FINE-GRAINED AND COARSE-GRAINED CAUSAL REASONING IN PROCEDURAL TEXTS

Hainiu Xu

A Master Thesis

in

Data Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Master of Science in Engineering

2023

Supervisor of Dissertation

Christopher Callison-Burch, Professor of Computer and Information Science

Graduate Group Chairperson

Susan Davidson, Professor of Computer and Information Science

FINE-GRAINED AND COARSE-GRAINED CAUSAL REASONING IN PROCEDURAL TEXTS

© COPYRIGHT

2023

Hainiu Xu

This work is licensed under the
Creative Commons Attribution
NonCommercial-ShareAlike 3.0
License

To view a copy of this license, visit

http://creativecommons.org/licenses/by-nc-sa/3.0/

Dedicated to my family and friends.

I would not have made it this far without your support.

ACKNOWLEDGEMENT

I would like to express my utmost appreciation to Professor Chris Callison-Burch for his continuous support throughout my Master's study. Professor Callison-Burch has not only been a great mentor in academics but also a mentor in life. Professor Callison-Bruch has set a great example of a mentor, a researcher, and a learner that I wish to grow up to be one day.

I wish to pay special gratitude to Yue Yang and Li (Harry) Zhang, both of whom provided invaluable advice and guidance throughout my Master's study. Their patience and whole-hearted assistance not only helped me to become a better researcher, but also taught me how to become a mature team leader. They are the group of people that I will look up to during my Ph.D. study.

I would also wish to thank my family. My father and mother provided enormous support both financially and emotionally during my Master's study. They are the reason why I have the privilege to study in the United States. My wife has been very considerate throughout my study. Her support and understanding allow me to focus on my research and thesis works. My wife also helped tremendously with data annotation and discussing ideas for this thesis.

Finally, I wish to show my sincere gratitude to my friends. Their encouragement gives me confidence and helps me to become a better person. I want to especially thank Yuezhi Xie and Shuangshuang Li for their help with data annotation. This thesis would not be complete without their help.

ABSTRACT

FINE-GRAINED AND COARSE-GRAINED CAUSAL REASONING IN PROCEDURAL TEXTS

Hainiu Xu

Christopher Callison-Burch

The ability to make causal inferences is inherent to humans and substantial to our intelligence and civilization. Yet, such a crucial capability is lacking even in the state-of-the-art Large Language Models (LLMs). To pave the pathway toward Artificial General Intelligence, granting machines a similar capability of conducting causal inferences is an indispensable building block. Causalities come in different granularity. At a coarse-grained level, causal relations exist between events. At a fine-grained level, causal relations exist between events and participating entities. From an event-centric perspective, an intelligent agent shall be able to infer the causal effects that one event could bring to related entities as well as other events. For instance, the event of *heating up a pan* will cause the attribute, *temperature*, of the participating entity, *pan*, to rise. Being able to comprehend the causal effect of events at both a coarse-grained and fine-grained level will bring significant benefits to downstream tasks such as commonsense reasoning, multi-hop question answering, planning, and so on. My thesis research focused on constructing benchmarks and building learning systems that can (1) discern entity state changes by inferring their causal relationship with events; (2) estimate the likelihood of an event happening by deducing its causal relation with context entities; (3) mitigate reporting bias in languages by leveraging external modalities such as vision (video) to provide extra information on participating entities; and (4) constructing dynamic causal diagrams based on fine-grained entity state information in procedural texts.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

CHAPTER 1 : Introduction

1.1. Historical Background

Statistical Learning and Machine Learning have achieved great success in the last decade in light of Artificial Neural Networks and Deep Learning. The increasingly large models are adept at exploiting the associative information between features and labels. Indeed, using associative information alone is sufficient for achieving a good performance on a wide range of tasks. For instance, in sentiment analysis, where a machine identifies the mood of a writer through a piece of text, exploiting the associative relationship at a semantic- or even syntactic-level oftentimes lead to promising results.

The exploitation of associative information dates back to the early age of statistics. In the late 19th century, Fracis Galton and Karl Pearson, attempted to tackle heredity from a causal perspective. They later concluded that causality is unsolvable at the time and decoupled associativity from causality. This causal-free, associative-based ideology greatly impacted the development of modern statistics and its effects extend all the way to deep learning.

Originally built based on the Perceptron, deep neural networks inherent spirits from regression models, which dedicate to capturing only the associative relationship amongst features. Leveraging associative information alone, deep learning models achieved superhuman performance on a variety of tasks. Yet, ignoring causality comes with a price. As deep learning models become more integrated with our daily life, the issue of associativity-oriented learning magnifies. For instance, people start to notice that deep learning models contain societal, gender, and racial biases. While some blame the training data, it is associativity-oriented learning that captured such inappropriate associations. Further, the lack of causal reasoning capabilities hinders the interpretability and faithfulness of deep learning models, severely limiting their application in domains like medical and pharmaceutical research. Therefore, can we integrate existing capable deep learning models with causal reasoning frameworks

to improve their reliability and intelligence?

1.2. Commonsense Causal Reasoning

The ability to make causal inferences is inherent to humans and substantial to our intelligence and civilization. As shown in Figure 1, our ability of making causal inference is reflected by the unique capability of reasoning about *intervention* and *counterfactuals*. Intervention allows humans to discern causal relationships by observing the difference in the outcomes of alternative actions. Based the rudimentary causal information obtained from intervention, *counterfactuals* helps humans to concretely conduct causal inference by reasoning about the imaginary events.

Historically, causal inference is done using controlled experiments such as Randomized Controlled Trials (RCTs) [23]. The major obstacle of conducting controlled trials is that they are extremely expensive to collect large amount of data. This is especially troublesome for deep learning as large models demand large amount of data to train. Advancements in Large Language Models (LLMs) introduced a new deep learning paradigm– In-Context Learning (ICL) [14]. In ICL, the model performs few-shot learning by looking over several in-context examples and learning the syntactic structure and semantic association. This learning approach alleviates the burden of data curation and avoids the risk of learning spurious correlations during finetuning. Furthermore, benefitting from training with a tremendous amount of data, LLMs encapsulate a vast amount of high-level knowledge. Therefore, RCT can be conveniently applied to conduct causal inference in NLP where LLMs are used to simulate RCT experiments. An example is the ROCK framework [138] for causal inference about events in which the intervention events are generated using the GPT-J model [113] (see Figure 3 for a demonstration of the ROCK framework and see Section 2.4 for details).

To pave the pathway toward Artificial General Intelligence, granting machines a similar capability of conducting causal inferences is an indispensable building block. Causalities come in different granularity. At a coarse-grained level, causal relations exist between

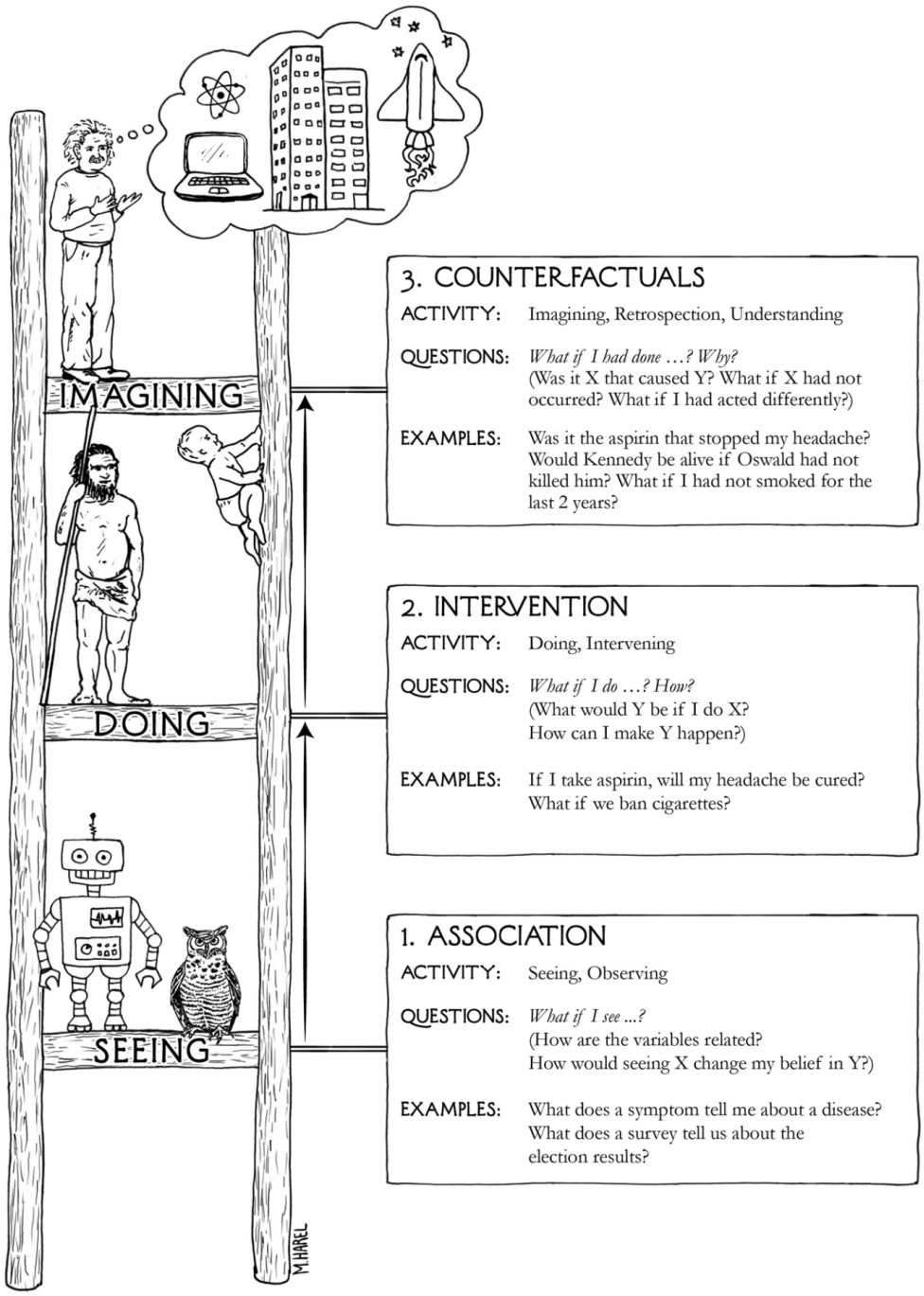

Figure 1: An illustration of the ladder of causation. Figure adapted from pearl2018book.

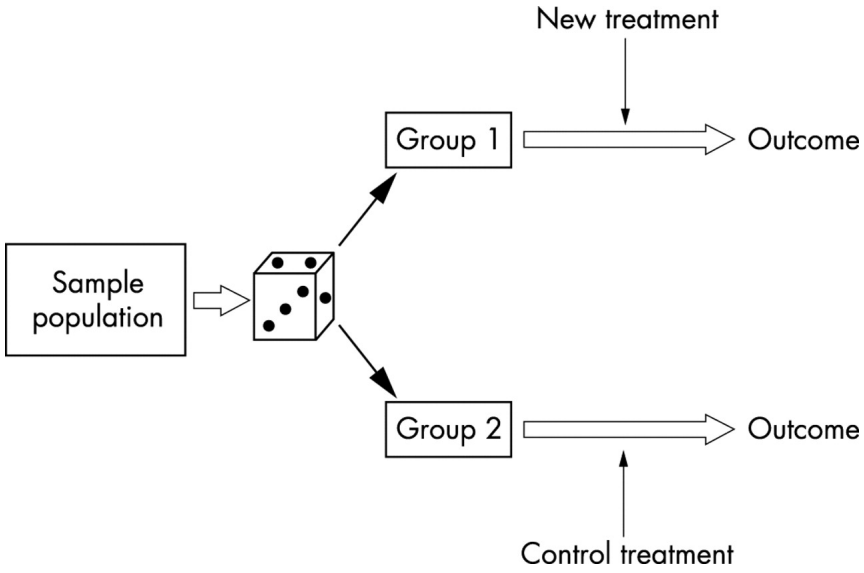

Figure 2: An illustration of the process of Randomized Controlled Trials. Figure adopted from RCT. To adept the RCT framework to NLP with language models, the "*New treatment*" and "*Control treatment*" group of texts can be autoregressively generated by language models.

events. At a fine-grained level, causal relations exist between events and participating entities. From an event-centric perspective, an intelligent agent shall be able to infer the causal effects that one event could bring to both the related entities and to other events [12]. For instance, the event of *heating up a pan* will cause the attribute, *temperature*, of the participating entity, *pan*, to rise. Being able to comprehend the causal effect of events at both a coarse-grained and fine-grained level will bring significant benefits to downstream tasks such as commonsense reasoning, multi-hop question answering, planning, and so on.

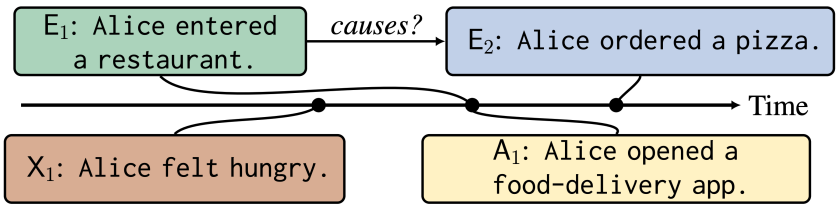

Figure 3: A demonstration of the ROCK framework adopted from the original paper [138]. E_1 is the potential cause and E_2 is the observed effect. The event X_1 is some event happened before E_1, which is used to regular the temporal relation between the intervention events and E_1. The event A_1 is an example of model-generated intervention event.

1.3. Procedural Texts

Instructions are educational texts composed by domain experts to help amateurs achieve a certain goal. Depending on the nature of the task, an instruction can be temporally ordered or not. For instance, "operating a surgery" needs to be done in a certain order while "making a friend" does not. The group of instructions that follow a strict temporal order is often referred to as *procedures*.

A procedure typically consists a *goal* and a series of *steps*. The temporality of procedures is manifested through its steps in that the successful execution of each step depends on that of the previous step. For instance, in a procedure of "making a cup of tea", the step "pour hot water onto the tea bag" depends on the execution of the previous step, "pick a flavor and place the tea bag in the cup".

As opposed to user manuals or text books, whose purpose is to give an all-around instruction, procedural texts aim at providing concise and accessible instruction. Therefore, procedures are extremely goal-oriented and the steps oftentimes only contain the most salient objects and high-level events. This conciseness brings two challenges– *reporting bias* and *entity state tracking*.

1.4. Commonsense and Reporting Bias

Reporting bias refers to the discrepancy between reality and its decription in text [41]. In the context of procedural texts, reporting bias mainly manifest through the circumstances where curtain information regarding either the entity or the events is ignored in a step. For example, in a procedure of "doing laundry", a step "take out the washed clothes and put them into the drier" ignore numerous entities and events such as "laundry machine door (entity)", "drier door (entity)", "open the laundry machine door (event)", and "open the drier door (event)". Humans are capable of inferring these information from the high-level text description, which makes explicitly mentioning of such fine-grained information redundant. People usually refer to this family of knowledge that humans understand without

explicit description as *commonsense knowledge*.

Commonsense is the capability of perceiving, understanding, and judging things that is shared by all people [42]. Therefore, to make communication concise and efficient, commonsense knowledge are oftentimes made implicit in both spoken and written language as it is reasonable to assume that all humans can fill in the missing information with their commonsense. This seemingly trivial and redundant task for humans turn out to be challenging and necessary for machines. The absence of commonsense knowledge in language (training data) hinders deep learning models' capability of conduct commonsense reasoning tasks [103].

1.5. Entity State Tracking

Entity State Tracking is a commonsense reasoning task targeting the issue of reporting bias. *Entity* refers to the participating objects in an event and *state* refers to the physical and psychological property of an entity. For example, the event "put coffee beans in a grinding machine" involves *entities* such as "coffee beans", "grinding machine", "coffee bean bags", and "machine lid". Some of their *states* are "coffee beans are in bags before (*pre-condition*) and in grinding machine afterwards (*post-condition*)", "coffee bean bags are full before (*pre-condition*) and emptier afterwards (*post-condition*)". Through the course of a procedure, entity states are constantly affected by events at each step. The association between step and entity state changes are commonsense and usually made implicit in procedures. Therefore, improving machine's capability of deducing the pre- and post-condition of entity states in each step of the procedure is crucial for commonsense reasoning with machines.

Entity state tracking has been a trendy research topic in multiple disciplines of AI. In CV and robotics, studies focus on reason about the existence and location of entities. Hence, such works are grouped under the term *object detection* or *object tracking* [142, 114]. In NLP, early efforts focus on tracking entity states in synthetic texts [122, 70]. Recent studies shifted attention to entity state tracking in human-written passages including recipes [59, 11],

scientific process [25, 83], short stories [98], and articles from online how-to websites such as WikiHow[1] and Instructables.com[2] [108, 125]. Due to the sequential nature of entity state changes, previous modeling attempts have mainly focused on building rule-based systems [22] or using RNN-based architectures [104, 11, 106, 46, 43]. Recent works opt for in-context learning models such as the GPT family [108, 140, 96, 14].

1.6. Thesis Statement

In this thesis, I argue that solely focusing on causal reasoning about high-level events are not sufficient for machines to learn causality. With the same arguments about *reporting bias* made above, fine-grained causal relationships between entities states are the key to granting machines a real sense of causality. My research aims to build learning systems that can (1) discern entity state changes by inferring their causal relationship with events (Chapter 3); (2) estimate the likelihood of an event happening by deducing its causal relation with context entities (Chapter 3); and (3) mitigate reporting bias in languages by leveraging external modalities such as vision (video) to provide information on participating entities (Chapter 4).

[1] `wikihow.com`
[2] `instructables.com`

CHAPTER 2 : Literature Review

2.1. Reasoning About Procedures in NLP

Extracting entities and attributes from unstructured texts and documenting their relations as structured or latent representations is at the core of procedural reasoning. The evolution of methods for representing entity-attribute relations has largely relied on the proposition of new tasks and datasets. Hence, after a brief review of the non-neural methods, works involving neural systems will be presented according to the specific task and dataset that they focused on. Despite the recent spike of research on neural networks, numerous studies proposed methods using rule-based or statistical learning models to represent entity-attribute relations as directed graphs [38, 141, 88, 7, 76, 58] or structured logical expressions [71, 17, 22]. Yet, due to the limited expressivity of the non-neural models, these works are domain-specific, demand laborious engineering effort, and assume a set of predetermined logical relations. Further, issues like entity co-reference, which is prevalent in natural language, will also hinder the accuracy of these representations. For instance, in a procedure describing the metabolism of the *sunflowers*, the entity, "*sunflowers*" can be also referred to as "*the plant*", "*flower*", or even its pronoun, "*it*". Therefore, using statistical learning method overlooked the flexibility that entity state tracking demands and have led to great loss in information during learning.

Neural networks alleviated these concerns by introducing versatile word embeddings [79, 91]. One pioneering works of using neural models for entity state tracking is [123], who proposed a LSTM-based Memory Neural Network model (MemNN) for simple question-answering (QA) tasks [50]. Originally designed to answer multihop questions, MemNN is later applied to a set of synthesized QA tasks (bAbI tasks[1]) that require more in-depth reasonings [122]. The bAbI tasks such as *Time Reasoning*, *Basic Deduction and Induction*, and *Positional and Size Reasoning* are the first set of tasks that are closely related to procedural reasoning.

Inspired by MemNN and the bAbI tasks, [45] proposed Recurrent Entity Network (Ent-

[1] `https://github.com/facebook/bAbI-tasks`

Net), which uses the control-gates mechanism introduced by [50] to dynamically document implicit entity-related information via embeddings in a fixed number of memory cells. Similarly, Query-Reduction Networks (QRN) [104] implicitly document answer-related entity attributes using modified vanilla-RNN. QRN avoids the gradient-vanishing issue of vanilla RNN by conducting implicit answer-related entity attribute updates for every input sentence. This line of work based on the bAbI tasks showcased the neural network's capability of performing entity-attribute tracking. The bAbI dataset, however, has its limitation– texts in bAbI are machine-generated. Such synthesized texts, although easy to collect, will potentially cause poor generalization to real-world data.

To address this issue, ProPara dataset [24] uses human-written texts on simple scientific procedures. The label of the procedures describes two attributes of an entity, namely its *existence* and *location*. The ProPara dataset excels comparing to previous entity state tracking datasets in that it is the first entity state tracking dataset that is completely human-annotated. Further, ProPara narrows the state of entity to the *existence* and *location*. Although such a reduction might seem undesirable nowadays, it makes the entity state tracking task more well-defined and allows researchers to draw connections with object tracking task in computer vision. Base on the ProPara dataset, a large group of works focus on entity state tracking, the most prominent of which include `ProLocal` and `ProGlobal` [24], which adopts a biLSTM with Attention approach, the KG-MRC model [27], which is the first graph neural network model tailored to entity-attribute tracking, the ProGraph model [143], which builds on the idea of KG-MRC by including more knowledge in the graph, the NCET model [43], which treats ProPara as a tagging problem and adepted the Neural Conditional Random Field (CRF) model from Named Entity Recognition, the REAL model [52], which combines the NCET framework with graph neural networks by constructing a graph representation of the relationships between entities, actions, and locations, and embeds the corresponding node with a Graph Attention Network [112].

ProPara challenges models to reason about an entity's two attributes: existence and loca-

tion. Procedural reasoning, however, demands a much more in-depth understanding and reasoning about entity attributes. To build a neural system that can conduct more complicated reasoning about procedural events, [11] annotated cooking procedures from *Now You're Cooking!* recipe library through crowd-sourcing and trained the Neural Process Network based on the resulting RECIEPS dataset. Neural Process Network tracks entity states by simulating the causal effects of actions applied to entities. Hence, the resulting entity attributes are much more versatile and informative than existence and location alone.

To further facilitate the model's capability of conducting complicated entity-attribute tracking and understanding causal relations between entity and actions, more recent works have shifted their source of data to large-scale how-to websites such as WikiHow[2] and Instructables [3] where diversified real-world procedural instructions are easily accessible. OpenPI [108] utilizes rich how-to guides from WikiHow and annotated entities' states change through crowd-sourcing. To model entity state dynamics, [108] follows the trend of pretraining-finetuning pipeline in NLP and proposed to fine-tune GPT-2 on the OpenPI dataset [96]. This work demonstrated the capability of foundation models on making complicated inferences and opened a new route toward conducting procedural reasoning. Also using the power of foundation models, [105] proposed to tackle procedural reasoning by injecting knowledge of the environment during pretraining. By sampling entity-action pairs from the SCoNE, RECIPES, and ProPara dataset, a BART model is pretrained with an auxiliary task of predicting the resulting entity state given an action [71, 11, 25, 61]. [125] crawled data from both WikiHow and Instructables and proposed an action conditions learning task where, given an action in the procedure, the model needs to infer the action's precondition as well as the resulting postcondition of the entities. [125] finetuned pretrained RoBERTa-LARGE model on labeled action pairs and again showcased that the modern pretrain-and-finetune strategy of foundation models can reliably learn the complicated entity dynamics in procedural texts.

[2]https://www.wikihow.com/Main-Page
[3]https://www.instructables.com/

In this thesis, we focus on application of entity state tracking models. Specifically, we look into how entity state information can be leveraged to conduct causal inference between events in procedures (Chapter 3). We utilized the OpenPI dataset to conduct experiment on how entity state produced by the OpenPI-finetuned model can help with elicit the causal relationship between events by completing a causal chain with the entity states (see the bottleneck model in Figure 4).

2.2. Reasoning About Procedures with External Modality

Similar to procedural reasoning with NLP, most multimodal procedural reasoning tasks are formulated as QA problems, the precursor to multimodal procedural reasoning in CV is the Visual Question Answering (VQA) task [5]. In the VQA task, the model is given an image and is required to answer a question regarding the image content. A significant work based on VQA is the Neural Module Networks (NMN) [4]. NMN is composed of several jointly trained neural network modules tailored for specific types of questions. During inference time, the intention of the question is determined by semantic parsing, and the specific task is passed to the designated neural network module. NMN inspired as a line of work [51, 55, 92] targeting visual reasoning tasks based on the CLEVR dataset [54].

VQA using a single static image imposes severe limitations on models' capability of inferring temporal information, which is indispensable to procedural reasoning. To address such an issue, [109] extended visual reasoning using images to multimodal reasoning by introducing the MovieQA task and dataset, which consists of both natural language (plot synopses, subtitles, movie scripts) and video clips. MovieQA is formulated as a multiple-choice reasoning task where, given a source of information (text or image) and 5 candidate answers, the model needs to pick the correct answer. For questions that involve multimodality, visual-semantic embeddings [145] are used to fuse independently-learned representations. The answer is then predicted by passing the aligned multimodal representation to the MemNN model [123]. Such an approach of independently learning the representation of different modalities and later aligning them with a fusion model is widely used in later research works with

different choices of text/image encoders and fusion methods [53, 56, 3, 132, 36, 135].

Concurrent with MovieQA are several other multimodal reasoning tasks that involve reason about simple scientific concepts [56], temporal information from comic panels [53], and cooking recipes [127]. These proposed tasks contain multiple-choice questions about visual cloze, visual coherence, visual ordering, multimodal textbook exercises as well as fill-in-the-blank questions formulated as textual cloze problems.

Another significant work in multimodal procedural reasoning is the Visual Commonsense Reasoning (VCR) task [132]. The VCR task is formulated as another multiple-choice problem consisting of an image with marked Regions of Interest (RoI), a textual question, and a set of answers that contain special tokens referring to the RoIs. Besides choosing the correct answer, VCR takes one step further by requiring the model to do another multiple-choice problem of choosing the correct justification for its previous choice. This extra multiple-choice problem makes this task significantly harder than the previous tasks as the model not only need to reason about the original procedural instructions but also to comprehend given explanations and determine their relevance to the original question.

To tackle the complicated multihop reasoning task, [132] proposed the Recognition to Cognition (R2C) Networks. Given an image, a question prompt, and a set of candidate answers, R2C first obtains independent representations of natural language and image through pre-trained foundation models (BERT, ResNet-50) [28, 44]. These modality-specific representations are fused with a biLSTM network. Further, two separate attention mechanisms are applied between (1) pairwise fused question prompt representation and fused answer representation as well as (2) fused answer representation and raw visual representation of RoIs for further fusion. The resulting two representations are concatenated and passed through another biLSTM network to produce the final rationale. A similar modeling approach is taken by the Procedural Reasoning Networks [3] to tackle the RecipeQA tasks. RecipeQA consists of data crawled from the cooking section of Instructables and proposes reasoning tasks including visual cloze, visual coherence, visual ordering, and textual cloze[127].

Also based on the visual tasks of the RecipeQA dataset (excluding the textual cloze task), MLMM-Trans model [68] adopts the original Transformer architecture [111] to tackle multimodal tasks. The original encoder is used to encode instructions from each step. The image representations are input to the decoder. Instead of the original masked multi-head attention module, the decoder block in MLMM-Trans consists an identical multi-head attention module as the encoder to conduct modality fusion. The fused representation is then concatenated with original image embeddings and passed to the uni-directional LSTM network to conduct another sequence modeling. Since MLMM-Trans only solve the visual multiple-choice tasks, the final representation is input to a feed-forward network to conduct the final prediction.

[137] extended the RecipeQA dataset by adding crafting procedures also from Instructables. Based on the extended dataset, [137] takes a multimodal graph neural network approach and proposed TMEG. TMEG first constructs two homogeneous graphs of different modalities. For language content, noun phrases, which are regarded as entities, are extracted via POS tagging. The embedded entity nodes from the same instruction are connected to form a homogeneous (text) graph. Similarly, RoIs of instruction images are extracted using Faster-RCNN, and a homogeneous (image) graph is constructed per instruction [99]. Further, a heterogeneous graph is built by adding temporal edges, which document the temporal evolution of entities as well as modal edges, which indicate the interaction between the two modalities. The node encoding different modalities is first projected to a common embedding space via two MLPs and further fused with a modified VisualBERT framework where information from temporal edges, as well as modal edges, are also injected into the modal [64].

There is also a line of work utilizing the multimodal how-to instructions from WikiHow [130, 124, 128]. While the instructions from WikiHow are less noisy compared to Instructables, the majority of images from WikiHow are human drawings of a particular style instead of real-world scenes. Such a discrepancy, similar to the issue with using synthesized data, will

cause generalization concerns.

A recent line of work shifted attention from discrete text-image instruction to continuous instructional videos. [136] proposed the MERLOT model that follows the pretraining-finetuning pipeline. MERLOT is pretrained via self-supervised learning tasks based on video and automatic transcriptions curated from YouTube. Instead of using the representation of RoIs, MERLOT's vision encoder produces grid-based feature representation based on Vision Transformers [30] which is computationally more efficient than traditional object recognition algorithms. The visual features and text embeddings from automatic transcriptions are fused using a modified RoBERTa model [69]. The pretraining objectives of MERLOT include (1) CLIP-style contrastive learning where the model aims at maximizing the similarity between encoded video clips and the corresponding transcription embeddings. (2) VisualBERT style Masked Language Modeling where the joint vision-language model learns to reconstruct input text with 20% corrupted tokens. (3) Temporal reordering objective, which is akin to the Sentence Order Prediction but applied to a sequence of images. The resulting MERLOT model showed promising performance both on video commonsense reasoning tasks as well as VCR tasks with static images. MERLOT demonstrated the necessity of utilizing visual sources that contain rich temporal information. A follow-up work, MERLOT Reserve [134] further included audio modality and improved model performance by a small margin.

In this thesis, we follow the ideologies proposed by the previous studies and attempt to incorporate external modalities to aid procedural reasoning. Specifically, we focus on using external modalities such as video and audio transcription to mitigate the reporting bias procedural texts (Chapter 4). We leverage multiple multimodal approaches including object detection, image captioning, and visual question answering to extract entities that participated in the procedure what are unmentioned in the texts.

2.3. In-Context Learning

The research of NLP with deep learning has gone through numerous stages, of which the most impactful are the Pretrain-and-Finetune paradigm and the in-context learning paradigm. The Pretrain-and-Finetune paradigm contains milestone works such as the Transformer architecture, and its applications such as BERT, GPT-1, and GPT-2 [111, 28, 95, 96]. This line of work had profound impacts on numerous areas including NLP, Computer Vision, Robotics, and Multi-modal learning [63, 30, 2, 94].

The In-Context Learning (ICL) paradigm is pioneered by Large Language Models (LLMs) such as GPT-3, GPT-4, ChatGPT, Flan-T5, T0, LaMDA, and PaLM[14, 1, 21, 102, 110, 20]. Different from the Pretrain-and-Finetune paradigm, where the workflow involves finetuning a pretrained model with training data, ICL is a few-shot learning approach and learns new tasks solely based on understanding the syntactic and semantic information provided in the few-shot demonstrations. While supervised finetuning risks the model from picking up spurious correlations in the training dataset, ICL conveniently mitigates the issue as no gradient is backpropagated– blocking one major source of spurious correlation.

Dataset	Quantity (Tokens)	Fraction in Training
Common Crawl	410 Billion	60%
WebText2	19 Billion	22%
Books1	12 Billion	8%
Books2	55 Billion	8%
Wikipedia	3 Billion	3%

Table 1: Training data used to pretrain GPT-3 (175B). These statistics are acquired from the original GPT-3 paper [14]

ICL is only possible with LLMs as ICL relies heavily on the *instruction following* capability of language models, which is one of the emergent abilities that are only present in models with more than 10 billion parameters [119]. Further, although ICL is a gradient-free learning method, the capability of ICL models heavily depends on the pretraining data. Traditionally, LLMs are pretrained with a massive amount of text data (see Table 1). Recent studies unveiled the benefit of pretrain language models with a combination of code and

text data [35]. The direct benefit of pretraining with code is the Chain-of-Thought (CoT) reasoning capability [120]. CoT is a prompting strategy for tackling complex, multi-hop reasoning tasks. Instead of showing LLMs in-context examples that directly solve complex reasoning tasks, CoT provides a reasoning chain that decomposes a complex reasoning task into a series of simpler components and combines them with logic. The CoT prompting paradigm brought huge improvements in complex reasoning problems such as mathematical reasoning, commonsense reasoning, multihop QA, planning, and logical reasoning [74].

Building on these ideas, we focus all our study on In-Context Learning to minimize the potential impact that subjectivity in the data annotation could have on our study. We conducted ICL with primarily OpenAI models such as GPT-3 and Codex as they are the state-of-the-art LLMs available during the time of our study. Further, we also investigate methods that utilize CoT prompting both as an Engligh prompt and as a Python code prompt. These results will be presented in Chapter 3 and Chatper 5.

2.4. Commonsense Causal Reasoning

Causal reasoning has been studied extensively in statistical inference and statistical learning [34, 89, 90]. In the early attempts of studying causality with machine learning, causal reasoning is formulated as a binary semantic relation classification task, where a model needs to classify whether there exists a cause-effect relationship between an entity pair given the context[39]. Enlightened by the SemEval task, several subsequent studies dedicated to curating datasets for semantic causal reasoning from online articles such as CNN news and Wikipedia[4][5][8, 29, 18, 48]. Further, these pioneering works on learning causal relationships draw the connection between causal and temporal relationships. In fact, numerous works have argued the relationship between temporal reasoning and causality can be leveraged for causal inference [101, 49, 15]. This relationship between causal and temporal relationship greatly affected the causal framework of later works.

[4]https://www.cnn.com/
[5]https://www.wikipedia.org/

In the era of deep learning, causal reasoning tasks are mostly formulated as classification problems with a fixed label space typically consisting of labels such as *causal*, *non-causal*, and *difference degrees/types of causal relationship*. A line of work focuses on the breadth of causal information and opts for the automatic curation of causal reasoning datasets such as the CausalNet dataset [73] and the CausalBank [66] dataset. There have also been efforts to construct high-quality, human-annotated datasets such as the SemEval 2007 dataset [39], the COPA dataset[40], and the e-CARE dataset [32]. Further, works on causal inference tackle causal reasoning with different syntactic granularity with works focusing on word-level granularity[39, 47, 29, 73, 84, 82, 86, 118], phrase-level causality [9, 81, 33, 18, 66], and sentence-level causality [100, 86, 32].

In NLP and CV, *Commonsense reasoning* is often used as an umbrella term for causal reasoning. Elements of causal inference are present in multiple commonsense reasoning works such as inductive reasoning [133], abductive reasoning [10], goal-step inference [139], entity state tracking [108], and choosing alternatives of events [100]. Recent works in causal NLP shifted attention to utilizing the notion of causality in model distillation [126] as well as probing the causal knowledge of pre-trained language models [65]. In the intersection of CV and NLP, works incorporate causal reasoning in the form of Visual Question Answering, Visual Commonsense Reasoning, Visual Abductive Reasoning, and Visual Goal-Step Inference [6, 132, 67, 130].

As most causal reasoning tasks are formulated as either a classification task or a Multi-Choice Reading Comprehension task, models used for causal reasoning are mostly autoencoding language models such as BERT, RoBERTa, ALBERT, and XLNet [57, 69, 60, 131]. Entering the era of LLMs, there are also attempts to use autoregressive or encoder-decoder models such as BART, GPT-2, and GPT-3 to conduct causal inference [62, 96, 14]. Since the prevalence of In-Context Learning (ICL), many works attempt to improve the interpretability and faithfulness of LLMs by including explanations for causal relation as a part of the dataset [129, 32]. Further, there are also primitive studies that propose LLM-based

causal reasoning frameworks such as the ROCK framework[138]. In the ROCK framework, the authors leveraged the close relation between temporal relation and causal relation and used the strength of temporality as a surrogate for measuring causality. The authors of the ROCK framework used the concept of Average Treatment Effect (ATE), which is widely used in the study of causality. The ATE is often used with invertention studies. In the ROCK framework, the interventions are generated by GPT-J [113] and the strength of temporal relationships are computed using the label probability given by a RoBERTa model finetuned for temporal prediction. The causal relationship is then determined by computing the propensity score, which is the difference in the strength of the temporal relationship between the effect following the treatment versus that of following the interventions.

CHAPTER 3 : Causal Reasoning about Entities and Events

3.1. Motivation

Event-centric natural language processing has been studied extensively from various aspects [19]. Among these studies, there are works that involve elements of causal reasoning such as event-wise causal reasoning [133, 107, 31, 125, 10], which dedicates to reason about the temporal and causal relation between high-level events, and event-entity causal reasoning [13, 26, 85, 108], which reasons about the cause and effect of high-level events on low-level entity states. Many of the previous studies choose procedural texts as the media because of their richness in events and dynamically changing entity states (Chapter 1.3).

The majority of previous works solely focus on causal reasoning between either two events or an event and an entity state– very few explored the possibility of using low-level entity state information as a bottleneck to aid the reasoning between high-level events (Figure 4). For instance, in a procedure of "*making coffee*", the event *"coffee can be made by pouring hot water onto the coffee beans"* is entailed by the entity state *"coffee beans are ground"*. This bottleneck model of event-centric causal reasoning ties the aforementioned two tasks, namely event-and-event and event-and-entity causal reasoning.

As shown in Figure 4, we wish to first deduce some salient entity state change by conducting *event-and-entity* causal reasoning. In the language of probability, this bottleneck model is providing a prominent prior to our causal reasoning process (e.g. converting the problem of $\mathbb{P}(\text{event1} \rightarrow \text{event2})$ to $\mathbb{P}(\text{event1} \rightarrow \text{event2} \mid \text{salient-entity-state})$). Ideally, the deduced entity state shall serve as a necessary component of the causal chain. For instance, it is difficult to deduce the causal relationship between "*pour coffee bean into the grinding machine*" and "*coffee can be made by pouring hot water onto the coffee beans*" due to missing information about the state of the coffee beans. Therefore, we wish to inject the knowledge that "*coffee beans are ground*" and form a causal chain as shown in Figure 4. With the deduced low-level entity state information, we then conduct another round of

entity-and-event causal reasoning by deducing the causal relationship between the entity state and the targeting event. We wish to demonstrate that, although implicit to humans, understanding the state of entities that are of mutual interest for both events is key to causal reasoning.

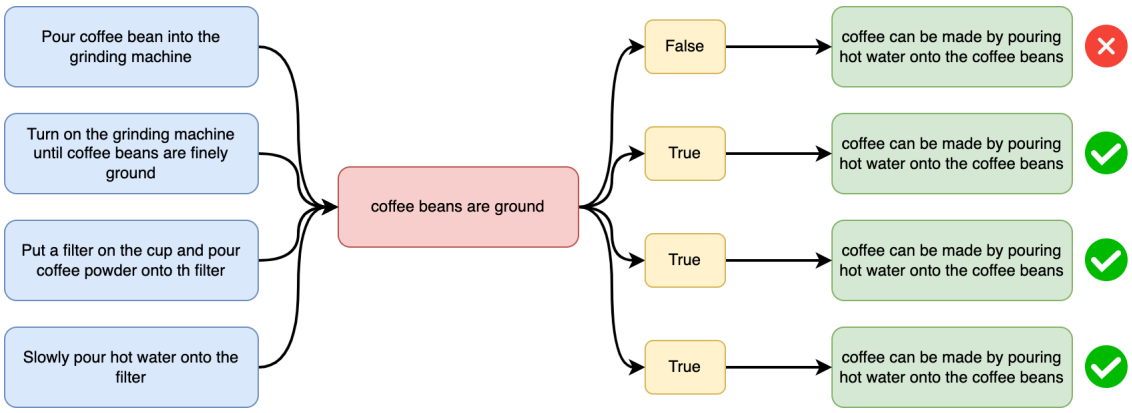

Figure 4: An illustration of the event-centric causal reasoning pipeline where the entity state *coffee beans are ground* serves as a bottleneck. Texts in the blue box are steps of the procedure. The red box shows an entity state. The yellow box shows the entity state status at each step of the procedure. In this example, the entity state has a causal relationship with the second step. This is reflected by the status of the entity state shift from `False` to `True`. The green box shows an imaginary event of which we wish to deduce the change in the likelihood of happening.

This idea is in line with the Chain-of-Though prompting paradigm where one elicits a multi-hop reasoning task by decomposing it into a chain of single-hop reasoning problems [121]. In our case, we break a multi-hop causal reasoning chain that involves both coarse-grained inter-event causal reasoning and fine-grained event-entity causal reasoning into two single-hop causal reasoning problems– one for event-and-entity causal reasoning and another for entity-and-event causal reasoning.

To demonstrate the effectiveness of the multi-hop bottleneck model of causal reasoning, we propose the task of **C**ausal **R**easoning of **E**ntities and **E**vents in **P**rocedural Texts (`CREPE`) with a demonstration in Figure 5. Given a procedure consisting of a *goal* ("stir fry vegetables") and a series of *steps* ("rinse vegetables" ...), a model is to predict the likelihood of some unobserved events ("there is a sizzling sound") after the execution of each step. To

benchmark the helpfulness of entity state information in event-wise causal reasoning, we proposed two causal reasoning tasks: a traditional *event-wise causal reasoning task* (how do events in a step alter the likelihood of another event) and a *bottleneck reasoning task* (how do events in a step alter the state of some entity, which entails a change in the likelihood of another event).

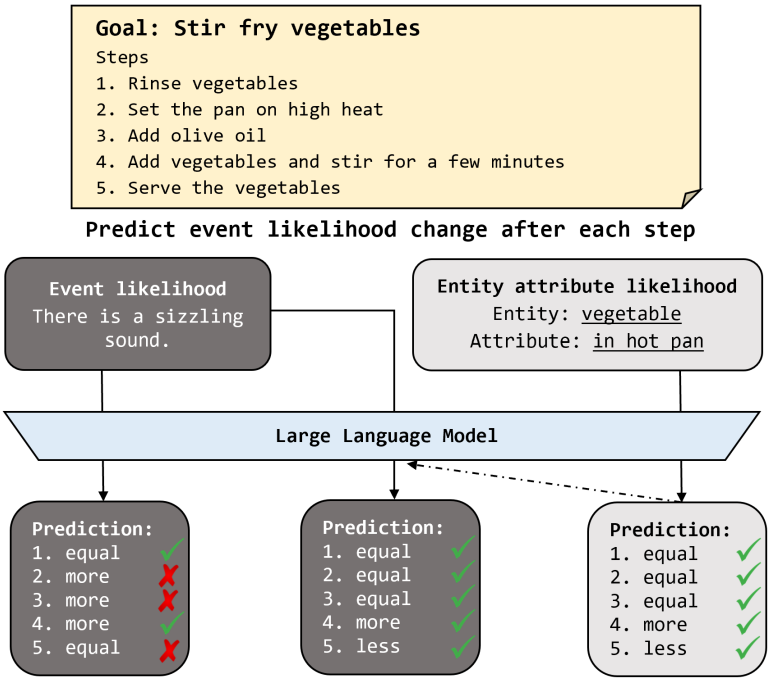

Figure 5: Demonstration of the CREPE task. Given a *goal* and a series of *steps* (texts in the yellow box), a Large Language Model (blue box) is asked to predict the change in the likelihood of some unobserved events (black box titled "Event Likelihood"). We compared the approaches of (1) direct reason between events (bottom left of the pipeline) and (2) reason with entity state information (entity state information is shown in the gray box at the bottom right of the figure).

3.2. Dataset

Aiming for In-Context Learning, the CREPE dataset is constructed with an emphasis on the quality, objectivity, and diversity of contents and annotations. To ensure quality, all data are curated and annotated by students at the University of Pennsylvania via an extra credit assignment. To reduce subjectivity and increase topic diversity, the annotated dataset is inspected and modified by two other students to remove annotations that are subjective

```
(1) Put pineapple on a baking tray
(2) Preheat the oven to 350 degrees F by rotating the oven dial to 350
(3) Open the baking oven's door
(4) Put the baking tray into the baking oven
(5) Close the baking oven's door
(6) When finished, open the baking oven's door
(7) Take out the baking tray and move the baked pineapple to a plate
(8) (Optional) Close the door of the baking oven
```

Figure 6: A more detailed instruction for step 3 of the procedure in Figure 7.

and of similar topics.

Subjectivity is a major concern in the annotation as people tend to have slightly different living habits. For instance, consider the annotated procedure shown in Figure 7. The *imaginary event* is "*I can see the inside of the oven*" and a likelihood change is annotated to be "more likely" at step 3. First, the narrative of this procedure is concise to humans but vague to machines. While we humans can easily follow step 3 by using our commonsense knowledge, machines will struggle as "baking a pineapple" involves numerous un-mentioned sub-steps shown in Figure 6. In the causal reasoning chain, the event "see the inside of the oven" depends on the entity, "the oven's door", and its state, "open/close". Therefore, depending on one's living habits, it is reasonable to assume the "oven door" to be either "open" or "closed" after step 3 of the procedure. Hence, the annotation will vary between annotators due to this subjectivity.

With a high standard for the diversity and quality of the procedure, the resulting `CREPE` dataset contains 183 human-annotated, high-quality procedures that cover a wide range of topics (refer to Table 2 for detailed statistics). Of the 183 procedures, we use 42 procedures as a development set, based on which we construct prompt and conduct primitive studies. With the established pipeline, we then conduct testing runs based on the 141 testing samples, which are strictly held out during development.

Data Statistics			
	Dev	Test	Total
Num. procedures	42	141	183
Num. steps	295	924	1219
Num. event changes	144	180	324
Avg. step per procedure	7.0	6.6	6.7
Avg. token per step	6.8	6.8	6.8
Procedure Topics			
	Dev	Test	Total
Recipe	10	33	43
Household	12	40	52
Craft	4	17	21
Technology	5	19	24
Travel	4	4	8
Sports	2	13	15
Others	5	15	20

Table 2: Statistics of the `CREPE` dataset.

3.3. Baselines

We measured the performance of 2 simple baselines and 4 LLMs baselines on the `CREPE` datasets. Since the `CREPE` task is a ternary classification problem, our simple baselines are a *chance* baseline and a *majority* baseline. The *chance* baseline randomly assigns one of the "more likely", "less likely", "equally likely" labels and the *majority* baseline assigns the mode, which is "equally likely", to all predictions.

For the LLMs baseline, we selected models from two mainstream In-Context Learning architectures, namely *encoder-decoder* language models and *decoder-only* language models (see Figure 8). For the encoder-decoder model, we used the state-of-the-art at the time of this study, which are `T5` (3B) and `T0` (11B) [97, 102]. `T5`-3B is a prompt-based model with 3 billion parameters. Different from decoder-only models, the context of `T5` is encoded using the Transformer encoder, which introduces bi-directionality, and the decoder is the standard Transformer decoder. `T0`-11B is a larger version of `T5` with 11 billion parameters and it is finetuned on a large set of NLP tasks with natural language prompts. For decoder-only models, we used the GPT-3 family. For the general-purpose language model, we tested the following GPT-3 checkpoints: `text-curie-001`, `text-davinci-002`, `text-davinci-003`,

Goal: Bake a Pineapple
Steps:
1. Preheat oven to 350 degrees F (175 degrees C). Grease a 9x9 inch baking dish.

2. In a mixing bowl, mix together the pineapple, sugar, cornstarch, water, eggs, and vanilla. Pour the mixture into the prepared baking dish. Dot the mixture with butter and sprinkle with cinnamon.

3. Bake in a preheated 350 degrees F (175 degrees C) oven for 1 hour.

Event: I can see the inside of the oven

Annotation:
1. Unlikely 2. Unlikely 3. Likely

Figure 7: An example procedure annotated with an imaginary event and the corresponding likelihood of the event happening at each step.

and `ChatGPT`[1] Except for `text-curie-002` that has 13 billion parameters, all other checkpoints are GPT-3 model with 175 billion parameters. In addition, we also tested Codex, which is a code language model obtained by finetuning GPT-3 with code. The Codex checkpoint we used is `code-davinci-002`, which has 175 billion parameters and it is the InstructGPT model finetuned with codes scraped from GitHub [35].

	Naive		**Large Language Models**								**Human**
	Cha.	Maj.	T5	T0	GPT3C	GPT3C+S	GPT3D2	GPT3D3	ChatGPT	Codex	
(ours) Params	-	-	3B	11B	13B	13B	175B	175B	175B	175B	-
Dev	.262	.297	.343	.336	.346	.341	.350	.424	.470	**.585**	.868
Test	.251	.296	.343	.337	.356	.346	.533	.423	.462	**.591**	-

Table 3: Macro F1 of baseline models on the CREPE dataset. Human performance is not benchmarked on the test set as we strictly hold out its labels during all experiments. GPT3C represents the `text-curie-001` model. GPT3D2 represents the `text-davinci-002` model with an abnormal performance on the test set that we have confirmed but regrettably cannot explain. GPT3D3 represents the `text-davinci-003` model. GPT3C+S represents the GPT-3 `curie` model finetuned on StrategyQA. All of the above models work with textual prompts. Codex represents the `code-davinci-002` model and works with our proposed code-like prompts.

As shown in Table 3, the `CREPE` dataset poses a challenge to all existing LLMs including

[1]The official `ChatGPT` API is not available at the time of this study. Therefore, we used an unofficial API implementation from `https://github.com/acheong08/ChatGPT`

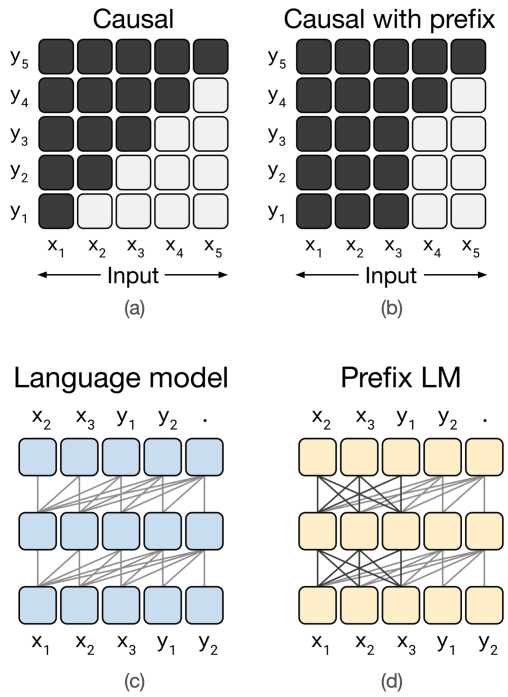

Figure 8: Comparison between the attention mask and the architecture of an Encoder-Decoder language model (figures (b) and (d)) in comparison between a Decoder-Only language model (figures (a) and (c)). This figure is adapted from the original T5 paper [97]

`ChatGPT`. More importantly, we see that there is a large performance gap between human-language language models, namely `T5`, `T0`, `GPT3`, and code-language models, namely `Codex` (see Figure 9 for the prompts used for `GPT3`, more details on code language prompt in Section 3.4). Further, while we see that while Reinforcement Learning with Human Feedback brings improvements to `text-davinci-003` and `ChatGPT` on this task, their performance is still not on par with code-language models trained without Reinforcement Learning (`code-davinci-002`) [87].

`CREPE` task can be thought of as a multi-hop reasoning problem, where, to reason about the change in the likelihood of an event, one needs to reason about the potential effects that the event brought to an entity state. From this perspective, we wish to examine the transfer learning capability of LLMs finetuned with existing multi-hop reasoning tasks. The dataset we used is the `StrategyQA` dataset [37]. In `StrategyQA`, given a multi-hop reasoning

question, a QA agent is responsible for first decomposing the multi-hop reasoning questions to single-hop questions, retrieving or generating answers to the single-hop questions, and producing the final answer by reasoning on the logical relationships between the answers to the sub-questions and the original multi-hop question. In this study, observed no performance gap between the `text-curie-001` and the `text-davinci-002` model, we opt to fine-tune the more economic `text-curie-001` model with the training set of `StrategyQA`. As shown in Table 3, the finetuning brings no performance gain to `text-curie-001`, further demonstrating the unique challenge that the `CREPE` dataset introduced.

```
Goal: Wash sneakers

Context: I remove shoelaces. I rinse.

Question: What is the likelihood that my feet get wet by wearing the sneakers?

Answer: likely
```

```
Goal: Wash sneakers
Context: I remove shoelaces. I rinse.
Question: What is the likelihood that my feet get wet by wearing the sneakers?
Answer: likely
```

Figure 9: The best-performing GPT-3 language prompt on the `CREPE` dataset.

3.4. Prompting Code Language Model

As shown in Section 3.3, in addition to traditional language models that output spoken language, we also attempted to use code language models. The attempt of using a code language model is enlightened by the previous work which shows that code language models are sometimes more capable of conducting reasoning tasks compared to traditional language models [75]. In this study, we use the most capable code language models, namely `Codex`.

`Codex` (`code-davinci-002`) is a checkpoint of GPT-3 that is pretrained with an integration of the original text datasets that are used to train the original GPT-3 with the addition of Python and Java code. Similar to the other GPT-3 models, `Codex` is an autoregressive language model that is capable of generating code either conditioning on a given chunk of

codes or from a doc-string. Therefore, to unleash the full power of `Codex`, we converted our original prompt (Figure 9) to the format of Python code (Figure 10).

To be specific, there are 4 key features of our code prompt.

Class Definition Each procedure is represented as a class object with the goal as the class name. In the example from Figure 10, the *goal* is "Wash Sneakers". Following the class definition is the *steps* of the procedure listed as comments.

Initial State One thing missing from the procedure is the initial environment. Without an initial environment, deducing the change in the likelihood of events is vague for the first step as it is unclear whether the model should treat the first step as the initial environment. Continue the "washing sneakers" example, it is unclear whether the event likelihood that "I can wear the sneakers and go out" will change in the first step without an initial environment. One can reason that the shoelaces are attached to the shoes before the first step and it is taken off afterward hence the likelihood decreases. One can also reason that the initial environment is defined by the first step where the shoelaces are detached from the shoe– hence the likelihood of the event does not change. To avoid this subjectivity, we additionally append an `Init` state before the first step that serves as an imaginary definition of the initial environment.

Function Definition Besides listing the steps beneath the class definition, each step is explicitly defined using a function definition. The content of the step is concatenated with an underscore to mimic the naming convention in Python. For instance, the step "remove shoelaces" will be defined as a function with the name "`remove_shoelaces`". A special case for this naming convention is the "Init" step. "Init" is defined as the "`__init__`" function of the class, in which the imaginary event is defined (see Figure 10).

Variables All the other components of the code prompt is represented as a variable. For instance, the imaginary event is "assigned" to `event0` by adding the narrative of the event as a comment following this variable:

Event Comment

```
self.event0 = event0 # My feet get wet by wearing the sneakers.
```

Further, we create some imaginary attributes to document the desired properties of a variable. For instance, deducing the change in the likelihood of an event is at the core of `CREPE`. Therefore, we assume that the variable `event0` has an imaginary attribute `change`, which documents the change in the likelihood of `event0`. To reiterate the content of the event, we again insert the event following the variable as a comment:

Event Change with Comment

```
self.event0.change = "more likely"
# My feet get wet by wearing the sneakers.
```

With all the above elements defining our code prompt, the overarching hypothesis we made here is that the structured representation (Python class object) can help with reasoning tasks like `CREPE`. The rationale behind our hypothesis is that procedural texts are semi-structured per se as they are oftentimes represented in an ordered list. Further, `CREPE` is asking for a change in the likelihood of events after every step, which demands a clear and concise representation of the relationship between steps (temporal). With all these design choices comes the final code prompt (Figure 10).

To study the effect of varying our formulation of the code prompt, we carried out an ablation study, which gradually moves from a Pythonic structured code prompt back to a free-form spoken language prompt.

3.5. Chain-of-Thought Reasoning in `CREPE`

Chain-of-Though (CoT) prompting methods have brought significant improvements in a range of reasoning tasks [120]. CoT exploits the reasoning capability of LLMs by requesting

```
class Wash_Sneakers:
  # Init
  # Remove shoelaces
  # Rinse
  def __init__(self, event0):
    self.event0 = event0 # My feet get wet by wearing the
    sneakers.
  def remove_shoelaces(self):
    self.event0.change = "equally likely"
    # My feet get wet by wearing the sneakers.
  def rinse(self):
    self.event0.change = "more likely"
    # My feet get wet by wearing the sneakers.
```

Figure 10: The best-performing Python code (bottom) representation of a procedure and hypothetical events from the CREPE dataset.

the model to explicitly generate the reasoning process. We attempted to adapt techniques and ideologies from the CoT prompting paradigm to the CREPE task by incorporating our bottleneck causal reasoning chain (Figure 4).

```
Goal: Wash sneakers
Context: I remove shoelaces. I rinse.
Question: What is the likelihood that my feet get wet by wearing the sneakers?
Answer: To get feet wet by wearing the sneakers, the sneakers must be wet. In the
given context, the sneakers are wet. Therefore, comparing to the previous step,
the likelihood change is ‘‘more likely”.
```

```
Goal: Wash sneakers
Context: I remove shoelaces. I rinse.
Question: What is the likelihood that my feet get wet by wearing the sneakers?
Follow up: Are the sneakers wet?
Intermediate answer: Yes
Follow up: Will my feet get wet by wearing wet sneakers?
Intermediate answer: Yes
Answer: likely
```

Figure 11: Our GPT-3 prompt with intermediate questions, mimicking the CoT prompt (top) and the Self-Ask prompt (bottom).

Overall, we attempted two types of CoT prompting methods with GPT3, namely the vanilla CoT prompt and the Self-Ask prompt [120, 93]. The vanilla CoT prompt simply adds the phrase “take it step by step” to the prompt, which magically triggers the LLMs to

autoregressively generate the reasoning process. The generated reasoning chain thus serves as a good context and leads to better generation (see Figure 11 for an example). `Self-Ask` improves upon the vanilla CoT prompt by eliciting LLMs to explicitly propose and answer reasoning questions. For instance, in the `Self-Ask` style prompt shown in Figure 11, to answer the question "What is the likelihood that my feet gets wet by wearing the sneakers?", LLMs need to first come up with a follow-up question regarding an entity state like "Are the sneakers wet?", which is concerned with the wetness (state) of sneakers (entity), and then another follow-up question regarding the logical relationship between the first question and the original event, "Will my feet get wet by wearing wet sneakers", which clearly has an entailment relationship (e.g. sneakers being wet $\rightarrow$ feet will be wet if wear sneakers). In this way, we incorporate our bottleneck model with the `Self-Ask` prompting method by explicitly asking the LLMs to first complete the causal chain by deducing some entity state change of interest and then reason about the causal relationship between the entity state (sneakers being wet) and the imaginary event (feet will be wet by wearing sneakers).

In addition to CoT with `GPT3`, we also attempted to incorporate CoT prompting with our novel code prompt by adding the following additional variables.

Entity Variable As shown in the earlier part of this section, we represent the imaginary event as a variable `event0`. To incorporate CoT prompting, we additionally add an "entity variable". Analogous to other variables in our code prompt, we first ask the code language model to initialize an entity variable in the `__init__` function. Continuing with the "Wash Sneakers" example where the imaginary event is "my feet get wet by wearing the sneakers.", a code language model shall generate the following init function:

Code Prompt Init (Hard-Variable)

```
def __init__(self, event0):
    self.sneakers = Sneakers()
    self.event0 = event0
    # My feet get wet by wearing the sneakers.
```

In the init function, a code language model creates another imaginary object, which is named after the entity of interest (`Sneakers()` in this case). Following the initialization of the entity, the language model is to first predict the state change of the entity and then deduce the event likelihood change. Hence, the generation of each step of the procedure becomes:

Code Prompt (Hard-Variable)

```
def rinse(self):
    self.sneakers.wet = True
    self.event0.change = "more likely"
    # My feet get wet by wearing the sneakers.
```

where the state of the entity is represented as an imaginary attribute of the entity variable (`self.sneakers.wet`) and the state change is represented with a simple boolean variable. This pipeline of generating answers, again, is in line with our bottleneck causal chain. We refer to this prompt where the entity state is explicitly encoded as variables as the **hard-variable prompt**

In addition to the above representation where the entity state is encoded as an imaginary attribute, we also attempted an alternative prompt where the entity state is thought of as a "subevent" and the explicit content of the entity state is represented as a comment. We refer to this way of representing the entity state information as **soft-variable prompt**. In

this case, the `__init__` function becomes:

Code Prompt Init (Soft-Variable)

```
def __init__(self, event0, subevent0):
    self.event0 = event0
    # My feet get wet by wearing the sneakers.
    self.event0.subevent = subevent0
    # The sneakers are wet
```

subsequently, the generation for each step of the procedure becomes:

Code Prompt (Soft-Variable)

```
def rinse(self):
    self.event0.subevent.change = "more likely"
    # The sneakers are wet
    self.event0.change = "more likely"
    # My feet get wet by wearing the sneakers.
```

	Naive	**LLMs**		**CoT Large Language Models**				**Human**
	Majority	GPT-3	Codex	GPT-3 + CoT	GPT-3+self-ask	Codex soft(ours)	Codex hard(ours)	
Dev	.297	.346	.585	0.359	.342	.624	**.667**	.868
Test	.296	.356	.591	0.379	.345	**.626**	.609	-

Table 4: Macro F1 of chain-of-thought models on the CREPE dataset. GPT-3 + CoT—self-ask represents the `text-davinci-002` model prompted with the CoT or self-ask style prompt.

3.6. Discussion

The performance of CoT prompting is shown in Table 4. Surprisingly, the CoT prompting does not bring any performance gain for GPT-3 on the `CREPE` tasks as all GPT-3 models score roughly similar to the regular prompt. We suspect that this is largely due to GPT-3 does not understand the bottleneck model. In other words, GPT-3 has limited capabil-

ity of building a causal chain leveraging the fine-grained entity state information through generating spoken language.

To our surprise, CoT prompting methods brought significant performance boost to `Codex`– especially `Codex` with **hard-variable prompt**. This result shows that, depending on the nature of the reasoning task, having a structured representation (code) could be beneficial and significantly outperforms the standard spoken-language prompt. In addition, contrary to the observation from GPT-3 result where the model does not know how to leverage entity state information to form a causal chain, `Codex` has a much better grasp of this intention and demonstrated that fine-grained information such as entity states could bring significant help to causal reasoning between events.

To elicit how CoT prompting works with `Codex`, here we provide a detailed explanation with examples. To have a better understanding of CoT with `CREPE`, first recall that the `CREPE` is a ternary classification task where the label space is {"more likely", "less likely", "equally likely"}. The majority label is "equally likely". This is because the world we live in is very sparse– when some event happens, the state of most objects remains unchanged. Likewise, the likelihood of some event happening is rarely changed by other events. Take the event of "playing basketball" as an example, most other events related to the player such as "doing homework", "feeding cats", "having dinner", and so on are not effected by the aforementioned event. Using this sparsity, we hypothesize that, for one event to affect the likelihood of another event, there must be an entity state change that connects the two events. This hypothesis can be thought of as an application of our bottleneck model (Figure 4). Therefore, there will be two scenarios in our code prompt. One prevalent scenario is that the current events do not alter the likelihood of another event ("equally likely"). In this case, the model shall not predict any entity state change and the generation result looks like

Code Prompt with no Entity State Change

```
def remove_shoelaces(self):
    self.event0.change = "equally likely"
    # My feet get wet by wearing the sneakers
```

The other scenario is when the current events in the step change the likelihood of another event. In this case, there must be some mutual entity state that is caused by the step, which in turn causes the change in the likelihood of another event. Here, the model shall first deduce the entity state change caused by the given step and then generate how this entity state change could affect the likelihood of the imaginary event

Code Prompt with CoT

```
def rinse_the_shoes(self):
    self.sneakers.wet = True
    self.event0.change = "more likely"
    # My feet get wet by wearing the sneakers.
```

To summarize, when the likelihood of the imaginary event does not change, we expect `Codex` to produce no entity state change. On the other hand, when the likelihood changes, we expect `Codex` to first produce some entity state change and then deduce how the likelihood of the imaginary event would change.

There are 3 variables in this CoT process:

1. Can `Codex` predict the entity state change at steps where the likelihood of the imaginary event changes?

2. How relevant are the predicted entity state?

3. Are the entity state change correctly predicted?

It turns out that the gap between our best performing CoT `Codex` (0.67) and human performance (0.87) is due to the first factor– We observed that there are many cases where there is a change in the likelihood of the imaginary event but `Codex` did not produce any entity state change to construct its chain of thought. For the second factor, 71/74 (96%) generated entities are mutually relevant to both the step and the imaginary event, which shows that `Codex` is competent at completing the causal chain. For the third factor, we computed the F1 score exclusively for `Codex` results that generated entity state change. In this case, the F1 score of `Codex` reaches 0.85, which is very close to human performance. Therefore, a future direction of tackling the `CREPE` task is to study ways to address the first factor. In other words, current language models are lacking in understanding the causal relationship between events and entities. Hence, the entity state changes are oftentimes ignored and the CoT style reasoning is broken in the first stage of reasoning. We suspect that this event-entity causal reasoning task is particularly challenging due to reporting bias. In practice, the steps of a procedure typically contain only high-level events. The causal relation between events and participating entities is what we perceive as commonsense knowledge and is rarely mentioned in the text. Trained with texts that contain very little fine-grained commonsense knowledge such as entity state changes, it makes intuitive sense that LLMs suffer with the event-entity reasoning tasks. In the following sections, I attempted two approaches to mitigate this issue– one with the aid of external modality (Chapter 4) and another with a newly proposed fine-grained causal reasoning task (Chapter 5)

CHAPTER 4 : Mitigate Reporting Bias with External Modalities

4.1. Introduction

In Chapter 3, we see that one of the main challenges that LLMs face is the incapability of doing fine-grained reasoning tasks. This is largely due to the reporting bias in its pretraining data. In this study, the specific reporting bias we are concerned with is the **fine-grained commonsense causal knowledge**. Continuing with the "Wash Sneakers" example, a typical initial step would be "First, remove shoelaces". While semantic information might be scarce, much of the causal information can be easily and sufficiently deduced by humans using commonsense causal knowledge. On a high level, it is obvious to humans that the action of "removing shoelaces" is a direct effect of the goal "wash sneakers" ([wash sneakers] $\rightarrow$ [remove shoelaces]). While each step might contain a limited amount of high-level event-event causal relationships, there are bountiful fine-grained causal relationships. For instance, here is a list of (entity, pre-state, post-state) tuples that are causally related to the step, "remove shoelaces"

- remove shoelaces $\rightarrow$ (shoelace, attached to the shoe, detached from the shoe)
- remove shoelaces $\rightarrow$ (shoelace, twisted, expanded)
- remove shoelaces $\rightarrow$ (shoe, tied, untied)
- remove shoelaces $\rightarrow$ (shoe, worn, unworn)
- remove shoelaces $\rightarrow$ (shoe, normal weight, lighter)
- remove shoelaces $\rightarrow$ (tongue, covered by shoelaces, uncovered)
- remove shoelaces $\rightarrow$ (eyelets, filled by shoelaces, empty)

These causal relationships between event and entity states are oftentimes ignored in writing, resulting in reporting bias. In this study, we aim to address the issue that some of the entities that are causally related to the current step are ignored in the narrative. In the

simple example above, the entities explicitly mentioned in the step "remove shoelaces" are

- {shoelaces}

and the unmentioned causally related entities are

- {shoe, tongue, eyelets}

In other words, in this simple example, 75% of the affected entities are implicit in written texts. Trained solely with written text, LLMs are likely to be severely affected by this reporting bias, which will give them a hard time conducting fine-grained reasoning tasks like deducing the causal relationship between events and entities.

As mentioned in Section 1.5, there have been extensive studies that aim to model the causal relationship between high-level events and low-level entity states. These tasks are often conducted with the umbrella term "entity state tracking". In this study, our main objective is to investigate if explicitly using external modalities such as videos and audio transcriptions can mitigate the effect of implicit commonsense knowledge in written text. Different from the implicit use of external modalities in the `OpenPI` dataset where the authors provided images from wikiHow to aid annotators with annotating entities that are implicit in the written procedure, we explicitly involve external modalities such as videos, images, and audio transcriptions to enrich the entity set.

4.2. Dataset

To have a fair comparison with the `OpenPI` dataset, which is based on procedural texts, we use the `YouCook2` dataset [144]. The two datasets are very similar in the narrative of the procedure. The only difference is that `OpenPI` is a uni-modal and open-domain entity state tracking dataset whereas `YouCook2` is a multimodal object detection dataset that focuses on recipes. In this study, we only use the procedural texts and the instruction videos in the `YouCook2` dataset to conduct multimodal entity extraction. Therefore, the results from the `YouCook2` dataset, which cover a subset of topics from the `OpenPI` dataset, can be

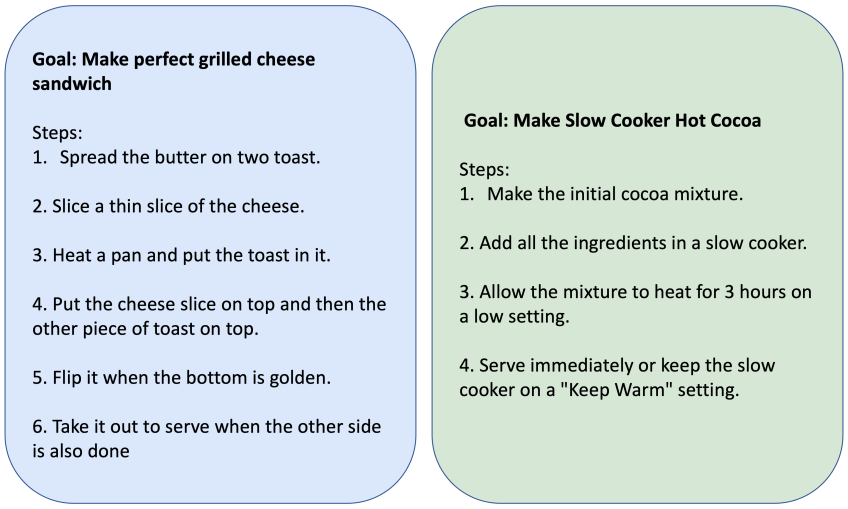

Figure 12: A comparison between procedure examples from `YouCook2` and `OpenPI`. The procedure on the left is an example from `YouCook2` whereas the procedure on the right is an example from `OpenPI`. The two datasets are very similar in the narrative of the procedure. The only difference is that `OpenPI` is a uni-modal and open-domain entity state tracking dataset whereas `YouCook2` is a multimodal object detection dataset.

generalized to procedures in the `OpenPI` dataset.

The original `YouCook2` dataset contains 175 hours of videos on 89 different cooking recipes scraped from YouTube[1]. Each recipe video is segmented based on the step of the recipe. The annotation contains the start and end time stamps of the segment as well as a description of the current step. To limit the size of this study and to compensate for the computational resources available, we take a subset of the `YouCook2` dataset which contains 22 distinct recipe videos. Unlike procedures, which contain a *goal* and a series of *steps*, the `YouCook2` dataset does not contain information on the *goal*. Therefore, the video title is scraped and used as the goal for each recipe. The resulting `YouCook2` cooking recipe looks analogous to the cooking procedures from `OpenPI` (Figure 12).

The resulting subset contains 22 procedures, 149 steps, and 846 human-annotated entities. On average, each step is annotated with 5.68 participating entities. The largest amount of

[1] `https://www.youtube.com/`

entities annotated for one step is 13 entities where as the least amount of entities annotated for a single step is 2 entities. Inevitably, the annotations are vulnerable to subjectivity. We make our best effort to mitigate these risks by having multiple annotators annotate the same dataset and aggregate the final results. Further, annotating entities in a step is heavily context-dependent as the annotator oftentimes needs to know the state of the world before and after the current step. For instance, in a recipe for "Making Coffee", the entities that participated in the step, "grind the coffee beans", are dependent on the next step– If the next step is "pour hot water into the cup", then we can deduce that the coffee beans have been transferred from the grinding machine to a mug. On the other hand, if the next step is "pour grounded coffee beans into a mug", then we can deduce that the grounded coffee beans are still in the grinding machine at the current step. To better address the demand for contextual information, we first ask annotators to read through the whole procedure and then start annotating for each step. Below is an example of the annotation following the above pipeline:

Annotation Example (Implicit Entities are Bolded)

Goal: How To Make Pierogi

Steps:

1. Combine flour and salt in a bowl.
2. Add two eggs into water and mix it well.
3. ......

Participating Entities in Each Step:

1. ["flour", "salt", "bowl", **"whisk"**, **"counter"**, **"salt bag"**, **"flour bag"**]
2. ["egg", "water", **"bowl"**, **"whisk"**, **"egg shell"**, **"tap"**, **"trash can"**]
3. ......

4.3. Models and Approaches

First, we simplify the task of object detection from video to image object detection by discretizing videos into frames (still images). Specifically, we set the sampling rate to be one image per second as we assume that salient events in the procedure will at least span several seconds in the video. After this simplification, we are able to deploy a large amount of object detection and other general-purpose text-image models to this study. With the above naive sampling strategy, there are be a large amount of redundant information as the contiguous sampled frames will likely be capturing similar scenes, thus containing the same set of objects. While such redundancy is acceptable for moderate-sized models, it severely impacts the computational efficiency when we apply large-scale models to this task. Therefore, we deal with large-scale models, we additionally eliminated the redundancy leveraging the Structured Similarity Index between contiguous frames [116, 117] (Figure 13).

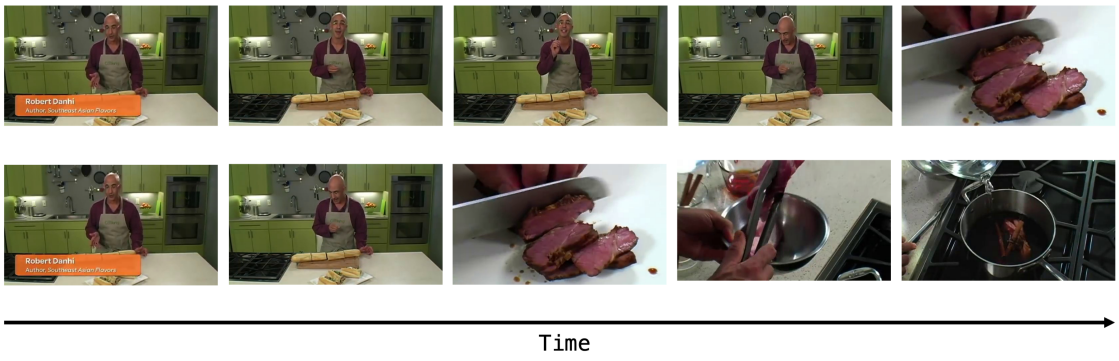

Figure 13: Sample frames extracted from the YouCook2 dataset. Pictures in the first row are sampled every second, leading to a large amount of repetitive content. Pictures in the bottom role are further filtered using Structured Similarity Index, resulting in a set of images that contain distinct semantic information

Recall that the purpose of incorporating visual information is to help language models to deduce implicit participating entities and mitigate reporting bias. Therefore, we are essentially tackling an object detection task. The only nuance is that the context information (the content of the procedure) is crucial as we only want entities that participated in the given step. For example, since we are exclusively targeting recipes, we oftentimes see objects such as refrigerators and stoves in the video. These objects, however, do not always

participate in the current step. To this extent, we attempted 3 approaches to extracting entities.

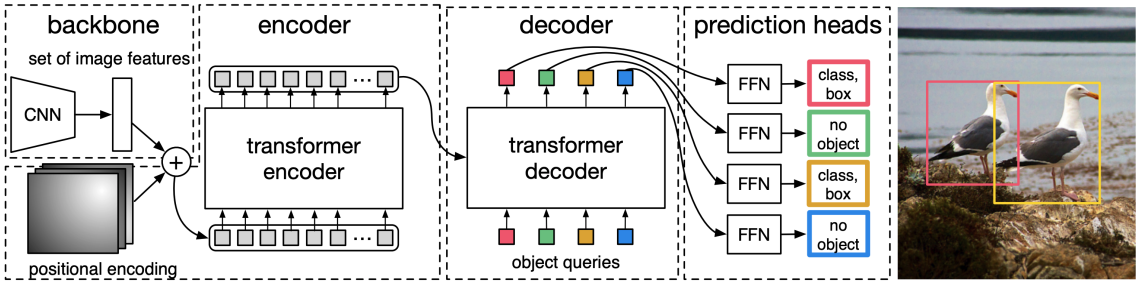

Figure 14: The DETR model architecture. Figure adopted from the original DETR paper [16].

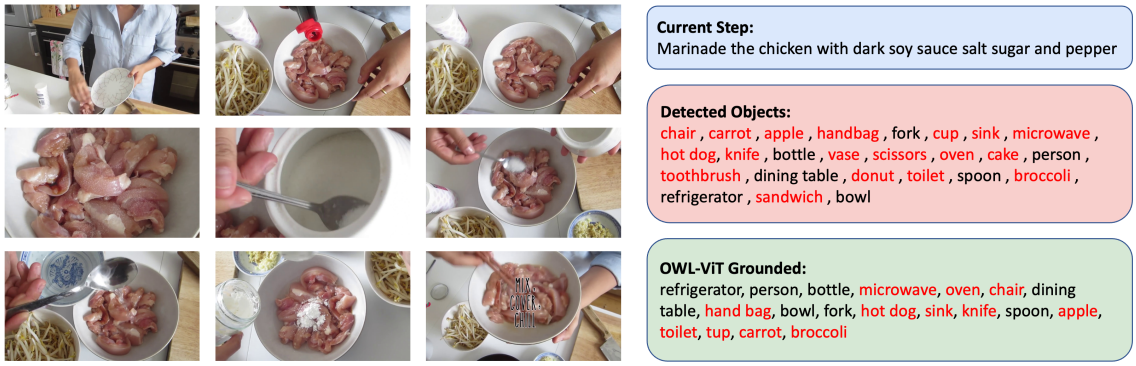

Figure 15: An example of the Object Detection + CLIP-Based Filtering approach. On the left are the sample frames that correspond to this step. On the right are the step narrative and model outputs. The top block displays the narrative for the current step, the middle block shows the raw objects detected by the ConditionalDETR model. The bottom block shows the objects grounded by the OWL-ViT model. The objects that are mistakenly detected and grounded are colored in red.

Object Detection with DEtection TRansformer (DETR) [16]. DETR is a Transformer-based object detection model (see Figure 14). Specifically, we employed the Conditional DETR which is more efficient in training and achieves better generalization capability compared to the DETR model of similar size [78]. Based on the output confidence score of the entity, we do the first round of filtering where we discarded objects with low (< 0.2) confidence scores. This first round of filtering is a lenient way with the intention to only filter out the obvious errors. After obtaining a list of candidate entities, we took two attempts

to further filter entities that are unrelated to the current step.

- **CLIP-Based Filtering** In this filtering approach, we used a Transformer-based open-vocabulary object detection model trained with contrastive learning as seen in CLIP, *Vision Transformer for Open-World Localization* (OWL-ViT) [80, 94]. On a high level, the vision input is encoded with a Vision Transformer, and the text is encoded with a Trasnformer-encoder model. The encodings are aligned with contrastive learning as seen in the original CLIP [94] (Figure 17). The OWL-ViT model achieves open-vocabulary object detection by allowing users to input the object candidate as a text prompt. In our case, the text prompt is replaced with the entity state list from the DETR model. Since the OWL-ViT model outputs a probability distribution on the given entity candidates, we set the threshold to 0.1 and discarded the other entities (Figure 15).

- **LLM-based Filtering** In this filtering approach, we used LLMs, namely `Codex` to help with filtering participating entities (Figure 16). The main advantage of LLM-based filtering is that we are able to incorporate context information (goal and steps of the procedure), which is crucial to this object detection task. With this in mind, we formulate the following code prompt template:

Code Prompt for Filtering

```
goal = {goal}
previous_steps =  {prev_steps}
current_step = {cur_step}
candidate_objects = {candidate_entities}
involved_objects = {involved_entities}
```

Image Captioning with general-purpose, unified models. In this case, we used the OFA model [115]. Different from the previous Object Detection approach where we put emphasis on the breadth of entities that the model can capture, here we emphasize the saliency of

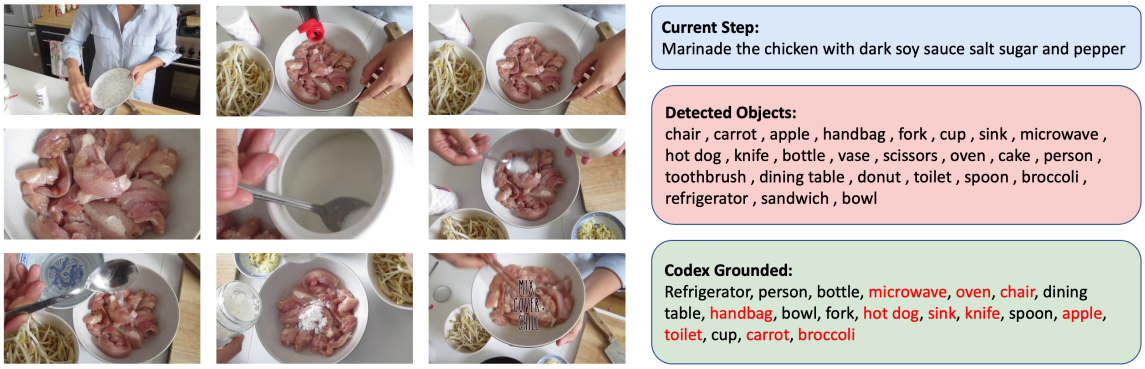

Figure 16: An example of the Object Detection + Codex-Based Filtering approach. On the left are the sample frames that correspond to this step. On the right are the step narrative and model outputs. The top block displays the narrative for the current step, the middle block shows the raw objects detected by the ConditionalDETR model. The bottom block shows the objects grounded by prompting the Codex model. The objects that are mistakenly detected and grounded are colored in red.

the entity. Concretely, we hypothesize that objects that are salient in the video may not necessarily be salient in the step. For instance, "human", a salient object in the video, is less relevant to understanding the events of the current step as we assume that all the recipe videos are made by some human cook and that the state of the human does not have a big impact on the content of the procedure. On the other hand, kitchen gadgets, which may only occupy a small portion of the video, are oftentimes an integral part of the step. Therefore, we wish the model to output only the most salient objects in the current step. One possible approach to doing so is to conduct image captioning, where the model narrative is the most salient event in the current step. After obtaining the image caption, we then use the off-the-shelf Stanford CoreNLP parser [77] to get the Part-of-Speech tags, based on which we then extracted all the noun phrases in the caption (see Figure 18).

Visual Question Answering with general-purpose, unified models. In this case, we used the Unified-IO model [72]. The ideology of this approach is similar to that of Image Captioning. Since it is implausible for any existing model to give a complete and reasonable set of objects in a step of a procedure, we opt to ask the model to provide only the salient objects. In Visual Question Answering, we have more control over the behavior of the

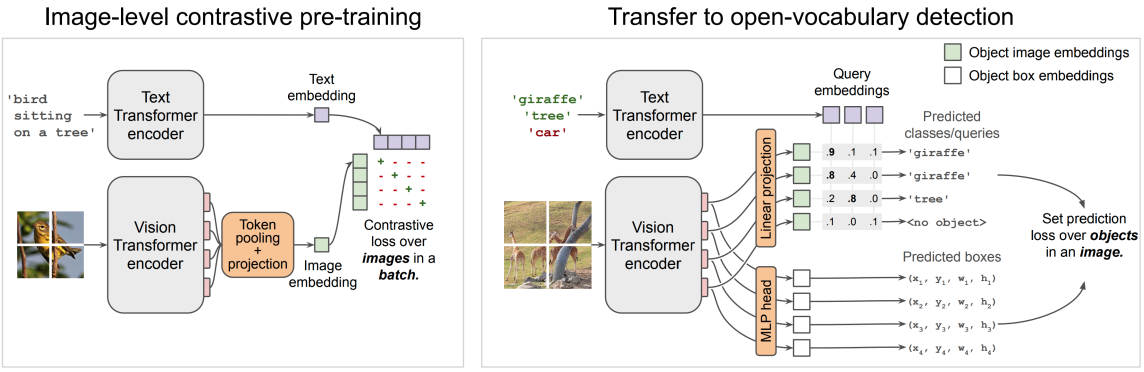

Figure 17: An illustration of the pre-training and inference of the OWL-ViT model. This figure is adapted from the original OWL-ViT paper [80].

model by adjusting the wording of the textual question. In this study, the question we ask Unified-IO is "What are the important objects in the scene?". Analogous to the Image Captioning approach, we run the Stanford CoreNLP parser on the generated answer to extract all the noun phrases and treat it as the list of salient objects (see Figure 19).

4.4. Result and Discussion

To measure the effectiveness of using visual information to mitigate reporting bias in recipes, we computed the coverage probability of the model-deduced entities with respect to human annotations. Our preliminary experiment shows that vision alone cannot be used to deduce participating entities in a step due to the large difference in the reasoning capabilities between vision and language models. Therefore, we use the language-model-only result as the baseline and investigate how much improvement visual information could bring. For the language model baseline, we used a GPT-3[2] model finetuned with the OpenPI dataset. We acknowledge that the annotation of the OpenPI dataset also involves multi-modality where the annotators are provided images from wikiHow to assist their entity state annotation [108]. Therefore, our OpenPI finetuned GPT-3 may also pick up some vision information by exploiting the association between steps and the OpenPI annotations.

[2]We fine-tuned the `davinci` model, which is the original GPT-3 release without any instruction fine-tuning.

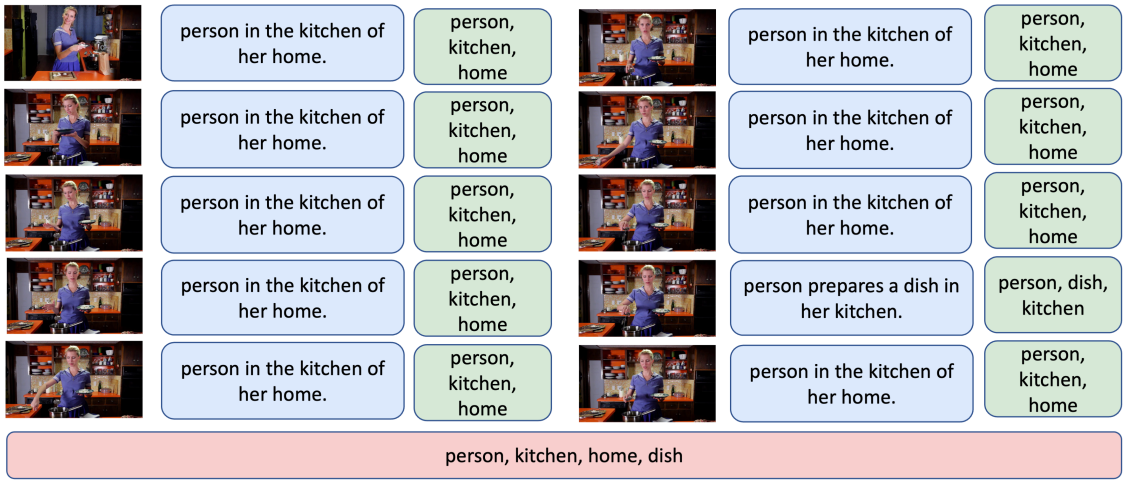

Figure 18: An example of the image captioning pipeline. The left column shows the original input image to the OFA model. The middle column shows the caption that the OFA model outputs for the given image. The right column displays the objects extracted from the image caption. The bottom block shows all objects obtained from the image captioning.

Table 5 show the coverage probability of different methods. The main conclusion is that current vision models are not capable of mitigating the reporting bias in language. On the contrary, they may bring more noise by producing entities that do not participate in the event (see "False Positive Count" in Table 5).

Method	Coverage	False Positive Count
OpenPI GPT-3	0.86	3.34
OpenPI GPT-3 with Object Detection	0.86	17.56
OpenPI GPT-3 with OFA caption	0.86	11.04
OpenPI GPT-3 with Unified-IO VQA	0.86	7.72
OpenPI GPT-3 with Audio Transcription	0.86	12.92

Table 5: The entity coverage probability of each method compared to human annotation. All methods are based on GPT-3 annotation, where we compute coverage probability by adding extra entities obtained from vision to existing GPT-3 annotations. The false positive count shows how many entities are there in the model-generated results that are not annotated by humans. Less "False Positive Count" means that the model is more precise and efficient at predicting participating entities.

From the sample output examples from the three different approaches (Figure 15, Figure 16, Figure 18, Figure 19), we can see the difficulty of this task. First of all, the recipe videos do not always have close-up shots. Take the images in Figure 18 as an example, the video

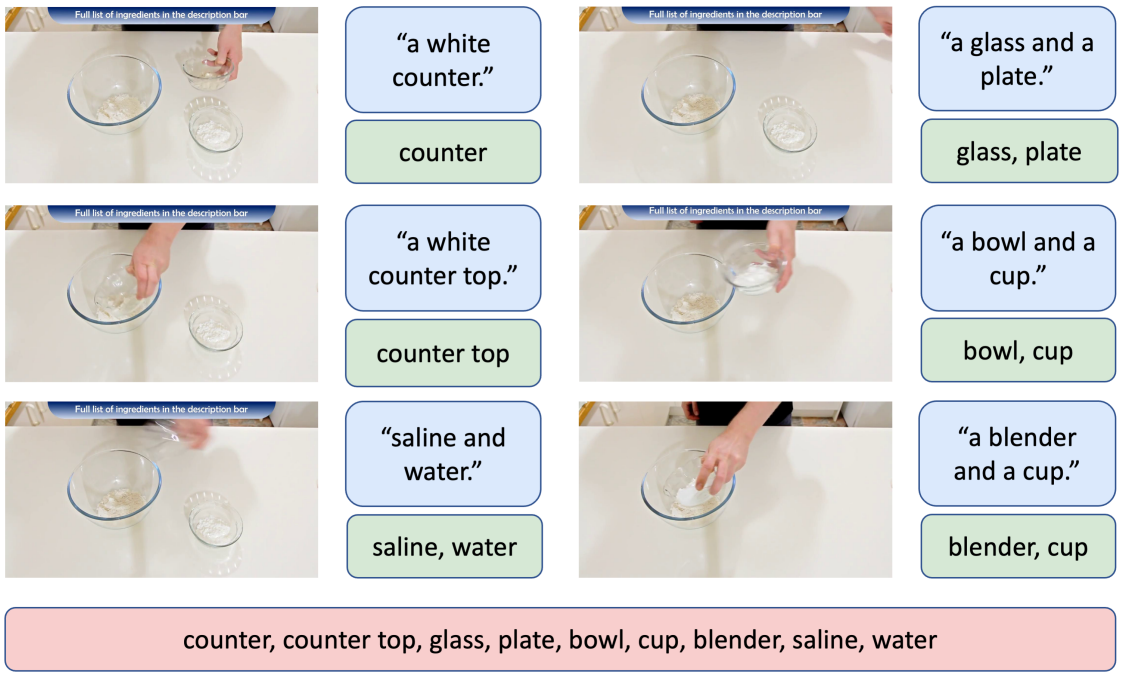

Figure 19: An example of the VQA pipeline. On the left column are the original input images. The upper right column displays the answer to the question, "What are the important objects in the scene?", from the Unified-IO model. The lower right column shows the entities extracted from the answer. The block on the bottom is all the entities extracted for the current step.

oftentimes focuses on the cook herself, instead of the equipment or ingredients. Therefore, when asking for the caption of the image, *"person in the kitchen of her home"* is an unwanted but reasonable description of the images. This leads to the second challenge in using visual cues to mitigate reporting bias– lack of concise context information. For the previous example where there is no close-up shot of the cooking instrument or ingredients that the cook is using, it is hard for a human to deduce participating entities without context information. Even for the cases of close-up shots like the ones in Figure 19, it is still hard for us to tell what are the ingredients in the bowls– they could be flour, salt, sugar, MSG, etc. Therefore, to be able to deduce the ingredients in the bowl, one must retrospectively access the previous context to see where the ingredients come from and ideally be able to read the text on the package. Therefore, the straightforward remedy to this lack of context information is to build a multimodal pipeline that can understand both context information

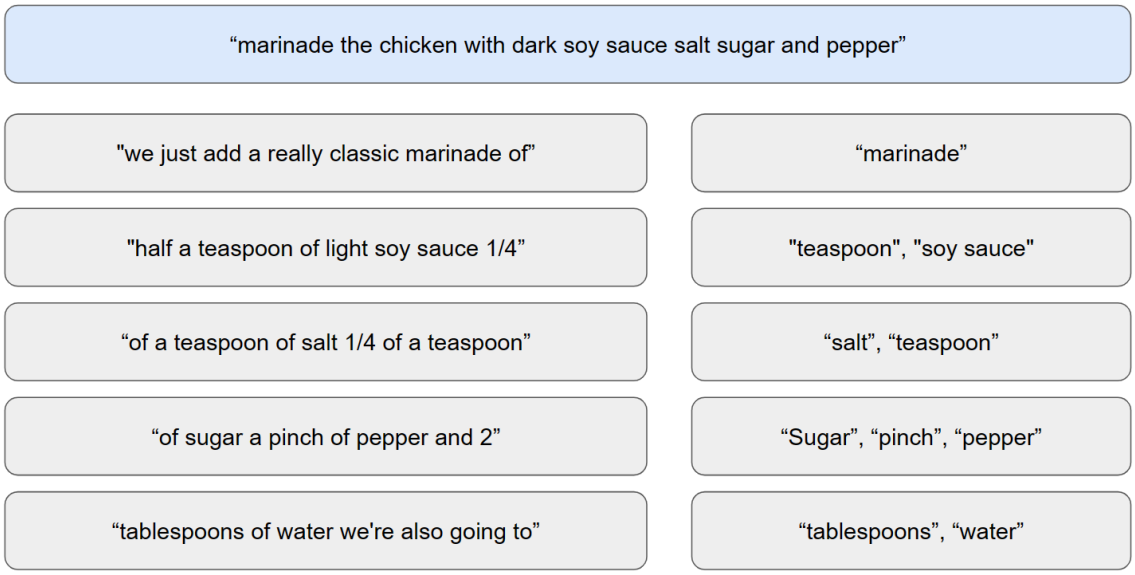

Figure 20: An example of audio transcriptions and the entities extracted from the transcription. The top blue box shows the human annotation from the `YouCook2` dataset and all the boxes below shows the audio transcription and the extracted entities from each corresponding transcription.

provided as texts and the visual cues provided as video/images.

For audio transcription, since the `Youcook2` data is curated using cooking videos from YouTube, we conveniently leveraged the `YouTube Transcript API`[3] to scrape the official audio transcriptions provided by YouTube. We then applied an off-the-shelf Part-of-Speech Tagging model[4] to extract the entities from the transcriptions. From the example shown in Figure 20, we see that the audio transcription is a noisier version of the written annotation– the only implicit participating entity that audio transcription introduced is “teaspoon” whereas entities such as “marinade” and “pinch” are wrong entities with regard to the current step. The same conclusion can be reached from the result shown in Table 5– The entities extracted from audio transcription does not contribute to the coverage probability.

For a detailed description of the top-5 most frequently missed/deduced entities predicted

[3]https://github.com/jdepoix/youtube-transcript-api

[4]We used the Stanford CoreNLP package to obtain the POS tags. `https://stanfordnlp.github.io/stanza/corenlp_client.html`

Approach	Entity
OpenPI GPT-3	(ingredients, mixture, bowl, pan, food)
OpenPI GPT-3 + Object Detection	(spoon, fork, cup, cake, oven)
OpenPI GPT-3 + OFA Caption	(food, ingredients, mixture, bowl, dish)
OpenPI GPT-3 + Unified-IO VQA	(spoon, ingredients, bowl, mixture, pan)
OpenPI GPT-3 + Audio Transcription	(ingredients, mixture, bowl, oil, pan)
OpenPI GPT-3	(bowl, pan, water, pot, stove)
OpenPI GPT-3 + Object Detection	(bowl, pan, water, pot, knife)
OpenPI GPT-3 + OFA Caption	(food, ingredients, mixture, bowl, dish)
OpenPI GPT-3 + Unified-IO VQA	(pan, bowl, oil, chickpeas, chicken)
OpenPI GPT-3 + Audio Transcription	(bowl, pan, water, oil, chickpeas)

Table 6: On the top half of the table are the top-5 missed entities by count using different approaches. The `Entity` column shows the top-5 missed entities names. On the bottom half of the table are the top-5 covered entities by count using different approaches.

by each approach, please refer to Table 6. We were able to identify some common patterns across different approaches. All approaches are having a hard time deducing *ingredients, mixture, bowl*, and *pan* from the step. While misses on entities such as *ingredients* and *mixture* are comprehensible as the object that these two entity names referring to is vague, misses on *bowl* and *pan* show that current multimodal models are incapable of identifying participating entities in a scene. Notice that entities that commonly appear in recipes such as *bowl, pan*, and *ingredients* exist both among the top-5 missed and deduced entities. This further demonstrate that external modalities such as vision and audio are incapable of tracking entities due to the absent of common sense knowledge and the flexibility of taking the context of the procedure into consideration.

CHAPTER 5 : Fine-Grained Causal Inference with Entitiy States

5.1. Introduction

From Chapter 3, the `CREPE` benchmark and the Chain-of-Though code prompt demonstrated the significant role that fine-grained causal relationship between events and entity states plays in the causal reasoning between high-level events. As mentioned in previous chapters, information on fine-grained causal relationships are largely absent from written texts as they are considered commonsense knowledge, which is the knowledge that most people comprehend without explicitly mentioning. In Chapter 4, we tackled the issue of reporting bias in the training dataset of LLMs. We showed that the capability of current multimodal models is insufficient for deducing implicit participating entities from modalities other than texts. Therefore, in this chapter, we aim to investigate how LLMs perform in fine-grained causal reasoning with entity states.

To better demonstrate the fine-grained causal inference between entities, consider the following example: In a procedure of "*doing laundry*", when asking LLMs *"can I open the washing machine door?"* after observing that *"the laundry is finished"*, a proper intermediate reasoning question would be *"Does finishing laundry result in washing machine door unlock?"*. Essentially, the previous question is asking for the causal relationship between two entities and their states. Suppose we represent the entity states with (`entity`, `state`) tuples, the question can then be formulated as "*Does (laundry, finished) cause (washing machine door, open)*?".

In this chapter, we introduce the **C**ommonsense **C**ausal **R**easoning about **Entity States** (`C2RES`) benchmark for causal inference between entity states. We perceive entity states as the fundamental unit of events. From this perspective, an event is composed of multiple entity state changes, and an event is a high-level, concise description of the aggregated effect of entity state changes that occurred in a time frame. For example, consider the event

> "*Sear the steak until both sides are golden.*"

The entity state change tuples that one can derive from this event are

(oil bottle, open)	(oil bottle, up-side-down)	(oil bottle, lighter)
(oil, in pan)	(oil, heated)	(pan heated)
(pan, greasy)	(stove dial, rotated)	(stove, heated)
(pan, on top of the stove)	(steak, in pan)	(steak, heated)
(steak, seared)	(sizzling sound, heard)	

Table 7: Entity state changes that can be deduced from the event "*Sear the steak until both sides are golden.*".

In the C2RES framework, we build on the idea of considering entity states as the unit of events and propose the task of deducing the causal relationships between the entity state changes that happened in the current step, which could contain multiple events. Take entity states listed in Table 7 as an example, some causal relationships are:

- (stove dial, rotated) $\rightarrow$ (stove, heated)
- (oil bottle, open) $\wedge$ (oil bottle, up-side-down) $\rightarrow$ (oil, in pan)
- (pan, on top of stove) $\wedge$ (stove, heated) $\rightarrow$ (pan, heated)
- (oil, in pan) $\rightarrow$ (oil bottle, lighter)
- (oil, in pan) $\rightarrow$ (pan, greasy)
- (oil, in pan) $\wedge$ (pan, heated) $\rightarrow$ (sizzling sound, heard)
- (oil, in pan) $\wedge$ (pan, heated) $\rightarrow$ (oil, heated)
- (pan heated) $\wedge$ (steak, in pan) $\rightarrow$ (steak, heated)
- (steak, heated) $\rightarrow$ (steak, seared)

As a by-product of inferring the causal relationship between entity states, a causal diagram can be constructed for each step of the procedure (Figure 21).

Therefore, with a reliable causal reasoning agent that can accurately predict fine-grained

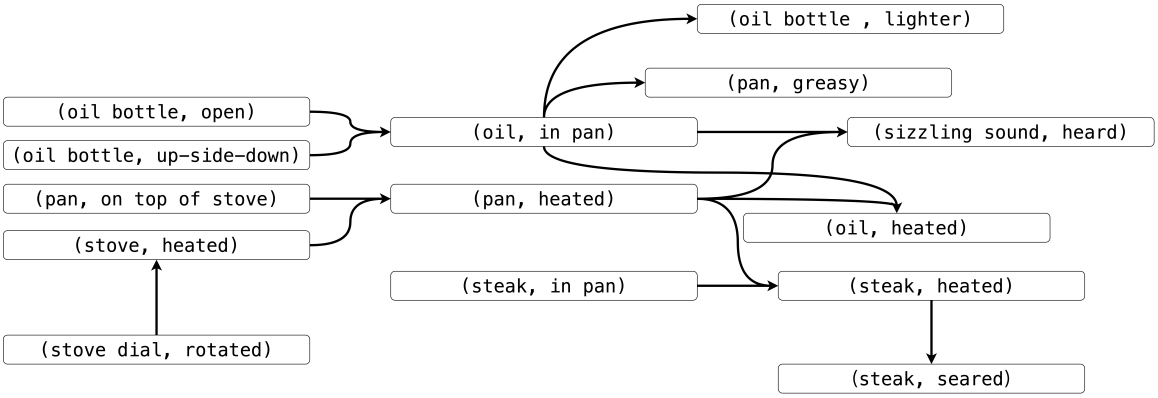

Figure 21: A sample causal diagram constructed from the causal relationships listed above. The direction of the edge represents the direction of the effect.

causal relationships between entity states, one can conveniently construct a dynamic causal diagram throughout the procedure, which can be used to study the interpretability and faithfulness of the neural reasoning models and pipelines.

5.2. Dataset

The C2RES dataset is constructed based on a subset of cooking procedures available in the YouCook2 dataset. Here is a sample cooking instruction from YouCook2:

Procedure Example from YouCook2

Goal: Garlic Steamed Mussels Recipe

Steps:

1. Rinse the mussels in water.
2. Add the onion to the pot.
3. Add the wine and the mussel.
4. Cover the pot.
5. Garnish with green onion.

To come up with the potential entity state changes in each step of the procedure, we leverage the OpenPI dataset, which consists of procedures with human-annotated entity state changes. As an open-domain entity state tracking dataset, there are bountiful cooking instructions in the OpenPI dataset, which matches the distribution of texts in the YouCook2 dataset. Furthermore, the annotators of the OpenPI dataset are also provided with images

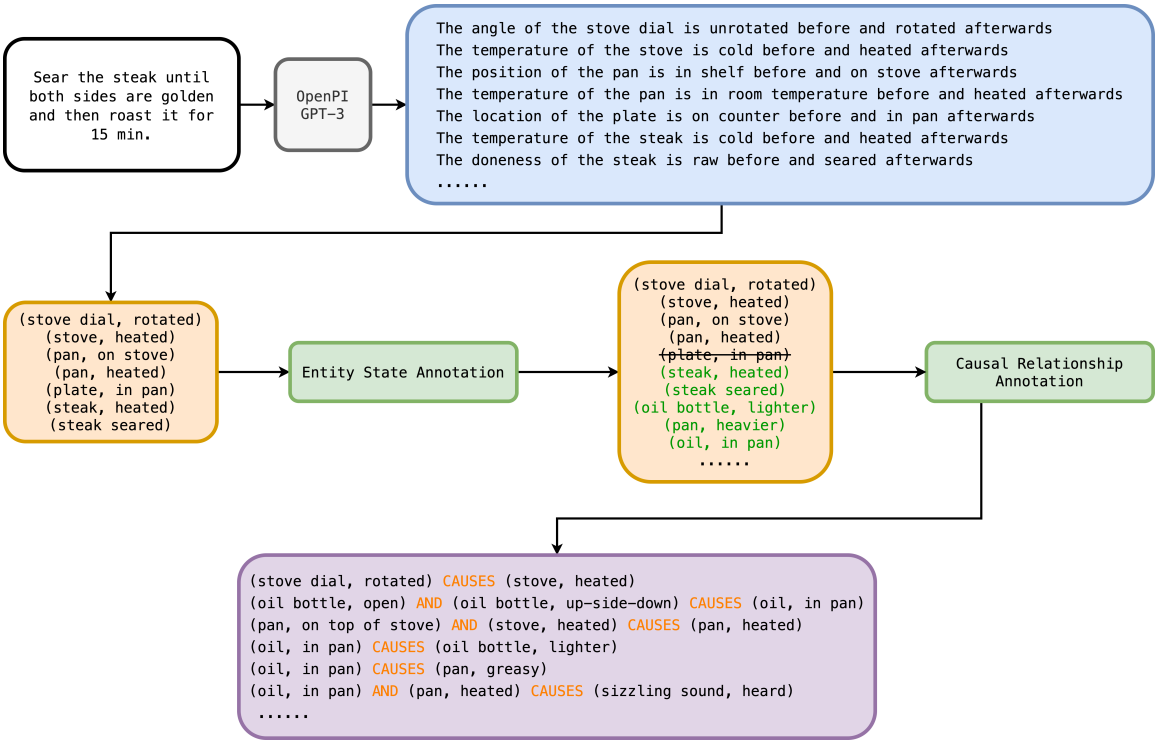

Figure 22: Demonstration of the data annotation pipeline of the C2RES benchmark. The original input (white box) is first input into an OpenPI-finetuned GPT-3 model. The output is entity state changes narrated in the original OpenPI format (blue box). The raw output of the GPT-3 model is first parsed to (entity, post-condition) tuples (left orange box) and then modified by humans (left green box). The causal relationships between the resulting set of entity state changes (right orange box) are then labeled by human annotators (right orange box).

to aid entity state annotation. Therefore, the existing state changes in the OpenPI dataset implicitly contains multimodal information.

Figure 22 shows the overall annotation pipeline of the C2RES benchmark, which consists of a collaboration between language models and human annotators. In the initial stage of the annotation, we first use a GPT-3 model[1] finetuned with the OpenPI dataset. The finetuned GPT-3 model is capable of generating potential entity state changes that occur at the current step in the following template

The {A} *of* {E} *is* {PRE} *before and* {POS} *afterwards.*

[1]We finetuned the GPT-3 davinci model that has 175B parameters.

where "{A}" is replaced by the attribute of the current entity, "{E}" is replaced by the entity itself, "{PRE}" and "{POS}" is replaced by the pre-condition and post-condition of the entity (see the blue box of Figure 22 for examples). Since the `C2RES` task is only concerned with the current entity state, we then parse the GPT-3 generation to `(entity, state)` tuples, where the `state` is the post-condition from above. The (entity, state) tuples provide a promising list of potential entity states at the current step.

In the next step, a human annotator is to edit the machine-generated list of entity states. Specifically, each annotator is assigned two tasks: (1) selected the correct entity states from the machine-generated list and (2) add entity states that machine did not generate. Figure 23 shows a screenshot of the annotation script. The annotators have access to the complete procedure and the current step so that they can have a better grasp of the contextual information in the procedure. In addition, the `(entity, state)` tuples are converted to a narrative following the template, "{entity} is {state}". To select correct machine-generated entity states, the annotators simply input the indices of the entity state and separate them with a comma. After selecting from the existing list of entity states, the annotator also has the option to write down entity states that are not in the list following the same template ("{entity} is {state}").

After human annotation, we now have a promising set of entity states for the current step. The next task is to annotate causal relationships between the entity states. At this stage of annotation, the annotator is provided with a description of the current step, a pivot entity, and a list of alternative entities (Figure 24). The pivot entity is the cause and the list of alternative entities are potential effects. The annotator is to select all entities from the alternative list that is a direct effect of the pivot entity.

The resulting `C2RES` dataset contains 67 cooking procedures, which consist of 565 steps. There are a total of 4896 entities annotated and there are 4842 causal relationships annotated amongst these entities.

```
------------------------------

All steps:
------------------------------
spread batter on pan
spread chutney on top
spread masala on top
drizzle oil on top and around the sides and let cook
fold in half and serve hot
------------------------------

Current Step:
spread batter on pan
------------------------------

Candidate Entity States:
#0: pan is dirty
#1: brush is wet
#2: the batter is on pan
#3: the batter mix is on pan
#4: the pan is heavier
#5: the pan is wet
#6: pan is full
#7: the bowl is emptier
#8: batter is on pan
#9: batter is mixed
#10: pan is covered in batter
------------------------------

Which entity states participated in the current step?
```

Figure 23: A screenshot of the annotation script. For each step, the annotator has access to the complete procedure and a list of machine-generated entity state candidates. The `(entity, state)` tuples are combined with "is" to form a sentence, which is more natural for the annotators.

```
Entity Progress: 1/11 Completed
Step Progress: 1/5 Completed


------------------------------

Current Step:
spread batter on pan
------------------------------

Current Entity State:
pan is dirty
------------------------------

Other Entity State:
2: brush is wet
3: the batter is on pan
4: the pan is heavier
5: the pan is wet
6: pan is full
7: the bowl is emptier
8: pan is covered in batter
10: brush is dirty
11: batter is spread
------------------------------

Which of "other entity states" do "current entity state" have caused?
(write down their indices and seperate with a comma ",")
```

Figure 24: A screenshot of the annotation script. For each step, the annotator has access to the complete procedure and a list of machine-generated entity state candidates. The `(entity, state)` tuples are combined with "is" to form a sentence, which is more natural for the annotators.

5.3. Discussion

Annotating causal relationships between entity states turn out to be a challenging process even for humans. We suspect the main difficulty is that we humans do not explicitly work with fine-grained causal inference such as deducing the causal relationships between entity states. Therefore, the annotators oftentimes mistake the direction of cause and effect or consider the spurious correlation with causation. Furthermore, many fine-grained entity states happen almost concurrently, which makes it difficult to draw a conclusion on cause and effect (see Table 8 for examples).

Entity State 1	Entity State 2	Source of Difficulty
steak is heated	steak is seared	concurrency of entity states
steak is seared	steak is cooked	concurrency of entity states
plate is full	plate is dirty	concurrency of entity states
plate is empty	steak is in pan	direction of causation

Table 8: Some sample entity state pairs and the corresponding difficulty of annotating the causal relationships between these entity state pairs.

With the aforementioned challenges, the inter-annotator agreement on the `C2RES` is 0.876. The reason for this seemingly high agreement is that most entity pairs have no causal relationships between them. Concretely, with 4896 annotated entities, there are only 4842 annotated pairwise causal relationships. Therefore, much of the agreement goes to the consensus on non-causal entity state pairs. When it comes to the macro-F1 score, the inter-annotator agreement drops to 0.443, which reflects a more realistic agreement score.

Experiments show that the fine-grained causal reasoning task is even more challenging for LLMs. For the baseline models, we used the most prominent model at the time of this study, which are GPT3.5 and ChatGPT[2] and we attempted two formulations of the `C2RES` task.

The formulation follows the classical binary Natural Language Inference (NLI) setup, where the premise is the cause entity state and the hypothesis is the effect entity state. The labels

[2]We utilize these models through OpenAI API. The model indices are `text-davinci-003` for GPT3.5 and `gpt-3.5-turbo` for ChatGPT.

dictate whether the two entity states are causally related and whether the current direction of cause and effect is sound. For the NLI formulation, we employed the following prompt

Code Prompt (NLI)

```
Goal: {goal}
Steps: {steps}
Causal Relation at the most recent
step: {causal_rel}
Is the above causal relation correct?
Answer with True or False. {ans}
```

where the placeholders are replaced with current information about the procedure and entity states. For {steps}, we replace it with the steps of the procedure up to the current step. For {causal_rel}, we replace it with a narrative of the causal relation in the following format

({entity state 1}) causes ({entity state 2})

we additionally surround the entity states with parenthesis to emphasize the syntactic structure of the template. For ChatGPT, we employed the same template with a slight tweak to match the style of a conversation:

ChatGPT Prompt (NLI)

```
{"role": "user", "content": "I am going to {goal}."},
{"role": "assistant", "content": "OK, sounds good."},
{"role": "user", "content": "{steps}"},
{"role": "assistant", "content": "OK, got it."},
{"role": "user", "content": "During the latest step, {causal_rel}. Is this causal
relation correct? Answer with True or False." },
{"role": "assistant", "content": "{ans}"}
```

The alternative formulation is a clustering task, where given a complete setup of entity states at the current step, LLMs need to determine all the causal relationships. This formulation mimics how human annotators are prompted during the annotation process.

Formulation	Precision	Recall	Macro-F1	Accuracy
Human Performance	0.468	0.421	0.443	0.876
ChatGPT-NLI	0.125	0.630	0.209	0.455
GPT3.5-NLI	0.130	0.478	0.205	0.577
ChatGPT-Cluster	0.175	0.214	0.192	–
GPT3.5-Cluster	0.147	0.236	0.181	–

Table 9: Performance score of GPT3.5 and ChatGPT on the C2RES dataset using two formulations.

Code Prompt

```
Goal: {goal}
Steps: {steps}
Current Object States: {entity_states}
What are the causes and effects of these object state
changes? Represent with their corresponding indices.
{ans}
```

Similarly, we adjusted the ChatGPT prompt to make it more dialogue-oriented:

ChatGPT Prompt

```
{"role": "user", "content": "I am going to {goal}"},
{"role": "assistant", "content": "OK, sounds good."},
{"role": "user", "content": "First, I spread the butter on two toast."},
{"role": "assistant", "content": "OK, got it."},
{"role": "user", "content": "During the latest step, the following object states
are changed:{entity states}"},
{"role": "assistant", "content": "Ok, got it."},
{"role": "user", "content": "What are the causes and effects of these object
state changes? Represent with their corresponding indices."},
{"role": "assistant", "content": "The causes and effects are:{ans}"}
```

Table 9 shows the result of the two formulations. We see that while C2RES is already a challenging task for humans, it is even more challenging for machines. Under the NLI formulation, both ChatGPT and GPT3.5 are falling way behind human performance both

in the precision and accuracy scores. Un the clustering formulation, both models are way behind human performance across all performance metrics. These results demonstrate that the C2RES reasoning task is challenging for state-of-the-art LLMs albeit with low inter-annotator agreement. Therefore, we believe that the C2RES benchmark will be of promising research resource once the annotation pipeline is refined and the quantity is increased.

CHAPTER 6 : Conclusion

In this thesis, we proposed a new perspective of tackling commonsense causal reasoning tasks in Chapter 3. Specifically, we used event likelihood change as a surrogate for causal relationships and demonstrated the effectiveness of leveraging fine-grained entity state information when deducing causal relationships between coarse-grained events (the bottleneck model for causal reasoning). Further, we also showed that formulating reasoning tasks to a structured representation (Python code) brings significant performance gain to LLMs. Inspired by the Chain-of-Thought (CoT) prompting paradigm and the effectiveness of code language models such as Codex, we integrated the bottleneck reasoning model using CoT prompting with code language prompt.

In Chapter 3, we see that current language models are lacking in their capability of reasoning about the fine-grained causal relationships between an event and an entity state change. We hypothesize that this issue is caused by reporting bias in the training data of LLMs. In our study of procedural texts and commonsense reasoning, we specifically investigate the reporting bias on commonsense causal knowledge in the effect of events on entities. To this extent, one possible approach is to leverage information in external modalities such as vision and audio to extract entities that appeared in the image/speech but do not exist in written texts. In Chapter 4, we attempted to leverage visual information and audio transcriptions. The results from both visual-guided entity extraction and audio-guided entity extraction show large research gaps. For visual information, the vision and multimodal models are not capable of understanding the context of the procedure at the level of pure language models. This limitation, combined with the emphasis on entity coverage rather than saliency in vision tasks like object detection, results in a large number of redundant entities being extracted from visual information. Yet, there is no adequate method to reliably filter entities that participate in a procedure from the large number of entities that exist in the current scene.

This thesis leads to the following challenges and future research directions for the field of

commonsense reasoning:

- In the `CREPE` benchmark (Chapter 3), we utilized event likelihood change as a surrogate for causal relationships. While this definition of causal relationship has good inter-annotator agreement, future work can be done to formalize this definition and potentially blend this definition with other established causal reasoning pipelines and frameworks.

- The error analysis in Chapter 3 shows that current LLMs are bad at reasoning about the causal relationship between events and entities. A potential future work would be to further investigate the capability of current LLMs on finer-grained causal inference. In the case of procedural texts, one possible task would be to reason about the causal relationships between entity states.

- The multimodal models are lacking in understanding the semantics of image conditioning on some context information. The studies carried out in Chatper 4 showed that the entities extracted by current multimodal models are oftentimes non-existent in the scene or did not participate in the current event. Ideally, a multimodal model should be able to reason about the semantics of an image or video using some language input as a prior. In the case of procedural reasoning, a multimodal agent should reason about the saliency of the entities presented in a scene conditioning on the current events in the step.

- The `C2RES` benchmarks is shown to be challenging even for state-of-the-art LLMs like ChatGPT and GPT3.5. However, the low inter-annotator agreement poses a question about the reliability of the performance scores. Therefore, a vital next step is to refine the annotation process and reformulate the annotation protocols. Furthermore, we discovered that existing definition of causality are tailored towards high-level events and may not be applicable for fine-grained causal inference. Therefore, another future step is to come up with a novel definition of fine-grained causality.

BIBLIOGRAPHY

[1] Openai (2023).

[2] Ahn, M., Brohan, A., Brown, N., Chebotar, Y., Cortes, O., David, B., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691* (2022).

[3] Amac, M. S., Yagcioglu, S., Erdem, A., and Erdem, E. Procedural reasoning networks for understanding multimodal procedures. *arXiv preprint arXiv:1909.08859* (2019).

[4] Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 39–48.

[5] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 2425–2433.

[6] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., and Parikh, D. VQA: Visual Question Answering. In *International Conference on Computer Vision (ICCV)* (2015).

[7] Berant, J., Srikumar, V., Chen, P.-C., Vander Linden, A., Harding, B., Huang, B., Clark, P., and Manning, C. D. Modeling biological processes for reading comprehension. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (2014), pp. 1499–1510.

[8] Bethard, S., Corvey, W. J., Klingenstein, S., and Martin, J. H. Building a corpus of temporal-causal structure. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)* (2008).

[9] Bethard, S., and Martin, J. H. Learning semantic links from a corpus of parallel temporal and causal relations. In *Proceedings of ACL-08: HLT, Short Papers* (2008), pp. 177–180.

[10] Bhagavatula, C., Le Bras, R., Malaviya, C., Sakaguchi, K., Holtzman, A., Rashkin, H., Downey, D., Yih, W.-t., and Choi, Y. Abductive commonsense reasoning. In *International Conference on Learning Representations* (2020).

[11] Bosselut, A., Levy, O., Holtzman, A., Ennis, C., Fox, D., and Choi, Y. Simulating action dynamics with neural process networks. *arXiv preprint arXiv:1711.05313* (2017).

[12] Bosselut, A., Levy, O., Holtzman, A., Ennis, C., Fox, D., and Choi, Y. Simulating action dynamics with neural process networks. In *International Conference on Learning Representations* (2018).

[13] Bosselut, A., Levy, O., Holtzman, A., Ennis, C., Fox, D., and Choi, Y. Simulating action dynamics with neural process networks. In *International Conference on Learning Representations* (2018).

[14] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems 33* (2020), 1877–1901.

[15] Bunge, M. *Causality and modern science*. Routledge, 2017.

[16] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16* (2020), Springer, pp. 213–229.

[17] Caselli, T., and Vossen, P. The event storyline corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop* (2017), pp. 77–86.

[18] Caselli, T., and Vossen, P. The event StoryLine corpus: A new benchmark for causal and temporal relation extraction. In *Proceedings of the Events and Stories in the News Workshop* (Vancouver, Canada, Aug. 2017), Association for Computational Linguistics, pp. 77–86.

[19] Chen, M., Zhang, H., Ning, Q., Li, M., Ji, H., McKeown, K., and Roth, D. Event-centric natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Tutorial Abstracts* (2021), pp. 6–14.

[20] Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311* (2022).

[21] Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, E., Wang, X., Dehghani, M., Brahma, S., et al. Scaling instruction-finetuned language models. *arXiv preprint arXiv:2210.11416* (2022).

[22] Clark, P., Dalvi, B., and Tandon, N. What happened? leveraging verbnet to predict the effects of actions in procedural text. *arXiv preprint arXiv:1804.05435* (2018).

[23] Cook, T. D., and Campbell, D. T. *Quasi-experimentation: Design & analysis issues for field settings*, vol. 351. 1979.

[24] Dalvi, B., Huang, L., Tandon, N., tau Yih, W., and Clark, P. Tracking state changes in procedural text: A challenge dataset and models for process paragraph comprehension. *NAACL* (2018).

[25] Dalvi, B., Huang, L., Tandon, N., Yih, W.-t., and Clark, P. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (2018), pp. 1595–1604.

[26] Dalvi, B., Huang, L., Tandon, N., Yih, W.-t., and Clark, P. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)* (New Orleans, Louisiana, June 2018), Association for Computational Linguistics, pp. 1595–1604.

[27] Das, R., Munkhdalai, T., Yuan, X., Trischler, A., and McCallum, A. Building dynamic knowledge graphs from text using machine reading comprehension. *arXiv preprint arXiv:1810.05682* (2018).

[28] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[29] Do, Q., Chan, Y. S., and Roth, D. Minimally supervised event causality identification. In *Proceedings of the 2011 conference on empirical methods in natural language processing* (2011), pp. 294–303.

[30] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[31] Du, L., Ding, X., Liu, T., and Qin, B. Learning event graph knowledge for abductive reasoning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)* (2021), pp. 5181–5190.

[32] Du, L., Ding, X., Xiong, K., Liu, T., and Qin, B. e-care: a new dataset for exploring explainable causal reasoning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2022), pp. 432–446.

[33] Dunietz, J., Levin, L., and Carbonell, J. G. The because corpus 2.0: Annotating causality and overlapping relations. In *Proceedings of the 11th Linguistic Annotation Workshop* (2017), pp. 95–104.

[34] Fisher, R. A. Cancer and smoking. *Nature 182*, 4635 (1958), 596–596.

[35] Fu, Yao; Peng, H., and Khot, T. How does gpt obtain its ability? tracing emergent abilities of language models to their sources. *Yao Fu's Notion* (Dec 2022).

[36] Gan, Z., Cheng, Y., Kholy, A. E., Li, L., Liu, J., and Gao, J. Multi-step reasoning via recurrent dual attention for visual dialog. *arXiv preprint arXiv:1902.00579* (2019).

[37] Geva, M., Khashabi, D., Segal, E., Khot, T., Roth, D., and Berant, J. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics 9* (2021), 346–361.

[38] Gil, Y., Ratnakar, V., and Frtiz, C. Tellme: Learning procedures from tutorial instruction. In *Proceedings of the 16th international conference on intelligent user interfaces* (2011), pp. 227–236.

[39] Girju, R., Nakov, P., Nastase, V., Szpakowicz, S., Turney, P., and Yuret, D. Semeval-2007 task 04: Classification of semantic relations between nominals. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)* (2007), pp. 13–18.

[40] Gordon, A. S., Kozareva, Z., and Roemmele, M. Semeval-2012 task 7: Choice of plausible alternatives: An evaluation of commonsense causal reasoning. *Introduction to* SEM 2012* (2012), 394.

[41] Gordon, J., and Van Durme, B. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 workshop on Automated knowledge base construction* (2013), pp. 25–30.

[42] Gunning, D. Machine common sense concept paper. *arXiv preprint arXiv:1810.07528* (2018).

[43] Gupta, A., and Durrett, G. Tracking discrete and continuous entity state for process understanding. In *Proceedings of the Third Workshop on Structured Prediction for NLP* (2019), pp. 7–12.

[44] He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 770–778.

[45] Henaff, M., Weston, J., Szlam, A., Bordes, A., and LeCun, Y. Tracking the world state with recurrent entity networks. *arXiv preprint arXiv:1612.03969* (2016).

[46] Henaff, M., Weston, J., Szlam, A., Bordes, A., and LeCun, Y. Tracking the world state with recurrent entity networks. In *5th International Conference on Learning Representations, ICLR 2017* (2019).

[47] Hendrickx, I., Kim, S. N., Kozareva, Z., Nakov, P., Séaghdha, D. O., Padó, S., Pennacchiotti, M., Romano, L., and Szpakowicz, S. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *SEW-2009 Semantic Evaluations: Recent Achievements and Future Directions* (2009), 94.

[48] Hidey, C., and McKeown, K. Identifying causal relations using parallel wikipedia articles. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2016), pp. 1424–1433.

[49] Hill, A. B. The environment and disease: association or causation?, 1965.

[50] Hochreiter, S., and Schmidhuber, J. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[51] Hu, R., Andreas, J., Rohrbach, M., Darrell, T., and Saenko, K. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 804–813.

[52] Huang, H., Geng, X., Pei, J., Long, G., and Jiang, D. Reasoning over entity-action-location graph for procedural text understanding. 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)* (2021), pp. 5100–5109.

[53] Iyyer, M., Manjunatha, V., Guha, A., Vyas, Y., Boyd-Graber, J., Daume, H., and Davis, L. S. The amazing mysteries of the gutter: Drawing inferences between panels in comic book narratives. In *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition* (2017), pp. 7186–7195.

[54] Johnson, J., Hariharan, B., Van Der Maaten, L., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 2901–2910.

[55] Johnson, J., Hariharan, B., Van Der Maaten, L., Hoffman, J., Fei-Fei, L., Lawrence Zitnick, C., and Girshick, R. Inferring and executing programs for visual reasoning. In *Proceedings of the IEEE international conference on computer vision* (2017), pp. 2989–2998.

[56] Kembhavi, A., Seo, M., Schwenk, D., Choi, J., Farhadi, A., and Hajishirzi, H. Are you smarter than a sixth grader? textbook question answering for multimodal machine comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern recognition* (2017), pp. 4999–5007.

[57] Kenton, J. D. M.-W. C., and Toutanova, L. K. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT* (2019), pp. 4171–4186.

[58] Kiddon, C., Ponnuraj, G. T., Zettlemoyer, L., and Choi, Y. Mise en place: Unsupervised interpretation of instructional recipes. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (2015), pp. 982–992.

[59] Kiddon, C., Zettlemoyer, L., and Choi, Y. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (2016), pp. 329–339.

[60] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations* (2020).

[61] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. 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* (Online, July 2020), Association for Computational Linguistics, pp. 7871–7880.

[62] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. 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* (2020), pp. 7871–7880.

[63] Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., and Chang, K.-W. Visualbert: Asimple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557* (2019).

[64] Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., and Chang, K.-W. What does bert with vision look at? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (2020), pp. 5265–5275.

[65] Li, S., Li, X., Shang, L., Dong, Z., Sun, C.-J., Liu, B., Ji, Z., Jiang, X., and Liu, Q. How pre-trained language models capture factual knowledge? a causal-inspired analysis. In *Findings of the Association for Computational Linguistics: ACL 2022* (2022), pp. 1720–1732.

[66] Li, Z., Ding, X., Liu, T., Hu, J. E., and Van Durme, B. Guided generation of cause and effect. In *Proceedings of the Twenty-Ninth International Conference on International Joint Conferences on Artificial Intelligence* (2021), pp. 3629–3636.

[67] Liang, C., Wang, W., Zhou, T., and Yang, Y. Visual abductive reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 15565–15575.

[68] Liu, A., Yuan, S., Zhang, C., Luo, C., Liao, Y., Bai, K., and Xu, Z. Multi-level multimodal transformer network for multimodal recipe comprehension. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval* (2020), pp. 1781–1784.

[69] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[70] Long, R., Pasupat, P., and Liang, P. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (Berlin, Germany, Aug. 2016), Association for Computational Linguistics, pp. 1456–1465.

[71] Long, R., Pasupat, P., and Liang, P. Simpler context-dependent logical forms via model projections. *arXiv preprint arXiv:1606.05378* (2016).

[72] Lu, J., Clark, C., Zellers, R., Mottaghi, R., and Kembhavi, A. Unified-io: A unified model for vision, language, and multi-modal tasks. *arXiv preprint arXiv:2206.08916* (2022).

[73] Luo, Z., Sha, Y., Zhu, K. Q., Hwang, S.-w., and Wang, Z. Commonsense causal reasoning between short texts. In *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning* (2016).

[74] Lyu, Q., Havaldar, S., Stein, A., Zhang, L., Rao, D., Wong, E., Apidianaki, M., and Callison-Burch, C. Faithful chain-of-thought reasoning. *arXiv preprint arXiv:2301.13379* (2023).

[75] Madaan, A., Zhou, S., Alon, U., Yang, Y., and Neubig, G. Language models of code are few-shot commonsense learners. *arXiv preprint arXiv:2210.07128* (2022).

[76] Maeta, H., Sasada, T., and Mori, S. A framework for procedural text understanding. In *Proceedings of the 14th International Conference on Parsing Technologies* (2015), pp. 50–60.

[77] Manning, C. D., Surdeanu, M., Bauer, J., Finkel, J. R., Bethard, S., and McClosky, D. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations* (2014), pp. 55–60.

[78] Meng, D., Chen, X., Fan, Z., Zeng, G., Li, H., Yuan, Y., Sun, L., and Wang, J. Conditional detr for fast training convergence. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 3651–3660.

[79] Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[80] Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., et al. Simple open-vocabulary object detection with vision transformers. *arXiv preprint arXiv:2205.06230* (2022).

[81] Mirza, P., Sprugnoli, R., Tonelli, S., and Speranza, M. Annotating causality in the tempeval-3 corpus. In *Proceedings of the EACL 2014 Workshop on Computational Approaches to Causality in Language (CAtoCL)* (2014), pp. 10–19.

[82] Mirza, P., and Tonelli, S. Catena: Causal and temporal relation extraction from natural language texts. In *The 26th international conference on computational linguistics* (2016), ACL, pp. 64–75.

[83] Mishra, B. D., Huang, L., Tandon, N., Yih, W.-t., and Clark, P. Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. *arXiv preprint arXiv:1805.06975* (2018).

[84] Mostafazadeh, N., Grealish, A., Chambers, N., Allen, J., and Vanderwende, L. Caters: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events* (2016), pp. 51–61.

[85] Mysore, S., Jensen, Z., Kim, E., Huang, K., Chang, H.-S., Strubell, E., Flanigan, J., McCallum, A., and Olivetti, E. The materials science procedural text corpus: Annotating materials synthesis procedures with shallow semantic structures. In *Proceedings of the 13th Linguistic Annotation Workshop* (Florence, Italy, Aug. 2019), Association for Computational Linguistics, pp. 56–64.

[86] Ning, Q., Feng, Z., Wu, H., and Roth, D. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2018), pp. 2278–2288.

[87] Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. Training language models to follow instructions with human feedback. *arXiv preprint arXiv:2203.02155* (2022).

[88] Pareti, P., Testu, B., Ichise, R., Klein, E., and Barker, A. Integrating know-how into the linked data cloud. In *International Conference on Knowledge Engineering and Knowledge Management* (2014), Springer, pp. 385–396.

[89] PEARL, J. Bayesian networks: a model of self-activated memory for evidential reasoning. In *Proceedings of the 7th Conference of the Cognitive Science Society, 1985* (1985), pp. 329–334.

[90] Pearl, J., et al. Models, reasoning and inference. *Cambridge, UK: CambridgeUniversityPress 19*, 2 (2000).

[91] Pennington, J., Socher, R., and Manning, C. D. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)* (2014), pp. 1532–1543.

[92] Perez, E., De Vries, H., Strub, F., Dumoulin, V., and Courville, A. Learning visual reasoning without strong priors. *arXiv preprint arXiv:1707.03017* (2017).

[93] Press, O., Muru Zhang, S. M., Schmidt, L., Smith, N. A., and Lewis, M. Measuring and narrowing the compositionality gap in language models.

[94] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning* (2021), PMLR, pp. 8748–8763.

[95] Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al. Improving language understanding by generative pre-training.

[96] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners.

[97] Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res. 21*, 140 (2020), 1–67.

[98] Rashkin, H., Bosselut, A., Sap, M., Knight, K., and Choi, Y. Modeling naive psychology of characters in simple commonsense stories. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2018), pp. 2289–2299.

[99] Ren, S., He, K., Girshick, R., and Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems 28* (2015).

[100] Roemmele, M., Bejan, C. A., and Gordon, A. S. Choice of plausible alternatives: An evaluation of commonsense causal reasoning.

[101] Russell, B. On the notion of cause. In *Proceedings of the Aristotelian society* (1912), vol. 13, JSTOR, pp. 1–26.

[102] Sanh, V., Webson, A., Raffel, C., Bach, S. H., Sutawika, L., Alyafeai, Z., Chaffin, A., Stiegler, A., Scao, T. L., Raja, A., et al. Multitask prompted training enables zero-shot task generalization. *arXiv preprint arXiv:2110.08207* (2021).

[103] Sap, M., Shwartz, V., Bosselut, A., Choi, Y., and Roth, D. Commonsense reasoning for natural language processing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts* (Online, July 2020), Association for Computational Linguistics, pp. 27–33.

[104] Seo, M., Min, S., Farhadi, A., and Hajishirzi, H. Query-reduction networks for question answering. *arXiv preprint arXiv:1606.04582* (2016).

[105] Shi, Q., Liu, Q., Chen, B., Zhang, Y., Liu, T., and Lou, J.-G. Lemon: Language-based environment manipulation via execution-guided pre-training. *arXiv preprint arXiv:2201.08081* (2022).

[106] Tandon, N., Dalvi, B., Grus, J., Yih, W.-t., Bosselut, A., and Clark, P. Reasoning about actions and state changes by injecting commonsense knowledge. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing* (Brussels, Belgium, Oct.-Nov. 2018), Association for Computational Linguistics, pp. 57–66.

[107] Tandon, N., Dalvi, B., Sakaguchi, K., Clark, P., and Bosselut, A. WIQA: A dataset for "what if..." reasoning over procedural text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 6076–6085.

[108] Tandon, N., Sakaguchi, K., Dalvi, B., Rajagopal, D., Clark, P., Guerquin, M., Richardson, K., and Hovy, E. A dataset for tracking entities in open domain procedural text. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020), pp. 6408–6417.

[109] Tapaswi, M., Zhu, Y., Stiefelhagen, R., Torralba, A., Urtasun, R., and Fidler, S. MovieQA: Understanding Stories in Movies through Question-Answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016).

[110] Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239* (2022).

[111] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems 30* (2017).

[112] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[113] Wang, B. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

[114] Wang, C.-C., Thorpe, C., Thrun, S., Hebert, M., and Durrant-Whyte, H. Simultaneous localization, mapping and moving object tracking. *The International Journal of Robotics Research 26*, 9 (2007), 889–916.

[115] Wang, P., Yang, A., Men, R., Lin, J., Bai, S., Li, Z., Ma, J., Zhou, C., Zhou, J., and Yang, H. Unifying architectures, tasks, and modalities through a simple sequence-to-sequence learning framework. *arXiv preprint arXiv:2202.03052* (2022).

[116] Wang, Z., and Bovik, A. C. Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE signal processing magazine 26*, 1 (2009), 98–117.

[117] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*, 4 (2004), 600–612.

[118] Weber, N., Rudinger, R., and Van Durme, B. Causal inference of script knowledge. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020), pp. 7583–7596.

[119] Wei, J., Tay, Y., Bommasani, R., Raffel, C., Zoph, B., Borgeaud, S., Yogatama, D., Bosma, M., Zhou, D., Metzler, D., et al. Emergent abilities of large language models. *arXiv preprint arXiv:2206.07682* (2022).

[120] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903* (2022).

[121] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E. H., Le, Q. V., Zhou, D., et al. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems* (2022).

[122] Weston, J., Bordes, A., Chopra, S., and Mikolov, T. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv: Artificial Intelligence* (2015).

[123] Weston, J., Chopra, S., and Bordes, A. Memory networks. *arXiv preprint arXiv:1410.3916* (2014).

[124] Wu, T.-L., Spangher, A., Alipoormolabashi, P., Freedman, M., Weischedel, R., and Peng, N. Understanding multimodal procedural knowledge by sequencing multimodal instructional manuals. In *Proceedings of the 60th Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (2022), pp. 4525–4542.

[125] Wu, T.-L., Zhang, C., Hu, Q., Spangher, A., and Peng, N. Learning action conditions from instructional manuals for instruction understanding. *arXiv preprint arXiv:2205.12420* (2022).

[126] Wu, Z., Geiger, A., Rozner, J., Kreiss, E., Lu, H., Icard, T., Potts, C., and Goodman, N. Causal distillation for language models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (Seattle, United States, July 2022), Association for Computational Linguistics, pp. 4288–4295.

[127] Yagcioglu, S., Erdem, A., Erdem, E., and Ikizler-Cinbis, N. Recipeqa: A challenge dataset for multimodal comprehension of cooking recipes. *arXiv preprint arXiv:1809.00812* (2018).

[128] Yang, J., Chen, X., Jiang, M., Chen, S., Wang, L., and Zhao, Q. Visualhow: Multimodal problem solving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 15627–15637.

[129] Yang, X., Obadinma, S., Zhao, H., Zhang, Q., Matwin, S., and Zhu, X. Semeval-2020 task 5: Counterfactual recognition. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation* (2020), pp. 322–335.

[130] Yang, Y., Panagopoulou, A., Lyu, Q., Zhang, L., Yatskar, M., and Callison-Burch, C. Visual goal-step inference using wikihow. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (2021), pp. 2167–2179.

[131] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems 32* (2019).

[132] Zellers, R., Bisk, Y., Farhadi, A., and Choi, Y. From recognition to cognition: Visual commonsense reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019).

[133] Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. Hellaswag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (2019), pp. 4791–4800.

[134] Zellers, R., Lu, J., Lu, X., Yu, Y., Zhao, Y., Salehi, M., Kusupati, A., Hessel, J., Farhadi, A., and Choi, Y. Merlot reserve: Neural script knowledge through vision and language and sound. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2022), pp. 16375–16387.

[135] Zellers, R., Lu, X., Hessel, J., Yu, Y., Park, J. S., Cao, J., Farhadi, A., and Choi, Y. Merlot: Multimodal neural script knowledge models. In *Advances in Neural Information Processing Systems 34* (2021).

[136] Zellers, R., Lu, X., Hessel, J., Yu, Y., Park, J. S., Cao, J., Farhadi, A., and Choi, Y. Merlot: Multimodal neural script knowledge models. *Advances in Neural Information Processing Systems 34* (2021), 23634–23651.

[137] Zhang, H., Zhang, Z., Zhang, Y., Wang, J., Li, Y., Yang, Z., et al. Modeling temporal-modal entity graph for procedural multimodal machine comprehension. *arXiv preprint arXiv:2204.02566* (2022).

[138] Zhang, J., Zhang, H., Su, W., and Roth, D. Rock: Causal inference principles for reasoning about commonsense causality. In *International Conference on Machine Learning* (2022), PMLR, pp. 26750–26771.

[139] Zhang, L., Lyu, Q., and Callison-Burch, C. Reasoning about goals, steps, and temporal ordering with wikihow. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020), pp. 4630–4639.

[140] Zhang, L., Xu, H., Yang, Y., Zhou, S., You, W., Arora, M., and Callison-Burch, C. Causal reasoning of entities and events in procedural texts. In *Findings of the Association for Computational Linguistics: EACL 2023* (Dubrovnik, Croatia, May 2023), Association for Computational Linguistics.

[141] Zhang, Z., Webster, P., Uren, V., Varga, A., and Ciravegna, F. Automatically extracting procedural knowledge from instructional texts using natural language processing. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)* (2012), pp. 520–527.

[142] Zhao, Z.-Q., Zheng, P., Xu, S.-t., and Wu, X. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems 30*, 11 (2019), 3212–3232.

[143] Zhong, W., Tang, D., Duan, N., Zhou, M., Wang, J., and Yin, J. A heterogeneous graph with factual, temporal and logical knowledge for question answering over dynamic contexts. *arXiv preprint arXiv:2004.12057* (2020).

[144] Zhou, L., Xu, C., and Corso, J. J. Towards automatic learning of procedures from web instructional videos. In *AAAI Conference on Artificial Intelligence* (2018), pp. 7590–7598.

[145] Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., and Fidler, S. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 19–27.