. Specifically, no model described in Section 3.2 managed to generate a commonsense sentence for the concept set *cut, pumpkin, knife, top*, and very few did so for *peel, hand, knife, orange, hold* and *use, gutter, stand, roof, clean*. Additionally, while the models produced commonsense outputs for the concept set *game, paper, kid, scissor, rock*, none of the generated sentences concerns the well-known game of *rock, paper, scissors*, indicating that our enhancements are not sufficient to induce the necessary world knowledge in all cases.

Enhancing input using GNNs

Tables 4.24, 4.25 and 4.26 show the GNN-enhanced model's outputs for the examples where the baseline generated satisfactory outputs. We see that the GNN-enhanced model's outputs are almost identical to the baseline's. While the GNN-enhanced model does not improve further, neither does it introduce new errors.

Tables 4.27, 4.28 and 4.29 show the GNN-enhanced model's generations for the examples where the baseline often made errors. As expected from the au-

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage.
	gnn	The crowd is dancing in front of the stage.
score, goal, team, player	basline	soccer player scoring a goal for his team
	gnn	football player celebrates after scoring a goal against sports team

Table 4.25: Examples of GNN-enhanced model's outputs for "easy" concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.14).

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps down a ramp.
	gnn	skateboarder wearing a red shirt jumps off a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face.
	gnn	A woman uses a brush to apply makeup to her face.

Table 4.26: Examples of GNN-enhanced model's outputs for "easy" concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.15).

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch.
	gnn	cat petting a cat on the couch A cat is petting a couch.
walk, street, leash	baseline	a woman walks along the street on a leash A woman walks down the street on a leash.
	gnn	a dog walks on a leash on a street A woman walks down the street on a leash.
picture, camel, take	baseline	cat petting a cat on the couch A cat is petting a couch.
	gnn	an image of a camel taken A man takes apicture of a camel.

Table 4.27: Examples of the GNN-enhanced model's outputs for "difficult" concept sets of size 3. For full output listings of all five runs, see Appendix A (Table A.16).

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair with toys hands sitting on a toy
	gnn	hands sitting on a chair with toys A man sits on a chair with a toy in his hands.
spray, water, hose, dog	baseline	A dog sprinkles water on a hose.
	gnn	A dog isspraying water on a hose. A dog sprinkles water on a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.
	gnn	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife.

Table 4.28: Examples of the GNN-enhanced model's outputs for "difficult" concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.17).

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors.
	gnn	A kid is playing a game with scissors and paper on a rock. A kid is playing a game of scissors on paper.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A man holding a knife with an orange peeling it.
	gnn	hands holding a knife with oranges and peaches A woman holds a knife with her hand, and peels an orange.
use, gutter, stand, roof, clean	baseline	A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof.
	gnn	A man uses a gutter to clean the roof. A man is standing in a gutter to clean the roof.

Table 4.29: Examples of the GNN-enhanced model's outputs for "difficult" concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.18).

tomatic metrics in Tables 4.5 and 4.6, the GNN enhancement did not manage to correct all errors. Similar to the baseline, the model tends to follow a specific sentence structure and match the concepts to specific constituents. The resulting sentences, while generally grammatically correct, are wrong from the commonsense aspect.

However, in some cases, the GNN-enhanced model succeeded in generating commonsense sentences where the baseline did not, for example for the concept set *picture, camel, take* (Table 4.27), *sit, toy, hand, chair* (Table 4.28), or *peel, hand, knife, orange, hold* (Table 4.29). That being said, it does not manage to do so consistently; while some runs of the experiment generate a satisfactory output, others produce erroneous sentences for the same input.

As far as we can tell, the GNN enhancement did not introduce new types of errors. The most common errors remain those seen in the baseline's generations: the model fails to capture the correct semantic relationship between two concepts and it often fails to produce sentences with all input concepts present.

Post-editing

Tables 4.30, 4.31 and 4.32 show the post-edited predictions on the "easy" examples. We see that the post-edited sentences are identical to their base counterparts. This shows that the error-correcting model does not make unnecessary changes to sentences that are already good enough. However, when a concept is omitted, the error-correcting model tries to add it to the sentence (see the *fully-connected* comparison for the concept set *field, look, stand* in Table 4.30). We found no evidence that the error-correcting model added any new systematic errors.

Tables 4.33, 4.34 and 4.35 show how the error-correcting model edited the base models' predictions. As the error-correcting model was trained to add a missing concept into a sentence, it does not change sentences that already include all input concepts, even if they contain other errors (see *cat, pet, couch* in Table 4.33, *sit, toy, hand, chair* in Table 4.34).

On the other hand, if a sentence misses one of the concepts, the error-

concepts	model	outputs
field, look, stand	baseline	A man standing in a field looking out. A woman is standing in a field looking at the camera.
	baseline + post	A man standing in a field looking out. A woman is standing in a field looking at the camera.
	all-concept	Two giraffes standing in a field looking towards the camera. A man is standing in a field looking out.
	all-concept + post	Two giraffes standing in a field looking towards the camera. A man is standing in a field looking out.
	fully-c.	A man standing in a field looking towards the camera. A man is standing in a field.
	fully-c. + post	A man standing in a field looking towards the camera. A man is standing in a field looking at the camera.
kid, room, dance	baseline	The kids are dancing in the living room. A kid is dancing in a room.
	baseline + post	The kids are dancing in the living room. A kid is dancing in a room.
	all-concept	A kid is dancing in a room.
	all-concept + post	A kid is dancing in a room.
	fully-c.	The kids are dancing in the living room. A kid is dancing in a room.
	fully-c. + post	The kids are dancing in the living room. A kid is dancing in a room.

Table 4.30: Examples of the post-edited models' outputs for "easy" concept sets of size 3. For full output listings of all five runs, see Appendix A (Table A.19).

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage. A man dancing in front of a crowd.
	baseline + post	The crowd is dancing in front of the stage. A man dancing in front of a crowd on stage.
	all-concept	A group of people are dancing in front of a stage. A crowd dancing in front of a stage.
	all-concept + post	A group of people are dancing in front of a stage with a crowd of people. A crowd dancing in front of a stage.
	fully-c.	The crowd is dancing in front of the stage. A crowd of people dance in front of a stage.
	fully-c. + post	The crowd is dancing in front of the stage. A crowd of people dance in front of a stage.
score, goal, team, player	baseline	football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team
	baseline + post	football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team
	all-concept	football player score a goal against football team soccer player celebrates after scoring a goal against sports team
	all-concept + post	football player score a goal against football team soccer player celebrates after scoring a goal against sports team
	fully-c.	football player scoring a goal against football team football player celebrates after scoring a goal against football team
	fully-c. + post	football player scoring a goal against football team football player celebrates after scoring a goal against football team

Table 4.31: Examples of the post-edited models' outputs for "easy" concept sets of size 4. For full output listings of all five runs, see Appendix A (Table A.20).

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps down a ramp.
	baseline + post	A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps down a ramp.
	all-concept	A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	all-concept + post	A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	fully-c.	A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	fully-c. + post	A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face.
	baseline + post	A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face.
	all-concept	use a brush to apply makeup to your face A woman uses a brush to apply makeup to her face.
	all-concept + post	use a brush to apply makeup to your face A woman uses a brush to apply makeup to her face.
	fully-c.	use a brush to apply makeup to the face A woman uses a brush to apply makeup to her face.
	fully-c. + post	use a brush to apply makeup to the face A woman uses a brush to apply makeup to her face.

Table 4.32: Examples of the post-edited models' outputs for "easy" concept sets of size 5. For full output listings of all five runs, see Appendix A (Table A.21).

correcting model usually modifies it to include the missing concept. Sometimes, the resulting sentence is consistent with reality (*cadets taking a picture* changed to *cadets taking a picture of a camel* in Table 4.33), other times, the modification is relatively unimportant for the original sentence (*A woman holding a knife in her hand.* changed to *A woman holding a knife in her hand and apeel.* in Table 4.34). This, of course, highly depends on the quality of the base prediction. Sentences that are already good candidates but omit a concept have a higher chance of being rectified by the error-correcting model; this model, however, does not have the capacity to fix nonsense sentences.

Our error-correcting model is also limited by the fact that it has been trained to add one new concept only. Therefore, sentences that lack more than one concept (see *peel, hand, knife, orange* in Table 4.35) only get improved by one concept. This leads to a possible further direction of research: post-editing models that can correct more than one type of error.

It needs to be noted that there is a discrepancy in how we train the error-correcting model and how we use it. During training, it saw synthetic sentences that were missing a dependency subtree whose root was a non-core dependent. However, base predictions can be missing a concept that should play the role of the subject, a verb, or a direct object. Since the model was never trained on such sentences, it is unable to correct them well.

We also noticed the tendency of the error-correcting model to extend the end of the sentence instead of inserting text in the middle. This is consistent with the prevalent syntactic structures in the training data.

Overall, the post-editing approach has proved to work well if we have base predictions of sufficient quality available. Post-editing is not dependent on the base model, therefore, it can be used to refine sentences generated by any model.

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch.
	baseline + post	cat petting a cat on the couch A cat is petting a couch.
	all-concept	cat petting a cat on a couch A cat is petting a couch.
	all-concept + post	cat petting a cat on a couch A cat is petting a couch.
	fully-c.	A cat is petting a cat. A man petting a cat on a couch.
	fully-c. + post	A cat is petting a cat on a couch. A man petting a cat on a couch.
walk, street, leash	baseline	A dog walks down the street on its leash. a woman walks down the street on a leash
	baseline + post	A dog walks down the street on its leash. a woman walks down the street on a leash
	all-concept	a dog walks on a leash on a street Two dogs walking on a leash on a street.
	all-concept + post	a dog walks on a leash on a street Two dogs walking on a leash on a street.
	fully-c.	A dog walks down the street on a leash. A dog walks on a leash on a street.
	fully-c. + post	A dog walks down the street on a leash. A dog walks on a leash on a street.
picture, camel, take	baseline	A camel takes a photo. A group of people are taking notes on a camel.
	baseline + post	A camel takes a photo of a man looking at a camera. A group of people are taking notes on a camel in a photo.
	all-concept	c camel taking a picture A man is taking a picture of a camel.
	all-concept + post	c camel taking a picture A man is taking a picture of a camel.
	fully-c.	cadets taking a picture c camel taking a picture
	fully-c. + post	cadets taking a picture of a camel c camel taking a picture

Table 4.33: Outputs of the post-edited models on "difficult" examples of length 3. For a more detailed summary, see Tables A.22 to A.24.

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair hands sitting on a chair with a toy
	basline + post	hands sitting on a chair with toys hands sitting on a chair with a toy
	all-concept	hands sitting on a chair with toys hands sitting on a chair with a toy
	all-concept + post	hands sitting on a chair with toys hands sitting on a chair with a toy
	fully-c.	hands sitting on a chair with toys A man sits on a chair with a toy in his hand.
	fully-c. + post	hands sitting on a chair with toys A man sits on a chair with a toy in his hand.
spray, water, hose, dog	baseline	A dog sprinkles water on a hose.
	basline + post	A dog sprinkles water on a hose.
	all-concept	A dog is spraying water on a hose. A dog is being sprayed with water from a hose.
	all-concept + post	A dog is spraying water on a hose. A dog is being sprayed with water from a hose.
	fully-c.	A dog is sprayding water on a hose. A dog is being sprayed with water.
	fully-c. + post	A dog is sprayding water on a hose. A dog is being sprayed with water from a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top.
	basline + post	A man cutting a pumpkin with a knife on top.
	all-concept	A man cutting a pumpkin with a knife on top.
	all-concept + post	A man cutting a pumpkin with a knife on top.
	fully-c.	A man cutting pumpkins with a knife on top. A man cutting a pumpkin with a knife on top.
	fully-c. + post	A man cutting pumpkins with a knife on top. A man cutting a pumpkin with a knife on top.

Table 4.34: Outputs of the post-edited models on "difficult" examples of length 4. For a more detailed summary, see Tables A.25 to A.27.

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game with paper and scissors. A kid is playing a game of scissor with paper.
	baseline + post	A kid is playing a game of paper with scissors and scissors. A kid is playing a game of paper with scissors on a rock. A group of kids playing a game of rock with paper and scissors. A kid is playing a game of rock with paper and scissors. A kid is playing a game of scissor with paper on a rock.
	all-concept	A kid is playing a game of scissors with paper and scissors. A kid is playing a video game with scissors and paper on a rock.
	all-concept + post	A kid is playing a game of scissors with paper and scissors on a rock. A kid is playing a video game with scissors and paper on a rock.
	fully-c.	A kid is playing a game of scissors on paper with rocks. A kid is playing a game with scissors on paper.
	fully-c. + post	A kid is playing a game of scissors on paper with rocks. A kid is playing a game of rock with scissors on paper.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A man holding a knife with an orange peeling it.
	baseline + post	A woman holding a knife in her hand with an orange. A man holding a knife with an orange peeling it from his hand.
	all-concept	A man holding a knife with oranges and oranges on it. A man holding a knife with an orange peel.
	all-concept + post	A man holding a knife with oranges and oranges on it in his hand. A man holding a knife with an orange peel in his hand.
	fully-c.	A man holding a knife with oranges in his hand. A man holding a knife with an orange peel on it.
	fully-c. + post	A man holding a knife with oranges in his hand. A man holding a knife with an orange peel on it in his hand.
use, gutter, stand, roof, clean	baseline	A man standing behind a gutter to clean the roof. A man uses a gutter to clean the roof.
	baseline + post	A man standing behind a gutter to clean the roof. A man uses a gutter to clean the roof as he stands.
	all-concept	A man uses a gutter to clean a roof. A man uses a gutter to clean a man standing on a roof.
	all-concept + post	A man uses a gutter to clean a roof while standing next to it. A man uses a gutter to clean a man standing on a roof.
	fully-c.	A man standing next to a gutter that is being used to clean the roof. A man is standing on a roof and cleaning a gutter.
	fully-c. + post	A man standing next to a gutter that is being used to clean the roof. A man is standing on a roof and cleaning a gutter for use.

Table 4.35: Outputs of the post-edited models on "difficult" examples of length 5. For a more detailed summary, see Tables A.28 to A.30.

concepts	outputs
field, look, stand	A man looking on the look of a tree. A man is a a a city. A man looks on the field of the field
cat, pet, couch	a dog in the of the petals. A cat is the cat on the carpet. A cat sleeping on a couch.
dance, front, stage, crowd	a group of a in the garden dancing crowd in the crowd. A crowd of a a crowd of people.
spray, water, hose, dog	water flowing water in the of the water A dog is the water in the water. A dog feeding the water in the water
brush, makeup, face, use, apply	the image of a the face of the earth A man is a face. A man is the face of her face.
peel, hand, knife, orange, hold	A plate of a in the garden A man holding a a hand. A panning of with aelelelmer holds her hand.

Table 4.36: Examples of outputs generated by the model with a composite loss function (Section 3.5).

In fact, this is often the case in research related to generative commonsense reasoning, as we have shown in Section 1.2.

Composite loss

The approach performs very poorly. We show some examples in Table 4.36 to demonstrate that the model trained with a composite loss function is bad both at common sense, at fluency, and at using the concepts at all.

Conclusion

In this thesis, we explore different approaches aimed to improve generative commonsense reasoning. The primary method we investigate is enhancing the input with knowledge extracted from ConceptNet. We employ graph algorithms and GNNs to extract the knowledge. We find that the best-performing graph-based input enhancements are those that contain a lot of additional information on input. The GNN-enhanced model also performs relatively well, although it does not achieve the best results. The best-performing models manage to generate sentences that capture commonsense relations significantly better than a model without any input enhancements. Nevertheless, all of them still fail at the concept sets with the most complex relationships.

We also experiment with post-editing of base predictions by a model trained to incorporate missing concepts in a given sentence. While this model is independent of the way the base model works, in this work, we use it on some of the input-enhanced models we have trained previously. Our results show that this method increases coverage substantially, proving that the error-correcting model does what it has been trained to do. Nonetheless, coverage is not indicative of commonsense correctness, and its high value could be a result of forced unnatural additions. While some examples certainly exhibit this behavior, post-editing also improves CIDEr and SPICE which are more relevant for the task. We confirm the improvement by manual analysis. The effect is more pronounced for weaker base models, but it also shows on models that are already relatively good.

Finally, we investigate whether learning two tasks at once (i.e., language generation and classification) leads to a better performance at one of them (the generation). As is clear from our results, the method we used did not have the desired effect. We hypothesize that this is due to the fact that the secondary task is comparatively very easy in relation to the main one, leading the model to focus on it completely and neglect the other one.

In conclusion, we have explored a previously under-investigated family of approaches to commonsense reasoning. While we do not outperform the current state-of-the-art, our results show that using knowledge graphs for commonsense reasoning is a reasonable method showing promising results. This thesis presents foundation that explores which research directions are worth further pursuit.

Future work

We show that extracting complex subgraphs from ConceptNet leads to relatively good performance; however, so does using tangentially related information. It is up for discussion whether applying more complex methods from graph theory would yield better results; we hold the opinion that they would, but only modestly.

On the other hand, we believe that there is a lot of potential in using GNNs in novel ways to improve models' performance. In this thesis, we treat the task as a recommender system, "recommending" concepts to be included in the same sentence. It is possible to predict edges defined in a different way, or even use edges of various importance (such as "obligatory" words vs. "possible" concepts). Advanced techniques from recommender systems, such as better negative sampling,

could be employed as well.

While in our GNN approach, we focused on adding information extracted from ConceptNet on the input to a language model, GNNs provide opportunities to be used in conjunction with a language model as a single end-to-end system. For example, node embeddings learned by the GNN could be used instead of word embeddings in the language model. We believe that using GNNs on knowledge graphs is an area that deserves further attention.

Finally, while our method of multi-task learning produced a poor performance, this does not mean that we should reject the idea altogether. We hypothesize that training the model on two similarly complex tasks, instead of two with a very different level of complexity, could lead to an improvement in the generation task.

References

Jay Alammar. The Illustrated Transformer [Blog post], Jun 2018. URL http://jalammar.github.io/illustrated-transformer/.

Chenxin An, Jiangtao Feng, Kai Lv, Lingpeng Kong, Xipeng Qiu, and Xuanjing Huang. Cont: Contrastive neural text generation, 2023.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. SPICE: Semantic Propositional Image Caption Evaluation. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 382–398, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46454-1.

Atlassian. What is a knowledge base? URL https://www.atlassian.com/itsm/knowledge-management/what-is-a-knowledge-base.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate, 2016.

Satanjeev Banerjee and Alon Lavie. 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, June 2005. Association for Computational Linguistics. URL https://aclanthology.org/W05-0909.

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unilmv2: Pseudo-masked language models for unified language model pre-training. *CoRR*, abs/2002.12804, 2020. URL https://arxiv.org/abs/2002.12804.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1075. URL https://aclanthology.org/D15-1075.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.,

2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

Ernest Davis and Gary Marcus. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Commun. ACM*, 58(9):92–103, aug 2015. ISSN 0001-0782. doi: 10.1145/2701413. URL https://doi.org/10.1145/2701413.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 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, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Emily Dinan, Varvara Logacheva, Valentin Malykh, Alexander Miller, Kurt Shuster, Jack Urbanek, Douwe Kiela, Arthur Szlam, Iulian Serban, Ryan Lowe, Shrimai Prabhumoye, Alan W. Black, Alexander Rudnicky, Jason Williams, Joelle Pineau, Mikhail Burtsev, and Jason Weston. The second conversational intelligence challenge (convai2). In Sergio Escalera and Ralf Herbrich, editors, *The NeurIPS '18 Competition*, pages 187–208, Cham, 2020. Springer International Publishing. ISBN 978-3-030-29135-8.

William B. Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Proceedings of the Third International Workshop on Paraphrasing (IWP2005)*, 2005. URL https://aclanthology.org/I05-5002.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. Unified language model pre-training for natural language understanding and generation. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c20bb2d9a50d5ac1f713f8b34d9aac5a-Paper.pdf.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1346. URL https://aclanthology.org/P19-1346.

Zhihao Fan, Yeyun Gong, Zhongyu Wei, Siyuan Wang, Yameng Huang, Jian Jiao, Xuanjing Huang, Nan Duan, and Ruofei Zhang. An enhanced knowledge injection model for commonsense generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2014–2025, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics. doi: 10.18653/v1/2020.coling-main.182. URL https://aclanthology.org/2020.coling-main.182.

Steven Y. Feng, Jessica Huynh, Chaitanya Prasad Narisetty, Eduard Hovy, and Varun Gangal. SAPPHIRE: Approaches for enhanced concept-to-text generation. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 212–225, Aberdeen, Scotland, UK, August 2021a. Association for Computational Linguistics. URL https://aclanthology.org/2021.inlg-1.21.

Steven Y. Feng, Kevin Lu, Zhuofu Tao, Malihe Alikhani, Teruko Mitamura, Eduard H. Hovy, and Varun Gangal. Retrieve, caption, generate: Visual grounding for enhancing commonsense in text generation models. *CoRR*, abs/2109.03892, 2021b. URL https://arxiv.org/abs/2109.03892.

C. Lee Giles, Kurt D. Bollacker, and Steve Lawrence. Citeseer: An automatic citation indexing system. In *Proceedings of the Third ACM Conference on Digital Libraries*, DL '98, page 89–98, New York, NY, USA, 1998. Association for Computing Machinery. ISBN 0897919653. doi: 10.1145/276675.276685. URL https://doi.org/10.1145/276675.276685.

William L. Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14(3), 2020.

Xingwei He, Yeyun Gong, A-Long Jin, Weizhen Qi, Hang Zhang, Jian Jiao, Bartuer Zhou, Biao Cheng, Sm Yiu, and Nan Duan. Metric-guided distillation: Distilling knowledge from the metric to ranker and retriever for generative commonsense reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 839–852, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.53.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, Tom Hennigan, Eric Noland, Katie Millican, George van den Driessche, Bogdan Damoc, Aurelia Guy, Simon Osindero, Karen Simonyan, Erich Elsen, Jack W. Rae, Oriol Vinyals, and Laurent Sifre. Training compute-optimal large language models, 2022.

Interspeech. *Interspeech 2022*, 2022. doi: 10.21437/interspeech.2022.

Tianbo Ji, Yvette Graham, Gareth Jones, Chenyang Lyu, and Qun Liu. Achieving reliable human assessment of open-domain dialogue systems. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6416–6437, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.445. URL https://aclanthology.org/2022.acl-long.445.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, and et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021. doi: 10.1038/s41586-021-03819-2.

Dan Jurafsky and James H. Martin. *Speech and Language Processing*. 2023. 3rd ed. draft.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks, 2017.

Tom Kocmi, Rachel Bawden, OndÅ™ej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Thamme Gowda, Yvette Graham, Roman Grundkiewicz, Barry Haddow, Rebecca Knowles, Philipp Koehn, Christof Monz, Makoto Morishita, Masaaki Nagata, Toshiaki Nakazawa, Michal NovÃ¡k, Martin Popel, Maja PopoviÄ‡, and Mariya Shmatova. Findings of the 2022 conference on machine translation (wmt22). In *Proceedings of the Seventh Conference on Machine Translation*, pages 1–45, Abu Dhabi, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.wmt-1.1.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1082. URL https://aclanthology.org/D17-1082.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, S. Auer, and Christian Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6:167–195, 2015.

Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The winograd schema challenge. In *Proceedings of the Thirteenth International Conference on Principles of Knowledge Representation and Reasoning*, KR'12, page 552–561. AAAI Press, 2012. ISBN 9781577355601.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 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,

July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.703. URL https://aclanthology.org/2020.acl-main.703.

Haonan Li, Yeyun Gong, Jian Jiao, Ruofei Zhang, Timothy Baldwin, and Nan Duan. KFCNet: Knowledge filtering and contrastive learning for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2918–2928, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.249. URL https://aclanthology.org/2021.findings-emnlp.249.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. CommonGen: A Constrained Text Generation Challenge for Generative Commonsense Reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online, November 2020. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/2020.findings-emnlp.165.

Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1.

Xin Liu, Dayiheng Liu, Baosong Yang, Haibo Zhang, Junwei Ding, Wenqing Yao, Weihua Luo, Haiying Zhang, and Jinsong Su. KGR4: Retrieval, Retrospect, Refine and Rethink for Commonsense Generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11029–11037, 2022.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S. Yu. KG-BART: knowledge graph-augmented BART for generative commonsense reasoning. *CoRR*, abs/2009.12677, 2020. URL https://arxiv.org/abs/2009.12677.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. NeuroLogic a*esque decoding: Constrained text generation with lookahead heuristics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 780–799, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.57. URL https://aclanthology.org/2022.naacl-main.57.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015*

Conference on Empirical Methods in Natural Language Processing, pages 1412–1421, Lisbon, Portugal, September 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL https://aclanthology.org/D15-1166.

Jiří Matoušek and Jaroslav Nešetřil. *Kapitoly z diskrétní matematiky*. Univerzita Karlova, nakladatelství Karolinum, 2022.

Andrew Kachites McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3(2):127–163, 2000. doi: 10.1023/a:1009953814988.

Tomas Mikolov, Kai Chen, Greg S. Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space, 2013. URL http://arxiv.org/abs/1301.3781.

Faidon Mitzalis, Ozan Caglayan, Pranava Madhyastha, and Lucia Specia. BERT-Gen: Multi-task generation through BERT. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6440–6455, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.503. URL https://aclanthology.org/2021.acl-long.503.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206. URL https://aclanthology.org/D18-1206.

Jekaterina Novikova, Ondřej Dušek, Amanda Cercas Curry, and Verena Rieser. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1238. URL https://aclanthology.org/D17-1238.

OpenAI. Introducing chatgpt, 2022. URL https://openai.com/blog/chatgpt.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1efde53be364a73914f58805a001731-Paper-Conference.pdf.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA, July 2002. Association for Computational Linguistics. doi: 10.3115/1073083.1073135. URL https://aclanthology.org/P02-1040.

Princeton University. About WordNet, 2010. URL https://wordnet.princeton.edu/.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

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

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683, 2019. URL http://arxiv.org/abs/1910.10683.

Sebastian Raschka. Accelerating large language models with mixed-precision techniques [blog post], May 2023. URL https://sebastianraschka.com/blog/2023/llm-mixed-precision.html.

Stephen Robertson. Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation*, 60(5):503–520, January 2004. ISSN 0022-0418. doi: 10.1108/00220410410560582. URL https://doi.org/10.1108/00220410410560582. Publisher: Emerald Group Publishing Limited.

Anna Rohrbach, Marcus Rohrbach, Niket Tandon, and Bernt Schiele. A dataset for movie description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3202–3212, 2015.

Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A modern approach*. Prentice-Hall, 2010.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. Social IQa: Commonsense Reasoning about Social Interactions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4463–4473, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1454. URL https://aclanthology.org/D19-1454.

Michael Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks, 2017.

Jaehyung Seo, Dongsuk Oh, Sugyeong Eo, Chanjun Park, Kisu Yang, Hyeonseok Moon, Kinam Park, and Heuiseok Lim. Pu-gen: Enhancing generative commonsense reasoning for language models with human-centered knowledge. *Knowledge-Based Systems*, 256:109861, 2022. ISSN 0950-7051. doi: https://doi.org/10.1016/j.knosys.2022.109861. URL https://www.sciencedirect.com/science/article/pii/S0950705122009546.

Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14972.

Milan Straka. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/K18-2020. URL https://www.aclweb.org/anthology/K18-2020.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper_files/paper/2014/file/a14ac55a4f27472c5d894ec1c3c743d2-Paper.pdf.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A Question Answering Challenge Targeting Commonsense Knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models, 2023.

Emiel van Miltenburg, Miruna Clinciu, Ondřej Dušek, Dimitra Gkatzia, Stephanie Inglis, Leo Leppänen, Saad Mahamood, Emma Manning, Stephanie Schoch, Craig Thomson, and Luou Wen. Underreporting of errors in NLG output, and what to do about it. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 140–153, Aberdeen, Scotland, UK, August 2021. Association for Computational Linguistics. URL https://aclanthology.org/2021.inlg-1.14.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.,

2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-Based Image Description Evaluation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1580. URL https://aclanthology.org/P19-1580.

Han Wang, Yang Liu, Chenguang Zhu, Linjun Shou, Ming Gong, Yichong Xu, and Michael Zeng. Retrieval enhanced model for commonsense generation. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3056–3062, Online, August 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-acl.269. URL https://aclanthology.org/2021.findings-acl.269.

PeiFeng Wang, Jonathan Zamora, Junfeng Liu, Filip Ilievski, Muhao Chen, and Xiang Ren. Contextualized scene imagination for generative commonsense reasoning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Oh1r2wApbPv.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language models with self-generated instructions, 2023a.

Yuyang Wang, Zijie Li, and Amir Barati Farimani. Graph neural networks for molecules, 2023b.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models, 2022.

Shiwen Wu, Fei Sun, Wentao Zhang, Xu Xie, and Bin Cui. Graph neural networks in recommender systems: A survey. *ACM Comput. Surv.*, 55(5), dec 2022. ISSN 0360-0300. doi: 10.1145/3535101. URL https://doi.org/10.1145/3535101.

Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery &amp Data Mining*. ACM, jul 2018. doi: 10.1145/3219819.3219890. URL https://doi.org/10.1145%2F3219819.3219890.

Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for*

Computational Linguistics, 2:67–78, 2014. doi: 10.1162/tacl_a_00166. URL https://aclanthology.org/Q14-1006.

Wenhao Yu, Chenguang Zhu, Zhihan Zhang, Shuohang Wang, Zhuosheng Zhang, Yuwei Fang, and Meng Jiang. Retrieval augmentation for commonsense reasoning: A unified approach. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4364–4377, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics. URL https://aclanthology.org/2022.emnlp-main.294.

Sina Zarrieß, Henrik Voigt, and Simeon Schüz. Decoding Methods in Neural Language Generation: A Survey. *Information*, 12(9), 2021. ISSN 2078-2489. doi: 10.3390/info12090355. URL https://www.mdpi.com/2078-2489/12/9/355.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. HellaSwag: Can a Machine Really Finish Your Sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1472. URL https://aclanthology.org/P19-1472.

Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. A robustly optimized BERT pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1218–1227, Huhhot, China, August 2021. Chinese Information Processing Society of China. URL https://aclanthology.org/2021.ccl-1.108.

A. Selected outputs of the experiments

As each experiment was repeated 5 times, we show sentences generated by all individual models trained in the specific experiment. We report the outputs as generated by the model, i.e., with the original punctuation and capitalization.

concepts	model	outputs
field, look, stand	**references**	The player stood in the field looking at the batter. The coach stands along the field, looking at the goalkeeper. I stood and looked across the field, peacefully. Someone stands, looking around the empty field.
	baseline	A woman stands in a field looking at the camera. A man is standing in a field looking at the camera. A group of people stand in a field looking at the camera. A man standing in a field looking out. A woman is standing in a field looking at the camera.
kid, room, dance	**references**	The silly kid loves to dance in her room. the dance kid room is full of kids A kid is dancing in the room. A group of kids are dancing around a living room.
	baseline	The kids are dancing in the living room. The kids are dancing in a room. A kid is dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.

Table A.1: References and baseline models' good generations for concept sets of size 3.

concepts	model	outputs
dance, front, stage, crowd	**references**	A front stage crowd dance. The woman saw the crowd dancing in front of the stage. The girl danced on stage in front of a crowd. A person is dancing on a stage in front of a crowd.
	baseline	The crowd is dancing in front of the stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. A man dancing in front of a crowd. A man is dancing in front of a crowd on stage.
score, goal, team, player	**references**	The player scored a goal for his team. The soccer player scores a goal for his team. A player scores a goal for his team. football player scores a goal for football team
	baseline	soccer player scoring a goal for his team football player celebrates after scoring a goal for football team players score a goal against sports team football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team

Table A.2: Baseline models' good generations for concept sets of size 4.

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	**references**	The skateboarder jumped over the ramp and tore the shirt he wore that day. The young skateboarder was wearing a red shirt when he jumped on the ramp and won the competition. The skateboarder wearing a shirt jumped the ramp. a skateboarder wearing a green shirt jumping on a ramp
	baseline	A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a black shirt jumps a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps down a ramp.
brush, makeup, face, use, apply	**references**	The woman uses her brush to apply makeup to her face. Most people use a brush to apply makeup to their faces. A woman uses a brush to apply makeup on her face. A woman uses a brush to apply makeup to her face
	baseline	A woman uses a brush to apply makeup to her face. A woman uses a brush to apply foundation to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face.

Table A.3: Baseline models' good generations for concept sets of size 5.

concepts	model	outputs
cat, pet, couch	**references**	A pet cat likes to sleep on a couch. My pet cat love to sleep on the couch. The woman pet the cat that was sat on the couch. actor petting long haired gray cat on couch
	baseline	cat petting a cat on the couch A cat is petting a couch. A cat is petting a couch. cat petting a cat on the couch A cat is petting a couch.
walk, street, leash	**references**	The children walk with their dog on a leash down the street. I grabbed the leash to walk my dog across the street. a man walks two dogs on leashes down the street
	baseline	a woman walks along the street on a leash A woman walks down the street on a leash. A man walks down a street with a leash. A dog walks down the street on its leash. a woman walks down the street on a leash
picture, camel, take	**references**	The museum will take the picture of the camel to the basement. Would you please take a picture of me sitting on the camel. woman taking pictures of a group of camels
	baseline	an image taken by a camel A camel takes a photo. A group of people are taking notes on a camel. A man takes a photo of a camel. A man taking a photo of a camel.

Table A.4: Baseline models' bad generations for concept sets of size 3.

concepts	model	outputs
sit, toy, hand, chair	**references**	The woman's hand grabbed the toy as she was sitting down in the chair. A baby sits on a chair with a toy in one of its hands. Sit down in the chair and I will hand you the toy. Sit on the chair and hand the toy to me. A woman chose to sit in the chair while she held a toy in her hand. A child is sitting on a chair with a toy in his hands.
	baseline	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a toy hands sitting on a chair hands sitting on a chair with a toy
spray, water, hose, dog	**references**	The dog got sprayed with the water hose. I sprayed my dog with water from my garden hose to help clean him off. She uses a hose to spray water all over the dog.
	baseline	A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
cut, pumpkin, knife, top	**references**	He cuts the top of the pumpkin with a knife. During October, people carve pumpkins with a knife by cutting their top off and removing the insides. A woman used a knife to cut the top off the pumpkin. The chef cut off the top of the pumpkin with a knife to make a jack-o-lantern. A knife cuts the top of a pumpkin.
	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.

Table A.5: Baseline models' bad generations for concept sets of size 4.

concepts	model	outputs
game, paper, kid, scissor, rock	**references**	The kids solved disagreements by playing the game rock, paper, scissors. The kids are playing a game called rock, paper, scissors. The kids played the game rock, paper, scissors. Two kids play a friendly game of rock,paper,scissors, shoot with one another.
	baseline	A kid is playing a game of paper with scissors. A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors. A kid is playing a game with paper and scissors. A kid is playing a game of scissor with paper.
peel, hand, knife, orange, hold	**references**	Steadily holding the knife in hand, Sarah peeled the rind from the orange. The man holds a knife in his hand to peel the orange. The cook's hand shook while holding the knife to peel the orange. A woman is peeling an orange with a knife while holding in its hand
	baseline	A woman holding a knife in her hand. A hand holds a knife with an orange and apeel. A man holding a knife with an orange peeling it. A man holding a knife with an orange and an orange on it. A man holding a knife with an orange and a peel.
use, gutter, stand, roof, clean	**references**	The woman was hired to clean the gutter people use every day. A bird can stand on a roof and sing for a long time.. My dad said to use the ladder to stand on to clean the gutter on the left side of the roof. The woman uses a ladder to reach the roof where she can stand to clean the gutter. A man stands on a roof and uses an object to clean out a gutter.
	baseline	A man standing behind a gutter to clean the roof. A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof. A man uses a gutter to clean the roof. A man uses a gutter to clean the roof.

Table A.6: Baseline models' bad generations for concept sets of size 5.

concepts	model	outputs
field, look, stand	baseline	A woman stands in a field looking at the camera. A man is standing in a field looking at the camera. A group of people stand in a field looking at the camera. A man standing in a field looking out. A woman is standing in a field looking at the camera.
	basic	A man stands in a field looking at the camera. Two men standing in a field looking at a camera. A man is standing in a field looking at the camera. A man standing in a field looking at the camera. A woman is standing in a field looking at the camera.
	all-concept	The kid is dancing in the living room. The kids are dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	pairwise	A man is standing in a field looking at the camera. A woman standing in a field looking at the camera. A man standing in a field looking out. Two men stand in a field looking at each other. A woman stands in a field looking at the camera.
	fully-c.	A man stands in a field looking at the camera. A man is standing in a field looking at the camera. A man is standing in a field looking at the camera. A man standing in a field looking towards the camera. A man is standing in a field.
	spaning-tree	A man stands in a field looking at the camera. A man stands in a field and looks at the camera. A man is standing in a field and looking at the camera. A man standing in a field. A man stands in a field looking out.
kid, room, dance	baseline	The kids are dancing in the living room. The kids are dancing in a room. A kid is dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	basic	The kid is dancing in the living room. The kids are dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	all-concept	A kid is dancing in a room. A kid is dancing in a room. A group of kids dancing in a room. A kid is dancing in a room. A kid is dancing in a room.
	pairwise	The kids are dancing in the living room. The kids are dancing in the living room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	fully-c.	A kid is dancing in a room. A kid is dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	spaning-tree	The kids are dancing in the room. A kid is dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.

Table A.7: Outputs of models from Section 3.2 on "easy" examples of length 3.

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. A man dancing in front of a crowd. A man is dancing in front of a crowd on stage.
	basic	A crowd of people are dancing in front of a stage. A crowd of people dancing in front of a stage. A crowd of people dance in front of a stage. The crowd is dancing in front of the stage. A crowd of people dancing in front of a stage.
	all-concept	A group of people are dancing in front of a stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. The crowd is dancing in front of the stage. The crowd is dancing in front of the stage.
	pairwise	A crowd of people are dancing in front of a stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. A crowd of people dance in front of a stage. A crowd of people dancing in front of a stage.
	fully-c.	The crowd is dancing in front of the stage. The crowd is dancing in front of the stage. A crowd of people dance in front of a stage. A crowd of people dance in front of a stage. A crowd of people dancing in front of a stage.
	spaning-tree	A man is dancing in front of a crowd on stage. A crowd is dancing in front of a stage. A crowd of people dance in front of a stage. A crowd of people dance in front of a stage. A crowd of people are dancing in front of a stage.
score, goal, team, player	baseline	soccer player scoring a goal for his team football player celebrates after scoring a goal for football team players score a goal against sports team football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team
	basic	players score a goal against sports team football player score a goal against football team players score a goal against sports team football player scoring a goal for the team soccer player celebrates after scoring a goal against football team
	all-concept	football player celebrates scoring a goal against sports team soccer player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against sports team football player score a goal against football team soccer player celebrates after scoring a goal against sports team
	pairwise	soccer player scoring a goal for his team football player scoring a goal for football team football player scoring a goal against sports team football player scoring a goal against football team soccer player scoring a goal against football team
	fully-c.	football player celebrates after scoring a goal against sports team football player score a goal for football team football player celebrates after scoring a goal against sports team football player scoring a goal against football team football player celebrates after scoring a goal against football team
	spaning-tree	football team score a goal against football team football player score a goal for football team players score a goal against sports team football player score a goal against football team players score a goal against sports team

Table A.8: Outputs of models from Section 3.2 on "easy" examples of length 4.

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a black shirt jumps a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps down a ramp.
	basic	A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a red shirt jumps a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps up a ramp.
	all-concept	A skateboarder wearing a red shirt is jumping down a ramp. A skateboarder wearing a red shirt jumps a ramp. A skateboarder wearing a t-shirt jumps down a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	pairwise	A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a red shirt jumps a ramp. A skateboarder wearing a white shirt jumps off a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a t-shirt jumps down a ramp.
	fully-c.	A skateboarder in a red shirt is doing a jump on a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder is jumping a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	spaning-tree	A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a red shirt is jumping down a ramp. A skateboarder wearing a white shirt is jumping down a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a red shirt is jumping down a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face. A woman uses a brush to apply foundation to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face.
	basic	use a brush to apply makeup to your face A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	all-concept	use a brush to apply makeup to your face A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	pairwise	use a brush to apply makeup to your face. A woman uses a brush to apply foundation to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	fully-c.	use a brush to apply makeup to the face A woman uses a brush to apply makeup to her face. use a brush to apply makeup to the face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	spaning-tree	A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.

Table A.9: Outputs of models from Section 3.2 on "easy" examples of length 5.

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch. A cat is petting a couch. cat petting a cat on the couch A cat is petting a couch.
	basic	A cat is petting a couch. A cat petting a cat on a couch. A cat is petting a couch. A cat is petting a couch. A cat petting a couch.
	all-concept	cat petting a cat on the couch A cat is petting a cat on a couch. A cat is petting a couch. cat petting a cat on a couch A cat is petting a couch.
	pairwise	cat petting a couch A cat is petting a couch. A cat is petting a couch. cat petting a cat on a couch A cat petting a couch.
	fully-c.	A cat is petting a cat. A man petting a cat on a couch. A man petting a cat on a couch. The cat is petting the couch. A man petting a cat on a couch.
	spaning-tree	cat petting a couch A cat petting a cat on a couch. A man petting a cat on a couch. A cat is petting a couch. A cat is petting on a couch.
walk, street, leash	baseline	a woman walks along the street on a leash A woman walks down the street on a leash. A man walks down a street with a leash. A dog walks down the street on its leash. a woman walks down the street on a leash
	basic	a woman walks down the street on a leash Two people walking on a leash on a street. A woman walks down a street on a leash. a woman walks on a leash in the street a woman walking on a leash in the street
	all-concept	a dog walks on a leash on a street Two dogs walking on a leash on a street. A man walks down a street with a leash. A dog walking on a leash in a street. A woman is walking on a street with a leash.
	pairwise	a dog walks on a leash on a street A woman walks on a leash in the street. A man walks down a street with a leash. a dog walks on a leash on a street A woman walking down the street on a leash.
	fully-c.	A dog walks down the street on a leash. A dog walks on a leash on a street. A dog walking on a leash on a street. Two zebras walk down the street. A dog walking down a street with a leash.
	spaning-tree	a dog walks on a leash in the street A dog walking on a leash in a street. A dog walking on a leash on a street. The dog is walking down the street on a leash. A dog with a leash walking down a street.
picture, camel, take	baseline	an image taken by a camel A camel takes a photo. A group of people are taking notes on a camel. A man takes a photo of a camel. A man taking a photo of a camel.
	basic	A man takes a picture of a camel. A man takes a picture of a camel. A man takes a picture of a camel. A woman takes a picture of a camel. a man taking a picture of a camel
	all-concept	c camel taking a picture A man is taking a picture of a camel. A camel taking a picture. A woman takes a picture of a camel. a man takes a picture of a camel
	pairwise	image of a camel taking a position A camel takes a photo. A camel takes a photo. A camel taking a photo. a man taking a photo of a camel
	fully-c.	cadets taking a picture c camel taking a picture A man takes a picture of a camel. A woman taking a picture of a camel. a man taking a picture of a camel
	spaning-tree	c camel taking a picture A man takes a picture of a camel. A man takes a picture of a camel a man taking a picture of a camel a man takes a picture of a camel

Table A.10: Outputs of models from Section 3.2 on "difficult" examples of length 3.

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a toy hands sitting on a chair hands sitting on a chair with a toy
	basic	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a chair with a toy hands sitting on a chair with toys hands sitting on a chair with a toy
	all-concept	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a chair with a toy hands sitting on a chair with toys hands sitting on a chair with a toy
	pairwise	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a chair with toys hands sitting on a chair hands sitting on a chair with toys
	fully-c.	hands sitting on a chair with toys A man sits on a chair with a toy in his hand. A toy sitting in a chair next to a hand. hands sitting on a chair with toys hands sitting on a chair with a toy
	spaning-tree	hands sit on a chair and play with toys A hand sits on a chair next to a toy. hands sit on a toy hands sitting on a toy hands sitting on a chair with a toy
spray, water, hose, dog	baseline	A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
	basic	a dog sprinkles water on a hose A dog isspraying water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
	all-concept	A dog is spraying water on a hose. A dog is being sprayed with water from a hose. A dog sprinkles water on a hose. A dog is being spray with water from a hose. A dog is spraying water with a hose.
	pairwise	A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
	fully-c.	A dog is sprayding water on a hose. A dog is being sprayed with water. A dog sprinkles water on a hose. A dog uses a hose to spray water on the ground. A dog is spraying water on a hose.
	spaning-tree	A dog is using a hose to spray water. A dog is being spray with water from a hose. A dog is sprayding water with a hose. A dog uses a hose to spray water on a dog. A dog is spraying water on a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.
	basic	A man cutting a pumpkin on top of a knife. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.
	all-concept	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.
	pairwise	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.
	fully-c.	A man cutting pumpkins with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.
	spaning-tree	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on the top. A man cutting a pumpkin with a knife on top.

Table A.11: Outputs of models from Section 3.2 on "difficult" examples of length 4.

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game of paper with scissors. A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors. A kid is playing a game with paper and scissors. A kid is playing a game of scissor with paper.
	basic	A kid is playing a game with scissors on a rock. A kid is playing a game of scissors with paper. A group of kids play a game with scissors and paper. A kid is playing a game with scissors on paper. A kid playing a game of scissor with paper on a rock.
	all-concept	A kid is playing a game of scissors with paper and scissors. A kid is playing a video game with scissors and paper on a rock. A group of kids playing a game with scissors and paper. A kid is playing a video game on paper with scissors. A kid is playing a game of scissor with paper and rocks.
	pairwise	A kid is playing a game of scissor and paper. A kid is playing a game with scissors and paper. A group of kids playing a game with scissors and paper. A kid is playing a video game with scissors and paper. A kid is playing a game of scissor with paper.
	fully-c.	A kid is playing a game of scissors on paper with rocks. A kid is playing a game with scissors on paper. A kid plays a game with paper and scissors. The kid is playing a game of scissors on paper. A kid is playing a game of scrabble with paper and scissors on a rock
	spaning-tree	A kid is playing a game of scissors on paper. A group of kids play a game with paper and scissors.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A hand holds a knife with an orange and apeel. A man holding a knife with an orange peeling it. A man holding a knife with an orange and an orange on it. A man holding a knife with an orange and a peel.
	basic	hands holding a knife with oranges A man holding a knife with an orange on it. A man holding a knife with oranges in his hands. A man holding a knife with an orange on it in his hand. A man holding a knife with an orange on it.
	all-concept	A man holding a knife with oranges and oranges on it. A hand holds a peel of an orange with a knife. A man holding a knife with an orange peel. A man holding a knife with oranges and lemon peel on it. A hand holding a knife and an orange peel.
	pairwise	A hand holding a knife with oranges in it. A hand holds a large knife with an orange on it. hands holding a knife with an orange A man holding a knife with an orange on his hand. A man holding a knife with an orange in his hand.
	fully-c.	A man holding a knife with oranges in his hand. A man holding a knife with an orange peel on it. A hand holding a knife and an orange peel. A man holding a knife with an orange peel on it. A hand holding an orange with a knife.
	spaning-tree	A man holding a knife with oranges and bananas in it. A hand is holding an orange peeled with a knife. A man holding a knife with an orange peel in his hands. A man holding a knife with an orange peel on it. A man holding a knife with an orange peel on it.
use, gutter, stand, roof, clean	baseline	A man standing behind a gutter to clean the roof. A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof. A man uses a gutter to clean the roof. A man uses a gutter to clean the roof.
	basic	A man uses a gutter to clean the roof. A man uses a gutter to clean the roof. A man uses a gutter to clean a roof. A man uses a gutter to clean a roof. A man is standing in a gutter to clean the roof.
	all-concept	A man uses a gutter to clean a roof. A man uses a gutter to clean a man standing on a roof. A man uses a gutter to clean a roof. A man uses a gutter to clean his roof while standing. A man uses a gutter to clean the roof.
	pairwise	use a gutter to clean the roof A man uses a gutter to clean a man standing on a roof. A man standing on a roof uses a gutter to clean the roof. A woman uses a gutter to clean the roof. A man uses a gutter to clean the roof while standing on the roof.
	fully-c.	A man standing next to a gutter that is being used to clean the roof. A man uses a gutter to clean the roof. A man is standing on a roof and cleaning a gutter. A man uses a gutter to clean his roof. A man uses a gutter to clean the roof.
	spaning-tree	A man uses a gutter to clean a standing roof. A man uses a gutter to clean the roof. A man uses a gutter to clean a roof. A man standing on a roof using a gutter to clean it. A man is standing in a gutter to use to clean the roof.

Table A.12: Outputs of models from Section 3.2 on "difficult" examples of length 5.

concepts	model	outputs
field, look, stand	baseline	A woman stands in a field looking at the camera. A man is standing in a field looking at the camera. A group of people stand in a field looking at the camera. A man standing in a field looking out. A woman is standing in a field looking at the camera.
	gnn	man stands in a field looking at the camera A man standing in a field looking at the camera. A man is standing in a field of wheat. A man standing in a field looking out. A man is standing in a field looking out.
kid, room, dance	baseline	The kids are dancing in the living room. The kids are dancing in a room. A kid is dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	gnn	The kids are dancing in the living room. A kid is dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.

Table A.13: Outputs of models from Section 3.3 on "easy" examples of length 3.

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. A man dancing in front of a crowd. A man is dancing in front of a crowd on stage.
	gnn	The crowd is dancing in front of the stage. A crowd of people dancing in front of a stage. A man is dancing in front of a crowd of people. The crowd dance in front of the stage. A man is dancing in front of a crowd.
score, goal, team, player	baseline	soccer player scoring a goal for his team football player celebrates after scoring a goal for football team players score a goal against sports team football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team
	gnn	cricket player celebrates after scoring a goal against sports team football player score a goal for football team football player celebrates after scoring a goal against sports team football player scoring a goal against football team cricket player scoring a goal against sports team

Table A.14: Outputs of models from Section 3.3 on "easy" examples of length 4.

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a black shirt jumps a ramp. A skateboarder wearing a red shirt jumps up a ramp.
	gnn	A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a t-shirt jumps a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a red shirt is jumping down a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face. A woman uses a brush to apply foundation to her face. A man uses a brush to apply makeup to his face.
	gnn	use a brush to apply makeup to the face A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.

Table A.15: Outputs of models from Section 3.3 on "easy" examples of length 5.

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch. A cat is petting a couch. cat petting a cat on the couch A cat is petting a couch.
	gnn	A cat is petting a couch. A cat is petting a cat on a couch. A cat is petting a couch. cat petting a cat on a couch A cat is petting a couch.
walk, street, leash	baseline	a woman walks along the street on a leash A woman walks down the street on a leash. A man walks down a street with a leash. A dog walks down the street on its leash. a woman walks down the street on a leash
	gnn	a dog walks on a leash on a street Two people walking on a leash in the street. A man walking on a leash on a street. A woman walks down a street with a leash. A woman is walking on a leash on a street.
picture, camel, take	baseline	an image taken by a camel A camel takes a photo. A group of people are taking notes on a camel. A man takes a photo of a camel. A man taking a photo of a camel.
	gnn	an image of a camel taken A man takes apicture of a camel. A man is taking apicture of a camel. A man takes apicture of a camel. a man takes apicture of a camel

Table A.16: Outputs of models from Section 3.3 on "difficult" examples of length 3.

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a toy hands sitting on a chair hands sitting on a chair with a toy
	gnn	hands sitting on a chair with toys hands sitting on a chair with a toy A man sits on a chair with a toy in his hands. hands sitting on a chair A hand sits on a chair with a toy.
spray, water, hose, dog	baseline	A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
	gnn	A dog isspraying water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.
	gnn	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife. A man cutting a pumpkin with a knife on top.

Table A.17: Outputs of models from Section 3.3 on "difficult" examples of length 4.

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game of paper with scissors. A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors. A kid is playing a game with paper and scissors. A kid is playing a game of scissor with paper.
	gnn	A kid is playing a game with scissors and paper on a rock. A kid is playing a game with scissors and paper. A group of kids playing a game with scissors and paper. A kid is playing a game of scissors on paper. A kid is playing a game of scissor with paper.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A hand holds a knife with an orange and apeel. A man holding a knife with an orange peeling it. A man holding a knife with an orange and an orange on it. A man holding a knife with an orange and a peel.
	gnn	hands holding a knife with oranges and peaches A woman holds a knife with her hand, and peels an orange. A man holding a knife with a knife in his hands. hands holding an orange and a peeling knife a hand holding a knife with an orange
use, gutter, stand, roof, clean	baseline	A man standing behind a gutter to clean the roof. A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof. A man uses a gutter to clean the roof. A man uses a gutter to clean the roof.
	gnn	A man uses a gutter to clean the roof. A man uses a gutter to clean a man standing on a roof. A man uses a gutter to clean the roof. A man uses a gutter to clean his roof. A man is standing in a gutter to clean the roof.

Table A.18: Outputs of models from Section 3.3 on "difficult" examples of length 5.

concepts	model	outputs
field, look, stand	baseline	A woman stands in a field looking at the camera. A man is standing in a field looking at the camera. A group of people stand in a field looking at the camera. A man standing in a field looking out. A woman is standing in a field looking at the camera.
	baseline + post	A man is standing in a field looking at the camera. A woman stands in a field looking at the camera. A group of people stand in a field looking at the camera. A man standing in a field looking out. A woman is standing in a field looking at the camera.
	all-concept	A group of people stand in a field. A man stands in a field and looks at the camera. A group of people standing in a field looking at each other. Two giraffes standing in a field looking towards the camera. A man is standing in a field looking out.
	all-concept + post	A group of people stand in a field looking at camera. A man stands in a field and looks at the camera. A group of people standing in a field looking at each other. Two giraffes standing in a field looking towards the camera. A man is standing in a field looking out.
	fully-c.	A man stands in a field looking at the camera. A man is standing in a field looking at the camera. A man is standing in a field looking at the camera. A man standing in a field looking towards the camera. A man is standing in a field.
	fully-c. + post	A man stands in a field looking at the camera. A man is standing in a field looking at the camera. A man is standing in a field looking at the camera. A man standing in a field looking towards the camera. A man is standing in a field looking at the camera.
kid, room, dance	baseline	The kids are dancing in the living room. The kids are dancing in a room. A kid is dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	baseline + post	The kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	all-concept	A kid is dancing in a room. A kid is dancing in a room. A group of kids dancing in a room. A kid is dancing in a room. A kid is dancing in a room.
	all-concept + post	A kid is dancing in a room. A kid is dancing in a room. A group of kids dancing in a room. A kid is dancing in a room. A kid is dancing in a room.
	fully-c.	A kid is dancing in a room. A kid is dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.
	fully-c. + post	A kid is dancing in a room. A kid is dancing in a room. A group of kids are dancing in a room. The kids are dancing in the living room. A kid is dancing in a room.

Table A.19: Outputs of the models from Section 3.4 on "easy" examples of length 3.

concepts	model	outputs
dance, front, stage, crowd	baseline	The crowd is dancing in front of the stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. A man dancing in front of a crowd. A man is dancing in front of a crowd on stage.
	baseline + post	A crowd dancing in front of a stage. The crowd is dancing in front of the stage. A crowd of people dance in front of a stage. A man dancing in front of a crowd on stage. A man is dancing in front of a crowd on stage.
	all-concept	A group of people are dancing in front of a stage. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. The crowd is dancing in front of the stage. The crowd is dancing in front of the stage.
	all-concept + post	A group of people are dancing in front of a stage with a crowd of people. A crowd dancing in front of a stage. A crowd of people dance in front of a stage. The crowd is dancing in front of the stage. The crowd is dancing in front of the stage.
	fully-c.	The crowd is dancing in front of the stage. The crowd is dancing in front of the stage. A crowd of people dance in front of a stage. A crowd of people dance in front of a stage. A crowd of people dancing in front of a stage.
	fully-c. + post	The crowd is dancing in front of the stage. The crowd is dancing in front of the stage. A crowd of people dance in front of a stage. A crowd of people dance in front of a stage. A crowd of people dancing in front of a stage.
score, goal, team, player	baseline	soccer player scoring a goal for his team football player celebrates after scoring a goal for football team players score a goal against sports team football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team
	baseline + post	football player celebrates after scoring a goal for football team soccer player scoring a goal for his team players score a goal against sports team football player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against football team
	all-concept	football player celebrates scoring a goal against sports team soccer player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against sports team football player score a goal against football team soccer player celebrates after scoring a goal against sports team
	all-concept + post	football player celebrates scoring a goal against sports team soccer player celebrates after scoring a goal for his team soccer player celebrates after scoring a goal against sports team football player score a goal against football team soccer player celebrates after scoring a goal against sports team
	fully-c.	football player celebrates after scoring a goal against sports team football player score a goal for football team football player celebrates after scoring a goal against sports team football player scoring a goal against football team football player celebrates after scoring a goal against football team
	fully-c. + post	football player celebrates after scoring a goal against sports team football player score a goal for football team football player celebrates after scoring a goal against sports team football player scoring a goal against football team football player celebrates after scoring a goal against football team

Table A.20: Outputs of the models from Section 3.4 on "easy" examples of length 4.

concepts	model	outputs
skateboarder, jump, ramp, wear, shirt	baseline	A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a black shirt jumps a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps down a ramp.
	baseline + post	A skateboarder wearing a red shirt jumps off a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a black shirt jumps a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a red shirt jumps down a ramp.
	all-concept	A skateboarder wearing a red shirt is jumping down a ramp. A skateboarder wearing a red shirt jumps a ramp. A skateboarder wearing a t-shirt jumps down a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	all-concept + post	A skateboarder wearing a red shirt is jumping down a ramp. A skateboarder wearing a red shirt jumps a ramp. A skateboarder wearing a t-shirt jumps down a ramp. A skateboarder wearing a red shirt jumps up a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	fully-c.	A skateboarder in a red shirt is doing a jump on a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder is jumping a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
	fully-c. + post	A skateboarder in a red shirt is doing a jump on a ramp. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder is jumping a ramp in a white shirt. A skateboarder wearing a red shirt jumps down a ramp. A skateboarder wearing a t-shirt jumping down a ramp.
brush, makeup, face, use, apply	baseline	A woman uses a brush to apply makeup to her face. A woman uses a brush to apply foundation to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face.
	baseline + post	A woman uses a brush to apply foundation to her face. A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face. A woman uses a brush to apply makeup to her face. A man uses a brush to apply makeup to his face.
	all-concept	use a brush to apply makeup to your face A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	all-concept + post	use a brush to apply makeup to your face A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	fully-c.	use a brush to apply makeup to the face A woman uses a brush to apply makeup to her face. use a brush to apply makeup to the face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.
	fully-c. + post	use a brush to apply makeup to the face A woman uses a brush to apply makeup to her face. use a brush to apply makeup to the face. A woman uses a brush to apply makeup to her face. A woman uses a brush to apply makeup to her face.

Table A.21: Outputs of the models from Section 3.4 on "easy" examples of length 5.

concepts	model	outputs
cat, pet, couch	baseline	cat petting a cat on the couch A cat is petting a couch. A cat is petting a couch. cat petting a cat on the couch A cat is petting a couch.
	baseline + post	A cat is petting a couch. cat petting a cat on the couch A cat is petting a couch. cat petting a cat on the couch A cat is petting a couch.
walk, street, leash	baseline	a woman walks along the street on a leash A woman walks down the street on a leash. A man walks down a street with a leash. A dog walks down the street on its leash. a woman walks down the street on a leash
	baseline + post	A woman walks down the street on a leash. a woman walks along the street on a leash A man walks down a street with a leash. A dog walks down the street on its leash. a woman walks down the street on a leash
picture, camel, take	baseline	an image taken by a camel A camel takes a photo. A group of people are taking notes on a camel. A man takes a photo of a camel. A man taking a photo of a camel.
	baseline + post	A camel takes a photo of a man looking at a camera. an image taken by a camel A group of people are taking notes on a camel in a photo. A man takes a photo of a camel. A man taking a photo of a camel.

Table A.22: Outputs of a baseline model and post-edits on "difficult" examples of length 3.

concepts	model	outputs
cat, pet, couch	all-concept	cat petting a cat on the couch A cat is petting a cat on a couch. A cat is petting a couch. cat petting a cat on a couch A cat is petting a couch.
	all-concept + post	cat petting a cat on the couch A cat is petting a cat on a couch. A cat is petting a couch. cat petting a cat on a couch A cat is petting a couch.
walk, street, leash	all-concept	a dog walks on a leash on a street Two dogs walking on a leash on a street. A man walks down a street with a leash. A dog walking on a leash in a street. A woman is walking on a street with a leash.
	all-concept + post	a dog walks on a leash on a street Two dogs walking on a leash on a street. A man walks down a street with a leash. A dog walking on a leash in a street. A woman is walking on a street with a leash.
picture, camel, take	all-concept	c camel taking a picture A man is taking a picture of a camel. A camel taking a picture. A woman takes a picture of a camel. a man takes a picture of a camel
	all-concept + post	c camel taking a picture A man is taking a picture of a camel. A camel taking a picture. A woman takes a picture of a camel. a man takes a picture of a camel

Table A.23: Outputs of a *all-concept*-enhanced model and post-edits on "difficult" examples of length 3.

concepts	model	outputs
cat, pet, couch	fully-c.	A cat is petting a cat. A man petting a cat on a couch. A man petting a cat on a couch. The cat is petting the couch. A man petting a cat on a couch.
	fully-c. + post	A cat is petting a cat on a couch. A man petting a cat on a couch. A man petting a cat on a couch. The cat is petting the couch. A man petting a cat on a couch.
walk, street, leash	fully-c.	A dog walks down the street on a leash. A dog walks on a leash on a street. A dog walking on a leash on a street. Two zebras walk down the street. A dog walking down a street with a leash.
	fully-c. + post	A dog walks down the street on a leash. A dog walks on a leash on a street. A dog walking on a leash on a street. Two zebras walk down the street on leash. A dog walking down a street with a leash.
picture, camel, take	fully-c.	cadets taking a picture c camel taking a picture A man takes a picture of a camel. A woman taking a picture of a camel. a man taking a picture of a camel
	fully-c. + post	cadets taking a picture of a camel c camel taking a picture A man takes a picture of a camel. A woman taking a picture of a camel. a man taking a picture of a camel

Table A.24: Outputs of a *fully-connected*-enhanced model and post-edits on "difficult" examples of length 3.

concepts	model	outputs
sit, toy, hand, chair	baseline	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a toy hands sitting on a chair hands sitting on a chair with a toy
	basline + post	hands sitting on a chair with a toy hands sitting on a chair with toys hands sitting on a toy in a chair hands sitting on a chair with toys hands sitting on a chair with a toy
spray, water, hose, dog	baseline	A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
	basline + post	A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose. A dog sprinkles water on a hose.
cut, pumpkin, knife, top	baseline	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.
	basline + post	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin on top of a knife.

Table A.25: Outputs of a baseline model and post-edits on "difficult" examples of length 4.

concepts	model	outputs
sit, toy, hand, chair	all-concept	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a chair with a toy hands sitting on a chair with toys hands sitting on a chair with a toy
	all-concept + post	hands sitting on a chair with toys hands sitting on a chair with a toy hands sitting on a chair with a toy hands sitting on a chair with toys hands sitting on a chair with a toy
spray, water, hose, dog	all-concept	A dog is spraying water on a hose. A dog is being sprayed with water from a hose. A dog sprinkles water on a hose. A dog is being spray with water from a hose. A dog is spraying water with a hose.
	all-concept + post	A dog is spraying water on a hose. A dog is being sprayed with water from a hose. A dog sprinkles water on a hose. A dog is being spray with water from a hose. A dog is spraying water with a hose.
cut, pumpkin, knife, top	all-concept	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.
	all-concept + post	A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.

Table A.26: Outputs of a *all-concept*-enhanced model and post-edits on "difficult" examples of length 4.

concepts	model	outputs
sit, toy, hand, chair	fully-c.	hands sitting on a chair with toys A man sits on a chair with a toy in his hand. A toy sitting in a chair next to a hand. hands sitting on a chair with toys hands sitting on a chair with a toy
	fully-c. + post	hands sitting on a chair with toys A man sits on a chair with a toy in his hand. A toy sitting in a chair next to a hand. hands sitting on a chair with toys hands sitting on a chair with a toy
spray, water, hose, dog	fully-c.	A dog is sprayding water on a hose. A dog is being sprayed with water. A dog sprinkles water on a hose. A dog uses a hose to spray water on the ground. A dog is spraying water on a hose.
	fully-c. + post	A dog is sprayding water on a hose. A dog is being sprayed with water from a hose. A dog sprinkles water on a hose. A dog uses a hose to spray water on the ground. A dog is spraying water on a hose.
cut, pumpkin, knife, top	fully-c.	A man cutting pumpkins with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.
	fully-c. + post	A man cutting pumpkins with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top. A man cutting a pumpkin with a knife on top.

Table A.27: Outputs of a *fully-connected*-enhanced model and post-edits on "difficult" examples of length 4.

concepts	model	outputs
game, paper, kid, scissor, rock	baseline	A kid is playing a game of paper with scissors. A kid is playing a game with scissors and scissors. A group of kids playing a game with paper and scissors. A kid is playing a game with paper and scissors. A kid is playing a game of scissor with paper.
	baseline + post	A kid is playing a game of paper with scissors and scissors. A kid is playing a game of paper with scissors on a rock. A group of kids playing a game of rock with paper and scissors. A kid is playing a game of rock with paper and scissors. A kid is playing a game of scissor with paper on a rock.
peel, hand, knife, orange, hold	baseline	A woman holding a knife in her hand. A hand holds a knife with an orange and apeel. A man holding a knife with an orange peeling it. A man holding a knife with an orange and an orange on it. A man holding a knife with an orange and a peel.
	baseline + post	A hand holds a knife with an orange and apeel. A woman holding a knife in her hand with an orange. A man holding a knife with an orange peeling it from his hand. A man holding a knife with an orange and an orange on it in his hand. A man holding a knife with an orange and a peel in his hand.
use, gutter, stand, roof, clean	baseline	A man standing behind a gutter to clean the roof. A man uses a gutter to clean the roof. A man uses a gutter to clean a standing roof. A man uses a gutter to clean the roof. A man uses a gutter to clean the roof.
	baseline + post	A man uses a gutter to clean the roof as he stands. A man standing behind a gutter to clean the roof. A man uses a gutter to clean a standing roof. A man uses a gutter to clean the roof as he stands. A man uses a gutter to clean the roof as he stands.

Table A.28: Outputs of a baseline model and post-edits on "difficult" examples of length 5.

concepts	model	outputs
game, paper, kid, scissor, rock	all-concept	A kid is playing a game of scissors with paper and scissors. A kid is playing a video game with scissors and paper on a rock. A group of kids playing a game with scissors and paper. A kid is playing a video game on paper with scissors. A kid is playing a game of scissor with paper and rocks.
	all-concept + post	A kid is playing a game of scissors with paper and scissors on a rock. A kid is playing a video game with scissors and paper on a rock. A group of kids playing a game with scissors and paper on rocks. A kid is playing a video game on paper with scissors on a rock. A kid is playing a game of scissor with paper and rocks.
peel, hand, knife, orange, hold	all-concept	A man holding a knife with oranges and oranges on it. A hand holds a peel of an orange with a knife. A man holding a knife with an orange peel. A man holding a knife with oranges and lemon peel on it. A hand holding a knife and an orange peel.
	all-concept + post	A man holding a knife with oranges and oranges on it in his hand. A hand holds a peel of an orange with a knife. A man holding a knife with an orange peel in his hand. A man holding a knife with oranges and lemon peel on it in his hand. A hand holding a knife and an orange peel.
use, gutter, stand, roof, clean	all-concept	A man uses a gutter to clean a roof. A man uses a gutter to clean a man standing on a roof. A man uses a gutter to clean a roof. A man uses a gutter to clean his roof while standing. A man uses a gutter to clean the roof.
	all-concept + post	A man uses a gutter to clean a roof while standing next to it. A man uses a gutter to clean a man standing on a roof. A man uses a gutter to clean a roof while standing next to it. A man uses a gutter to clean his roof while standing. A man uses a gutter to clean the roof as he stands.

Table A.29: Outputs of a *all-concept*-enhanced model and post-edits on "difficult" examples of length 5.

concepts	model	outputs
game, paper, kid, scissor, rock	fully-c.	A kid is playing a game of scissors on paper with rocks. A kid is playing a game with scissors on paper. A kid plays a game with paper and scissors. The kid is playing a game of scissors on paper. A kid is playing a game of scrabble with paper and scissors on a rock
	fully-c. + post	A kid is playing a game of scissors on paper with rocks. A kid is playing a game of rock with scissors on paper. A kid plays a game of rock with paper and scissors. The kid is playing a game of scissors on paper with rocks. A kid is playing a game of scrabble with paper and scissors on a rock
peel, hand, knife, orange, hold	fully-c.	A man holding a knife with oranges in his hand. A man holding a knife with an orange peel on it. A hand holding a knife and an orange peel. A man holding a knife with an orange peel on it. A hand holding an orange with a knife.
	fully-c. + post	A man holding a knife with oranges in his hand. A man holding a knife with an orange peel on it in his hand. A hand holding a knife and an orange peel. A man holding a knife with an orange peel on it in his hand. A hand holding an orange with a knife.
use, gutter, stand, roof, clean	fully-c.	A man standing next to a gutter that is being used to clean the roof. A man uses a gutter to clean the roof. A man is standing on a roof and cleaning a gutter. A man uses a gutter to clean his roof. A man uses a gutter to clean the roof.
	fully-c. + post	A man standing next to a gutter that is being used to clean the roof. A man uses a gutter to clean the roof as he stands. A man is standing on a roof and cleaning a gutter for use. A man uses a gutter to clean his roof as he stands. A man uses a gutter to clean the roof as he stands.

Table A.30: Outputs of a *fully-connected*-enhanced model and post-edits on "difficult" examples of length 5.