Penguins Don't Fly: Reasoning about Generics through Instantiations and Exceptions

Anonymous ACL submission

Abstract

Generic statements (e.g.,Birds can fly) express generalizations about the world. However, generics are not universally true – while sparrows and penguins are both birds, penguins can't fly. Understanding cases when a generic statement is true or false is crucial for machine reasoning. In this work, we present a novel framework to generate pragmatically relevant true and false instances of a generic. We use pre-trained language models, constraining the generation based on our computational framework, and produce $\sim 20k$ EXEMPLARS for $\sim$650 generics. Our system outperforms few-shot generation from GPT-3 (by 12.5 precision points) and our analysis highlights the importance of constrained decoding for this task and the implications of generics EXEMPLARS for non-monotonic reasoning and NLI.

1 Introduction

Generics are statements that express generalizations about the world (see Figure 1). These statements are accepted as true even if real-word prevalence of the asserted phenomenon is unspecified (e.g., baby birds can't fly). They have been extensively studied in semantics, philosophy, and psychology for their puzzling properties such as generalizing about an uncommon property (e.g., "Mosquitoes carry malaria.[1]" Krifka 1987; Cohen 1996), and for their connections to non-monotonic reasoning (Elio and Pelletier, 1996). Understanding generics and generating instances when they do and do not hold is crucial for replicating the nuances of human reasoning, particularly the efficient use of generalizations (Mercier and Sperber, 2017).

A generic asserts a relationship between a *concept* ("Birds") and a *property* ("fly") without a quantifier[2] that signals prevalence of the property with respect to the concept (Figure 1). Since this is a generalization without quantification, it allows for INSTANTIATIONS—cases where specified relationship holds (e.g., "Sparrows can fly") and EXCEPTIONS—cases where it does not hold (e.g., "Penguins cannot fly"). Identifying EXCEPTIONS is particularly challenging because an EXCEPTION must both violate the relationship asserted by the generic and be *pragmatically relevant* ("Cats can't fly" is not a valid EXCEPTION in Figure 1).

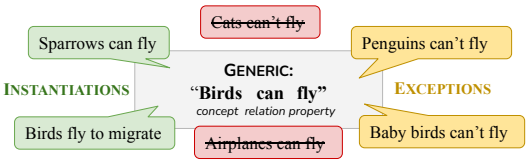

Figure 1: We study understanding and reasoning with generics by generating valid applications (i.e., INSTANTIATIONS) of and EXCEPTIONS to the generic, ~~exclud-ing~~ pragmatically irrelevant instances.

In this work, we present a novel computational framework for constructing and generating **EXEMPLARS** (INSTANTIATIONS *and* EXCEPTIONS) for a generic that incorporates various theories from semantics. Bringing together categories of generics (Leslie, 2007, 2008) and exceptions (Greenberg, 2007) (see categorization in Table 1), we use generics from (Anonymous, 2022) and automatically generate 8429 EXCEPTIONS and 11771 INSTANTIATIONS. We analyze our output using human evaluation and ablation studies.

Recent advances in language modeling have been extremely successful at generating text for a range of tasks in a few-shot manner (Brown et al., 2020). However, such generation is both expensive and does not provide the degree of control necessary for this task (i.e., the output must have a specific semantic relationship to the input). Therefore, in this work we present a novel constrained gen-

[1] Approximately 7-9% of the females of the species *Anopheles* (one among 3500 species) transmit malaria (CDC).

[2] We specifically focus on statements without quantification. Statements with explicit quantification (e.g., "*Most* birds can fly" or "Birds can *usually* fly") do not allow exceptions and are excluded from this study.

Category	Generic (G)	INSTANTIATION	EXCEPTION
(a) quasi-def	"Quakes produce seismic waves"	"Quakes produce pressure waves"	"Quakes produce shaky ground"
	$K(x) \wedge r(x,y) \implies P(y)$	$K(x) \wedge r(x,y) \wedge P(y)$	$K(x) \wedge r(x,y) \wedge \sim P(y)$
(b) principled	"Birds can fly"	"Owls can fly"	"Penguins can't fly"
	"Sharks attack swimmers"	"Threatened sharks attack swimmers"	"Sharks don't attack swimmers in the shallows"
	$K(x) \wedge P(y) \implies r(x,y)$	$K(x) \wedge r(x,y) \wedge P(y)$	$K(x) \wedge \neg r(x,y) \wedge P(y)$
(c) characterizing	"Cars have radios"	"2014 Prius model C has a radio "	"Cars have CD Players"
	L_G is ambiguous	$K(x) \wedge r(x,y) \wedge P(y)$	$K(x) \wedge r(x,y) \wedge \sim P(y)$
			"Newer cars don't have radios"
			$K(x) \wedge \neg r(x,y) \wedge P(y)$

Table 1: Generic types with their EXEMPLARS. The logical forms for the generic (L_G) and its INSTANTIATION and EXCEPTION are also below the examples. K is the concept, P the property, $\sim P$ the semantic negation (§3.3).

eration approach using the NeuroLogic Decoding algorithm (Lu et al., 2021) with output constraints derived from semantic theories. Our system both outperforms (by 12.5 precision points on average) and is more controllable than few-shot generation.

We note that although generics admit EXCEPTIONS, increasing research in psychology and philosophy has shown that humans, from children to adults, often accept generics as the default for a concept (Khemlani et al., 2009, 2012; Leslie et al., 2011) even when the inference is not deductively valid. Such *default inheritance reasoning* (Lifschitz, 1989) is a specific form of non-monotonic reasoning (i.e., adding new premises can cause the withdrawal of previous conclusions without altering existing premises) that underpins human gut-reactions (i.e., generalizations) to new information and situations. Such generalization ability is fundamental to human reasoning (Mercier and Sperber, 2017). Thus, recognizing and automatically producing the cases when and when not to generalize is critical for flexible machine reasoning and decision making (Reiter, 1978; Ginsberg, 1987a).

Our contributions are as follows : **(1)** we present a **novel framework** grounded in linguistic theory for representing generics and EXEMPLARS, **(2)** we present the first, to the best of our knowledge, **method** to automatically generate generic EXEMPLARS and show it outperforms few-shot generation, and **(3)** we present analysis showing the **importance of controllability** for this task and we use our generated data to **highlight the insufficiency of current NLI** methods for representing default inheritance reasoning. Our system and data will be made publicly available.

2 Related Work

Generics have been studied extensively in semantics, philosophy, and psychology to develop a single logical form for all generics (Lewis and Keenan, 1975; Carlson, 1977, 1989; Krifka, 1987) or a probabilistic definition (Cohen, 1996, 1999, 2004; Kochari et al., 2020), categorize generics (Leslie, 2007, 2008), and analyze specific types (Prasada and Dillingham, 2006, 2009; Haward et al., 2018; Mari et al., 2012; Krifka et al., 2012). Mechanisms to tolerate EXCEPTIONS have also been proposed (Kadmon and Landman, 1993; Greenberg, 2007; Lazaridou-Chatzigoga and Stockall, 2013) but these are primarily theoretical and use carefully chosen examples. In contrast, our work combines existing EXCEPTION tolerance mechanisms with generic categorization and proposes a novel, large-scale, computational framework for EXEMPLARS.

While large-scale KBs capture a range of commonsense knowledge (Speer et al., 2017; Sap et al., 2019; Forbes et al., 2020; Hwang et al., 2021) these resources do not distinguish between generic (e.g., "Birds can fly") and non-generic facts (e.g., "Birds usually fear cats"). Although GenericsKB (Bhakthavatsalam et al., 2020) does explicitly contain generics, no attempt is made to provide EXEMPLARS for the generics and many of the statements are specific scientific facts, rather than generalizations. In contrast, in our work we categorize a large set of machine-generated generics from (Anonymous, 2022) using crowdsourcing and automatically generate EXEMPLARS for these generics.

The application of generics to specific individuals is influenced by prototypicality (Rips, 1975; Osherson et al., 1990), with small sets of prototypical norms collected in cognitive science for a range of kinds (Devereux et al., 2014; McRae et al., 2005; Overschelde et al., 2004). However, recent work has shown that neural models have only moderate success at mimicking human prototypicality (Misra et al., 2021; Boratko et al., 2020) or producing commonsense facts without guidance (Petroni et al.,

Generic	INSTANTIATION	EXCEPTION
"A chest pain has a physical cause."	"an angina pectoris has an underlying cause" (5)	"a chest pain has an emotional or psychological origin" (1)
"Aloe is used to treat dry skin."	"aloe vera can be used to relieve the symptoms of eczema" (6)	"aloe vera plant is used to relieve pain and inflammation" (1)
"A gun are used for hunting."	"a shotgun is used for small game" (7)	"semiautomatics can be used for target practice" (2)

Table 2: Examples of generated INSTANTIATIONS and EXCEPTIONS. The template used in the prompt for generation is indicated in parentheses (see Table 3).

2019) and additionally exceptions are often not prototypical. Hence, we combine neural models with a KB of concepts, using linguistic-theory-guided decoding, to generate generics EXEMPLARS.

Reasoning with generics is closely related to non-monotonic reasoning (Ginsberg, 1987b,a); specifically default inheritance reasoning (Brewka, 1987; Hanks and McDermott, 1986; Horty and Thomason, 1988; Imielinski, 1985; Poole, 1988; Reiter, 1978, 1980). Contrary to the proposed solutions for linguistic tests on default inheritance reasoning (Lifschitz, 1989) (e.g., can a conclusion about inheritance be inferred based on provided evidence?), later works showed that the presence of generics EXEMPLARS in the evidence impacts what humans perceive as the correct answer (Elio and Pelletier, 1996; Pelletier and Elio, 2005; Pelletier, 2009). These results highlight the importance of identifying generics and analyzing how to accurately model their relationships in machine reasoning. While natural language inference (NLI), a form of deductive reasoning that has been well studied in NLP (e.g., Dagan et al. (2013); Bowman et al. (2015); Rudinger et al. (2020)), captures notions of inference, studies on non-monotonic reasoning and NLI are limited (Wang et al., 2019) and do not include default inheritance reasoning. Therefore, in this work we analyze the interactions between generics EXEMPLARS and NLI and highlight the importance of modeling this relationship in machine reasoning.

3 Framework for EXEMPLARS

We will discuss how theories on generic types and interpretations (§3.1) are combined (§3.2) to derive logical forms for generics (§3.3) and for EXEMPLARS (§3.4). From our logical forms we derive templates that are suitable for generation (§3.5).

3.1 Generics Background

Generic statements express generalizations about the world without explicit quantification. A generic statement describes a relationship (*relation*) between a *concept* and a *property* (see Figure 1). A **concept** is typically a type or kind (e.g., cat) while a **property** is typically an ability (e.g., purr) or quality (e.g., furry). As proposed by Greenberg (2007), the relationship in a generic may either be true **in-virtue-of** a second unspecified but normative property of the concept (e.g., birds can fly *in-virtue-of* having wings) or may be merely **descriptive** of a non-accidental relationship between concept and property.

3.2 Generic Type Definitions

To categorize a given generic, we unify the theories from Greenberg (2007) with five generic types proposed by Leslie (2007, 2008) (see Appendix B for detailed discussion) and formulate three categories of generics, for which we collect human annotations on a set of generics. Our three generic categories are (see examples Table 1):

(a) **Quasi-definitional**: concern properties that are assumed to be universal among a concept. They are descriptive, since the property is considered defining for the concept.

(b) **Principled**: concern properties that are prevalent among or connected to a concept in a principled way (Prasada and Dillingham, 2006, 2009; Haward et al., 2018) and generics that concern properties that are uncommon and often dangerous (Leslie, 2017).

(c) **Characterizing**: concern properties that are not deeply connected with a concept.

3.3 Logical Forms for Generics

We assert that each generic category corresponds to a specific logical form L_G (Table 1). For quasi-definitional generics, since the property is defining we assert that the property logically follows from the combination of concept and relationship. In contrast, for principled generics the focus is on

principled relationship and so we assert that concept and property together then logically imply the relationship. Finally, for characterizing generics, the logical form depends on the interpretation of the generic as either principled or descriptive. Logical forms are shown in Table 1.

Given a logical form, we define the following *satisfaction criteria*. For a concept T (e.g., cat) or property (e.g., sleep): we will say that $T(i)$ is true if i is a subtype of T (e.g., i=Tabby cat or i=Garfield, for T=cat), or T itself. Additionally, we say that $\sim T(i)$ is true if $T'(i)$ is true for some contextually relevant second type T', where T' is not T nor any of its subtypes (e.g., T'=dog for T=cat). We say a relation $r(x, y)$ is satisfied if r holds between the individuals x and y.

3.4 EXEMPLARS Logical Forms

INSTANTIATIONS An INSTANTIATION for a generic is a contextually relevant individual of the concept that possesses the desired property. Specifically, an INSTANTIATION is member of the concept for which L_G is satisfied. The same logical form applies to all cases of INSTANTIATIONS regardless of the category (see Table 1).

EXCEPTIONS Greenberg (2007) notes that EXCEPTIONS can be established by specifying a members of the concept without the generic property (e.g.,"Owls can fly" for "Birds can fly") or by positing alternative properties when the concept cannot do without the property (e.g., "Quakes produce seismic waves" for "Quakes produce shaky ground"). Therefore, an EXCEPTION is not only an instance where L_G is not satisfied but where $\neg L_G$ is *also* satisfied[3]. The logical form of the exception depends on the type of the generic (see Table 1).

3.5 Logical Forms to Templates

Based on our proposed formulae (Table 1) for EXEMPLARS and their satisfaction criteria (§3.3), we define seven templates for generation (Table 3). Each template represents an instance that satisfies the logical form of an EXEMPLAR, potentially with subtypes. Each template consists of two sets of content requirements: for the *input* and for the *completion* (i.e., the decoder output).

For INSTANTIATIONS, we define three templates with subtypes of the concept, property, or both. However, for EXCEPTIONS we subtype only the concept *or* property. This is because when the exception has two subtypes, the individual described is now no longer exceptional (i.e., they do not lack property entirely) nor contextually irrelevant (i.e., they are still a member of the concept).

[3]In L_G, for the concept/property T, $\neg T \equiv \sim T$

Output For	Template	
EXCEPTIONS: quasi-def & characterizing	`[K + REL]`^{input} `[NEG-P]`^{comp}	(1)
	`[`K_{sub}` + REL]`^{input} `[NEG-P]`^{comp}	(2)
EXCEPTIONS: principled & characterizing	`[K + NEG-REL]`^{input} `[`P_{sub}`]`^{comp}	(3)
	`[`K_{sub}` + NEG-REL]`^{input} `[P]`^{comp}	(4)
INSTANTIATIONS: all categories	`[`K_{sub}` + REL]`^{input} `[P]`^{comp}	(5)
	`[K + REL]`^{input} `[`P_{sub}`]`^{comp}	(6)
	`[`K_{sub}` + REL]`^{input} `[`P_{sub}`]`^{comp}	(7)

Table 3: Templates for generating EXEMPLARS, derived from their logical forms (§3.4). *sub* indicates a subtype, `K` the concept, `P` the property and its negation `NEG-P` (§3.3). *comp* is the completion of the input.

4 Methodology

Our system takes as input a generic G, along with its type and associated templates (§3.5) and outputs a set of generated EXEMPLARS (Figure 2). The system populates the templates according to the input generic (§4.1). Filled templates are converted into a set of prompts and constraints that control the decoding process (§4.2). The final output is filtered to remove false (§4.3) or invalid EXEMPLARS (§4.4).

4.1 Template Assembly

To populate our templates, we use a dependency parser[4] to identify text spans for the concept, relation, and property in a generic. Then, we extract subtypes for the concept and property and use these to populate the input template, via generation prompts, and the completion template, through lexical constraints.

Subtype Extraction We first extract subtypes from ConceptNet (Speer et al., 2017)[5]. However, many natural and valid subtypes may be missing from ConceptNet (e.g., modifier phrases attached to a concept: "*young* Arctic fox"). Therefore, to increase the coverage and diversity of our subtypes we also use GPT-3[6] (Brown et al., 2020) by categorizing the concepts and using category-specific prompt to obtain subtypes (see Appendix E).

[4]https://spacy.io/

[5]Relations: `IsA`, `InstanceOf`, `Synonym`

[6]We only use GPT-3 for subtypes of the concept, since by increasing the diversity in the prompt we may encourage diversity in the generated properties.

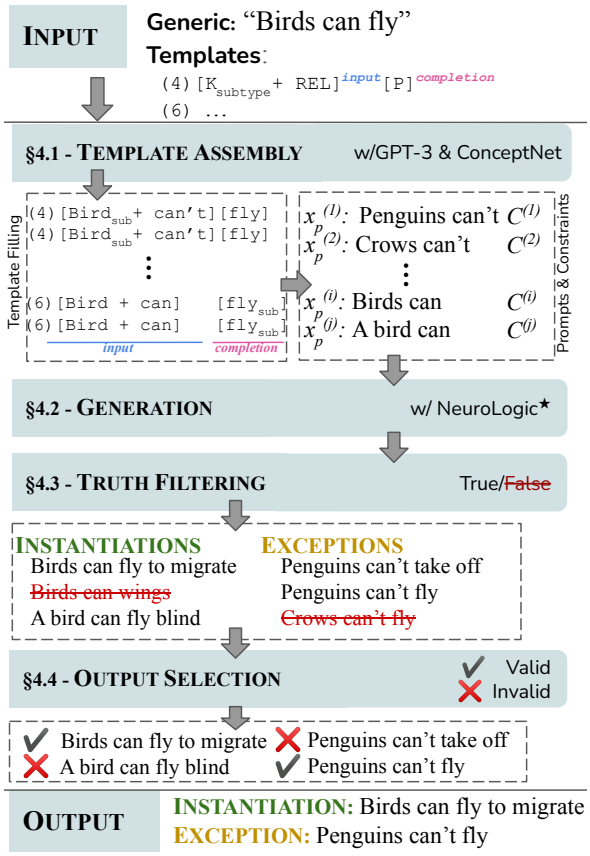

Figure 2: Overview of our method for an input generic.

Input Template Assembly We populate the input template by constructing generation prompts. Following the template, each prompt consists of either the concept (or a subtype) and the relationship (or its negation) (see Table 3). To each prompt, we additionally prepend the generic itself and a connective (e.g., "however"). We rank the prompts by perplexity and take the top k_p prompts across subtypes to use for generation.

Completion Template Assembly Following the templates, we want to constrain the generation output to describe the property (or a subtype) or its negation (see Table 3). We construct a set of completion constraints (e.g., "include 'fly' in the output") to specify that the output should include the property and exclude the concept itself. We use lexical items including subtypes, synonyms, and morphological derivations to define constraints.

4.2 Generation

In order to generate output that follows specific semantic requirements with respect to the input without requiring training, we use the NeuroLogic* (Lu et al., 2021) decoding algorithm. NeuroLogic* is an unsupervised decoding algorithm that takes as input a prompt x_p and set of lexical constraints $\mathcal{C}$ and produces a completion of the prompt $\hat{y}$ which has high likelihood given the prompt *and* high satisfaction of the constraints (estimated throughout the decoding). A lexical constraint consists of a set of n-grams $w = (w_i^1, \ldots, w_i^m)$ and is satisfied when at least one $w_i \in w$ is in $\hat{y}$ (inclusion constraints) or is not in $\hat{y}$ (exclusion constraints).

By using the input prompts (as x_p) and completion constraints (as $\mathcal{C}$) derived from our templates (§4.1), we can control the output content, syntactic form, *and* pragmatic relevance. Additionally, since we cannot define the set of relevant potential candidates for a property's negation (§3.3), decoding constraints must be used to generate EXCEPTIONS.

Output Ranking We rank the outputs from NeuroLogic* per template and prompt and we take the top k_r outputs as potential EXEMPLARS. We rank the outputs by perplexity (for fluency) and by the probability of a specific NLI label (for relevance) and average the two ranks. For NLI labels we use contradiction for EXCEPTIONS and entailment for INSTANTIATIONS. We hypothesize that a good EXCEPTION aligns with NLI's contradiction, as does a good INSTANTIATION with entailment (see Figure 2). While this alignment is useful for ranking, the relationship between the EXEMPLARS and NLI labels is not straightforward as we will discuss (§6.3). Note that ranking only by perplexity could limit the diversity of the output set, since small variations (e.g., word order changes) may result in multiple similar outputs ranked highly, and could also result in non-salient outputs (e.g., output "Hats can be made of many materials" for the generic "Hats are made of wool") ranked highly.

4.3 Filtering For Truthfulness

Since pre-trained language models have a tendency to hallucinate facts (Rohrbach et al., 2018), we apply a truth filtering step to the ranked outputs from our generation. To do this, we train a discriminator to predict whether an output is true and viable or not viable (e.g., false or too vague) using human annotated examples (see Appendix C for details). The generations predicted not viable by the trained discriminator are removed from the dataset.

4.4 Output Selection

After removing the non-viable generations, our final task is to select the examples that are valid EXEMPLARS. To do this, we collect gold labels

from humans for whether an EXEMPLARS is valid. We use separate annotation tasks for the generations from the INSTANTIATION and EXCEPTION templates. We use our human annotations to train two validity discriminators: one for EXCEPTIONS, one for INSTANTIATIONS. The trained validity discriminators are used to rank and select the best generations for each generic as our output.

5 Experiment Details

Full hyperparameters are given in Appendix D.

5.1 Data Source

We use a subset of the GenGen dataset (Anonymous, 2022), a set of 30K generics built upon common everyday concepts (e.g., "hammers") and relations (e.g., "used for") sourced from resources such as GenericsKB (Bhakthavatsalam et al., 2020) and ConceptNet (Speer et al., 2017). The dataset includes a diverse variety of concepts, e.g., general knowledge ("Dogs bark"), locative generics ("In a hotel, you will find a bed"), and comparative generics ("Cars are faster than people"). For this study, we use 653 generics from GenGen, excluding human referents as the concept (e.g., nationalities, professions) due to concerns of social biases.

5.2 Annotations

All annotations are done using Amazon Mechanical Turk (paid at $15/hour) and processed using MACE (Hovy et al., 2013) to filter annotators and determine the most likely label.

For **generic type** (§3.2), we conduct two annotation passes to partition the generics into three groups (the three groups in Table 1). Crowdworkers annotate *all* 653 *generics* with a moderate Fleiss' κ of 0.41 and 0.58 for the two passes. Our categorization results in 296 quasi-definitional, 125 principled, and 232 characterizing generics.

For the **truthfulness filter** (§4.3), we annotate a set of 7665 *system generations* from 150 generics with three annotators for each example. The Fleiss' κ (Fleiss, 1971) using the binned labels is 0.53 (for un-binned it is 0.45) indicating moderate inter-annotator agreement on this task.

To obtain **EXEMPLARS gold labels** (§4.4), annotators are whether a system-generated EXEMPLAR contradicts (e.g., for an EXCEPTION) an interpretation of the generic (see Appendix C for full details). The inter-annotator agreement, Fleiss' κ is 0.40 for the INSTANTIATION task and 0.45 for the EXCEPTION task. Thus, while these tasks are more challenging than determining truthfulness, annotators achieve reasonable agreement.

# Gens	G3-Sub	CN-Sub	All
Original	42272	10496	52768
+True	22865	5452	28317
TOTAL Valid	**17204**	**2996**	**20200**
EXCEPTION	6221	2208	8429
INSTANTIATION	10983	788	11771

Table 4: Statistics of the generated dataset, with GPT-3 (G3) and ConcepNet (CN) subtypes (sub) used.

5.3 Discriminators

For all discriminators, we fine-tune RoBERTa (Liu et al., 2019). All labeled data is split 80/10/10 into train/dev/test such that all generations for a particular generic are in the same data partition.

For our truth discriminator (§4.3), the accuracy on the test set is 75.2. For each of our two validity discriminators (§4.4) (i.e., to determine whether an INSTANTIATION or exception is valid), our data consists $\sim 1k$ randomly sampled generations across $\sim$300 generics. The accuracies of the trained validity discriminators on the test are 77.4 for INSTANTIATIONS and 75.0 for EXCEPTIONS.

5.4 Few-Shot Baseline

As a baseline for generation, we use GPT-3 (Brown et al., 2020) with few-shot prompting. Specifically, for each template (Table 3) we construct a few-shot prompt (Appendix E) that consists of three examples. Each example is two sentences: first the generic, second a connective (e.g., "But also") followed by an EXEMPLAR that adheres to the desired template. A fourth generic and connective is appended to the prompt and the model should then generate a completion that follows the illustrated template. Note that this setup is very similar to the prompts used in our method except our method is not provided examples and the baseline is not provided with subtypes (when appropriate). Note that our goal is not to produce the best possible generations from GPT-3 but rather to show that constrained generation from GPT-2 (i.e., NeuroLogic*) outperforms (and is cheaper and more computationally feasible) than a natural use of GPT-3.

6 Evaluation

Using our computational framework, we generate 20200 EXEMPLARS for 653 generics (Table 4). See Table 2 for example outputs. While close to half the

	EXCEPTIONS		INSTANTIATIONS	
	$P@1$	$P@5$	$P@1$	$P@5$
Few-shot	0.515	0.557	0.762	0.686
Ours	**0.615**	**0.595**	**0.909**	**0.876**

Table 5: Precision at k ($P@k$).

	(1)	(2)	(3)	(4)	(5)	(6)	(7)
#Gens	429	970	36	203	1159	58	890
Few-Shot	0.64	**0.52**	**0.56**	0.52	0.78	**0.71**	0.50
Ours	**0.69**	**0.52**	0.42	**0.59**	**0.86**	0.62	**0.86**

Table 6: Precision by template. #Gens: per template, is minimum of the models.

output generations are untrue or not salient, the majority of salient generations are valid EXEMPLARS.

To evaluate our approach, we conduct a human evaluation (§6.1), as well as an ablation study (§6.2) and analysis of the relationship to NLI (§6.3). Our results show that our approach produces a large set of high quality generations for this difficult task. They also highlight current limitations in machine reasoning and potential directions for future work.

6.1 Human Evaluation

To evaluate our model, we compute precision at k (for $k = 1$ and $k = 5$) using our human annotations (§4.4,5.2) as the gold labels (Table 5).

Our model outperforms the few-shot baseline in all cases, and by a large gap (average 12.5 points). This is especially significant for EXCEPTIONS, which are more challenging to generate than INSTANTIATIONS, and where the baseline performance is close to random. Since generics are defaults, it follows that INSTANTIATIONS should be easier to produce than EXCEPTIONS. The fact that more generated INSTANTIATIONS are true (71% versus 40%) and more true INSTANTIATIONS are accepted by the discriminator (73% versus 36%), compared to the EXCEPTIONS, supports this intuition. Hence, the large improvements by our model over the baseline are significant towards generating these difficult EXCEPTIONS.

Additionally, we examine our model performance across templates (§3.5). Specifically, we compute the fraction of generations for a template that annotators label as valid, using the same number[7] of generations for both models for a specific template (Table 6). We see that not only does our model outperform the baseline for the majority of templates, these templates constitute the majority of the generations ('#Gens' in Table 6).

Note that the performance comparison by template does not account for generations that are accepted *because* they do not adhere to the desired template. Therefore, we conduct a manual analysis of the best 40 baseline generations per template, ranked by perplexity. For EXCEPTIONS, the baseline produces on average only $2.5/40$ generations that fit the desired templates (2)-(4). Additionally, for the one EXCEPTION template, (1), where most baseline generations fit the template ($37/40$), our model still outperforms the baseline. For INSTANTIATIONS, the baseline performs slightly better (average $10/40$ fitting generations) but still poorly. From this we observe that not only is the baseline not controllable, our model outperforms the baseline where it does adhere to output requirements.

6.2 System Analysis

We first ablate the decoding algorithm by removing the constraints (i.e., using beam search) (Table 7a). Although both systems condition their outputs on the same prompts, NeuroLogic*, with linguistic-theory-guided constraints, produces over seven times as many unique generations as unconstrained decoding (i.e., beam search). Additionally, the proportion of valid generations (i.e., accepted by our discriminator) is nearly twice as many for NeuroLogic*. This illustrates the importance of incorporating linguistic-theory-based control into decoding in order to generate a large set of unique, and valid, EXEMPLARS.

Next, we vary the source of subtypes in the template-based prompts and constraints for our system, comparing GPT-3 and ConceptNet (CN) (used in our method) to Masked Language Model (Devlin et al., 2018; Taylor, 1953) (MLM) infilling (Table 7b), which has been used to study prototypicality in LMs (e.g., Boratko et al. (2020)) and thus should produce valid examples of a concept. Using GPT-3 for subtypes produces the most generations, likely by increasing the number of distinct and meaningful generation prompts. While using MLM for subtypes produces fewer generations than using GPT-3, the proportion of valid generations is comparable and hence MLM could be used as a substitute if using GPT-3 is not feasible. Additionally, CN produces the fewest generations, with the lowest proportions valid. This highlights the

[7]The models produce similar numbers of generations, except for template (5), where we obtain significantly more generations from GPT3.

	Beam		NeuroLogic*	
	#Gens	%Val	#Gens	%Val
Excep.	5119	9.7	30060	14.4
Inst.	2185	38.0	22708	51.8
ALL	7304	18.2	52768	30.5

(a) Decoding method ablation: beam search vs. NeuroLogic*.

	MLM		CN		GPT-3	
	#Gens	%Val	#Gens	%Val	#Gens	%Val
Excep.	10350	18.7	7619	12.6	22441	15.0
Inst.	4459	59.7	2877	27.4	19831	55.4
ALL	14809	31.0	10496	16.7	42272	33.9

(b) Subtype ablation: MLM, ConceptNet (CN), and GPT-3.

Table 7: Ablation results. #Gens: generations after ranking and filtering. %Val: percent accepted by the corresponding validity discriminator.

	Excep.		Inst.	
	$P@1$	$P@5$	$P@1$	$P@5$
Ours	0.615	0.595	0.909	0.876
+ NLI-neu	0.524	0.532	0.906	**0.889**
+ NLI-sim	**0.808**	**0.775**	0.862	0.860
+ NLI-neu-sim	0.620	0.532	**0.910**	0.888

Table 8: Precision at k with NLI label filtering.

insufficiency of the KB as the only source of subtypes. However, CN does provide subtypes that are suitable for template (5), which both GPT-3 and MLM are unable to do. These results show the importance of the knowledge source(s) used to control the generation of EXEMPLARS.

6.3 Generics EXEMPLARS and NLI

Generics and their EXEMPLARS are closely related to default inheritance reasoning and we observe that for EXCEPTIONS we can improve precision (by 19.3 points) by limiting generations to only those that contradict the generic (premise) (Table 8). For INSTANTIATIONS, the precision only increases slightly when we apply analogous filtering (with NLI entailment and neutral). However, as mentioned previously, this seemly clear cut relationship between NLI labels and EXEMPLARS (i.e., EXCEPTIONS contradict, INSTANTIATIONS are entailed) actually indicates systematic NLI model errors and the inability of the NLI schema to capture the nuances of default inheritance reasoning.

Consider the EXEMPLARS in Figure 3, relating to the generic "Birds can fly". Here we see that while some EXCEPTIONS contradict the generic as premise, these are actually false statements. True EXCEPTIONS *should be labeled neutral* by NLI

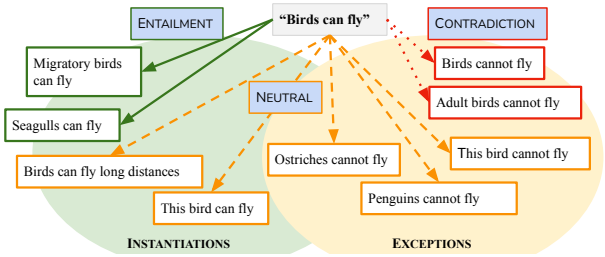

Figure 3: Example generic with EXEMPLARS and correct NLI labels.

with respect to the generic, since they are not "unlikely to be true given the information in the premise" (Dagan et al., 2013) (i.e., NLI contradictions). Note that this relies the lack of explicit quantification in generics. With INSTANTIATIONS, we observe that the NLI relationship may be either neutral or entailment. These highlight the complicated relationship between NLI and EXEMPLARS and the systematic errors made by NLI models when presented with such pairs involving default inheritance reasoning.

Additionally, The examples in Figure 3 also highlight that the NLI neutral label does not distinguish between statements that are true but not entailed/contradictory (e.g., Ostriches cannot fly) and statements that may not even be true (e.g., "This bird can/cannot fly"). Our generics EXEMPLARS emphasize the importance of developing a more fine-grained notion of NLI to model this default inheritance reasoning.

7 Conclusion

In this work, we draw on insights from linguistics to propose a novel computational framework to automatically generate valid EXEMPLARS for generics, as a step towards capturing the nuances of human reasoning for generics. Our system generates $\sim 20k$ EXEMPLARS for 653 generics and outperforms our few-shot baseline at generating viable examples, while remaining more controllable. We also demonstrate the importance of carefully constraining the decoding and underline the inability of current NLI models to reason about and represent the relationship between generics and EXEMPLARS. In the future, we plan to further investigate the role of generics EXEMPLARS in reasoning and NLI and to additionally study generics involving people and actions.

References

Anonymous. 2022. GenGen dataset.

Sumithra Bhakthavatsalam, Chloe Anastasiades, and Peter Clark. 2020. Genericskb: A knowledge base of generic statements. *ArXiv*, abs/2005.00660.

Michael Boratko, Xiang Lorraine Li, Rajarshi Das, Timothy J. O'Gorman, Daniel Le, and Andrew McCallum. 2020. Protoqa: A question answering dataset for prototypical common-sense reasoning. *ArXiv*, abs/2005.00771.

Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Gerhard Brewka. 1987. The logic of inheritance in frame systems. In *IJCAI*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Greg N. Carlson. 1977. Reference to kinds in english. In *Ph.D. dissertation, University of Massachusetts, Amherst*.

Greg N. Carlson. 1989. On the semantic composition of english generic sentences. In Gennaro Chierchia, Barbara H Partee, and Raymond Turner, editors, *Properties, Types and Meaning, Vol. II. Semantic Issues*. Dordrecht: Kluwer.

CDC. About malaria. https://www.cdc.gov/malaria/about/biology/index.html. Accessed: 2022-01-15.

Ariel Cohen. 1996. *Think generic! The meaning and use of generic sentences*. Carnegie Mellon University.

Ariel Cohen. 1999. Generics, frequency adverbs, and probability. *Linguistics and philosophy*, 22(3):221–253.

Ariel Cohen. 2004. Generics and mental representations. *Linguistics and Philosophy*, 27(5):529–556.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. Recognizing textual entailment: Models and applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220.

Barry Devereux, Lorraine K. Tyler, Jeroen Geertzen, and Billi Randall. 2014. The centre for speech, language and the brain (cslb) concept property norms. *Behavior Research Methods*, 46:1119 – 1127.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Renée Elio and Francis Jeffry Pelletier. 1996. On reasoning with default rules and exceptions. In *Proceedings of the 18th conference of the Cognitive Science Society*, pages 131–136.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Maxwell Forbes, Jena D. Hwang, Vered Shwartz, Maarten Sap, and Yejin Choi. 2020. Social chemistry 101: Learning to reason about social and moral norms. In *EMNLP*.

M. Ginsberg. 1987a. Introduction. Morgan Kaufmann, Los Altos, CA.

Matthew L. Ginsberg. 1987b. Readings in nonmonotonic reasoning. In *AAAI*.

Yael Greenberg. 2007. Exceptions to generics: Where vagueness, context dependence and modality interact. *Journal of Semantics*, 24(2):131–167.

Steve Hanks and Drew McDermott. 1986. Default reasoning, nonmonotonic logics, and the frame problem. In *AAAI*.

Paul Haward, Laura Wagner, Susan Carey, and Sandeep Prasada. 2018. The development of principled connections and kind representations. *Cognition*, 176:255–268.

John F. Horty and Richmond H. Thomason. 1988. Mixing strict and defeasible inheritance. In *AAAI*.

Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning whom to trust with mace. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1120–1130.

Jena D. Hwang, Chandra Bhagavatula, Ronan Le Bras, Jeff Da, Keisuke Sakaguchi, Antoine Bosselut, and Yejin Choi. 2021. Comet-atomic 2020: On symbolic and neural commonsense knowledge graphs. In *AAAI*.

Tomasz Imielinski. 1985. Results on translating defaults to circumscription. In *IJCAI*.

Nirit Kadmon and Fred Landman. 1993. Any. *Linguistics and philosophy*, 16(4):353–422.

Sangeet Khemlani, Sarah-Jane Leslie, and Sam Glucksberg. 2009. Generics, prevalence, and default inferences. *Proceedings of the 31st annual cognitive science society*, pages 443–448.

Sangeet Khemlani, Sarah-Jane Leslie, and Sam Glucksberg. 2012. Inferences about members of kinds: The generics hypothesis. *Language and Cognitive Processes*, 27(6):887–900.

Arnold Kochari, Robert Van Rooij, and Katrin Schulz. 2020. Generics and alternatives. *Frontiers in Psychology*, 11:1274.

Manfred Krifka. 1987. An outline of genericity. Seminar für natürlich-sprachliche Systeme der Universität Tübingen.

Manfred Krifka et al. 2012. Definitional generics. *Genericity*, pages 372–389.

Dimitra Lazaridou-Chatzigoga and Linnaea Stockall. 2013. Genericity, exceptions and domain restriction: experimental evidence from comparison with universals. In *Proceedings of Sinn und Bedeutung*, volume 17, pages 325–343.

Sarah-Jane Leslie. 2007. Generics and the structure of the mind. *Philosophical perspectives*, 21:375–403.

Sarah-Jane Leslie. 2008. Generics: Cognition and acquisition. *Philosophical Review*, 117(1):1–47.

Sarah-Jane Leslie. 2017. The original sin of cognition: Fear, prejudice, and generalization. *The Journal of Philosophy*, 114(8):393–421.

Sarah-Jane Leslie, Sangeet Khemlani, and Sam Glucksberg. 2011. Do all ducks lay eggs? the generic overgeneralization effect. *Journal of Memory and Language*, 65(1):15–31.

David Lewis and Edward L. Keenan. 1975. *Adverbs of quantification*, page 3–15. Cambridge University Press.

Vladimir Lifschitz. 1989. Benchmark problems for nonmonotonic reasoning. In *Proceedings of the Second international Workshop on Non-monotonic Reasoning*.

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

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, et al. 2021. Neurologic a* esque decoding: Constrained text generation with lookahead heuristics. *arXiv preprint arXiv:2112.08726*.

Alda Mari, Claire Beyssade, and Fabio Del Prete. 2012. *Genericity*, volume 43. OUP Oxford.

Ken McRae, George S. Cree, Mark S. Seidenberg, and Chris McNorgan. 2005. Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, 37:547–559.

Hugo Mercier and Dan Sperber. 2017. *The enigma of reason*. Harvard University Press.

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

Kanishka Misra, Allyson Ettinger, and Julia Taylor Rayz. 2021. Do language models learn typicality judgments from text? *ArXiv*, abs/2105.02987.

Daniel N. Osherson, Edward E. Smith, Ormond Wilkie, Alejandro López, and Eldar Shafir. 1990. Category-based induction. *Psychological Review*, 97:185–200.

James P. Van Overschelde, Katherine A. Rawson, and John Dunlosky. 2004. Category norms: An updated and expanded version of the battig and montague (1969) norms. *Journal of Memory and Language*, 50:289–335.

Francis Jeffry Pelletier. 2009. Are all generics created equal? *Kinds, Things, and Stuff: Mass Terms and Generics*, pages 60–79.

Francis Jeffry Pelletier and Renée Elio. 2005. The case for psychologism in default and inheritance reasoning. *Synthese*, 146(1):7–35.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *EMNLP*.

David L. Poole. 1988. A logical framework for default reasoning. *Artificial Intelligence*, 36:27–47.

Sandeep Prasada and Elaine M Dillingham. 2006. Principled and statistical connections in common sense conception. *Cognition*, 99(1):73–112.

Sandeep Prasada and Elaine M Dillingham. 2009. Representation of principled connections: A window onto the formal aspect of common sense conception. *Cognitive Science*, 33(3):401–448.

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

R. Reiter. 1978. On reasoning by default. In *Proceedings of TINLAP-2*, pages 210–218, University of Illinois. Association of Computational Linguistics.

Raymond Reiter. 1980. A logic for default reasoning. *Artificial Intelligence*, 13:81–132.

Lance J. Rips. 1975. Inductive judgments about natural categories. *Journal of Verbal Learning and Verbal Behavior*, 14:665–681.

Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. 2018. Object hallucination in image captioning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium. Association for Computational Linguistics.

Rachel Rudinger, Vered Shwartz, Jena D Hwang, Chandra Bhagavatula, Maxwell Forbes, Ronan Le Bras, Noah A Smith, and Yejin Choi. 2020. Thinking like a skeptic: Defeasible inference in natural language. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4661–4675.

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

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Wilson L Taylor. 1953. "cloze procedure": A new tool for measuring readability. *Journalism quarterly*, 30(4):415–433.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *"Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*.

A Limitations and Risks

The generics we source (see §5.1) is exclusively in English. Therefore, our approach may not be suited to all possible generics in all languages. In particular, our system does not handle generics where valid INSTANTIATIONS include negating (§3.3) the concept. This is due to the restriction that most English generation is left-to-right and it is not possible to define a closed set of possible concept negations for the prompt.

In this work, we do not generate EXEMPLARS for generics involving human referents (e.g., professions, nationalities). We exclude generics involving human referents to mitigate the risk of generating socially biased EXEMPLARS or harmful stereotypes (e.g., "Black folks go to jail for crimes" for the generic "People go to jail for crimes"). Additionally, handling of human stereotypes require methods that are beyond the scope of this paper. For example, a socially-aware EXCEPTIONS to a generic like "Girls wear dresses" would be "Boys wear dresses, too". This would require the understanding of the possible subtext of such a statement (e.g. "Only girls wear dresses"), which is beyond the current capabilities of this study and worthy of future exploration.

Finally, we note that while it is not the intended purpose of our system, a malicious user could still use our system to generate EXEMPLARS for a generic involving a person and propagate potentially harmful social biases.

B Generics Definitions

We condense the five generic types proposed by Leslie (2007, 2008) into our three categories (§3.2). The five types are:

- **Quasi-definitional**: generics concerning properties that are assumed to be universal among a concept. This is the same as our quasi-definitional category, see (a) Table 1. The property is considered a defining characteristic of the concept.
- **L-Principled**: generics concerning properties that are prevalent among a concept and are viewed as inherent, or connected in a principled way (Prasada and Dillingham, 2006, 2009; Haward et al., 2018). These generics are called principled in Leslie (2007, 2008). Note, these generics make up only one half of our "principled" category (§3.2). See first example for category (b) in Table 1; the second example there does *not* fit Leslie (2007, 2008)'s definition of principled (i.e., L-principled).
- **Striking**: generics describing properties that are uncommon and often dangerous, and members of the concept are *disposed* to possess them if given the chance (Leslie, 2017). For example, the striking generic "Sharks attack swimmers" assumes all sharks are capable of attacking swimmers. These generics constitute the second half of our "principled" category. See second example (not first) for category (b) in Table 1.
- **Majority characteristic**: generics concerning properties that are neither deeply connected to the concept nor striking but occur in the majority of members of the concept. These constitute one half of our "characterizing category". See example for (c) in Table 1.
- **Minority characteristic**: generics concerning properties that are neither deeply connected to the concept nor striking but occur in the minority of members of the concept. For example, "Lions have manes", since only adult male lions (the minority of the lion population) have manes. These constitute the second half of our "characterizing category".

Both L-principled and striking generics are true in-virtue-of a secondary factor and therefore we group these into one category (i.e., "principled"; see §3.2). For L-principled generics, this may be a factor that causes the property to occur in the concept (e.g., Birds can fly because they have wings). For striking generics, it is the assumed predisposition of the kind to possess the property if given the chance.

For quasi-definitional generics, because the property is considered defining to concept, there is no implied secondary factor in-virtue-of which the generic is true. Therefore, these generics are descriptive and we put them in a separate category from striking and L-principled generics.

Finally, majority and minority characteristic generics are ambiguous in their interpretation. For example, "Lions have manes" can be interpreted as being true in-virtue-of some secondary factor (e.g., as a signal of fitness) or as being a merely accidental relationship. If the interpretation is the former, then lions without manes are valid EXCEPTIONS (e.g., lion cubs, female lions), while if the

interpretation is the latter then then other attributes of lions are valid EXCEPTIONS (e.g., claws, fur).

Additionally, we note that a generic can focus on the presence of the property within the concept (e.g., "Birds can fly" is concerned with which birds can fly) **or** can focus on the presence of the concept within holders of the property (e.g., "Triangles have three sides" is more concerned with what concepts have three sides). We will say that the former kind of generic is *concept-oriented* and the latter is *property-oriented*. A generic can be both concept and property oriented if it is ambiguous between the two readings (e.g., "Aspirin relieves headaches").

In this work, we have discussed and used definitions only for concept-oriented generics. However, similar definitions and logical forms can be derived for property-oriented generics. Note that only the logical forms for quasi-definitional generics and their EXCEPTIONS change if the generic is property-oriented. In particular, the K and P in both logicals form for (a) in Table 1) can swapped to obtain the property-oriented versions. In this work, we do not deal with property-oriented generics and their EXEMPLARS due to the limitations of English generation, as mentioned in Appendix A.

C Annotation

For all annotation tasks, three annotators are used per HIT. When filtering annotators using MACE, we remove annotators with competence below 0.5 (or the median, if lower).

Generic Type Instructions for annotating generic types (§3.2) are shown Figure 4 (for the first pass) and Figure 5 (for the second pass). The first pass categorizes generics as either characterizing or not (either quasi-definitional or principled). The second pass categorizing non-characterizing generics as either quasi-definitional or principled.

Truthfulness Task Instructions for annotating output generations for truthfulness (§4.3) are shown in Figure 6.

EXEMPLARS Gold Labels For the INSTANTIATION template generations, annotators are asked whether the generation contradicts the original generic. Instructions are shown in Figure 8. However, for the exception template generations, an EXCEPTION is not a contradiction of the generic itself but of an associated logical form. For example, "Penguins cannot fly" does not actually contradict the generic itself ("Birds can fly") but a modified form of the generic involving quantification (i.e., "All birds can fly"). Therefore, we ask annotators whether the generation contradicts two modified forms of the generic. Instructions are shown in Figure 7.

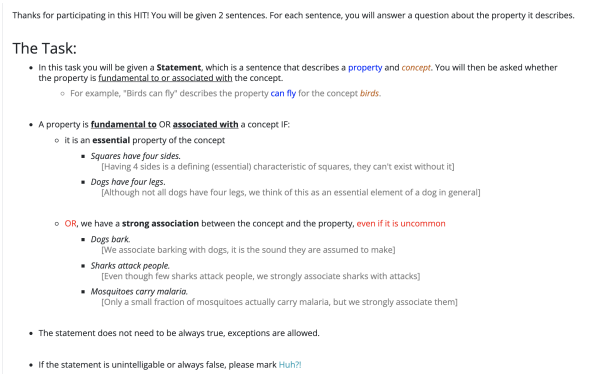

Figure 4: Task instructions for first part of the generic type categorization annotation (§5.2).

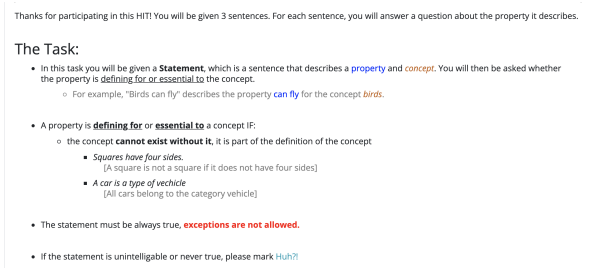

Figure 5: Task instructions for second part of the generic type categorization annotation (§5.2).

We obtain modified forms of the generic by first converting the logical forms in Table 1 into a natural language templates by adding a universal quantifier. Then we apply the template to the generic itself. Specifically, from $K(x) \wedge r(x,y) \implies P(y)$ (e.g., for quasi-definitional generics) we derive "`[K] [REL] ONLY [P]`". For example, "mosquitoes drink *only* blood", which is contradicted by mosquitoes that drink something other than blood. Notice, that exceptions from templates 1 and 2 will contradict these statements. Similarly, for $K(x) \wedge P(y) \implies r(x,y)$ we derive "`ALL [K] [REL] [P]`". For example, "*All* birds can fly", which is contradicted by birds that cannot fly. Exceptions from templates 3 and 4 will contradict these statements.

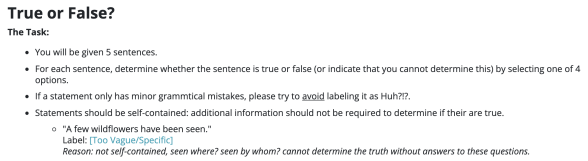

Figure 6: Task instructions for annotating truthfulness (§5.2).

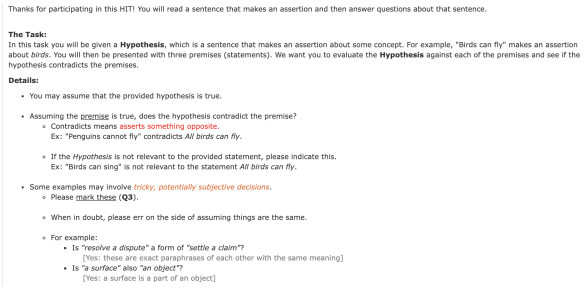

Figure 7: Task instructions for annotating validity of EXCEPTIONS (§5.2).

D Implementation Details

D.1 Data

We use the in-submission GenGen data (Anonymous, 2022). The dataset contains English generics, automatically generated via NeuroLogic* (Lu et al., 2021) with GPT2-XL. For this study, we source from the subset of GenGen's test set found to be valid by the discriminator with probability at least 0.5 (768 generics). Of these, we exclude all mentions of human referents (e.g., kinship labels, nationalities, titles, professions) and actions (e.g., studying for a test) to arrive at a dataset of 653 generics. We remove human referents using a seed set of human referent terms compiled based on WordNet (Miller, 1995) and will be provided with the system code. We remove mentions of actions by excluding generics beginning with "In order to". The GenGen dataset is licensed under CC-BY and our usage aligns with the intended use of the data.

Preprocessing We remove adverbs of quantification (i.e., usually, typically, generally) from the generics and exclude generics with verbs of consideration (i.e., consider, posit, suppose, suspect, think). We also convert hedging statements to more explicit forms (e.g., "may have to be" to "must be").

Partitions The data splits for training the truth discriminator and validity discriminators are shown in Table 9 and Table 10 respectively.

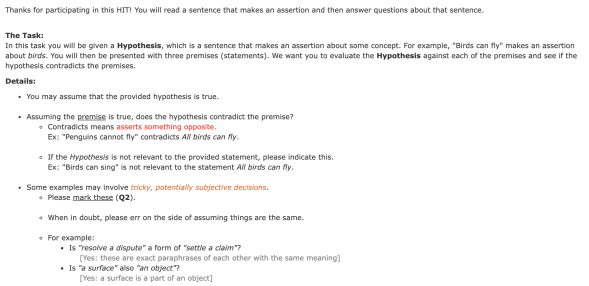

Figure 8: Task instructions for annotating validity of insts (§5.2).

	Train	Dev	Test	All
True	2831	412	433	3676
False/Non-salient	3180	367	442	3989
Total	6011	779	875	7665

Table 9: Data split statistics for truthfulness discriminator (§4.3).

D.2 Tools

For extracting components of the generic data we use spacy[8] for dependency parsing. We use *inflect*[9] to obtain plural and singular word forms and *mlconjug3*[10] to conjugate verbs. We use *nltk*[11] for additional synonyms.

D.3 Hyperparameters

To obtain subtypes from GPT-3 we use the *davinci* model and top-p sampling with $p = 0.9$, temperature 0.8 and maximum length 100 tokens. We use the top 5 sequences to obtain subtypes. For NLI scores, we use RoBERTa fine-tuned on MNLI (Williams et al., 2018) available from AllenNLP[12]. For the GPT-3 baseline we use the davinci model and top-p sampling 1.0, temperature 0.8, maximum length 50 tokens and top 5 sequences. Prompts for GPT-3 are given in Appendix E. GPT2-XL has 1.5 billion parameters, GPT-3 has 175 billion parameters. Our experiments are done using Quadro RTX 8000 GPUs.

For generation with NeuroLogic*, we use GPT2-XL (Radford et al., 2019) with a maximum length of 50 tokens and a beam size of 10 with temperature 10000000. We set the constraint satisfaction tolerance to 3. This means that at each step, only candidates whose number of satisfied constraints

[8] https://spacy.io/
[9] https://pypi.org/project/inflect/
[10] https://pypi.org/project/mlconjug3/
[11] https://www.nltk.org/
[12] https://demo.allennlp.org/textual-entailment/roberta-mnli

		Train	Dev	Test	All
EXCEPTION	Valid	342	35	35	412
	Invalid	462	72	53	587
	Total	804	107	88	999
INSTANTIATION	Valid	374	38	29	441
	Invalid	466	38	33	537
	Total	840	76	62	978

Table 10: Data split statistics for validity discriminators (§4.4).

is within three of the maximum so far are kept. The 'look ahead' is also set to 3; look ahead three generation steps during decoding to estimate future constraint satisfaction. During prompt construction, take the top $k_p = 10$ prompts. If the generic produced less than 10 prompts total, we take half so that low quality prompts are not used even if few are produced. After ranking the output, we keep the top $k_r = 10$ generations for a template, keeping at most 2 per prompt.

For the truth discriminator, we fine-tune the model for 5 epochs using a batch size of 16 and learning rate $1e-5$ and random seed 29725, selected by manual grid search.

For the validity discriminators, we fine-tune *the truth discriminator* for 3 epochs with a batch size of 16 and learning rate $3e-5$. The instantiation discriminator uses a random seed of 4427 and the exception discriminator 4457. Hyperparameters are again selected by manual grid search.

E GPT-3 Prompts

E.1 Subtyping

To obtain subtypes from GPT-3, we first categorize the kinds into six categories: person, animal, other living (e.g., plants), location, temporal (e.g., Thursday), and other (e.g., candle, soup) (Table 11). For each category, we construct a separate prompt for GPT-3 containing one type and five example subtypes. Then, for each kind we use the prompt from its assigned category to obtain subtypes. Note that we exclude all generics where the kind is "person". This is to avoid producing or repeating stereotypes.

To determine the category, we use seed lists, for person, animal, other living, and locative, or the presence of prepositional beginnings ("On", "In", "At", "During"), for locative and temporal. The "other" category encompasses all kinds that do not fit into another category.

E.2 Few-shot Baseline

The prompts for our few-shot baseline are shown in Table 12. The three examples in the table are provided each on a separate line. Appended to the prompt is a fourth generic and the necessary connective. The same connective is used across all exception (instantiation) templates and is chosen through manual experimentation. We use "But also" for EXCEPTIONS and "For example" for INSTANTIATIONS.

Category	Prompt Concept	Prompt Subtypes
Animal	birds	sparrow, canary, large bird,bird of prey, sea bird
Other living	apple tree	small apple tree, flowering apple tree,apple tree with ripe apples, granny smith apple tree, young apple tree
Locative	hotels	beach hotel, boutique hotel, resort, bed and breakfast, five star hotel
Temporal	day	morning, hot day, short day, afternoon, evening
Other	candles	scented candle, advent candle, tealight, candle made from beeswax, candle that smells floral
	can of soup	can of tomato soup, can of mushroom bisque, expired can of soup, unopened can of soup, organic can of soup

Table 11: Prompts for generating subtypes with GPT-3.

	Template	Prompt Examples
(1)	[KIND + REL]^{p} [NEG-PROP]$^{\mathcal{C}}$	Elephants are found in zoos. But also elephants are found in the wild in Africa. Viruses are spread through body fluids. But also viruses are spread in the air. A hair dryer is used to dry hair. But a hair dryer can also be used to dry clothes.
(2)	[KIND$_{sub}$ + REL]^{p} [NEG-PROP]$^{\mathcal{C}}$	Elephants are found in zoos. But also African elephants are found in the wild in Africa. Viruses are spread through body fluids. But also coronaviruses are spread in the air. A hair dryer is used to dry hair. But also an electric hair dryer can be used to dry clothes.
(3)	[KIND + NEG-REL]^{p} [PROP$_{sub}$]$^{\mathcal{C}}$	Dogs protect buildings from intruders. But also dogs do not protect apartment buildings from intruders. Cowsheds are found on farms. But also cowsheds are not found in orchards. The sun produces radiation. But also the sun does not produce x-rays.
(4)	[KIND$_{sub}$ + NEG-REL]^{p} [PROP]$^{\mathcal{C}}$	Birds can fly. But also penguins cannot fly. Ducks lay eggs. But also male ducks do not lay eggs. Dogs protect buildings from intruders. But also very small dogs do not protect buildings from intruders.
(5)	[KIND$_{sub}$ + REL]^{p} [PROP]$^{\mathcal{C}}$	Birds can fly. For example, seagulls can fly. Dogs protect buildings from intruders. For example, pitbulls protect buildings from intruders. Ducks lay eggs. For example, female ducks lay eggs.
(6)	[KIND + REL]^{p} [PROP$_{sub}$]$^{\mathcal{C}}$	Viruses are spread through body fluids. For example, viruses are spread through saliva. Dogs protect buildings from intruders. For example, dogs protect some private homes from intruders. Cowsheds are found on farms. For example, cowsheds are found on dairy farms.
(7)	[KIND$_{sub}$ + REL]^{p} [PROP$_{sub}$]$^{\mathcal{C}}$	Birds can fly. For example, Canadian geese fly long distances to migrate. Ostriches lay eggs. For example, female ostriches lay large spotted eggs. Elephants are found in zoos. For example, African elephants are found in most large zoos.

Table 12: Prompts for GPT-3 as Few-shot Baseline.