

MECHPUNBENCH: A Bilingual Mechanism-Centered Benchmark for Pun Understanding and Generation

Tianwei Zhang^{1,2} Zihan Gu^{1,2,*} Jiayu Wang^{1,2}

Tianyi Ma³ Jiacheng Lu³ Jinyu Li^{1,2} Hua Zhang^{1,2,†}

¹Institute of Information Engineering, Chinese Academy of Sciences

²School of Cyber Security, University of Chinese Academy of Sciences

³Shanghai Jiao Tong University,

{zhangtianwei, guzihan, wangjiayu, zhanghua}@iie.ac.cn

Abstract

Puns pose a challenging test case for large language models (LLMs), as their interpretation requires mapping a surface expression to an implicit meaning through a specific pun-forming mechanism. Existing benchmarks typically evaluate isolated abilities such as detection, localization, explanation, or generation quality, leaving unclear whether LLMs can consistently identify, explain, and apply the mechanisms that make puns work. To address this gap, we introduce MECHPUNBENCH, a bilingual, mechanism-oriented benchmark for Chinese and English pun understanding and generation. Each positive instance is annotated with trigger spans, surface meanings, implicit meanings, rhetorical effects, and mechanism labels. Based on a taxonomy of Lexical–Semantic, Structural–Syntactic, and Structural–Semantic puns, we formulate six diagnostic tasks spanning surface recognition to mechanism-guided generation. Experiments on representative closed-source and open-source LLMs reveal a substantial surface–mechanism gap: although the best model achieves 82.8% detection accuracy and 88.2% containment localization accuracy, it remains below the 4/5 threshold used in our evaluation on implicit meaning explanation, mechanism explanation, rhetorical effect explanation, and mechanism-consistent generation. These findings suggest that recognizing puns or generating fluent pun-like text does not necessarily indicate robust mechanism-level pun understanding.

1 Introduction

Puns are a compact yet challenging form of language use: a surface expression evokes an additional implicit interpretation through a specific linguistic trigger (Coulson, 2001; Giora, 2003). Such effects may arise from lexical ambiguity, phonological similarity, syntactic reanalysis, or seman-

tic reinterpretation of larger expressions (Binsted and Ritchie, 1994; Hempelmann, 2003; Ritchie, 2004). Because verbal humor depends on modeling competing meanings and their triggering conditions, rather than surface amusement alone (Raskin, 1985; Attardo and Raskin, 1991; Attardo, 1994), puns provide a useful testbed for mechanism-level language understanding.

Although LLMs perform strongly on many language tasks (Brown et al., 2020; OpenAI, 2023), pun processing remains difficult. Prior work has studied pun detection, localization, interpretation, explanation, and generation (Miller et al., 2017; Sun et al., 2022; Xu et al., 2024); however, many evaluations focus on isolated abilities, such as detecting whether a sentence contains a pun, identifying a trigger span, or judging the fluency of generated pun-like text. These settings are useful, but they do not directly test whether models understand how a pun is formed. Recent studies suggest that LLMs may rely on pun-like surface patterns without fully capturing the underlying double meaning or formation mechanism (Xu et al., 2024; Zangari et al., 2025), consistent with broader concerns about shortcut learning (Geirhos et al., 2020). This raises our central question: can LLMs move beyond surface-level pun recognition and exhibit stable mechanism-sensitive behavior in explanation and generation?

To address this question, we introduce MECHPUNBENCH¹, a bilingual, mechanism-oriented benchmark for Chinese and English pun understanding and generation. As illustrated in Figure 1, our benchmark represents each pun as a structured relation among its surface form, implicit meaning, triggering mechanism, and rhetorical effect. This annotation design makes explicit what expression triggers the pun, what implicit meaning is evoked, which linguistic mechanism connects the two, and

*Project Leader

†Corresponding Author

¹<https://github.com/ZhangTianweo/MechPunBench>

(A) Benchmark Design

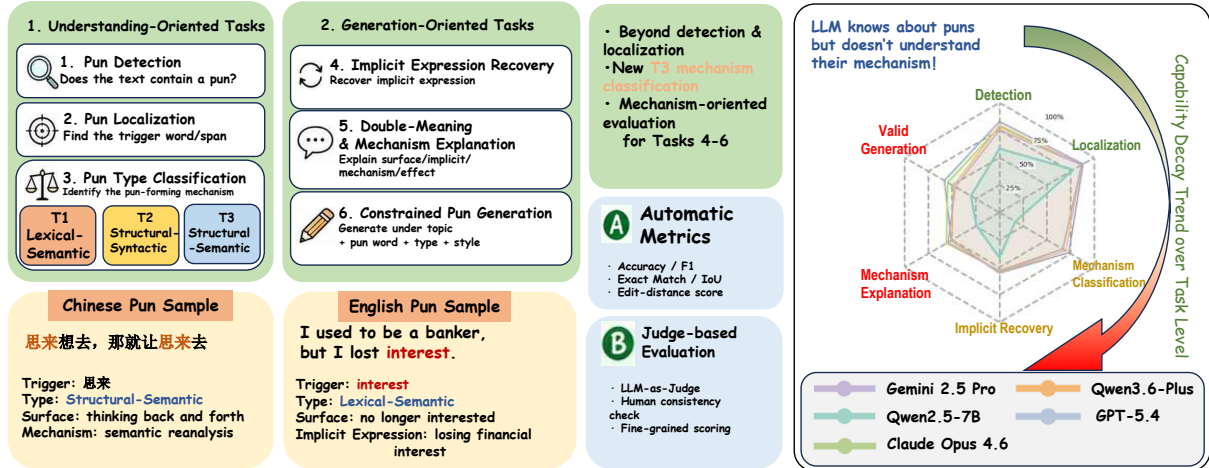


Figure 1: Overview of MECHPUNBENCH. (A) Six diagnostic tasks evaluate Chinese and English puns under three minimal direct licensing mechanisms. (B) Results reveal a recurring surface-mechanism pattern: strong surface recognition does not consistently translate into mechanism-sensitive explanation and generation.

what rhetorical effect the pun produces. Unlike evaluations that treat puns only as binary labels or final generated texts, MECHPUNBENCH enables direct assessment of whether models can recover, explain, and manipulate the mechanisms connecting what is said to what is implied.

We further organize puns into three mechanism types: Lexical-Semantic, Structural-Syntactic, and Structural-Semantic. These categories distinguish word-level ambiguity or similarity, alternative syntactic analyses, and reinterpretation at the level of larger expressions or compositional meanings. The taxonomy allows us to evaluate not only whether a model recognizes a pun, but also whether it can distinguish different ways in which double meanings are constructed. The bilingual setting further tests whether such mechanism-level behavior is robust across Chinese and English, two languages that differ substantially in writing systems, morphology, and common pun-forming resources.

Based on this annotation schema, we formulate six diagnostic tasks: pun detection, pun localization, pun type classification, implicit expression recovery, mechanism explanation, and constrained pun generation. These tasks form a progressive evaluation pipeline from surface recognition to mechanism recovery and mechanism application. Detection and localization test whether models can identify pun-like inputs and their triggering spans; classification, recovery, and explanation test whether models can infer and articulate the mechanisms behind double meanings; and constrained generation tests whether models can apply a speci-

fied mechanism to produce valid puns rather than merely fluent text. Together, these tasks provide a fine-grained evaluation of pun understanding and generation beyond surface-level success.

Experiments on representative closed-source and open-source LLMs reveal a clear surface-mechanism gap. The best model reaches 0.828 detection accuracy and 0.882 containment localization accuracy, suggesting that current LLMs can often identify pun-like inputs and locate their triggers. However, mechanism-sensitive tasks remain substantially harder: several strong models obtain Type-3 F1 scores of only 0.00–0.36 without CoT, explanation scores for implicit meaning, mechanism, and rhetorical effect generally fall below the predefined 4/5 usability threshold, and constrained generation achieves high language naturalness but much lower mechanism validity and dual-meaning clarity. These results show that surface recognition or fluent generation does not necessarily indicate stable mechanism-level pun understanding.

Overall, our work makes the following contributions:

- We construct a bilingual mechanism-centered pun dataset with structured annotations of triggers, meanings, mechanisms, rhetorical effects, and generation constraints.
- We operationalize pun-forming mechanisms through a mechanism-oriented taxonomy and six diagnostic tasks.
- We reveal a clear surface-mechanism gap in current LLMs through systematic evaluation.

2 Related Work

Pun Understanding and Benchmarks. Computational humor is challenging because it involves ambiguity, incongruity, pragmatic inference, and rhetorical intention (Ritchie, 2001; Mihalcea and Strapparava, 2005). Classic theories of verbal humor suggest that humor understanding requires more than recognizing amusing surface patterns (Raskin, 1985; Attardo and Raskin, 1991). Puns are a representative form of wordplay, as their effects are often grounded in identifiable triggers and mechanisms such as polysemy, phonological similarity, syntactic ambiguity, or semantic reanalysis (Binsted and Ritchie, 1994; Hempelmann, 2003; Ritchie, 2004). Early pun processing work mainly focused on detection, localization, and interpretation. SemEval-2017 Task 7 defined subtasks for detecting puns, locating punning words, and interpreting intended meanings (Miller et al., 2017), and participating systems explored knowledge-based or supervised methods for these subtasks (Mikhalkova and Karyakin, 2017). Later work further modeled detection and localization jointly (Zou and Lu, 2019). These studies provide important foundations, but they offer limited evidence about whether models can identify the mechanisms by which puns are formed.

Pun Explanation, Generation, and LLMs. Recent work extends pun processing to explanation and generation. ExPUNations augments pun data with keywords, explanations, and funniness ratings (Sun et al., 2022), while pun generation studies explore neural generation, humor principles, and preference learning (Yu et al., 2018; He et al., 2019; Tian et al., 2022; Chen et al., 2024). Recent studies have also examined LLMs on pun recognition, explanation, and generation. Xu et al. (2024) show that LLMs can identify familiar punning patterns but struggle with deeper explanation and faithful generation. Zangari et al. (2025) further argue that LLMs may respond to pun-like surface patterns even when the actual double meaning is removed, which is consistent with broader concerns about shortcut learning in neural models (Geirhos et al., 2020). These works show that pun processing requires more than detection or fluent generation, yet many evaluations still focus on post-hoc explanation quality, final fluency, funniness, or target-word inclusion, rather than on the mechanisms that connect triggers, meanings, and effects.

Multilingual Coverage and Evaluation. Most existing pun benchmarks are English-centered or focus on isolated tasks. Chinese resources such as ChinesePun support pun generation and evaluation (Chen et al., 2024), but existing Chinese or cross-lingual work often focuses on generation or translation rather than a unified diagnosis of understanding and generation. Open-ended pun explanation and generation also cannot be fully evaluated by string matching, motivating the use of LLM-as-Judge and rubric-based evaluation methods (Zheng et al., 2023; Liu et al., 2023). In contrast, our benchmark adopts a mechanism-centered annotation schema, covers both Chinese and English puns, and organizes evaluation as six progressive diagnostic tasks: detection, trigger localization, mechanism classification, implicit expression recovery, mechanism explanation, and constrained generation. This design enables a fine-grained analysis of whether LLMs can move from surface-level recognition to mechanism-sensitive explanation and generation.

3 Method

This section describes the construction of our benchmark. We first define the mechanism taxonomy used to assign pun types, then describe data sources, annotation fields, and quality control, and finally formulate the six diagnostic tasks derived from the annotated fields.

3.1 Pun Mechanism Taxonomy

To evaluate mechanism-sensitive behavior in pun processing, we introduce a three-way mechanism taxonomy of pun formation, motivated by prior accounts of ambiguity, script opposition, phonological similarity, and semantic reinterpretation in verbal humor and punning (Raskin, 1985; Attardo, 1994; Hempelmann, 2003; Ritchie, 2004). The taxonomy classifies each pun by its *minimal direct licensing mechanism*, namely the most direct linguistic operation that connects the surface expression with the implicit interpretation and without which the intended double meaning would not be licensed. Under this criterion, the three categories are mutually exclusive mechanism types rather than overlapping sets of surface cues: auxiliary cues may co-occur in a punning expression, but they determine the label only when they constitute the minimal direct mechanism that licenses the double meaning.

Type	Example	Explanation
Lexical–Semantic	<i>I used to be a banker, but I lost interest.</i>	<i>Interest</i> activates both personal interest and financial interest.
Structural–Syntactic	<i>One morning I shot an elephant in my pajamas.</i>	<i>In my pajamas</i> can attach to either the shooting event or <i>an elephant</i> .
Structural–Semantic	<i>The Titanic passengers had a sinking feeling.</i>	The idiom <i>sinking feeling</i> is reinterpreted through the literal sinking of the ship.

Table 1: Examples of the three mechanism types.

- **Lexical–Semantic Pun:** the double meaning is licensed by a localized lexical unit through ambiguity, homophony, or related lexical-semantic activation.
- **Structural–Syntactic Pun:** the punning effect is licensed by alternative syntactic analyses of the same surface form.
- **Structural–Semantic Pun:** the double meaning is licensed by semantic reinterpretation of a larger expression, idiom, fixed phrase, or construction under stable syntax.

This taxonomy is used as the label space for pun type classification, the basis for mechanism explanation, and the mechanism constraint for constrained pun generation. Detailed decision rules and additional examples are provided in Appendix A.

3.2 Data Construction and Annotation

Data Sources. We construct a bilingual dataset of 2,100 positive pun instances and 400 negatives, balanced by language (1,050 Chinese and 1,050 English) and mechanism type (700 instances per category). Part of the data is sampled from SemEval-2017 Task 7 (Miller et al., 2017) and ChinesePun (Chen et al., 2024), with the rest collected from diverse real-world sources. We reuse only original texts from existing resources and manually re-annotate all mechanism-oriented fields. The detection set uses 600 positives and 400 negatives, while the remaining 1,500 positives are used for SFT and mechanism-oriented distillation. Details are provided in Appendix C.

Filtering and Annotation. For pun detection, we construct hard negatives, including trigger-removed minimal pairs, topic-matched non-puns, and ambiguous or homophonic sentences without coherent double-meaning activation. For example, *The musician fixed the broken string before the concert* preserves a music-related context but does not

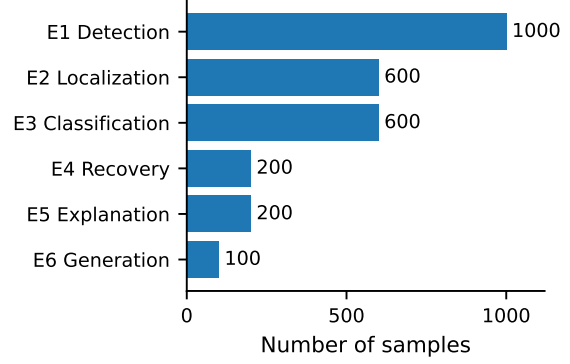


Figure 2: **Task-specific evaluation data scale.** Detection includes positive and hard negative samples; the other tasks use annotated positive puns.

activate both the literal meaning of *string* and an idiomatic meaning such as *string someone along*. Positive samples are annotated with trigger spans, mechanism types, surface meanings, implicit expressions, triggering mechanisms, and rhetorical effects. Two annotators label mechanism types, achieving Cohen’s $\kappa = 0.863$ (Cohen, 1960; Landis and Koch, 1977); disagreements are resolved by a third annotator.

Task-specific Data Scale. Figure 2 summarizes the task-specific evaluation sizes. Detection uses both positive and hard negative samples, while the other tasks use positive pun samples with task-specific annotations.

3.3 Task Formulation

Based on the mechanism-oriented annotation schema, we formulate six diagnostic tasks to evaluate models’ mechanism-sensitive behavior in pun understanding and generation. Following diagnostic evaluation principles (Ribeiro et al., 2020), these tasks are designed to target specific capabilities rather than only report aggregate benchmark performance. The tasks are divided into two groups. The understanding-oriented tasks include pun detection, pun localization, and pun type classification, which respectively evaluate whether models can recognize punning expressions, identify the trigger span, and distinguish the minimal direct licensing mechanism behind a pun. The generation-oriented tasks include implicit expression recovery, mechanism explanation, and constrained pun generation, which evaluate whether models can recover hidden meanings, explain how a pun works, and generate valid puns under explicit constraints.

Formally, given an input text x , pun detection

predicts a binary label $y \in \{0, 1\}$, where $y = 1$ indicates that x contains a pun and $y = 0$ indicates a non-punning text. For a positive pun sample x^+ , pun localization predicts the trigger span t , and pun type classification predicts a mechanism label $m \in \{\text{LEXSEM}, \text{STRSYN}, \text{STRSEM}\}$. For open-ended tasks, the model generates an implicit expression h , a structured explanation e , or a constrained punning sentence g conditioned on task-specific inputs. Each task is grounded in a specific annotation field, allowing the benchmark to diagnose models from surface recognition to mechanism-guided generation. Detailed task definitions are provided in Appendix B, and prompt templates with output formats are provided in Appendix E.

4 Experiments

4.1 Experimental Setup

We evaluate five representative LLMs on the six diagnostic tasks: GPT-5.4 (OpenAI, 2026b), Claude Opus 4.6 (Anthropic, 2026a), Gemini 2.5 Pro (Gemini Team, Google, 2025), Qwen3.6-Plus (Qwen Team, 2026), and Qwen2.5-7B (Qwen Team, 2024b,a). This selection covers stronger closed-source systems and a smaller open-source model under the same benchmark framework.

All models are evaluated with the same task-specific prompt formats. For discriminative tasks, including pun detection, localization, and type classification, we use deterministic decoding with temperature set to 0; for open-ended tasks, including implicit expression recovery, mechanism explanation, and constrained generation, we set the temperature to 0.7. CoT prompting is applied only to pun type classification, mechanism explanation, and constrained generation, where explicit reasoning may affect mechanism-level performance (Wei et al., 2022; Kojima et al., 2022). Detection, localization, and recovery are evaluated with direct-answer prompts.

We report task-specific metrics. Detection and type classification use accuracy and F1-based metrics; localization uses exact match, containment match, and IoU; implicit expression recovery uses edit-distance weighted scores and LLM-as-Judge scores over Top-1 and Top-5 candidates; mechanism explanation and constrained generation use rubric-based judge scores. For additional fine-tuning analysis, we compare standard SFT (Ouyang et al., 2022) and mechanism-oriented distillation (Hinton et al., 2015) on Qwen2.5-7B for

pun type classification.

For open-ended tasks, GPT-5.4-mini is used as the primary judge to score all model outputs according to task-specific rubrics, following recent LLM-as-Judge evaluation practice (Zheng et al., 2023; Liu et al., 2023). To assess reliability, we conduct sampled reference evaluation with human annotators and Claude Sonnet 4.6 on the same subset of outputs and compute Spearman correlations (Spearman, 1904) among their scores. Model identities are anonymized during judging to reduce potential model-identity bias.

4.2 Understanding-oriented Experiments

The understanding-oriented experiments examine whether models can move from pun detection to pun localization and mechanism discrimination.

4.2.1 Pun Detection

Model	Acc.	Prec.	Rec.	F1	BAcc.
GPT-5.4	0.828	0.822	0.910	0.864	0.807
Qwen3.6-Plus	0.794	0.842	0.810	0.826	0.789
Claude Opus 4.6	0.781	0.816	0.816	0.816	0.772
Gemini 2.5 Pro	0.753	0.752	0.878	0.810	0.722
Qwen2.5-7B	0.582	0.677	0.580	0.625	0.583

Table 2: Results on the pun detection task. BAcc. denotes balanced accuracy.

As shown in Table 2, strong models perform reasonably well on pun detection, with GPT-5.4 achieving the best accuracy and balanced accuracy. Since the detection set contains 600 positive and 400 negative samples, the majority-class baseline is 60.0%; thus, balanced accuracy is also reported to account for class imbalance (He and Garcia, 2009). Qwen2.5-7B falls below the majority baseline in raw accuracy and obtains only 0.583 BAcc., suggesting that smaller models are less reliable at distinguishing puns from hard non-pun instances. Overall, detection is not the main bottleneck for strong LLMs, but it only tests surface-level recognition. Full detection results over all evaluated models and language-specific breakdowns are provided in Appendix F.

4.2.2 Pun Localization

To avoid trivial containment, full-sentence or excessively long predictions are marked invalid. As shown in Table 3, closed-source models achieve high Cont. scores but substantially lower EM scores, indicating that they can often identify the approximate trigger region while still producing coarse span boundaries. Qwen2.5-7B performs

Model	EM	Cont.	IoU
GPT-5.4	0.660	0.882	0.800
Qwen3.6-Plus	0.658	0.867	0.778
Claude Opus 4.6	0.695	0.867	0.802
Gemini 2.5 Pro	0.665	0.865	0.789
Qwen2.5-7B	0.442	0.780	0.603

Table 3: Results on the pun localization task. EM, Cont., and IoU denote exact match, containment match, and character-level intersection-over-union, respectively.

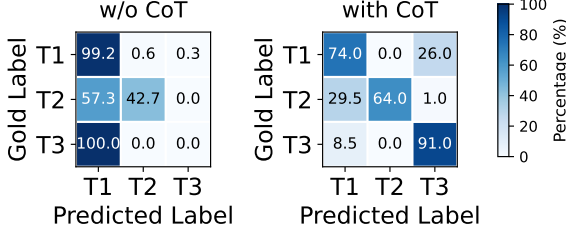


Figure 3: **Qwen3.6-Plus Type Confusion.** Row-normalized confusion matrices (%) of Qwen3.6-Plus on pun type classification. Without CoT, Type-3 cases collapse into Type-1; with CoT, predictions become more balanced.

notably worse, especially in IoU. Overall, pun localization is easier than pun type classification, but trigger identification is not fully precise. Detailed matching rules and full results are provided in Appendix G.

4.2.3 Pun Type Classification

Model	Acc.	T1	T2	T3
GPT-5.4	.75/.82	.81/.73	.89/.93	.36/.81
Claude Opus 4.6	.75/.77	.82/.67	.91/.88	.08/.80
Qwen3.6-Plus	.67/.76	.78/.70	.59/.78	.00/.84
Gemini 2.5 Pro	.70/.70	.78/.67	.78/.84	.26/.76
Qwen2.5-7B	.47/.50	.65/.49	.07/.17	.21/.69
Qwen2.5-7B-Distill	.76	.62	.79	.85
Qwen2.5-7B-SFT	.56	.70	.28	.40

Table 4: Mechanism-oriented pun type classification results. Values before/after the slash denote results without/with CoT. T1–T3 denote F1 scores for Type1–Type3 puns. Fine-tuned variants are evaluated under a single inference setting.

Pun type classification directly evaluates whether models can move from recognizing pun-like cues to distinguishing the mechanism that licenses the double meaning. As shown in Table 4, direct prompting exposes a strong mechanism-level category bias. The most salient failure occurs on Type-3 puns: Qwen3.6-Plus obtains 0.00 Type-3 F1 without CoT, and Claude Opus 4.6 obtains only 0.08. This is not simply a general accuracy drop. Type-3 puns require models to recognize

expression- or construction-level semantic reinterpretation under relatively stable syntax, which is less locally marked than lexical ambiguity or syntactic reanalysis. The low Type-3 scores therefore suggest that, when no explicit reasoning is required, models tend to reduce pun type classification to more salient local cues.

The confusion matrix of Qwen3.6-Plus provides a representative view of this failure mode. As shown in Figure 3, without CoT, Qwen3.6-Plus maps almost all Type-3 instances to Type-1, and many Type-2 instances are also absorbed into Type-1. This indicates a lexical-semantic shortcut: the model tends to search for a salient word or phrase that can be associated with two meanings, even when the intended pun is licensed by a larger expression or structural relation. In other words, the main error is not random confusion among three labels, but a systematic collapse from higher-level mechanisms into the lexical-semantic category.

CoT guidance substantially changes this pattern. Type-3 F1 improves from 0.00 to 0.84 for Qwen3.6-Plus and from 0.08 to 0.80 for Claude Opus 4.6, suggesting that explicit step-by-step comparison helps models attend to the distinction between lexical ambiguity, syntactic reanalysis, and expression-level semantic reinterpretation. However, this improvement should be interpreted cautiously. The Qwen3.6-Plus heatmap also shows that CoT shifts the decision boundary rather than simply eliminating errors: some Type-1 cases are now predicted as Type-3. Thus, CoT appears to make mechanism definitions more salient and encourages rule-like comparison, but it does not prove that models have fully internalized a stable mechanism taxonomy.

Prompt Sensitivity. To examine whether classification performance is primarily driven by detailed mechanism definitions or structured output scaffolding, we conduct a 2×2 ablation on GPT-5.4. The experiment crosses full mechanism definitions versus label-only descriptions with full JSON versus minimal label outputs, allowing us to isolate the effects of ontology guidance and output formatting. All other settings, including the 600 evaluation samples, sample construction, decoding parameters, and number of runs, are held fixed. As shown in Table 5, replacing full JSON with minimal outputs produces only small changes under either definition condition. With full definitions, minimal output reduces accuracy from 0.745 to 0.738 and Macro-F1 from 0.687 to 0.678. Under

Setting	Acc.	M-F1	T1	T2	T3
Full def. + Full JSON	.745	.687	.809	.894	.359
Full def. + Minimal	.738	.678	.803	.887	.344
Label only + Full JSON	.718	.657	.794	.874	.303
Label only + Minimal	.710	.647	.788	.866	.287

Table 5: Prompt-sensitivity results for GPT-5.4 on pun type classification. M-F1 denotes Macro-F1, and T1–T3 denote type-specific F1 scores.

label-only descriptions, the corresponding changes are similarly small, from 0.718 to 0.710 in accuracy and from 0.657 to 0.647 in Macro-F1. In contrast, removing the detailed definitions produces a somewhat larger, though still moderate, performance reduction under both output formats.

When both forms of scaffolding are removed, accuracy decreases from 0.745 to 0.710 and Macro-F1 decreases from 0.687 to 0.647, corresponding to drops of 3.5 and 4.0 percentage points, respectively. Importantly, the qualitative error pattern remains stable across all four conditions. Type-1 and Type-2 performance remains substantially higher than Type-3 performance, while Type-3 F1 ranges only from 0.287 to 0.359. Moreover, Type-3 instances continue to be predominantly misclassified as Type-1, indicating that the lexical-semantic shortcut is not an artifact of a particular prompt format. These results suggest that detailed definitions provide a modest performance scaffold by making the distinctions among the three mechanism types more explicit, whereas JSON formatting mainly standardizes output parsing. Neither factor, individually or jointly, accounts for the persistent difficulty of distinguishing expression-level semantic reinterpretation from localized lexical ambiguity.

Effect of Mechanism-oriented Distillation. As a preliminary fine-tuning analysis, we compare standard SFT and mechanism-oriented distillation on Qwen2.5-7B under the same classification setting. As shown in Table 4, standard SFT yields only limited improvement and still leaves clear category imbalance, especially on mechanism types that require non-lexical reasoning. In contrast, the distillation variant achieves higher overall accuracy and more balanced Type-2 and Type-3 F1 scores. This suggests that label-only SFT is insufficient to correct the observed category bias, whereas teacher signals that expose mechanism-oriented reasoning provide more useful supervision for type discrimination. Since this analysis is limited to one base model and one classification task, we treat it as

suggestive rather than conclusive. Overall, pun type classification shows that current LLMs struggle to maintain stable mechanism-level distinctions, while explicit reasoning and mechanism-oriented supervision can partially alleviate this issue. Detailed results are provided in Appendix H.

Language-level Observation. Because MECH-PUNBENCH covers both Chinese and English, we further inspect language-specific results in the understanding-oriented tasks. We do not observe a consistent Chinese–English performance gap across detection, localization, or type classification; instead, the more stable pattern is a task-level surface–mechanism gap. Full language-specific results are provided in Appendices F, G, and H.

4.3 Generation-oriented Experiments

The generation-oriented experiments evaluate whether models can reconstruct and apply pun mechanisms through implicit expression recovery, mechanism explanation, and constrained pun generation.

For open-ended tasks, we employ GPT-5.4-mini (OpenAI, 2026a) as the primary evaluator for large-scale assessment, following recent LLM-as-Judge evaluation practice (Zheng et al., 2023; Liu et al., 2023). Detailed scoring rubrics for the three open-ended tasks are provided in Appendix D. These rubrics define a 0-to-5 scale with explicit semantic anchors, thereby treating scores as semantically meaningful intervals rather than arbitrary numerical values. In consistency checks, we observed that the model evaluator is more lenient in assessing generation quality than human annotators. We therefore designate a score of 4—anchored to human-perceived semantic adequacy—as the operational threshold, representing the minimum level at which an output satisfies the primary task requirements and is deemed usable without substantial revision. Accordingly, we adopt scores ≥ 4 as the criterion-referenced benchmark for acceptable performance. To assess reliability, we randomly sample outputs for supplementary evaluation by both human annotators and Claude Sonnet 4.6 (Anthropic, 2026b) under anonymized conditions. As reported in Table 6, Spearman rank correlations (Spearman, 1904) among the three evaluators remain consistently high across all tasks, indicating that the automated scores provide stable rank-order reference signals. Additional human binary acceptability checks for the score-4 threshold are pro-

vided in Appendix D.

Task	n	GPT-Hum. ρ	GPT-Cl. ρ	Hum.-Cl. ρ
Recovery	200	0.83	0.87	0.82
Explanation	200	0.86	0.84	0.89
Generation	100	0.84	0.81	0.85

Table 6: Rank-level consistency among GPT-5.4-mini, human annotators, and Claude Sonnet 4.6 on sampled outputs. ρ denotes Spearman correlation.

4.3.1 Implicit Expression Recovery

Model	T1 Edit	T5 Edit	T1 Judge	T5 Judge
GPT-5.4	0.364	0.501	3.705	4.180
Qwen3.6-Plus	0.344	0.484	3.750	4.235
Claude Opus 4.6	0.316	0.437	3.710	4.065
Gemini 2.5 Pro	0.337	0.482	3.700	4.190
Qwen2.5-7B	0.168	0.229	3.280	4.220

Table 7: Results on the implicit expression recovery task. T1/T5 denote Top-1/Top-5 candidate outputs; Edit and Judge denote edit-distance weighted scores and LLM-as-Judge scores, respectively. Bold values indicate scores of 4 or above.

Implicit expression recovery evaluates whether models can infer the hidden semantic expression behind a pun. Models output Top-1 to Top-5 candidates, which are evaluated with two complementary metrics: an edit-distance weighted score for surface-form closeness to the annotated expression, and a judge-based score for semantic recovery. The former is sensitive to paraphrases, wording differences, and specificity mismatches, while the latter assesses whether the intended hidden meaning is captured even without lexical identity. Following the judge rubric, scores of 4 or above are treated as usable recovery results.

As shown in Table 7, edit-distance scores remain low, especially for Top-1 outputs, indicating that recovered expressions often differ in form from the gold annotation. Judge scores are much higher, particularly for Top-5, because at least one candidate often captures the intended meaning through a paraphrase, broader concept, or related expression. This suggests that models can often approach the latent semantic space of the pun, but still struggle to produce the exact intended expression and rank it first. Full recovery results and detailed scoring rubrics are provided in Appendices I and D, respectively.

4.3.2 Mechanism Explanation

Mechanism explanation evaluates whether models can explain a pun across four rubric dimen-

Model	Surface	Implicit	Mechanism	Effect
Claude Opus 4.6	3.20/ 4.02	2.70/3.44	2.82/3.58	2.90/3.63
Gemini 2.5 Pro	3.18/3.92	2.53/3.13	2.82/3.35	2.82/3.51
GPT-5.4	2.83/ 4.04	2.48/3.30	2.69/3.62	2.49/3.66
Qwen3.6-Plus	2.92/3.78	2.41/2.95	2.62/3.23	2.53/3.34
Qwen2.5-7B	2.43/3.21	1.30/1.91	1.34/1.68	1.01/1.49

Table 8: Results on the mechanism explanation task. Values before/after the slash denote results without/with CoT. Surface, Implicit, Mechanism, and Effect denote the four rubric dimensions; bold values indicate scores of 4 or above.

sions: surface meaning, implicit meaning, triggering mechanism, and rhetorical effect. In the judge rubric, a score of 4 or above is predefined as a usable explanation, meaning that the output is accurate, clear, and sufficiently informative for the corresponding dimension. As shown in Table 8, CoT improves scores across most dimensions, but the improvement is uneven. Models can explain surface meanings relatively well, with the strongest CoT scores reaching the usable threshold. However, implicit meaning, mechanism explanation, and rhetorical effect remain below 4 for all models. This shows that current LLMs can often paraphrase what is explicitly expressed, but still struggle to provide stable mechanism-sensitive explanations of how the pun works. Full explanation results are provided in Appendix J, and detailed scoring rubrics are provided in Appendix D.

4.3.3 Constrained Pun Generation

Model	Mech.	Dual	Lang.	Creat.
GPT-5.4	2.85/3.72	2.94/3.05	4.72/4.14	2.45/1.96
Qwen3.6-Plus	2.77/3.21	2.73/2.58	4.75/4.34	2.40/1.47
Claude Opus 4.6	2.70/3.52	2.64/2.86	4.70/4.29	2.20/1.82
Gemini 2.5 Pro	2.11/2.82	2.06/2.48	4.69/4.69	1.68/1.68
Qwen2.5-7B	1.47/2.03	1.34/1.29	4.67/3.86	1.22/1.21

Table 9: Constrained pun generation results. Values are reported as w/o CoT / with CoT. Mech., Dual, Lang., and Creat. denote mechanism validity, dual-meaning clarity, language naturalness, and creativity.

Constrained pun generation evaluates whether models can generate valid puns under constraints such as topic, target word, mechanism type, and style, following controllable text generation (Keskar et al., 2019). Outputs are scored on mechanism validity, dual-meaning clarity, language naturalness, and creativity. In the judge rubric, scores of 4 or above are predefined as basically usable;

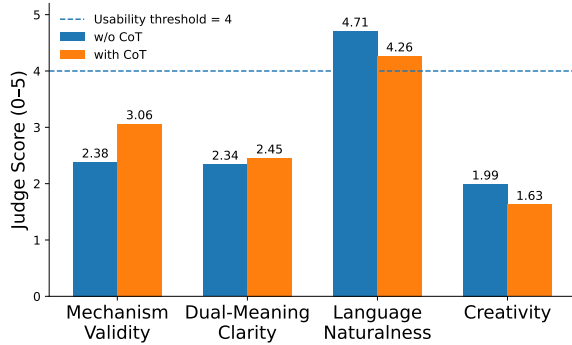


Figure 4: **Fluency–Mechanism Gap in Generation.** Average constrained generation scores across models. Fluency remains high, while mechanism validity, dual-meaning clarity, and creativity stay below the usability threshold.

Full generation results are provided in Appendix K; prompts and scoring rubrics are provided in Appendices E and D, respectively.

As shown in Table 9 and Figure 4, models produce fluent sentences, with language naturalness often reaching the usable threshold. However, mechanism validity, dual-meaning clarity, and creativity remain much lower, revealing a clear fluency–mechanism gap. CoT improves mechanism validity for all models and dual-meaning clarity for some, but often reduces language naturalness and creativity, suggesting that explicit reasoning makes generation more mechanism-oriented but also more conservative.

A parallel 2×2 prompt ablation shows that the same fluency–mechanism pattern persists across mechanism-definition and output-format conditions. Removing detailed definitions causes a modest decline, whereas simplifying the JSON output has only a small effect; full results are reported in Table 28.

Performance also varies by target mechanism type: GPT-5.4 obtains lower mechanism-validity and dual-meaning scores for T3 than for T1, while naturalness remains stable across types; detailed results and significance tests are provided in Table 27.

Case Study. Qualitative inspection reveals two recurring failures in constrained generation: *weak semantic grounding*, where a surface association fails to support a coherent second meaning, and *target drift*, where the pun moves from the required target word to another salient word.

In a representative GPT-5.4 case with *topic* = *travel and transport*, *target type* = *Lexical–Semantic*, *pun word* = *harried*, and *style* = *short*

natural one-line, the non-CoT output is “*The airport made me so harried, I nearly got married to my suitcase,*” while the CoT output is “*The subway delays harried me so much, I arrived both late and mentally derailed.*” The former suggests *harried/married* but lacks a coherent second meaning licensed by *harried*; the latter shifts the actual pun to *derailed*. Both outputs remain fluent and topic-related, explaining why language naturalness can be high while mechanism validity and dual-meaning clarity remain low.

Robustness to Sampling Variability. To assess whether the main findings are sensitive to sampling variability in the open-ended evaluation sets, we report percentile 95% confidence intervals from 10,000 instance-level bootstrap resamples stratified by language and mechanism type (random seed: 42). For GPT-5.4, the Top-1 and Best-of-5 recovery judge scores are 3.705 [3.57, 3.84] and 4.180 [4.08, 4.28], respectively; the CoT mechanism-explanation score is 3.619 [3.49, 3.75]; and the without-CoT generation scores are 2.85 [2.64, 3.06] for mechanism validity and 4.72 [4.61, 4.82] for language naturalness. These intervals support the conclusions drawn from the point estimates: considering multiple candidates increases recovery coverage, mechanism explanation remains below the usability threshold, and generation exhibits substantially higher naturalness than mechanism validity.

5 Conclusion

We introduced MECHPUNBENCH, a bilingual mechanism-centered benchmark with structured annotations, a three-way mechanism taxonomy, and six diagnostic tasks. Experiments reveal a clear surface–mechanism gap: strong LLMs can detect puns and roughly locate triggers, but remain unstable in mechanism classification, recovery, explanation, and constrained generation. These findings suggest that current LLMs can recognize or fluently produce pun-like text, but still struggle to reliably recover, explain, and apply the mechanisms that make puns work.

Limitations

This work has several limitations. The benchmark covers only Chinese and English puns with a limited data scale, so the findings may not generalize to other languages or culturally specific forms of

wordplay. The mechanism taxonomy is an operational annotation schema, and some multi-cued puns may still involve subjective labeling decisions despite the minimal direct licensing criterion, inter-annotator agreement, and adjudication. Open-ended tasks rely partly on rubric-based LLM-as-Judge evaluation, which may introduce bias and may not fully capture humor, creativity, or pragmatic appropriateness (Zheng et al., 2023; Liu et al., 2023). Results on closed-source LLMs may also be affected by model updates and API changes. Therefore, our findings should be interpreted as a diagnostic snapshot of observable task performance rather than direct claims about the internal cognitive states of LLMs (Bender and Koller, 2020).

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant 62372448.

References

- Anthropic. 2026a. Introducing Claude Opus 4.6. <https://www.anthropic.com/news/claude-opus-4-6>. Accessed: 2026-05-25.
- Anthropic. 2026b. Introducing Claude Sonnet 4.6. <https://www.anthropic.com/news/claude-sonnet-4-6>. Accessed: 2026-05-25.
- Salvatore Attardo. 1994. *Linguistic Theories of Humor*. Mouton de Gruyter, Berlin and New York.
- Salvatore Attardo and Victor Raskin. 1991. Script theory revis(it)ed: Joke similarity and joke representation model. *HUMOR: International Journal of Humor Research*, 4(3–4):293–347.
- Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198. Association for Computational Linguistics.
- Kim Binsted and Graeme Ritchie. 1994. An implemented model of punning riddles. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI)*, pages 633–638.
- Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.
- Yang Chen, Chong Yang, Tu Hu, Xinhao Chen, Man Lan, Li Cai, Xinlin Zhuang, Xuan Lin, Xin Lu, and Aimin Zhou. 2024. Are U a joke master? pun generation via multi-stage curriculum learning towards a humor LLM. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 878–890, Bangkok, Thailand. Association for Computational Linguistics.
- Jacob Cohen. 1960. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20(1):37–46.
- Seana Coulson. 2001. *Semantic Leaps: Frame-Shifting and Conceptual Blending in Meaning Construction*. Cambridge University Press, Cambridge.
- Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. 2020. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2(11):665–673.
- Gemini Team, Google. 2025. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*.
- Rachel Giora. 2003. *On Our Mind: Salience, Context, and Figurative Language*. Oxford University Press, Oxford.
- Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.
- He He, Nanyun Peng, and Percy Liang. 2019. Pun generation with surprise. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1734–1744, Minneapolis, Minnesota. Association for Computational Linguistics.
- Christian F. Hempelmann. 2003. *Paronomasic Puns: Target Recoverability Towards Automatic Generation*. Ph.D. thesis, Purdue University.
- Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.
- Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.
- Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, volume 35, pages 22199–22213.

- J. Richard Landis and Gary G. Koch. 1977. [The measurement of observer agreement for categorical data](#). *Biometrics*, 33(1):159–174.
- Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. [G-eval: NLG evaluation using GPT-4 with better human alignment](#). In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.
- Rada Mihalcea and Carlo Strapparava. 2005. Making computers laugh: Investigations in automatic humor recognition. In *Proceedings of the Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 531–538. Association for Computational Linguistics.
- Elena Mikhalkova and Yuri Karyakin. 2017. PunFields at SemEval-2017 task 7: Employing Roget’s thesaurus in automatic pun recognition and interpretation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 426–431. Association for Computational Linguistics.
- Tristan Miller, Christian Hempelmann, and Iryna Gurevych. 2017. [SemEval-2017 task 7: Detection and interpretation of English puns](#). In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 58–68, Vancouver, Canada. Association for Computational Linguistics.
- OpenAI. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.
- OpenAI. 2026a. GPT-5.4 mini model. <https://developers.openai.com/api/docs/models/gpt-5.4-mini>. OpenAI API documentation. Accessed: 2026-05-25.
- OpenAI. 2026b. Introducing GPT-5.4. <https://openai.com/index/introducing-gpt-5-4/>. Accessed: 2026-05-25.
- Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744.
- Qwen Team. 2024a. Qwen2.5-7B. <https://huggingface.co/Qwen/Qwen2.5-7B>. Accessed: 2026-05-25.
- Qwen Team. 2024b. [Qwen2.5 technical report](#). *arXiv preprint arXiv:2412.15115*.
- Qwen Team. 2026. Qwen3.6-Plus: Towards real world agents. <https://qwen.ai/blog?id=qwen3.6>. Accessed: 2026-05-25.
- Victor Raskin. 1985. *Semantic Mechanisms of Humor*, volume 24 of *Studies in Linguistics and Philosophy*. D. Reidel Publishing Company, Dordrecht, Netherlands.
- Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. [Beyond accuracy: Behavioral testing of NLP models with CheckList](#). In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912. Association for Computational Linguistics.
- Graeme Ritchie. 2001. [Current directions in computational humour](#). *Artificial Intelligence Review*, 16(2):119–135.
- Graeme Ritchie. 2004. *The Linguistic Analysis of Jokes*. Routledge, London.
- Charles Spearman. 1904. [The proof and measurement of association between two things](#). *The American Journal of Psychology*, 15(1):72–101.
- Jiao Sun, Anjali Narayan-Chen, Shereen Oraby, Alessandra Cervone, Tagyoung Chung, Jing Huang, Yang Liu, and Nanyun Peng. 2022. [ExPUNations: Augmenting puns with keywords and explanations](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4590–4605, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Yufei Tian, Divyanshu Sheth, and Nanyun Peng. 2022. [A unified framework for pun generation with humor principles](#). In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3253–3261, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837.
- Zhijun Xu, Siyu Yuan, Lingjie Chen, and Deqing Yang. 2024. [“a good pun is its own reward”: Can large language models understand puns?](#) In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 11766–11782, Miami, Florida, USA. Association for Computational Linguistics.
- Zhiwei Yu, Jiwei Tan, and Xiaojun Wan. 2018. [A neural approach to pun generation](#). In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1650–1660, Melbourne, Australia. Association for Computational Linguistics.
- Alessandro Zangari, Matteo Marcuzzo, Andrea Albarelli, Mohammad Taher Pilehvar, and Jose Camacho-Collados. 2025. [Pun unintended: LLMs and the illusion of humor understanding](#). In *Proceedings of the 2025 Conference on Empirical Methods in*

Natural Language Processing, pages 27936–27971, Suzhou, China. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. [Judging LLM-as-a-judge with MT-bench and chatbot arena](#). In *Advances in Neural Information Processing Systems*, volume 36.

Yanyan Zou and Wei Lu. 2019. [Joint detection and location of English puns](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2117–2123, Minneapolis, Minnesota. Association for Computational Linguistics.

A Taxonomy Details and Examples

This appendix provides additional explanations and examples for the mechanism-oriented taxonomy introduced in Section 3.1. The key criterion is the *minimal direct licensing mechanism*: the most direct linguistic operation that licenses the intended double meaning of a pun. The taxonomy does not classify a pun by all linguistic cues that appear in the sentence, but by the mechanism without which the punning effect would not be established.

A.1 Definition and Decision Rules

Our taxonomy classifies each pun according to the *minimal direct licensing mechanism* that makes its double meaning possible. The classification target is not the set of all linguistic cues appearing in the sentence, but the single most direct linguistic operation without which the punning effect would not be licensed. Under this criterion, each valid pun instance receives exactly one mechanism label. Instances whose double meaning cannot be reduced to a sufficiently clear minimal direct mechanism are excluded during annotation.

A *Lexical–Semantic Pun* is licensed by the dual semantic activation of a localized lexical unit, such as a word, morpheme, character, or short lexical expression. The crucial condition is that the localized unit itself directly supports two interpretations through polysemy, homonymy, homophony, lexical association, or domain-specific meaning shift. The double meaning must be established without requiring an alternative syntactic parse or the reinterpretation of a larger expression as a conventional construction.

A *Structural–Syntactic Pun* is licensed by an alternative syntactic analysis of the same surface

form. The crucial condition is that the double meaning arises from changes in constituent boundaries, attachment relations, part-of-speech assignment, segmentation, punctuation, or dependency structure. The pun is not labeled as Lexical–Semantic if the competing interpretation depends on a different structural organization rather than on the ambiguity of a single localized lexical item. It is not labeled as Structural–Semantic if the effect is produced by syntactic reanalysis rather than by semantic reinterpretation under a stable syntactic structure.

A *Structural–Semantic Pun* is licensed by the semantic reinterpretation of a larger expression, idiom, fixed phrase, sentence pattern, or construction under a stable syntactic analysis. The crucial condition is that the expression has a conventional or contextually expected meaning, but is reoriented in a new semantic frame to produce an additional interpretation. The pun is not labeled as Lexical–Semantic merely because one word in the expression is semantically salient; the classification depends on whether the larger expression, rather than an isolated lexical unit, directly licenses the double meaning. It is not labeled as Structural–Syntactic because no alternative syntactic parse is required.

In short, the three labels are mutually exclusive because the minimal direct licensing mechanism is located at one of three distinct levels: localized lexical meaning, syntactic structure, or expression-level semantic reinterpretation. Auxiliary cues may co-occur, but they do not determine the label unless they constitute the minimal direct mechanism that licenses the double meaning.

A.2 Illustrative Chinese Examples

We provide additional Chinese examples to illustrate how the same taxonomy can be applied across languages.

Lexical–Semantic Pun. In 东边日出西边雨，道是无晴却有晴, the pun is licensed by the homophonic relation between 晴 and 情. On the surface, 晴 refers to clear weather, forming a contrast with 雨. At the implicit level, however, 晴 evokes 情, meaning affection or romantic feeling. The double meaning is therefore directly licensed by a localized lexical unit, making this a Lexical–Semantic Pun.

Structural–Syntactic Pun. In 下雨天留客天留我不留, the punning effect is licensed by alternative segmentation and punctuation. The same surface string can be parsed in different ways, such as

下雨天，留客天。留我不？留！， which means that the rainy day is suitable for keeping the guest and the speaker should stay. With a different segmentation, however, the string can also be interpreted as 下雨天，留客天，留我不留， where the final part expresses uncertainty or refusal. The crucial mechanism is not the ambiguity of a single word, but the alternative syntactic organization of the same string. Therefore, this instance is classified as a Structural–Syntactic Pun.

Structural–Semantic Pun. In 思来想去，那就让思来去， the expression 思来想去 conventionally means thinking repeatedly. The second clause reinterprets 思来 and 去 as semantically manipulable components, as if 思来 could be made to “go.” The syntactic form remains stable, and the effect does not rely on a single lexical ambiguity. Instead, the pun is licensed by reinterpreting a larger conventional expression and its internal semantic relations, making it a Structural–Semantic Pun.

A.3 Annotation Decision Rules

When assigning a mechanism label, annotators follow the minimal direct licensing mechanism criterion. The following decision rules are used:

1. If the double meaning is directly licensed by a localized word, morpheme, character, or short lexical expression through polysemy, homonymy, homophony, or lexical association, the instance is labeled as LEXSEM.
2. If the double meaning is directly licensed by an alternative syntactic analysis of the same surface form, including changes in attachment, constituent boundary, segmentation, part-of-speech assignment, or dependency relation, the instance is labeled as STRSYN.
3. If the double meaning is directly licensed by the semantic reinterpretation of a larger expression, idiom, fixed phrase, sentence pattern, or construction under stable syntax, the instance is labeled as STRSEM.

If an instance contains multiple auxiliary cues, annotators do not label all cues. Instead, they identify the mechanism without which the intended double meaning would not be licensed. For example, a sentence may contain both a salient word association and a fixed expression, but if the punning effect depends on reinterpreting the fixed expression as a

whole, it is labeled as Structural–Semantic rather than Lexical–Semantic.

A.4 Boundary Cases

Some puns contain multiple surface cues, such as salient words, contextual hints, homophony, idiomatic expressions, or structural ambiguity. We still assign a single label according to the minimal direct licensing mechanism: the most direct linguistic operation without which the intended double meaning would not be licensed. The following examples illustrate how this criterion is applied in boundary cases.

Lexical–Semantic cases. *I used to be a banker, but I lost interest.* This example contains a banking context, but the double meaning is directly licensed by the localized lexical item *interest*, which activates both the psychological meaning of interest and the financial meaning of interest. Since no alternative syntactic parse or larger-expression reinterpretation is needed, it is classified as Lexical–Semantic.

东边日出西边雨，道是无晴却有晴。 The surface reading concerns weather, where 晴 means clear weather. The implicit reading is licensed by the homophonic relation between 晴 and 情, evoking affection or romantic feeling. Although the surrounding poetic contrast helps the interpretation, the minimal direct mechanism is the localized homophonic lexical unit, so this is Lexical–Semantic.

The fisherman was hooked on his new hobby. This case contains the expression *hooked on*, which might appear construction-like. However, the punning effect is directly licensed by *hooked*, which activates both the literal fishing-related meaning and the figurative meaning of being strongly interested. The larger phrase provides context, but the minimal direct mechanism is lexical-semantic activation.

莲子心中苦，梨儿腹内酸。 The punning effect is licensed by the homophonic relations between 莲 and 怜, and between 梨 and 离. The surface reading describes the bitterness of lotus seeds and the sourness of pears, while the implicit reading evokes emotional suffering caused by pity and separation. The double meaning is triggered by localized homophonic lexical units, so the case is classified as Lexical–Semantic.

Structural–Syntactic cases. *Time flies like an arrow; fruit flies like a banana.* Although the words

flies and *like* are salient, the pun is licensed by alternative syntactic and part-of-speech analyses of the repeated surface pattern. In the first clause, *flies* functions as a verb and *like* as a preposition; in the second, *fruit flies* is a noun phrase and *like* functions as a verb. The crucial mechanism is syntactic reanalysis, so it is Structural–Syntactic.

下雨天留客天留我不留。 The same unpunctuated string can be segmented and punctuated differently, yielding different pragmatic interpretations about whether the guest should stay. For example, 下雨天, 留客天。留我不? 留! supports staying, while a different grouping can express refusal or uncertainty. The double meaning is licensed by alternative segmentation and syntactic organization, not by one word carrying two meanings.

Visiting relatives can be boring. The ambiguity arises from two syntactic analyses: *visiting relatives* can mean relatives who are visiting, or the act of visiting relatives. The word *visiting* is salient, but the double reading depends on whether the phrase is analyzed as a noun phrase or a gerundive construction. Therefore, the minimal direct mechanism is structural-syntactic.

I saw the man with the telescope. The ambiguity is licensed by alternative attachment of the prepositional phrase *with the telescope*. Under one parse, the speaker used a telescope to see the man; under another, the man had the telescope. Since the double meaning depends on syntactic attachment rather than lexical ambiguity, this case is Structural–Syntactic.

Structural–Semantic cases. *The Titanic passengers had a sinking feeling.* The word *sinking* is salient, but the pun is not classified as Lexical–Semantic because the humorous effect depends on reinterpreting the larger idiomatic expression *sinking feeling* in the Titanic context. The syntactic structure remains stable, and the minimal direct mechanism is expression-level semantic reinterpretation.

思来想去, 那就让思来去。 The fixed expression 思来想去 conventionally means thinking repeatedly. The second clause reinterprets 思来 and 去 as semantically manipulable components, as if 思来 could be made to “go.” The pun is licensed by reinterpreting a larger expression rather than by localized lexical ambiguity alone.

The broken elevator had its ups and downs. The expression *ups and downs* conventionally refers to alternating good and bad experiences. In the ele-

vator context, the same expression is reinterpreted as literal upward and downward movement. Since the syntax remains stable and the effect depends on the reinterpretation of the larger expression, this is Structural–Semantic.

The bakery had a lot of dough to rise. The expression combines the business sense of having money with the baking frame in which dough literally rises. The pun is not licensed by a single lexical item alone, but by reinterpreting the larger expression *dough to rise* within the bakery context. The syntax remains stable, so the case is Structural–Semantic.

These cases show that the taxonomy does not classify puns by all cues present in the sentence. A sentence may contain salient words, contextual hints, idioms, or phonological associations, but the label is determined only by the minimal direct mechanism that licenses the double meaning. This keeps the three categories mutually exclusive while still allowing real puns to contain auxiliary cues from multiple linguistic levels.

B Task Details

This appendix provides detailed definitions of the six diagnostic tasks introduced in Section 3.3. Each task is derived from one or more fields in the mechanism-oriented annotation schema.

B.1 Understanding-oriented Tasks

Pun Detection. Pun detection is formulated as a binary classification task. Given an input text x , the model is required to determine whether it contains a pun. The output is a binary label $y \in \{0, 1\}$, where $y = 1$ indicates that the text contains a pun and $y = 0$ indicates a non-punning text. This task is derived from the positive and negative sample labels in the dataset and evaluates models’ surface-level sensitivity to punning expressions.

Pun Localization. Pun localization is conducted on positive pun samples. Given a text x^+ known to contain a pun, the model is required to output the word or span t that triggers the punning effect. This task corresponds to the trigger-span annotation. It evaluates whether models can identify the linguistic unit responsible for activating the double meaning, rather than merely recognizing that a pun exists.

Pun Type Classification. Pun type classification is also conducted on positive samples. Given a punning text x^+ , the model must classify it into one of

the three mechanism types: LEXSEM, STRSYN, or STRSEM. This task corresponds to the mechanism-type annotation and evaluates whether models can identify the minimal direct licensing mechanism by which the punning effect is produced.

B.2 Generation-oriented Tasks

Implicit Expression Recovery. Implicit expression recovery asks the model to recover the implicit expression behind a pun. Given a positive pun sample x^+ , the model outputs one or more candidate implicit expressions h . This task corresponds to the implicit-expression annotation and evaluates whether models can infer the hidden semantic target that is not directly expressed on the surface but is necessary for the punning effect.

Mechanism Explanation. Mechanism explanation requires the model to provide a structured explanation of the pun. Given a positive pun sample, the model is asked to explain its surface meaning, implicit meaning, triggering mechanism, and rhetorical effect. This task corresponds to the surface-meaning, implicit-expression, triggering-mechanism, and rhetorical-effect fields in the annotation schema. It evaluates whether models can explicitly articulate why the pun works and how its double meaning is produced.

Constrained Pun Generation. Constrained pun generation evaluates whether models can actively apply pun-forming mechanisms in generation. Given constraints such as a topic, a target pun word, a target mechanism type, and a style requirement, the model is required to generate a sentence that forms a valid pun. This task is derived from the controllable generation fields in the dataset and evaluates whether models can produce mechanism-valid punning expressions under explicit semantic and stylistic constraints.

C Data Sources and Annotation Guidelines

C.1 Data Sources and Composition

Our benchmark is constructed from both existing pun resources and newly collected punning expressions. The positive pool contains 2,100 pun instances, balanced across languages with 1,050 Chinese and 1,050 English puns. Part of the data is sampled from SemEval-2017 Task 7 (Miller et al., 2017) and ChinesePun (Chen et al., 2024), while the remaining samples are collected from diverse

real-world sources. From existing resources, we use only the original punning texts and do not reuse their original task labels or auxiliary annotations. All fields required by our benchmark are manually re-annotated under our mechanism-centered schema.

To increase source diversity and reduce dependence on a single benchmark source, we collect candidate punning expressions from online joke and wordplay collections, advertising slogans, film and television lines, social media posts and comments, literary or idiomatic expressions, classroom and everyday conversational examples, and manually collected bilingual pun examples. Table 10 summarizes the source composition of the positive pun pool. Candidate samples are retained only when they contain a recoverable double meaning, an identifiable pun trigger, and a mechanism that can be assigned under our taxonomy.

In addition to positive pun instances, we construct 400 hard negative samples for the pun detection task. These negative samples are designed to reduce source and style artifacts and to test whether models can distinguish genuine double-meaning activation from ordinary ambiguity, humor, or wordplay-like surface cues. Table 11 summarizes the composition of the hard negative set.

Data and Code Availability. MECHPUNBENCH, including the benchmark annotations, data splits, task prompts, evaluation scripts, and documentation, is publicly available at <https://github.com/ZhangTianweo/MechPunBench>. We release the newly created annotations and accompanying code under their specified licenses. Texts originating from existing datasets or third-party sources remain subject to their original licenses and terms of use; where redistribution is not permitted, we provide source identifiers or links together with our derived annotations.

The retained positive samples are reorganized into evaluation and training subsets. For benchmark evaluation, 600 positive samples are used together with 400 hard negatives for pun detection; the same 600 positive samples support trigger localization and mechanism type classification, with smaller subsets used for implicit expression recovery, mechanism explanation, and constrained generation. The remaining 1,500 positive samples are used for the SFT and mechanism-oriented distillation experiments.

Source	Chinese		English	
	Candidates	Retained	Candidates	Retained
SemEval-2017 Task 7	0	0	243	200
ChinesePun	216	200	0	0
Online joke/wordplay collections	221	151	236	158
Advertising slogans	146	112	128	95
Film and television lines	132	94	151	107
Social media posts/comments	238	158	247	165
Literary or idiomatic expressions	176	127	104	76
Classroom/everyday conversations	113	78	101	71
Manually collected bilingual examples	171	130	258	178
Total	1413	1050	1468	1050

Table 10: Source composition of the positive pun pool. Candidates denotes initially collected candidate texts, and Retained denotes samples kept after manual filtering and mechanism-centered re-annotation.

Negative type	Chinese	English	Total
Trigger-removed minimal pairs	73	77	150
Source/topic-matched non-puns	68	64	132
Ambiguous or homophonic non-puns	59	59	118
Total	200	200	400

Table 11: Composition of hard negative samples for pun existence detection.

C.2 Filtering and Hard Negative Construction

To ensure that the pun detection task does not rely on superficial source or style artifacts, we construct hard negative samples in addition to ordinary non-punning texts. A negative sample is retained only if it is fluent, contextually natural, and does not license a stable double meaning under our pun definition. In particular, we exclude sentences that contain an unintended pun, require excessive world knowledge to disambiguate, or remain ambiguous after adjudication.

We use three main types of hard negatives. First, *trigger-removed minimal pairs* are constructed by replacing or rewriting the pun trigger in a positive sample while preserving the topic and sentence style. For example, the English pun *I used to be a banker, but I lost interest* can be rewritten as *I used to be a banker, but I changed careers*, where the financial/psychological ambiguity of *interest* is removed. Similarly, the Chinese pun 东边日出西边雨，道是无晴却有晴 can be rewritten as 东边日出西边雨，天气变化难分明, where the homophonic relation between 晴 and 情 is removed.

Second, *source- or topic-matched non-puns* are collected from similar domains, such as advertisements, social media comments, film lines, everyday conversations, or humorous texts, but are retained only when they do not contain a recoverable double meaning. For example, *The train was delayed again, so everyone arrived late* is travel-related and may sound like a complaint, but it does not contain a punning trigger. Likewise, 今天下雨，路上很堵，大家都迟到了 is a natural everyday sentence about rain and traffic, but it does not activate an implicit meaning through a shared trigger.

Third, we include *ambiguous or homophonic non-puns*, where a word or expression could in principle be ambiguous but the context does not support coherent double-meaning activation. For example, *The bank is closed on Sunday* contains the ambiguous word *bank*, but the sentence only supports the financial-institution reading and does not form a pun. Similarly, 今天的天气很晴朗 contains 晴, which is homophonic with 情, but the context only supports the weather meaning and does not license a punning interpretation.

During filtering, annotators check whether each candidate negative contains an identifiable trigger, whether a surface meaning and an implicit meaning can both be recovered, and whether the two meanings are connected by the same lexical item, syntactic structure, or larger expression. If a coherent double meaning can be recovered, the instance is removed from the negative set or relabeled as a positive pun candidate. If annotators disagree

Subset	Language	Mechanism	Source	Domain	Count
Evaluation positives	Zh	T1/T2/T3	Mixed sources	Mixed domains	300
Evaluation positives	En	T1/T2/T3	Mixed sources	Mixed domains	300
Training positives	Zh	T1/T2/T3	Mixed sources	Mixed domains	750
Training positives	En	T1/T2/T3	Mixed sources	Mixed domains	750
Detection negatives	Zh	N/A	Minimal pairs / matched non-puns / ambiguous non-puns	Mixed domains	200
Detection negatives	En	N/A	Minimal pairs / matched non-puns / ambiguous non-puns	Mixed domains	200
Total positives	Zh	Balanced over T1/T2/T3	Mixed sources	Mixed domains	1,050
Total positives	En	Balanced over T1/T2/T3	Mixed sources	Mixed domains	1,050
Total negatives	Zh/En	N/A	Hard negative construction	Mixed domains	400

Table 12: Dataset composition by language, mechanism, source type, task split, and domain. T1, T2, and T3 denote Lexical–Semantic, Structural–Syntactic, and Structural–Semantic puns, respectively.

about whether the sentence contains a pun, the case is sent to a third annotator; unresolved cases are discarded.

C.3 Positive Sample Annotation

Each positive pun sample is annotated with a set of mechanism-oriented fields. The core fields include the original text, language, trigger span, mechanism type, surface meaning, implicit expression, triggering mechanism, and rhetorical effect. For samples used in constrained generation, additional fields are annotated, including topic, target pun word, target mechanism type, and style constraint.

Trigger Span. The trigger span refers to the minimal word, phrase, character sequence, or surface string that directly activates the punning effect. Annotators are instructed to mark the shortest span that is necessary for licensing the double meaning. If a longer phrase contains the trigger but the punning effect is carried by a smaller unit, the smaller unit is preferred.

Mechanism Type. The mechanism type is assigned according to the minimal direct licensing mechanism defined in Section 3.1. If the double meaning is directly licensed by a localized lexical unit, the sample is labeled as LEXSEM. If it is licensed by an alternative syntactic analysis, it is labeled as STRSYN. If it is licensed by the semantic reinterpretation of a larger expression or construction under stable syntax, it is labeled as STRSEM.

Surface Meaning and Implicit Expression. The surface meaning describes the interpretation directly supported by the literal or conventional reading of the text. The implicit expression records the hidden meaning, concept, or expression that is not explicitly stated but is necessary for the punning effect. Annotators are asked to make the implicit

expression concise and semantically explicit.

Triggering Mechanism and Rhetorical Effect.

The triggering mechanism explains how the trigger connects the surface meaning with the implicit expression. The rhetorical effect describes the expressive function of the pun, such as humor, irony, emphasis, surprise, or stylistic playfulness.

C.4 Quality Control

The annotation is conducted by two annotators. Each annotator first determines whether the sample contains a valid pun, then identifies the trigger span and assigns a mechanism type. For positive samples, annotators further provide or verify the surface meaning, implicit expression, triggering mechanism, and rhetorical effect. We compute Cohen’s kappa for mechanism-type annotation, obtaining a score of 0.863. Disagreements are resolved by a third annotator, and unresolved cases are discarded.

D Scoring Rubrics

For open-ended tasks, we use task-specific 0–5 rubrics. A score of 4 or above is predefined as a usable output, indicating that the response satisfies the target requirement with only minor omissions or weaknesses. Scores are assigned independently for each evaluated dimension.

D.1 Human Binary Acceptability Check

To validate the operational threshold used in the judge-based evaluation, we further conduct a sampled human binary acceptability check for the three open-ended tasks. Human annotators are asked to judge whether each output is acceptable for the corresponding task requirement. We then compare human acceptability labels with the criterion-referenced judge decision, where scores ≥ 4 are

Task	Human Acc.@< 4	Human Acc.@≥ 4	Agreement	κ
Recovery	0.027	0.963	0.970	0.925
Explanation	0.037	0.947	0.960	0.875
Generation	0.053	1.000	0.960	0.896

Table 13: Human binary acceptability validation of the score-4 threshold. Human Acc.@< 4 and Human Acc.@≥ 4 denote the proportion of outputs judged acceptable by humans among outputs receiving judge scores below 4 and at least 4, respectively. Each task uses 100 sampled outputs.

treated as acceptable and scores < 4 as unacceptable. As shown in Table 13, outputs scored ≥ 4 by the rubric-based judge are more likely to be accepted by humans, supporting the use of 4 as the usability threshold.

D.2 Implicit Expression Recovery Rubric

Implicit expression recovery evaluates whether the model output captures the intended hidden expression or latent meaning behind the pun. For Top-1 evaluation, only the first candidate is scored. For Top-5 evaluation, the candidate list is evaluated according to whether at least one candidate successfully recovers the intended implicit expression.

Score	Definition
5	The output exactly or nearly exactly recovers the intended implicit expression, with no substantive semantic error.
4	The output correctly captures the intended implicit meaning, with only minor wording differences, omissions, or level-of-specificity differences.
3	The output is semantically related to the intended implicit expression, but is incomplete, under-specified, or partially misaligned.
2	The output captures only a vague association or a peripheral clue, but fails to recover the intended implicit meaning.
1	The output is weakly related to the pun context but does not identify the hidden expression.
0	The output is irrelevant, empty, invalid, or contradicts the intended implicit meaning.

Table 14: Scoring rubric for implicit expression recovery.

D.3 Mechanism Explanation Rubric

Mechanism explanation is evaluated along four dimensions: surface meaning, implicit meaning, triggering mechanism, and rhetorical effect. Each dimension receives an independent 0–5 score.

Surface Meaning. This dimension evaluates whether the model correctly explains the literal

or contextually explicit meaning of the punning text.

Implicit Meaning. This dimension evaluates whether the model correctly identifies the hidden meaning, latent expression, or second interpretation activated by the pun.

Triggering Mechanism. This dimension evaluates whether the model explains how the pun is produced, including the trigger span and the mechanism that connects the surface meaning with the implicit meaning.

Rhetorical Effect. This dimension evaluates whether the model explains the humorous, ironic, emphatic, surprising, or stylistic effect created by the pun.

Score	Definition
5	The explanation is accurate, complete, and clearly articulated for the target dimension, with no substantive error.
4	The explanation is mostly accurate and usable, with only minor omissions, simplifications, or wording issues.
3	The explanation is partially correct but incomplete, vague, or missing an important component of the target dimension.
2	The explanation captures only a weak or peripheral aspect and fails to explain the target dimension adequately.
1	The explanation is mostly incorrect, only loosely related to the pun, or confuses the relevant meaning or mechanism.
0	The explanation is irrelevant, empty, invalid, or completely contradicts the pun interpretation.

Table 15: General 0–5 scoring rubric for each dimension in mechanism explanation.

D.4 Constrained Pun Generation Rubric

Constrained pun generation is evaluated along four independent 0–5 dimensions: *mechanism validity*, *dual-meaning clarity*, *language naturalness*, and *creativity*. Mechanism validity measures whether the output follows the target mechanism type and uses the target pun word or structure to license a genuine punning effect. Dual-meaning clarity measures whether both the surface and implicit meanings are clearly supported and connected through the intended trigger. Language naturalness measures fluency, grammaticality, and consistency with the topic and style constraint. Creativity measures whether the pun is novel, engaging, and rhetorically effective rather than generic, forced, or mechanical.

Score	Definition
5	The output fully satisfies the dimension, with a strong, clear, and effective punning construction.
4	The output is basically usable for the dimension, with only minor weaknesses or imperfections.
3	The output partially satisfies the dimension, but the pun is somewhat weak, unclear, forced, or incomplete.
2	The output shows a limited attempt, but the target dimension is poorly satisfied or only marginally present.
1	The output is mostly unsuccessful for the dimension, with serious problems in mechanism, meaning, fluency, or creativity.
0	The output is invalid, irrelevant, non-punning, empty, or violates the given constraints completely.

Table 16: General 0–5 scoring rubric for each dimension in constrained pun generation.

E Task Prompts

This appendix presents the task-specific prompt templates used in our experiments. Figures 5–10 show the prompts for pun detection, localization, type classification, implicit expression recovery, mechanism explanation, and constrained pun generation, respectively.

Detection Prompt

System:
You are a bilingual pun detection assistant for both Chinese and English.
...
User:
Determine whether the following text contains a pun.
Return JSON only.
text: {text}
...

Figure 5: Prompt template for pun detection.

Localization Prompt

LOCALIZATION_SYSTEM_PROMPT = ""You are a bilingual pun trigger localization assistant for Chinese and English.
Your task is to identify the minimal surface span that directly triggers the punning effect in a text known to contain a pun.
The trigger span should be the shortest necessary expression that activates the double meaning. For lexical-semantic puns, it is usually a word or short phrase. For structural-syntactic puns, it is the smallest span responsible for alternative segmentation, attachment, or syntactic analysis. For structural-semantic puns, it may be a whole idiom, fixed expression, or construction rather than a single word.
Do not output the full sentence unless the whole sentence is an irreducible trigger. Return exactly one valid JSON object and nothing else.
Output schema:
{
 "trigger_span": "string",
 "trigger_type_hint": "lexical | syntactic | structural_semantic | null",
 "confidence": 0.0
}
""
LOCALIZATION_USER_PROMPT = ""The following text is known to contain a pun. Identify the minimal pun trigger span. Return JSON only.
text: {text}
""

Figure 6: Prompt template for pun localization.

Classification Prompt with CoT

CLASSIFICATION_COT_SYSTEM_PROMPT = ""You are a bilingual pun mechanism classification assistant for Chinese and English.
Given a text known to contain a pun, classify its minimal direct licensing mechanism. You may reason privately, but the final answer must be exactly one valid JSON object and must not include the reasoning process.
Target labels:
1 = lexical-semantic pun: the double meaning is licensed by a localized word, character, morpheme, phrase, or surface form, such as through polysemy, homonymy, homophony, near-homophony, or lexical association.
2 = structural-syntactic pun: the double meaning is licensed by alternative syntactic analyses of the same surface string, such as different segmentation, attachment, part-of-speech assignment, punctuation, or constituent structure.
3 = structural-semantic pun: the double meaning is licensed by semantic reinterpretation of a larger expression, idiom, fixed phrase, collocation, sentence pattern, or construction under stable syntax.
Classification rule:
Choose the single mechanism without which the pun would not work. Do not classify by all possible cues in the sentence. If a whole idiom or fixed expression is reinterpreted, do not label it as lexical-semantic merely because one word inside it is salient.
Use private reasoning to compare the three labels, then output JSON only.
Output schema:
{
 "target_type_high": 1,
 "target_type_label": "LEXSEM",
 "trigger_span": "string or null",
 "hidden_meaning_hint": "string or null",
 "confidence": 0.0
}
""

Figure 7: Prompt template for pun type classification with CoT.

Recovery Prompt

RECOVERY_SYSTEM_PROMPT = ""You are a bilingual implicit expression recovery assistant for Chinese and English puns. Given a text containing a pun, recover the implicit expression or hidden meaning activated by the pun trigger. The implicit expression is the unstated secondary meaning that supports the double meaning of the pun. It should not be a paraphrase of the whole sentence or merely the surface meaning. For lexical-semantic puns, it is usually the alternative meaning of the trigger. For structural-syntactic puns, it may come from an alternative segmentation, attachment, part-of-speech assignment, or syntactic analysis. For structural-semantic puns, it may come from the reinterpretation of an idiom, fixed expression, collocation, or construction.
Return five concise candidates ranked from most likely to least likely. Return exactly one valid JSON object and nothing else.
Output schema:
{
 "top1_implicit_expression": "string or null",
 "candidates": [{
 "rank": 1,
 "implicit_expression": "string or null"
 }, {
 "rank": 2,
 "implicit_expression": "string or null"
 }, {
 "rank": 3,
 "implicit_expression": "string or null"
 }, {
 "rank": 4,
 "implicit_expression": "string or null"
 }, {
 "rank": 5,
 "implicit_expression": "string or null"
 }
]
""

Figure 8: Prompt template for implicit expression recovery.

Explanation Prompt

EXPLANATION_SYSTEM_PROMPT = ""You are a bilingual pun mechanism explanation assistant for Chinese and English.
Given a text containing a pun, explain how the pun works in terms of four dimensions: surface meaning, implicit meaning, triggering mechanism, and rhetorical effect.
The surface meaning is the literal, conventional, or explicit contextual meaning of the text. The implicit meaning is the hidden or secondary meaning activated by the pun. The triggering mechanism explains how the surface expression connects the two meanings, such as through lexical ambiguity, homophony, syntactic reanalysis, or semantic reinterpretation of a fixed expression or construction. The rhetorical effect describes the humor, irony, cleverness, contrast, or semantic surprise produced by the double meaning.
Return exactly one valid JSON object and nothing else.
Output schema:
{
 "surface_meaning": "string or null",
 "implicit_meaning": "string or null",
 "triggering_mechanism": "string or null",
 "rhetorical_effect": "string or null",
 "trigger_span": "string or null",
 "target_type_high": 1,
 "target_type_label": "LEXSEM"
}
""

Figure 9: Prompt template for mechanism explanation.

Generation Prompt with CoT

GENERATION_COT_SYSTEM_PROMPT = ""You are a controlled pun generation assistant for Chinese and English. Given a set of control conditions, generate one original punning sentence. Privately plan the surface meaning, hidden meaning, and mechanism before answering, but do not reveal the reasoning process.
The sentence should satisfy the specified language, topic, target mechanism type, pun word, and style constraint whenever possible. Use the specified pun_word as the main pun trigger, or a very close inflected, homophonic, near-homophonic, segmented, or orthographically related form if necessary.
Target mechanism types:
1 = lexical-semantic pun: the same word, phrase, or surface form evokes two meanings.
2 = structural-syntactic pun: the pun depends on alternative segmentation, pausing, punctuation, attachment, part-of-speech assignment, or syntactic reanalysis.
3 = structural-semantic pun: a whole expression, idiom, collocation, fixed phrase, sentence pattern, or construction carries both a conventional/global meaning and a literal/compositional meaning.
Privately self-check that the output is original, short, natural, uses the intended trigger, and matches the target mechanism type. Return exactly one valid JSON object and nothing else.
Output schema:
{
 "generated_text": "string",
 "pun_word_used": "string or null",
 "target_type_high": 1,
 "intended_surface_meaning": "string or null",
 "intended_hidden_form": "string or null",
 "mechanism_hint": "string or null",
 "brief_explanation": "string or null"
}
""

Figure 10: Prompt template for constrained pun generation with CoT.

F Full Results for Pun Existence Detection

This appendix reports the full results for the pun existence detection task. Table 17 presents the overall results across all evaluated models, and Table 18 reports the language-specific results.

Model	Acc.	Prec.	Rec.	F1	Spec.	BAcc.	AUC
GPT-5.4	0.828	0.822	0.910	0.864	0.704	0.807	0.731
Qwen3.6-Plus	0.794	0.842	0.810	0.826	0.769	0.789	0.820
GPT-5.4-mini	0.784	0.893	0.727	0.801	0.870	0.798	0.785
Claude Opus 4.6	0.781	0.816	0.816	0.816	0.728	0.772	0.698
Claude Sonnet 4.6	0.756	0.790	0.805	0.798	0.683	0.744	0.772
Gemini 2.5 Pro	0.753	0.752	0.878	0.810	0.565	0.722	0.671
Gemini 2.5 Flash	0.750	0.771	0.832	0.800	0.627	0.729	0.776
Qwen3.5-9B	0.710	0.761	0.753	0.757	0.645	0.699	0.684
Mistral-7B	0.628	0.737	0.592	0.656	0.683	0.637	0.824
Gemma 3 12B	0.593	0.646	0.713	0.678	0.412	0.563	0.669
Qwen2.5-7B	0.582	0.677	0.580	0.625	0.585	0.583	0.589
Llama 3.1	0.459	0.537	0.720	0.615	0.068	0.394	0.652

Table 17: Overall results on the pun existence detection task. Spec., BAcc., and AUC denote specificity, balanced accuracy, and ROC-AUC, respectively.

Model	Lang.	Acc.	Prec.	Rec.	F1	BAcc.	AUC
GPT-5.4	Zh	0.86	0.86	0.91	0.89	0.85	0.68
Qwen3.6-Plus	Zh	0.81	0.87	0.80	0.84	0.81	0.83
GPT-5.4-mini	Zh	0.79	0.93	0.70	0.80	0.81	0.74
Claude Opus 4.6	Zh	0.80	0.79	0.90	0.84	0.77	0.74
Claude Sonnet 4.6	Zh	0.77	0.75	0.93	0.83	0.73	0.86
Gemini 2.5 Pro	Zh	0.76	0.74	0.92	0.82	0.72	0.78
Gemini 2.5 Flash	Zh	0.79	0.84	0.80	0.82	0.79	0.77
Qwen3.5-9B	Zh	0.64	0.75	0.61	0.67	0.65	0.68
Mistral-7B	Zh	0.59	0.76	0.47	0.58	0.62	0.87
Gemma 3 12B	Zh	0.61	0.65	0.74	0.69	0.57	0.71
Qwen2.5-7B	Zh	0.62	0.72	0.60	0.66	0.63	0.66
Llama 3.1	Zh	0.45	0.53	0.69	0.60	0.39	0.55
GPT-5.4	En	0.80	0.79	0.91	0.84	0.77	0.78
Qwen3.6-Plus	En	0.78	0.81	0.82	0.82	0.76	0.81
GPT-5.4-mini	En	0.78	0.86	0.76	0.81	0.79	0.83
Claude Opus 4.6	En	0.79	0.81	0.85	0.83	0.78	0.75
Claude Sonnet 4.6	En	0.75	0.78	0.82	0.80	0.74	0.77
Gemini 2.5 Pro	En	0.76	0.79	0.83	0.81	0.75	0.58
Gemini 2.5 Flash	En	0.71	0.72	0.86	0.78	0.67	0.80
Qwen3.5-9B	En	0.78	0.77	0.90	0.83	0.75	0.72
Mistral-7B	En	0.66	0.72	0.71	0.72	0.65	0.79
Gemma 3 12B	En	0.58	0.64	0.69	0.66	0.55	0.64
Qwen2.5-7B	En	0.54	0.64	0.56	0.59	0.54	0.49
Llama 3.1	En	0.47	0.54	0.75	0.63	0.40	0.76

Table 18: Language-specific results on the pun existence detection task. Each language-specific subset contains 500 samples. Zh and En denote Chinese and English, respectively; BAcc. and AUC denote balanced accuracy and ROC-AUC.

G Full Results for Pun Localization

This appendix provides the evaluation details and full results for the pun localization task. Models are instructed to output a single contiguous trigger span for each positive pun sample. If a response contains additional explanations, we first extract the explicitly marked trigger field. If no trigger field can be extracted, the prediction is treated as invalid. To prevent trivial containment matches, predictions that reproduce the entire input sentence or return an excessively long span are marked as invalid for all localization metrics.

We report exact match (EM), containment match, and character-level intersection-over-union (IoU). EM requires the normalized predicted span to be identical to the gold trigger span. Containment match is directional: a prediction is counted as correct only if the predicted span contains the gold trigger span, while cases where the gold span merely contains the prediction are not counted as correct. IoU is computed over character-level spans after normalization, using the same character-based granularity for both Chinese and English. When a predicted string occurs multiple times in the input, we use the occurrence with the maximum overlap with the gold span. For completeness, we also report soft precision, soft recall, and soft F1 based on character-level span overlap.

Table 19 presents the overall results across all evaluated models, and Table 20 reports the language-specific results.

Model	n	EM Count	EM Acc.	Cont. Count	Cont. Acc.	Soft Prec.	Soft Rec.	Soft F1	IoU
GPT-5.4	600	396	0.660	529	0.882	0.859	0.878	0.843	0.800
GPT-5.4-mini	600	370	0.617	523	0.872	0.844	0.847	0.813	0.765
Qwen3.6-Plus	600	395	0.658	520	0.867	0.889	0.807	0.818	0.778
Gemini 2.5 Flash	600	368	0.613	520	0.867	0.868	0.826	0.815	0.765
Claude Opus 4.6	600	417	0.695	520	0.867	0.878	0.838	0.836	0.802
Gemini 2.5 Pro	600	399	0.665	519	0.865	0.848	0.862	0.829	0.789
Claude Sonnet 4.6	600	389	0.648	512	0.853	0.835	0.868	0.826	0.783
Qwen3.5-9B	600	310	0.517	498	0.830	0.854	0.702	0.729	0.669
Gemma 3 12B	600	292	0.487	497	0.828	0.853	0.688	0.714	0.648
Qwen2.5-7B	600	265	0.442	468	0.780	0.801	0.637	0.666	0.603
Llama 3.1	600	228	0.380	443	0.738	0.787	0.587	0.629	0.558
Mistral-7B	600	158	0.263	387	0.645	0.710	0.498	0.542	0.461

Table 19: Overall results on the pun localization task. EM, Cont., and IoU denote exact match, containment match, and character-level intersection-over-union, respectively. Soft Prec., Soft Rec., and Soft F1 are computed from character-level span overlap.

Model	Lang.	EM	Cont.	Soft P	Soft R	Soft F1
Claude Opus 4.6	Zh	0.71	0.84	0.82	0.82	0.80
Claude Sonnet 4.6	Zh	0.70	0.84	0.80	0.85	0.80
Gemini 2.5 Pro	Zh	0.69	0.87	0.81	0.86	0.81
Qwen3.6-Plus	Zh	0.67	0.86	0.86	0.77	0.78
Gemini 2.5 Flash	Zh	0.67	0.87	0.82	0.85	0.81
GPT-5.4	Zh	0.66	0.87	0.82	0.85	0.81
GPT-5.4-mini	Zh	0.62	0.87	0.81	0.82	0.79
Qwen3.5-9B	Zh	0.59	0.86	0.85	0.71	0.73
Gemma 3 12B	Zh	0.53	0.85	0.83	0.68	0.70
Qwen2.5-7B	Zh	0.51	0.82	0.79	0.67	0.68
Llama 3.1	Zh	0.45	0.72	0.70	0.60	0.61
Mistral-7B	Zh	0.25	0.62	0.62	0.43	0.46
Claude Opus 4.6	En	0.73	0.90	0.91	0.85	0.86
GPT-5.4	En	0.70	0.90	0.86	0.89	0.85
Qwen3.6-Plus	En	0.70	0.88	0.89	0.83	0.84
Gemini 2.5 Pro	En	0.69	0.86	0.84	0.86	0.82
Claude Sonnet 4.6	En	0.66	0.87	0.82	0.86	0.82
GPT-5.4-mini	En	0.64	0.88	0.82	0.86	0.81
Gemini 2.5 Flash	En	0.64	0.87	0.84	0.83	0.81
Qwen3.5-9B	En	0.46	0.80	0.80	0.62	0.66
Gemma 3 12B	En	0.46	0.80	0.78	0.62	0.65
Qwen2.5-7B	En	0.39	0.74	0.74	0.54	0.58
Llama 3.1	En	0.32	0.76	0.75	0.48	0.54
Mistral-7B	En	0.29	0.68	0.66	0.45	0.50

Table 20: Language-specific results on the pun localization task. Each language-specific subset contains 300 samples. Zh and En denote Chinese and English, respectively.

H Full Results for Pun Type Classification

H.1 Fine-tuning and Distillation Setup

We conduct an additional fine-tuning analysis on Qwen2.5-7B for the pun type classification task. This analysis compares ordinary label-based supervision with mechanism-oriented distillation under the same base model and evaluation setting.

Training Data. We collect an additional training set of 1,500 positive pun samples that is separate from the test set used in the main classification experiment. The training set is balanced by language and mechanism type, containing 750 Chinese and 750 English samples. For each language, the three mechanism types are evenly distributed, with 250 Lexical–Semantic, 250 Structural–Syntactic, and 250 Structural–Semantic samples. All samples are annotated with gold mechanism labels under the same taxonomy used in the main benchmark.

Standard SFT. For standard supervised fine-tuning, we apply LoRA-based fine-tuning to Qwen2.5-7B using the gold mechanism labels as direct supervision. The model is trained to map each input pun to one of the three mechanism labels without being given explicit intermediate reasoning about the trigger, surface meaning, implicit meaning, or pun-forming mechanism.

Mechanism-oriented Distillation. For mechanism-oriented distillation, we use GPT-5.4 as the teacher model. Given each training sample, GPT-5.4 is prompted with CoT guidance to analyze the trigger, surface meaning, implicit meaning, and mechanism distinction before

producing the final mechanism label. We save this teacher-generated reasoning process together with the final label as the distillation target, and then fine-tune Qwen2.5-7B to reproduce the mechanism-oriented output. Compared with standard SFT, this setting provides intermediate teacher signals that explicitly expose the decision process behind the type distinction.

Inference and Evaluation. After fine-tuning, both Qwen2.5-7B-SFT and Qwen2.5-7B-Distill are evaluated on the same 600-sample pun type classification test set used in the main experiment. The test set contains 300 Chinese and 300 English samples, with balanced mechanism types within each language. The fine-tuned variants are evaluated under a single direct inference setting and are therefore reported without the non-CoT/CoT slash notation used for prompting-based evaluations. We parse the final predicted mechanism label and compute the same classification metrics as in the main experiment.

This appendix reports the full results and fine-tuning details for the pun type classification task. The task evaluates whether models can identify the minimal direct licensing mechanism of a pun. We use three mechanism labels: T1 for Lexical–Semantic puns, T2 for Structural–Syntactic puns, and T3 for Structural–Semantic puns. Compared with detection or localization, this task more directly probes mechanism-level discrimination, since models must distinguish whether the double meaning is licensed by a localized lexical unit, an alternative syntactic analysis, or expression-level semantic reinterpretation.

We first describe the additional fine-tuning analysis on Qwen2.5-7B, including the standard SFT setting and the mechanism-oriented distillation setting. Table 21 then presents the overall classification results under without-CoT and CoT prompting settings, together with the fine-tuned variants. In addition to accuracy, we report macro precision, macro recall, macro F1, weighted F1, and type-specific F1 scores. Macro-level metrics evaluate balanced mechanism recognition, while type-specific F1 scores reveal whether models are biased toward particular pun mechanisms.

Model	Setting	Acc.	Macro P	Macro R	Macro F1	Weighted F1	T1 F1	T2 F1	T3 F1
GPT-5.4	w/o CoT	0.745	0.734	0.675	0.687	0.721	0.809	0.894	0.359
Claude Opus 4.6	w/o CoT	0.748	0.805	0.630	0.605	0.669	0.824	0.908	0.083
Qwen3.6-Plus	w/o CoT	0.668	0.534	0.473	0.458	0.571	0.782	0.591	0.000
Gemini 2.5 Pro	w/o CoT	0.695	0.646	0.606	0.607	0.662	0.781	0.776	0.264
GPT-5.4-mini	w/o CoT	0.713	0.769	0.566	0.600	0.670	0.802	0.687	0.311
Claude Sonnet 4.6	w/o CoT	0.745	0.774	0.619	0.629	0.690	0.826	0.842	0.218
Gemini 2.5 Flash	w/o CoT	0.668	0.540	0.468	0.453	0.569	0.783	0.575	0.000
Gemma 3 12B	w/o CoT	0.580	0.294	0.345	0.293	0.466	0.736	0.144	0.000
Mistral-7B	w/o CoT	0.477	0.377	0.407	0.375	0.468	0.585	0.104	0.436
Qwen2.5-7B	w/o CoT	0.465	0.308	0.316	0.311	0.453	0.654	0.066	0.214
Llama 3.1	w/o CoT	0.438	0.326	0.307	0.290	0.425	0.631	0.155	0.084
GPT-5.4	with CoT	0.822	0.832	0.822	0.824	0.824	0.731	0.928	0.811
Claude Opus 4.6	with CoT	0.773	0.813	0.773	0.782	0.782	0.665	0.878	0.803
Qwen3.6-Plus	with CoT	0.763	0.811	0.763	0.771	0.771	0.698	0.780	0.835
Gemini 2.5 Pro	with CoT	0.703	0.831	0.703	0.758	0.758	0.669	0.841	0.763
GPT-5.4-mini	with CoT	0.797	0.808	0.797	0.799	0.799	0.706	0.863	0.827
Claude Sonnet 4.6	with CoT	0.763	0.807	0.763	0.776	0.776	0.673	0.845	0.811
Gemini 2.5 Flash	with CoT	0.753	0.780	0.753	0.753	0.753	0.637	0.826	0.795
Gemma 3 12B	with CoT	0.490	0.555	0.490	0.427	0.427	0.454	0.093	0.736
Mistral-7B	with CoT	0.420	0.512	0.420	0.366	0.366	0.501	0.127	0.471
Qwen2.5-7B	with CoT	0.502	0.571	0.502	0.447	0.447	0.486	0.167	0.686
Llama 3.1	with CoT	0.450	0.475	0.450	0.379	0.379	0.496	0.029	0.613
Qwen2.5-7B-Distill	Fine-tuned	0.758	0.763	0.758	0.752	0.752	0.617	0.789	0.849
Qwen2.5-7B-SFT	Fine-tuned	0.555	0.458	0.460	0.458	0.557	0.698	0.281	0.396

Table 21: Overall results on the pun type classification task. Each setting is evaluated on 600 samples. T1–T3 denote F1 scores for Lexical–Semantic, Structural–Syntactic, and Structural–Semantic puns, respectively. Fine-tuned variants are evaluated under a single inference setting.

Table 22 further reports row-normalized confusion matrices for representative models. Each row corresponds to a gold mechanism type, and each value indicates the percentage of instances assigned to a predicted label. The confusion matrices make the category bias more explicit: without CoT, several models tend to collapse T3 Structural–Semantic puns into T1 Lexical–Semantic predictions, whereas CoT prompting generally leads to more balanced predictions across the three mechanism types. The invalid column records outputs that cannot be mapped to one of the three predefined labels.

Model	Gold	Without CoT				CoT			
		T1	T2	T3	Inv.	T1	T2	T3	Inv.
Claude Opus 4.6	T1	98.3	1.1	0.6	0.0	60.0	1.0	37.5	1.5
	T2	13.6	86.4	0.0	0.0	13.5	79.0	1.0	6.5
	T3	95.6	0.0	4.4	0.0	7.0	0.0	93.0	0.0
Claude Sonnet 4.6	T1	97.8	0.8	1.4	0.0	66.5	0.5	29.5	3.5
	T2	20.4	74.8	4.9	0.0	20.0	73.5	1.0	5.5
	T3	86.9	0.0	13.1	0.0	11.0	0.0	89.0	0.0
GPT-5.4	T1	89.2	1.9	8.6	0.3	71.5	0.5	28.0	0.0
	T2	12.6	86.4	1.0	0.0	12.0	87.0	1.0	0.0
	T3	73.0	0.0	27.0	0.0	12.0	0.0	88.0	0.0
GPT-5.4-mini	T1	95.8	0.6	3.6	0.0	72.5	4.0	23.5	0.0
	T2	44.7	53.4	1.9	0.0	20.5	79.0	0.5	0.0
	T3	79.6	0.0	20.4	0.0	12.5	0.0	87.5	0.0
Gemini 2.5 Pro	T1	86.9	4.2	8.9	0.0	60.0	0.5	25.5	14.0
	T2	22.3	75.7	1.9	0.0	11.0	73.0	1.0	15.0
	T3	77.4	3.6	19.0	0.0	8.5	0.0	78.0	13.5
Gemini 2.5 Flash	T1	99.7	0.3	0.0	0.0	61.0	1.0	38.0	0.0
	T2	59.2	40.8	0.0	0.0	24.5	71.0	4.5	0.0
	T3	100.0	0.0	0.0	0.0	6.0	0.0	94.0	0.0
Qwen3.6-Plus	T1	99.2	0.6	0.3	0.0	74.0	0.0	26.0	0.0
	T2	57.3	42.7	0.0	0.0	29.5	64.0	1.0	5.5
	T3	100.0	0.0	0.0	0.0	8.5	0.0	91.0	0.5

Table 22: Row-normalized confusion matrices (%) for representative models on pun type classification under without-CoT and CoT settings. T1–T3 denote predicted mechanism labels, and Inv. denotes invalid predictions. Each row sums to approximately 100%, up to rounding.

I Full Results for Implicit Expression Recovery

This appendix reports the full results for the implicit expression recovery task. Given a positive pun sample, models are asked to recover the implicit expression or hidden meaning that is evoked by the pun but not directly stated in the surface text. Since the target expression may be paraphrased in multiple valid ways, we evaluate model outputs using both automatic string-based metrics and judge-based semantic scores.

We report Top-1 and Top-5 accuracy, edit-distance weighted scores, and LLM-as-Judge scores. Top-1 metrics evaluate whether the model ranks the intended implicit expression as its first answer, while Top-5 metrics evaluate whether the intended meaning is covered by any of the generated candidates. The edit-distance weighted score gives partial credit to close textual matches, and the judge score evaluates semantic adequacy according to the rubric in Appendix D. Scores of 4 or above are treated as usable recovery results under the judge rubric.

Table 23 presents the overall results across all

evaluated models. The results show that exact recovery remains difficult, especially under Top-1 accuracy, while the Best Top-5 judge scores are substantially higher. This suggests that models often generate semantically relevant candidates but still struggle to rank the intended implicit expression as the top answer.

Model	Samples	Top1 Acc.	Top5 Acc.	Top1 Edit-w.	Top5 Edit-w.	Top1 Judge	Best Top5 Judge
GPT-5.4	200	0.145	0.235	0.364	0.501	3.705	4.180
Gemini 2.5 Flash	200	0.105	0.230	0.337	0.514	3.543	3.975
Qwen3.6-Plus	200	0.110	0.205	0.344	0.484	3.750	4.235
Claude Opus 4.6	200	0.090	0.180	0.316	0.437	3.710	4.065
Gemini 2.5 Pro	200	0.105	0.170	0.337	0.482	3.700	4.190
GPT-5.4-mini	200	0.060	0.160	0.292	0.442	3.645	4.230
Claude Sonnet 4.6	200	0.085	0.145	0.280	0.417	3.736	4.198
Gemma 3 12B	200	0.035	0.080	0.248	0.356	3.035	3.845
Mistral 7B	200	0.025	0.065	0.197	0.341	2.720	3.940
Llama 3.1	200	0.020	0.025	0.188	0.293	2.855	3.780
Qwen2.5-7B	200	0.005	0.010	0.168	0.229	3.280	4.220

Table 23: Overall results on the implicit expression recovery task. Top-1 metrics evaluate the first generated candidate, while Top-5 metrics evaluate whether any of the five generated candidates matches or semantically captures the target implicit expression. Edit-w. denotes the edit-distance weighted score. Best Top5 Judge denotes the highest judge score among the five candidates.

J Full Results for Mechanism Explanation

mains more difficult than surface paraphrasing.

This appendix reports the language-specific results for the mechanism explanation task. Given a positive pun sample, models are asked to explain the surface meaning, the implicit meaning, the triggering mechanism, and the rhetorical effect of the pun. This task evaluates whether models can move beyond recognizing a pun and explicitly articulate how its double meaning is licensed.

We evaluate each explanation along four dimensions: surface meaning, implicit meaning, triggering mechanism, and rhetorical effect. Each dimension is scored on a 0–5 scale according to the rubric in Appendix D, and the overall score is the sum of the four sub-scores, with a maximum of 20. Table 24 reports language-specific results under both the direct-answer and CoT settings. Each language-specific subset contains 100 samples.

The results show that CoT prompting improves explanation quality for most models, especially on the implicit meaning, mechanism, and rhetorical-effect dimensions. However, the gains are not uniform across languages or model families. Even under CoT, models generally obtain higher scores on surface-meaning explanation than on implicit meaning, mechanism, and effect explanation, suggesting that mechanism-sensitive explanation re-

Model	Lang.	Overall/20	Literal/5	Implicit/5	Mechanism/5	Effect/5
Claude Opus 4.6	English	12.347/15.175	3.296/4.138	2.888/3.488	3.133/3.875	3.031/3.675
GPT-5.4	English	12.120/14.925	3.230/4.013	2.860/3.563	3.080/3.725	2.950/3.625
Claude Sonnet 4.6	English	11.781/14.488	3.292/3.938	2.854/3.463	2.896/3.538	2.740/3.550
GPT-5.4-mini	English	11.770/14.063	3.120/3.838	2.740/3.388	3.150/3.425	2.760/3.413
Qwen3.6-Plus	English	11.500/14.150	3.180/3.888	2.610/3.250	2.920/3.563	2.790/3.450
Gemini 2.5 Pro	English	11.490/14.502	3.250/4.056	2.560/3.315	2.890/3.464	2.790/3.667
Gemini 2.5 Flash	English	10.790/12.800	3.120/3.750	2.490/2.900	2.570/3.025	2.610/3.125
Llama 3.1	English	6.790/8.363	2.530/3.100	1.470/1.925	1.380/1.888	1.410/1.450
Mistral-7B	English	6.650/8.513	2.870/3.450	1.300/1.925	1.160/1.525	1.320/1.613
Gemma 3 12B	English	6.212/8.700	2.495/3.513	1.333/2.000	1.242/1.838	1.141/1.350
Qwen2.5-7B	English	6.070/8.625	2.410/3.150	1.390/2.050	1.340/1.788	0.930/1.638
Gemini 2.5 Pro	Chinese	11.200/13.324	3.110/3.782	2.500/2.947	2.740/3.240	2.850/3.355
Claude Opus 4.6	Chinese	11.150/14.150	3.080/3.900	2.600/3.388	2.640/3.288	2.830/3.575
Qwen3.6-Plus	Chinese	9.440/12.438	2.650/3.663	2.200/2.650	2.320/2.888	2.270/3.238
GPT-5.4	Chinese	9.196/14.325	2.515/4.075	2.196/3.038	2.299/3.513	2.186/3.700
Claude Sonnet 4.6	Chinese	9.131/13.863	2.747/3.975	2.030/3.313	2.101/3.263	2.253/3.313
Gemini 2.5 Flash	Chinese	8.230/11.788	2.770/3.763	1.820/2.500	1.790/2.550	1.850/2.975
GPT-5.4-mini	Chinese	7.903/12.738	2.301/3.825	1.806/2.675	1.946/2.950	1.849/3.288
Qwen2.5-7B	Chinese	6.070/7.938	2.450/3.263	1.200/1.775	1.330/1.563	1.090/1.338
Mistral-7B	Chinese	5.897/7.575	2.722/3.188	0.969/1.388	0.918/1.225	1.289/1.775
Gemma 3 12B	Chinese	5.820/8.013	2.420/3.475	1.160/1.700	0.860/1.413	1.380/1.425
Llama 3.1	Chinese	5.500/7.450	2.110/2.950	1.030/1.625	1.090/1.450	1.270/1.425

Table 24: Language-specific results on the mechanism explanation task. Each score is reported as Direct without CoT / CoT. Overall/20 denotes the sum of four judge sub-scores. Literal/5, Implicit/5, Mechanism/5, and Effect/5 denote scores for surface meaning explanation, implicit meaning explanation, triggering mechanism explanation, and rhetorical effect explanation, respectively.

K Full Results for Constrained Pun Generation

This appendix reports the full results for the constrained pun generation task. Given a topic, a target pun word, a target mechanism type, and a style constraint, models are asked to generate a punning sentence that satisfies all specified constraints. We evaluate generated outputs along four judge-scored dimensions: mechanism validity, dual-meaning clarity, language naturalness, and creativity. Each dimension is scored on a 0–5 scale according to the rubric in Appendix D, and the overall score is the sum of the four sub-scores, with a maximum of 20. Valid Rate denotes the proportion of outputs judged as valid puns, and Recov. Rate denotes the proportion of outputs whose intended implicit meaning can be recovered from the generated text.

Tables 25 and 26 present the results under the without-CoT and CoT-guided settings, respectively. The results show a consistent fluency–mechanism gap: most models achieve relatively high language

Model	Overall	Mech.	Dual	Nat.	Creat.	Valid	Recov.
GPT-5.4-mini	13.01	2.95	2.89	4.71	2.46	0.56	0.58
GPT-5.4	12.96	2.85	2.94	4.72	2.45	0.58	0.60
Claude Sonnet 4.6	12.92	2.90	3.01	4.52	2.49	0.59	0.61
Qwen3.6-Plus	12.65	2.77	2.73	4.75	2.40	0.53	0.55
Claude Opus 4.6	12.24	2.70	2.64	4.70	2.20	0.50	0.51
Gemini 2.5 Flash	10.58	2.14	2.11	4.57	1.76	0.37	0.39
Gemini 2.5 Pro	10.54	2.11	2.06	4.69	1.68	0.37	0.38
Gemma 3 12B	9.21	1.76	1.69	4.21	1.55	0.27	0.27
Qwen2.5-7B	8.70	1.47	1.34	4.67	1.22	0.24	0.24
Llama 3.1	8.34	1.43	1.35	4.37	1.19	0.21	0.22
Mistral-7B	7.67	1.40	1.34	3.68	1.25	0.22	0.24

Table 25: Constrained pun generation results under the without-CoT setting.

naturalness, but obtain much lower scores on mechanism validity and dual-meaning clarity. CoT guidance often improves mechanism validity, but may also reduce naturalness or creativity, suggesting that explicit reasoning can make generation more mechanism-oriented but also more conservative.

Model	Overall	Mech.	Dual	Nat.	Creat.	Valid	Recov.
Claude Sonnet 4.6	12.99	3.64	3.12	4.27	1.96	0.53	0.57
GPT-5.4	12.87	3.72	3.05	4.14	1.96	0.60	0.61
Claude Opus 4.6	12.49	3.52	2.86	4.29	1.82	0.52	0.54
GPT-5.4-mini	11.97	3.39	2.83	3.97	1.78	0.50	0.50
Gemini 2.5 Pro	11.67	2.82	2.48	4.69	1.68	0.37	0.38
Qwen3.6-Plus	11.60	3.21	2.58	4.34	1.47	0.39	0.39
Gemini 2.5 Flash	10.53	2.86	2.21	4.09	1.37	0.30	0.33
Gemma 3 12B	8.40	1.66	1.68	3.58	1.48	0.30	0.32
Qwen2.5-7B	8.39	2.03	1.29	3.86	1.21	0.22	0.25
Llama 3.1	7.16	1.20	1.17	3.72	1.07	0.15	0.19
Mistral-7B	6.67	1.09	1.06	3.51	1.01	0.13	0.15

Table 26: Constrained pun generation results under the CoT-guided setting.

Results by Target Mechanism Type. To examine whether generation performance varies with the target mechanism constraint, we further analyze GPT-5.4 outputs in the without-CoT setting by mechanism type. Table 27 reports the results.

Type	<i>n</i>	Mech.	Dual	Nat.	Valid
T1 Lexical–Semantic	34	3.08	3.18	4.73	0.65
T2 Structural–Syntactic	33	2.84	2.94	4.72	0.58
T3 Structural–Semantic	33	2.62	2.69	4.70	0.51

Table 27: GPT-5.4 constrained generation results by target mechanism type in the without-CoT setting. Mech., Dual, Nat., and Valid denote mechanism validity, dual-meaning clarity, language naturalness, and valid-pun rate, respectively.

A Kruskal–Wallis test indicates significant differences across mechanism types in mechanism validity ($H(2) = 8.64$, $p = .013$) and dual-meaning clarity ($H(2) = 7.92$, $p = .019$), but not in language naturalness ($H(2) = 0.41$, $p = .814$). Holm-corrected Dunn tests identify significant T1–T3 differences in mechanism validity ($p = .010$) and dual-meaning clarity ($p = .016$); the remaining pairwise comparisons are not significant. Although the valid-pun rate follows the same ordering, differences across types are not significant ($\chi^2(2) = 1.20$, $p = .549$). These results indicate type-dependent variation in mechanism-sensitive generation performance for GPT-5.4, whereas a more general analysis of item-level difficulty would require independent difficulty annotations.

Prompt-Sensitivity Analysis. To examine whether constrained-generation performance is primarily driven by detailed mechanism definitions or structured output scaffolding, we conduct a 2×2 ablation on GPT-5.4 in the without-CoT setting. The experiment crosses full mechanism definitions versus label-only descriptions with full

JSON versus minimal outputs. All four conditions use the same 100 evaluation instances, decoding parameters, sample construction, and number of runs. Table 28 reports the results.

Setting	Mech.	Dual	Nat.	Creat.	Valid	Recov.	Parse
Full def. + Full JSON	2.85	2.94	4.72	2.45	0.58	0.60	0.00
Full def. + Minimal	2.80	2.89	4.74	2.49	0.56	0.58	0.01
Label only + Full JSON	2.72	2.82	4.70	2.43	0.53	0.55	0.02
Label only + Minimal	2.66	2.76	4.73	2.47	0.51	0.53	0.00

Table 28: Prompt-sensitivity results for GPT-5.4 on constrained pun generation in the without-CoT setting. Mech., Dual, Nat., Creat., Valid, Recov., and Parse denote mechanism validity, dual-meaning clarity, language naturalness, creativity, valid-pun rate, implicit-meaning recoverability rate, and parse-failure rate, respectively.

Changing the output format alone produces only small differences. Under full definitions, replacing full JSON with minimal outputs changes mechanism validity from 2.85 to 2.80 and dual-meaning clarity from 2.94 to 2.89. Removing detailed definitions produces a somewhat larger but still moderate decline: under minimal outputs, mechanism validity decreases from 2.80 to 2.66 and dual-meaning clarity from 2.89 to 2.76. Crucially, language naturalness remains high in every condition (4.70–4.74), while mechanism validity and dual-meaning clarity remain substantially lower. Thus, detailed definitions provide a modest scaffold and JSON mainly standardizes output parsing; neither accounts for the observed fluency–mechanism gap.