# Probabilistic Commonsense Knowledge

Xiang Li
*University of Massachusetts Amherst*

# PROBABILISTIC COMMONSENSE KNOWLEDGE

A Dissertation Presented

by

XIANG LI

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

September 2022

Manning College of Information and Computer Sciences

# PROBABILISTIC COMMONSENSE KNOWLEDGE

A Dissertation Presented

by

XIANG LI

Approved as to style and content by:

## ACKNOWLEDGMENTS

It would not have been possible for me to complete this dissertation without the support of many people. First, I want to express my deepest gratitude to my advisor, Andrew McCallum, who has been incredibly supportive during my Ph.D. He is a great researcher who is always enthusiastic about brainstorming ideas and embracing new methods. Also, he is caring. Whenever possible, he thinks from the student's perspective. He connected me with other senior researchers at conferences in my early years. Later in my Ph.D., we spent many hours discussing my career choices. My career dream was made possible because of his encouragement. His enthusiasm for research and kind, caring personality makes him my role model. I strive to become someone like him in the many years to come. I also want to thank my committee: Mohit Iyyer, Laure Thompson, Brian Dillon, and Yejin Choi, who gave me numerous suggestions regarding my job talk and this thesis.

Also, I would like to extend my gratitude to Luke Vilinis and Michael Boratko, two of my biggest collaborators. Luke is a talented engineer who can speed up code execution in many different creative ways. He also has a solid mathematical background and knows almost all the math definitions I questioned. Later in my Ph.D., I collaborated with Michael and am consistently impressed with his diligence, carefulness, and persistence under challenging conditions like conference and report deadlines. I have also learned a lot from Michael, including how to think and organize thoughts clearly by writing things down in great detail.

I am fortunate to have summer internships that allowed me to work with many brilliant industry mentors. I was mentored by Colin Evans, Chris Waterson, and Travis Wolfe during my internships at Google. Colin and Chris encouraged me to present my research for feedback, which was an excellent way to learn more. Travis and I would brainstorm and have pair programming sessions, which later led to one of my best publications. At Bloomberg, Gideon Mann and David Rosenberg taught me the importance of clearly presenting my work. Also, I learned about imposter syndrome from Amanda Stent and was able to overcome it later. At FAIR, Douwe Kiela and Sebastian Riedel allowed me to explore NLP QA problems. Lastly, Aida Nematzadeh, Adhiguna Kuncoro, and Phil Blunsom from the DeepMind NLP team taught me many things about writing research papers and designing experiments. The weekly project meetings during my DeepMind internship were something I looked forward to every week.

I want to thank many collaborators and friends in IESL. Emma Strubell chatted with me when I wanted to quit and later set up a female role model. Shib Sankar Dasgupta and Dhruvesh Patel always put 200% consistent effort into experiments with detailed analysis and beautiful plots. Rajashi Das was always ready to help on many projects. I also cherish the Amherst nature walks with Neha Kennard. I want to thank everyone in IESL and UMass NLP: Javier Burroni, Ari Kobren, Nicholas Monath, Patrick Verga, Jeffrey Flanigan, Tim O'Gorman, Shikhar Murty, Jay Yoon Lee, Haw-Shiuan Chang, Dung Thai, Dongxu Zhang, Nishant Yadav, Sheshera Mysore, Andrew Drozdov, Rico Angell, Wenlong Zhao, Nader Akoury, Trapit Bansal, Katherine Keith, Kalpesh Krishna, and Shufan Wang. It is an amazing group for collaboration, and many great papers come from hallway discussions.

I have been blessed to have many friends in the department who made the away from home time in Amherst much more enjoyable, especially during the global pandemic: Chenyun Wu, Simeng Sun, Zhan Xu, Pengshan Cai, Difan Liu, Cezhou Cheng, Mengxue Zhang, Weiqiu You, Marzena Karpinska, Yixiao Song, Scott Jordan, Zhiqi Huang, Zhichao Yang, Rumeng Li, Boya Ren, Aishwarya Kamath, Qingyao Ai, and Chenghao Lv.

A huge thanks to my closest friends, Hailin Wang and Sabrina Mielke. They have always stood by me and supported me no matter what happened. More importantly, I thank my family, especially my mother, for raising me, understanding me, and supporting me. She is always the bedrock of my support. Without her, I would not have been able to study abroad and complete this dissertation. Thank you to my mom Hongxia Zhang for everything.

# ABSTRACT

## PROBABILISTIC COMMONSENSE KNOWLEDGE

SEPTEMBER 2022

XIANG LI

B.Sc., EAST CHINA NORMAL UNIVERSITY

M.Sc., UNIVERSITY OF CHICAGO

Ph.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Andrew McCallum

Commonsense knowledge is critical to achieving artificial general intelligence. This shared common background knowledge is implicit in all human communication, facilitating efficient information exchange and understanding. But commonsense research is hampered by its immense quantity of knowledge because an explicit categorization is impossible. Furthermore, a plumber could repair a sink in a kitchen or a bathroom, indicating that common sense reveals a probable assumption rather than a definitive answer. To align with these properties of commonsense fundamentally, we want to not only model but also evaluate such knowledge human-like using abstractions and probabilistic principles.

Traditional combinatorial probabilistic models, e.g., probabilistic graphical model approaches, have limitations to modeling large-scale probability distributions containing thousands or even millions of commonsensical events. On the other hand, although embedding-based representation learning has the advantage of generalizing to large combinations of events, they suffer from producing consistent probabilities under different styles of queries.

Combining benefits from both sides, we introduce probabilistic box embeddings, which represent joint probability distributions on a learned latent space of geometric embeddings. By using box embeddings, it is now possible to handle queries with intersections, unions, and negations in a way similar to Venn diagram reasoning, which has faced difficulty even when using large language models.

Meanwhile, existing evaluations do not reflect the probabilistic nature of commonsense knowledge. The popular multiple-choice evaluation style often misleads us into the paradigm that commonsense solved. To fill in the gap, we propose a method of retrieving commonsense related question answer distributions from human annotators as well as a novel method of generative evaluation. We utilize these approaches in two new commonsense datasets.

Finally, we draw a connection between the-state-of-art NLP models — large language models and their ability to perform commonsense reasoning tasks. According to the previous study [120], large language models would make inconsistent predictions while given different input texts for plausible commonsense situations. We intend to evaluate their performance using more rigorous probabilistic measurements.

# TABLE OF CONTENTS

# LIST OF TABLES

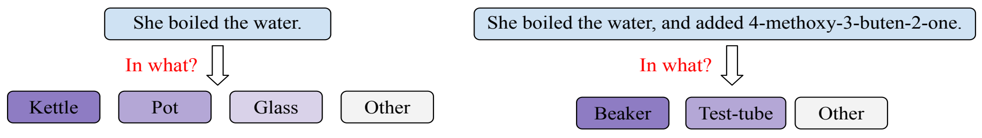
# LIST OF FIGURES

# CHAPTER 1

# INTRODUCTION

## 1.1 Overview

One of the most important and challenging problems facing in AI development is the incorporation of common sense [94, 87, 51], which is the ability to model implicit information. Humans often draw on this knowledge when tackling everyday tasks; thus, an AI system needs to hold this knowledge to become generally applicable. Common sense can be divided into two types: physical [12, 165] and conceptual [17, 132, 129]. Given a direction such as “put the water on the burner to boil,” it is physical common sense which allows us to know if we need to move other objects out of the way and conceptual common sense which allows us to understand that the water is likely in a kettle and not simply dumped on the burner. My thesis is focused on conceptual common sense, which exists implicitly in all human communication, facilitating efficient information exchange and understanding. Any AI system that interacts with language must have the ability to make correct predictions about unstated assumptions.



Common sense is both probabilistic and contextual. In the above example, the water could be placed in a “kettle”, “pot” or “glass”, although the former answers are more probable. But consider how the relative probability shifts if we append the phrase “and add the spaghetti” or changes entirely if we append “and add 4-methoxy-3-buten-2-one,” in which case the vessel is likely a beaker or test-tube. The implicit distribution over the ”burner” also changes from a stove to a Bunsen burner.

Incorporating implicit knowledge is not only applicable in general knowledge settings but is also fundamental to develop models which can assist in high-impact domains such as science and medicine. For example, correctly modeling implicit domain knowledge could be necessary to categorize scientific papers based on the time an experiment typically takes, or the physical environment required. In the clinical and legal domains, correctly modeling domain experts' common unstated knowledge can make obtuse documents more accessible to the general public. Even if the task itself does not explicitly probe this implicit knowledge, any model which does not rely (even in a latent fashion) on this implicit information is not making full use of massive background knowledge — models which do so incorrectly are in danger of drawing incorrect conclusions [117, 63, 64].

This thesis studies designing and deploying models to utilize massive commonsense background knowledge. I believe this is essential to the success of artificial general intelligence (AGI). Furthermore, it can accelerate interdisciplinary research by making common knowledge within a domain *accessible* outside of the domain.

Since modeling common sense inherently involves a joint distribution over multiple possible facts, this thesis aim to model such knowledge using probabilistic abstractions and principles. Typical loss functions for vector embeddings are not naturally capable of encoding a full joint probability distribution, relying on a post-hoc normalization over vector dot product scores to encode a specific form of conditional probability [57, 149]. Thus, in this thesis, we develop a region-based representation, probabilistic box embedding, which is designed for such a calibrated probabilistic representation. Unfortunately, there are no standard probability-based evaluations for common sense: to address this, in this thesis, we also developed a novel evaluation methods to (a) retrieve implicit answer distributions from human annotators, and (b) assess the model generated answers, and show the utility of these methods in new commonsense datasets. In addition, this thesis draw connections between current large language models and common sense by evaluating the extent to which these models exhibit common sense. The performed systematic study finds that while these models have synthesized vast quantity of background information, they are still far from reliable predictors of plausible implicit information — reemphasizing the importance of modeling and evaluating them explicitly in a probabilistic fashion.

## 1.2 Probabilistic Modeling for Common Sense

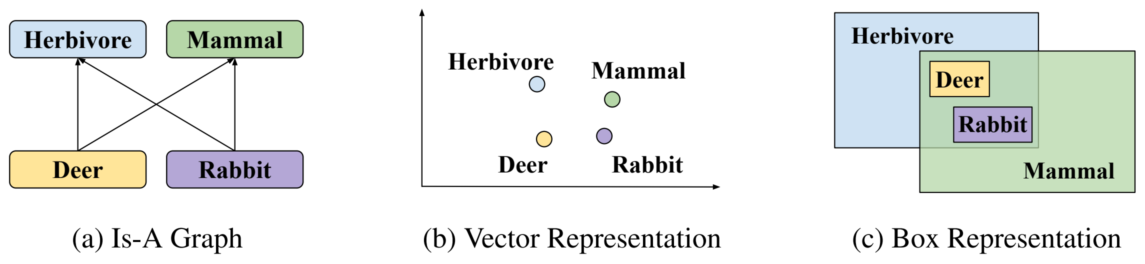


Figure 1.1: The vector and box representation on the is-a hierarchy of two animal species and their types.

Building probabilistic models for common sense is hampered by the immense quantity of knowledge, making an explicit categorization challenging. Traditional combinatorial probabilistic models, such as probabilistic graphical models [72, 69], have limitations when modeling large-scale probability distributions containing thousands or even millions of commonsensical events. On the other hand, vector embedding based representation learning, including large-scale language modeling [123, 23, 154], has the advantage of generalizing to large combinations of events from a large corpus, but their inner workings are opaque and suffer from producing inconsistent probabilities under different contexts [120]. Any AGI models with common sense should be able to learn contextually-dependent representations of implicit information with internally-consistent probabilistic semantics.

In this thesis, we combines the benefits of both approaches by modeling common sense with probabilistic box embeddings, which represent joint probability distributions in a latent space of geometric embeddings [152] shown in Fig. 1.1. We illustrate the superiority of this representation for transitive relational data on both word and sentence levels due to its transitive inductive bias, e.g., entailment can be represented as inclusion among boxes [33, 109]. This representation is not only applicable to transitive relations by capitalizing on the containment property, but is also efficient for modeling commonsense knowledge base graphs, e.g., CONCEPTNET triples with graded uncertainty scores [28]. By using box embeddings, we show that we can now handle queries with intersections, unions, and negations in a way similar to Venn diagram reasoning [32, 152]. Chapter 2 describes this in more detail.

While this model is capable of representing the joint probability of random variables with complex latent dependency structures, it can be challenging to train due to its "hard" box edges. The probabilistic model suffers from a lack of local identifiability. In particular, a large area of the box parameter space yields equivalent scores between boxes, resulting in areas with zero-gradient during training, e.g., making two disjoint boxes overlap can be impossible, because moving two disjoint boxes around does not change the resulting intersection volume until they start overlapping. To mitigate this problem with disjoint boxes, this thesis proposes *SmoothedBox* by transforming the hard edges to smoothed density functions using a Gaussian convolution kernel [81], more details are demonstrated in Chapter 3 .

## 1.3 Probabilistic Evaluation for Common Sense

As commonsense knowledge is inherently probabilistic, we need to have a probabilistic evaluation when measuring progress towards AGI. However, most existing commonsense evaluations were framed as multiple-choice selection question answering tasks [167, 132]. This evaluation requires the model to choose the right answer from a list of candidates, including the correct choice and a few incorrect ones (negatives). High accuracy in this evaluation is misleading as the candidate answer sets are unrealistically small, and generating hard negatives is challenging. Recent benchmarks attempt to overcome this limitation via generative commonsense evaluation [83], which is more challenging as it can be viewed as multiple-choice question answering with practically unlimited choices. While more challenging, it does not reflect the probabilistic nature of common sense.

My work proposes a generative evaluation which incorporates probabilistic elements by comparing a ranked answer list. The work is still framed as a question answering task, however we explicitly take into account that multiple answers could be correct with varying likelihood. In order to estimate the distribution of these commonsense answers, we propose and collect a large number of human responses and manually cluster similar responses into categories in this thesis. We ask models to generate a ranked list of answers and compare this with the clustered human responses, ranked by probability. Therefore, we introduce a new dataset for commonsense question answering with the proposed evaluation, ProtoQA [17]. Chapter 4 depicts this in more detail.

The proposed work broadens the scope of ProtoQA with greater applicability to downstream tasks where we focus on the implicit information for a given sentence, e.g., "The plumber is fixing the sink, where does this happen?" A reasonable answer distribution could include "bathroom", "basement" with a higher probability for the former. In general, we consider a short context sentence and aim to fill in the implicit information by framing it as a question — commonsense frame completion (CFC). We are also extensively exploring options for automatic evaluation, defining a novel approach that measures the KL divergence between probabilities directly. We justify the automatic evaluation with rigorous theoretical motivation and empirical results by demonstrating a high correlation with human scoring. This part of the proposed work is described in section 6.

## 1.4 Common Sense in Large Language Models

More recently, large language models have shown impressive performance on multiple natural language downstream tasks and have been used as foundational models for many applications [23, 111]. It remains an open question to what extent these large language models exhibit commonsense knowledge. To answer this, I performed a systematic investigation evaluating commonsense benchmarks using large language models [78].

A large language model with 7 billion parameters is evaluated against four commonsense benchmarks using zero-shot evaluation. These models show close-to-human performance when trained specifically for commonsense tasks, however without this additional training, these models are still far from being accurate in predicting the implicit commonsense knowledge. Due to the need to rely on implicit distributions of common sense in downstream tasks, errors in predicting them could result in naive mistakes on these tasks.

We have shown in this work (Chapter 5) that the current methods of multiple-choice evaluation permit cheating via memorizing the training corpus instead of reasoning over the new context, and there are diminishing returns as the model size grows: building an ever-larger model on ever-larger data sets will not teach machine common sense. This work demonstrates the importance of modeling commonsense knowledge explicitly [79, 80] and the need for a robust probabilistic evaluation method. This evaluation is not only desirable because common sense is inherently probabilistic, but it also provides a more reliable method of assessing the model generalization ability other than memorization.

## 1.5 Declaration of Collaborations

The following research papers are described in this thesis. They are produced in collaboration with the researchers listed below and have been published or submitted for publication:

- **Xiang Lorraine Li**, Michael Boratko, Pranay Kumar Yelugam, Tim O'Gorman, Nalini Singh, Andrew McCallum. "Commonsense Frame Completion and its Probabilistic Evaluation. " In submission to ACL Rolling Review.

- **Xiang Lorraine Li**, Adhiguna Kuncoro, Jordan Hoffmann, Cyprien de Masson d'Autume, Phil Blunsom, Aida Nematzadeh. "A Systematic Investigation of Commonsense Understanding in Large Language Models." In submission to EMNLP.
- Michael Boratko*, **Xiang Lorraine Li***, Tim O'Gorman*, Rajarshi Das*, Dan Le, Andrew McCallum. "ProtoQA: A Question Answering Dataset for Prototypical Common-Sense Reasoning." The 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020.
- **Xiang Li***, Luke Vilnis*, Dongxu Zhang, Michael Boratko, Andrew McCallum "Smoothing the Geometry of Probabilistic Box Embeddings.", International Conference on Learning Representations (ICLR) 2019. **Oral presentation. 1.5%**
- Luke Vilnis*, **Xiang Li***, Shikhar Murty, Andrew McCallum "Probabilistic Embedding of Knowledge Graphs with Box Lattice Measures.", The Annual Meeting of the Association for Computational Linguistics (ACL), 2018.

# CHAPTER 2

# PROBABILISTIC EMBEDDING OF KNOWLEDGE GRAPHS WITH BOX LATTICE MEASURES

## 2.1 Introduction

A core problem in artificial intelligence is to capture, in machine-usable form, the collection of implicit information that an ordinary person would have, known as *commonsense knowledge*. For example, a machine should know that a room may have a door, and that when a person enters a room, it is generally through a door. This background knowledge is crucial for solving many difficult, ambiguous natural language problems in coreference resolution and question answering, as well as the creation of other reasoning machines. More than just curating a static collection of facts, we would like commonsense knowledge to be represented in a way that lends itself to machine reasoning and inference of missing information. We concern ourselves in this chapter with the problem of learning commonsense knowledge representations.

In machine learning settings, knowledge is usually represented as a hypergraph of triplets such as Freebase [15], WordNet [96], and ConceptNet [141]. In these knowledge graphs, nodes represent entities or terms $t$, and hyperedges are relations $R$ between these entities or terms, with each fact in the knowledge graph represented as a triplet $< t_1, R, t_2 >$. Researchers have developed many models for knowledge representation and learning in this setting [19, 156, 104, 79, 140], under the umbrella of *knowledge graph completion*. However, none of these naturally lend themselves to traditional methods of logical reasoning such as transitivity and negation.

While a knowledge graph completion model can represent relations such as IS-A and entailment, there is no mechanism to ensure that its predictions are internally consistent. For example, if we know that a dog is a mammal, and a pit bull is a dog, we would like the model to also predict that a pit bull is a mammal. These transitive entailment relations describe ontologies of hierarchical data, a key component of commonsense knowledge which we focus on in this work.

Recently, a thread of research on representation learning has aimed to create embedding spaces that automatically enforce consistency in these predictions using the intrinsic geometry of the embedding space [153, 150, 103]. These structured embeddings based on regions, densities, and orderings have gained popularity in recent years for their inductive bias towards the essential asymmetries inherent in problems such as image captioning [150], lexical and textual entailment [38, 153, 73, 2], and knowledge graph completion and reasoning [54, 105, 80].

Models that easily encode asymmetry, and related properties such as transitivity (the two components of commonplace relations such as partially ordered sets and lattices), have great utility in these applications, leaving less to be learned from the data than arbitrary relational models. At their best, they resemble a hybrid between embedding models and structured prediction. As noted by [150] and [80], while the models learn sets of embeddings, these parameters obey rich structural constraints. The entire set can be thought of as one, sometimes provably consistent, structured prediction, such as an ontology in the form of a single directed acyclic graph.

While the structured prediction analogy applies best to Order Embeddings (OE), which embeds consistent partial orders, other region- and density-based representations have been proposed for the express purpose of inducing a bias towards asymmetric relationships. For example, the Gaussian Embedding (GE) model [153] aims to represent the asymmetry and uncertainty in an object's relations and attributes by means of uncertainty in the representation. However, while the space of representations is a manifold of probability distributions, the model is not truly probabilistic in that it does not model asymmetries and relations in terms of probabilities, but in terms of asymmetric comparison functions such as the originally proposed KL divergence and the recently proposed *thresholded divergences* [2].

Probabilistic models are especially compelling for modeling ontologies, entailment graphs, and knowledge graphs. Their desirable properties include an ability to remain consistent in the presence of noisy data, suitability towards semi-supervised training using the expectations and uncertain labels present in these large-scale applications, the naturality of representing the inherent uncertainty of knowledge they store, and the ability to answer complex queries involving more than 2 variables. Note that the final one requires a true joint probabilistic model with a tractable inference procedure, not something provided by e.g. matrix factorization.

In this chapter, We take the dual approach to density-based embeddings and model uncertainty about relationships and attributes as explicitly probabilistic, while basing the probability on a latent space of geometric objects that obey natural structural biases for modeling transitive, asymmetric relations. The most similar work are the probabilistic order embeddings (POE) of Lai [73], which apply a probability measure to each order embedding's forward cone (the set of points greater than the embedding in each dimension), assigning a finite and normalized volume to the unbounded space. However, POE suffers severe limitations as a probabilistic model, including an inability to model negative correlations between concepts, which motivates the construction of our box lattice model.

Our model represents objects, concepts, and events as high-dimensional products-of-intervals (*hyperrectangles* or *boxes*), with an event's unary probability coming from the box volume and joint probabilities coming from overlaps. This contrasts with POE's approach of defining events as the forward cones of vectors, extending to infinity, integrated under a probability measure that assigns them finite volume.

One desirable property of a structured representation for ordered data, originally noted in [150] is a "slackness" shared by OE, POE, and our model: when the model predicts an "edge" or lack thereof (i.e. $P(a|b) = 0$ or $1$, or a zero constraint violation in the case of OE), being exposed to that fact again will not update the model. Moreover, there are large degrees of freedom in parameter space that exhibit this slackness, giving the model the ability to embed complex structure with 0 loss when compared to models based on symmetric inner products or distances between embeddings, e.g. bilinear GLMs [31], Trans-E [19], and other embedding models which must always be pushing and pulling parameters towards and away from each other.

Our experiments demonstrate the power of our approach to probabilistic ordering-biased relational modeling. First, we investigate an instructive 2-dimensional toy dataset that both demonstrates the way the model self organizes its box event space, and enables sensible answers to queries involving arbitrary numbers of variables, despite being trained on only pairwise data. We achieve a new state of the art in denotational probability modeling on the Flickr entailment dataset [73], and a matching state-of-the-art on WordNet hypernymy [150][96] with the concurrent work on thresholded Gaussian embedding of [2], achieving our best results by training on additional co-occurrence expectations aggregated from leaf types.

We find that the strong empirical performance of probabilistic ordering models, and our box lattice model in particular, and their endowment of new forms of training and querying, make them a promising avenue for future research in representing structured knowledge.

## 2.2 Background

We begin with a brief review of some definitions from order theory, a useful formalism for describing ontologies, then we introduce the vector lattices — order embeddings and its probabilistic extension — probabilistic order embeddings.

### 2.2.1 Partial Orders and Lattices

A non-strict *partially ordered set* (*poset*) is a pair $P, \preceq$, where $P$ is a set, and $\preceq$ is a binary relation. For all $a, b, c \in P$,

**Reflexivity**: $a \preceq a$

**Antisymmetry**: $a \preceq b \preceq a$ implies $a = b$

**Transitivity**: $a \preceq b \preceq c$ implies $a \preceq c$

This generalizes the standard concept of a totally ordered set to allow some elements to be incomparable. Posets provide a good formalism for the kind of acyclic directed graph data found in many knowledge bases with transitive relations.

A *lattice* is a poset where any subset of elements has a single unique least upper bound, and greatest lower bound. In a *bounded lattice*, the set $P$ contains two additional elements, $\top$ (*top*), and $\bot$ (*bottom*), which denote the least upper bound and greatest lower bound of the entire set.

A lattice is equipped with two binary operations, $\vee$ (*join*), and $\wedge$ (*meet*). $a \vee b$ denotes the least upper bound of $a, b \in P$, and $a \wedge b$ denotes their greatest lower bound. A bounded lattice must satisfy these properties:

**Idempotency**: $a \wedge a = a \vee a = a$

**Commutativity**: $a \wedge b = b \wedge a$ and $a \vee b = b \vee a$

**Associativity**: $a \wedge b \wedge c = a \wedge (b \wedge c)$ and $(a \vee b \vee c) = a \vee (b \vee c)$

**Absorption**: $a \vee (a \wedge b) = a$ and $a \wedge (a \vee b) = a$

**Bounded**: $\bot \preceq a \preceq \top$

Note that the extended real numbers, $\mathbb{R} \cup \{-\infty, \infty\}$, form a bounded lattice (and in fact, a totally ordered set) under the $\min$ and $\max$ operations as the meet ($\wedge$) and join ($\vee$) operations. So do sets partially ordered by inclusion, with $\cap$ and $\cup$ as $\wedge$ and $\vee$. Thinking of these special cases gives the intuition for the fourth property, *absorption*.

The $\wedge$ and $\vee$ operations can be swapped, along with reversing the poset relation $\preceq$, to give a valid lattice, called the *dual lattice*. In the real numbers this just corresponds to a sign change. A *semilattice* has only a meet or join, but not both.

**Note.** In the rest of the thesis, when the context is clear, we will also use $\wedge$ and $\vee$ to denote $\min$ and $\max$ of real numbers, in order to clarify the intuition behind our model.

### 2.2.2 Order Embeddings as Vector lattice (OE)

[150] introduced a method for embedding partially ordered sets and a task, *partial order completion*, an abstract term for things like hypernym or entailment prediction (learning transitive relations). The goal is to learn a mapping from the partially-ordered data domain to some other partially-ordered space that will enable generalization.

**Definition 1.** *[150]*
*A function* $f : (X, \preceq_X) \to (Y, \preceq_Y)$ *is an* order-embedding *if for all* $u, v \in X$

$$u \preceq_X v \iff f(u) \preceq_Y f(v)$$

They choose $Y$ to be a vector space, and the order $\preceq_Y$ to be based on the *reverse product order* on $\mathbb{R}^n_+$, which specifies

$$x \preceq y \iff \forall i \in \{1..n\},\ \ x_i \geq y_i$$

so an embedding is below another in the hierarchy if all of the coordinates are larger, and 0 provides a top element.

Although [150] do not explicitly discuss it, their model does not just capture partial orderings, but is a standard construction of a vector (Hilbert) lattice, in which the operations of meet and join can be defined as taking the pointwise maximum and minimum of two vectors, respectively [164]. This observation is also used in [80] to generate extra constraints for training order embeddings.

Under this order, meet and join operations are pointwise $\min$ and $\max$, which gives a lattice structure. In this formalism, the Order Embeddings of [150] embed partial orders as vectors using the *reverse* product order, corresponding to the dual lattice, and restrict the vectors to be positive. The vector of all zeroes represents $\top$, and embedded objects become "more specific" as they get farther away from the origin.

### 2.2.3 Probabilistic Order Embeddings (POE)

[73] built on the "region" idea to derive a probabilistic formulation (which we will refer to as POE) to model entailment probabilities in a consistent, hierarchical way.

Noting that all of OE's regions obviously have the same infinite area under the standard (Lebesgue) measure of $\mathbb{R}^n_+$, they propose a probabilistic interpretation where the Bernoulli probability of each concept $a$ or joint set of concepts $\{a, b\}$ with corresponding vectors $\{x, y\}$ is given by its volume under the exponential measure:

$$p(a) = \exp(-\sum_i x_i) = \int_{z \preceq x} \exp(-\|z\|_1) dz$$

$$p(a, b) = p(x \wedge y) = \exp(-\| \max(x_i, y_i)\|_1)$$

since the meet of two vectors is simply the intersection of their area cones, and replacing sums with $\ell_1$ norms for brevity since all coordinates are positive. While having the intuition of measuring the areas of cones, this also automatically gives a valid probability distribution over concepts since this is just the product likelihood under a coordinatewise exponential distribution.

However, they note a deficiency of their model — it can only model positive (Pearson) correlations between concepts (Bernoulli variables).

Consider two Bernoulli variables $a$ and $b$, whose probabilities correspond to the areas of cones $x$ and $y$. Recall the Bernoulli covariance formula (we will deal with covariances instead of correlations when convenient, since they always have the same sign):

$$\text{cov}(a, b) = p(a, b) - p(a)p(b) =$$

$$\exp(-\| \max(x_i, y_i)\|_1) - \exp(-\|x_i + y_i\|_1)$$

Since the sum of two positive vectors can only be greater than the sum of their pointwise maximum, this quantity will always be nonnegative. This has real consequences for probabilistic modeling in KBs: conditioning on more concepts will only make probabilities higher (or unchanged), e.g. $p(\text{dog}|\text{plant}) \geq p(\text{dog})$.

## 2.3 Method

We develop a probabilistic model for lattices based on hypercube embeddings that can model both positive and negative correlations. Before describing this, we first motivate our choice to abandon OE/POE type cone-based models for this purpose.

### 2.3.1 Correlations from Cone Measures

**Claim.** *For a pair of Bernoulli variables $p(a)$ and $p(b)$, cov$(a,b) \geq 0$ if the Bernoulli probabilities come from the volume of a cone as measured under any product (coordinate-wise) probability measure $p(x) = \prod_i^n p_i(x_i)$ on $\mathbb{R}^n$, where $F_i$, the associated CDF for $p_i$, is monotone increasing.*

*Proof.* For any product measure we have

$$\int_{z \preceq x} p(z)dz = \prod_i^n \int_{x_i \leq z_i} p_i(z_i)dz_i = \prod_i^n 1 - F_i(x_i)$$

This is just the area of the unique box corresponding to $\prod_i^n [F_i(x_i), 1] \in [0,1]^n$, under the uniform measure. This box is unique as a monotone increasing univariate CDF is bijective with $(0,1)$ — cones in $\mathbb{R}^n$ can be invertibly mapped to boxes of equivalent measure inside the unit hypercube $[0,1]^n$. These boxes have only half their degrees of freedom, as they have the form $[F_i(x_i), 1]$ per dimension, (intuitively, they have one end "stuck at infinity" since the cone integrates to infinity.

So W.L.O.G. we can consider two transformed cones $x$ and $y$ corresponding to our Bernoulli variables $a$ and $b$, and letting $F_i(x_i) = u_i$ and $F_i(y_i) = v_i$, their intersection in the unit hypercube is $\prod_i^n [\max(u_i, v_i), 1]$.

Pairing terms in the right-hand product, we have

$$\begin{aligned} &p(a,b) - p(a)p(b) = \\ &\prod_i^n (1 - \max(u_i, v_i)) - \prod_i^n (1 - u_i)(1 - v_i) \geq 0 \end{aligned}$$

since the right contains all the terms of the left and can only grow smaller. This argument is easily modified to the case of the nonnegative orthant, *mutatis mutandis*. □

An open question for future work is what non-product measures this claim also applies to. Note that some non-product measures, such as multivariate Gaussian, can be transformed into product measures easily (*whitening*) and the above proof would still apply. It seems probable that some measures, nonlinearly entangled across dimensions, could encode negative correlations in cone volumes. However, it is not generally tractable to integrate high-dimensional cones under arbitrary non-product measures.

### 2.3.2 Box Lattices

The above proof gives us intuition about the possible form of a better representation. Cones can be mapped into boxes within the unit hypercube while preserving their measure, and the lack of negative correlation seems to come from the fact that they always have an overly-large intersection due to "pinning" the maximum in each dimension to 1. To remedy this, we propose to learn representations in the space of *all* boxes (axis-aligned hyperrectangles), gaining back an extra degree of freedom. These representations can be learned with a suitable probability measure in $\mathbb{R}^n$, the nonnegative orthant $\mathbb{R}^n_+$, or directly in the unit hypercube with the uniform measure, which we elect.

We associate each concept with 2 vectors, the minimum and maximum value of the box at each dimension. Practically for numerical reasons these are stored as a minimum, a positive offset plus an $\epsilon$ term to prevent boxes from becoming too small and underflowing.

Let us define our box embeddings as a pair of vectors in $[0, 1]^n$, $(x_m, x_M)$, representing the maximum and minimum at each coordinate.

Then we can define a partial ordering by inclusion of boxes, and a lattice structure as

$$\begin{aligned}
&x \wedge y = \bot \ \text{ if } x \text{ and } y \text{ disjoint, else} \\
&x \wedge y = \prod_i [\max(x_{m,i}, y_{m,i}), \min(x_{M,i}, y_{M,i})] \\
&x \vee y = \prod_i [\min(x_{m,i}, y_{m,i}), \max(x_{M,i}, y_{M,i})]
\end{aligned}$$

where the meet is the intersecting box, or bottom (the empty set) where no intersection exists, and join is the smallest enclosing box. This lattice, considered on its own terms as a non-probabilistic object, is strictly more general than the order embedding lattice in any dimension, which is proven in Appendix A.2.

However, the finite sizes of all the lattice elements lead to a natural probabilistic interpretation under the uniform measure. Joint and marginal probabilities are given by the volume of the (intersection) box. For concept $a$ with associated box $(x_m, x_M)$, probability is simply $p(a) = \prod_i^n (x_{M,i} - x_{m,i})$ (under the uniform measure). $p(\bot)$ is of course zero since no probability mass is assigned to the empty set.

It remains to show that this representation can represent both positive and negative correlations.

**Claim.** *For a pair of Bernoulli variables $p(a)$ and $p(b)$, corr$(a, b)$ can take on any value in $[-1, 1]$ if the probabilities come from the volume of associated boxes in $[0, 1]^n$.*

*Proof.* Boxes can clearly model disjointness (exactly $-1$ correlation if the total volume of the boxes equals 1). Two identical boxes give their concepts exactly correlation 1. The area of the meet is continuous with respect to translations of intersecting boxes, and all other terms in correlation stay constant, so by continuity of the correlation function our model can achieve all possible correlations for a pair of variables. □

This proof can be extended to boxes in $\mathbb{R}^n$ with product measures by the previous reduction.

**Limitations:** Note that this model cannot perfectly describe all possible probability distributions or concepts as embedded objects. For example, the complement of a box is not a box. However, queries about complemented variables can be calculated by the Inclusion-Exclusion principle, made more efficient by the fact that all non-negated terms can be grouped and calculated exactly. We show some toy exact calculations with negated variables in Appendix A.1. Also, note that in a knowledge graph often true complements are not required — for example *mortal* and *immortal* are not actually complements, because the concept *color* is neither.

Additionally, requiring the total probability mass covered by boxes to equal 1, or exactly matching marginal box probabilities while modeling all correlations is a difficult box-packing-type problem and not generally possible. Modeling limitations aside, the union of boxes having mass $< 1$ can be seen as an open-world assumption on our KB (not all points in space have corresponding concepts, yet).

### 2.3.3 Learning

While inference (calculation of pairwise joint, unary marginal, and pairwise conditional probabilities) is quite straightforward by taking intersections of boxes and computing volumes (and their ratios), learning does not appear easy at first glance. While the (sub)gradient of the joint probability is well defined when boxes intersect, it is non-differentiable otherwise. Instead we optimize a lower bound.

Clearly $p(a \vee b) \geq p(a \cup b)$, with equality only when $a = b$, so this can give us a lower bound:

$$\begin{aligned} p(a \wedge b) &= p(a) + p(b) - p(a \cup b) \\ &\geq p(a) + p(b) - p(a \vee b) \end{aligned}$$

Where probabilities are always given by the volume of the associated box. This lower bound always exists and is differentiable, even when the joint is not. It is guaranteed to be nonpositive except when $a$ and $b$ intersect, in which case the true joint likelihood should be used.

While a negative bound on a probability is odd, inspecting the bound we see that its gradient will push the enclosing box to be smaller, while increasing areas of the individual boxes, until they intersect, which is a sensible learning strategy.

Since we are working with small probabilities it is advisable to negate this term and maximize the negative logarithm:

$$-\log(p(a \vee b) - p(a) - p(b))$$

This still has an unbounded gradient as the lower bound approaches 0, so it is also useful to add a constant within the logarithm function to avoid numerical problems.

Since the likelihood of the full data is usually intractable to compute as a conjunction of many negations, we optimize binary conditional and unary marginal terms separately by maximum likelihood.

In this work, we parametrize the boxes as $(\text{min}, \Delta = \text{max} - \text{min})$, with Euclidean projections after gradient steps to keep our parameters in the unit hypercube and maintain the minimum/delta constraints.

Now that we have the ability to compute probabilities and (surrogate) gradients for arbitrary marginals in the model, and by extension conditionals, we will see specific examples in the experiments.

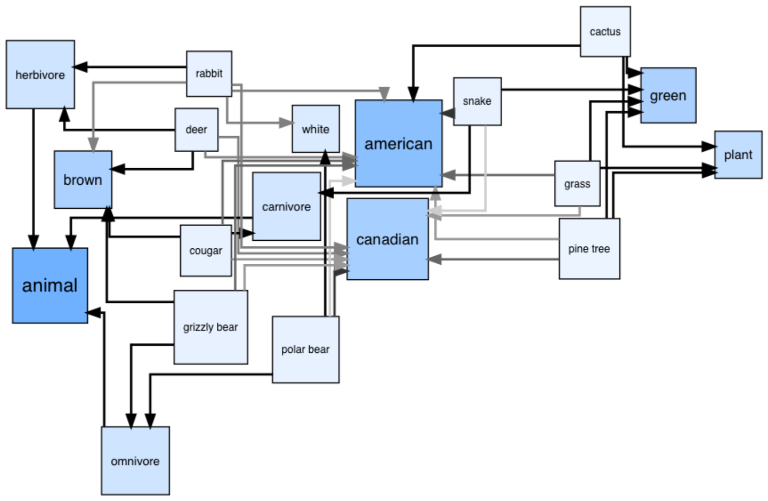
## 2.4 Experiments

### 2.4.1 Warmup: 2D Embedding of a Toy Lattice

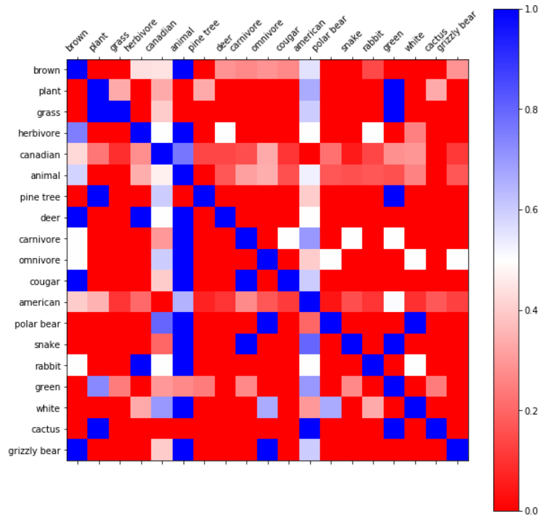
We begin by investigating properties of our model in modeling a small toy problem, consisting of a small hand constructed ontology over 19 concepts, aggregated from atomic synthetic examples first into a probabilistic lattice (e.g. some rabbits are brown, some are white), and then a full CPD. We model it using only 2 dimensions to enable visualization of the way the model self-organizes its “event space", training the model by minimize weighted cross-entropy with both the unary marginals and pairwise conditional probabilities. We also conduct a parallel experiment with POE as embedded in the unit cube, where each representation is constrained to touch the faces $x = 1, y = 1$. In Figure 2.2, we show the representation of lattice structures by POE and the box lattice model as compared to the abstract probabilistic lattice used to construct the data, shown in Figure 2.1, and compare the conditional probabilities produced by our model to the ground truth, demonstrating the richer capacity of the box model in capturing strong positive and negative correlations. In Table 2.1, we perform a series of multivariable conditional queries and demonstrate intuitive results on high-order queries containing up to 4 variables, despite the model being trained on only 2-way information.

### 2.4.2 WordNet

We experiment on WordNet hypernym prediction, using the same train, development and test split as [150], created by randomly taking 4,000 hypernym pairs from the 837,888-edge transitive closure of the WordNet hypernym hierarchy as positive training examples for the development set, 4,000 for the test set, and using the rest as training data. Negative

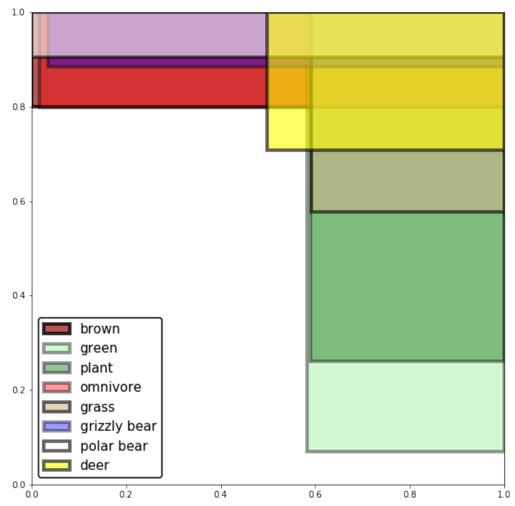


(a) Original lattice

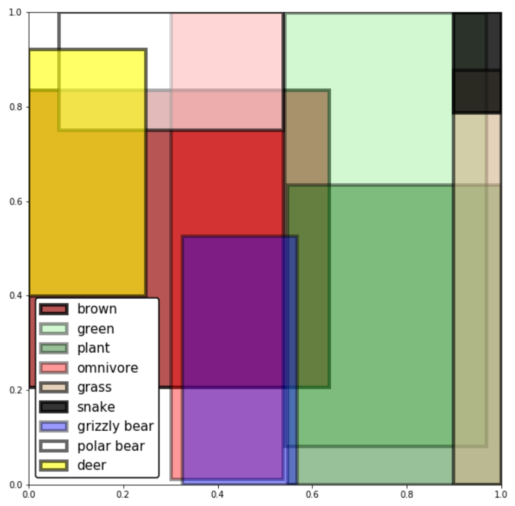


(b) Ground truth CPD

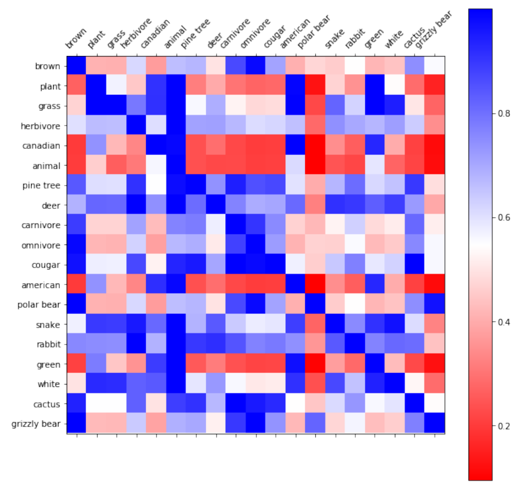
Figure 2.1: Representation of the toy probabilistic lattice used in Section 2.4.1. Darker color corresponds to more unary marginal probability. The associated CPD is obtained by a weighted aggregation of leaf elements.



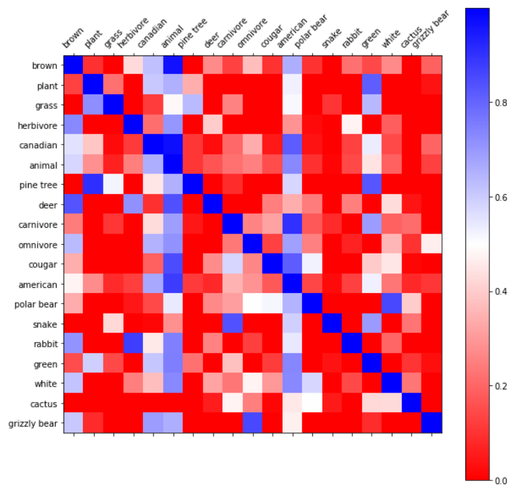
(a) POE lattice



(b) Box lattice



(c) POE CPD



(d) Box CPD

Figure 2.2: Lattice representations and conditional probabilities from POE vs. box lattice. Note how the box lattice model's lack of "anchoring" to a corner allows it vastly more expressivity in matching the ground truth CPD seen in Figure 2.1.

| P(grizzly bear \| ... ) | | P(cactus \| ... ) | | P(plant \| ... ) | |
|---|---|---|---|---|---|
| P(grizzly bear) | 0.12 | P(cactus) | 0.10 | P(plant) | 0.20 |
| omnivore | 0.29 | green | 0.16 | green | 0.37 |
| white | 0.00 | plant | 0.39 | snake | 0.00 |
| brown | 0.30 | american, green | 0.19 | carnivore | 0.00 |
| omnivore, white | 0.00 | plant, green, american | 0.40 | cactus | 0.78 |
| omnivore, brown | 0.38 | american, carnivore | 0.00 | american, cactus | 0.85 |

Table 2.1: Multi-way queries: conditional probabilities adjust when adding additional evidence or contradiction. In constrast, POE can only raise or preserve probability when conditioning.

| term1 | term2 |
|---|---|
| craftsman.n.02 | shark.n.03 |
| homogenized_milk.n.01 | apple_juice.n.01 |
| tongue_depresser.n.01 | paintbrush.n.01 |
| deerstalker.n.01 | bathing_cap.n.01 |
| skywriting.n.01 | transcript.n.01 |

Table 2.2: Negatively correlated variables produced by the model.

| Method | Test Accuracy % |
|---|---|
| transitive | 88.2 |
| word2gauss | 86.6 |
| OE | 90.6 |
| [80] | 91.3 |
| DOE (KL) | **92.3** |
| POE | 91.6 |
| POE (100 dim) | 91.7 |
| Box | 92.2 |
| Box + CPD | **92.3** |

Table 2.3: Classification accuracy on WordNet test set.

training examples are created by randomly corrupting a train/development/test edge $(u, v)$ by replacing either $u$ or $v$ with a randomly chosen negative node. We use their specific train/dev/test split, while [2] use a different train/dev split with the same test set (personal communication) to examine the effect of different negative sampling techniques. We cite their best performing model, called DOE (KL).

Since our model is probabilistic, we would like a sensible value for $P(n)$, where $n$ is a node. We assign these marginal probabilities by looking at the number of descendants in the hierarchy under a node, and normalizing over all nodes, taking $P(n) = \frac{|\textit{ descendants(n) }|}{|\textit{ nodes }|}$.

Furthermore, we use the graph structure (only of the subset of edges in the training set to avoid leaking data) to augment the data with approximate conditional probabilities $P(x|y)$. For each leaf, we consider all of its ancestors as pairwise co-occurences, then aggregate and divide by the number of leaves to get an approximate joint probability distribution, $P(x, y) = \frac{|\textit{ x, y co-occur in ancestor set }|}{|\textit{ leaves }|}$. With this and the unary marginals, we can create a conditional probability table, which we prune based on the difference of $P(x|y)$ and $P(y|x)$ and add cross entropy with these conditional "soft edges" to the training data. We refer to experiments using this additional data as Box + CPD in Table 3.4.

We use 50 dimensions in our experiments. Since our model has 2 parameters per dimension, we also perform an apples-to-apples comparison with a 100D POE model. As seen in Table 3.4, we outperform POE significantly even with this added representational power. We also observe sensible negatively correlated examples, shown in 2.2, in the trained box model, while POE cannot represent such relationships. We tune our models on the development set, with parameters documented in Appendix A.4.1. We observe that not only does our model outperform POE, it beats all previous results on WordNet, aside from the concurrent work of [2] (using different train/dev negative examples), the baseline POE model does as well. This indicates that probabilistic embeddings for transitive relations are a promising avenue for future work. Additionally, the ability of the model to learn from the expected "soft edges" improves it to state-of-the-art level. We expect that co-occurrence counts gathered from real textual corpora, rather than merely aggregating up the WordNet lattice, would further strengthen this effect.

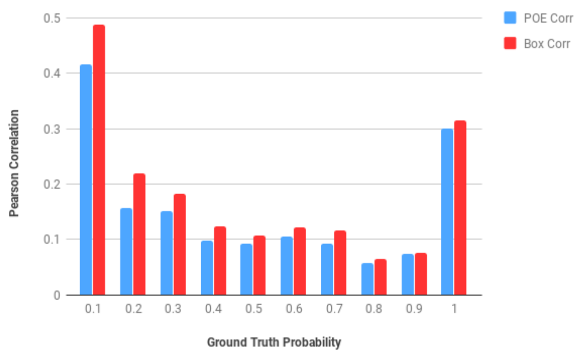


Figure 2.3: R between model and gold probabilities.

| | $P(x\|y)$ | |
|---|---|---|
| *Full test data* | KL | Pearson R |
| POE | 0.031 | 0.949 |
| POE* | 0.031 | 0.949 |
| Box | **0.020** | **0.967** |
| *Unseen pairs* | | |
| POE | 0.048 | 0.920 |
| POE* | 0.046 | 0.925 |
| Box | **0.025** | **0.957** |
| *Unseen words* | | |
| POE | 0.127 | 0.696 |
| POE* | 0.084 | 0.854 |
| Box | **0.050** | **0.900** |

Table 2.4: KL and Pearson correlation between model and gold probability.

### 2.4.3 Flickr Entailment Graph

We conduct experiments on the large-scale Flickr entailment dataset of 45 million image caption pairs. We use the exactly same train/dev/test from [73]. We use a slightly different unseen word pairs and unseen words test data, obtained from the author. We include their published results and also use their published code, marked $*$, for comparison.

For these experiments, we relax our boxes from the unit hypercube to the nonnegative orthant and obtain probabilities under the exponential measure, $p(x) = \exp(-x)$. We enforce the nonnegativity constraints by clipping the LSTM-generated embedding [57] for the box minimum with a ReLU, and parametrize our $\Delta$ embeddings using a softplus activation to prevent dead units. As in [73], we use 512 hidden units in our LSTM to compose sentence vectors. We then apply two single-layer feed-forward networks with 512 units applied to the final LSTM state to produce the embeddings.

As we can see from Table 3.6, we note large improvements in KL and Pearson correlation to the ground truth entailment probabilities. In further analysis, Figure 2.3 demonstrates that while the box model outperforms POE in nearly every regime, the highest gains come from the comparatively difficult to calibrate small entailment probabilities, indicating the greater capability of our model to produce fine-grained distinctions.

## 2.5 Related Work

In addition to the related work in structured embeddings mentioned in the introduction, our focus on directed, transitive relational modeling and ontology induction shares much with the rich field of directed graphical models and causal modeling [112], as well as learning the structure of those models [55]. Work in undirected structure learning such the Graphical Lasso [43] is also relevant due to our desire to learn from pairwise joint/conditional probabilities and moment matrices, which are closely related in the setting of discrete variables.

Especially relevant research in Bayesian networks are applications towards learning taxonomic structure of relational data [6], although this work is often restricted towards tree-shaped ontologies, which allow efficient inference by Chu-Liu-Edmonds' algorithm [29], while we focus on arbitrary DAGs.

As our model is based on populating a latent "event space" into boxes (products of intervals), it is especially reminiscent of the Mondrian process [127]. However, the Mondrian process partitions the space as a high dimensional tree (a non-parametric kd-tree), while our model allows the arbitrary box placement required for DAG structure, and is much more tractable in high dimensions compared to the Mondrian's Bayesian non-parametric inference.

Embedding applications to relational learning constitute a huge field to which it is impossible to do justice, but one general difference between our approaches is that e.g. a matrix factorization model treats the embeddings as objects to score relation links with, as opposed to POE or our model in which embeddings represent subsets of probabilistic event space which are directly integrated. They are full probabilistic models of the joint set of variables, rather than embedding-based approximations of only low-order joint and conditional probabilities. That is, any set of our parameters can answer any arbitrary probabilistic question (possibly requiring intractable computation), rather than being fixed to modeling only certain subsets of the joint.

Embedding-based learning's large advantage over the combinatorial structure learning presented by classical PGM approaches is its applicability to large-scale probability distributions containing hundreds of thousands of events or more, as in both our WordNet and Flickr experiments.

# CHAPTER 3

# SMOOTHING THE GEOMETRY OF PROBABILISTIC BOX EMBEDDINGS.

## 3.1 Introduction

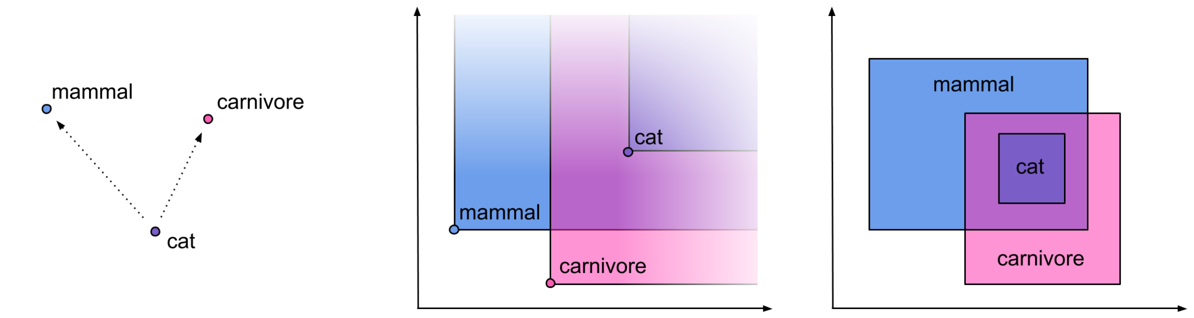
In the previous Chapter, we introduce probabilistic box embeddings. We show its strong empirical performance in modeling transitive relations, probabilistic interpretation (edges in a relational DAG are replaced with conditional probabilities), and ability to model complex joint probability distributions including negative correlations. Box embeddings (BE) are a generalization of order embeddings (OE) [150] and probabilistic order embeddings (POE) [73] that replace the vector lattice ordering (notions of overlapping and enclosing convex cones) in OE and POE with a more general notion of overlapping boxes (products of intervals).

While intuitively appealing, the “hard edges” of boxes and their ability to become easily disjoint, present difficulties for gradient-based optimization: when two boxes are disjoint in the model, but have overlap in the ground truth, no gradient can flow to the model to correct the problem. This is of special concern for (pseudo-)sparse data, where many boxes should have nearly zero overlap, while others should have very high overlap. This is especially pronounced in the case of e.g. market basket models for recommendation, where most items should not be recommended, and entailment tasks, most of which are currently artificially resampled into a 1:1 ratio of positive to negative examples. To address the disjoint case, [151] introduce an ad-hoc surrogate function. In contrast, we look at this problem as inspiration for a new model, based on the intuition of relaxing the hard edges of the boxes into smoothed density functions, using a Gaussian convolution with the original boxes.

We demonstrate the superiority of our approach to modeling transitive relations on WordNet, Flickr caption entailment, and a MovieLens-based market basket dataset. We match or beat existing state of the art results, while showing substantial improvements in the pseudosparse regime.

## 3.2 Method

### 3.2.1 Box Lattice



(a) Ontology (b) Order Embeddings (c) Box Embeddings

Figure 3.1: Comparison between the Order Embedding (vector lattice) and Box Embedding representations for a simple ontology. Regions represent concepts and overlaps represent their entailment. Shading represents density in the probabilistic case.

We covered the background of lattice theory and vector lattice in Chapter 2.2. Using the same notation, box lattice represents each concept in a knowledge graph is associated with two vectors, the minimum and maximum coordinates of an axis-aligned hyperrectangle, or *box* (product of intervals).

Using the notion of set inclusion between boxes, there is a natural partial order and lattice structure. To represent a box $\mathbf{x}$, let the pairs $(x_{m,i}, x_{M,i})$ be the maximum and minimum of the interval at each coordinate $i$. Then the box lattice structure (least upper bounds and greatest lower bounds), with $\vee$ and $\wedge$ denoting $\max$ and $\min$ when applied to the scalar coordinates, is

$$\mathbf{x} \wedge \mathbf{y} = \bot \text{ if } x \text{ and } y \text{ disjoint, else}$$
$$\mathbf{x} \wedge \mathbf{y} = \prod_i [x_{m,i} \vee y_{m,i}, x_{M,i} \wedge y_{M,i}]$$
$$\mathbf{x} \vee \mathbf{y} = \prod_i [x_{m,i} \wedge y_{m,i}, x_{M,i} \vee y_{M,i}]$$

Here, $\prod$ denotes a set (cartesian) product — the lattice meet is the largest box contained entirely within both $\mathbf{x}$ and $\mathbf{y}$, or bottom (the empty set) where no intersection exists, and the lattice join is the smallest box containing both $\mathbf{x}$ and $\mathbf{y}$.

To associate a measure, marginal probabilities of (collections of) events are given by the volume of boxes, their complements, and intersections under a suitable probability measure. Under the uniform measure, if event $\mathbf{x}$ has an associated box with interval boundaries $(x_m, x_M)$, the probability $p(\mathbf{x})$ is given by $\prod_i^n (x_{M,i} - x_{m,i})$. Use of the uniform measure requires the boxes to be constrained to the unit hypercube, so that $p(\mathbf{x}) \leq 1$. $p(\bot)$ is taken to be zero, since $\bot$ is an empty set. As boxes are simply special cases of sets, it is intuitive that this is a valid probability measure, but it can also be shown to be compatible with the meet semilattice structure in a precise sense [75].

Figure 3.1c demonstrates a toy, two-dimensional example of both Order Embeddings and Box Embeddings representations of a simple ontology.

### 3.2.2 Motivation: Optimization and Sparse Data

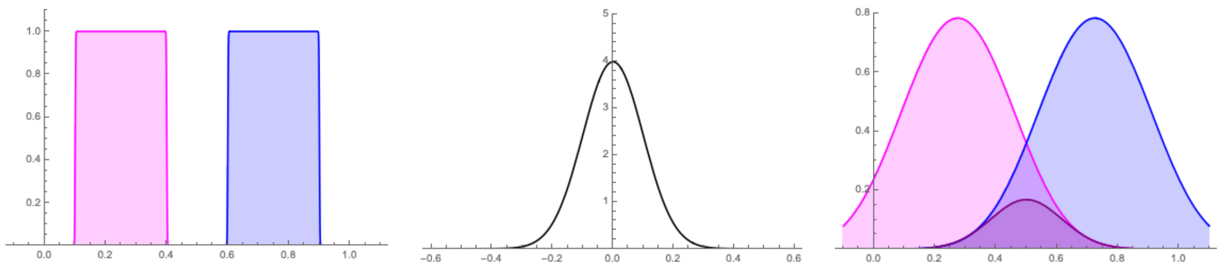
When using gradient-based optimization to learn box embeddings, an immediate problem identified in the original work is that when two concepts are incorrectly given as disjoint by the model, no gradient signal can flow since the meet (intersection) is exactly zero, with zero derivative. To see this, note that for a pair of 1-dimensional boxes (intervals), the volume of the meet under the uniform measure $p$ as given in Section 3.2.1 is

$$p(\mathbf{x} \wedge \mathbf{y}) = m_h(\min(x_M, y_M) - \max(x_m, y_m)) \tag{3.1}$$

where $m_h$ is the standard hinge function, $m_h(x) = 0 \vee x = \max(0, x)$.

The hinge function has a large flat plateau at $0$ when intervals are disjoint. This issue is especially problematic when the lattice to be embedded is (pseudo-)sparse, that is, most boxes should have very little or no intersection, since if training accidentally makes two boxes disjoint there is no way to recover with the naive measure. The authors propose a surrogate function to optimize in this case, but we will use a more principled framework to develop alternate measures that avoid this pathology, improving both optimization and final model quality.

### 3.2.3 Relaxed Geometry



(a) Unsmoothed Indicators (b) Convolution Kernel (c) Smoothed w/ Overlap

Figure 3.2: One-dimensional example demonstrating two disjoint indicators of intervals before and after the application of a smoothing kernel. The area under the purple product curve is proportional to the degree of overlap.

The intuition behind our approach is that the “hard edges” of the standard box embeddings lead to unwanted gradient sparsity, and we seek a relaxation of this assumption that maintains the desirable properties of the base lattice model while enabling better optimization and preserving a geometric intuition. For ease of exposition, we will refer to 1-dimensional intervals in this section, but the results carry through from the representation of boxes as products of intervals and their volumes under the associated product measures.

The first observation is that, considering boxes as indicator functions of intervals, we can rewrite the measure of the joint probability $p(\mathbf{x} \wedge \mathbf{y})$ between intervals $\mathbf{x} = [a, b]$ and

$\mathbf{y} = [c, d]$ as an integral of the product of those indicators:

$$p(\mathbf{x} \wedge \mathbf{y}) = \int_{\mathbb{R}} \mathbb{1}_{[a,b]}(x)\mathbb{1}_{[c,d]}(x)dx$$

since the product has support (and is equal to 1) only in the areas where the two intervals overlap.

A solution suggests itself in replacing these indicator functions with functions of infinite support. We elect for kernel smoothing, specifically convolution with a normalized Gaussian kernel, equivalent to an application of the diffusion equation to the original functional form of the embeddings (indicator functions) and a common approach to mollified optimization and energy smoothing [102, 50, 98]. This approach is demonstrated in one dimension in Figure 3.2.

Specifically, given $\mathbf{x} = [a, b]$, we associate the smoothed indicator function

$$f(x; a, b, \sigma^2) = \mathbb{1}_{[a,b]}(x) * \phi(x; \sigma^2) = \int_{\mathbb{R}} \mathbb{1}_{[a,b]}(z)\phi(x - z; \sigma^2)dz = \int_a^b \phi(x - z; \sigma^2)dz$$

We then wish to evaluate, for two lattice elements $\mathbf{x}$ and $\mathbf{y}$ with associated smoothed indicators $f$ and $g$,

$$p_\phi(\mathbf{x} \wedge \mathbf{y}) = \int_{\mathbb{R}} f(x; a, b, \sigma_1^2)g(x; c, d, \sigma_2^2)dx \tag{3.2}$$

This integral admits a closed form solution.

**Proposition 1.** *Let* $m_\Phi(x) = \int \Phi(x)dx$ *be an antiderivative of the standard normal CDF. Then the solution to* (3.2) *is given by,*

$$p_\phi(\mathbf{x} \wedge \mathbf{y}) = \sigma \left( m_\Phi(\tfrac{b-c}{\sigma}) + m_\Phi(\tfrac{a-d}{\sigma}) - m_\Phi(\tfrac{b-d}{\sigma}) - m_\Phi(\tfrac{a-c}{\sigma}) \right) \tag{3.3}$$

$$\approx \left( \rho \operatorname{soft}(\tfrac{b-c}{\rho}) + \rho \operatorname{soft}(\tfrac{a-d}{\rho}) \right) - \left( \rho \operatorname{soft}(\tfrac{b-d}{\rho}) + \rho \operatorname{soft}(\tfrac{a-c}{\rho}) \right) \tag{3.4}$$

*where* $\sigma = \sqrt{\sigma_1^2 + \sigma_2^2}$, $\operatorname{soft}(x) = \log(1+\exp(x))$ *is the softplus function, the antiderivative of the logistic sigmoid, and* $\rho = \frac{\sigma}{1.702}$.

*Proof.* The first line is proved in Appendix A.1, the second approximation follows from the approximation of $\Phi$ by a logistic sigmoid given in [21]. $\square$

Note that, in the *zero-temperature limit*, as $\rho$ goes to zero, we recover the formula

$$\begin{aligned} p_\phi(\mathbf{x} \wedge \mathbf{y}) &= \lim_{\rho \to 0} \left( \rho \operatorname{soft}(\tfrac{b-c}{\rho}) + \rho \operatorname{soft}(\tfrac{a-d}{\rho}) \right) - \left( \rho \operatorname{soft}(\tfrac{b-d}{\rho}) + \rho \operatorname{soft}(\tfrac{a-c}{\rho}) \right) \\ &= \left( m_h(b-c) + m_h(a-d) \right) - \left( m_h(b-d) + m_h(a-c) \right) \\ &= m_h(b \wedge d - a \vee c) \end{aligned}$$

with equality in the last line because $(a, b)$ and $(c, d)$ are intervals. This last line is exactly our original equation (3.1), which is expected from convolution with a zero-bandwidth kernel (a Dirac delta function, the identity element under convolution). This is true for both the exact formula using $\int \Phi(x)dx$, and the softplus approximation.

Unfortunately, for any $\rho > 0$, multiplication of Gaussian-smoothed indicators does not give a valid meet operation on a function lattice, for the simple reason that $f^2 \neq f$, except in the case of indicator functions, violating the idempotency requirement of Section 2.2.1.

More importantly, for practical considerations, if we are to treat the outputs of $p_\phi$ as probabilities, the consequence is

$$p_\phi(\mathbf{x}|\mathbf{x}) = \frac{p_\phi(\mathbf{x}, \mathbf{x})}{p_\phi(\mathbf{x})} = \frac{p_\phi(\mathbf{x} \wedge \mathbf{x})}{p_\phi(\mathbf{x})} \neq 1 \tag{3.5}$$

which complicates our applications that train on conditional probabilities. However, by a modification of (3.3), we can obtain a function $p$ such that $p(\mathbf{x} \wedge \mathbf{x}) = p(\mathbf{x})$, while retaining the smooth optimization properties of the Gaussian model.

Recall that for the hinge function $m_h$ and two intervals $(a, b)$ and $(c, d)$, we have

$$\big(m_h(b-c) + m_h(a-d)\big) - \big(m_h(b-d) + m_h(a-c)\big) = m_h(b \wedge d - a \vee c) \tag{3.6}$$

where the left hand side is the zero-temperature limit of the Gaussian model from (3.3). This identity is true of the hinge function $m_h$, but not the softplus function.

However, an equation with a similar functional form as (3.6) (on both the left- and right-hand sides) is true not only of the hinge function from the unsmoothed model, but also true of the softplus. For two intervals $\mathbf{x} = (a, b)$ an $\mathbf{y} = (c, d)$, by the commutativity of $\min$ and $\max$ with monotonic functions, we have

$$\big(\operatorname{soft}(b-c) \vee \operatorname{soft}(a-d)\big) \wedge \big(\operatorname{soft}(b-d) \vee \operatorname{soft}(a-c)\big) = \operatorname{soft}(b \wedge d - a \vee c) \tag{3.7}$$

In the zero-temperature limit, all terms in equations 3.3 and 3.7 are equivalent. However, outside of this, (3.7) is idempotent for $\mathbf{x} = \mathbf{y} = (a, b) = (c, d)$ (when considered as a measure of overlap, made precise in the next paragraph), while (3.3) is not.

This inspires us to define the probabilities $p(\mathbf{x})$ and $p(\mathbf{x}, \mathbf{y})$ using a normalized version of (3.7) in place of (3.3). For the interval (one-dimensional box) case, we define

$$p(\mathbf{x}) \propto \operatorname{soft}(b-a)$$
$$p(\mathbf{x}, \mathbf{y}) \propto \operatorname{soft}(b \wedge d - a \vee c)$$

which satisfies the idempotency requirement, $p(\mathbf{x}) = p(\mathbf{x}, \mathbf{x})$.

Because softplus upper-bounds the hinge function, it is capable of outputting values that are greater than 1, and therefore must be normalized. In our experiments, we use two different approaches to normalization. For experiments with a relatively small number of entities (all besides Flickr), we allow the boxes to learn unconstrained, and divide each dimension by the measured size of the global minimum and maximum $(G_m^{(i)}, G_M^{(i)})$ at that dimension

$$m_{\text{soft}}^{(i)}(x) = \frac{\text{soft}(\frac{x}{\rho})}{\text{soft}(\frac{G_m - G_m}{\rho})}$$

For data where computing these values repeatedly is infeasible, we project onto the unit hypercube and normalize by $m_{\text{soft}}(1)$. The final probability $p(\mathbf{x})$ is given by the product over dimensions

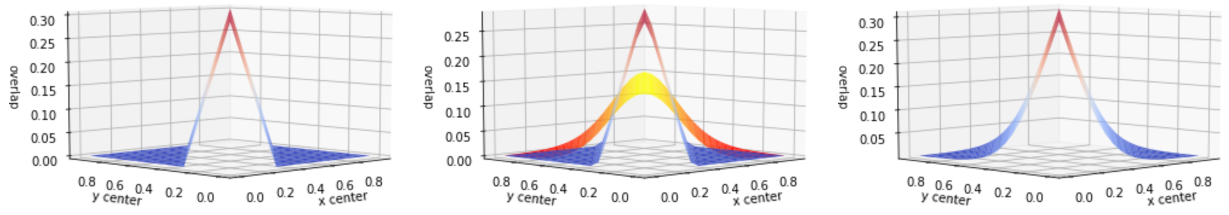
$$p(\mathbf{x}) = \prod_i m_{\text{soft}}^{(i)}(x_{M,i} - x_{m,i})$$
$$p(\mathbf{x}, \mathbf{y}) = \prod_i m_{\text{soft}}^{(i)}(x_{M,i} \wedge y_{M,i} - x_{m,i} \vee y_{m,i})$$

Note that, while equivalent in the zero temperature limit to the standard uniform probability measure of the box model, this function, like the Gaussian model, is not a valid probability measure on the entire joint space of events (the lattice). However, neither is factorization of a conditional probability table using a logistic sigmoid link function, which is commonly used for the similar tasks. Our approach retains the inductive bias of the original box model, is equivalent in the limit, and satisfies the necessary condition that $p(\mathbf{x}, \mathbf{x}) = p(\mathbf{x})$. A comparison of the 3 different functions is given in Figure 3.3, with the softplus overlap showing much better behavior for highly disjoint boxes than the Gaussian model, while also preserving the meet property.

## 3.3 Experiments

### 3.3.1 WordNet

We perform experiments on the WordNet hypernym prediction task in order to evaluate the performance of these improvements in practice. The WordNet hypernym hierarchy con-



(a) Standard (hinge) overlap (b) Gaussian overlap, $\sigma \in \{2, 6\}$ (c) Softplus overlap

Figure 3.3: Comparison of different overlap functions for two boxes of width 0.3 as a function of their centers. Note that in order to achieve high overlap, the Gaussian model must drastically lower its temperature, causing vanishing gradients in the tails.

| Method | Test Accuracy % |
|---|---|
| transitive | 88.2 |
| word2gauss | 86.6 |
| OE | 90.6 |
| [**?**] | 91.3 |
| POE | 91.6 |
| Box | 92.2 |
| Smoothed Box | 92.0 |

Table 3.4: Classification accuracy on WordNet test set.

tains 837,888-edges after performing the transitive closure on the direct edges in WordNet. We used the same train/dev/test split as in [150]. Positive examples are randomly chosen from the 837k edges, while negative examples are generated by swapping one of the terms to a random word in the dictionary. Experimental details are given in Appendix A.4.1.

The smoothed box model performs nearly as well as the original box lattice in terms of test accuracy[1]. While our model requires less hyper-parameter tuning than the original, we suspect that our performance would be increased on a task with a higher degree of sparsity than the 50/50 positive/negative split of the standard WordNet data, which we explore in the next section.

[1]Accuracy is calculated by applying the same threshold which maximized accuracy in dev set.

### 3.3.2 Imbalanced WordNet

In order to confirm our intuition that the smoothed box model performs better in the sparse regime, we perform further experiments using different numbers of positive and negative examples from the WordNet *mammal* subset, comparing the box lattice, our smoothed approach, and order embeddings (OE) as a baseline. The training data is the transitive reduction of this subset of the *mammal* WordNet, while the dev/test is the transitive closure of the training data. The training data contains 1,176 positive examples, and the dev and test sets contain 209 positive examples. Negative examples are generated randomly using the ratio stated in the table.

As we can see from the table, with balanced data, all models include OE baseline, Box, Smoothed Box models nearly match the full transitive closure. As the number of negative examples increases, the performance drops for the original box model, but Smoothed Box still outperforms OE and Box in all setting. This superior performance on imbalanced data is important for e.g. real-world entailment graph learning, where the number of negatives greatly outweigh the positives.

| Positive:Negative | Box | OE | Smoothed Box |
|---|---|---|---|
| 1:1 | 0.9905 | 0.9976 | **1.0** |
| 1:2 | 0.8982 | 0.9139 | **1.0** |
| 1:6 | 0.6680 | 0.6640 | **0.9561** |
| 1:10 | 0.5495 | 0.5897 | **0.8800** |

Table 3.5: F1 scores of the box lattice, order embeddings, and our smoothed model, for different levels of label imbalance on the WordNet *mammal* subset.

### 3.3.3 Flickr

We conduct experiments on the Flickr entailment dataset. Flickr is a large-scale caption entailment dataset containing of 45 million image caption pairs. In order to perform an apples-to-apples comparison with existing results we use the exact same dataset from [151]. In this case, we do constrain the boxes to the unit cube, using the same experimental setup as [151], except we apply the softplus function before calculating the volume of the boxes. Experimental details are given in Appendix A.4.3.

We report KL divergence and Pearson correlation on the full test data, unseen pairs (caption pairs which are never occur in training data) and unseen captions (captions which are never occur in training data). As shown in Table 3.6, we see a slight performance gain compared to the original model, with improvements most concentrated on unseen captions.

|  | $P(x\|y)$ |  |
|---|---|---|
| *Full test data* | KL | Pearson R |
| POE | 0.031 | 0.949 |
| POE* | 0.031 | 0.949 |
| Box | 0.020 | 0.967 |
| Smoothed Box | **0.018** | **0.969** |
| *Unseen pairs* |  |  |
| POE | 0.048 | 0.920 |
| POE* | 0.046 | 0.925 |
| Box | 0.025 | **0.957** |
| Smoothed Box | **0.024** | **0.957** |
| *Unseen captions* |  |  |
| POE | 0.127 | 0.696 |
| POE* | 0.084 | 0.854 |
| Box | 0.050 | 0.900 |
| Smoothed Box | **0.036** | **0.917** |

Table 3.6: KL and Pearson correlation between model and gold probability.

### 3.3.4 MovieLens

We apply our method to a market-basket task constructed using the MovieLens dataset. Here, the task is to predict users' preference for movie A given that they liked movie B. We first collect all pairs of user-movie ratings higher than 4 points (strong preference) from the MovieLens-20M dataset. From this we further prune to just a subset of movies which have more than 100 user ratings to make sure that counting statistics are significant enough. This leads to 8545 movies in our dataset. We calculate the conditional probability $P(A|B) =$

$\frac{P(A,B)}{P(B)} = \frac{\#rating(A,B)_{>4}/\#users}{\#rating(B)_{>4}/\#users}$. We randomly pick 100K conditional probabilities for training data and 10k probabilities for dev and test data [2].

We compare with several baselines: low-rank matrix factorization, complex bilinear factorization [148], and two hierarchical embedding methods, POE [73] and the Box Lattice [151]. Since the training matrix is asymmetric, we used separate embeddings for target and conditioned movies. For the complex bilinear model, we added one additional vector of parameters to capture the "imply" relation. We evaluate on the test set using KL divergence, Pearson correlation, and Spearman correlation with the ground truth probabilities. Experimental details are given in Appendix A.4.4.

From the results in Table 3.7, we can see that our smoothed box embedding method outperforms the original box lattice as well as all other baselines' performances, especially in Spearman correlation, the most relevant metric for recommendation, a ranking task. We perform an additional study on the robustness of the smoothed model to initialization conditions in Appendix A.3.

| | KL | Pearson R | Spearman R |
|---|---|---|---|
| Matrix Factorization | 0.0173 | 0.8549 | 0.8374 |
| Complex Bilinear Factorization | 0.0141 | 0.8771 | 0.8636 |
| POE | 0.0170 | 0.8548 | 0.8511 |
| Box | 0.0147 | 0.8775 | 0.8768 |
| Smoothed Box | **0.0138** | **0.8985** | **0.8977** |

Table 3.7: Performance of the smoothed model, the original box model, and several baselines on MovieLens.

## 3.4 Related Work

As mentioned in the introduction, there is much related work on structured or geometric embeddings. Most relevant to this work are the order embeddings of [150], which embed

[2]In the dev and test data, we also remove all the $P(A|B)$ where $P(B|A)$ appears in the training data.

a non-probabilistic DAG or lattice in a vector space with order given by inclusion of embeddings' forward cones, the probabilistic extension of that model due to [73], and the *box lattice* or *box embedding* model of [151], which we extend. Concurrently to [151], another hyperrectangle-based generalization of order embeddings was proposed by [144], also called *box embeddings*. The difference between the two models lies in the interpretation: the former is a probabilistic model that assigns edges conditional probabilities according to degrees of overlap, while the latter is a deterministic model in the style of order embeddings — an edge is considered present only if one box entirely encloses another.

Methods based on embedding points in hyperbolic space [105, 44] have also recently been proposed for learning hierarchical embeddings. These models, similar to order embeddings and the box embeddings of [144], are non-probabilistic and optimize an energy function. Additionally, while the negative curvature of hyperbolic space is attractively biased towards learning tree structures (since distances between points increase the farther they are from the origin), this constant curvature makes the models not as suitable for learning non-treelike DAGs.

Our approach to smoothing the energy landscape of the model using Gaussian convolution is common in mollified optimization and continuation methods, and is increasingly making its way into machine learning models such as Mollifying Networks [50], diffusion-trained networks [98], and noisy activation functions [49].

Our focus on embedding orderings and transitive relations is a subset of knowledge graph embedding. While this field is very large, the main difference of our probabilistic approach is that we seek to learn an embedding model which maps concepts to subsets of event space, giving our model an inductive bias especially suited for transitive relations as well as fuzzy concepts of inclusion and entailment.

# CHAPTER 4

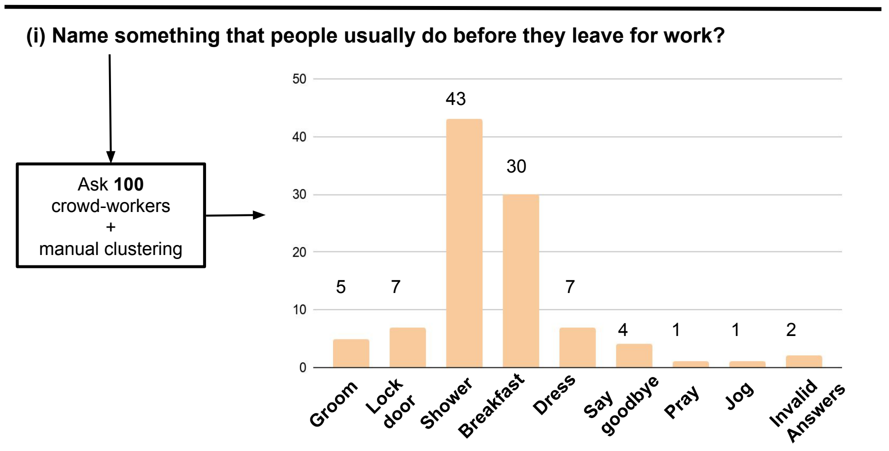
# COMMONSENSE QUESTION ANSWERING DATASET WITH PROBABILISTIC EVALUATION

## 4.1 Introduction

We introduced probabilistic box embeddings, and its improved training methods smoothed box embeddings to efficiently model commonsense concepts with probabilistic guarantees in previous Chapters. As commonsense knowledge is inherently probabilistic, we need to have a probabilistic evaluation when measuring progress towards AGI. However, existing evaluations do not reflect the probabilistic nature of commonsense knowledge. High accuracy in multiple-choice evaluation is misleading since the answer spaces are artificially constrained. To fill in the gap, we propose a method of sampling commonsense distributions from human annotators as well as a new question-answering dataset that makes probabilistic evaluation possible.

This Chapter introduces a new question answering dataset for training and evaluating common sense reasoning capabilities of artificial intelligence systems. The training set is gathered from an existing set of questions played in a long-running international game show – FAMILY-FEUD. The hidden evaluation set is created by gathering answers for each question from 100 crowd-workers. We also propose a generative evaluation task where a model has to output a ranked list of answers, ideally covering all prototypical answers for a question. After presenting multiple competitive baseline models, we find that human performance still exceeds model scores on all evaluation metrics with a meaningful gap, supporting the challenging nature of the task.

Humans possess the ability to implicitly reason using a wealth of common background knowledge, much of which is acquired through shared experiences. For example, consider



**(ii) Name a piece of equipment that you are likely to find at your office and not at home?**
Categories: *printer/copier (**37**), office furniture (**15**), computer equipment (**17**), stapler (**11**) files (**10**), office appliances (**5**), security systems (**1**)*

**(iii) Name something that an athlete would not keep in her refrigerator.**
Categories: *unhealthy / fast food (**36**), unhealthy drinks (**24**), clothing/shoes (**24**) accessories (**7**)*

**(iv) Name something that you might forget in a hotel room?**
Categories: *phone (**24**), toothbrush/towels (**17**), clothing/shoes (**15**) keys (**14**), purse/wallet (**14**), accessories (**8**), charger (**5**)*

Figure 4.1: We focus on common-sense reasoning over prototypical situations when there could be many different answers but some are more common than others. Our task is in generative style (*not* multiple-choice format). Answers to a question are crowd-sourced from 100 workers and are then manually clustered into categories. To perform well, a model has to output a ranked list of answers covering multiple categories.

the question in Figure 6.1 — "*Name something that people usually do before they leave the house for work.*". Humans can agree about the details and characteristics of a prototypical event or situation [133, 134] due to commonalities in their shared lived experiences, cultural norms and expectations. This rough agreement extends beyond an agreement on a single top response, but can be viewed as a ranked list of plausible answers, as demonstrated in Figure 6.1. Such sets of diverse answers represent the nature of common sense knowledge and may be useful in applications such as dialogue systems, where multiple responses are appropriate for a given context [172].

We present a new question/answer dataset capturing both the plausibility of the answers and the ranking preference of each answer about such prototypical situations inspired by the long-running American game show FAMILY-FEUD, which also provides the training data for the task.[1] The game show is played by prompting participants with queries such as *Name something that people usually do before they leave the house for work* (as shown in Figure 6.1). The answers to such questions are provided by 100 randomly selected individuals and clustered into general categories by a professional polling company. Contestants attempt to provide an answer which matches these categories and get points according to the proportion of surveyed responses within a matched category. For example, when we polled 100 people with the same question (Figure 6.1), they provided 43 answers involving showering/cleaning, 30 answers mentioning breakfast, and the remainder fell into smaller groups such as locking a door/grabbing keys, saying goodbye, and praying. In a FAMILY-FEUD game, if two participants on a team answered "grab a shower" and "eggs and coffee", they would receive 73 points for providing answers which matched these two large categories. We suggest that this is an appealing paradigm for such question answering tasks where a wide range of acceptable answers exist, as it encourages both highly popular answers as well as wide coverage over the range of good answers.

We frame this task as a generative evaluation task in which a model outputs a ranked list of answers to a given question. Each answer string is then matched to one or more clusters of reference answers for that question. Matching an answer cluster gives the model a score equal to the cluster size. Our evaluation metrics (§ 4.3) reward models which provide the most common answers, while also measuring the model's ability to provide a diverse set of answers in order to match all the answer clusters. While such an approach can penalize a correct model prediction when it does not match an existing reference answer, we counter this issue by (a) gathering and clustering a large number of reference answers, and (b) utilizing methods of matching non-exact matches, such as WordNet [96] and contextual

[1] Dataset: `https://github.com/iesl/protoqa-data`.
Interactive demo: `http://protoqa.com`.

language models such as RoBERTa [88]. Generative evaluation approaches are also used in other NLP tasks such as summarization [121] and translation [25].

We evaluate on a set of competitive baseline models — from QA models powered by large masked LMs such as BERT, to the direct prediction of answers in a language-modeling paradigm using a large GPT-2 LM [122], as well as GPT-2 fine-tuned upon the training data. While most models perform quite poorly at this challenging task, when GPT-2 was fine-tuned using the FAMILY-FEUD training set its performance did improved drastically, although remaining significantly below the score of human-level performance.

The contributions of this Chapter to the research community are as follows.

1. We introduce a large-scale QA dataset of 9.7k questions regarding common sense knowledge of prototypical situations with 7-8 labeled answer categories per question, and a corresponding evaluation set of 15,400 crowd-sourced human judgments over 154 unseen questions.

2. We present methods for robust evaluation of this task to encourage models to provide diverse answers covering all plausible answer categories.

3. We evaluate against a range of plausible baselines, showing that while large contextualized language models fine-tuned on this data can perform well at the task, a meaningful gap still exists between model and human performance, suggesting room for improvement.

## 4.2 Dataset Creation and Analysis

### 4.2.1 Training Corpus Collection

A number of fan websites exist which have transcribed FAMILY-FEUD questions and answer clusters. We use publicly available text from two such websites to provide a training dataset on this task.[2] Well over 10,000 questions (with answer clusters) were collected, and a set of 9,762 questions remained after filtering, quality control, and de-duplication.

That filtering included the omission of questions that were taxonomic in character rather than probing common sense knowledge, such as *name a vegetable*, as well as the omission of questions encoding stereotypes. A small set of training instances which ascribe specific stereotypes or expectations to a particular group or gender – such as *"name something little boys love to build models of"* – were separated from the main training data set to avoid encouraging trained models to learn such biases [3]. We note, however, that common sense questions may carry a wide range of more nuanced culturally-specific information and biases. Studying the bias in such datasets, and natural stereotypical biases which pre-trained language models have been shown to have [137], would be a valuable topic of future work.

### 4.2.2 Test Corpus Collection

In order to establish a rich, open-ended answer generation task, we created new questions similar to those seen in the training set, collected 100 answers for each question[4] from the crowd-sourcing platform FigureEight[5] and manually clustered them. Because we gathered large sets of possible answers and clustered them, the evaluation set represents rough distributions over the expected raw string answers for each question, thereby increasing the ability to recognize any way of expressing one of those answers.

We attempted to make sure that this set of new questions maintained the same domain and the same common sense reasoning seen in the training data. In order to maintain similarity to existing questions, these questions were created by removing a set of questions from the scraped data and perturbing important aspects, making sure that the perturbations were

[2]Scraping details and site names are provided in the datasheet (following [45]) provided with the data

[3]Criteria for exclusion are listed in the appendix

[4]Each worker, on average, provides 41 judgments, and 5 cents per judgment.

[5]Now `https://appen.com/`.

sufficient to meaningfully change the answer set (thus being similar to the "counterfactually augmented" permutations of [67]). For example, given an existing question of "*Name something a person might forget to put on if they leave the house in a hurry.*", changes of polarity and events would derive a related question "*Name something that people usually do before they leave the house for work*". Deriving such unseen test questions was especially important to avoid the risk of having a publicly-available question be included in the training data for contextual language models; by making new data, we can be confident that any high-performing model has not yet seen the data. In order to control the quality of perturbed questions, the quality of each each perturbed question was scored by four experts (criteria listed in the appendix), and only the top-scoring questions were used to build the evaluation set.

We then created tasks on FigureEight for each selected question to be answered by 100 workers. To match the training data (which is inherently grounded in US culture), we limited workers to US locations. Low-quality workers were automatically detected through test questions during annotation, and the clustering pass provided a second manual quality control check. This left us with 154 questions which we split into a test set and development set of 102 and 52 respectively.

### 4.2.3 Answer Clustering

Each list of 100 raw string answers was manually clustered by two different experts familiar with the task. Clusters were assigned separately and then compared, and a final clustering was agreed on.[6] During this clustering phase answers could be marked as invalid as well — most commonly, either due to low-quality annotations or a clear misunderstanding of a question. In order to keep these clusters roughly similar to the granularity of answers

[6]The four total expert annotators annotated a random set of 10 questions together to calibrate their clustering granularity. Furthermore, two annotator's results are aggregated by a third person to reduce bias.

| Question | Example Answers | Types |
|---|---|---|
| Name a profession where you might be fired if you lost your voice | radio host , teacher | 3, 4, 6 |
| Name something a boy scout might learn. | knot tying, camping | 2, 5, 6 |
| Name a bad sport for someone who is afraid of the water. | diving, water polo | 1, 3 ,6 |
| Name something a monk probably would not own. | weapons, smartphone | 2, 4, 6 |
| Name something parents tell their kids not to do | steal, smoke | 1, 2, 4, 6 |
| Name a reason why someone would wear gloves | cold weather, cleaning | 2, 3 |

Table 4.1: Examples of questions from collected (top 3) and crowd-sourced (bottom 3) development sets, characterized with reasoning types described in § 4.2.4

used in the training data and to avoid low-quality evaluation we eliminated questions for which the 8 most popular clusters did not contain at least 85 of the 100 responses.

Since each set of answers was clustered twice and adjudicated, we measure the agreement with a cluster agreement metric BLANC [125, 92], an extension of the Rand index used to score coreference clustering. Using this, the similarity between the clusters produced by any two annotators averaged out to a BLANC score of 83.17, suggesting a coherent amount of agreement regarding the clustering of answers.

### 4.2.4 Analysis of the Dataset

The data presented here involves a range of different types of common sense knowledge. To explore the distribution of different kinds of reasoning, and to test whether that distribution of reasoning varied between the publicly available data and the crowdsourced development and test set, we propose a small inventory of six types of common sense reasoning.

We are not aware of an agreed-upon typology of all commonsense reasoning types. Categorizations of different types of commonsense reasoning exist [90, 18], but since each provided categorizations needed for specific tasks (RTE and the ARC dataset, respectively), neither fully covered the range of commonsense types seen in the current work. After consulting both those prior works and a separate part of the training data, we characterize the data into the following six types.

These types consist of (1) MENTAL OR SOCIAL REASONING, (2) KNOWLEDGE OF PROTOTYPICAL SITUATIONS which one is familiar with, (3) REASONING ABOUT NOVEL, COMPLEX EVENTS, (4) NEGATION AND EXCEPTIONS and understanding their consequences, (5) SPECIFIC ENTITY KNOWLEDGE of named people, locations, or organizations, and finally (6) KNOWLEDGE OF HABITUAL ACTIVITIES of specific occupations or types of entities.

Following other characterizations of reasoning type [90, 18], we annotated a random sample of questions (25 from dev and 25 from train) using six basic common sense reasoning categories in order to provide a simple approximation of the distribution over reasoning types contained in the data. Table 4.1 illustrates examples of questions with these types, and one can see the frequency of each type used in Table 4.2. The counts shown for each dataset illustrate that while the creation methodology varied between the two resources, the kind of common sense reasoning tasks evaluated by these models is quite similar between the two corpus types. The greatest difference to note is that the crowd-sourced data makes less use of questions regarding specific entities, which were avoided as they tended to involve fact-based world-knowledge rather than common sense reasoning.

| Reasoning type | Scraped Dev | Crowd-sourced |
|---|---|---|
| Mental/Social | 16% | 12% |
| Prototypical Events | 68% | 80% |
| Event Reasoning | 28% | 40% |
| Negation | 12% | 20% |
| Specific Entities | 20% | 4% |
| Habitual Activity | 40% | 24% |

Table 4.2: Percentage of questions utilizing each reasoning type

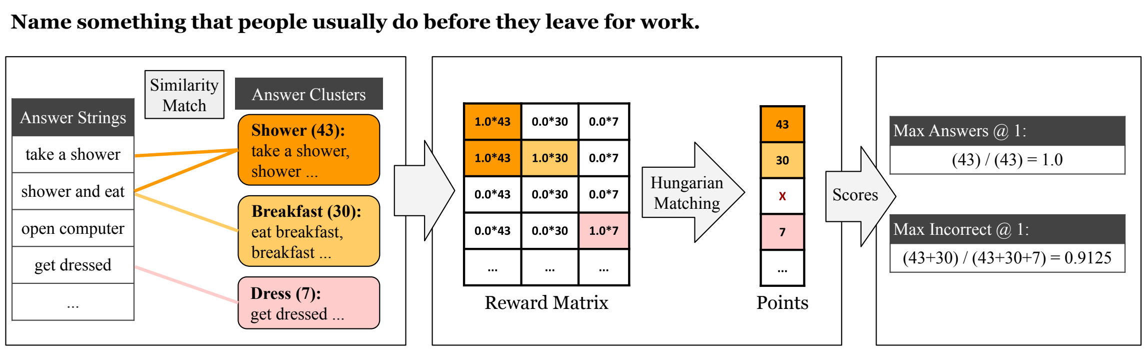


Figure 4.2: Example steps for evaluating a ranked list of answers

## 4.3 Evaluation

We present a number of methods for evaluating system-generated answers against these sets of clustered answers. In each, models are evaluated by providing a ranked list of answers in response to a question. These answers are then compared to the set of reference answers for that question and scored based upon how similar they are to the known answers. While one might instead convert question-answer pairs into a multiple-choice paradigm by generating negatives, it is difficult to generate good negative examples, and the quality of a dataset can be compromised if such examples are either too easy or easily identified using biases in the negative example generation process [99, 166, 145, 135, 52, 119].

We outline here our proposed method for scoring these ranked lists of predicted answers. The dataset ground truth is a ranked list of clusters of answers, including weights(cluster sizes) associated with each cluster. A first component in such an evaluation is to match each answer to an existing cluster of answers, if any cluster is acceptable. We try both simple methods such as exact match as well as more flexible ways of matching to clusters, such as using synonyms from WordNet [96] or a vector-based similarity method using RoBERTa [88]. The second component in this generative evaluation is to provide an overall score for the entire ranked list of answers by mapping individual answers to answer clusters or marking them wrong. Scoring answers against clusters alone does not take into account the ranking. To that end, we propose two different metrics, one similar to hits@k in traditional information retrieval task and one which limits the number of incorrect answers, which is closer to how humans are typically evaluated on this task.

In each case the score reported is calculated as a percentage of the oracle score. Both proposed methods of scoring reward models which provide a diverse set of guesses to a given query and penalize models which provide many variations of the same answer. (See figure 4.2 for a general idea of the steps involved.)

### 4.3.1 Matching Answers to Clusters

- **Exact Match.** In our simplest way of matching answers to clusters, we compare each answer with the answer strings from crowd-source workers for a given cluster, returning a score of 1 if it matched any string in the cluster and returning 0 if not. By construction, therefore, a given answer string will match at most a single cluster with this method.

- **WordNet Similarity.** Reasonable answer strings may be incorrectly marked as wrong with an exact string match, even when they are clear synonyms of a reference answer. METEOR [5, 74] addressed similar issues in machine translation via stemming and synonym matching. We take a similar approach, tokenizing a proposed answer string and comparing it to the tokenization of the answers in each answer cluster. Since some words in WordNet are multi-word phrases (eg. “chewing gum") we furthermore perform this matching on all possible partitions of the tokenization. For each answer in an answer cluster we return the maximum (over all possible partitions) of the average number of matched tokens. The assignment of answers to clusters proceeds as in the exact match case. Further details are included in the appendix.

- **RoBERTa Similarity.** Recent works in MT evaluation [171, 136] used pre-trained language models to compare predictions to reference answers. We implement a simple version of such vector-based comparisons, but this current task differs in that we assign each predicted answer to a particular cluster of correct answers, or decide whether to reject the answer. As clusters vary in size and specificity we cannot determine a universal threshold for how similar a mention must be to a cluster. Instead, we train a small classifier in L2 distance space for each answer cluster in order to decide membership in that answer cluster. We do this by obtaining a vector representation of each answer from RoBERTa [88], concatenating each answer with the question, and taking the mean of answer token representations. For each cluster we train a small one-vs-all classifier over the 100 answers to that question, predicting membership in that cluster (using gaussian process regression [160] with an RBF kernel). At test time, a given answer is assigned to the highest-scoring cluster, as long as its likelihood of membership exceeds a minimum probability threshold, set at 0.1. Such an approach allows us to match answers to clusters while omitting answers which do not match existing clusters.

### 4.3.2 Evaluating Diverse Lists of Answers

As mentioned previously, we want to design evaluation metrics that favor models which take into account the ranking while still covering all plausible answer categories. We first compute an alignment score between each answer in the ranked list and each of our answer clusters. After computing the alignment scores between all pairs of answers and clusters we create a reward matrix where, for each answer and cluster, we assign a reward equal to the cluster size if the alignment score was a 1 and 0 otherwise. We employ the Hungarian matching algorithm [71, 100] to compute the exact optimal matching of answers to clusters based on this reward matrix, so that an answer is assigned to only one cluster. It is worth noting that a model which produces a ranked list of answers only in one cluster will do worse than a model which maximally covers all plausible clusters. Lastly, to make the comparison between lists of different lengths uniform, we propose the following metrics.

1. MAX ANSWERS@$k$ limits the total number of answers allowed to up to $k$ answers.[7]

2. MAX INCORRECT@$k$ allows unlimited answers, but stops after $k$ unmatched answers are provided.

In both conditions, we report the score as the percentage of the max score one could receive given that number of guesses, and only give credit for a given cluster once.

## 4.4 Baselines

We explore three baseline models for this task: a QA-based model which retrieves related posts in a discussion forum for each question, a language-modeling baseline which examines how well modern pre-trained language models do at directly producing the answers, and a fine-tuned version of the language-model baseline.

### 4.4.1 Question-Answering Baseline

As this dataset is in the form of questions and answers it may be treated as a QA dataset, although the content is far from the fact-based data usually modeled in QA tasks. As the training set only shows answers out of context, one must use distant supervision in order to train a QA model on the data, a well-explored situation in modern QA work [65]. Unlike factoid-based QA, one may expect a limit in the performance of such QA models for common sense reasoning, as common sense data is well-known to have a *reporting bias* [48] wherein many facts that are part of the common ground of known knowledge are less likely to be stated.

To train a model in this approach, we collected up to 20 documents for each of the 9.7k questions in the FAMILY-FEUD training dataset by using a web search for each question constrained to Reddit. This resulted in a set of 85,781 Reddit posts total. Searches were

[7]Note that since our scores are always calculated as a percentage of the max score one could receive, MAX ANSWERS is slightly different than hits@k in this setting.

| Metrics % | | | QA Model | GPT-2 | GPT-2 Fine Tune | Human |
|---|---|---|---|---|---|---|
| **Exact Match** | **Max Answers** | 1 | 2.1 | 5.6 | 29.4 | **78.4** |
| | | 3 | 4.4 | 15.9 | 37.6 | **74.4** |
| | | 5 | 6.8 | 18.3 | 40.1 | **72.5** |
| | | 10 | 11.0 | 23.2 | 45.9 | **73.3** |
| | **Max Incorrect** | 1 | 0.8 | 3.3 | 18.7 | **55.8** |
| | | 3 | 3.6 | 15.1 | 35.0 | **69.4** |
| | | 5 | 6.4 | 19.3 | 41.1 | **72.4** |
| **WordNet Similarity** | **Max Answers** | 1 | 3.4 | 6.2 | 36.4 | **78.4** |
| | | 3 | 6.4 | 18.5 | 44.4 | **76.8** |
| | | 5 | 9.1 | 23.0 | 46.6 | **76.0** |
| | | 10 | 15.7 | 30.5 | 53.5 | **77.0** |
| | **Max Incorrect** | 1 | 1.4 | 4.3 | 26.1 | **59.0** |
| | | 3 | 5.3 | 17.9 | 41.7 | **74.0** |
| | | 5 | 8.4 | 24.2 | 48.2 | **77.9** |
| **RoBERTa Similarity** | **Max Answers** | 1 | 49.1 | 38.7 | 55.0 | **81.2** |
| | | 3 | 53.3 | 48.8 | 60.7 | **78.9** |
| | | 5 | 57.1 | 52.0 | 63.0 | **80.1** |
| | | 10 | 65.0 | 60.5 | 71.2 | **83.5** |
| | **Max Incorrect** | 1 | 49.1 | 38.7 | 55.0 | **81.2** |
| | | 3 | 53.3 | 48.8 | 60.7 | **78.9** |
| | | 5 | 57.1 | 52.0 | 63.0 | **80.1** |

Table 4.3: Results on the annotated test set. Scores are normalized by the maximum score obtainable with that number of guesses, and therefore may go down as k increases

constrained to Reddit in order to focus upon advice and personal narratives which might discuss common sense questions. For any post matching that query, any strings matching an answer to that question in the training data would be treated as a positive example for the QA model. The QA model used was the "Bert for QA" implementation within the Hugging Face Transformers package [161]; training details, and examples of the kind of noisy training data generated through this process, are provided in the appendix.

At test time documents were obtained by searching for the question in a google search restricted to Reddit, and the QA model was run on that set, taking the 20 best answers in context as possible answer strings. Those best answer strings from each passage were combined together, summing scores for identical strings, to provide a ranked list.

### 4.4.2 Language Model Baseline

We also report a language model generation baseline, due to the improved representation power of modern language models and recent evidence of their power in modeling common sense reasoning tasks [159, 146]. The baseline is performed using the AI2 GPT-2 large model [123] (specifically, the Hugging Face PyTorch implementation [161]). We perform both a zero-shot evaluation and an evaluation after fine-tuning with using our training data.

Because the original FAMILY-FEUD prompts are not structured as completion tasks, we transform the original question by hand-designed transformation rules in order for it to be compatible with the GPT-2 training data. E.g "Name something people do when they wake up." $\rightarrow$ "One thing people do when they wake up is ...". The hand-designed rules are including in the appendix. The transformed questions are used as input to the language model, and GPT-2 finishes the sentence. The reported fine-tuning result is trained on the scraped training corpus and the best model selected based on performance on our annotated development set. Training details and parameter setting for the model is provided in the appendix.

In order to generate diverse answers for a given sentence we use Nucleus Sampling [58] as our decoding method. We get 300 sampled answers for each question and group them by counts, returning a ranked list of 20 answers from most to least common.

### 4.4.3 Human Performance

To measure human performance against such models, we collected 30 additional human responses per question with the same setup in collecting test data and aggregated them by counts, just as the sampled answers from GPT-2 models were ranked. The last column in table 4.3 reports this human performance. We can see that the best-performing automatic system is still meaningfully behind human performance in all metrics.

## 4.5 Discussion

Table 4.3 shows the results of the baseline models using different measures of similarity, and different measures for the MAX ANSWERS and MAX INCORRECT metrics. One can see that GPT-2 without fine-tuning outperforms the baseline QA implementation, and fine-tuned GPT-2 outperforms both, but a large gap still remains between human performance and any of the baselines, even on the generous RoBERTa-based similarity metric. The human baseline scores are relatively stable regardless of which similarity metric is used, whereas the model scores change drastically (most significantly for the QA model) as more generous similarity metrics are used. We suggest that WordNet Similarity be used as the primary similarity metric as it strikes a reasonable balance between precision and recall, as discussed in § 4.5.2.

### 4.5.1 Knowledge Base Comparison

To show the dataset indeed containing meaningful commonsense knowledge, we did an additional analysis between our dataset and ConceptNet. ConceptNet [141] is a knowledge base containing triples related to common sense which has been shown to be helpful for various downstream tasks [174, 155] and conversational text generation [162, 169]. We

| | **Precision** | **Recall** | **F1** |
|---|---|---|---|
| **Exact Match** | **1.0** | 0.466 | 0.636 |
| **WordNet Similarity** | 0.996 | 0.581 | **0.734** |
| **RoBERTa Similarity** | 0.762 | **0.661** | 0.708 |

Table 4.4: Measurement of different score function against human cluster assignment.

evaluate its potential relevance to this task by evaluating how often a (question, answer cluster) pair has a possible matching triple within ConceptNet. We extract a list of keywords from the question and a ground-truth answer string (by removing stop words) and similarly extract keywords from the head and tail of each ConceptNet relation. We then evaluate whether a given question-answer pair has potential "coverage" in ConceptNet by checking whether a keyword in the question is related to a keyword in the answer. For example, given the question "*Besides music, name something you might hear on a morning radio show*" and the answer "*weather report*", we would find the triples *(listen to radio, Cause, you hear local weather report)* and *(listen to radio, HasSubevent, hear weather report)*. By this measure, we find that 24.3% of the answer clusters in our test set have some match within ConceptNet. This suggests that a common sense KB might provide a useful resource for this task, however ConceptNet has a large number of relations with no direct ability to provide a ranking and thus we exclude such a model from our baseline comparisons. A similar analysis shows that the human baseline match 46.5% of the clusters, whereas a list of 20 top answers from the fine-tuned GPT-2 model match 30.3%.

### 4.5.2 Score Function Comparison

In order to compare the various similarity functions outlined in § 4.3, we manually annotated answers – from both the human baseline and fine-tuned GPT-2 outputs – to the

| **Prompt** | ***Name something around the house that's often replaced.*** | | | |
|---|---|---|---|---|
| Human | 🟧 light bulbs | 🟨 toilet paper | 🟦 furniture | 🟦 food |
| GPT-2 | TV | refrigerator | fridge | 🟥 trash |
| GPT-2 Fine Tune | dishes | 🟨 toilet | kitchen | 🟦 furniture |
| QA | tune | time | name | song |
| **Prompt** | ***Name something a monk probably would not own*** | | | |
| Human | 🟧 gun | 🟦 wife | 🟧 knife | 🟨 pornography |
| GPT-2 | 🟧 gun | 🟥 car | 🟧 sword | 🟥 motorcycle |
| GPT-2 Fine Tune | 🟧 weapon | 🟧 sword | 🟥 car | 🟥 cell phone |
| QA | arch | everything | togashi | power |
| 🟧 largest cluster | 🟨 cluster 2 | 🟥 cluster 3 | 🟦 smaller clusters | |

Table 4.5: Top responses from human and model predictions for each prompt, color-coded with the answer cluster they might be aligned to

correct answer clusters. Four annotators separately mapped each answer string to an existing cluster.

Table 4.4 measures how well different similarity functions performed in comparison to the manual human cluster assignment. Precision in this context measures how often an answer assigned by the automatic similarity measure is correctly assigned; recall measures how often an answer which a should be assigned to a cluster is correctly assigned. Unsurprisingly, exact match has perfect precision in this context, but has relatively low recall. WordNet similarity increases recall while adding very little false positives. As was hoped, RoBERTa similarity does dramatically increase how often an answer is mapped to the correct cluster, but does so at the expense of a large loss in precision; we therefore suggest that the WordNet similarity is the safest evaluation option.

### 4.5.3 Error Analysis

To provide some notion for the tendencies of different models on this task we provide actual model outputs in Table 4.5. One can see that, before fine-tuning, GPT-2 results are often acceptable and plausible situations (e.g. refrigerators might be replaced), but can fail to answer the specific criteria requested by the prompt. In contrast, the QA-based model is much noisier – occasionally providing very good answers, but often (as in the examples provided) failing to find answers that are even plausible. Fine-tuned GPT-2, in contrast to both, clearly learns to actually focus upon the expected format and details of such prototypical activities, however it fails in situations where a high-scoring answer would be very rarely discussed, such as knowing that light bulbs are commonly changed around the house.

## 4.6 Related Work

A wide variety of common sense reasoning datasets address related topics. Many datasets cover physical and spatial reasoning [12], social common sense [131], and common sense understanding of plausible sequences of events [166, 167, 61, 10, 130] or understanding of the entailments of a sentence [170, 22, 126, 76]. There is also a long history of work in modeling scripts and frames [134, 27, 41, 40, 157], which is related to the current focus on prototypical situations.

Recent works have also sought to characterize the ability of pre-trained language models to understand common sense reasoning, showing such models perform well at common sense reasoning tasks even without fine-tuning, allowing one to explore the common sense reasoning inherent in those models [146, 159]. Of particular relevance to the current work, [159] explored the ability of pre-trained models to predict *stereotypic tacit assumptions*, generalizing about entire classes of entities with statements such as "everyone knows that a bear has ________".

Interestingly, ProtoQA is not the first time FAMILY-FEUD has been referenced in the commonsense literature. Common Consensus [82] was a web-based game created with the intention of being a self-sustaining platform to collect and validate commonsense knowledge based on human goals. Prior work had established the idea of using online games to simultaneously entertain and collect commonsense knowledge [1], however the authors of

Common Consensus found that the format of FAMILY-FEUD questions was more amenable to high-quality commonsense knowledge acquisition. Common Consensus serves as an excellent proof of concept for future gamification of the style of data presented in this dataset.

ProtoQA differs from other datasets in three different ways:

1. ProtoQA focuses on proto-typical situations. Humans can agree about the details and characteristics of a prototypical event or situation due to commonalities in their shared lived experiences, cultural norms and expectations. This rough agreement extends beyond an agreement on a single top response and that's why our task and evaluation values diversity of answers.

2. The evaluation ProtoQA is a generative evaluation task where a model has to output a ranked list of answers, ideally covering all prototypical answers for a question.

3. ProtoQA has a large number of annotations for each example which makes the generation evaluation possible.

# CHAPTER 5

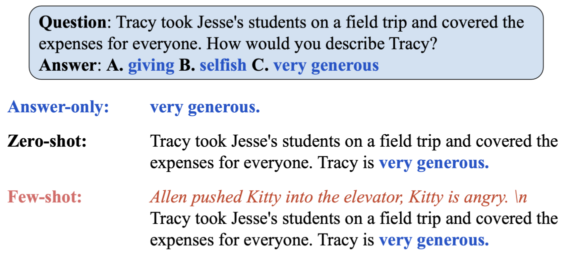
# DO LANGUAGE MODELS LEARN COMMONSENSE KNOWLEDGE?

## 5.1 Introduction

In previous Chapters, we discussed modeling commonsense using probabilistic box embeddings and evaluated this knowledge using a generative ranking-based evaluation. Concurrently, the field of NLP is advanced dramatically by using language models with billions of parameters that are trained on large corpora. These large LMs [23, 111] have achieved remarkable performance at various common-sense benchmarks [129, 167, 12, 132], even when they are evaluated in a zero-shot or few-shot fashion, *without* explicit commonsense supervision. We revisit this apparent success, and conduct a rigorous study to better understand the extent to which such pre-trained LMs are able to capture commonsense knowledge.

In this work, we focus on zero- and few-shot evaluations of pre-trained LMs without commonsense-specific fine-tuning for two reasons: First, we aim to examine if a pre-trained LM is able to acquire *general* commonsense knowledge. As pre-trained LMs constitute a *foundational* building block of NLP today, any deficiencies in their commonsense understanding can thus adversely manifest in downstream applications [16]. Fine-tuning the LM would make it hard to disentangle how much of the commonsense knowledge is acquired by the underlying LM, as opposed to the *task-specific* supervision from a benchmark [163]. Second, human-annotated commonsense datasets are expensive to collect due to the vast, diverse, and growing nature of commonsense knowledge [37].

Concretely, our work differs from prior work on commonsense evaluation of LMs [23, 111] by way of a more rigorous evaluation, in which we: (i) carefully control for the


**Question**: Tracy took Jesse's students on a field trip and covered the expenses for everyone. How would you describe Tracy?
**Answer**: **A.** **giving** **B.** **selfish** **C.** **very generous**

| | |
|---|---|
| **Answer-only:** | **very generous.** |
| **Zero-shot:** | Tracy took Jesse's students on a field trip and covered the expenses for everyone. Tracy is **very generous.** |
| **Few-shot:** | *Allen pushed Kitty into the elevator, Kitty is angry. \n* Tracy took Jesse's students on a field trip and covered the expenses for everyone. Tracy is **very generous.** |

Figure 5.1: The experiment settings with their corresponding input to the LM. The example is taken from Social IQa [132] where we convert questions to natural text using the rules of [139]; this conversion yields to better performance (§5.5).

LM's ability to exploit potential surface cues and annotation artefacts to predict the answer, without reasoning over the context. We further (ii) account for the variations in factors influencing the LM's performance, which arise from certain evaluation design choices — independently of commonsense knowledge in the models. We systematically conduct this study on four commonsense benchmarks, six model sizes (up to a very large LM with 280B parameters), and multiple evaluation settings (e.g., different score functions and prompt format).

We begin with our first question: When evaluating a large LM in a zero-shot setting, *how does its zero-shot performance compare to a strong baseline (§5.3)*? Controlling for the LM's ability to guess the correct answer, *without* even looking at the question **Answer-only baseline**, top of Fig. 5.1][118, 147], we find that, despite the LM's strong zero-shot performance, the Answer-only baseline can nevertheless perform surprisingly well on some benchmarks. Despite the clear importance of comparing with answer-only baselines, these comparisons are absent from recent work on large LMs [175, 23, 124]. Furthermore, increasing model size alone is unlikely to bridge the gap with human performance in the near future: Our analysis of scaling behavior suggests that much larger dense LMs (with 100T to $10^{18}$ parameters — which are infeasibly large at present) are needed to achieve human performance for 3 out of 4 benchmarks.

*Does familiarizing the LM with the task format using a few-shot evaluation setting substantially improve performance (§5.4)?* We find that the few-shot evaluation (using up to 64 examples) does not substantially improve the LMs' performance for most tasks except Social IQa. Moreover, using the few-shot/in-context demonstration setting fails to bridge the gap between the LM and current SOTA.

Finally, we ask: *to what extent does the model's zero-shot performance vary depending on certain evaluation design choices, such as the format of the prompt or the score function (§5.5)*? We find that these design choices — though they have little to do with common sense — can result in large fluctuations in performance (up to 19%). This finding challenges the notion that large LMs are largely able to work well out-of-the-box with minimal task-specific tuning. Based on these findings, we emphasize the need to carefully select such design choices, explicitly state them to enable fair comparison with prior work, and quantify the robustness of the observed results across different design choices.

All in all, our findings suggest that acquiring *human-level* commonsense knowledge, without relying on surface cues or task-specific supervision, remains beyond the reach of current large LMs. Given the marginal improvements from increasing model size, we conjecture that other techniques, such as explicit commonsense supervision, multi-modal grounding, or physical embodiment [11], are promising ways forward.

## 5.2 Experimental Setting

| | **Choices** | **Main Knowledge Types** | **Questions** |
|---|---|---|---|
| **HellaSwag** [167] | 4 | Temporal, Physical | 10042 |
| **WinoGrande** [129] | 2 | Social, Physical | 1267 |
| **Social IQa** [132] | 3 | Social | 1954 |
| **PIQA** [12] | 2 | Physical | 1838 |

Table 5.1: Benchmark Statistics. For each benchmark, "Choices" and "Questions" show the number of candidate answers for each question and the number of questions in the validation split, respectively.

We begin by outlining our experimental setup, and describe the benchmarks, model, baselines, and other relevant experimental settings.

### 5.2.1 Commonsense Benchmarks

Commonsense knowledge spans many categories, such as physical common sense (e.g., a car is heavier than an apple), social common sense (e.g., a person will feel happy after receiving gifts), and temporal common sense (e.g., cooking an egg takes less time than baking a cake). Given this diverse nature of commonsense knowledge, various benchmarks have been proposed to test these different types of knowledge [167, 129, 132, 12, 85, 17].

Commonsense benchmarks broadly consist of two tasks: (a) multiple-choice evaluation [166, 167, 132, 12], where a model needs to choose the correct answer from a list of plausible answers; (b) generative evaluation [17, 85, 84], which requires a model to generate an answer given a question and some additional context. Here we focus on multiple-choice benchmarks, since they provide a more reliable automatic metric (i.e., accuracy), whereas automated metrics used to evaluate language generation e.g.BLUE [108] do not correlate perfectly with human judgment [86, 107].[1] We use a diverse set of four representative multiple-choice commonsense benchmarks to better understand the extent to which pre-trained LMs are able to acquire different types of commonsense knowledge. We use the validation split of each benchmark, as their test splits are not public.

**HellaSwag** [167] is designed to evaluate a model's ability to understand physical, grounded, and temporal common sense. Given a four-sentence story, the model must choose the correct ending from four candidates. The stories are either video captions from AcitivityNet [56], or WikiHow passages [70]. When evaluating LMs on a similar dataset [166], incorrect answers can be easy to distinguish from correct ones; hence in constructing HellaSwag, [167] removed easy negatives through adversarial filtering.

[1]Human judgment of LM output is not only costly to obtain, but also imperfect [30], compounding the difficulty of commonsense evaluation in a generation setup.

| Dataset | Prompt: $x$ | Answer: $y$ |
|---|---|---|
| **HellaSwag** | A woman is outside with a bucket and a dog. The dog is running around trying to avoid a bath. She | gets the dog wet, then it runs away again. |
| **WinoGrande** | The GPS and map helped me navigate home. I got lost when | the **GPS** got turned off. |
| **Social IQa** | Jordan was in charge of taking the food on the camping trip and left all the food at home. Jordan felt | horrible that he let his friends down on the camping trip. |
| **PIQA** | Make Halloween lanterns. | Draw ghost faces on empty milk bottles, put a candle in each one. |

Table 5.2: Examples of the prompt $x$ and the correct answer $y$ in different benchmarks.

**WinoGrande** [129] is a co-reference resolution benchmark that mainly examines physical and social common sense. Each example consists of a sentence (e.g., "The trophy did not fit the suitcase because it is too big.") and two candidate *entities* (e.g., "trophy" or "suitcase"). The task is to choose the correct entity for the pronoun, e.g., "it" refers to "trophy" in the example.

**Social IQa** [132] focuses on evaluating social commonsense, in particular theory of mind — the capacity to reason about others' mental states [42]. Given context sentences and a corresponding question, the task is to choose the correct response from three candidates. Annotators use the ATOMIC knowledge base [130] to create context sentence and questions; the answers are provided by additional annotators.

**PIQA** [12], short for physical interaction question answering, mainly covers the physical aspect of common sense. Each data point consists of a task and two alternative solutions to finish the task; one of which is correct. The tasks are curated from a website[2] with instructions for everyday tasks (e.g., separating egg yolks from eggs); the solutions are provided by human annotators.

### 5.2.2 Pre-trained Language Model

We use the pre-trained language model of [124], Gopher, which is an autoregressive Transformer [149] language model with 280 billion parameters. We choose Gopher because of its excellent zero-shot and few-shot performance at various benchmarks, in addition to its

[2]`https://www.instructables.com/`

large model size, which has been shown to improve language modeling and downstream performance [66]. Notably, Gopher is more than 50% larger than GPT3 and as of March 2022, is one of the largest dense LMs developed to date.

#### 5.2.2.1 Gopher hyper-parameters.

The pre-trained Gopher language model has 80 layers, 128 attention heads, 128-dimensional key/value vectors, and a feedforward layer dimension of 16,384. To better understand the effect of different model sizes (§5.3.3), we experiment with five other model sizes: 44M, 117M, 417M, 1.4B, and 7.1B. Similar to Gopher, each of these models was pre-trained by [124]; a full list of model hyper-parameters is summarized in Table 1 of [124]. Each model is trained by subsampling from the MassiveText dataset, which consists of more than 2 trillion tokens from various domains including web pages, news, books, and codes [124]. We use TPUv3 to conduct all evaluations, with an estimated total compute budget of $2 \times 10^{20}$ FLOPs.

#### 5.2.2.2 Score function.

On the multiple-choice benchmarks, we evaluate the pre-trained LM by calculating the score for each answer choice under the model, and select the highest-scoring answer $\hat{\mathbf{y}}$:

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y} \in Y(\mathbf{x})} s_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x});$$

here $\mathbf{x}$ denotes the question or prompt, $Y(\mathbf{x})$ the set of answer choices for a given question, and $s_{\boldsymbol{\theta}}(\cdot)$ the score of an answer choice $\mathbf{y}$ given $\mathbf{x}$, under the pre-trained LM with parameters $\boldsymbol{\theta}$. We provide some examples in Table 5.2.[3] For Social IQa, we convert questions to natural text using the rules of [139]; we find this natural text format to yield better results, as discussed in §5.5.

[3]For Social IQa, we concatenate the context sentence and question together to form the prompt $\mathbf{x}$.

Unless otherwise stated, we use *cross-entropy* (or token-level log prob) to score each answer:

$$s_\theta(\mathrm{y}|\mathrm{x}) = \frac{\sum_{i=0}^{||\mathrm{y}||} log(p_\theta(y_i|x, y_0...y_{i-1}))}{||\mathrm{y}||}. \tag{5.1}$$

This score function reduces the impact of length; without dividing by $||\mathrm{y}||$, longer answers might have lower probabilities [142]. GPT3 [23] also employs this score function for zero-shot evaluation.

### 5.2.3 Baselines

We compare the performance of Gopher with two baselines. The first, simple baseline is to randomly select an answer candidate, where the chance of selecting the correct one is $\frac{1}{\textit{number of choices}}$. We henceforth refer to this as the *Random Baseline*. We experiment with two other baselines: Either choosing the majority label from the training data, or choosing the longest answer. We omit these baselines as they perform similarly to the Random Baseline.

More importantly, we consider an *Answer-only Baseline*, where we select the highest-scoring answer choice under the LM, *without* conditioning on the question. More formally, this baseline considers $s_\theta(\mathrm{y})$, as opposed to $s_\theta(\mathrm{y}|\mathrm{x})$ in Eq. 5.1. This baseline reveals the extent to which the pre-trained LM conducts the appropriate reasoning over the context to select the answer, as opposed to relying on potential surface cues or annotation artefacts that make the correct answer *a priori* more probable than the rest. We illustrate this baseline at the top of Fig. 5.1. For WinoGrande, we calculate the cross-entropy of the text starting by the pronoun replacement, as shown in Table 5.2. Ideally, each answer choice should be equally likely if we do not consider the question, and the Answer-only performance should be close to the Random baseline. Similar hypothesis-only baselines are well-studied for natural language inference datasets [118]; [147] further explored such an Answer-only baseline, albeit only on the SWAG benchmark [166].

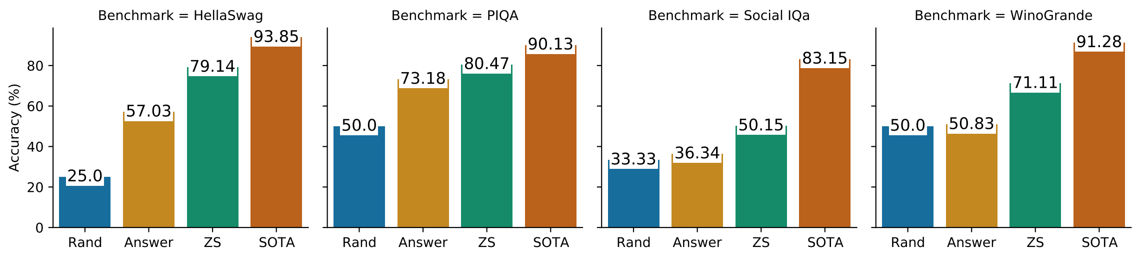


Figure 5.2: Random Baseline (Rand), Answer-only Baseline (Answer), zero-shot (ZS), and the current state-of-the-art (SOTA) for each benchmark, which is achieved by UNICORN [91].

## 5.3 Zero-shot Performance

In Fig. 5.2, we report the zero-shot performance of our pre-trained LM (with 280B parameters, §5.2.2) on the four commonsense benchmarks, alongside: (i) the Random and Answer-only baselines, and (ii) the current state-of-the-art (SOTA) result. The SOTA results are achieved by the UNICORN [91] model with 11B parameters, which is pre-trained on 6 existing commonsense datasets [167, 12, 132, 129, 9, 60].

### 5.3.1 Zero-shot performance.

At first glance, we observe strong zero-shot results, outperforming the Random Baseline in all benchmarks (compare "Rand" and "ZS" in Fig. 5.2). However, the gap between the stronger Answer-only baseline and the zero-shot result is smaller for all benchmarks (compare "Answer" and "ZS"): Whereas this gap is still sizable for HellaSwag and WinoGrande (>20), it is much smaller for Social IQa and PIQA. Finally, in all cases, there is still a large gap between the SOTA and zero-shot performance (>10); this gap is largest for WinoGrande and Social IQa, suggesting that social and physical commonsense is challenging for pre-trained LMs — even a large one with 280B parameters — without task-specific supervision.[4]

[4]We remark that the 530B-parameter LM of [111] achieves slightly better performance than Gopher on HellaSwag (80.2), PIQA (82), and WinoGrande (73), although there remains a large gap with the SOTA performance.

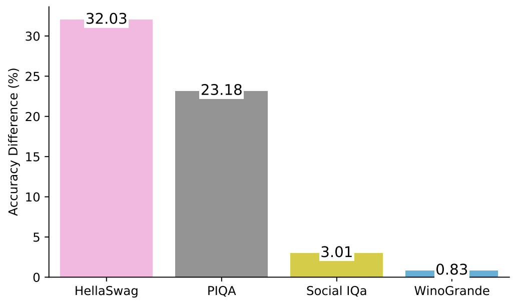


Figure 5.3: The performance gap between Answer-only and Random baselines for each benchmark.

### 5.3.2 Answer-only bias

As shown in Fig. 5.3, the performance gap between the Random and Answer-only baselines is notably large for HellaSwag and PIQA, where the Answer-only baseline outperforms the Random baseline by more than 32% and 23%, respectively. This large gap highlights an existing answer-only bias in these benchmarks: the correct answer can, in fact, be selected by the LM without conducting the appropriate commonsense reasoning over the provided context. On the other hand, the Answer-only baseline performs similarly to the random baseline on WinoGrande and Social IQa; hence, the zero-shot performance on these benchmarks is a more reliable estimate of the model's acquisition of commonsense knowledge. Given the existing (and sometimes inevitable) answer-only biases in some benchmarks, it is important to contextualize the zero-shot results by comparing with strong baselines, although such comparisons are missing from recent work e.g.[175, 23, 124].

### 5.3.3 Does Increasing Model Size Help?

Gopher (the largest LM we have access to) achieves a decent zero-shot performance for most commonsense benchmarks, but maintains a notable gap with fine-tuned SOTA results. Can we eventually reach human-level performance on these commonsense benchmarks by increasing model size alone?

Since we do not have access to larger language models than Gopher, we examine the extent to which zero-shot performance improves when using Gopher compared to a range of

| | | Answer | ZS | FS(1) | FS(10) | FS(64) |
|---|---|---|---|---|---|---|
| **HellaSwag** | **44M** | 25.8 | 28.0 | 28.0 | **28.1** | 27.9 |
| | **117M** | 29.2 | 33.5 | 33.3 | **34.0** | 33.5 |
| | **417M** | 35.6 | **44.1** | 43.4 | 43.3 | 43.3 |
| | **1.4B** | 43.2 | **56.7** | 56.4 | 56.2 | 56.5 |
| | **7.1B** | 50.4 | **69.5** | 67.6 | 67.9 | 67.9 |
| | **Gopher** | 57.0 | 79.1 | 77.8 | 79.2 | **79.3** |
| **WinoGrande** | **44M** | 48.5 | **51.3** | 51.1 | 50.8 | 50.6 |
| | **117M** | 50.8 | 52.0 | **51.9** | 50.9 | 50.8 |
| | **400M** | 49.9 | 52.2 | 51.8 | 50.8 | **52.5** |
| | **1.3B** | 49.7 | **58.1** | 56.4 | 56.0 | 57.3 |
| | **7B** | 52.4 | **64.6** | 62.1 | 63.1 | 62.0 |
| | **Gopher** | 50.8 | 71.1 | 69.2 | 71.4 | **74.6** |
| **Social IQa** | **44M** | 35.5 | **42.0** | 41.2 | 40.9 | 40.9 |
| | **117M** | 36.1 | **43.7** | 42.7 | 42.1 | 42.2 |
| | **400M** | 36.0 | **45.6** | 44.5 | 45.2 | 45.3 |
| | **1.3B** | 35.8 | 46.9 | 46.4 | 48.6 | **50.5** |
| | **7B** | 36.9 | 48.1 | 48.1 | 52.9 | **54.2** |
| | **Gopher** | 36.3 | 50.2 | 50.2 | 55.3 | **57.5** |
| **PIQA** | **44M** | 60.2 | **62.6** | 62.1 | 62.3 | 61.3 |
| | **117M** | 62.1 | **65.5** | 64.6 | 65.1 | 65.3 |
| | **400M** | 65.9 | **70.9** | 68.8 | 70.5 | 70.1 |
| | **1.3B** | 68.4 | 74.4 | 73.3 | 74.4 | **74.6** |
| | **7B** | 70.0 | 77.4 | 75.5 | 77.6 | **78.1** |
| | **Gopher** | 73.2 | 80.5 | 79.3 | 81.4 | **81.5** |

Table 5.3: Performance of all models across benchmarks under different experimental settings. Ans: Answer-only Baseline; ZS: zero-shot performance; FS($n$): few-shot performance where $n$ is the number of examples.

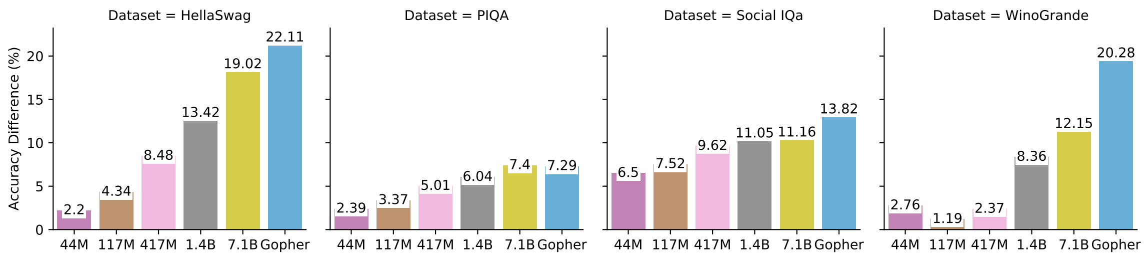


Figure 5.4: The difference between zero-shot performance and Answer-only baseline for different model sizes.

smaller models (i.e., scaling plots). Such scaling plot can help us predict the performance for larger models than Gopher. To that end, we use 6 pre-trained model sizes from 44M to 280B parameters (see §5.2.2).[5] We present the findings in Table 5.3. On all four benchmarks, the LM's zero-shot performance (Table 5.3, **ZS** column) consistently gets better as we use increasingly larger models. This finding is also consistent with that of [23], who showed that larger models have better performance at HellaSwag, WinoGrande, and PIQA. But, crucially, we argue that this does *not* necessarily mean that larger models are better at commonsense reasoning: For HellaSwag and PIQA, the Answer-only baseline also substantially improves with model size (Table 5.3, **Ans** column). Hence, for these benchmarks, larger models are *also* better at exploiting potential surface cues and annotation artefacts to guess the correct answer, without reasoning over the context. To properly assess commonsense reasoning, we should focus on the *performance difference* between the zero-shot and the Answer-only baseline.

We plot this performance difference with respect to different model sizes in Fig. 5.4. We observe that larger models have better performance across benchmarks — when increasing model size, the zero-shot performance gains are *more* than the performance gains of the Answer-only baseline. Nevertheless, the magnitude of this improvement varies depending on the benchmark: We see a substantial improvement on WinoGrande, but smaller improvements on HellaSwag, Social IQa and PIQA.

#### 5.3.3.1 Scaling behavior.

Based on these trends, what model size would be required to achieve human-level performance on these benchmarks? Through a linear regression analysis (see Appendix A.3 for more details), given the current rate of improvement in performance when gradually increasing the model size from 44M up to 280B, we need a model of at least 1.4T parameters

[5]Each model size is trained on the same dataset; hence any performance differences can be attributed to model size.

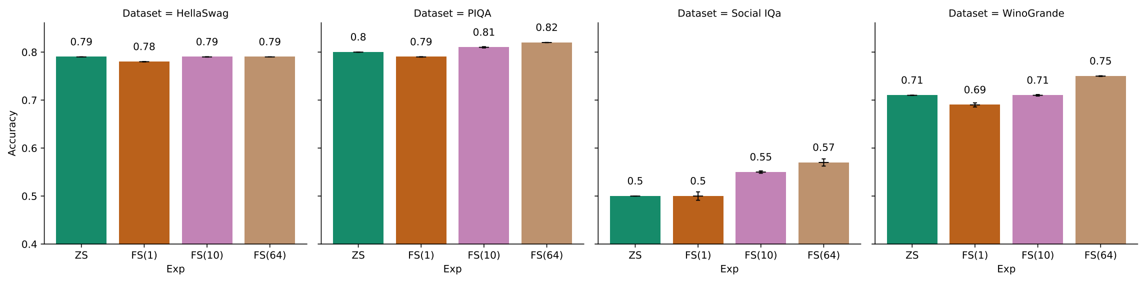


Figure 5.5: Accuracy on the benchmarks for zero-shot (ZS) and few-shot (FS) settings (with 1, 10, and 64 examples). We additionally report the error bars, although the error bars are not always visible due to the very small variance.

to achieve human performance on HellaSwag, and a model of >100T parameters (∼400x larger than Gopher) for other benchmarks. This result suggests that training ever-larger models may not help us reach human performance, at least in the near future. Indeed, given the enormous compute costs for training even larger LMs than the Gopher model with 280B parameters, we conjecture that there are more efficient ways of acquiring commonsense knowledge in an unsupervised fashion, for instance through multi-modal learning and grounding [11].

## 5.4 Few-shot Performance

Recent work has shown that large LMs can perform surprisingly well at various tasks in a few-shot fashion [23, 111]. Under this setup, the model is provided with $n$ examples of the downstream task, which are then appended to the prefix. Concretely, for the four commonsense benchmarks, we append $n$ examples that include the question and the correct answer; these examples — which are randomly sampled from the training split of each benchmark — appear before the evaluated question, as shown in Fig. 5.1. This few-shot formulation is appealing as it relies only on a small number of task-specific examples to get the LM accustomed to the task, *without* any fine-tuning. To what extent can we improve the model performance on commonsense benchmarks, by shifting from the zero-shot to the few-shot evaluation protocol?[6]

In Fig. 5.5, we compare the performance of Gopher under different evaluation protocols: (i) zero-shot and (ii) few-shot ($n$) where we use $n \in \{1, 10, 64\}$ examples. We run the few-shot experiments between 5 and 10 times — sampling different examples each time — and report the average performance. The variance across runs is very small and is shown as the error bar in Fig. 5.5.[7] Interestingly, model performance with few-shot (1) is sometimes *worse* than the zero-shot model, but the few-shot (10) and (64) models outperform their zero-shot counterpart (albeit sometimes by small margins). On HellaSwag and PIQA, we do not observe substantial improvement from few-shot evaluation compared to the zero-shot baseline (less than $2\%$).[8] While few-shot evaluation does not help much for most datasets, the only exception is Social IQa, where the few-shot (64) model outperforms the zero-shot model by a $> 7\%$ margin. We attribute this to the less natural text of Social IQa;[9] hence adding task-specific examples provides information about what is expected of the task.

Overall, we observe that the usefulness of the few-shot setting is benchmark dependent. Moreover, using task-specific examples in a few-shot setting does not bridge the gap to SOTA or human performance for any of the benchmarks.

### 5.4.1 Knowledge base retrieval.

We further examine if adding pre-extracted commonsense knowledge base triplets to the context — as a different form of few-shot/in-context learning — helps improve model

[6]The ability of large LMs to perform few-shot/in-context learning was first demonstrated by GPT3. Here we use an even-larger model than GPT3, which we expect to be able to leverage in-context learning to a similar extent as GPT3.

[7]Our findings on the small variance with different few-shot examples is consistent with [97], who found that replacing real examples with random labels can work as well.

[8]In few-shot experiments ($n = 50$), [23] also found small improvements for PIQA and HellaSwag (<1.5%), with a larger improvement (7.5%) for WinoGrande.

[9]We found that Gopher has the highest perplexity when predicting Social IQa answers compared to the other datasets.

performance. (See Appendix A.2 for details.) In contrast to work of [138], we observe no improvements when appending the triplets; we attribute this discrepancy to the strong performance of our base models (see §5.5).

## 5.5 Robustness of Reported Results

Different evaluation design choices — such as the format of the prompt or the choice of score functions — can impact the LM's zero-shot performance, and crucially result in different conclusions about a model's commonsense understanding ability. Moreover, the lack of a standardized zero-shot LM evaluation protocol makes direct comparisons between papers difficult [139, 20]. To what extent can we attribute variance in the reported results to these evaluation design choices — even though they have little to do with commonsense knowledge?

### 5.5.1 Model.

Quantifying the robustness of the reported results necessitates scoring a large number of examples under different evaluation design choices, which is infeasible to do with the largest (280B-parameter) model that has a slow inference speed. Hence, we conduct the following experiments using the 7B-parameter model, which is still $\sim$5 times larger than GPT2 [123].

### 5.5.2 Score functions.

Prior work employs different score functions to assess the plausibility of each answer choice given a question [23, 139, 20, 59], which makes a direct comparison between different results challenging. Here we investigate the impact of different score functions on the reported performance. In addition to cross-entropy (defined in §5.2.2), we experiment with two other score functions. The first is *sequence log probability*, defined as the log probability of the answer choice $\mathrm{y}$ conditional on the question $\mathrm{x}$. Letting $y_i$ be the $i$-th token in the answer $\mathrm{y}$:

$$s(\mathrm{y}|\mathrm{x}) = \sum_{i=0}^{||\mathrm{y}||} log(p(y_i|\mathrm{x}, y_0...y_{i-1})) \tag{5.2}$$

Another widely used score function [20, 59] is *point-wise mutual information*. This score function takes into account the probability of the answer choices alone, and the probability of the answer choices conditional on the question. This metric assesses whether the question adds additional information, as commonsense reasoning should be should be established within the context of the question. As this score function accounts for the prior probability of answer options, it can yield lower accuracy than score functions like cross-entropy that do *not* account for such factor (Answer-only baseline, §5.2.3).

$$s(\mathrm{y}|\mathrm{x}) = PMI(\mathrm{y}, \mathrm{x}) = log\frac{p(\mathrm{y}|\mathrm{x})}{p(\mathrm{y})} \tag{5.3}$$

### 5.5.3 Prompt format.

Another important factor is the format of the prompt; here we consider a few such choices. In addition to the concatenation of the question and the answer, we experiment with adding special symbols "[Question]" and "[Answer]" to specify the question and the answer [23]. Moreover, for Social IQa and PIQA, we experiment with a set of predefined rules [139] to convert the questions into sentences, which are closer to the LM's pre-training data format. Finally, we find that having the correct lower/upper case and punctuation is important; thus we manually checked all benchmarks to correct for case and punctuation.[10]

[10]Recent work learns the prefix that would maximize performance e.g.[77]. Here we focus on evaluation setups with no parameter updates, and leave this extension to future work. Our findings also indicate that the score function choice — which is not covered by lightweight fine-tuning approaches — is more important than the prompt format (§5.5.5).

### 5.5.4 Scored text.

The next option is whether to score the entire question–answer pair [139], or only the answer choice (conditional on the given question as prefix) as done by [23] i.e., whether to calculate $s(\mathrm{x};\mathrm{y})$ or $s(\mathrm{y}|\mathrm{x})$, where ; implies text concatenation.

### 5.5.5 Do These Design Choices Matter?

Table 5.4 shows the performance difference of using the worst versus the best design choices, which are independently optimized for each task. To sweep over the above design choices, instead of considering all combinations of parameters, we iterate the options in one category (e.g., score function), while fixing the parameters in the other categories.[11]

Overall, we observe a difference between the best and worst settings on all benchmarks; this gap is especially large for HellaSwag and PIQA. This result shows that *large language models do not simply work out of the box for some commonsense benchmarks*, because for some tasks, these evaluation design choices can account for a large variation in model performance. We find that the score function plays the most important role — cross-entropy yields the highest accuracy values across most benchmarks, but sequence log probability achieves a slightly better performance for WinoGrande. However, when using these scores, we should account for the Answer-only baseline (§5.3). Moreover, converting questions to sentences makes the largest difference for Social IQa. We also find that scoring the answer conditional on the question — as opposed to scoring the concatenation of questions and answers — works best, except for WinoGrande, which has no questions.

#### 5.5.5.1 Answer-length bias.

Although cross-entropy generally achieves the best reported performance, this score function is sensitive to answer lengths. As shown in Appendix A.4, cross-entropy tends to

[11]This decision saves compute resources, while offering a **lower bound** on the performance variations. Our goal here is not to seek the highest achievable performance, but to understand how much performance varies across different settings.

| | **Worst** | **Best** | **Difference** |
|---|---|---|---|
| **HellaSwag** | 50.8 | **70.5** | 19.7 |
| **PIQA** | 62.5 | **78.7** | 16.2 |
| **Social IQa** | 43.9 | **48.5** | 4.6 |
| **WinoGrande** | 59.7 | **62.0** | 2.3 |

Table 5.4: The performance difference between the worst and best design choices for each benchmark.

assign higher scores to longer answers; to varying extent, this pattern holds for PIQA, Social IQa, and WinoGrande. We attribute this to the higher probability assigned to subsequent tokens in the sequence, as such tokens have the most context and thus can be more easily predicted than tokens in the beginning of the answer. As longer answers have more such easier-to-predict tokens, their cross-entropy tends to be lower. This pattern is reversed in metrics such as sequence log probability, where shorter sequences often have higher scores [68, 142]. Note that this bias does not change the results reported in this work since there is no correlation between answer length and correctness (Appendix A.4).

#### 5.5.5.2 Takeaways.

We conclude this section with three concrete recommendations for future work.

- Although cross-entropy often achieves the best performance, it does not take into account the probability of selecting the correct answer without reasoning over the context (§5.3). We recommend future work to either: (i) use cross-entropy and report the *gap* with the answer-only baseline, or (ii) use the PMI score function, which *already* takes the probability of the answer into account.

- In the same way that we search for the best model hyper-parameters, future work should search over certain important evaluation design choices, such as the format of the prompt, and whether to convert the questions into declarative sentences.

- Lastly, we strongly encourage future work to report the variance of the observed results across different design choices. This can provide an indication of the *robustness* of the language models' performance on commonsense benchmarks.

## 5.6 Related Work

While recent work evaluates LMs against commonsense benchmarks in a zero- and few-shot fashion, they do not examine the extent to which model performance can be attributed to superficial cues or annotation artefacts in a given dataset (e.g., through strong baselines), nor do they quantify how robust the model performance is under different evaluation design choices. [147, 37] investigate the existence of dataset bias in commonsense co-reference resolution benchmarks [76, 129] and SWAG [166]; here we conduct a more comprehensive investigation on four diverse commonsense benchmarks.

Another line of work probe for commonsense knowledge in LMs through knowledge base completion [115, 34] or manually-designed probing tasks [158, 138]. [175] evaluate pre-trained LMs against commonsense benchmarks and propose a new dataset requiring multi-hop reasoning. In contrast, we focus on zero- and few-shot evaluation of commonsense understanding using the existing benchmarks.

## 5.7 Conclusion

We conduct a systematic and rigorous study of large LM performance on a diverse set of commonsense benchmarks, in a zero-shot and few-shot fashion. While pre-trained LMs can seemingly achieve a good zero-shot performance on these benchmarks, these results can be partially attributed to the LM's ability to exploit potential surface cues and annotation artefacts to guess the correct answer, without reasoning over the provided context. We further observed that substantially increasing model size yields rather small improvements on most commonsense benchmarks: Based on the scaling plots, achieving human-level performance requires much larger model sizes than what is currently feasible. In addition, model performance can be highly sensitive to certain evaluation design choices. Overall, our findings offer valuable insights and best practices for rigorously evaluating large LMs.

## Ethical Considerations

The primary aim of this paper is to conduct a systematic and rigorous commonsense evaluation of a large language model, which — in the case of this work — is achieved by using the pre-trained Gopher language model [124] with 280B parameters. Hence, the same risks stemming from large language model research are also broadly applicable to this work [8]. We briefly discuss these ethical considerations below.

### 5.7.1 Training compute.

In practice, pre-training large language models like Gopher requires an enormous amount of compute, which may contribute to increased carbon emissions [143, 110]. In this work, we do not pre-train the language model from scratch, although we acknowledge that conducting inference and evaluation with large language models like Gopher still has substantial computational costs. Given the need to construct even-larger language models ($>$100 trillion parameters) to achieve human-level performance on most of these benchmarks in an unsupervised fashion (§5.3.3), we encourage future work to focus on potentially more efficient ways of acquiring commonsense knowledge directly from data, e.g., through multi-modal learning, grounding, and human interaction [11].

### 5.7.2 Fairness and bias.

Given the enormous size of the pre-training data — about 2 trillion tokens in the case of Gopher pre-training — it is conceivable that the training dataset may inadvertently contain toxic and biased material. Such toxic material — which is not always easily identifiable in the large training dataset — can in turn encourage the model to produce biased, harmful, or toxic output, especially when they are prompted with toxic text [46]. In fact, [124] demonstrated that — up to a certain model size — larger language models may respond to toxic prompts with greater toxicity compared to smaller ones. Furthermore, the enormous size of the training data does not necessarily guarantee diversity: We expect the training data to contain a smaller proportion of vernacular or regional English that is used by underrepresented communities [13, 8]. Furthermore, the language model may also acquire harmful biases and stereotypes, e.g., assign lower probabilities to women becoming doctors as opposed to men [128, 26].

### 5.7.3 Language model misuse.

Our work highlights both the success and limitations of large language models at multiple commonsense benchmarks. Nevertheless, the success and expressive power of large language models come at the expense of potential misuse. Given their ability to generate realistic-looking — albeit not necessarily factual — content, large language models can also be used for malicious purposes. For instance, large language models can be used to generate convincing fake news [168], and more powerful generator can in turn generate even more convincing and influential fake news. Given the difficulty of manually distinguishing between human-generated text and machine-generated ones [30], how we can better detect and defend against malicious use of large language models is an important and exciting avenue for future work.

# CHAPTER 6

# COMMONSENSE FRAME COMPLETION

## 6.1 Introduction

Chapter 4 introduced ProtoQA, a question-answering task focused on evaluating common sense using generative measures. In ProtoQA, each question has multiple correct answers and explicitly inquires about a prototypical scenario, for example, “Name something you usually do before you leave the house for work.” ProtoQA served as the initial exploration of generative evaluation for commonsense knowledge.

While generative evaluation avoids the difficulty of generating hard negatives, it does not reflect the fact that there are often multiple correct answers, nor does it incorporate the probabilistic nature of language semantics and commonsense knowledge [39]. For example, given a sentence "The plumber is fixing the sink", we can infer using our common sense that the most probable locations include the kitchen and the bathroom, and with some lower probability perhaps a basement or utility closet.

In this chapter, we take the perspective that *commonsense knowledge is an implicit probability distribution over missing information in a context.* Emphasizing the implicit nature of common sense in a given context enhances the utility of our proposed task for downstream applications, such as home assistants, where the need for common sense is very rarely *explicit.* For example, a home assistant providing cooking directions should only implicitly be aware that "boil the water and add the spaghetti" requires the water to be in a container. Explicitly instructing a human with every minute detail would render the assistant useless, and thus it is paramount that the assistant understand what information can be implicitly inferred from context. Leveraging a probabilistic evaluation also emphasizes

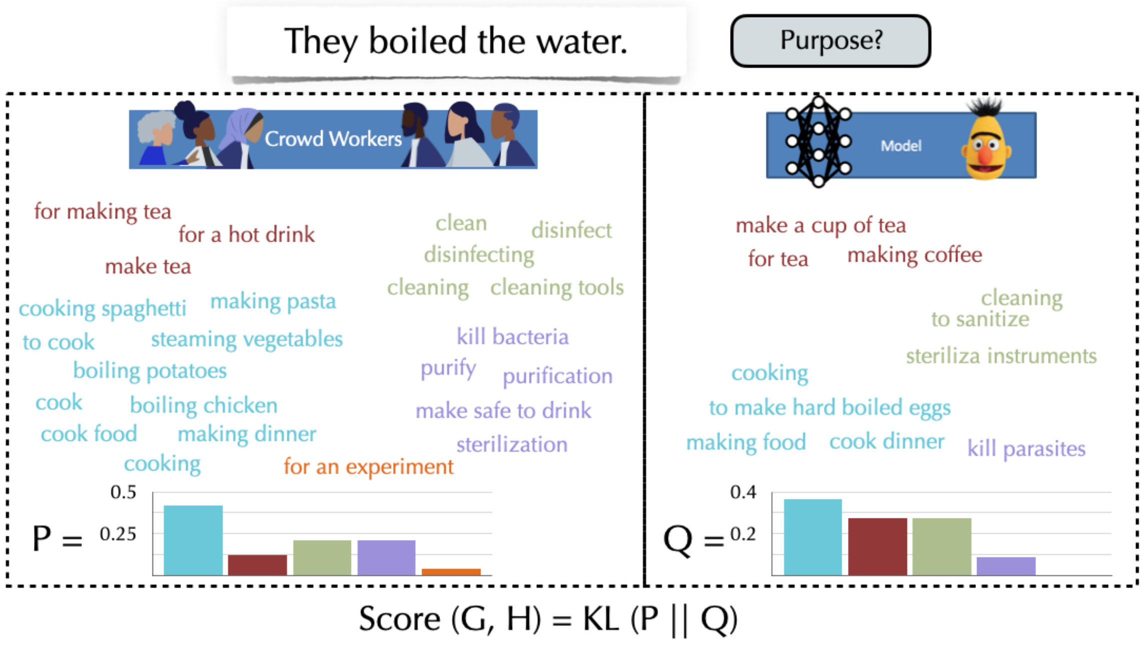


Figure 6.1: Example from the CFC dataset. Given a short sentence and a slot of interest (in this case, the purpose of boiling water). Human annotators provide ground-truth answer sets $\boldsymbol{G}$, and model prediction is denoted as answer sets $\boldsymbol{H}$. Each example in the dataset contains multiple current answers. To evaluate these answers as a probability distribution, we construct a categorical distribution for each answer set, and we calculate KL Divergence between these distributions (details in Section 6.4)

the uncertain nature of common sense - for example, the water may be heated on a stove, but it also may be heated using a kettle. This distribution also changes with respect to context - for example, consider how the implicit distribution would change if the instruction was "boil the water and add 4-methoxy-3-buten-2-one".

So, we propose the task of commonsense frame completion (CFC), in which models are provided with a context sentence and asked to generate potential values for a missing information or "slot-fillers" for the semantic frame in the sentence, where potential slots include "time", "location", "cause", etc. - see Table 6.1. We wish to evaluate the proposed slot-fillers probabilistically by comparing them to a large number of ground-truth crowd-sourced answers. Having an automatic evaluation is crucial to accelerating the development of strong models, however our setting (probabilistic evaluation of generative text) is novel, and thus we performed a rigorous study of potential contenders. We ultimately define a

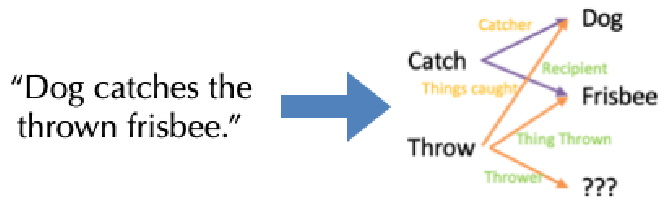


Figure 6.2: Representing context sentence using semantic representation (AMR) identifies the missing slots.

novel approach which aligns answers and measures the KL divergence between probabilities directly, which we justify on both theoretical and empirical grounds, where we observe a reasonable correlation with human judgements.

## 6.2 CFC Task Description

Given a direction such as "put the water on the burner to boil," it is *physical* common sense which allows us to know if we need to move other objects out of the way, and *conceptual* common sense which allows us to understand that the water is likely in a kettle and not simply dumped on the burner. In this chapter we aim to create a task which evaluates both these aspects of common sense. If we had a way of identifying that the object containing the water is unspecified, we could pose this as a question answering task (i.e. "What is the water contained in?"). Unlike most question answering tasks, however, there is no single correct answer. In this example, the water could be placed in a "kettle", "pot", "cup", or "glass", although the former answers are more probable. This distribution is also *contextual* - consider how the relative probability shifts if we append the phrase "and add the spaghetti", or changes drastically if we append "and add 4-methoxy-3-buten-2-one," in which case the vessel is likely a beaker or test-tube.

It is clearly necessary for any machine learning model which claims to capture common sense to have some sense of the distribution over the implicit information, and moreover it may be absolutely integral to the safety of any model which provides directions to share

| Missing Slot | Definition | Examples |
|---|---|---|
| Arg0 | Who/what does the event? | Sentence: putting cheese on the pizza. Arg0?<br>Answers: person, cook |
| Purpose | What is the goal for doing the event? | Sentence: putting cheese on the pizza. Purpose?<br>Answers: get nutrition, stop being hungry |
| Instrument | What kind of tools are used to accomplish the event? | Sentence: putting cheese on the pizza. Instrument?<br>Answers: hands, spoon |
| Time | What is a particular time (time of day, season, etc.) for doing the event? | Sentence: putting cheese on the pizza. Time?<br>Answers: lunch time, dinner time |
| Location | Where would the event usually happen? | Sentence: putting cheese on the pizza. Location?<br>Answers: kitchen, restaurant |

Table 6.1: Examples for different missing slot types

the same distribution as humans. To assess a model's ability in this regard, we consider the context sentence as a structured semantic frame, identify a missing slot, and ask the model to provide a distribution of potential slot fillers as shown in Figure 6.2.

## 6.3 Dataset Creation and Analysis

In this section we describe the method of creating a dataset amenable to evaluating the task of CFC. The first item to be addressed is where to collect reasonable context sentences which contain some natural element of common sense. CommonGen [85] is a recently released commonsense dataset which contains many short sentences describing basic information about daily life, and so we use this dataset as the source for potential context sentences.

Given a short sentence, we next need a way of identifying potential missing information. To this end, we perform semantic parsing on the sentence, aligning it with a structured semantic frame, and identify potential missing slots. We use AMR [4] for semantic parsing based on its ability to provide a rich representation of the sentence with a pre-defined fixed schema for the predicate roles. If a predicate is found, AMR parsing will match it to a schema and fill in the values for any identified slots. Any slots marked with `amr-unkown` indicate potential items of missing information, enabling us to obtain human annotations for the missing slot values.

We uniformly randomly sampled 63,788 sentences from the CommonGen dev dataset, and parsed them using the AMR parser from [24], generating 228,170 pairs of context questions with missing slots. From this, we randomly sampled 101 (sentence, missing slot) pairs for crowd workers to annotate, such that we had a balanced distribution of missing slot types, as detailed in Section 6.3.2.2. We present the context sentence and missing slot to crowdworkers, who were also provided with training examples and descriptions of the meaning of each slot type (see Table 6.1). The number of answers is chosen such that the resulting answer distribution is stable (see Section 6.3.2.1). Each element of the raw dataset therefore includes a context sentence, missing slot value, and a collection of slot fillers.

### 6.3.1 Probability Distribution

In an open-ended task where multiple humans are asked to provide answers as raw strings of text there are a multitude of answers which may essentially capture the same underlying idea. Ultimately we are not interested in the minute variations of the surface form, but rather in capturing the essence of the underlying concept. In the case of the boiling water example, for instance, we may want to treat "kettle" and "teapot" as though they were representative of the same general concept. As originally proposed in [17], we consider *clustering* the responses, converting a set of answer strings into a categorical distribution over answer clusters, where the probability of obtaining an answer from a given cluster is proportional to the number of answer strings contained within it. We explore both manual clustering and automated clustering methods (see Section 6.4.2).

### 6.3.2 Analysis

#### 6.3.2.1 Number of Answers

The number of potential slot fillers might be very large, and we want to ensure we sample enough to approximate the true distribution over answer concepts. An essential question, therefore, is how many samples are enough to approximate the true distribution with reasonable error rate? This is a classic problem in statistics, for which the Neyman-

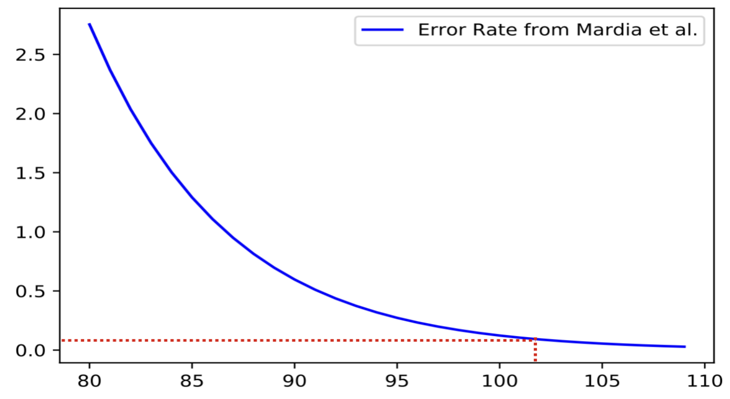


Figure 6.3: The relationship between the number of examples (x-axis), and the approximation error rate (y-axis).

Pearson lemma proves that the uniformly most powerful test is to consider the KL divergence $D_{\mathrm{KL}}(g\|f) = \sum_x g(x) \log \frac{g(x)}{f(x)}$ where $g$ is the empirical distribution and $f$ is the true distribution [53]. The recent work from [93] showed that this can be bounded by the following equation

$$\mathbb{P}(D_{KL}(g_{n,k}\|f) \geq \epsilon) \leq e^{-n\epsilon} \left[ \frac{3c_1}{c_2} \sum_{i=0}^{k-2} K_{i-1}(\frac{e\sqrt{n}}{2\pi})^i \right]$$

where $c_1$ and $c_2$ are constant values, $n$ is the number of samples, and $k$ is the number of categories in the categorical distribution.

For our setting, we manually clustered 50 questions, and found that the number of categories is not more than 8. To get a bound on the number of answers we should collect, we set $\epsilon = 0.2$, $k = 8$, and solve $e^{-n\epsilon} \left[ \frac{3c_1}{c_2} \sum_{i=0}^{k-2} K_{i-1}(\frac{e\sqrt{n}}{2\pi})^i \right]$ for $n$. Figure 6.3 shows the value of this bound on the $y$-axis for increasing numbers of samples $n$ on the $x$-axis. As we can see from the graph, for $100$ samples, the error rate is less than $0.5$, allowing us to approximate the true answer distribution with $95\%$ confidence if there are fewer than 8 categories in the categorical distribution.

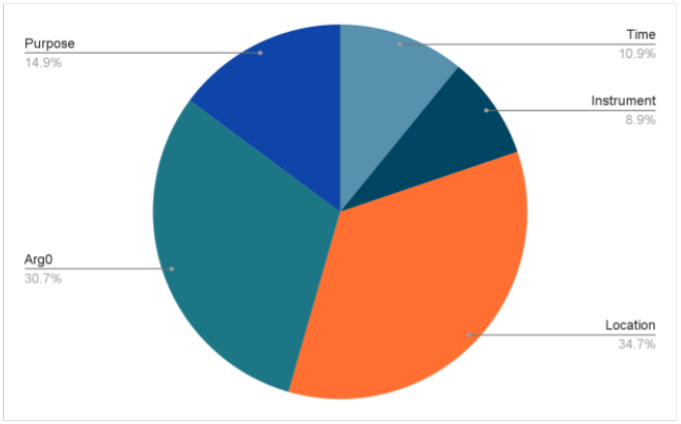


Figure 6.4: Question type distribution for CFC.

#### 6.3.2.2 Question Types

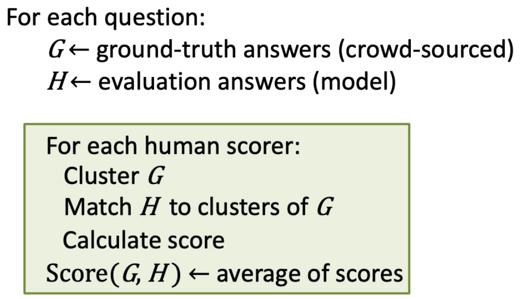
We collected **101** (context, missing slot) pairs, and obtained 100 slot fillers for each from crowdworkers, resulting in 10,100 annotations overall. The annotators are paid 0.15 per answer, and they are all English speakers who are based in the US. We split the data, creating a dev set with 55 examples and a test set with 46 examples. The distribution of missing slot types are shown in Figure 6.4. Each question type is associated with a different type of commonsense reasoning, e.g time represents temporal commonsense reasoning. The dataset will be released.

## 6.4 Probabilistic Evaluation

In this section, we detail the method of evaluating the CFC task on the provided dataset. As commonsense is inherently probabilistic, a rigorous probabilistic evaluation is required; however the task is presented (both to humans and models) as a generative question answering task. Therefore, we need a way to compare two large sets of answer strings. We will proceed by how human evaluators may go about comparing these sets of answers to determine if they were drawn from similar distributions and then describe the various ways by which this process can be automated.

### 6.4.1 Human Evaluation

Our proposed framework for evaluating model prediction is depicted in Figure 6.5: Given a question, the ground truth answer set $\mathbf{G}$ and the model generated answers $\mathbf{H}$, the goal is to evaluate the similarity between these two answer sets.



```
For each question:
    $G$ ← ground-truth answers (crowd-sourced)
    $H$ ← evaluation answers (model)

    For each human scorer:
        Cluster $G$
        Match $H$ to clusters of $G$
        Calculate score
    Score($G$, $H$) ← average of scores
```

Figure 6.5: Human Evaluation Process

This is a difficult task even for a humans, particularly if the answer sets are large and diverse, however bearing in mind that we are more interested in *concepts* being captured rather than unique surface forms, a human might choose to cluster the answer strings in $\mathbf{G}$.[1] The expert annotator could then match the answers in $\mathbf{H}$ to the proposed ground-truth clusters in $\mathbf{G}$. At this point we can define categorical probability distributions over the clusters, $\boldsymbol{P_g}$ and $\boldsymbol{P_h}$, where the probability assigned to a given cluster is equal to the number of answer strings assigned to it.[2] The similarity between $\mathbf{G}$ and $\mathbf{H}$ can be inferred by comparing the KL divergence of the two distributions, $\boldsymbol{D}_{\mathrm{KL}}(\hat{\boldsymbol{P}}_{\boldsymbol{g}}||\hat{\boldsymbol{P}}_{\boldsymbol{h}})$. To ensure evaluation robustness, we propose to repeat the same process with multiple human annotators and average the KL score to remove noise. In the end, the average KL value is the manual assessment of the quality of the model's answers.

[1]When clustering, a new category "wrong" could be added to the answer set to account for the wrong answers for a question. These will then be discarded prior to model evaluation.

[2]To eliminate zero probabilities, we use Laplace smoothing on all categories before calculating the probabilities, — adding one dummy answer to all categories.

Although this approach yields reliable results, it poses the following challenges: 1. Human experts must cluster the answers in $\mathbf{G}$, which is an expensive, labor-intensive task. 2. Manually matching answers to clusters at evaluation time is infeasible.

### 6.4.2 Automatic Evaluation

Due to the disadvantages mentioned above of human evaluation, we aim to design an automatic method that could ease the human evaluation process while achieving a high correlation with human evaluation results.

The high-level approach is: 1. Embed ground-truth answers from $\mathbf{G}$ into a dense vector space. 2. Automatically cluster the embeddings to obtain ground-truth clusters of $\mathbf{G}$. 3. Match elements of $\mathbf{H}$ to clusters of $\mathbf{G}$ by assignment function score.

Each step presents a number of options, which we detail in the following sections. We evaluate the quality of a particular approach by calculating the Spearman correlation of KL divergence using the automatic evaluation compared with that of the manual evaluation across a variety of answer distributions (see Section 6.4.3 and 6.4.4).

#### 6.4.2.1 Embedding

We first embed the discrete word tokens in $\mathbf{G}$ and $\mathbf{H}$ as word vectors. We experimented with various word embedding models, both without context (Word2Vec [95], GloVe [114] and FastText [14]) and with context (BERT [35], and RoBERTa [89]) We found FastText to perform best, and use it for all future embedding components.

#### 6.4.2.2 Clustering

Given the vector representation of the word answers, we experimented with various clustering algorithms including X-means [113], G-means [173] and hierarchical agglomerative clustering (HAC) [101] We used the implementation from pyclustering [106]. The parameters used by these clustering algorithms are treated as hyper-parameters and are tuned

| **Clustering** | Human | Human | Human | Hierarchical | Hierarchical |
|---|---|---|---|---|---|
| **Matching** | Human | WordNet | Embedding | WordNet | Embedding |
| Vanila Sample | 1 | 0.351 | 0.333 | 0.199 | 0.148 |
| Diverse Sample | 1 | 0.800 | 0.890 | 0.748 | 0.754 |
| Centered Sample | 1 | 0.752 | 0.714 | 0.700 | 0.593 |

Table 6.2: Average Spearman correlation between human evaluation and automatic evaluation under different sampling strategies for ProtoQA dev questions. The top two rows indicate the supervision source: cluster results can be annotated by human or clustering algorithms, and matching could be done via human annotation or automatic similarity functions (wordnet or embedding-based function)

based on the correlation score as we discuss in section 6.4.3 and 6.4.4. We found HAC to perform best.

#### 6.4.2.3 Matching

Given the predicted answers, we want to match the answers to one or multiple ground truth answer clusters. This was also a requirement for ProtoQA [17], and we leverage the WordNet matching function which performed best in that setting. As we also have embeddings for our answers, we consider approaches based on embedding-based similarity functions.[3] We train a Gaussian regression model for each cluster in the ground-truth answers. The regression takes one answer representation as input, and output is the label of whether the answer belongs to one particular cluster. If an answer matches with multiple clusters we divide the weight evenly among all matching clusters.

### 6.4.3 Evaluator on ProtoQA

In order to validate the automatic evaluator's performance, we compared the automatic evaluator results with the human evaluation results on two generative datasets. We first evaluated the proposed evaluator using ProtoQA.

[3]We tried cosine similarity with FastText embeddings, but it is hard to decide the threshold for answers that belong to the "wrong" cluster. We tuned a few values and found that the results are unstable, so we don't report these results here.

#### 6.4.3.1 Sampling

A robust automatic evaluation method should align well with human judgment on the best and worst predicted answers, and any in between. To achieve this, we propose three different sampling strategies to generate different answer distributions for each question.

- **Vanilla Sample.** We take random samples from model predictions directly.
- **Diverse Sample.** We take a linear combination of the ground-truth distribution and a uniform distribution to create a new distribution that interpolates between the ideal ground truth answers to random noise:

$$p = \alpha \hat{P}_g + (1 - \alpha)\text{uniform}$$

- **Centered Sample.** Arguably, the most important area to assess the quality of the evaluator is around answers which are likely to be returned from a model. We achieve this by taking a linear combination of the answer distributions of a given baseline model, the ground-truth distribution, and a uniform distribution, with most of the weight assigned to the answers from a baseline model:

$$\begin{aligned}
&p = z\hat{P}_h + w_1^{'}\hat{P}_g + w_2^{'}\text{uniform} \\
&w_1^{'} = \frac{w_1 * (1 - z)}{w_1 + w_2} \\
&w_2^{'} = \frac{w_2 * (1 - z)}{w_1 + w_2} \\
&z \sim U(0.5, 1),\ w_1 \sim U(0, 1), w_2 \sim U(0, 1)
\end{aligned}$$

The ProtoQA dev set has 100 ground-truth answers and 30 additional human responses that were collected to measure human performance. For each question, in addition to the 130 human responses, we also use the 300 generated answers from the fine-tuned GPT-2 model. All of these answers are annotated by expert annotators with cluster matching to

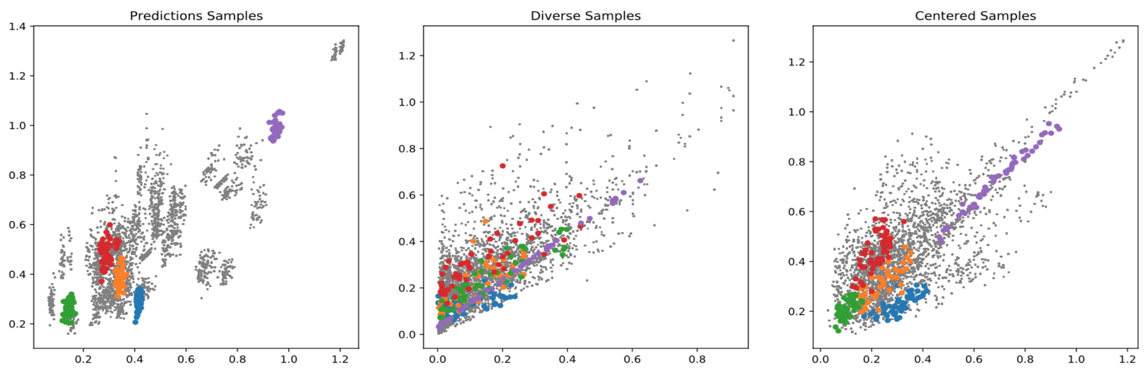


Figure 6.6: Correlation for sampled questions in ProtoQA with ground-truth clusters. The X-axis is the KL value with human assignment, and the y-axis is the KL value with WordNet assignment. This corresponds to the Human / WordNet column in Table 6.2. Different questions are annotated with different colors.

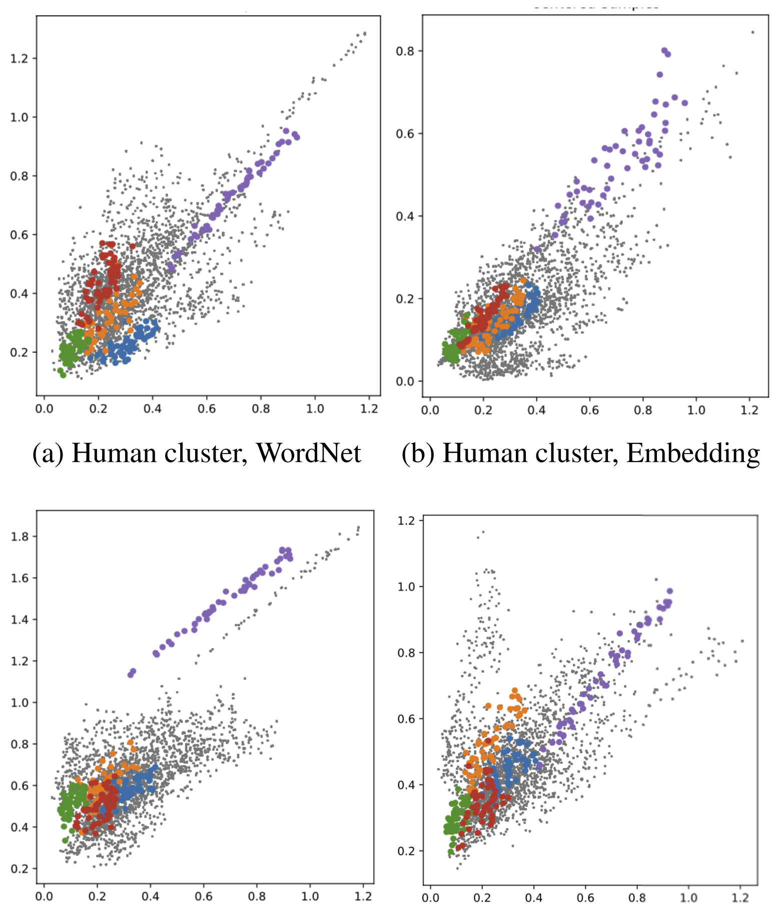
the ground-truth clusters. We use the union of the 30 human responses[4] and the GPT-2 answers as the prediction set, $\mathbf{H}$. We sample 50 answer sets for each question from $\mathbf{H}$ and $\mathbf{G}$ according to the sampling procedure mentioned above.

We use automatic clustering and matching to get the automatic $D_{\mathrm{KL}}(\hat{P}_g||\hat{P}_h)$. We can also evaluate the KL for manual clustering and matching, as all answers in ProtoQA have been annotated by human experts with clusters and assignments. After getting the human and automatic KL values for various sampled answer sets, we use the Spearman correlation coefficients across questions to measure the alignment between automatic and human evaluation.

#### 6.4.3.2 Results

As we can see from Table 6.2, the correlation value from the Vanilla sample is fairly low; however, the correlation number for both Diverse sample and Centered Sample strategy are both much higher. Inspecting Figure 6.6 shows that the Vanilla sample strategy does not provide diverse answer sets. This suggests that our automatic evaluation may struggle to

[4]we scale up the 30 additional human answers to 300, in order to balance the model predictions and human answers.


(a) Human cluster, WordNet (b) Human cluster, Embedding

(c) Auto cluster, WordNet (d) Auto cluster, Embedding

Figure 6.7: Centered sample correlation plots under different cluster and assignment methods: (a) human and WordNet (b) human and embedding (c) HAC and WordNet (d) HAC and embedding

provide fine-grained distinctions, however in reality we predominately care about scoring results from *different* models, which is better represented by the Centered Sample and Diverse Sample approaches.

We also note that automating the matching function only yields higher correlation with scores based on human annotations, which is promising as this would only require manual annotation at dataset creation time, not for each evaluation. As we can see from Figure 6.7, the automatic predicted score is positively correlated with the score based on human-annotations under most conditions.

| **Cluster** | Human | Hierarchical | Hierarchical |
|---|---|---|---|
| **Matching** | Human | WordNet | Embedding |
| Diverse Sample | 1 | 0.865 | 0.857 |

Table 6.3: Average spearman correlation between human and automatic evaluation under Diverse Sample for dev questions in CFC.

### 6.4.4 Evaluation on CFC

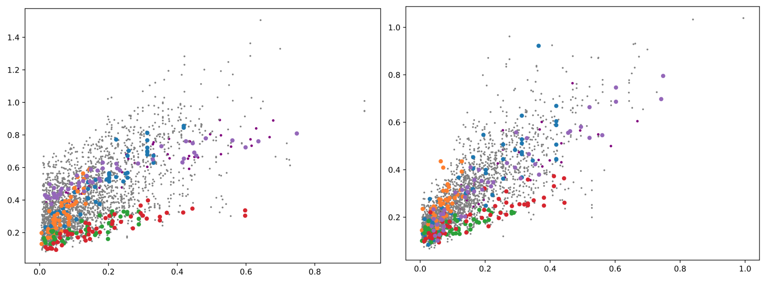
After preliminary experiments on ProtoQA, we verified our proposed evaluator on 55 dev questions in CFC. As in ProtoQA, expert annotators clustered the human responses into less than 8 clusters. Based on the results from the ProtoQA, we avoid the need to manually annotate model answers and instead focus on calculating the correlation between automated matching vs. automated matching and clustering. For this reason, we also solely evaluated using Diverse Sample. As shown in Table 6.3, the average correlation is fairly high ($> 0.85$).

We fixed the clustering parameters that gave us the best performance on these 55 questions to evaluate model performance on the test set. We also used these parameters to obtain the ground-truth evaluation number using both the WordNet similarity function and FastText similarity function. For WordNet we get a KL value of $0.237$, while for FastText we get a KL value of $0.091$. The human KL value should be $0$ since it is the ground-truth answer set. So we use embedding-based similarity methods to report model performance in Section 6.5. From Figure 6.8, we see that the WordNet score function tends to produce a higher KL value compared to Human judgment, which explains the higher KL even for ground-truth answer sets.

## 6.5 Model Performance

### 6.5.1 GPT2

Our baseline is a generative language model, as modern language models have improved representational power, and recent evidence has demonstrated their effectiveness



(a) Auto cluster, WordNet (b) Auto cluster, Embedding

Figure 6.8: Diverse sample correlation plots under hierarchical clustering, and different matching methods: (a) human cluster with WordNet matching (b) human cluster with embedding matching

| | | GPT2-L | GPT2-XL | ProtoQA FT | GPT2-L FT | Human | GT |
|---|---|---|---|---|---|---|---|
| Dev | ZS | 1.301 | 1.069 | 0.631 | 0.613 | 0.170 | 0.091 |
| | FS(1) | 0.848 | 0.740 | 0.562 | 0.585 | | |
| | | GPT2-L | GPT2-XL | ProtoQA FT | GPT2-L FT | Human | GT |
| Test | ZS | 1.197 | 0.962 | 0.576 | 0.612 | 0.040 | 0.076 |
| | FS(1) | 1.020 | 0.748 | 0.623 | 0.658 | | |

Table 6.4: Model performance on CFC Data (**lower is better**). ZS means zero-shot, and FS(1) means one-shot prediction. GPT2-L and GPT2-XL is the GPT2 large and XL model respectively, ProtoQA FT is the ProtoQA fine-tuned, while GPT2-L FT is our own fined-tuned model. The GT column represents the KL values with the ground-truth answers.

in modeling commonsense reasoning tasks [159, 146]. We use the Hugging Face PyTorch implementation [161]) of GPT-2 Large and XL [123]. Our evaluation includes zero-shot and one-shot evaluations, as well as an evaluation after fine-tuning with the ProtoQA training data.

We convert CFC questions to a format "[Q]: context sentence, question, [A]". For the one-shot experiment, we sample one question and one answer from the CFC dev data, then we do the same conversion but pre-pend the converted question-answer pair to the actual question. The assumption is that as part of the prompt provided to the model, the model could get familiar with the task format.

For fine-tuning experiments, we took the ProtoQA pre-trained model[5]. We also trained the GPT-2 Large model with a task format that is similar to our task with the same "[Q]: question. [A]" format using the ProtoQA training data denoted as GPT2-L FT in Table 6.4. The models are fine-tuned for 3 epochs on an nVidia M40 GPU.

In order to generate different answers for the same prompt, we use Nucleus Sampling [58]. We generate 200 sampled answers from the GPT-2 Large model and 100 answers for the GPT-2 XL model for each question and treat them as the model prediction set. We experimented with temperatures from 0.1 to 1.0, and chose the model parameters with the best dev performance, then reported the test performance here.

### 6.5.2 Human Performance

In order to get a human performance on this task, we collected 30 additional human responses and evaluated them the same was as a model prediction.

### 6.5.3 Discussion

As we can see from Table 6.4, the model performance and human performance still have a large gap in terms of KL value, while the human performance is very close to ground truth answers. This indicates that the dataset is a challenging dataset for models, while humans could perform very well on this.

Moreover, GPT2-XL performs better despite the fact that the number of sampled answers is much less than the GPT2-large model (100 samples vs. 200 samples). Both of these non-fine-tuned models benefit a lot from zero-shot to one-shot. When the model gets fined-tuned with the ProtoQA training data, the performance improvement is more significant. Nevertheless, all model performances are still far from human-level performance, which leaves us ample space to improve the model.

[5]`https://github.com/iesl/ProtoQA_GPT2`

## 6.6 Related Work

Creating commonsense benchmarks to evaluate model performance is a long-standing research topic [129, 85, 132]. However, most benchmarks are created using a multiple-choice selection paradigm, which is simpler to evaluate but misaligned with the real-world use-case of commonsense knowledge, and most egregiously ignores the existence of multiple correct answers. We are not the first ones to gather multiple human answers to facilitate robust evaluations, however. [3] and [17] also collected multiple human responses for each question to get aggregated human ground-truth answer sets.

Our work differs from these due to our emphasis on commonsense as *implicit* and *probabilistic*. We don't treat each answer equally; rather, we aim to match the answer distribution given by human responses. For this purpose, we propose a novel probabilistic evaluation for open-ended generation tasks with multiple correct answers. A similar probabilistic evaluation was studied from a language model generation point of view [116]. They proposed a KL-based evaluation to measure language model generations, while our focus is on the implicit answer distribution.

## 6.7 Conclusion

In this chapter, we assert that commonsense is an implicit probability distribution over missing information, and propose a dataset that aims to evaluate commonsense in this setting via a generative question answering task; moreover, we embrace the probabilistic nature of commonsense knowledge in both the dataset creation and the metric design. We propose a probabilistic automatic evaluation for evaluating answer distributions that is highly correlated to human judgment. Using this metric, we observe that model performance on our new dataset is significantly worse than human performance, indicating that the task is sufficiently challenging. In the future, we aim to further extend the size of the dataset, both in number of instances as well as answer length, which will involve challenging problems on both the dataset creation and probabilistic evaluation front.

# BIBLIOGRAPHY

[1] Ahn, Luis Von, Kedia, Mihir, and Blum, Manuel. Verbosity: a game for collecting common-sense facts. In *In Proceedings of ACM CHI 2006 Conference on Human Factors in Computing Systems, volume 1 of Games* (2006), ACM Press, pp. 75–78.

[2] Athiwaratkun, Ben, and Wilson, Andrew Gordon. Hierarchical density order embeddings. In *International Conference on Learning Representations* (2018).

[3] Aydin, Bahadir Ismail, Yilmaz, Yavuz Selim, Li, Yaliang, Li, Qi, Gao, Jing, and Demirbas, Murat. Crowdsourcing for multiple-choice question answering. In *AAAI* (2014), Citeseer, pp. 2946–2953.

[4] Banarescu, Laura, Bonial, Claire, Cai, Shu, Georgescu, Madalina, Griffitt, Kira, Hermjakob, Ulf, Knight, Kevin, Koehn, Philipp, Palmer, Martha, and Schneider, Nathan. Abstract meaning representation for sembanking. In *LAW@ACL* (2013).

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

[6] Bansal, Mohit, Burkett, David, De Melo, Gerard, and Klein, Dan. Structured learning for taxonomy induction with belief propagation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)* (2014), vol. 1, pp. 1041–1051.

[7] Bauer, Lisa, and Bansal, Mohit. Identify, align, and integrate: Matching knowledge graphs to commonsense reasoning tasks. *EACL* (2021).

[8] Bender, Emily M., Gebru, Timnit, McMillan-Major, Angelina, and Shmitchell, Shmargaret. On the dangers of stochastic parrots: Can language models be too big? . In *Proc. of FAccT* (2021).

[9] Bhagavatula, Chandra, Bras, Ronan Le, Malaviya, Chaitanya, Sakaguchi, Keisuke, Holtzman, Ari, Rashkin, Hannah, Downey, Doug, tau Yih, Wen, and Choi, Yejin. Abductive commonsense reasoning. In *International Conference on Learning Representations* (2020).

[10] Bhagavatula, Chandra, Bras, Ronan Le, Malaviya, Chaitanya, Sakaguchi, Keisuke, Holtzman, Ari, Rashkin, Hannah, Downey, Doug, Yih, Scott Wen-tau, and Choi, Yejin. Abductive commonsense reasoning. *arXiv preprint arXiv:1908.05739* (2019).

[11] Bisk, Yonatan, Holtzman, Ari, Thomason, Jesse, Andreas, Jacob, Bengio, Yoshua, Chai, Joyce, Lapata, Mirella, Lazaridou, Angeliki, May, Jonathan, Nisnevich, Aleksandr, Pinto, Nicolas, and Turian, Joseph. Experience grounds language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2020).

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

[13] Blodgett, Su Lin, Green, Lisa, and O'Connor, Brendan. Demographic dialectal variation in social media: A case study of African-American English. In *Proc. of EMNLP* (2016).

[14] Bojanowski, Piotr, Grave, Edouard, Joulin, Armand, and Mikolov, Tomas. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics 5* (2017), 135–146.

[15] Bollacker, Kurt, Evans, Colin, Paritosh, Praveen, Sturge, Tim, and Taylor, Jamie. Freebase: a collaboratively created graph database for structuring human knowledge. In *ACM SIGMOD International Conference on Management of Data* (2008).

[16] Bommasani, Rishi, Hudson, Drew A., Adeli, Ehsan, Altman, Russ, Arora, Simran, von Arx, Sydney, Bernstein, Michael S., Bohg, Jeannette, Bosselut, Antoine, Brunskill, Emma, Brynjolfsson, Erik, Buch, S., Card, D., Castellon, Rodrigo, Chatterji, Niladri S., Chen, Annie, Creel, Kathleen, Davis, Jared, Demszky, Dora, Donahue, Chris, Doumbouya, Moussa, Durmus, Esin, Ermon, Stefano, Etchemendy, John, Ethayarajh, Kawin, Fei-Fei, Li, Finn, Chelsea, Gale, Trevor, Gillespie, Lauren E., Goel, Karan, Goodman, Noah D., Grossman, Shelby, Guha, Neel, Hashimoto, Tatsunori, Henderson, Peter, Hewitt, John, Ho, Daniel E., Hong, Jenny, Hsu, Kyle, Huang, Jing, Icard, Thomas F., Jain, Saahil, Jurafsky, Dan, Kalluri, Pratyusha, Karamcheti, Siddharth, Keeling, Geoff, Khani, Fereshte, Khattab, O., Koh, Pang Wei, Krass, Mark S., Krishna, Ranjay, Kuditipudi, Rohith, Kumar, Ananya, Ladhak, Faisal, Lee, Mina, Lee, Tony, Leskovec, Jure, Levent, Isabelle, Li, Xiang Lisa, Li, Xuechen, Ma, Tengyu, Malik, Ali, Manning, Christopher D., Mirchandani, Suvir P., Mitchell, Eric, Munyikwa, Zanele, Nair, Suraj, Narayan, Avanika, Narayanan, Deepak, Newman, Ben, Nie, Allen, Niebles, Juan Carlos, Nilforoshan, Hamed, Nyarko, J. F., Ogut, Giray, Orr, Laurel, Papadimitriou, Isabel, Park, Joon Sung, Piech, Chris, Portelance, Eva, Potts, Christopher, Raghunathan, Aditi, Reich, Robert, Ren, Hongyu, Rong, Frieda, Roohani, Yusuf H., Ruiz, Camilo, Ryan, Jackson K., R'e, Christopher, Sadigh, Dorsa, Sagawa, Shiori, Santhanam, Keshav, Shih, Andy, Srinivasan, Krishna Parasuram, Tamkin, Alex, Taori, Rohan, Thomas, Armin W., Tramèr, Florian, Wang, Rose E., Wang, William, Wu, Bohan, Wu, Jiajun, Wu, Yuhuai, Xie, Sang Michael, Yasunaga, Michihiro, You, Jiaxuan, Zaharia, Matei A., Zhang, Michael, Zhang, Tianyi, Zhang, Xikun, Zhang, Yuhui, Zheng, Lucia, Zhou, Kaitlyn, and Liang, Percy. On the opportunities and risks of foundation models. *ArXiv abs/2108.07258* (2021).

[17] Boratko*, Michael, Li*, Xiang Lorraine, O'Gorman*, Tim, Das*, Rajarshi, Le, Dan, and McCallum, Andrew. ProtoQA: A question answering dataset for prototypical common-sense reasoning. In *Conference on Empirical Methods in Natural Language Processing, EMNLP* (2020).

[18] Boratko, Michael, Padigela, Harshit, Mikkilineni, Divyendra, Yuvraj, Pritish, Das, Rajarshi, McCallum, Andrew, Chang, Maria, Fokoue-Nkoutche, Achille, Kapanipathi, Pavan, Mattei, Nicholas, et al. A systematic classification of knowledge, reasoning, and context within the arc dataset. *arXiv preprint arXiv:1806.00358* (2018).

[19] Bordes, Antoine, Usunier, Nicolas, Garcia-Duran, Alberto, Weston, Jason, and Yakhnenko, Oksana. Translating embeddings for modeling multi-relational data. In *NIPS* (2013).

[20] Bosselut, Antoine, Le Bras, Ronan, and Choi, Yejin. Dynamic neuro-symbolic knowledge graph construction for zero-shot commonsense question answering. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI)* (2021).

[21] Bowling, Shannon R, Khasawneh, Mohammad T, Kaewkuekool, Sittichai, and Cho, Byung R. A logistic approximation to the cumulative normal distribution. *Journal of Industrial Engineering and Management 2*, 1 (2009).

[22] Bowman, Samuel R, Angeli, Gabor, Potts, Christopher, and Manning, Christopher D. A large annotated corpus for learning natural language inference. *arXiv preprint arXiv:1508.05326* (2015).

[23] Brown, Tom, Mann, Benjamin, Ryder, Nick, Subbiah, Melanie, Kaplan, Jared D, Dhariwal, Prafulla, Neelakantan, Arvind, Shyam, Pranav, Sastry, Girish, Askell, Amanda, Agarwal, Sandhini, Herbert-Voss, Ariel, Krueger, Gretchen, Henighan, Tom, Child, Rewon, Ramesh, Aditya, Ziegler, Daniel, Wu, Jeffrey, Winter, Clemens, Hesse, Chris, Chen, Mark, Sigler, Eric, Litwin, Mateusz, Gray, Scott, Chess, Benjamin, Clark, Jack, Berner, Christopher, McCandlish, Sam, Radford, Alec, Sutskever, Ilya, and Amodei, Dario. Language models are few-shot learners. In *Conference on Neural Information Processing Systems, NeurIPS* (2020).

[24] Cai, Deng, and Lam, Wai. AMR parsing via graph-sequence iterative inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (Online, July 2020), Association for Computational Linguistics, pp. 1290–1301.

[25] Callison-Burch, Chris, Koehn, Philipp, Monz, Christof, Peterson, Kay, Przybocki, Mark, and Zaidan, Omar. Findings of the 2010 joint workshop on statistical machine translation and metrics for machine translation. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR* (Uppsala, Sweden, July 2010), Association for Computational Linguistics, pp. 17–53.

[26] Cao, Yang Trista, and Daumé III, Hal. Toward gender-inclusive coreference resolution: An analysis of gender and bias throughout the machine learning lifecycle*. *Computational Linguistics* (2021).

[27] Chambers, Nathanael, and Jurafsky, Dan. Unsupervised learning of narrative schemas and their participants. In *ACL* (2009).

[28] Chen*, Xuelu, Boratko*, Michael, Chen, Muhao, Dasgupta, Shib Sankar, Li, Xiang Lorraine, and McCallum, Andrew. Probabilistic box embeddings for uncertain knowledge graph reasoning. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL* (2021).

[29] Chu, Y. J., and Liu, T. H. On the shortest arborescence of a directed graph. *Science Sinica 20* (1995).

[30] Clark, Elizabeth, August, Tal, Serrano, Sofia, Haduong, Nikita, Gururangan, Suchin, and Smith, Noah A. All that's 'human' is not gold: Evaluating human evaluation of generated text. 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)* (Online, Aug. 2021), Association for Computational Linguistics, pp. 7282–7296.

[31] Collins, Michael, Dasgupta, Sanjoy, and Schapire, Robert E. A generalization of principal components analysis to the exponential family. In *Advances in neural information processing systems* (2002), pp. 617–624.

[32] Dasgupta, Shib Sankar, Boratko, Michael, Atmakuri, Shriya, Li, Xiang Lorraine, Patel, Dhruvesh, and McCallum, Andrew. Word2box: Learning word representation using box embeddings. *arXiv preprint arXiv:2106.14361* (2021).

[33] Dasgupta, Shib Sankar, Li, Xiang Lorraine, Boratko, Michael, Zhang, Dongxu, and McCallum, Andrew. Box-to-box transformations for modeling joint hierarchies. In *Workshop on Representation Learning for NLP at ACL, ACL WS* (2021).

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

[35] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL* (2019).

[36] Devlin, Jacob, Chang, Ming-Wei, Lee, Kenton, and Toutanova, Kristina. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL* (2019).

[37] Elazar, Yanai, Zhang, Hongming, Goldberg, Yoav, and Roth, Dan. Back to square one: Bias detection, training and commonsense disentanglement in the winograd schema. In *EMNLP* (2021).

[38] Erk, Katrin. Representing words as regions in vector space. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning* (Stroudsburg, PA, USA, 2009), CoNLL '09, Association for Computational Linguistics, pp. 57–65.

[39] Erk, Katrin. The probabilistic turn in semantics and pragmatics. *Annual Review of Linguistics 8*, 1 (2022), 101–121.

[40] Ferraro, Francis, and Van Durme, Benjamin. A unified bayesian model of scripts, frames and language. In *Thirtieth AAAI Conference on Artificial Intelligence* (2016).

[41] Fillmore, Charles J, et al. Frame semantics and the nature of language. In *Annals of the New York Academy of Sciences: Conference on the origin and development of language and speech* (1976).

[42] Flavell, John H. Theory-of-mind development: Retrospect and prospect. *Merrill-Palmer Quarterly (1982-)* (2004), 274–290.

[43] Friedman, Jerome, Hastie, Trevor, and Tibshirani, Robert. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics 9*, 3 (2008), 432–441.

[44] Ganea, Octavian-Eugen, Bécigneul, Gary, and Hofmann, Thomas. Hyperbolic entailment cones for learning hierarchical embeddings. *ICML* (2018).

[45] Gebru, Timnit, Morgenstern, Jamie, Vecchione, Briana, Vaughan, Jennifer Wortman, Wallach, Hanna, Daumé III, Hal, and Crawford, Kate. Datasheets for datasets. *arXiv preprint arXiv:1803.09010* (2018).

[46] Gehman, Samuel, Gururangan, Suchin, Sap, Maarten, Choi, Yejin, and Smith, Noah A. RealToxicityPrompts: Evaluating neural toxic degeneration in language models. In *Findings of EMNLP* (2020).

[47] Glorot, Xavier, and Bengio, Yoshua. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 249–256.

[48] Gordon, Jonathan, and Van Durme, Benjamin. Reporting bias and knowledge acquisition. In *AKBC* (2013).

[49] Gulcehre, Caglar, Moczulski, Marcin, Denil, Misha, and Bengio, Yoshua. Noisy activation functions. In *International Conference on Machine Learning* (2016), pp. 3059–3068.

[50] Gulcehre, Caglar, Moczulski, Marcin, Visin, Francesco, and Bengio, Yoshua. Mollifying networks. *arXiv preprint arXiv:1608.04980* (2016).

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

[52] Gururangan, Suchin, Swayamdipta, Swabha, Levy, Omer, Schwartz, Roy, Bowman, Samuel R, and Smith, Noah A. Annotation artifacts in natural language inference data. In *NAACL* (2018).

[53] Harremoës, Peter, and Tusnády, Gábor. Information divergence is more $\chi$ 2-distributed than the $\chi$ 2-statistics. In *2012 IEEE International Symposium on Information Theory Proceedings* (2012), IEEE, pp. 533–537.

[54] He, Shizhu, Liu, Kang, Ji, Guoliang, and Zhao, Jun. Learning to represent knowledge graphs with gaussian embedding. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management* (New York, NY, USA, 2015), CIKM '15, ACM, pp. 623–632.

[55] Heckerman, David, Geiger, Dan, and Chickering, David M. Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning 20*, 3 (1995), 197–243.

[56] Heilbron, Fabian Caba, Escorcia, Victor, Ghanem, Bernard, and Niebles, Juan Carlos. Activitynet: A large-scale video benchmark for human activity understanding. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 961–970.

[57] Hochreiter, Sepp, and Schmidhuber, Jürgen. Long short-term memory. *Neural computation 9*, 8 (1997), 1735–1780.

[58] Holtzman, Ari, Buys, Jan, Forbes, Maxwell, and Choi, Yejin. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751* (2019).

[59] Holtzman, Ari, West, Peter, Schwartz, Vered, Choi, Yejin, and Zettlemoyer, Luke. Surface form competition: Why the highest probability answer isn't always right. *arXiv preprint arXiv:2104.08315* (2021).

[60] Huang, Lifu, Bras, Ronan Le, Bhagavatula, Chandra, and Choi, Yejin. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. *EMNLP abs/1909.00277* (2019).

[61] Huang, Lifu, Le Bras, Ronan, Bhagavatula, Chandra, and Choi, Yejin. Cosmos QA: Machine reading comprehension with contextual commonsense reasoning. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)* (Hong Kong, China, Nov. 2019), Association for Computational Linguistics, pp. 2391–2401.

[62] Jebara, Tony, Kondor, Risi, and Howard, Andrew. Probability product kernels. *Journal of Machine Learning Research 5*, Jul (2004), 819–844.

[63] Jia, Robin, and Liang, Percy. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (Copenhagen, Denmark, Sept. 2017), Association for Computational Linguistics, pp. 2021–2031.

[64] Jia, Robin, Raghunathan, Aditi, Göksel, Kerem, and Liang, Percy. Certified robustness to adversarial word substitutions. In *EMNLP* (2019).

[65] Joshi, Mandar, Choi, Eunsol, Weld, Daniel S, and Zettlemoyer, Luke. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL* (2017).

[66] Kaplan, Jared, McCandlish, Sam, Henighan, Tom, Brown, Tom B., Chess, Benjamin, Child, Rewon, Gray, Scott, Radford, Alec, Wu, Jeffrey, and Amodei, Dario. Scaling laws for neural language models. *CoRR abs/2001.08361* (2020).

[67] Kaushik, Divyansh, Hovy, Eduard, and Lipton, Zachary C. Learning the difference that makes a difference with counterfactually-augmented data. *arXiv preprint arXiv:1909.12434* (2019).

[68] Koehn, Philipp, and Knowles, Rebecca. Six challenges for neural machine translation. In *NMT@ACL* (2017).

[69] Koller, Daphne, and Friedman, Nir. *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[70] Koupaee, Mahnaz, and Wang, William Yang. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305* (2018).

[71] Kuhn, Harold W. The hungarian method for the assignment problem. *Naval research logistics quarterly* (1955).

[72] Lafferty, John D., McCallum, Andrew, and Pereira, Fernando. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML* (2001).

[73] Lai, Alice, and Hockenmaier, Julia. Learning to predict denotational probabilities for modeling entailment. In *EACL* (2017).

[74] Lavie, Alon, and Denkowski, Michael J. The meteor metric for automatic evaluation of machine translation. *Machine translation 23*, 2-3 (2009), 105–115.

[75] Leader, Solomon. Measures on semilattices. *Pacific Journal of Mathematics 39*, 2 (1971), 407–423.

[76] Levesque, Hector, Davis, Ernest, and Morgenstern, Leora. The winograd schema challenge. In *International Conference on the Principles of Knowledge Representation and Reasoning* (2012).

[77] Li, Xiang Lisa, and Liang, Percy. Prefix-tuning: Optimizing continuous prompts for generation. In *Proc. of ACL-IJCNLP* (2021).

[78] Li, Xiang Lorraine, Kuncoro, Adhi, d'Autume, Cyprien de Masson, Blunsom, Phil, and Nematzadeh, Aida. A systematic investigation of commonsense understanding in large language models. *arXiv preprint arXiv:2111.00607* (2021).

[79] Li, Xiang Lorraine, Taheri, Aynaz, Tu, Lifu, and Gimpel, Kevin. Commonsense knowledge base completion. In *Annual Meeting of the Association for Computational Linguistics, ACL* (2016).

[80] Li, Xiang Lorraine, Vilnis, Luke, and McCallum, Andrew. Improved representation learning for predicting commonsense ontologies. In *Workshop on Deep Structured Prediction at the International Conference on Machine Learning, ICML WS* (2017).

[81] Li*, Xiang Lorraine, Vilnis*, Luke, Zhang, Dongxu, Boratko, Michael, and McCallum, Andrew. Smoothing the geometry of probabilistic box embeddings. In *International Conference on Learning Representations, ICLR* (2019).

[82] Lieberman, Henry, and et al. Common consensus: a web-based game for collecting commonsense goals, 2007.

[83] Lin, Bill Yuchen, Shen, Minghan, Zhou, Wangchunshu, Zhou, Pei, Bhagavatula, Chandra, Choi, Yejin, and Ren, Xiang. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Conference on Empirical Methods in Natural Language Processing, EMNLP Findings* (2020).

[84] Lin, Bill Yuchen, Sun, Haitian, Dhingra, Bhuwan, Zaheer, Manzil, Ren, Xiang, and Cohen, William W. Differentiable open-ended commonsense reasoning. *NAACL* (2021).

[85] Lin, Bill Yuchen, Zhou, Wangchunshu, Shen, Ming, Zhou, Pei, Bhagavatula, Chandra, Choi, Yejin, and Ren, Xiang. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020* (Online, Nov. 2020), Association for Computational Linguistics, pp. 1823–1840.

[86] Liu, Chia-Wei, Lowe, Ryan, Serban, Iulian, Noseworthy, Mike, Charlin, Laurent, and Pineau, Joelle. How NOT to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing* (2016).

[87] Liu, Hugo, and Singh, Push. Commonsense reasoning in and over natural language. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems* (2004), Springer, pp. 293–306.

[88] Liu, Yinhan, Ott, Myle, Goyal, Naman, Du, Jingfei, Joshi, Mandar, Chen, Danqi, Levy, Omer, Lewis, Mike, Zettlemoyer, Luke, and Stoyanov, Veselin. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692* (2019).

[89] Liu, Yinhan, Ott, Myle, Goyal, Naman, Du, Jingfei, Joshi, Mandar, Chen, Danqi, Levy, Omer, Lewis, Mike, Zettlemoyer, Luke, and Stoyanov, Veselin. Roberta: A robustly optimized bert pretraining approach. *ArXiv abs/1907.11692* (2019).

[90] LoBue, Peter, and Yates, Alexander. Types of common-sense knowledge needed for recognizing textual entailment. In *ACL* (2011).

[91] Lourie, Nicholas, Bras, Ronan Le, Bhagavatula, Chandra, and Choi, Yejin. Unicorn on rainbow: A universal commonsense reasoning model on a new multitask benchmark. In *AAAI* (2021).

[92] Luo, Xiaoqiang, Pradhan, Sameer, Recasens, Marta, and Hovy, Eduard. An extension of blanc to system mentions. In *ACL* (2014).

[93] Mardia, Jay, Jiao, Jiantao, Tánczos, Ervin, Nowak, Robert D, and Weissman, Tsachy. Concentration inequalities for the empirical distribution of discrete distributions: beyond the method of types. *Information and Inference: A Journal of the IMA 9*, 4 (2020), 813–850.

[94] McCarthy, John, et al. *Programs with common sense*. RLE and MIT computation center, 1960.

[95] Mikolov, Tomas, Chen, Kai, Corrado, Gregory S., and Dean, Jeffrey. Efficient estimation of word representations in vector space. In *ICLR* (2013).

[96] Miller, George A. WordNet: a lexical database for English. *Communications of the ACM* (1995).

[97] Min, Sewon, Lyu, Xinxi, Holtzman, Ari, Artetxe, Mikel, Lewis, Mike, Hajishirzi, Hannaneh, and Zettlemoyer, Luke. Rethinking the role of demonstrations: What makes in-context learning work? *arXiv preprint* (2022).

[98] Mobahi, Hossein. Training recurrent neural networks by diffusion. *arXiv preprint arXiv:1601.04114* (2016).

[99] Mostafazadeh, Nasrin, Chambers, Nathanael, He, Xiaodong, Parikh, Devi, Batra, Dhruv, Vanderwende, Lucy, Kohli, Pushmeet, and Allen, James. A corpus and evaluation framework for deeper understanding of commonsense stories. In *NAACL* (2016).

[100] Munkres, James. Algorithms for the assignment and transportation problems. *SIAM* (1957).

[101] Murtagh, Fionn, and Legendre, Pierre. Ward's hierarchical agglomerative clustering method: which algorithms implement ward's criterion? *Journal of classification 31*, 3 (2014), 274–295.

[102] Neelakantan, Arvind, Vilnis, Luke, Le, Quoc V, Sutskever, Ilya, Kaiser, Lukasz, Kurach, Karol, and Martens, James. Adding gradient noise improves learning for very deep networks. *arXiv preprint arXiv:1511.06807* (2015).

[103] Nickel, Maximilian, and Kiela, Douwe. Poincaré embeddings for learning hierarchical representations. *CoRR* (2017).

[104] Nickel, Maximilian, Tresp, Volker, and Kriegel, Hans-Peter. A three-way model for collective learning on multi-relational data. In *ICML* (2011).

[105] Nickel, Maximillian, and Kiela, Douwe. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. Curran Associates, Inc., 2017, pp. 6338–6347.

[106] Novikov, Andrei. PyClustering: Data mining library. *Journal of Open Source Software 4*, 36 (apr 2019), 1230.

[107] Novikova, Jekaterina, Dušek, Ondřej, Cercas Curry, Amanda, and Rieser, Verena. Why we need new evaluation metrics for NLG. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing* (2017).

[108] Papineni, Kishore, Roukos, Salim, Ward, Todd, and Zhu, Wei-Jing. Bleu: a method for automatic evaluation of machine translation. In *ACL* (2002).

[109] Patel*, Dhruvesh, Dasgupta*, Shib Sankar, Li, Xiang Lorraine, Vilnis, Luke, and McCallum, Andrew. Representing joint hierarchies with box embeddings. In *Conference on Automated Knowledge Base Construction, AKBC* (2020).

[110] Patterson, David A., Gonzalez, Joseph, Le, Quoc V., Liang, Chen, Munguia, Lluis-Miquel, Rothchild, Daniel, So, David R., Texier, Maud, and Dean, Jeff. Carbon emissions and large neural network training. *CoRR abs/2104.10350* (2021).

[111] Patwary, Mostofa, Shoeybi, Mohammad, LeGresley, Patrick, Prabhumoye, Shrimai, Casper, Jared, Korthikanti, Vijay, Singh, Vartika, Bernauer, Julie, Houston, Michael, Catanzaro, Bryan, Smith, Shaden, Norick, Brandon, Rajbhandari, Samyam, Liu, Zhun, Zerveas, George, Zhang, Elton, Aminabadi, Reza Yazdani, Song, Xia, He, Yuxiong, Zhu, Jeffrey, Cruzan, Jennifer, Madan, Umesh, Vargas, Luis, and Tiwary, Saurabh. Using deepspeed and megatron to train megatron-turing nlg 530b, the world's largest and most powerful generative language model, 2021.

[112] Pearl, Judea. *Probabilistic reasoning in intelligent systems*. 1988.

[113] Pelleg, Dan, Moore, Andrew W, et al. X-means: Extending k-means with efficient estimation of the number of clusters. In *Icml* (2000), vol. 1, pp. 727–734.

[114] Pennington, Jeffrey, Socher, Richard, and Manning, Christopher D. Glove: Global vectors for word representation. In *EMNLP* (2014).

[115] Petroni, Fabio, Rocktäschel, Tim, Lewis, Patrick, Bakhtin, Anton, Wu, Yuxiang, Miller, Alexander H, and Riedel, Sebastian. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066* (2019).

[116] Pillutla, Krishna, Swayamdipta, Swabha, Zellers, Rowan, Thickstun, John, Welleck, Sean, Choi, Yejin, and Harchaoui, Zaid. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Advances in Neural Information Processing Systems 34* (2021), 4816–4828.

[117] Poliak, Adam, Naradowsky, Jason, Haldar, Aparajita, Rudinger, Rachel, and Durme, Benjamin Van. Hypothesis only baselines in natural language inference. In **SEMEVAL* (2018).

[118] Poliak, Adam, Naradowsky, Jason, Haldar, Aparajita, Rudinger, Rachel, and Van Durme, Benjamin. Hypothesis only baselines for natural language inference. In *The Seventh Joint Conference on Lexical and Computational Semantics (*SEM)* (2018).

[119] Poliak, Adam, Naradowsky, Jason, Haldar, Aparajita, Rudinger, Rachel, and Van Durme, Benjamin. Hypothesis only baselines in natural language inference. In **SEM* (2018).

[120] Porada, Ian, Suleman, Kaheer, Trischler, Adam, and Cheung, Jackie Chi Kit. Modeling event plausibility with consistent conceptual abstraction. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics, NAACL* (2021).

[121] Radev, Dragomir R, Teufel, Simone, Saggion, Horacio, Lam, Wai, Blitzer, John, Qi, Hong, Celebi, Arda, Liu, Danyu, and Drabek, Elliott. Evaluation challenges in large-scale document summarization. In *ACL* (2003).

[122] Radford, Alec, Narasimhan, Karthik, Salimans, Tim, and Sutskever, Ilya. Improving language understanding by generative pre-training. *Tech report,OpenAI* (2018).

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

[124] Rae, Jack W., Borgeaud, Sebastian, Cai, Trevor, Millican, Katie, Hoffmann, Jordan, Song, H. Francis, Aslanides, John, Henderson, Sarah, Ring, Roman, Young, Susannah, Rutherford, Eliza, Hennigan, Tom, Menick, Jacob, Cassirer, Albin, Powell, Richard, van den Driessche, George, Hendricks, Lisa Anne, Rauh, Maribeth, Huang, Po-Sen, Glaese, Amelia, Welbl, Johannes, Dathathri, Sumanth, Huang, Saffron, Uesato, Jonathan, Mellor, John, Higgins, Irina, Creswell, Antonia, McAleese, Nat, Wu, Amy, Elsen, Erich, Jayakumar, Siddhant M., Buchatskaya, Elena, Budden, David, Sutherland, Esme, Simonyan, Karen, Paganini, Michela, Sifre, Laurent, Martens, Lena, Li, Xiang Lorraine, Kuncoro, Adhiguna, Nematzadeh, Aida, Gribovskaya, Elena, Donato, Domenic, Lazaridou, Angeliki, Mensch, Arthur, Lespiau, Jean-Baptiste, Tsimpoukelli, Maria, Grigorev, Nikolai, Fritz, Doug, Sottiaux, Thibault, Pajarskas, Mantas, Pohlen, Toby, Gong, Zhitao, Toyama, Daniel, de Masson d'Autume, Cyprien, Li, Yujia, Terzi, Tayfun, Mikulik, Vladimir, Babuschkin, Igor, Clark, Aidan, de Las Casas, Diego, Guy, Aurelia, Jones, Chris, Bradbury, James, Johnson, Matthew, Hechtman, Blake A., Weidinger, Laura, Gabriel, Iason, Isaac, William S., Lockhart, Edward, Osindero, Simon, Rimell, Laura, Dyer, Chris, Vinyals, Oriol, Ayoub, Kareem, Stanway, Jeff, Bennett, Lorrayne, Hassabis, Demis, Kavukcuoglu, Koray, and Irving, Geoffrey. Scaling language models: Methods, analysis & insights from training gopher. *CoRR abs/2112.11446* (2021).

[125] Recasens, Marta, and Hovy, Eduard. Blanc: Implementing the rand index for coreference evaluation. *Natural Language Engineering* (2011).

[126] Roemmele, Melissa, Bejan, Cosmin Adrian, and Gordon, Andrew S. Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium* (2011).

[127] Roy, Daniel M, and Teh, Yee W. The mondrian process. In *Advances in neural information processing systems* (2009), pp. 1377–1384.

[128] Rudinger, Rachel, Naradowsky, Jason, Leonard, Brian, and Van Durme, Benjamin. Gender bias in coreference resolution. In *Proc. of NAACL-HLT* (2018).

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

[130] Sap, Maarten, Le Bras, Ronan, Allaway, Emily, Bhagavatula, Chandra, Lourie, Nicholas, Rashkin, Hannah, Roof, Brendan, Smith, Noah A, and Choi, Yejin. Atomic: An atlas of machine commonsense for if-then reasoning. In *AAAI* (2019).

[131] Sap, Maarten, Rashkin, Hannah, Chen, Derek, Le Bras, Ronan, and Choi, Yejin. Social IQa: Commonsense reasoning about social interactions. 4463–4473.

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

[133] Schank, Roger C, and Abelson, Robert P. Scripts, plans, and knowledge. In *IJCAI* (1975), vol. 75, pp. 151–157.

[134] Schank, Roger C, and Abelson, Robert P. *Scripts, plans, goals, and understanding : An inquiry into human knowledge structures.* Psychology Press, 1977.

[135] Schwartz, Roy, Sap, Maarten, Konstas, Ioannis, Zilles, Li, Choi, Yejin, and Smith, Noah A. The effect of different writing tasks on linguistic style: A case study of the roc story cloze task. In *CoNLL* (2017).

[136] Sellam, Thibault, Das, Dipanjan, and Parikh, Ankur P. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696* (2020).

[137] Sheng, Emily, Chang, Kai-Wei, Natarajan, Premkumar, and Peng, Nanyun. The woman worked as a babysitter: On biases in language generation. 3407–3412.

[138] Shwartz, Vered, and Choi, Yejin. Do neural language models overcome reporting bias? In *COLING* (2020).

[139] Shwartz, Vered, West, Peter, Bras, Ronan Le, Bhagavatula, Chandra, , and Choi, Yejin. Unsupervised commonsense question answering with self-talk. In *EMNLP* (2020).

[140] Socher, Richard, Chen, Danqi, Manning, Christopher D, and Ng, Andrew. Reasoning with neural tensor networks for knowledge base completion. In *NIPS* (2013).

[141] Speer, Robyn, Chin, Joshua, and Havasi, Catherine. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI Conference on Artificial Intelligence, AAAI* (2017).

[142] Stahlberg, Felix, and Byrne, Bill. On NMT search errors and model errors: Cat got your tongue? 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. 3354–3360.

[143] Strubell, Emma, Ganesh, Ananya, and McCallum, Andrew. Energy and policy considerations for deep learning in NLP. In *Proc. of ACL* (2019).

[144] Subramanian, Sandeep, and Chakrabarti, Soumen. New embedded representations and evaluation protocols for inferring transitive relations. *SIGIR 2018* (2018).

[145] Talmor, Alon, Herzig, Jonathan, Lourie, Nicholas, and Berant, Jonathan. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL* (2019).

[146] Tamborrino, Alexandre, Pellicano, Nicola, Pannier, Baptiste, Voitot, Pascal, and Naudin, Louise. Pre-training is (almost) all you need: An application to commonsense reasoning.

[147] Trichelair, Paul, Emami, Ali, Trischler, Adam, Suleman, Kaheer, and Cheung, Jackie Chi Kit. How reasonable are common-sense reasoning tasks: A case-study on the Winograd schema challenge and SWAG. 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. 3382–3387.

[148] Trouillon, Théo, Welbl, Johannes, Riedel, Sebastian, Gaussier, Éric, and Bouchard, Guillaume. Complex embeddings for simple link prediction. In *International Conference on Machine Learning* (2016), pp. 2071–2080.

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

[150] Vendrov, Ivan, Kiros, Ryan, Fidler, Sanja, and Urtasun, Raquel. Order-embeddings of images and language. In *ICLR* (2016).

[151] Vilnis, Luke, Li, Xiang, Murty, Shikhar, and McCallum, Andrew. Probabilistic embedding of knowledge graphs with box lattice measures. In *ACL* (2018), Association for Computational Linguistics.

[152] Vilnis*, Luke, Li*, Xiang Lorraine, Murty, Shikhar, and McCallum, Andrew. Probabilistic embedding of knowledge graphs with box lattice measures. In *Annual Meeting of the Association for Computational Linguistics, ACL* (2018).

[153] Vilnis, Luke, and McCallum, Andrew. Word representations via gaussian embedding. In *ICLR* (2015).

[154] Wang, Ben, and Komatsuzaki, Aran. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`, May 2021.

[155] Wang, Xiaoyan, Kapanipathi, Pavan, Musa, Ryan, Yu, Mo, Talamadupula, Kartik, Abdelaziz, Ibrahim, Chang, Maria, Fokoue, Achille, Makni, Bassem, Mattei, Nicholas, et al. Improving natural language inference using external knowledge in the science questions domain. In *AAAI* (2019).

[156] Wang, Zhen, Zhang, Jianwen, Feng, Jianlin, and Chen, Zheng. Knowledge graph embedding by translating on hyperplanes. In *AAAI* (2014).

[157] Weber, Noah, Rudinger, Rachel, and Van Durme, Benjamin. Causal inference of script knowledge. *arXiv preprint arXiv:2004.01174* (2020).

[158] Weir, Nathaniel, Poliak, Adam, and Durme, Benjamin Van. Probing neural language models for human tacit assumptions. *arXiv: Computation and Language* (2020).

[159] Weir, Nathaniel, Poliak, Adam, and Van Durme, Benjamin. On the existence of tacit assumptions in contextualized language models.

[160] Williams, Christopher KI, and Rasmussen, Carl Edward. Gaussian processes for regression. In *Advances in neural information processing systems* (1996), pp. 514–520.

[161] Wolf, Thomas, Debut, Lysandre, Sanh, Victor, Chaumond, Julien, Delangue, Clement, Moi, Anthony, Cistac, Pierric, Rault, Tim, Louf, R'emi, Funtowicz, Morgan, and Brew, Jamie. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv abs/1910.03771* (2019).

[162] Wu, Sixing, Li, Ying, Zhang, Dawei, Zhou, Yang, and Wu, Zhonghai. Diverse and informative dialogue generation with context-specific commonsense knowledge awareness. In *Proceedings of the 58th Annual Meeting of the Association of Computational Linguistics* (2020).

[163] Yogatama, Dani, d'Autume, Cyprien de Masson, Connor, Jerome, Kocisky, Tomas, Chrzanowski, Mike, Kong, Lingpeng, Lazaridou, Angeliki, Ling, Wang, Yu, Lei, Dyer, Chris, et al. Learning and evaluating general linguistic intelligence. *arXiv preprint arXiv:1901.11373* (2019).

[164] Zaanen, Adriaan C. *Introduction to Operator Theory in Riesz Spaces*. Springer Berlin Heidelberg, 1997.

[165] Zellers, Rowan, Bisk, Yonatan, Farhadi, Ali, and Choi, Yejin. From recognition to cognition: Visual commonsense reasoning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2019).

[166] Zellers, Rowan, Bisk, Yonatan, Schwartz, Roy, and Choi, Yejin. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *EMNLP* (2018).

[167] Zellers, Rowan, Holtzman, Ari, Bisk, Yonatan, Farhadi, Ali, and Choi, Yejin. Hellaswag: Can a machine really finish your sentence? In *Annual Meeting of the Association for Computational Linguistics, ACL* (2019).

[168] Zellers, Rowan, Holtzman, Ari, Rashkin, Hannah, Bisk, Yonatan, Farhadi, Ali, Roesner, Franziska, and Choi, Yejin. *Defending against Neural Fake News*. 2019.

[169] Zhang, Houyu, Liu, Zhenghao, Xiong, Chenyan, and Liu, Zhiyuan. Grounded conversation generation as guided traverses in commonsense knowledge graphs.

[170] Zhang, Sheng, Rudinger, Rachel, Duh, Kevin, and Van Durme, Benjamin. Ordinal common-sense inference. *TACL* (2017).

[171] Zhang, Tianyi, Kishore, Varsha, Wu, Felix, Weinberger, Kilian Q, and Artzi, Yoav. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675* (2019).

[172] Zhang, Yichi, Ou, Zhijian, and Yu, Zhou. Task-oriented dialog systems that consider multiple appropriate responses under the same context. *ArXiv abs/1911.10484* (2019).

[173] Zhao, Zhonghua, Guo, Shanqing, Xu, Qiuliang, and Ban, Tao. G-means: A clustering algorithm for intrusion detection. In *International Conference on Neural Information Processing* (2008), Springer, pp. 563–570.

[174] Zhong, Wanjun, Tang, Duyu, Duan, Nan, Zhou, Ming, Wang, Jiahai, and Yin, Jian. Improving question answering by commonsense-based pre-training. In *CCF International Conference on Natural Language Processing and Chinese Computing* (2019).

[175] Zhou, Xuhui, Zhang, Yue, Cui, Leyang, and Huang, Dandan. Evaluating commonsense in pre-trained language models. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2020), vol. 34, pp. 9733–9740.

# APPENDIX

# APPENDIX FOR PROBABILISTIC BOX EMBEDDINGS

## A.1 Queries with Negated Variables

Section 2.3.2 mentions that although the complement of a box is not a box, queries involving negated variables can be calculated exactly with Inclusion-Exclusion, demonstrated in Table A.1. While there are many more interesting and efficient approaches, we simply use the formula for calculating the volume of the union of hyperrectangles (a standard Inclusion-Exclusion formula).

This is equivalent since the intersection of complements of boxes is the complement of the union of boxes. We first intersect all of the non-negated variables into one conjunction box, $\boldsymbol{T}$. We then calculate the volume of the union of $\boldsymbol{T}$ with all of the boxes representing complements of negated variables $\boldsymbol{F = \neg f_1, \neg f_2, \neg f_3, ..., v_1 = (T \cup f_1 \cup f_2 \cup f_3 ...) = 1 - P(\neg T, \neg f_1, \neg f_2, \neg f_3, ...)}$, and the volume of just the negated variables' boxes, $\boldsymbol{v_2 = (f_1 \cup f_2 \cup f_3 ...) = 1 - P(\neg f_1, \neg f_2, \neg f_3, ...)}$. The probability of the query is $\boldsymbol{v_1 - v_2 = P(F) - P(\neg T, F) = (P(T, F) + P(\neg T, F)) - P(\neg T, F) = P(T, F)}$, which was the original query.

| P(deer \| ... ) | |
|---|---|
| P(deer) | 0.12 |
| ¬white | 0.13 |
| animal | 0.50 |
| ¬white,animal | 0.54 |
| ¬white,animal,herbivore | 0.73 |
| ¬white, animal, herbivore, ¬rabbit | 0.80 |
| ¬white, animal, ¬herbivore,¬rabbit | 0.00 |

Table A.1: Negated variables: queries on the toy data with negated variables, calculated with Inclusion-Exclusion.

## A.2 Properties of the Box Lattice

In this section, we cover some technical details about the box lattice model and its properties especially as compared to the order embedding model.

### A.2.1 Non-Distributivity

A lattice is called *distributive* if the following identity holds for all members $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$:

$$\boldsymbol{x} \wedge (\boldsymbol{y} \vee \boldsymbol{z}) = (\boldsymbol{x} \wedge \boldsymbol{y}) \vee (\boldsymbol{x} \wedge \boldsymbol{z})$$

**Claim.** *Order embeddings form a distributive lattice.*

*Proof.* This is a standard results on vector lattices shown in e.g. [164]

A non-distributive lattice is a strictly more general object, capable of modeling more objects since it does not necessarily need to fulfill the above identity for all triples $\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$.

**Claim.** *The box lattice is non-distributive.*

*Proof.* Consider the box lattice in 1-dimension. Let $\boldsymbol{x} = [\boldsymbol{0}, \boldsymbol{0.3}]$, $\boldsymbol{y} = [\boldsymbol{0.2}, \boldsymbol{0.6}]$, and $\boldsymbol{z} = [\boldsymbol{0.5}, \boldsymbol{1.0}]$. Then $\boldsymbol{x} \wedge (\boldsymbol{y} \vee \boldsymbol{z}) = [\boldsymbol{0.2}, \boldsymbol{0.3}]$, but $(\boldsymbol{x} \wedge \boldsymbol{y}) \vee (\boldsymbol{x} \wedge \boldsymbol{z}) = [\boldsymbol{0}, \boldsymbol{0.6}] \vee \perp = [\boldsymbol{0}, \boldsymbol{0.6}]$.

This proves that the box lattice is a strict generalization of order embeddings, and not equivalent to order embeddings of any dimensionality. Additionally, our choice of an example containing disjoint elements hints at the importance of non-distributivity for our goal of modeling disjoint events.

### A.2.2 Pseudocomplemented

A lattice is called *pseudocomplemented* if for every element $\boldsymbol{x}$ there exists a unique greatest element in the lattice $\boldsymbol{x}^*$ that is disjoint from $\boldsymbol{x}$ and $\boldsymbol{x} \wedge \boldsymbol{x}^* = \perp$. The box lattice is almost always pseudocomplemented, aside from symmetry concerns (for example, a perfectly centered cube in the 2-dimensional box lattice of side length $< \boldsymbol{1}$ has 4 possible equally large pseudocomplements. However any such symmetries can always be infinitesimally perturbed without breaking order structure so the box lattice is pseudocomplemented in a measure-theoretic sense. However, these pseudocomplements can be arbitrarily bad approximations of the true complement set of a box, with the worst case scenario coming from large, nearly-centered cubes.

## A.3 Asymmetrizing Score Matrices

### A.3.1 Probabilistic Models

Assume we have a pairwise CPD between Bernoulli variables, and also have access to the unary marginals for each Bernoulli, and further that no unary marginals are exactly identical. If they are exactly identical, we can generate random independent Bernoulli parameters and their JPD, and take a small convex combination with that to infinitesimally perturb the statistics, so this proof is valid everywhere but on a set of measure 0 which we can approximate arbitrarily well.

**Claim.** *If all unary marginals are distinct, taking the elements of the pairwise CPD, removing the diagonal, and deleting an entry if $\boldsymbol{P}(\boldsymbol{A}|\boldsymbol{B}) < \boldsymbol{P}(\boldsymbol{B}|\boldsymbol{A})$, that is if $\boldsymbol{A}_{ij} < \boldsymbol{A}_{ji}$, will result in a weighted adjacency matrix for an acyclic directed graph*

*Proof.* Order the variables $x_1 ... x_n$ so that $p(x_i) < p(x_j)$ if $i < j$. Now an entry of the CPD $p(x_i|x_j) = p(x_i, x_j)/p(x_j) = C_{ij}$ is less than $C_{ji} = p(x_i, x_j)/p(x_i)$ if $p(x_i) < p(x_j)$. So with the variables so ordered, if we use the CPD to create an adjacency matrix with an edge $C_{ij} = 1$ if and only if $p(x_i) < p(x_j)$, it will be upper triangular with 0 on the diagonal. This is a nilpotent matrix which means it is the adjacency matrix of an acyclic graph. This can be easily seen since the entries of $A^k$ are the set of $K$-hop neighbors, and if this set eventually becomes empty, as in a nilpotent matrix, we have no cycles.

Since the labeling of our vertices is arbitrary, this means that our adjacency matrix created by the proposed asymmetrizing procedure is always acyclic since it is similar to an upper triangular matrix with 0s on the diagonal.

This holds as long as the unary marginals can always be ordered (which they can be except on a set of measure zero, and in practice on it seems to work even if you ignore this constraint.

### A.3.2 KL Divergences and Gaussian Embeddings

Assume the same setup as section A.3.1, but the scores in the matrix come from (possibly thresholded if $A_{ij} - A_{ji} < c$) pairwise divergences between Gaussian embeddings.

**Claim.** *There exist graphs produced by the above procedure that do not lead to directed acyclic graphs if thresholded by deleting entries when $A_{ij} < A_{ji}$:*

*Proof.* Consider the following set of 5 2-dimensional Gaussians with diagonal covariance:

$$G_1 = \mathcal{N}(x_1; [-5, -3], \text{diag}([3, 7]))$$
$$G_2 = \mathcal{N}(x_2; [-3, 5], \text{diag}[(7, 4]))$$
$$G_3 = \mathcal{N}(x_3; [-5, -6], \text{diag}([8, 1]))$$
$$G_4 = \mathcal{N}(x_4; [-7, 6], \text{diag}([5, 5]))$$
$$G_5 = \mathcal{N}(x_5; [9, 3], \text{diag}([5, 9]))$$

Applying asymmetrization and even pruning at a threshold of $c = 1$ (which is non-nilpotent and does affect edges) produces a cycle between nodes 5, 1, and 3. There are certain repeated numbers in the parameters, but this is not the cause of the issue. They are whole numbers for ease of exposition, they were randomly generated and many more examples can be created with arbitrary floating point numbers.

### A.3.3 Order Embeddings

We simulated many millions of random sets of order embedding parameters, and created pairwise graphs using the order embedding energy function, and were never able to find a cycle in the resulting asymmetrized graphs. We conjecture that this is because the order embedding energy is essentially a Lagrangian relaxation term penalizing the violation of a true partial order relation, but have not proven it.

**Conjecture.** *Sets of Order Embeddings can be consistently asymmetrized into directed acyclic graphs according to the procedure in section A.3.1.*

## A.4 Model Parameters

### A.4.1 WordNet Parameters

Since the WordNet data has binary $\mathbf{0}, \mathbf{1}$ links instead of calibrated probabilities, and the negative links are found from random negative sampling, we constrain the delta embedding to not update for negative samples during optimization. We found this was effective in preventing random negative samples from decreasing the volume of the boxes and creating artificially disjoint pairs.

The WordNet parameters that achieved best performance on the development set (whose train set performance we reported) are:

```
batch size: 800
dimension: 50
edge loss weight: 1.0
unary loss weight: 9.0
learning rate: 0.001
minimum dimension delta size: 1e-6
dimension-max regularization weight: 0.005
optimizer: Adam
```

For WordNet training with additional soft CPD edges, we use the same parameters. We also perform pruning on the generated CPD file. We only include $\langle \boldsymbol{t_1}, \boldsymbol{t_2} \rangle$ pairs with probability $\geq$ 0.6 and the reverse pair $\langle \boldsymbol{t_2}, \boldsymbol{t_1} \rangle \leq$ 0.4 probability.

We tune the batch size of the model between 800 and 40000 because bigger batch size facilitates faster training. We also sweep over 1.0 to 9.0 for edge loss weight and 9.0 to 1.0 for the unary loss weight. The learning rate we tune in $\boldsymbol{\lambda} \in$ {0.001, 0.0001}. The minimum dimension delta size we tune in $\in$ {0.01, 0.001, 0.0001, 0.00001, 0.000001}. The dimension-max regularization encourages the upper bound of box to be close 1.0 with an L1 penalty to prevent collapse. We perform parameter search in {0.0, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5}.

### A.4.2 Flickr Parameters

The Flickr parameters that achieved best performance on the development set (whose train set performance we reported) are:

```
batch size: 512
dropout: 0.5
unary loss weight: 8.0
edge loss weight: 2.0
learning rate: 0.0001
minimum dimension delta size: 1e-6
optimizer: Adam
```

The LSTM parameters are initialized with Glorot initialization [47], as are the weight and bias parameters for the feedforward networks to produce the box minimums. The network to produce the $\boldsymbol{\Delta}$ embedding is initialized from a uniform distribution from $[\mathbf{15.0, 15.50}]$. We clip to zero for min embeddings (apply a ReLU), and apply a softplus to enforce the positivity and minimum dimension size constraints on the $\boldsymbol{\Delta}$ embeddings.

We also sweep over 1.0 to 9.0 for edge loss weight and 9.0 to 1.0 for the unary loss weight. The learning rate $\boldsymbol{\lambda} \in \{0.001, 0.0001\}$. We tried Glorot initialization with the $\boldsymbol{\Delta}$ network as well, but since we wanted a high degree of overlap at the beginning of training, we simply swept over different uniform initialization ranges in $[\mathbf{5.0, 5.5}]$, $[\mathbf{10.0, 10.5}]$ and $[\mathbf{15.0, 15.5}]$.

# APPENDIX

# APPENDIX FOR SMOOTHED BOX EMBEDDINGS

## A.1 Proof of Gaussian Overlap Formula

We wish to evaluate, for two lattice elements $\mathrm{x}$ and $\mathrm{y}$, with associated smoothed indicators $f$ and $g$,

$$f(x; a, b, \sigma^2) = \mathbb{1}_{[a,b]}(x) * \phi(x; \sigma^2) = \int_{\mathbb{R}} \mathbb{1}_{[a,b]}(z)\phi(x - z; \sigma^2)dz = \int_a^b \phi(x - z; \sigma^2)dz$$

$$p_\phi(\mathrm{x} \wedge \mathrm{y}) = \int_{\mathbb{R}} f(x; a, b, \sigma_1^2)g(x; c, d, \sigma_2^2)dx \tag{B.1}$$

Since the Gaussian kernel is normalized to have total integral equal to 1, so as not to change the overall areas of the boxes, the concrete formula is

$$\phi(z; \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-z^2}{2\sigma^2}}$$

Since the antiderivative of $\phi$ is the normal CDF, this may be recognized as the difference $\Phi(x; a, \sigma^2) - \Phi(x; b, \sigma^2)$, but this does not allow us to easily evaluate the integral of interest, which is the integral of the product of two such functions.

To evaluate (B.1), recall the identity [62, 153]

$$\int_{\mathbb{R}} \phi(x - \mu_1; \sigma_1^2)\phi(x - \mu_2; \sigma_2^2)dx = \phi(\mu_1 - \mu_2; \sigma_1^2 + \sigma_2^2) \tag{B.2}$$

For convenience, let $\tau := \frac{1}{\sqrt{\sigma_1^2+\sigma_2^2}}$. Applying Fubini's theorem and using (B.2), we have

$$\begin{aligned}
p_\phi(\mathrm{x} \wedge \mathrm{y}) &= \int_{\mathbb{R}} \int_a^b \phi(x - y; \sigma_1^2)\, dy \int_c^d \phi(x - z; \sigma_2^2)\, dz\, dx \\
&= \int_c^d \int_a^b \phi(y - z; \tau^{-2})\, dy\, dz \\
&= \int_c^d \int_a^b \Phi'(\tau(y - z))\tau\, dy\, dz \\
&= \int_c^d \Phi(\tau(b - z)) - \Phi(\tau(a - z))\, dz \\
&= \frac{-1}{\tau}(m_\Phi(\tau(b - d)) - m_\Phi(\tau(a - d)) - m_\Phi(\tau(b - c)) + m_\Phi(\tau(a - c)))
\end{aligned}$$

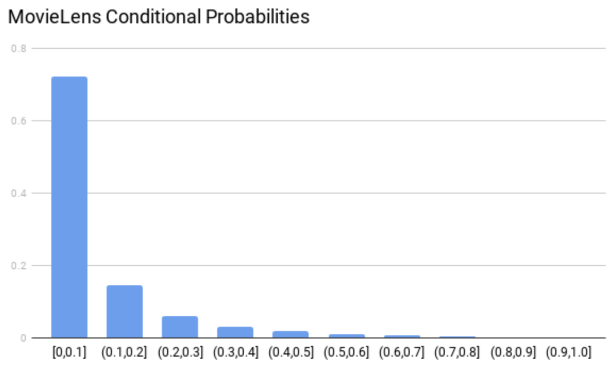


Figure B.1: Distribution of probabilities in MovieLens Dataset.

and therefore, with $\sigma = \tau^{-1}$,

$$p_\phi(\mathrm{x} \wedge \mathrm{y}) = \sigma\left(m_\Phi(\tfrac{b-c}{\sigma}) + m_\Phi(\tfrac{a-d}{\sigma}) - m_\Phi(\tfrac{b-d}{\sigma}) - m_\Phi(\tfrac{a-c}{\sigma})\right)$$

as desired.

## A.2 MovieLens Pseudosparsity

The MovieLens dataset, while not truly sparse, has a large proportion of small probabilities which make it especially suitable for optimization by the smoothed model. The rough distribution of probabilities, in buckets of width $0.1$, is shown in Figure B.1.

## A.3 MovieLens Initialization Sensitivity

We perform an additional set of experiments to determine the robustness of the smoothed box model to initialization. While the model is normally initialized randomly so that each box is a product of intervals that almost always overlaps with the other boxes, we would like to determine the models robustness to disjoint boxes in a principled way. While we can control initialization, we cannot always control the intermediate results of optimization, which may drive boxes to be disjoint, a condition from which the original, hard-edged box model may have difficulty recovering. So, parametrizing the initial distribution of boxes with a minimum coordinate and a positive width, we adjust the width parameter so that approximately 0%, 20%, 50%, and 100% of boxes are disjoint at initialization before learning on the MovieLens dataset as usual. These results are presented in table B.1. The smoothed model does not seem to suffer at all from disjoint initialization, while the performance of the original box model degrades significantly. From this we can speculate that part of the strength of the smoothed box model is its ability to smoothly optimize in the disjoint regime.

| Approx. % Disjoint | KL | | Pearson | | Spearman | |
|---|---|---|---|---|---|---|
| | Box | Smooth | Box | Smooth | Box | Smooth |
| 0% | 0.0147 | 0.0138 | 0.8775 | 0.8985 | 0.8768 | 0.8977 |
| 20% | 0.0172 | 0.0141 | 0.8668 | 0.8917 | 0.8608 | 0.8898 |
| 50% | 0.0182 | 0.0141 | 0.8613 | 0.8908 | 0.8551 | 0.8910 |
| 100% | 0.0346 | 0.0142 | 0.8401 | 0.8921 | 0.8167 | 0.8947 |

Table B.1: Performance of the original box model and smoothed box model on MovieLens, as a function of different degrees of disjointness upon initialization.

## A.4 Model Parameters

We give a brief overview of our methodology and hyperparameter selection methods for each experiment.

### A.4.1 WordNet Parameters

For the WordNet experiments, the model is evaluated every epoch on the development set for a large fixed number of epochs, and the best development model is used to score the test set. Baseline models are trained using the parameters of [151], with the smoothed model using hyperparameters determined on the development set.

### A.4.2 Imbalanced WordNet Parameters

We follow the same routine as the WordNet experiments section to select best parameters. For the 12 experiments we conducted in this section, negative examples are generated randomly based on the ratio for each batch of positive examples. We do a parameter sweep for all models then choose the best result for each model as our final result.

```
batch size: 8000
dimension: 50
edge loss weight: 3.0
unary loss weight: 7.0
learning rate: 0.001
minimum dimension delta size: 1e-6
optimizer: Adam
sigma for softplus: 1.0
box min initialization: uniformly from 1e-4 to 1e-2
box delta initialization: uniformly from 0.9 to 0.99
global regularization method: l1
global regularization strength: 0.1
max training steps: 100,000
```

### A.4.3 Flickr Parameters

The experimental setup uses the same architecture as boxlattice and [73], a single-layer LSTM that reads captions and produces a box embedding parameterized by *min* and *delta*. Embeddings are produced by feedforward networks on the output of the LSTM. The model is trained for a large fixed number of epochs, and tested on the development data at each epoch. The best development model is used to report test set score. Hyperparameters were determined on the development set.

```
batch size: 512
dropout: 0.5
dimension: 300
hidden layer size: 512
unary loss weight: 1.0
edge loss weight: 9.0
learning rate: 0.0001
optimizer: Adam
min feed forward network weight initialization: xavier_W
min feed forward network bias initialization: xavier_b
delta feed forward network weight initialization: xavier_w
delta feed forward network bias initialization: uniform from -15 to -1
min parameterization: relu
delta parameterization: softplus
epoch: 10
```

### A.4.4 MovieLens Parameters

For all MovieLens experiments, the model is evaluated every 50 steps on the development set, and optimization is stopped if the best development set score fails to improve after 200 steps. The best development model is used to score the test set.

Both the original and smoothed box model use the following hyperparameters (aside from the softplus temperature *sigma*), tuned to maximize performance of the baseline original box model on the development set:

```
batch size: 128
dimension: 50
learning rate: 0.001
unary loss weight: 0.001
edge loss weight: 0.999
optimizer: Adam
sigma for softplus: 1.0
box min initialization: uniformly from 1e-4 to 0.5
box delta initialization: uniformly from 0.9 to 0.999
```

# APPENDIX

# APPENDIX FOR QUESTION ANSWERING DATASET

## A.1 WordNet Similarity Function

1. Let $S$ be the set of synsets in WordNet, and let $S(x)$ be the set of synsets associated with the string $x$.

2. Let $\text{SynsetSim}(X, Y) : S \times S \to [0, 1]$ be a score for synset similarity, eg.

$$\text{SynsetSim}(X, Y) := \begin{cases} 1 & \text{if } X = Y, \\ 0 & \text{otherwise}. \end{cases}$$

3. A given string may corresponse to multiple synsets. Given two strings $x$ and $y$ we define

$$\text{SynsetsScore}(x, y) = \\ \max\{\text{SynsetSim}(S_x, S_y) : S_x \in S(x), S_y \in S(y)\}.$$

4. Some valid answer strings may not correspond to a synset at all, so we define

$$\text{SubstringScore} = \\ \max(\text{SynsetsScore}(x, y), \text{ExactMatch}(x, y))$$

5. Some answers are several words long, and therefore won't map to a synset even if some substring would. To account for this, we tokenize and strip stopwords from both the predicted and ground-truth answer strings. To compare these sets of tokens $A, B$ we let $M(A, B)$ be the set of all possible (partial) matchings between elements in $A$ and $B$, and then define

$$\text{TokensScore}(A, B) \\ = \max_{m \in M(A,B)} \frac{\sum_{(a,b)\in m} \text{SubstringScore}(a, b)}{\max(|A|, |B|)}$$

6. We repeat this process for every element in an answer cluster $C$, which is a set of strings obtained from the survey, and then set the overall score for this answer cluster to be

$$\text{WordNetScore}(x, C) = \\ \max\{\text{TokensScore}(T(x), T(y)) : y \in C\}$$

**Remark.** *Fully tokenizing the input has the potential to lose information, since some WordNet clusters are labeled with multiple words. Consider comparing "chewing gum" with "gum". The above process would assign this a score of* $0.5$*, because tokenizing yields ["chewing", "gum"], however "chewing gum" is, itself, in the same WordNet synset as "gum". The solution to this problem in general is to compare all possible* partitions *of the tokens, and define the overall* $\mathbf{PartitionsScore}$ *to be the maximum among all pairs of possible partitions for the predicted answer and the ground-truth string. We replace the* $\mathbf{TokensScore}$ *with this* $\mathbf{PartitionsScore}$ *to capture such situations.*

With a scoring method as described, it is possible for an answer to receive a positive score for multiple clusters. We take the following approach:

1. Round the scores to $\{0, 1\}$ to make a "hard" cluster decision.
2. For a given question, if some predicted answers match with multiple clusters, we choose the maximum matching with respect to the final score.

## A.2 GPT-2 Transformation rules

| Original Sentence | Transformed Sentence |
|---|---|
| Name something ... | One thing ... is |
| Tell me something ... | One thing ... is |
| Name a/an ... | One ... is |
| How can you tell ... | One way to tell ... is |
| Give me a/an ... | One ... is |

Table C.1: Transformation rules from original question sentence to GPT-2 format sentence

In order to make the question more natural for GPT-2 model to answer, we use rule in Table C.1 to re-write the questions.

## A.3 Criteria for test question acceptance

When creating new questions using the perturbation method described in § 2.2, we scored each question with the following criteria in mind:

- Most people are expected to be able to answer.
- The answer set category is relatively small; less than eight big categories of different answers.
- The question is hard for systems relying on co-occurrence patterns to answer, e.g., BERT

- The answers to the question are not too culturally dependent (e.g., we want to avoid questions such as *Name a dish made with ground meat*).
- Not accidentally re-creating a well-explored question: We then searched all Family Feud data to ensure that no questions were being re-created, and searched online to make sure no obvious lists of answers can be found via search with Google. E.g., if we create a question and the top search for that question is a list of answers to that question, regardless of origin, we remove the question.

## A.4 Criteria for stereotypical bias issues

We define a relatively strict measure for stereotypical bias, primarily to avoid having overly problematic examples; we expect that more nuanced issues of stereotypes are common in the data, but are not as easy to measure with an all-or-nothing measure. We rule out questions if they match any of the following:

1. Attaining the right answer requires stereotypes regarding what activities are affiliated with each gender (e.g., that only girls play with dolls)
2. Questions that measure activities a particular gender would be proud or embarrassed to do.
3. We could not find any questions addressing race, sexual orientation, religion, or national origin, but these were searched for and would have also been removed if found.

Types of potentially biased questions which we could not consistently remove from all the training data, but which we note to be worthy of consideration, are:

1. Questions with heteronormative assumptions (questions about what women like, romantically, in men or vice versa)
2. Questions that can be specific to Western US culture: a vast array of questions would have different distributions over answers if asked to people of specific cultures, where stereotypical foods, greetings, habits, or objects may be different.
3. Questions that reference gender, but which might have similar answer clusters if the gender was removed – e.g., *Name something your parents always want to know about the man you're dating*.

## A.5 QA model details

For the baseline results reported, we fine-tune the "Bert for QA" model of the Huggingface transformers package, v2.6.0 [161], using BERT-large-uncased [36].

Table C.2 illustrates examples of answer strings for the query "name something you do at a concert", illustrating both that such a method finds passages that are relevant to the questions, but also illustrating the kind of noise being introduced by such a distance learning approach.

Q: Name something you do at a concert:
A: *But you are always expected to* ***clap*** *for the spalla .*
A: *I'll often buy a* ***drink*** *for something to do, or check my email on my phone, or whatever, to kill time . once the band starts i 'm focused on that*

Table C.2: Examples of distant-learning positive examples used for training QA baseline

## A.6 GPT-2 model details

For the baseline results reported, we fine-tune GPT-2 Large model using the scrapped training data. The parameter for the best performing model is as follows: batch_size:1, training epoch: 1, gradient accumulation step: 8. The other parameters are the default value in the hugging face implementation. In generation phrase, the temperature is 0.69, top_p is 0.9, and other parameter values are using the default values. All parameters are tuned using dev data, and searched via greed search. The code will be publicly available upon publication.

| Metrics % | | | Single-Human Ranking |
|---|---|---|---|
| **Exact Match** | **Max Answers** | 1 | 40.5 |
| | | 3 | 39.4 |
| | | 5 | 41.0 |
| | | 10 | 45.6 |
| | **Max Incorrect** | 1 | 23.9 |
| | | 3 | 36.0 |
| | | 5 | 40.5 |
| **WordNet Similarity** | **Max Answers** | 1 | 45.2 |
| | | 3 | 47.8 |
| | | 5 | 50.7 |
| | | 10 | 55.3 |
| | **Max Incorrect** | 1 | 29.2 |
| | | 3 | 44.6 |
| | | 5 | 50.6 |
| **RoBERTa Similarity** | **Max Answers** | 1 | 59.0 |
| | | 3 | 64.0 |
| | | 5 | 66.2 |
| | | 10 | 71.7 |
| | **Max Incorrect** | 1 | 59.0 |
| | | 3 | 64.0 |
| | | 5 | 66.2 |

Table C.4: Results for the “single human” ranking scores, replaced by a human evaluation closer to actual task

## A.7 Alternative Human Performance Answers

| | | | | |
|---|---|---|---|---|
| Prompt | *Name something around the house that's often replaced.* | | | |
| Single-human ranking | food | toilet paper | paper towels | garbage bags |
| Prompt | *Name something a monk probably would not own.* | | | |
| Single-human ranking | a fancy car | a fancy house | too much food | a bank account |
| largest cluster | cluster 2 | cluster 3 | smaller clusters | |

Table C.3: Top three responses from human ranking evaluation for the same data

The human performance numbers reported in § 4.3 were collected to be maximally similar to the proposed task: like both the training data and the crowdsourced evaluation data, they were generated by asking many humans for a single best answer. We also collected sets of answers from a small set of in-person annotators using a slightly different questioning paradigm, providing a prompt and asking a single annotator to provide eight different answers to that question. In practice, we found that this shift in evaluation this could penalize human performance. One primary issue with this was that the human annotator asked for all answers to the same question would generally only provide a single answer string corresponding to the top answer clusters. This means that even if the human matched the correct answer, they would miss that answer cluster entirely if they provided a novel string for that answer cluster. Annotators also found it be to be quite difficult to provide many answers for the same prompt and would go far afield with later answers, making such answers differ from the distribution of answers in the train and evaluation set. To avoid confusion using these noticeably different human performance scores, we shifted reporting to a set of data that is closer to the actual task evaluation but report those ranking scores here for transparency. One can see from Table C.3 and C.4 that such human answers look good, but that the actual scores are dramatically lower than what is seen when humans are evaluated on the same task as the evaluation set, and only barely outperforms a fine-tuned GPT-2 system.

## A.1 Appendix Structure

We begin by quantifying the scaling behavior of the model to predict how performance changes with larger model sizes (Appendix A.3). We then plot the relationship between cross-entropy and answer length for each of the four datasets (Appendix A.4). After that, we describe experiments that use knowledge base triplets as a form of in-context learning (Appendix A.2). Lastly, in Appendix A.5, we provide qualitative examples that show which examples: (i) all model sizes get right, (ii) all model sizes get wrong, and (iii) only the larger models get right.

## A.2 Commonsense Knowledge Bases

Given the implicit nature of commonsense knowledge, a language model's pretraining corpora might not contain all of the supporting evidence that is required to answer commonsense understanding questions — a phenomenon widely known as the reporting bias problem [48]. Thus, prior work has proposed to use external knowledge bases for improving the zero-shot performance of LMs on commonsense benchmarks [20, 7]. These approaches are particularly interesting, as the knowledge base augmentation only happens at test time, rendering this approach compatible with *any* pretrained generative LM. While prior work has shown the effectiveness of this approach over the zero-shot baseline that lacks access to commonsense knowledge bases (CSKBs), we find that the performance of the baseline model is highly sensitive to certain evaluation design choices (§5.5). A natural question, therefore, is the following: If we carefully optimize the evaluation design choices of the baseline model, would we *still* observe similar improvements through CSKB augmentation?

**A.2.0.0.1 Setup.** To answer this, we replicate prior work by adding commonsense knowledge base entries at test time; such knowledge base triplets can potentially provide the relevant implicit commonsense knowledge that makes the correct answer more likely than the rest. To ensure the generality of our findings, we apply this approach to multiple model sizes that we explored in §5.3.3. Here we consider the pre-extracted knowledge base triplets that are made publicly available by self$_{t}alk_{2}020.Weuseasimilarscorefunctionasself_{t}alk_{2}020, w$
$\in \boldsymbol{Y}(\mathrm{x})$, we choose the knowledge base triplet that yields the highest score:[1]

$$s_{kg}(\mathrm{y}|\mathrm{x}) \triangleq \sum_{\mathrm{t}\in \boldsymbol{T}} s(\mathrm{y};\mathrm{t}|\mathrm{x}) \approx max_{\mathrm{t}\in \boldsymbol{T}}\, s(\mathrm{y};\mathrm{t}|\mathrm{x}),$$

| | ZS | w/t Comet | w/t Atomic | w/t CN |
|---|---|---|---|---|
| **44M** | 42.3 | **42.9** | 42.3 | 40.6 |
| **117M** | 43.6 | **44.0** | 43.6 | 42.2 |
| **400M** | 46.3 | **46.8** | 44.7 | 44.1 |
| **1.3B** | **47.0** | 46.8 | 46.4 | 44.7 |
| **7B** | 48.5 | **48.6** | 47.5 | 46.1 |

| | ZS | w/t Comet | Self-Talk |
|---|---|---|---|
| **GPT2** | 41.1[2] | 47.5 | 46.2 |

Table 5: Zero-shot performance on Social IQa when using different knowledge bases. GPT2 results are taken from self$_{t}alk_{2}020.ZS : zero - shotperformance; CN : ConceptNet.$

where $s(\mathrm{y};\mathrm{t}|\mathrm{x})$ denotes the cross-entropy of the concatenated answer choice $\mathrm{y}$ and the extracted knowledge base triplet $\mathrm{t}$, conditional on the question/context $\mathrm{x}$. Here $T$ denotes the set of all extracted commonsense knowledge triplets, which are generated from Comet Bosselut2019COMETCT. One key difference is that we score the answer and knowledge base triplet conditional on the question, whereas self$_{t}alk_{2}020scoredtheconcatenationofquestion, an$

In Table 5, we summarize our results on Social IQa, which has the highest gap between the zero-shot and SOTA performance (Fig. 5.2). We compare our results with those of self$_{t}alk_{2}020, whousedGPT2asthebasemodel.OurresultsinTable\ 5provideaninterest$
$Ourbaselinezero-shotmodelwith1.3Bparametersachievesanaccuracyof47.0\%onS$
$---whichachieves41.1\%---despitethefactthatGPT2hasmoreparameters(1.5Bvs$
$shotmodel---whichdoesnotbenefitfromanycommonsenseknowledgebasetriplet$
$---nearlymatchestheperformanceofGPT2augmentedwithCometBosselut2019C$
$shot1.3Bmodelvs47.5\%forGPT2augmentedwithCOMET; Table\ 5), andalsooutper$
$talk.Nevertheless, wefindthataddingknowledgebasetripletsfailstoyieldsubstantia$
$shotbaseline.$

We remark on two significant aspects of our findings. First, it is important to compare proposed improvements against strong, well-tuned baselines henderson$_{d}eeprl, melis_{l}stm_{2}018, whichca$
$Whereasself_{t}alk_{2}020scoredtheGPT2modelontheconcatenationofbothquestionand$
$entropyoftheanswergiventhequestion.Second, certainimprovementsthatareobser$

[1]We experimented with other score functions, such as appending the extracted knowledge base triplets to the question instead of the answer, although this approach does not yield better results than the one proposed by self$_{t}alk_{2}020.$

## A.3 Scaling Behavior

When we estimate the performance needed to reach human-level performance, we fit a linear model to estimate accuracy from $\log(\text{params})$. We derive the human performance from each respective paper and/or leaderboard. For HellaSwag and PIQA, human-level performance is at 95%. For WinoGrande, it is at 94% and for Social IQa it is at 84%. On HellaSwag, we predict that 1.4T parameters are needed to achieve human-level performance; on PIQA we predict 102T parameters; on WinoGrande we predict over 2000 Trillion parameters. Social IQa scales particularly poorly, and we estimate over $10^{18}$ parameters being needed.

## A.4 Cross-entropy vs answer length for all datasets

figures/token_bias_piqa.png

(a) Answer length vs cross-entropy (average log probability across tokens) for PIQA.

figures/token_bias_siqa.png

(b) Answer length vs cross-entropy (average log probability across tokens) for SocialIQA.

figures/token_bias_hellaswag.png

(a) Answer length vs cross-entropy (average log probability across tokens) for HellaSWAG.

figures/token_bias_wino.png

(b) Answer length vs cross-entropy (average log probability across tokens) for Winogrande.

## A.5 Examples

### A.5.1 Social IQa

**All Models Incorrect** [breaklines] 'context': "Tracy didn't go home that evening and resisted Riley's attacks.", 'question': 'What does Tracy need to do before this?', 'answerA': 'make a new plan', 'answerB': 'Go home and see Riley', 'answerC': 'Find somewhere to go', 'correct': 'C' [breaklines] 'context': 'Aubrey kept the baby up at night to watch for a concussion.', 'question': 'What will happen to Aubrey?', 'answerA': "The baby fell asleep despite Aubrey's best effort", 'answerB': 'gets so sleepy but stays awake anyway', 'answerC': 'and the baby both fell asleep late in the night', 'correct': 'B'

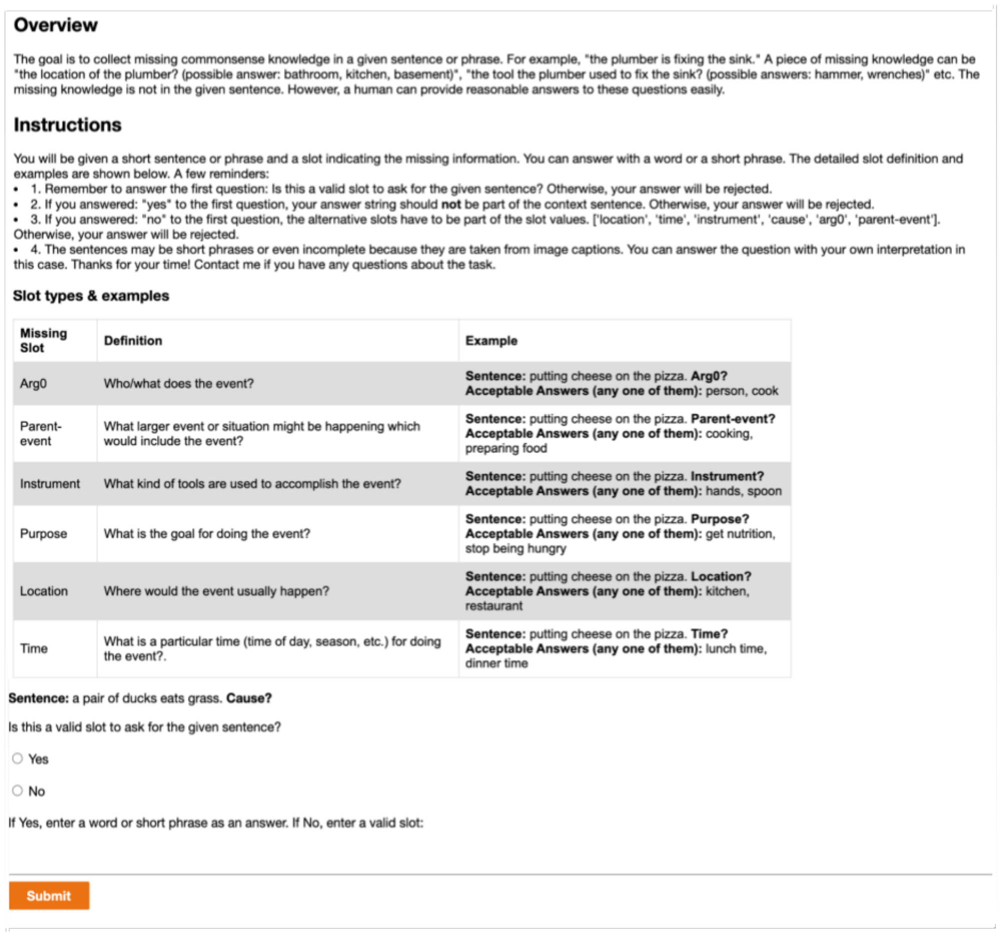
**All Models Correct** [breaklines] 'context': 'Kendall opened their mouth to speak and what came out shocked everyone.', 'question': 'How would you describe Kendall?', 'answerA': 'a very quiet person', 'answerB': 'a very passive person', 'answerC': 'a very aggressive and talkative person', 'correct': 'C' [breaklines] 'context': 'Sydney went to our family farm, taking the trash with her, and set it on fire on the ground.', 'question': 'How would Sydney feel afterwards?', 'answerA': 'feeling strong', 'answerB': 'burning down', 'answerC': 'upset because the fire has gotten out of control', 'correct': 'C' [breaklines] 'context': 'Robin always gets pizza on the way home from work for her family on Fridays.', 'question': 'What will Robin want to do next?', 'answerA': 'pick up the pizza', 'answerB': 'complain to the others', 'answerC': 'finish work', 'correct': 'A'

**Larger Models Correct** The 1.4B, 7.1B, and 280B model all got the following correct: [breaklines] 'context': 'Alex paid extra money to get more secret details about the game strategy.', 'question': 'What will Alex want to do next?', 'answerA': 'play the game more', 'answerB': 'ignore the advice', 'answerC': 'stop playing the video game', 'correct': 'A' The 417M, 7.1B, and 280B model all got the following correct: [breaklines] 'context': 'Kai and Skylar were good friends. Kai had finally worked up the courage to ask Skylar on a date. They gave Skylar a meaningful gift to test the waters.', 'question': 'What will Kai want to do next?', 'answerA': 'say thank you for the gift', 'answerB': 'Find out whether Skylar reciprocates the feelings', 'answerC': "Tell Skylar they'd like to just be friends", 'correct': 'B'

### A.5.2 WinoGrande

**All Models Incorrect** [breaklines] 'label': 1, 'option1': 'Tanya', 'option2': 'Sarah', 'sentence': 'Tanya was unrecognizable after Sarah was done beating them, so $_{e}ndedupgoingtojail.'$ [breaklines] 'label': 1, 'option1': 'Logan', 'option2': 'Justin', 'sentence': 'After Logan pitched a ball that got clobbered for a home run by Justin in a baseball game, $_{f}eltexultant.'$

**All Models Correct** [breaklines] 'label': 1, 'option1': 'sausage', 'option2': 'ball', 'sentence':b'When the dog behaves I like to give him a sausage otherwise I give him a ball. I gave him the $_{s}incehewasbad.'$ [breaklines] 'label': 1, 'option1': 'Kayla', 'option2': 'Natalie', 'sentence': 'Kayla always wears sunscreen outdoors but Natalie doesn't because $_{i}sn'tconcernedaboutgettingneckwrinkles.'$



**Overview**

The goal is to collect missing commonsense knowledge in a given sentence or phrase. For example, "the plumber is fixing the sink." A piece of missing knowledge can be "the location of the plumber? (possible answer: bathroom, kitchen, basement)", "the tool the plumber used to fix the sink? (possible answers: hammer, wrenches)" etc. The missing knowledge is not in the given sentence. However, a human can provide reasonable answers to these questions easily.

**Instructions**

You will be given a short sentence or phrase and a slot indicating the missing information. You can answer with a word or a short phrase. The detailed slot definition and examples are shown below. A few reminders:

- 1. Remember to answer the first question: Is this a valid slot to ask for the given sentence? Otherwise, your answer will be rejected.
- 2. If you answered: "yes" to the first question, your answer string should **not** be part of the context sentence. Otherwise, your answer will be rejected.
- 3. If you answered: "no" to the first question, the alternative slots have to be part of the slot values. ['location', 'time', 'instrument', 'cause', 'arg0', 'parent-event']. Otherwise, your answer will be rejected.
- 4. The sentences may be short phrases or even incomplete because they are taken from image captions. You can answer the question with your own interpretation in this case. Thanks for your time! Contact me if you have any questions about the task.

**Slot types & examples**

| Missing Slot | Definition | Example |
|---|---|---|
| Arg0 | Who/what does the event? | **Sentence:** putting cheese on the pizza. **Arg0?** **Acceptable Answers (any one of them):** person, cook |
| Parent-event | What larger event or situation might be happening which would include the event? | **Sentence:** putting cheese on the pizza. **Parent-event?** **Acceptable Answers (any one of them):** cooking, preparing food |
| Instrument | What kind of tools are used to accomplish the event? | **Sentence:** putting cheese on the pizza. **Instrument?** **Acceptable Answers (any one of them):** hands, spoon |
| Purpose | What is the goal for doing the event? | **Sentence:** putting cheese on the pizza. **Purpose?** **Acceptable Answers (any one of them):** get nutrition, stop being hungry |
| Location | Where would the event usually happen? | **Sentence:** putting cheese on the pizza. **Location?** **Acceptable Answers (any one of them):** kitchen, restaurant |
| Time | What is a particular time (time of day, season, etc.) for doing the event?. | **Sentence:** putting cheese on the pizza. **Time?** **Acceptable Answers (any one of them):** lunch time, dinner time |

**Sentence:** a pair of ducks eats grass. **Cause?**

Is this a valid slot to ask for the given sentence?

○ Yes

○ No

If Yes, enter a word or short phrase as an answer. If No, enter a valid slot:

**Only Large Models Correct** Models 400M and larger got the following correct: [breaklines] 'label': 0, 'option1': 'Nick', 'option2': 'Ryan', 'sentence': 'Nick did not like sauces made from tomato, only creamy sauces. Ryan knew this so he only made white sauce when $_{c}ameover.'$ Models 1.4B and larger got the following correct: [breaklines] 'label': 0, 'option1': 'Adam', 'option2': 'Jason', 'sentence': 'Adam loved dogs but Jason was afraid of them, so only $_{p}ettedthepoodle.'$

## A.6 Appendix for Commonsense Frame Completion

This shows the dataset collection Amazon MTurk screen shot.