"What do you call a *dog* that is incontrovertibly true? *Dog*ma": Testing LLM Generalization through Humor

Alessio Cocchieri* **Luca Ragazzi*** **Paolo Italiani***
Giuseppe Tagliavini **Gianluca Moro***
Department of Computer Science and Engineering, University of Bologna, Italy
{a.cocchieri, l.ragazzi, paolo.italiani,
giuseppe.tagliavini, gianluca.moro}@unibo.it

Abstract

Humor, requiring creativity and contextual understanding, is a hallmark of human intelligence, showcasing adaptability across linguistic scenarios. While recent advances in large language models (LLMs) demonstrate strong reasoning on various benchmarks, it remains unclear whether they truly adapt to new tasks like humans (i.e., generalize) or merely replicate memorized content. To explore this, we introduce PHUNNY, a new humor-based question-answering benchmark designed to assess LLMs' reasoning through carefully crafted puns. Our dataset is manually curated to ensure novelty and minimize data contamination, providing a robust evaluation of LLMs' linguistic comprehension. Experiments on pun comprehension, resolution, and generation reveal that most LLMs struggle with generalization, even on simple tasks, consistently underperforming the human baseline. Additionally, our detailed error analysis provides valuable insights to guide future research.[1]

1 Introduction

What makes humans laugh? Humor combines creativity with linguistic nuance to transform ordinary concepts into funny and unexpected ideas. Yet, despite its playful nature, humor is one of the most sophisticated forms of linguistic reasoning (Brock, 2017). It challenges us to infer meanings, navigate ambiguity, and understand context–skills closely linked to higher intelligence (Christensen et al., 2016; Arslan et al., 2021). For example, "*what do you call a rat that thinks logically?*" The answer, "*rational*," leverages linguistic constructs between the word "*rat*" and the concept of reasoning, creating humor that humans easily grasp through logic (Kao et al., 2016). This ability to connect the dots is a hallmark of human intelligence. When we laugh, we implicitly rely on our capacity to apply knowledge to new scenarios (i.e., *generalize*)–a skill central to how we understand and use language (Keysers et al., 2020).

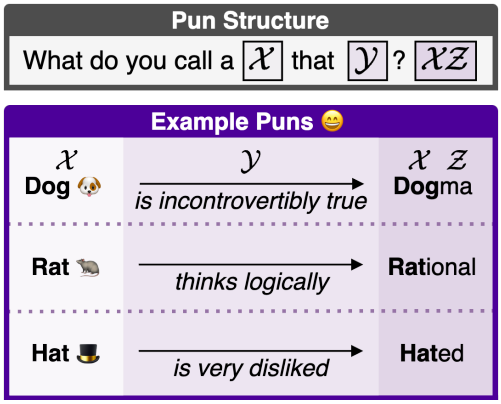

Figure 1: **The pun-based reasoning framework behind our PHUNNY dataset.** We assess linguistic generalization through humor in a human-intuitive format.

Can AI achieve the same? Large language models (LLMs) have shown strong performance in NLP tasks like text summarization (Moro and Ragazzi, 2022, 2023; Frisoni et al., 2023; Ragazzi et al., 2024), question answering (QA) (Kwiatkowski et al., 2019; Frisoni et al., 2024) and even humor detection (Xu et al., 2024). Still, a critical question remains: do they truly master language and generalize like humans, or are they simply reproducing patterns from their training data? To investigate whether LLMs exhibit robust performance on data outside their training distribution, we design a humor-based QA task grounded in *structured puns* (see Figure 1). Unlike existing benchmarks that test reasoning through complex, multi-step tasks (Ribeiro et al., 2023; Srivastava et al., 2023), we simplify the challenge to single-step word transformations in a humorous context, isolating reasoning from task complexity. For instance, consider the pun: "***Q:** what do you call a dog that is incontrovertibly true? **A:** dogma.*" Here, humor arises

*These authors contributed equally (co-first authors).
[1] https://disi-unibo-nlp.github.io/Phunny

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22922–22937
July 27 - August 1, 2025

by transforming the word "*dog*" into "*dogma*" via a morpho-semantic rule. To support this, we introduce **PHUNNY**,[2] a curated benchmark of 350 novel structured puns, constructed through a two-stage pipeline: (1) *pun design*, where puns were crafted manually for creativity and humor, and (2) *contamination check*, using an automatic pipeline with web searches and LLM-as-a-judge to exclude online content potentially present in pretraining data. By focusing exclusively on *suffix additions* (e.g., "*dog*" → "*dogma*"), PHUNNY aligns with human intuition, making jokes easy to understand.

Extensive experiments across three macro-tasks– pun comprehension, resolution, and generation– reveal the generalization limitations of state-of-the-art LLMs, with performance consistently falling below human thresholds. Notably, models perform especially poorly in distinguishing non-puns, frequently fabricating justifications even when no pun exists. These results highlight fundamental shortcomings in NLP, exposing failures in language understanding and compositional reasoning.

Our work makes the following key contributions:

❶ **PHUNNY:** A humor-based QA dataset of unconventional puns for evaluating model generalization in compositional reasoning.

❷ **Empirical insights:** LLMs struggle significantly on PHUNNY, revealing key limitations in language understanding absent in humans.

❸ **Broader implications:** This task exposes challenges in model robustness and generalization, prompting further NLP research.

2 Related Work

Puns in NLP Unlike standard tasks like summarization (Moro et al., 2022, 2023a,b,c, 2024a; Italiani et al., 2024) and question answering (Moro et al., 2024b), puns pose a distinct challenge for NLP models due to their intricate interplay of syntax, semantics, and humor. Prior work has focused on tasks such as pun recognition (Diao et al., 2018; Zou and Lu, 2019; Zhou et al., 2020; Jentzsch and Kersting, 2023; Horvitz et al., 2024), explanation (Sun et al., 2022b), and generation (He et al., 2019; Yu et al., 2020b; Mittal et al., 2022; Sun et al., 2022a; Tian et al., 2022), facilitated by benchmark datasets like SemEval 2017 Task 7 (Miller et al., 2017). More recently, Xu et al. (2024) evaluated LLMs on standard puns, such as homographic and heterographic types, which are among the most studied forms of wordplay.

Generalization A core challenge in NLP is to achieve robust performance in out-of-distribution settings. Benchmarks like GLUE (Wang et al., 2019) have driven progress but are often criticized for high overlap with pretraining corpora, which makes it difficult to disentangle reasoning from memorization. Initiatives like BIG-bench (Srivastava et al., 2023) addressed this by including challenging tasks to assess performance on unseen data. This concern aligns with earlier work on cross-domain transfer in NLP, which highlighted the importance of building systems that generalize beyond training distributions (Domeniconi et al., 2014b, 2016b, 2017; Moro et al., 2018).

Reasoning and Creativity Commonsense reasoning tasks, such as BoolQ (Clark et al., 2019), ReClor (Yu et al., 2020a), and CommonsenseQA (Talmor et al., 2019), rarely require linguistic creativity. Conversely, tasks like metaphor comprehension (Shutova and Sun, 2013), storytelling (Clark et al., 2018), and pun generation (Yu et al., 2018) emphasize creativity but lack structured reasoning.

3 Preliminaries

We introduce a novel type of structured pun, designed to evaluate the generalization capabilities of models in a QA task. Unlike traditional puns, mainly relying on homographic or heterographic wordplay (Xu et al., 2024), our puns involve syntactic transformations that align closely with the semantic structure of the question, requiring models to navigate explicit logical relationships.

Formally, we define a pun as a structured triplet $\mathcal{P} = \langle \mathcal{X}, \mathcal{Y}, \mathcal{XZ} \rangle$, where $\mathcal{X}$ is the question's subject, $\mathcal{Y}$ represents its intended purpose, and $\mathcal{XZ}$ is the punchline, formed by appending a suffix $\mathcal{Z}$ to $\mathcal{X}$. The resulting $\mathcal{XZ}$ must be a real word that is not created through morphological derivation (e.g., "*dog*" → "*doggy*") or compounding (e.g., "*rain*" → "*rainbow*"), but instead has a distinct etymological origin (e.g., "*dog*" → "*dogma*").

$\mathcal{P}$ is instantiated in the following QA format:

`What do you call a` $\mathcal{X}$ `that` $\mathcal{Y}$`?` $\mathcal{XZ}$ (1)

These minimal transformations enhance clarity and remove ambiguity, allowing us to assess LLMs on tasks that humans intuitively understand.

[2] https://disi-unibo-nlp.github.io/publications/

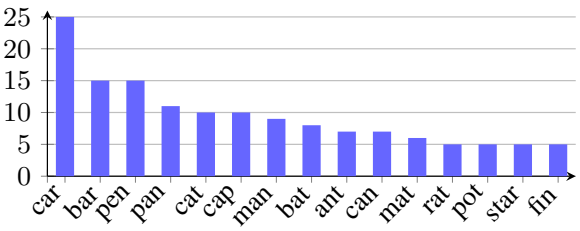

Figure 2: Top-15 most frequent subjects in PHUNNY.

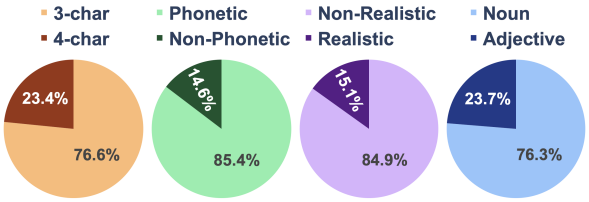

Figure 3: PHUNNY's data statistics distribution.

4 PHUNNY

To evaluate the generalization capabilities of LLMs, we present PHUNNY, a meticulously curated benchmark dataset of 350 hand-crafted structured puns.

4.1 Annotation Process

The construction of PHUNNY followed a rigorous two-stage pipeline designed to create a corpus that is both high-quality and innovative.

Step 1: Pun Design We manually crafted the puns using a systematic process to ensure linguistic creativity, coherence, and humor. ❶ We selected a base word ($\mathcal{X}$) to serve as the subject of the question (e.g., "*dog*"). ❷ We identified a word beginning with the base word to serve as the punchline ($\mathcal{XZ}$), ensuring it was meaningful as a standalone term (e.g., "*dogma*"). ❸ We retrieved the definition of the punchline ($\mathcal{Y}$) and refined it for clarity (e.g., "*incontrovertibly true*"). ❹ Finally, we combined these elements to form the complete pun, following the structure defined in Equation 1.

Step 2: Data Contamination To ensure the originality of the puns in PHUNNY and minimize overlap with LLM pretraining corpora, we developed an automated pipeline. For each complete pun, we used the DuckDuckGo API to retrieve the top-10 web search results. We then employed Gemini-1.5-Flash to verify whether each pun appeared in the retrieved documents. Pre-existing puns were excluded from the dataset, resulting in the removal of 20 contaminated entries. The prompt used and the full list of excluded puns are shown in Appendix C.

4.2 Data Statistics

Figure 2 shows the 15 most frequent subjects in PHUNNY used for pun formation. Figure 3 presents binary distributions of key features, highlighting linguistic patterns in wordplay, as discussed below.

Phonetic vs. Non-Phonetic. A substantial 85% of puns in PHUNNY rely on phonetic associations, where the punchline also begins with the same pronunciation as the subject (e.g., *dog* → *dogma*), making pun recognition more intuitive. In contrast, 15% are non-phonetic, involving words where the punchline starts with the same letters as the subject but differs in pronunciation (e.g., *rat* → *rational*).

Realistic vs. Non-Realistic. Plausibility plays a minor role in pun construction, as 85% of our puns are non-realistic, relying on absurd or illogical setups for humor. Conversely, only 15% are realistic, meaning their question-answer pairs follow a coherent–albeit humorous–logic.

3-Char vs. 4-Char. The dataset skews toward 3-character subjects (77%), while 4-character subjects appear in 23% of cases. This imbalance reflects the greater ease of finding punchlines that begin with 3-character words, making them more suitable for pun formation.

Noun vs. Adjective. Punchlines overwhelmingly favor nouns (76%), while adjectives account for 24%. This distribution reflects the dataset's design, where nouns were more frequently chosen as punchlines due to their ability to form clearer and more recognizable wordplay structures.

4.3 Task Formulation

We introduce three progressively challenging tasks to evaluate LLMs' ability to understand and generate our specific types of puns (see Figure 4).

❶ Pun Comprehension We assess if LLMs truly understand puns by distinguishing coherent ones from nonsensical counterparts. Unlike Xu et al. (2024), who separate recognition and explanation, we treat them as complementary, requiring the model to both identify and justify its interpretation. Given a pun as input, the model must determine whether it grasps the joke without being told it is one. To this end, we structure this task into two subtasks based on the type of input:

- **Coherent:** The model receives a valid pun with a well-formed punchline.
- **Misleading:** The model is given a nonsensical pun, generated by altering a coherent one.

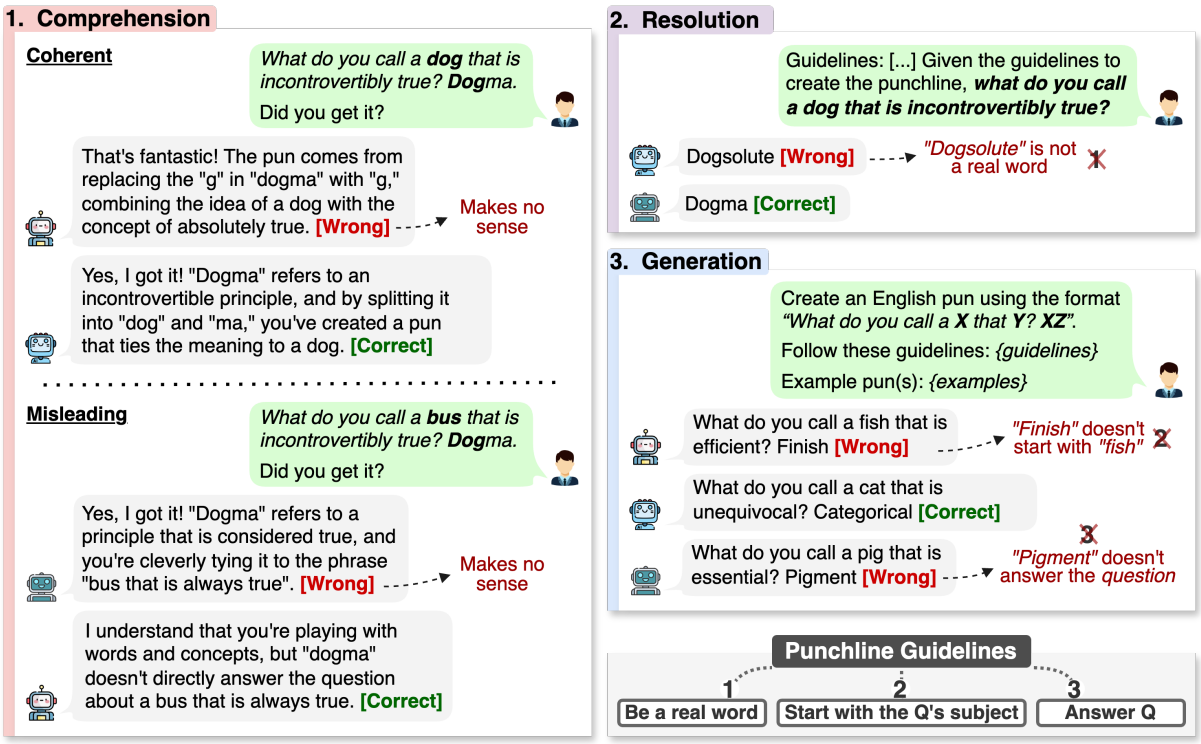

Figure 4: **Illustration of the tasks in PHUNNY.** (1) *Comprehension* evaluates the ability to interpret coherent and misleading puns; (2) *Completion* involves generating a valid punchline; and (3) *Generation* asks creating new puns following specific guidelines. Correct and incorrect outputs highlight reasoning capabilities and common errors.

We design misleading puns to test whether LLMs forcefully interpret non-puns as real puns, fabricating justifications even when none exist. To construct misleading puns, we modify each original pun's subject ($\mathcal{X}$) in two different ways:

- **Semantically similar swap:** We replace $\mathcal{X}$ with a term of high cosine similarity (e.g., "*cat*" $\rightarrow$ "*pet*").[3] While this introduces a subtle distortion in meaning, the original linguistic structure of the pun is no longer preserved.
- **Semantically dissimilar swap:** We substitute $\mathcal{X}$ with the most unrelated term (e.g., "*cat*" $\rightarrow$ "*bus*"), rendering the pun illogical.

In this task, we evaluate models on 1,050 puns: 350 coherent and 700 misleading, the latter derived from 350 similar and 350 dissimilar swaps.

❷ Pun Resolution This task probes if LLMs can generate an appropriate punchline to complete a given setup. We use the 350 coherent puns, with their punchlines serving as gold labels.

❸ Pun Generation This task assesses LLMs' ability to generate PHUNNY-style puns under two conditions. In the first, denoted **Free**, there are no restrictions on subject selection. Here, we evaluate LLMs on generating 50 puns. However, preliminary tests reveal a tendency for models to reuse the same subjects across generations, even under high-temperature settings, which limits diversity. To mitigate this issue and ensure a fairer evaluation, we also introduce the **Constrained** variant, where the subject is fixed. To support this setup, we identified all unique subjects in our dataset of coherent puns, resulting in a total of 104. Given that multiple puns can be generated from the same subject and that creativity varies across models, this task does not have predefined gold labels.

5 Experimental Setup

We conduct a series of experiments to evaluate model performance on PHUNNY, assessing both standard LLMs and reasoning-focused ones. Further, we perform an in-depth error analysis, providing qualitative examples to illustrate our findings.

Environmental Setup We conducted experiments on workstation equipped with two GPUs: an NVIDIA A100 (80 GB VRAM) for open models with $\geq$15B parameters and an NVIDIA RTX 3090 (24 GB VRAM) for models with $\leq$8B parameters.

[3]We compute word embeddings with the all-MiniLM-L6-v2 model available in the Hugging Face hub.

To enhance inference efficiency and throughput, we employed the vLLM library. For 70B-parameter models, AWQ quantization was applied to optimize resource utilization and accelerate generation. Other open-source models were run using the precision settings specified in their respective configuration files. OpenAI models were processed via the OpenAI Batch API to reduce costs, while Gemini models were accessed through the Gemini API.

5.1 Large Language Models

We evaluate 9 state-of-the-art LLMs, categorized into standard and reasoning-focused models, with further distinction between closed and open-source.
Standard: They follow conventional pretraining without explicit reasoning enhancements. For closed-source models, we use **GPT-4o** (Hurst et al., 2024) and **Gemini-2.0-Flash**. For open-source models, we employ **LLaMA-3.1-8B** (Dubey et al., 2024), **Phi-3.5** (Abdin et al., 2024b), **Phi-4-14B** (Abdin et al., 2024a), and **LLaMA-3.3-70B** (Dubey et al., 2024).
Reasoning: They are specifically optimized for enhanced reasoning capabilities. We evaluate the latest closed-source models, such as **Gemini-2.0-Flash-Thinking (FT)** and OpenAI's **o3-mini**.

5.2 Evaluation Criteria

To evaluate the models' ability to generalize to novel tasks–as humans typically do with ease–we used 5 few-shot demonstrations (see Appendix D for details) for Generation and Resolution. In contrast, Comprehension was assessed under a strict zero-shot setting. We then employed two prompting modalities: **Direct** and **Chain-of-Thought (CoT)** (Wei et al., 2022). The direct approach, applied only to open-source LLMs, requires the model to provide an immediate answer without engaging in explicit reasoning. In contrast, the CoT approach, applied to all LLMs, allows the model to reason step by step before arriving at a final answer. For Resolution and Comprehension tasks, we use *greedy decoding* to minimize randomness of results, while for Generation we set a default temperature of 1.0. Details on prompts are provided in Appendix A. Notably, inspired by benchmarks like IFEval (Zhou et al., 2023), our tasks are designed around verifiable instructions, which facilitates reliable and efficient evaluation.

Metrics for Comprehension We use a structured protocol combining automatic evaluation with human oversight. We report two main metrics: Coherent Pun Accuracy (**CPA**) and Misleading Pun Accuracy (**MPA**), measuring performance on coherent and misleading pun sets, respectively.

For CPA, the model has to explain whether it understood the pun. Responses starting with "No" are automatically marked as incorrect. Affirmative answers are judged using Gemini-1.5-Flash, which compares the model's explanation against our gold manually-crafted rationale and determines whether they are semantically equivalent. This automatic approach proved robust, with fewer than 1% of cases requiring manual correction due to ambiguity.

For MPA, the model must explain its rejection of a misleading pun. Affirmative responses are considered incorrect. Among negative answers, we verify whether the explanation is grounded and relevant. We observed that incorrect rejections were rare (under 1%), and limited manual checks were sufficient to ensure consistency. We further split MPA into MPA^{+} and MPA^{-}, based on whether the misleading pun involves semantically similar or dissimilar word substitutions.

Metrics for Resolution While both Generation and Comprehension are open-ended tasks, the Resolution task may also have multiple valid answers. To accommodate this, we adopt a hybrid evaluation approach that combines automatic metrics with human assessment when necessary. Specifically, to ensure flexibility while maintaining consistency, our evaluation follows three main criteria: (1) *exact match* (case-insensitive); (2) *derivative match*, which employs the WordNet lemmatizer to obtain base forms, verifies derivational relationships via WordNet synsets, and applies linguistic heuristics for common suffix transformations (e.g., `ing` $\rightarrow$ `er`, `ful` $\rightarrow$ `able`) as well as participle-agent relationships (e.g., `startle` $\rightarrow$ `startler`); and (3) *substring match*, where one answer is a substring of the other (e.g., `shocking` vs. `shock`).

If an answer satisfies any of these conditions while preserving the pun structure, it is automatically accepted as correct. Differently, if the answer does not start with the pun prefix, it is marked as surely false and excluded from further evaluation. For all remaining cases that do not meet these criteria, we conducted human evaluation to determine correctness. However, we emphasize that this is required for only a small percentage of cases–around 3% in the worst case per model.

We use three key metrics: Accuracy (**ACC**),

Models	CPA	MPA$^-$	MPA$^+$	MPA
o3-mini ✯	78.3	6.0	3.4	4.7
Gemini-2.0-FT ✯	71.1	6.9	24.6	15.8
LLaMA-3.3 (70B)	70.0	29.4	29.1	29.3
GPT-4o ✯	64.9	14.9	17.4	16.2
Phi-4 (14B)	64.6	9.4	13.7	11.6
Gemini-2.0-Flash ✯	44.6	44.9	35.7	40.3
GPT-4o-mini ✯	36.3	10.9	11.7	23.6
LLaMA-3.1 (8B)	29.7	0.0	0.3	0.2
Phi-3.5 (3B)	14.9	48.3	47.1	47.7
Humans	87.9	87.3	94.4	90.9

✯ Closed-source models. Violet: Reasoning models.

Table 1: **Model performance on the Comprehension task.** Models are ordered based on decreasing CPA.

Models	ACC	VPA	EWA
o3-mini ✯	93.9	98.0	99.1
GPT-4o ✯	79.9	84.6	96.2
Gemini-2.0-FT ✯	70.6	80.2	88.1
Gemini-2.0-Flash ✯	69.5	75.9	87.8
LLaMA-3.3 (70B)	67.3	77.4	83.4
GPT-4o-mini ✯	64.5	73.0	89.5
Phi-4 (14B)	53.9	69.3	80.5
LLaMA-3.1 (8B)	27.9	31.7	96.8
Phi-3.5 (3B)	22.4	27.3	78.5
Humans	85.7	95.1	100.0

✯ Closed LLMs. Violet: Reasoning models.

Table 2: **Model performance on the Resolution task.** Models are ordered based on decreasing Accuracy.

Valid Prefix Accuracy (**VPA**), and Existing Word Accuracy (**EWA**). These metrics measure correctness (whether the model resolves the pun), linguistic adherence (whether the response starts with the question's subject), and word validity (ensuring the output is not a non-existent word), respectively.

Metrics for Generation We follow a three-step evaluation. First, we check if the pun's answer ($\mathcal{XZ}$) starts with the question's subject ($\mathcal{X}$). If so, we use Gemini-1.5-Flash as a judge to verify whether $\mathcal{XZ}$ derives from $\mathcal{X}$. If not, we check if $\mathcal{XZ}$ is meaningfully associated with the question's purpose ($\mathcal{Y}$). The pun is valid only if all conditions are met. We report Accuracy (**ACC**) as the primary metric for both task variants (i.e., Constrained and Free) and supplement the Free setting with a Creativity measure, evaluated by computing the proportion of unique subjects ($\mathbf{C_S}$) and unique answers ($\mathbf{C_A}$) generated by the models.

6 Results

6.1 Can LLMs Comprehend Puns?

Table 1 presents model results. For coherent puns, closed-source reasoning models perform best, with o3-mini reaching a CPA of 78.3. Among open-source models, LLaMA-3.3 (70B) closely follows Gemini-2.0-FT (70.0 vs. 71.1) and outperforms GPT-4o (64.9). As expected, larger models tend to achieve better CPA scores, suggesting a correlation between model size and pun comprehension. These results highlight an acceptable ability of LLMs to recognize true puns in our benchmark.

However, for misleading puns (see MPA), models perform remarkably poorly, revealing a critical weakness in their language understanding. Despite being strong at detecting real puns, o3-mini fails almost entirely in recognizing misleading ones, identifying only 4.7% correctly. This drop suggests that its reasoning abilities, while effective for structured patterns, break down when faced with nonsensical wordplay. In contrast, Phi-3.5, the weakest model in CPA, unexpectedly achieves the highest MPA score. Finally, when comparing different types of misleading puns (MPA$^+$ vs. MPA$^-$), we found no clear trend showing if semantically similar subject swaps confuse models more.

6.2 Can LLMs Resolve Puns?

Table 2 shows model results. o3-mini achieves an impressive 93.9% accuracy, even surpassing the human baseline. As in the Comprehension task, reasoning models outperform their standard counterpart, with o3-mini and Gemini-2.0-FT registering a +16% and +1.1% of absolute accuracy points over GPT-4o and Gemini-2.0-Flash, respectively. This correlates also with higher VPA and EWA. Again, performance correlates with model size, as smaller models like Phi-3.5 (3B) and LLaMA-3.1 (8B) struggle significantly, scoring only 22.38% and 27.91%. While all models seem to perform better in generating real words (EWA), they often fail to start punchlines with the question's subject, leading to lower VPA scores. This issue is especially severe for smaller models (e.g., Phi-3.5 at 27.4%), which struggle despite the task's simplicity. These results expose a fundamental flaw in instruction adherence, as models–regardless of size–fail to reliably follow even basic structural constraints.

Models	Constr. ACC	Free ACC	Free C_S	Free C_A
o3-mini ★	93.5	100.0	38.0	52.0
GPT-4o ★	85.3	46.0	60.0	88.0
Gemini-2.0-Flash ★	80.1	40.0	48.0	78.0
Gemini-2.0-FT ★	66.7	36.0	58.0	82.0
LLaMA-3.3 (70B)	25.5	15.0	15.0	57.5
GPT-4o-mini ★	41.7	24.0	36.0	84.0
Phi-4 (14B)	15.4	6.0	34.0	76.0
LLaMA-3.1 (8B)	13.1	20.5	59.0	89.7
Phi-3.5 (3B)	4.7	95.4	2.3	7.0
Humans	88.7	92.8	82.3	95.2

★ Closed-source LLMs. Violet: Reasoning models.

Table 3: **Model performance on both Generation tasks.** Models are ordered based on decreasing Accuracy on the Constrained task.

6.3 Can LLMs Generate Puns?

Table 3 presents model results. In the Constrained setting, o3-mini remains the top performer, achieving 93.5% accuracy in generating correct puns. Meanwhile, the smallest models, such as LLaMA-3.1 (8B) and Phi variants struggle significantly, highlighting their difficulty in maintaining humor within structured constraints. In the Free setting, all open-source models perform poorly in accuracy, except for Phi-3.5, whose result is undermined by extremely low creativity, indicating it repeatedly generates the same pun. Similarly, o3-mini achieves a perfect 100% accuracy, but this is offset by limited diversity, producing only 38% of unique puns. Notably, models show higher creativity in punchline generation than in subject selection but struggle to balance accuracy and originality, sometimes failing in both. Overall, this task is substantially more challenging than the Constrained setting, revealing notable shortcomings in model performance.

6.4 CoT vs. Direct Answering

Figure 5 provides a comparative analysis of open-source LLMs, contrasting CoT prompting with direct answers. The results suggest that the effectiveness of CoT is highly size-dependent. In the larger LLaMA-3.3 (70B) model, CoT substantially enhances performance across both resolution and generation tasks, likely due to its superior capacity to benefit from intermediate reasoning steps. In contrast, for smaller models such as LLaMA-3.1 (8B) and Phi variants, direct responses tend to perform better on key metrics, although CoT can still boost certain aspects (e.g., EWA in resolution).

6.5 How Humans Perform?

We conducted a series of human evaluations to assess the performance of English-fluent speakers in pun comprehension and resolution tasks. Specifically, we recruited 60 respondents to serve as the human baseline and subjected them to the same challenges that were presented to the LLMs. Further details regarding the human evaluation procedure can be found in Appendix E. Human scores are reported in the main tables for each task. In comprehension, participants achieved a CPA score of 87.9 and MPA of 90.9, outperforming all evaluated models on both dimensions. Notably, humans were able to accurately recognize misleading puns, an ability that the models did not demonstrate. For pun resolution, participants obtained an accuracy score of 85.7, ranking second only to the o3-mini model. While this result does not definitively indicate superior human performance, it is important to note that this task may pose additional challenges for humans, as it involves not only reasoning (the primary aspect we aimed to assess) but mainly associative memory. Finally, interesting patterns emerge in the generation task. First, compared to LLMs, humans find free generation easier, effectively leveraging their creative abilities. Second, while human accuracy remains below that of o3-mini, their creativity stands out. Both C_S and C_A scores are substantially higher than those of all tested LLMs, highlighting the unique human ability to produce diverse puns beyond learned patterns. This suggests that LLMs still struggle to match human linguistic flexibility in creative wordplay.

7 Error Analysis

Table 4 and Table 6 (in Appendix B) shows examples of LLM limitations on our benchmark.

7.1 Comprehension Errors

Our analysis of pun comprehension (see Table 4) reveals surprising behaviors difficulties in how LLMs distinguish why a question-answer pair constitutes a pun and whether a pun actually exists.

Coherent Errors In the Coherent Comprehension task, where the expected answer is always "Yes," models often misinterpret the underlying wordplay. For example, GPT-4o justifies "What do you call a car that fixes things? Carpenter" by linking "penter" to "repairer," overlooking the play

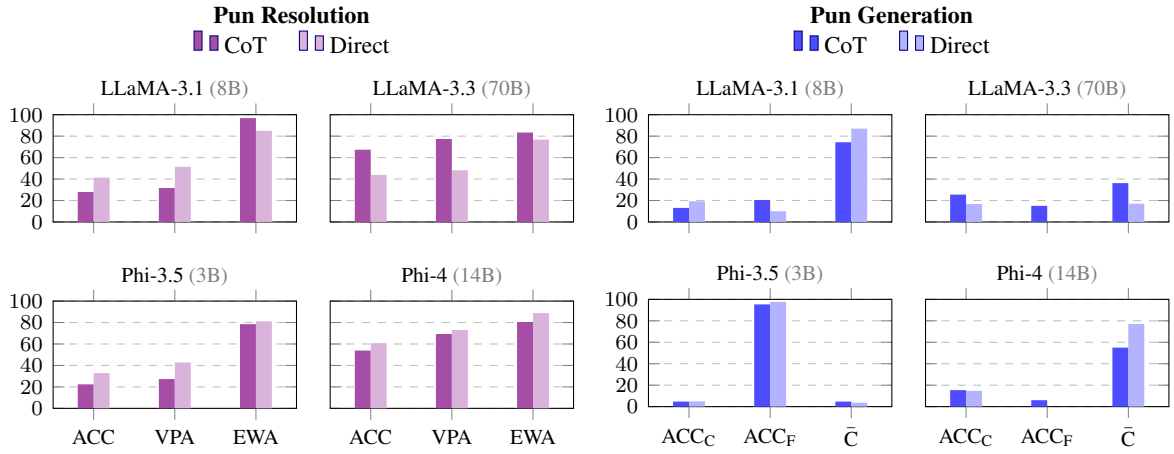

Figure 5: Comparison of open-source standard LLMs using CoT reasoning or direct responses.

on "car" and "carpenter." Similarly, o3-mini incorrectly rejects the pun, arguing that a carpenter works with wood rather than fixing cars. These errors suggest that while LLMs can recognize puns, they often fail to reason correctly about their linguistic mechanisms, relying on surface-level associations rather than true wordplay comprehension.

Misleading Errors In the Misleading Comprehension task, where the expected answer is always "No," models frequently exhibit overconfidence, forcing explanations to justify non-existent puns. This tendency becomes more pronounced in larger, reasoning-focused models like o3-mini, suggesting that stronger general knowledge may lead to overfitting spurious patterns rather than correctly rejecting non-puns. For example, o3-mini explains "What do you call a van that draws maps? Cartographer" as a valid pun, despite lacking wordplay. Similarly, Gemini-2.0-FT justifies "What do you call a jar that deals with language rules? Grammar" by forcing a wordplay connection to "gram jar." These errors reveal a key limitation: while LLMs excel at recognizing word associations, they struggle to distinguish genuine puns from coincidental phonetic similarities.

7.2 Generation and Resolution Errors

We identify three common errors made by LLMs during pun generation and resolution, as shown in Table 6. **(1) The model fails to use the subject as a prefix in the final answer**, indicating that it does not fully grasp the structural constraints of the wordplay, despite clear guidance in the prompt. **(2) The model generates a response that is directly derived from the subject**, meaning the answer is semantically related to the root word rather than adhering to the intended pun structure. **(3) The model creates an answer word that does not exist in the English vocabulary**, demonstrating a tendency to prioritize phonetic or morphological plausibility over linguistic validity.

Missing Prefix LLMs frequently omit the subject as a prefix in their generated answers, even when few-shot examples clearly demonstrate the expected structure. Instead, models often prioritize phonetic or semantic associations, deviating from the intended format. For instance, during generation, GPT-4o generates "Cavity" for "What do you call a cave that is hollow?" and Gemini-2.0-FT answers "Pane" for "What do you call a pain that is a window?"–both leveraging homophony rather than maintaining the prefix constraint. This issue persists in resolution tasks, where models misinterpret the prompt's structure, as seen in Gemini-2.0-FT's response of "Fashion" to "What do you call a fan that is stylish?" and GPT-4o's "Supernova" for "What do you call a star that is surprising?" These errors reveal a fundamental limitation: even advanced LLMs struggle to enforce explicit structural constraints, favoring surface-level linguistic patterns over compositional fidelity when phonetic or semantic cues offer alternative interpretations.

Derivative Answer A common error is the generation of derivative answers, where the model appends a suffix to the subject, producing a semantically related but humorless response. This reflects a lack of humor comprehension. For instance, in resolution, GPT-4o answers "batfish" to "What do you call a bat that swims?"–a valid term but devoid of wordplay. Similarly, Gemini-2.0-FT

	Coherent	Misleading
GPT-4o	**Pun:** What do you call a car that fixes things? carpenter **Explanation:** Yes, because "carpenter" sounds like "car" and "penter," where "penter" suggests someone who fixes things, like a "repairer."	**Pun:** What do you call a fact that is strong? potent **Explanation:** Yes, because "potent" can mean strong or powerful, and a strong fact can be described as potent.
Gemini-2.0-Flash	**Pun:** What do you call a prof that makes money? profit **Explanation:** Yes, for exactly two reasons. First, the word "profit" sounds like "prophet." Second, the joke plays on the similar sound between "prof" (short for professor) and "profit."	**Pun:** What do you call a pet that thinks logically? rational **Explanation:** What do you call a pet that blocks vision? cataract Yes, because a cataract is a clouding of the lens in the eye, which can block vision, and it sounds like "cataract" which includes "cat," a common pet.
o3-mini	**Pun:** What do you call a car that fixes things? carpenter **Explanation:** No, because a carpenter works with wood rather than fixing cars, which is what a mechanic does.	**Pun:** What do you call a van that draws maps? cartographer **Explanation:** Yes, because it's a pun—the term "cartographer" normally means someone who makes maps, so imagining a van that draws maps plays on that double meaning.
Gemini-2.0-FT	**Pun:** What do you call a arm that provides protection? armor **Explanation:** No, because armor is not an arm, it protects arms.	**Pun:** What do you call a jar that deals with language rules? grammar **Explanation:** Yes, because it's a wordplay on "gram jar" which sounds like "grammar," and grammar deals with language rules.
LLaMA-3.3-70B	**Pun:** What do you call a card that is a religious official? cardinal **Explanation:** Yes, for exactly two reasons. First, a cardinal is a high-ranking bishop in the Catholic Church. Second, it sounds like the playing card.	**Pun:** What do you call a van that causes destruction? carnage **Explanation:** Yes, for exactly two reasons. First, carnage is a word that means destruction. Second, it sounds like caravan, which is another word for van.

Table 4: Examples of errors in the Pun Comprehension task.

responds with "carseat" for "What do you call a car that is a painful swelling under the skin?", entirely missing the intended meaning. This issue persists in generation, where models fail to follow explicit instructions. Despite prompts discouraging derivative words, models often default to the most immediate association, as seen in GPT-4o's "What do you call a book that is small and concise? Booklet". These errors suggest an overreliance on lexical similarity at the expense of creative constraint adherence.

Nonexistent Answer Word In contrast to previous errors, this mistake arises when models overextend their creativity, generating unnatural words in an attempt to enforce humor or structural constraints. For example, GPT-4o produces "What do you call a seal that makes a request for reconsideration? Sealappeal," forming an unnatural blend that lacks meaningful wordplay. Similarly, LLaMA-3.3-70B generates "Beear" for "What do you call a bee that is a good listener?", an awkward fusion of "bee" and "hear" that does not align with our natural pun structures. A similar pattern appears in resolution, where GPT-4o responds with "cap-hate" to "What do you call a hat that is very disliked?" and LLaMA-3.3-70B answers "catipede" for "What do you call a cat that crawls on multiple legs?"–forced constructions that lack linguistic validity. These errors indicate that models sometimes prioritize phonetic resemblance and humor at the expense of fluency, underscoring the challenge of balancing structural adherence with meaningful wordplay.

8 Conclusion

We introduced PHUNNY, a novel benchmark dataset of human-crafted puns based on single-step word transformations that mirror human humor reasoning. While LLMs perform well on conventional tasks, our controlled setup reveals their limitations in handling out-of-distribution, creativity-driven challenges. Models often understand coherent puns but are easily misled by implausible ones. Even with reasoning-enhanced methods, they struggle with context-sensitive inference and often violate structural constraints. These results underscore a gap between surface-level language proficiency and the deeper reasoning abilities that underlie human-like generalization.

Limitations

It is important to acknowledge some limitations in our work. First, for simplicity, (1) in dataset construction, we limited the answer words to only nouns and adjectives, excluding verbs, and (2) focused solely on suffix additions in the punchline (e.g., "*dog*" → "*dogma*"), thereby ignoring other possible approaches such as prefix additions or fill-in-the-blank methods. These choices constrain the scope of our investigation, limiting our exploration of other potential patterns in pun-based tasks, which we aim to address in future work. Second, all puns were manually crafted in English. The ability of LLMs to comprehend and generate puns may differ across languages, as puns in languages other than English might have distinct definitions, structures, or purposes, suggesting a direction for future research. Third, building on our contaminated subset of web-mined puns, we aim to explore perturbation-based tasks that test whether models reconstruct inputs from incomplete or altered inputs (Domeniconi et al., 2014a, 2016a)–offering a more targeted analysis of data leakage.

Ethical Considerations

Since our work delves into humor, which is deeply contextual and culturally situated, we acknowledge that it could pose ethical challenges in ensuring that does not inadvertently perpetuate harmful stereotypes, biases, or offensive content. To address this, during pun construction, we took extra care to exclude subjects and answers that could be interpreted as perpetuating harmful or derogatory content. In particular, our design process was guided by explicit ethical considerations: we omitted topics known to be sensitive or historically marginalized, and our team critically reviewed all contributions to ensure that humor was achieved without compromising respect for any group or individual.

Acknowledgements

Research partially supported by AI-PACT project (CUP B47H22004450008, B47H22004460001); National Plan PNC-I.1 DARE initiative (PNC0000002, CUP B53C22006450001); PNRR Extended Partnership FAIR (PE00000013, Spoke 8); 2024 Scientific Research and High Technology Program, project "AI analysis for risk assessment of empty lymph nodes in endometrial cancer surgery", the Fondazione Cassa di Risparmio in Bologna; Chips JU TRISTAN project (G.A. 101095947). We thank the Maggioli Group[4] for partially supporting the Ph.D. scholarship granted to Paolo Italiani.

References

Marah I Abdin, Jyoti Aneja, Harkirat S. Behl, et al. 2024a. Phi-4 technical report. *CoRR*, abs/2412.08905.

Marah I Abdin, Sam Ade Jacobs, Ammar Ahmad Awan, et al. 2024b. Phi-3 technical report: A highly capable language model locally on your phone. *CoRR*, abs/2404.14219.

Deniz Arslan, Uğur Sak, and Nazmiye Nazli Atesgoz. 2021. Are more humorous children more intelligent? a case from turkish culture. *HUMOR*, 34:567 – 588.

Alexander Brock. 2017. Modelling the complexity of humour – insights from linguistics. *Lingua*, 197:5–15.

Alexander P. Christensen, Paul J. Silvia, Emily C. Nusbaum, and Roger E. Beaty. 2016. Clever people: Intelligence and humor production ability. *Psychology of Aesthetics, Creativity, and the Arts*, 12:136–143.

Christopher Clark, Kenton Lee, Ming-Wei Chang, et al. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL-HLT 2019*, pages 2924–2936. Association for Computational Linguistics.

Elizabeth Clark, Yangfeng Ji, and Noah A. Smith. 2018. Neural text generation in stories using entity representations as context. In *NAACL-HLT 2018*, pages 2250–2260. Association for Computational Linguistics.

Yufeng Diao, Hongfei Lin, Di Wu, et al. 2018. Weca : a wordnet-encoded collocation-attention network for homographic pun recognition. In *EMNLP 2018*, pages 2507–2516. Association for Computational Linguistics.

Giacomo Domeniconi, Marco Masseroli, Gianluca Moro, and Pietro Pinoli. 2014a. Discovering new gene functionalities from random perturbations of known gene ontological annotations. In *KDIR 2014 - Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, Rome, Italy, 21 - 24 October, 2014*, pages 107–116. SciTePress.

Giacomo Domeniconi, Marco Masseroli, Gianluca Moro, and Pietro Pinoli. 2016a. Cross-organism learning method to discover new gene functionalities. *Comput. Methods Programs Biomed.*, 126:20–34.

[4] https://www.maggioli.com/who-we-are/company-profile

Giacomo Domeniconi, Gianluca Moro, Andrea Pagliarani, and Roberto Pasolini. 2017. On deep learning in cross-domain sentiment classification. In *Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management - (Volume 1), Funchal, Madeira, Portugal, November 1-3, 2017*, pages 50–60. SciTePress.

Giacomo Domeniconi, Gianluca Moro, Roberto Pasolini, and Claudio Sartori. 2014b. Cross-domain text classification through iterative refining of target categories representations. In *KDIR 2014 - Proceedings of the International Conference on Knowledge Discovery and Information Retrieval, Rome, Italy, 21 - 24 October, 2014*, pages 31–42. SciTePress.

Giacomo Domeniconi, Konstantinos Semertzidis, Vanessa López, Elizabeth M. Daly, Spyros Kotoulas, and Gianluca Moro. 2016b. A novel method for unsupervised and supervised conversational message thread detection. In *DATA 2016 - Proceedings of 5th International Conference on Data Management Technologies and Applications, Lisbon, Portugal, 24-26 July, 2016*, pages 43–54. SciTePress.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, et al. 2024. The llama 3 herd of models. *CoRR*, abs/2407.21783.

Giacomo Frisoni, Alessio Cocchieri, Alex Presepi, Gianluca Moro, and Zaiqiao Meng. 2024. To generate or to retrieve? on the effectiveness of artificial contexts for medical open-domain question answering. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pages 9878–9919. Association for Computational Linguistics.

Giacomo Frisoni, Paolo Italiani, Stefano Salvatori, and Gianluca Moro. 2023. Cogito ergo summ: Abstractive summarization of biomedical papers via semantic parsing graphs and consistency rewards. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 12781–12789. AAAI Press.

He He, Nanyun Peng, and Percy Liang. 2019. Pun generation with surprise. In *NAACL-HLT*, pages 1734–1744. Association for Computational Linguistics.

Zachary Horvitz, Jingru Chen, Rahul Aditya, Harshvardhan Srivastava, Robert West, Zhou Yu, and Kathleen R. McKeown. 2024. Getting serious about humor: Crafting humor datasets with unfunny large language models. *CoRR*, abs/2403.00794.

Aaron Hurst, Adam Lerer, Adam P. Goucher, et al. 2024. Gpt-4o system card. *CoRR*, abs/2410.21276.

Paolo Italiani, Giacomo Frisoni, Gianluca Moro, Antonella Carbonaro, and Claudio Sartori. 2024. Evidence, my dear watson: Abstractive dialogue summarization on learnable relevant utterances. *Neurocomputing*, 572:127132.

Sophie F. Jentzsch and Kristian Kersting. 2023. Chatgpt is fun, but it is not funny! humor is still challenging large language models. In *Proceedings of the 13th Workshop on Computational Approaches to Subjectivity, Sentiment, & Social Media Analysis, WASSA@ACL 2023, Toronto, Canada, July 14, 2023*, pages 325–340. Association for Computational Linguistics.

Justine T. Kao, Roger Levy, and Noah D. Goodman. 2016. A computational model of linguistic humor in puns. *Cogn. Sci.*, 40(5):1270–1285.

Daniel Keysers, Nathanael Schärli, Nathan Scales, et al. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *ICLR 2020*. OpenReview.net.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, et al. 2019. Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*, 7:452–466.

Tristan Miller, Christian Hempelmann, and Iryna Gurevych. 2017. Semeval-2017 task 7: Detection and interpretation of english puns. In *SemEval@ACL 2017*, pages 58–68. Association for Computational Linguistics.

Anirudh Mittal, Yufei Tian, and Nanyun Peng. 2022. Ambipun: Generating humorous puns with ambiguous context. In *NAACL 2022*, pages 1053–1062. Association for Computational Linguistics.

Gianluca Moro, Andrea Pagliarani, Roberto Pasolini, and Claudio Sartori. 2018. Cross-domain & in-domain sentiment analysis with memory-based deep neural networks. In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, IC3K 2018, Volume 1: KDIR, Seville, Spain, September 18-20, 2018*, pages 125–136. SciTePress.

Gianluca Moro, Nicola Piscaglia, Luca Ragazzi, and Paolo Italiani. 2024a. Multi-language transfer learning for low-resource legal case summarization. *Artif. Intell. Law*, 32(4):1111–1139.

Gianluca Moro and Luca Ragazzi. 2022. Semantic self-segmentation for abstractive summarization of long documents in low-resource regimes. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 11085–11093. AAAI Press.

Gianluca Moro and Luca Ragazzi. 2023. Align-then-abstract representation learning for low-resource summarization. *Neurocomputing*, 548:126356.

Gianluca Moro, Luca Ragazzi, and Lorenzo Valgimigli. 2023a. Carburacy: Summarization models tuning and comparison in eco-sustainable regimes with a novel carbon-aware accuracy. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 14417–14425. AAAI Press.

Gianluca Moro, Luca Ragazzi, and Lorenzo Valgimigli. 2023b. Graph-based abstractive summarization of extracted essential knowledge for low-resource scenarios. In *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland - Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023)*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, pages 1747–1754. IOS Press.

Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, and Davide Freddi. 2022. Discriminative marginalized probabilistic neural method for multi-document summarization of medical literature. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 180–189. Association for Computational Linguistics.

Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, and Lorenzo Molfetta. 2023c. Retrieve-and-rank end-to-end summarization of biomedical studies. In *Similarity Search and Applications - 16th International Conference, SISAP 2023, A Coruña, Spain, October 9-11, 2023, Proceedings*, volume 14289 of *Lecture Notes in Computer Science*, pages 64–78. Springer.

Gianluca Moro, Luca Ragazzi, Lorenzo Valgimigli, Fabian Vincenzi, and Davide Freddi. 2024b. Revelio: Interpretable long-form question answering. In *The Second Tiny Papers Track at ICLR 2024, Tiny Papers @ ICLR 2024, Vienna, Austria, May 11, 2024*. OpenReview.net.

Luca Ragazzi, Paolo Italiani, Gianluca Moro, and Mattia Panni. 2024. What are you token about? differentiable perturbed top-k token selection for scientific document summarization. In *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 9427–9440. Association for Computational Linguistics.

Danilo Neves Ribeiro, Shen Wang, Xiaofei Ma, et al. 2023. STREET: A multi-task structured reasoning and explanation benchmark. In *ICLR 2023*. OpenReview.net.

Ekaterina Shutova and Lin Sun. 2013. Unsupervised metaphor identification using hierarchical graph factorization clustering. In *NAACL-HLT 2013*, pages 978–988. The Association for Computational Linguistics.

Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, et al. 2023. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Trans. Mach. Learn. Res.*, 2023.

Jiao Sun, Anjali Narayan-Chen, Shereen Oraby, et al. 2022a. Context-situated pun generation. In *EMNLP 2022*, pages 4635–4648. Association for Computational Linguistics.

Jiao Sun, Anjali Narayan-Chen, Shereen Oraby, et al. 2022b. Expunations: Augmenting puns with keywords and explanations. In *EMNLP 2022*, pages 4590–4605. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL-HLT 2019*, pages 4149–4158. Association for Computational Linguistics.

Yufei Tian, Divyanshu Sheth, and Nanyun Peng. 2022. A unified framework for pun generation with humor principles. In *EMNLP 2022*, pages 3253–3261. Association for Computational Linguistics.

Alex Wang, Amanpreet Singh, Julian Michael, et al. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *ICLR 2019*. OpenReview.net.

Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS 2022*.

Zhijun Xu, Siyu Yuan, Lingjie Chen, and Deqing Yang. 2024. "a good pun is its own reword": Can large language models understand puns? In *EMNLP 2024*, pages 11766–11782. Association for Computational Linguistics.

Weihao Yu, Zihang Jiang, Yanfei Dong, and Jiashi Feng. 2020a. Reclor: A reading comprehension dataset requiring logical reasoning. In *ICLR 2020*. OpenReview.net.

Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. 2018. A neural approach to pun generation. In *ACL 2018*, pages 1650–1660. Association for Computational Linguistics.

Zhiwei Yu, Hongyu Zang, and Xiaojun Wan. 2020b. Homophonic pun generation with lexically constrained rewriting. In *EMNLP 2020*, pages 2870–2876. Association for Computational Linguistics.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. 2023. Instruction-following evaluation for large language models. *ArXiv*, abs/2311.07911.

Yichao Zhou, Jyun-Yu Jiang, Jieyu Zhao, et al. 2020. "the boating store had its best sail ever": Pronunciation-attentive contextualized pun recognition. In *ACL 2020*, pages 813–822. Association for Computational Linguistics.

Yanyan Zou and Wei Lu. 2019. Joint detection and location of english puns. In *NAACL-HLT 2019*, pages 2117–2123. Association for Computational Linguistics.

A Prompts

Table 5 presents all prompts used in our experiments, covering both the pun-based tasks (above) and the LLM-as-a-judge evaluations for generation and comprehension (below).

B Error Analysis

Table 6 presents representative failure cases for both the Pun Generation and Pun Resolution tasks, organized into three distinct error types:

- **Non-prefix answers:** Here the model produces a valid English word that does not share the required morphological relationship with the prompt subject. For generation (left column), examples include "cavity" for a hollow cave (no prefix overlap) and "pane" for a pain/window pun (orthographic but not etymological). In resolution (right column), comparable mistakes include "super-nova" for a surprising star or "poll" for a controversial pole–both semantically plausible but failing to exploit the intended wordplay.
- **Derivative answers:** In these cases, the model correctly derives a new form of the prompt word (e.g. "booklet" from book), yet the resulting pun either lacks novelty or violates our morphological constraints. For instance, "joker" from joke is technically a derivative but does not inject a fresh pun sense; similarly, "billing" from bill repeats the original meaning rather than exploiting a homophonic twist.
- **Nonexistent forms:** Finally, some outputs are entirely unattested in English. In generation, models may hallucinate words like "sealappeal" or "beear", while in resolution they propose "caphate", "catipede", or "cat-block", none of which occur in standard usage. Such errors point to over-generalized suffix rules or spurious concatenations learned during training.

Together, these examples highlight common failure modes across state-of-the-art LLMs (e.g., GPT-4o, Gemini-2.0, LLaMA-3.3) when tasked with creative wordplay.

C Contamination

Figure 6 shows the prompt used to guide Gemini-1.5-Flash in detecting potential contamination of puns found on web pages. Table 7 lists all puns identified as contaminated.

D Few-Shot Examples

Table 8 shows the examples used in our experiments as in-context learning prompts. We deliberately chose a small number of demonstrations to mirror human behavior, where only a few examples are typically sufficient to grasp the task. Specifically, we used the first five examples across Resolution and Generation tasks (Comprehension is conducted in zero-shot settings). However, some models–such as Gemini–struggled to produce diverse outputs in the free pun generation task. To address this, we extended the pool of demonstrations to include more varied examples, while still using only five per prompt. This allowed us to rotate the examples across runs, helping to mitigate the lack of diversity and encourage more varied pun generation.

E Human Evaluation

We conducted a series of human evaluations to assess the performance of native English speakers and individuals fluent in English (C1 level or higher) on pun comprehension and resolution tasks. In total, we recruited 60 volunteers, each representing different backgrounds, including master's and PhD students, as well as employees from IT, finance, and economics sectors. To ensure diverse perspectives, participants were divided into three groups: 20 volunteers for comprehension, 20 for resolution, and 20 for generation tasks. To maintain consistency and minimize bias, all participants received the same instructions as the models. For each task, 30 instances were randomly sampled for each participant. The tasks were timed as follows: 20 minutes for pun comprehension, 30 minutes for pun resolution, and 60 minutes for pun generation. Results were evaluated based on the criteria outlined in Section 5.2.

Setting	Prompt
	Task Instruction
Comprehension (Coherent)	{pun} Did you get it? Possible answers: - Yes, for exactly two reasons. First, ... Second, ... - No, because ... Don't add further information.
Comprehension (Misleading)	{pun} Did you get it? Possible answers: - Yes, because [your explanation] - No, because [your explanation] Don't add further information.
Resolution	Examples: {few_shot_examples} New input: {pun} **Direct**: Answer by returning only one word as output, prefixed by "### Answer:". **CoT**: Answer by reasoning step by step, and eventually return your final answer (only one word) prefixed by "### Answer:".
Generation (Free)	Given a subject X, create an English pun using the format "What do you call a X that Y? XZ". Follow these guidelines: - Choose a prefix word X (the subject of the question). - Attach a suffix Z to X, forming a real word XZ (the punchline). - Ensure XZ's actual definition naturally replaces Y, making the joke logical. - Do not use compound words (e.g., dog → dogsitter, star → starlight) or derivatives of X (e.g., dog → doggy, rat → rats, pay → payment) as value of XZ. Example pun(s): {few_shot_examples} **Direct:** Answer by returning the new pun, prefixed by "### pun:". **CoT:** First, think step by step and eventually return the new pun. This must be your output format: ### rationale: {step by step reasoning} ### pun: {your new pun}. Don't add further information.
Generation (Constrained)	Given a subject X, create an English pun using the format "What do you call a X that Y? XZ". Follow these guidelines: - Attach a suffix Z to X, forming a real word XZ (the punchline). - Ensure XZ's actual definition naturally replaces Y, making the joke logical. - Do not use compound words (e.g., dog → dogsitter, star → starlight) or derivatives of X (e.g., dog → doggy, rat → rats, pay → payment) as value of XZ. Example pun(s): {few_shot_examples} New input: What do you call a X='{subject}' that Y? XZ. **Direct:** Answer by returning the final values of Y and XZ, prefixed by "### answer:". **CoT:** First, think step by step and eventually return the final values of Y and XZ. This must be your output format: ### rationale: step by step reasoning ### answer: Y='...', XZ='...' Don't add further information.
	Evaluation Instruction
Generation (step 1)	Determine whether the given word's meaning is derived from the provided root word. Examples: {few_shot_examples} New imput: Is "{answer}" derived from "{prefix}"? Explain briefly your decision and then answer with "yes" or "no" prefixed by "Answer:".
Generation (step 2)	Examples: {few_shot_examples} New input: Are "{predicate}" and "{answer}" semantically related? Explain briefly your decision and then answer with "yes" or "no" prefixed by "Answer:".
Comprehension (Coherent)	Determine whether the given explanations are equivalent. This means that the predicted answer should match both of the two reasons given by the gold answer. Gold explanation: {gold_rationale} Predicted explanation: {model_explanation} Question: Are the two explanations equivalent? Explain briefly your decision and then answer with "yes" or "no" prefixed by "Answer:".

Table 5: Overview of prompts used for Pun Comprehension, Resolution, and Generation, along with the one employed in generation evaluation process.

Model	Pun Generation (Free)	Pun Resolution
The answer does not use the subject as prefix		
GPT-4o	What do you call a cave that is hollow? cavity	What do you call a star that is surprising? supernova
Gemini-2.0-FT	What do you call a plane that is a kind of banana? plantain	What do you call a pole that is controversial? poll
Gemini-2.0-Flash	What do you call a pain that is a window? pane	What do you call a fan that is stylish? fashion
The answer is a derivative of the subject		
GPT-4o	What do you call a book that is small and concise? booklet	What do you call a bat that swims? batfish
Gemini-2.0-FT	What do you call a joke that is playful? joker	What do you call a win that occurs during the coldest season? winner
Gemini-2.0-Flash	what do you call a bill that is the act of invoicing? billing	What do you call a car that is a painful swelling under the skin? carseat
The answer does not exist		
GPT-4o	What do you call a seal that makes a request for reconsideration? sealappeal	What do you call a hat that is very disliked? caphate
LLaMA-3.3-70B	What do you call a bee that is a good listener? beear	What do you call a cat that crawls on multiple legs? catipede
GPT-4o-mini	What do you call a fish that is on a mission? fishion	What do you call a cat that blocks vision? catblock

Table 6: Examples of errors in the Pun Generation and Resolution tasks, organized into three error types. FT stands for Flash-Thinking. The text actually generated by the LLM is highlighted in violet.

This is an English pun:
What do you call a SUBJECT that {PREDICATE}? {ANSWER}. In this case we have the SUBJECT "{SUBJECT}", the PREDICATE "{PREDICATE}", and the ANSWER "{ANSWER}".

Your task is to determine whether the provided English pun has been contaminated by or exists within the content of a given web page.

Contamination is confirmed only if both of the following conditions are met:

- The web page contains sufficient information to reconstruct the question part of the pun, including EXPLICIT information on both the SUBJECT and the PREDICATE.
- The web page also provides explicitly the ANSWER to the pun.

If all the conditions are satisfied, explain why, otherwise return only the string "No evidence found."

Web Page Content: {content}

Figure 6: Prompt used to check contamination within a web page content by using Gemini-1.5-Flash.

Contaminated Puns
What do you call a can that is a deep valley? canyon
What do you call a can that is a way a horse runs? canter
What do you call a can that projects from a vertical support? canopy
What do you call a can that is truthful? candid
What do you call a can that is used to paint on? canvas
What do you call a cow that is scared? coward
What do you call a cat that causes disaster? catastrophe
What do you call a can that is also a bird? canary
What do you call a ant that is old-fashioned? antique
What do you call a can that gives light? candle
What do you call a cat that is in a state of immobility or stupor? catatonic
What do you call a can that is part of a state? canton
What do you call a can that crosses off? cancel
What do you call a can that is sweet? candy
What do you call a can that is savage? cannibal
What do you call a can that you can row in? canoe
What do you call a can that is an astronomical sign? cancer
What do you call a can that is a fruit? cantaloupe
What do you call a can that is a Chinese language? cantonese
What do you call a can that is country? canada

Table 7: List of contaminated puns found in the web.

#	Pun Question	Answer
1	What do you call a fan that plays an instrument?	fanfare
2	What do you call a cat that is clear and obvious?	categorical
3	What do you call a rat that is obsessed with stats?	ratio
4	What do you call a star that is served by a waiter?	starter
5	What do you call a man that does nails?	manicure
6	What do you call a bowl that throws balls?	bowler
7	What do you call a port that is part of a whole?	portion
8	What do you call a gene that works everywhere?	generalizable
9	What do you call a dog that is incontrovertibly true?	dogma
10	What do you call a trip that multiplies by three?	triple

Table 8: List of few-shot pun demonstrations.