# Treasure Hunt: Real-time Targeting of the Long Tail using Training-Time Markers

**Daniel D'souza**
Cohere Labs

**Julia Kreutzer**
Cohere Labs

**Adrien Morisot**
Cohere

**Ahmet Üstün***
Cohere

**Sara Hooker***†
Adaption Labs

## Abstract

One of the most profound challenges of modern machine learning is performing well on the long-tail of rare and underrepresented features. Large general-purpose models are trained for many tasks, but work best on high-frequency use cases. After training, it is hard to adapt a model to perform well on specific use cases underrepresented in the training corpus. Relying on prompt engineering or few-shot examples to maximize the output quality on a particular test case can be frustrating, as models can be highly sensitive to small changes, react in unpredicted ways or rely on a fixed system prompt for maintaining performance. In this work, we ask: *Can we optimize our training protocols to both improve controllability and performance on underrepresented use cases at inference time?* We revisit the divide between training and inference techniques to improve long-tail performance while providing users with a set of control levers the model is trained to be responsive to. We create a detailed taxonomy of data characteristics and task provenance to explicitly `control` generation attributes and implicitly `condition` generations at inference time. We fine-tune a base model to infer these markers automatically, which makes them optional at inference time. This principled and flexible approach yields pronounced improvements in performance on examples from the long tail of the training distribution. Overall, we observe lifts of 5.7% across all tasks. However, treasure markers are particularly effective at finding difficult to obtain gains in the long-tail. We observe relative lifts of up to 14.1% on underrepresented tasks like CodeRepair and absolute improvements of 35.3% on length instruction following evaluations.

## 1 Introduction

Large language models (LLMs) are expected to perform well on many different tasks. Therefore, training data is a heterogeneous mix, where instances can vary greatly in terms of format, contents, tasks, and languages, e.g. code generation [31, 33, 20, 63] vs. MCQA [48, 39]. At inference time, data points are not equally relevant, but it is often prohibitively expensive to go back and change the training distribution for each individual inference request. Hence, there is a mismatch in the distribution at training and inference time: training time distribution is often determined by ease of access to prior data collections and prior data augmentation efforts, while at inference time, new use cases might be underrepresented in the data but highly relevant to the user.

*Equal Mentorship.
†Work done at Cohere Labs.
**Corresponding authors:** {danieldsouza, juliakreutzer, ahmet}@cohere.com

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

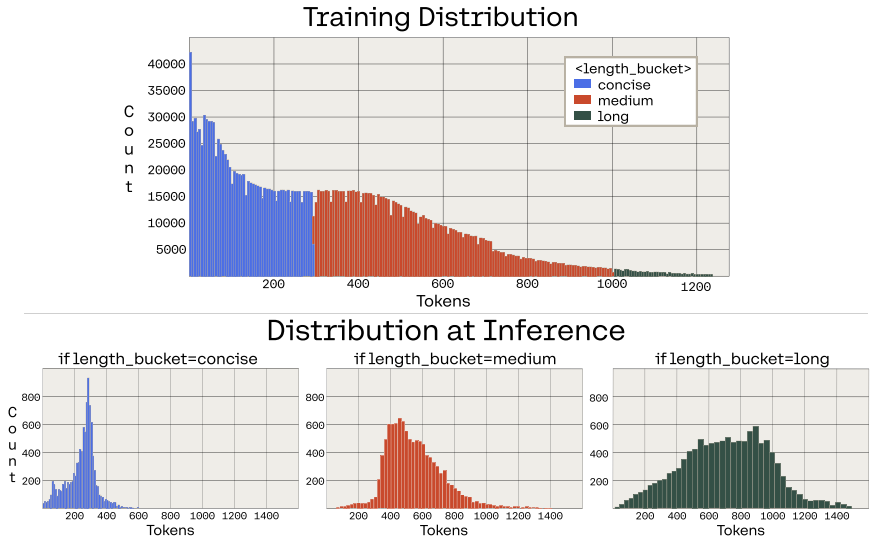



Figure 1: **Tapping into Distributions**: **(above)** illustrates the representation of various length buckets in the training distribution. **(below)** demonstrates the flexibility of the marker intervention on the mArena Hard test distribution. By modifying the `<length_bucket>..</length_bucket>` marker, the model can effectively tap into diverse training distributions, even for underrepresented length buckets.

To overcome this mismatch, techniques have been proposed to improve the conditioning of the output generation at inference. These involve prompt engineering [57, 60, 54], multi-shot examples [5, 29, 55, 30], chain-of-thought [53, 52, 41] or decoding strategies [46, 49]. However, these approaches place an enormous burden on practitioners and developers to anticipate what strategies deliver the best performance. Furthermore, the effectiveness is dependent on the exact configuration for a particular model, e.g. the order of multi-shot examples plays a role [32], and the wording of the prompt [2]. In this work, we ask *Can we optimize our training protocols to both improve controllability by the user and improve performance on rare use cases at inference time?*

Our approach amounts to building a treasure map of hyper-detailed task-specific markers, to allow for real-time automatic targeting of long-tail features during inference. We note that some of the earliest generative models have used tags to improve performance. However, these often targeted a single feature at a time or were applied uniformly to an entire dataset. These early tags fell out of favor over the last few years, with the focus turning to prompt engineering for users to guide the generation themselves. However, there have been a few wider ecosystem changes which prompt (*no pun intended*) revisiting the paradigm of adding markers to training, and also motivate this work: **1)** LLMs are now used by a far wider group of everyday users who can't be expected to be familiar with the complexities of prompt engineering, **2)** Many models are now served using an API which means training markers can be added automatically behind the API call (not visible to users), and hence can be far more complex and varied to guide and improve generations.

Our work is motivated by these two trends. We take a far wider view of training markers and explore a setting where a single data point can have up to 90 complex characteristics. We describe these as Treasure markers, introduced at training time to provide a map to guide towards higher performance at inference time. We motivate that this approach is particularly beneficial for long-tail modeling. Our goal is that the treasure map approach is robust at test-time, so we also aggressively experiment with marker dropout during training. This is akin to asking the model to still find the treasure even with missing clues.

In this work, our primary contributions are as follows:

1. **Introducing a more general framework for controllability.** We show that explicitly targeting controllability during training leads to pronounced gains at inference time, with little burden placed on the user. Training markers leads to significant downstream gains, ranging from a win rates increase on open ended generations of 5.7% on ArenaHard [27] across the entire distribution of

tasks relative to a model with no tags. We also see radical improvements in instruction following, with a 35.3% absolute decrease in violation rates on length instruction following tasks. Our training marker framework offers remarkable flexibility, allowing for control over both aspects of form (output format, length) and semantic qualities (quality, tone, style) while also being completely *optional* at inference, because the markers can be inferred accurately.

2. **Long Tail Lifts** The training with explicit markers is an effective method for leveraging long-tail features at inference time, unlocking high-performance even for the distributions that are underrepresented in the training data. While our framework enables a relative improvement of 7.9% on Code tasks over the baseline, we observe relative lifts of up to 14.1% on tasks like CodeRepair, which are highly underrepresented in the training data.
3. **Modeling Underlying Relationships**: We demonstrate that our approach effectively models underlying relationships in the data, as evidenced by a drastic violation reduction ($36.58\% \rightarrow 1.25\%$) in length-constraint evaluation, despite never seeing a training sample with a prompt instruction designating length constraints.

## 2 Methodology

### 2.1 Overview of Training Time Markers

We condition the output sequence $y$ given an instruction $x$ with added training markers $m$:

$$p(y|x,m) = \prod_{i=1}^{n} p(y_i \mid x, m, y_{<i}). \tag{1}$$

These markers encompass several different attributes of the data, including estimated quality scores, domains, and languages (2.2), which we store as a list of markers associated with a given data point (see Table 1 for an example). This template is treated as natural language and encoded with the same tokenizer as the text. We include the markers in both input (*appended* to the prompt) and output space (*prepended* to the completion), to induce the model to associate the properties of the generations with these characteristics. This reduces the burden on the practitioner or researcher at inference time, as the model learns to infer the correct markers.

The finetuning objective thus becomes to minimize the negative log likelihood of the target generations including the template, given a prompt with an optional input template:

$$-\frac{1}{|\mathcal{D}|}\sum_{d=1}^{|\mathcal{D}|} \log p_\theta(y_d, m_d \mid \text{dropout}(m_d), x_d) \tag{2}$$

This approach ensures that the model learns to faithfully generate and adhere to the training markers when provided on the prompt side.

⟨MARKER_LIST⟩
⟨domain⟩ Sciences ⟨ /domain⟩
⟨language⟩ French ⟨ /language⟩
⟨/MARKER_LIST⟩

Table 1: **An example list of training time markers** formatted in a standardized template.

**Training markers dropout** To avoid the model from becoming overly reliant on markers for completion or learning to trivially replicate the markers, we employ dual dropout strategies (dataset-level, sample-level) on the prompt space. In dataset-level dropout, we completely remove the training markers from the prompt for a random selection (*defined as a percentage of the dataset*). In sample-level dropout, we completely remove a random subset of training markers from each example (*defined as a percentage of all markers associated with a given example*). To ensure the model consistently produces markers at inference time, we do not introduce dropout on the generation side.

### 2.2 Taxonomy of Training Markers

We develop a comprehensive taxonomy around distinct groups of desired characteristics to capture key attributes of the training data, such as quality of the data, style, format, domain, and task. Table 8 contains the taxonomy with definitions and the set of valid marker values. Our goal was to study a broad range of markers encompassing both deterministic properties (e.g., length, language) and annotatable attributes (e.g., domain, task) to better characterize the SFT data mix. We selected

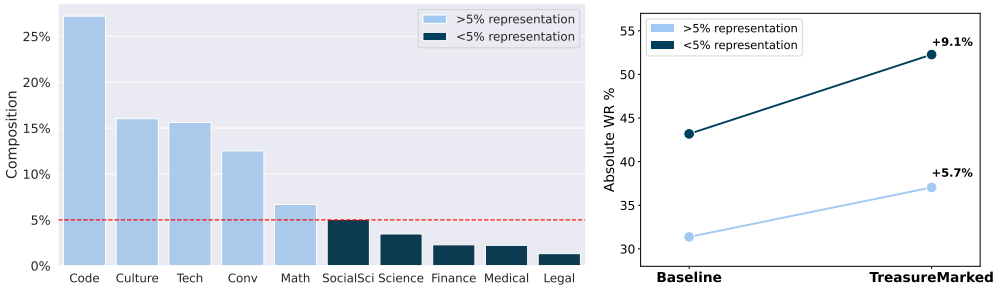
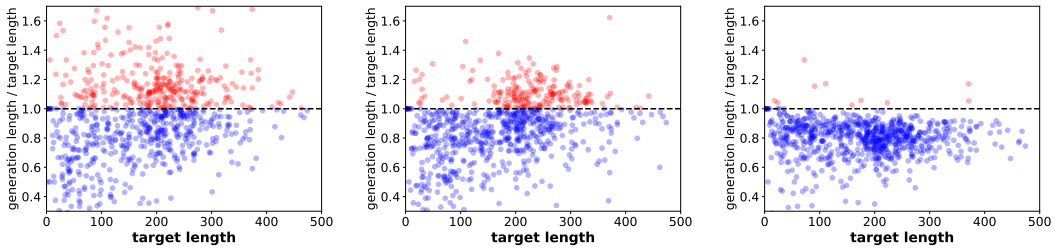



Figure 2: **Long tail domains benefit more from training markers: (left)** Domain distribution in the training data used to fine-tune the model. **(right)** Improvements in win rates over the baseline against Gemma2-9B on both majority and minority subsets on Arena-Hard-Auto dataset [27]. We group the test data into two sets of domains that have high (>5%) and low (< 5%) presence in training data.




(a) Baseline Model (b) **TreasureMarked** (c) **TreasureMarked** (*fixed*)

Figure 3: **Modeling data features flexibly with training time markers:** Results on the length instruction following on the AlpacaEval-Length-Instructed(LI) dataset. While **(a)** the *baseline* violates the length constraint with **36.58%**, **(b)** using the *TreasureMarked* model and allowing to infer tags on the exact same dataset reduces the violation to **24.7%**. **(c)** Conveying the requirement via an explicitly inserted length marker to the prompts, the *TreasureMarked* model violates the instruction on only **1.25%** of the dataset.

this taxonomy with inference-time use cases in mind: properties like quality, tone, style, and completion length are very desirable to control at inference. Moreover, to assess gains across both well-represented and low-frequency features, we intentionally designed the taxonomy to capture the inherent skew in data distributions. To that end, we add hyper-detailed markers for task, domain and code type which tend to have highly skewed frequencies with some instances occurring far more frequently than others.

To assign markers to samples in the training dataset, we utilize dataset-related information whenever possible and use an LLM to tag missing meta-information. Specifically, we use the multilingual open-weights model Command R+[1] for tagging of markers for `<domain>`, `<task>`, `<format>` whenever unavailable from the dataset. To improve tagging performance, we use detailed definitions paired with few-shot examples to provide context for markers during annotation. We add markers across 23 languages, so we use in-language few-shot examples in each language.

Our extensive set of 90 unique markers fall into 13 categories such as **Length**, **Style**, **Format**, **Quality**, **Source**, **Domain**, **Task**. We include an extensive description of all markers in appendix A. We describe the most frequently referenced categories below:

- **Length**: are markers that allow for control of completion length. It includes a level of granularity ranging from `<length_tokens>` and `<length_sentences>` to broader categories such as *concise*, *medium*, and *long*.
- **Language**: `<lang>` describes the language the completion is written in (i.e. Arabic, Japanese), enabling the model to improve language-specific generations and reduce language switching during

[1]Release blog of Command R+ `https://cohere.com/blog/command-r-plus-microsoft-azure`

inference. <code_type> is specifically used to identify programming languages for coding-related tasks (i.e. python, c++).

- **Quality**: <quality> provides a measurable score indicating the quality of a sample, often derived from human annotations or a Reward Model (RM). We also create a categorical marker <quality_bucket> by using quartiles within language-specific subsets into {1,2,3,4}, offering a broader description of quality.

- **Domain**: overarching category of the knowledge required to answer a given prompt (i.e. Sciences, Technology, Medical). We annotate domain markers either using LLM tagging or derive from the source of the dataset for domains like Math and Code.

- **Task**: <task> helps capture more fine-grained differences in task characteristics within a domain (i.e. summarization, reasoning, openended, explanation). Similar to the domain marker, we use LLM tagging or the data source information for obtaining task markers.

### 2.3 Experimental Set-up

**Training with Markers** We use a 7-billion parameters proprietary base model which is pretrained using a data mixture that consists of texts from 23 languages covering half the worlds population. We train our base model on a training corpus containing 2.7M examples made up of our mixture of instruction-style data sources. See Appendix B for full list of languages covered and more granular details about the training protocol.

**Inference Settings** At inference time, we evaluate performance gains under two different settings. In the default setting, which we refer to as "**TreasureMarked**", we do not fix any of the markers at inference. This setting asks: *Has the model learnt to infer the right markers without any intervention?* In the second setting which we refer to as "**TreasureMarked** (*fixed*)", we explicitly hardcode some of the markers at inference. This asks: *if we manually set the value of some markers, can we drive gains in performance?* This is very reasonable for cases like quality, where we always want to steer model behavior towards higher quality generations.

We compare both "**TreasureMarked**" and "**TreasureMarked** (*fixed*)" against a model trained on the same data, but without added markers that we refer to as *Baseline*. This allows for a clean comparison, and controls for the same amount of data seen in both variants.

**Core experimental variants and ablations** In the next section, we evaluate a variety of ways a model trained with markers shines at inference time. We inspect three axes of control: **(1)** quality in section 3.1.1, **(2)** length in section 3.3, and **(3)** language in section 3.5). Furthermore, we show how long-tail examples benefit from markers, even when only inferred at inference time (section 3.1), specifically in coding tasks (section 3.2) and for long generations (section 3.3). We present key experimental ablations, including understanding the impact of dropout applied to markers on downstream performance at inference time (Section 4).

#### 2.3.1 Evaluation

**Open-ended generation quality** We evaluate the impact of markers on both the English Arena-Hard-Auto v0.1 [27], and a translated version of this dataset, m-Arena Hard [8] used for multilingual evaluation. Arena-Hard-Auto is a challenging open-ended generation benchmark with prompts selected from user queries on Chatbot Arena. We measure Win Rate % against our *Baseline* model using GPT-4o.[2]

**Task-specific evaluations** In addition, we evaluate the models on benchmarks specific to tasks such as code (generation, repair, translation) and length conditioned instruction following to narrow in on long-tail effects and controllability levers. We introduce each of these evaluations within the respective results sections.

[2]We used gpt-4o-2024-05-13 as our judge model. Details: `https://platform.openai.com/docs/models/gpt-4o`

# 3 Results

## 3.1 Impact of Treasure Markers on Open-Ended Generation

**Open ended performance gains.** We measure Win Rates (%) of the *Baseline* and *TreasureMarked* models against Gemma2-9B [51] as a common point of comparison, visualized in Figure 2. We first consider our *TreasureMarked* variant, markers are only included in training but are inferred from the model itself during inference. Overall, we obtain an absolute increase of 5.7% in Win Rates from 32.1% to 37.8% across all tasks. This is reassuring, because it shows that markers at training time of the *TreasureMarked* model can already make a positive change at inference time, *even when only inferred by the model itself*, and even if the respective markers are rarely seen during training (e.g., for underrepresented domains).

**Performance on the long-tail** One of our core hypotheses is that treasure markers will be particularly helpful at preserving or unearthing gains on the long-tail. To validate this hypothesis, we evaluate performance post-training on domains represented with different frequencies in the training-set. As seen in Figure 2, `SocialScience, Sciences, Finance, Medical, and Legal` domains are particularly sparsely represented in the training data, each making up less than 5% of the training data. In contrast, Code is best represented in the training dataset. With inferred treasure markers, while there is an improvement of +5.7% across the higher-represented domains, we observe an even more pronounced gain of +9.1% in the underrepresented domains.

### 3.1.1 Fixed Treasure Markers

We also explore adding explicit markers in *Treasure-Marked (fixed)*. Here, we specifically target quality and ask *Can we control the generation quality of the model as a latent feature, using training time markers?* To test this, we measure generation quality on m-ArenaHard [8] across 23 languages, by only adding markers related to quality. For each value `[1,2,3,4]` of `<quality_bucket>`, we also include a `<quality>` score in conjunction with it. To obtain the `<quality>` score, we pick the 95% percentile calculated language-wise from the samples in the training data from each respective bucket. As evaluation, we measure the generation quality by the same Reward Model used to score the data during training to compute win rates against the *Baseline* model.

Figure 4 demonstrates the amount of `control` introduced by training time markers with win rates under the RM going from $48.21\% \rightarrow 56.5\%$ just by changing `<quality>, <quality_bucket>` at inference. These results showcase the potential of our framework, where markers representing a desired quality metric used during training yields control levers to leverage generations that tap into that quality metric at inference time.

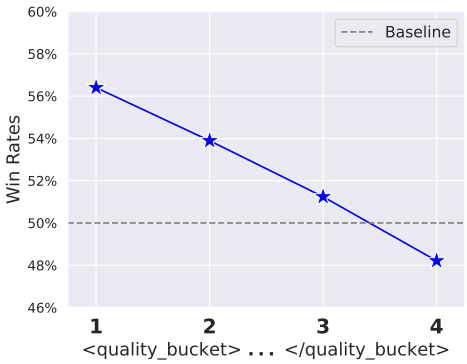



Figure 4: **Levers for Controlling Quality:** Changing the `<quality>, <quality_bucket>` markers at inference time provides control over generation quality with Win Rates (as measured by internal Reward Model) going from $48.21\% \rightarrow 56.5\%$ over the *Baseline* model, demonstrating successful control over quality as annotated in the training data.

## 3.2 Impact of Treasure Markers on Targeted Performance of Specific Sub-tasks

### 3.2.1 Code Performance

For code, we evaluate our model on three tasks from HumanEvalPack [38] dataset, and measure pass@1 rates. We use CodeSynthesis, CodeRepair, and CodeTranslation[3], covering *python, rust, java, javascript, go, c++*. These map to the following task markers in our taxonomy: `CodeGeneration`, `CodeFix`, and `CodeTranslation`.

[3]The CodeTranslation task is created by an all-to-all mapping between the 6 languages in HumanEvalPack

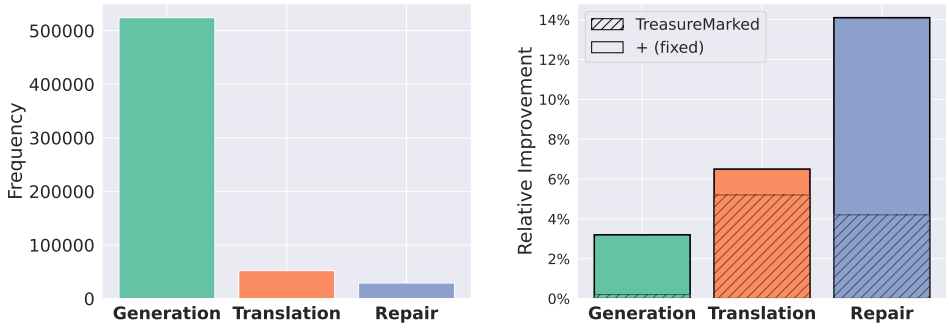



Figure 5: **Improvement on the Long Tail for Code tasks**: **(left)** Frequency of coding <task>s in the training dataset. **(right)** Despite being poorly represented in the training data, *CodeRepair* achieves a 14.1% relative improvement by leveraging targeted markers during inference further improving on the performance from the *TreasureMarked* model with inferred markers.

| **Original AlpcaEval-LI** | **TreasureMarked** *(fixed)* |
|---|---|
| `Answer the following instruction using 199 words or less.`<br><br>`What are the names of some famous actors that started their careers on Broadway?` | `What are the names of some famous actors that started their careers on Broadway?`<br>`<MARKER_LIST>`<br>`<length_tokens>199</length_tokens>`<br>`</MARKER_LIST>` |

Table 3: **Examples of length control strategies**: **(left)** Original instruction from AlpacaEval-LI dataset; **(right)** Modified instruction with constraint in the marker list.

During training, code comprises of 27.2% of the overall training corpus. However, we specifically pick this domain because the distribution of coding subtasks differs significantly in frequency in the training corpus, as shown in Figure 5. `CodeRepair` and `CodeTranslation` are very rare coding subproblems, while `CodeGeneration` is heavily represented at 75.8% within the coding data.

**Long-tail gains** We observe the largest gains on the long-tail code tasks. As seen in Figure 5, whether we provide the markers (*TreasureMarked (fixed)*) or the model infers them, both rare coding problems (`CodeTranslation` and `CodeRepair`) show large improvements of $0.597 \rightarrow 0.636$(+3.9% absolute, +6.5% relative) and $0.216 \rightarrow 0.246$(+3% absolute, +14.1% relative) over the baseline respectively. We note that these gains are far higher than the gains observed for the far more frequent task of `CodeGeneration`, which only improves $0.406 \rightarrow 0.419$ (+1.3% absolute, +3.2% relative). This shows that our framework benefits all parts of the distribution, but has disproportionate success enabling large lifts to highly infrequent features during training.

### 3.3 Length Control in Inference Time

| **Model** | **Violation** | **Win Rate** |
|---|---|---|
| *Baseline* | 36.58% | 14.36% |
| *TreasureMarked* | 24.74% | 19.48% |
| *+ (fixed)* | 1.25% | 21.22% |

Table 2: **Length Instruction Following** & generation quality on Alpaca-Eval LI.

To assess the impact of length conditioning during inference, we benchmark on the AlpacaEval-LI dataset [61], which evaluates how faithfully LLMs adhere to length constraints. We complement the measurements for length violation with *Win Rates(%)* by evaluating valid samples against the dataset provided completions using GPT-4o. We establish our baseline using completions generated by the *Baseline* model. Following a similar approach to [61], we assess *Violation(%)* as the proportion of samples exceeding the specified length constraint. We include additional details in Appendix B.

**Improvements to length control.** In Table 2, we show improvements of up to 35.3% in length violation rates. This pronounced improvement results in a mere 1.25% remaining violations for this evaluation set (essentially close to saturating performance on this evaluation). Even when the treasure markers are not explicitly provided but inferred directly by the model, we observe up to 11.8% absolute decrease in violation rates. These improvements to instruction following are non-trivial, and

also lead to overall win-rate gains of up to 6.86%, ensuring quality is not compromised as length constraints are enforced.

### 3.4 Machine Translation

To study the effects of the markers on machine translation, we benchmark on WMT'24++[10] and report translation performance from English to 22 languages ($en \rightarrow xx$) based on the languages seen in pretraining. We use XCOMET-XL [7] for evaluation, a state-of-the-art machine translation evaluation metric [12] via the `comet-compare`[4] library to conduct statistical significance analyses based on paired t-tests and bootstrap resampling.

Table 4 shows the results with the relative improvement over the *Baseline*. Training the model with markers and using them at inference time improves performance on 5 languages (*es, id, it, pt, ro)* significantly with up to 1.18 point gains, while retaining performance on all other languages. This constitutes a remarkable improvement, especially given that the training data, up to the markers, is identical. According to the metric delta analysis in [21], improvements of such magnitudes are very likely to be confirmed in human evaluations.

| en → xx | **ar** | **cs** | **de** | **el** | **es** | **fa** | **fr** | **he** | **hi** | **id** | **it** |
|---|---|---|---|---|---|---|---|---|---|---|---|
| *Baseline* | .6865 | .7485 | .8824 | .7463 | .8249 | .7099 | .789 | .7214 | .5158 | .776 | .8126 |
| *TreasureMarked (fixed)* | .6844 | .755 | .8848 | .7500 | .8307 | .7072 | .7948 | .7166 | .5229 | .7874 | .8194 |
| | (-0.21) | (+0.65) | (+0.24) | (+0.37) | (**+0.58**) | (-0.27) | (+0.58) | (-0.48) | (+0.71) | (**+1.14**) | (**+0.68**) |
| en → xx | **ja** | **ko** | **nl** | **pl** | **pt** | **ro** | **ru** | **tr** | **uk** | **vi** | **zh** |
| *Baseline* | .7368 | .7281 | .8103 | .7578 | .822 | .8048 | .7675 | .6669 | .7625 | .7593 | .7176 |
| *TreasureMarked (fixed)* | .7342 | .7318 | .8117 | .7546 | .8281 | .8166 | .7627 | .6723 | .7575 | .756 | .7200 |
| | (-0.26) | (+0.37) | (+0.14) | (-0.32) | (**+0.61**) | (**+1.18**) | (-0.48) | (+0.54) | (-0.50) | (-0.33) | (+0.24) |

Table 4: X-CometXL scores [7] on WMT'24++ test sets [10]. **Bold** differences are significant at $p \leq 0.05$ according to a paired T-Test and bootstrap resampling [22] as implemented in `comet-compare`

### 3.5 Language Control in Inference Time

As the final set of results, we focus on the effect of our training markers on ensuring a model responds in the language specified by the user. To evaluate this, we use the Language Confusion Benchmark [35] which measures the ability of a model to follow cross-lingual instructions such as "Respond in French...", to request completions in another language. We measure performance on the *Complex Prompts* subset of the cross-lingual benchmark across 14 languages. Following [35], we measure Line-level Pass Rate (**LPR**) that only deems a response "correct" if all lines in the generation match the user's desired language.

Table 5 shows results across 14 languages. Our model with training markers significantly improves language control performance in 13 out of 14 languages with an absolute gain of 10.98% on average across 14 languages, showcasing a remarkable improvement in controllability of inference time. We observe the largest gains for Russian (+18.6%) and the lowest gains for Chinese (+5.5%).

| | **ar** | **de** | **es** | **fr** | **hi** | **id** | **it** | **ja** | **ko** | **pt** | **ru** | **tr** | **vi** | **zh** | **Avg.** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Baseline* | 81.1 | 71.9 | 65.7 | 70.4 | 68.0 | 49.0 | 72.5 | 68.4 | 75.8 | 60.6 | 68.0 | 84.7 | 67.4 | 57.1 | 68.6 |
| *TreasureMarked (fixed)* | 88.4 | 84.4 | 82.7 | 79.8 | 73.7 | 66.7 | 82.8 | 83.8 | 85.0 | 72.2 | 86.6 | 78.6 | 82.8 | 62.6 | 79.58 (↑ **10.98**) |

Table 5: Line-level pass rate on Complex Prompts from the Language Confusion Benchmark [35].

## 4 Key ablations and Discussion

**How do markers interact?** We perform an additional ablation on the AlpacaEval-LI dataset from section 3.3 to study the effect of adding more *useful* markers at inference time. In addition to the `<length_tokens>` marker that conveys the explicit length constraint, we annotate and add the `<domain>` marker, which we suspect carries implicit length biases (e.g. legal text might be longer than conversations), but should add helpful context to the prompt.

[4] `https://unbabel.github.io/COMET/html/running.html`

With this we ask – *If multiple markers are added at inference, do their effects add up or cancel out?*

From Table 6, we observe that the effect of adding <domain> has a positive impact on the generation quality with a +3.5% jump in win rates albeit at the cost of a slight increase in Violation Rate(%). This indicates that there are multidimensional relationships that form between treasure markers during training and can be leveraged in conjunction to achieve desired characteristics at inference.

| Model | Violation | Win Rates |
|---|---|---|
| *TreasureMarked (fixed)* | 1.25% | 21.22% |
| + <domain> | 1.87% | 24.72% |

Table 6: **Multidimensional control** (Alpaca-Eval LI): Adding <domain> marker improves generation quality and hence Win Rates by +3.5% working in conjunction with <length_tokens>, without hurting the length control.

**What is the impact of the dropout on the marker prediction?**

To understand the impact of the marker dropout (§ 2.1), we train three variants with **dataset-level** dropouts of [0%, 50%, 70%] while **sample-level** dropout is fixed to 50%. Our goal with dropout is to teach the model to infer markers without needing explicit guidance at inference time. However, too much dropout may impede the model from learning key patterns between tags and output properties. To evaluate this, we calculate the accuracy of the markers inferred by the model to the underlying markers assigned to m-Arena Hard and average across all 23 languages [8]

In Table 7, we observe that the least extreme **dataset-level** dropout variant 0_50 struggles to predict the correct marker at inference time. This is expected performance, since at training time, 0% dropout of markers across the dataset implies all training sample prompts have markers associated with it which makes it overly dependent on the presence of markers at inference time. At inference time, as this is not provided, accuracy is very low at 3.42%. We note that at both 50% and 70% dataset level dropout, we observe similar final abilities to infer the correct markers. Given this, unless specified elsewhere, 50% **dataset-level** dropout is the default specification used throughout experiments since it strikes the best balance between learning and generalization.

| dataset_sample | <*domain*> | <*task*> | <*format*> | <*lang*> |
|---|---|---|---|---|
| 0_50 | 3.3% | 1.9% | 7.3% | 1.2% |
| 50_50 | 74.9% | 53.6% | 47.4% | 99.2% |
| 70_50 | 75.1% | 51.4% | 46.8% | 99.1% |

Table 7: **Effect of dropout** on marker prediction. Using no dropout (dataset-level) prevents the model to learn predicting the correct marker across categories, hence, hurts the flexibility of our framework.

## 5 Related Work

The idea of marking inputs goes back to works in neural sequence prediction before LLMs. The motivation was similar: to leverage discrete features to overcome data sparsity or imbalance and introduce levers of control. There are many individual works targeting one or two characteristics at a time, targeting special models and datasets, spanning diverse aspects such as language [17], quality [6, 50, 13] and domain [19, 4]. We build a much more general framework on top of a vast multi-task training corpus. We design a flexible approach that can be used for any text generation task with many characteristics, encompassing the above. Furthermore, we explicitly care about improving performance on the long-tail of underrepresented features.

LLM control has been pursued with respect to web domains in pretraining [18] and length in finetuning [61]. At inference time, values for these markers are specified to steer the generation. [14] further proposes a cooldown schedule in pretraining going from tagged data to untagged data in order to not require prefixes during inference. We focus on the instruction finetuning stage and incorporate nuanced multi-dimensional markers (i.e. the user can specify length *and* domain *and* format). We circumvent a cooldown schedule with dropout, hence not requiring a complete population of markers at inference time.

Related prefix and prompt tuning methods [28, 26, 45] use continuous embeddings learned for special tokens representing markers in training to condition predictions for specific tasks at inference time. In our case, we directly embed prefixes with the same vocabulary as the LLM, smoothly integrating them into the sequence. In our experiments in section 3.3, we found that this helps format following

even when specified in natural language and not markers. We include a more detailed discussion of the relevant literature in Appendix C.

## 6 Limitations

Firstly, although we design a comprehensive taxonomy for our training markers, there can be alternate categorizations of domains and tasks, and other data properties. However, our work primarily showcases the benefits of various data markers and how to leverage them during training (to model) and inference (to guide model outputs). We also acknowledge that our annotation setup of restricting each example to a single value per marker was adopted for simplicity. In practice, examples often span multiple overlapping concepts. Therefore, we believe that an extended taxonomy and annotation setup would likely further improve the results we demonstrate.

Secondly, we conduct experiments using a 7-billion parameters language model. We opt for 7B parameter scale to be able to run more experiments with different test cases as presented in the paper, instead of using a much larger model size with very limited experiments due to the underlying compute costs. However, considering a larger model could learn to model training-time markers with higher performance, we believe that our findings hold for larger model sizes.

Finally, we showcase the effectiveness of learning training-time markers in the supervised fine-tuning phase. We did not experiment with learning these markers in the pre-training phase as pre-training requires multiple degrees of higher costs. Therefore, we leave this exploration to the future work.

## 7 Conclusion

In this work, we proposed adding markers to training data to map out potential "treasures" that can be retrieved at inference time, such as specific task configurations or quality characteristics. In our experiments on multilingual instruction-finetuning, we showed that these markers are a powerful tool to execute control (quality, length, output language) over generations, and at the same time have beneficial effects for generation quality of underrepresented portions of the training data, such as rare coding tasks. We found that dropout of training markers trains the model to infer missing markers at inference time. With this flexibility, we allow users to "hunt treasures" without having to tediously engineer prompts or few-shot examples for optimized performance.

## Acknowledgments

We thank John Dang, Yiyang Nan, Thomas Euyang, Tom Kocmi, Tom Sherbone, Manoj Govindassamy, Cécile Robert-Michon, Leila Chan Currie, and other colleagues at Cohere and Cohere Labs for their support and thoughtful feedback.

## References

[1] Sweta Agrawal and Marine Carpuat. Controlling text complexity in neural machine translation. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1549–1564, Hong Kong, China, November 2019. Association for Computational Linguistics.

[2] Sotiris Anagnostidis and Jannis Bulian. How susceptible are llms to influence in prompts? *arXiv preprint arXiv:2408.11865*, 2024.

[3] Walid Aransa, Holger Schwenk, and Loïc Barrault. Improving continuous space language models auxiliary features. In Marcello Federico, Sebastian Stüker, and Jan Niehues, editors, *Proceedings of the 12th International Workshop on Spoken Language Translation: Papers*, pages 151–158, Da Nang, Vietnam, December 3-4 2015.

[4] Denny Britz, Quoc Le, and Reid Pryzant. Effective domain mixing for neural machine translation. In Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, and Julia Kreutzer,

editors, *Proceedings of the Second Conference on Machine Translation*, pages 118–126, Copenhagen, Denmark, September 2017. Association for Computational Linguistics.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[6] Isaac Caswell, Ciprian Chelba, and David Grangier. Tagged back-translation. In Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, André Martins, Christof Monz, Matteo Negri, Aurélie Névéol, Mariana Neves, Matt Post, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Fourth Conference on Machine Translation (Volume 1: Research Papers)*, pages 53–63, Florence, Italy, August 2019. Association for Computational Linguistics.

[7] Pierre Colombo, Nuno Guerreiro, Ricardo Rei, Daan Van, Luisa Coheur, and André Martins. xcomet: Transparent machine translation evaluation through fine-grained error detection. *Transactions of the Association for Computational Linguistics*, 2023.

[8] John Dang, Shivalika Singh, Daniel D'souza, Arash Ahmadian, Alejandro Salamanca, Madeline Smith, Aidan Peppin, Sungjin Hong, Manoj Govindassamy, Terrence Zhao, et al. Aya expanse: Combining research breakthroughs for a new multilingual frontier. *arXiv preprint arXiv:2412.04261*, 2024.

[9] Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*, 2020.

[10] Daniel Deutsch, Eleftheria Briakou, Isaac Caswell, Mara Finkelstein, Rebecca Galor, Juraj Juraska, Geza Kovacs, Alison Lui, Ricardo Rei, Jason Riesa, et al. Wmt24++: Expanding the language coverage of wmt24 to 55 languages & dialects. *arXiv preprint arXiv:2502.12404*, 2025.

[11] Weston Feely, Eva Hasler, and Adrià de Gispert. Controlling Japanese honorifics in English-to-Japanese neural machine translation. In Toshiaki Nakazawa, Chenchen Ding, Raj Dabre, Anoop Kunchukuttan, Nobushige Doi, Yusuke Oda, Ondřej Bojar, Shantipriya Parida, Isao Goto, and Hidaya Mino, editors, *Proceedings of the 6th Workshop on Asian Translation*, pages 45–53, Hong Kong, China, November 2019. Association for Computational Linguistics.

[12] Markus Freitag, Nitika Mathur, Daniel Deutsch, Chi-Kiu Lo, Eleftherios Avramidis, Ricardo Rei, Brian Thompson, Frederic Blain, Tom Kocmi, Jiayi Wang, David Ifeoluwa Adelani, Marianna Buchicchio, Chrysoula Zerva, and Alon Lavie. Are LLMs breaking MT metrics? results of the WMT24 metrics shared task. In Barry Haddow, Tom Kocmi, Philipp Koehn, and Christof Monz, editors, *Proceedings of the Ninth Conference on Machine Translation*, pages 47–81, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[13] Markus Freitag, David Vilar, David Grangier, Colin Cherry, and George Foster. A natural diet: Towards improving naturalness of machine translation output. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 3340–3353, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[14] Tianyu Gao, Alexander Wettig, Luxi He, Yihe Dong, Sadhika Malladi, and Danqi Chen. Metadata conditioning accelerates language model pre-training. *arXiv preprint arXiv:2501.01956*, 2025.

[15] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[16] Laura Jehl and Stefan Riezler. Document-level information as side constraints for improved neural patent translation. In Colin Cherry and Graham Neubig, editors, *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 1–12, Boston, MA, March 2018. Association for Machine Translation in the Americas.

[17] Melvin Johnson, Mike Schuster, Quoc V. Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017.

[18] Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*, 2019.

[19] Catherine Kobus, Josep Crego, and Jean Senellart. Domain control for neural machine translation. In Ruslan Mitkov and Galia Angelova, editors, *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 372–378, Varna, Bulgaria, September 2017. INCOMA Ltd.

[20] Denis Kocetkov, Raymond Li, Loubna Ben Allal, Jia Li, Chenghao Mou, Carlos Muñoz Ferrandis, Yacine Jernite, Margaret Mitchell, Sean Hughes, Thomas Wolf, et al. The stack: 3 tb of permissively licensed source code. *arXiv preprint arXiv:2211.15533*, 2022.

[21] Tom Kocmi, Vilém Zouhar, Christian Federmann, and Matt Post. Navigating the metrics maze: Reconciling score magnitudes and accuracies. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1999–2014, Bangkok, Thailand, August 2024. Association for Computational Linguistics.

[22] Philipp Koehn. Statistical significance tests for machine translation evaluation. In Dekang Lin and Dekai Wu, editors, *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain, July 2004. Association for Computational Linguistics.

[23] James Kuczmarski and Melvin Johnson. Gender-aware natural language translation. *Technical Disclosure Commons*, 2018.

[24] Nathan Lambert, Valentina Pyatkin, Jacob Morrison, LJ Miranda, Bill Yuchen Lin, Khyathi Chandu, Nouha Dziri, Sachin Kumar, Tom Zick, Yejin Choi, et al. Rewardbench: Evaluating reward models for language modeling. *arXiv preprint arXiv:2403.13787*, 2024.

[25] Samuel Larkin, Michel Simard, and Rebecca Knowles. Like chalk and cheese? on the effects of translationese in MT training. In Kevin Duh and Francisco Guzmán, editors, *Proceedings of Machine Translation Summit XVIII: Research Track*, pages 103–113, Virtual, August 2021. Association for Machine Translation in the Americas.

[26] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[27] Tianle Li, Wei-Lin Chiang, Evan Frick, Lisa Dunlap, Tianhao Wu, Banghua Zhu, Joseph E Gonzalez, and Ion Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. *arXiv preprint arXiv:2406.11939*, 2024.

[28] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online, August 2021. Association for Computational Linguistics.

[29] Xi Victoria Lin, Todor Mihaylov, Mikel Artetxe, Tianlu Wang, Shuohui Chen, Daniel Simig, Myle Ott, Naman Goyal, Shruti Bhosale, Jingfei Du, Ramakanth Pasunuru, Sam Shleifer, Punit Singh Koura, Vishrav Chaudhary, Brian O'Horo, Jeff Wang, Luke Zettlemoyer, Zornitsa Kozareva, Mona Diab, Veselin Stoyanov, and Xian Li. Few-shot learning with multilingual generative language models. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages

9019–9052, Abu Dhabi, United Arab Emirates, December 2022. Association for Computational Linguistics.

[30] Robert Logan IV, Ivana Balazevic, Eric Wallace, Fabio Petroni, Sameer Singh, and Sebastian Riedel. Cutting down on prompts and parameters: Simple few-shot learning with language models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2824–2835, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[31] Anton Lozhkov, Raymond Li, Loubna Ben Allal, Federico Cassano, Joel Lamy-Poirier, Nouamane Tazi, Ao Tang, Dmytro Pykhtar, Jiawei Liu, Yuxiang Wei, et al. Starcoder 2 and the stack v2: The next generation. *arXiv preprint arXiv:2402.19173*, 2024.

[32] Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8086–8098, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[33] Dung Nguyen Manh, Nam Le Hai, Anh TV Dau, Anh Minh Nguyen, Khanh Nghiem, Jin Guo, and Nghi DQ Bui. The vault: A comprehensive multilingual dataset for advancing code understanding and generation. *arXiv preprint arXiv:2305.06156*, 2023.

[34] Kelly Marchisio, Jialiang Guo, Cheng-I Lai, and Philipp Koehn. Controlling the reading level of machine translation output. In Mikel Forcada, Andy Way, Barry Haddow, and Rico Sennrich, editors, *Proceedings of Machine Translation Summit XVII: Research Track*, pages 193–203, Dublin, Ireland, August 2019. European Association for Machine Translation.

[35] Kelly Marchisio, Wei-Yin Ko, Alexandre Berard, Théo Dehaze, and Sebastian Ruder. Understanding and mitigating language confusion in LLMs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 6653–6677, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[36] Benjamin Marie, Raphael Rubino, and Atsushi Fujita. Tagged back-translation revisited: Why does it really work? In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5990–5997, Online, July 2020. Association for Computational Linguistics.

[37] Tomas Mikolov and Geoffrey Zweig. Context dependent recurrent neural network language model. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 234–239, 2012.

[38] Niklas Muennighoff, Qian Liu, Armel Zebaze, Qinkai Zheng, Binyuan Hui, Terry Yue Zhuo, Swayam Singh, Xiangru Tang, Leandro Von Werra, and Shayne Longpre. Octopack: Instruction tuning code large language models. In *NeurIPS 2023 Workshop on Instruction Tuning and Instruction Following*, 2023.

[39] Ankit Pal, Logesh Kumar Umapathi, and Malaikannan Sankarasubbu. Medmcqa: A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on health, inference, and learning*, pages 248–260. PMLR, 2022.

[40] Sahana Ramnath, Melvin Johnson, Abhirut Gupta, and Aravindan Raghuveer. HintedBT: Augmenting Back-Translation with quality and transliteration hints. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1717–1733, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[41] Leonardo Ranaldi and Andre Freitas. Aligning large and small language models via chain-of-thought reasoning. In Yvette Graham and Matthew Purver, editors, *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1812–1827, St. Julian's, Malta, March 2024. Association for Computational Linguistics.

[42] Parker Riley, Isaac Caswell, Markus Freitag, and David Grangier. Translationese as a language in "multilingual" NMT. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7737–7746, Online, July 2020. Association for Computational Linguistics.

[43] Rico Sennrich and Barry Haddow. Linguistic input features improve neural machine translation. In Ondřej Bojar, Christian Buck, Rajen Chatterjee, Christian Federmann, Liane Guillou, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Aurélie Névéol, Mariana Neves, Pavel Pecina, Martin Popel, Philipp Koehn, Christof Monz, Matteo Negri, Matt Post, Lucia Specia, Karin Verspoor, Jörg Tiedemann, and Marco Turchi, editors, *Proceedings of the First Conference on Machine Translation: Volume 1, Research Papers*, pages 83–91, Berlin, Germany, August 2016. Association for Computational Linguistics.

[44] Rico Sennrich, Barry Haddow, and Alexandra Birch. Controlling politeness in neural machine translation via side constraints. In Kevin Knight, Ani Nenkova, and Owen Rambow, editors, *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California, June 2016. Association for Computational Linguistics.

[45] Junhong Shen, Neil Tenenholtz, James Brian Hall, David Alvarez-Melis, and Nicolò Fusi. Tag-llm: repurposing general-purpose llms for specialized domains. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2024.

[46] Chufan Shi, Haoran Yang, Deng Cai, Zhisong Zhang, Yifan Wang, Yujiu Yang, and Wai Lam. A thorough examination of decoding methods in the era of LLMs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 8601–8629, Miami, Florida, USA, November 2024. Association for Computational Linguistics.

[47] Raphael Shu, Hideki Nakayama, and Kyunghyun Cho. Generating diverse translations with sentence codes. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1823–1827, Florence, Italy, July 2019. Association for Computational Linguistics.

[48] Shivalika Singh, Freddie Vargus, Daniel Dsouza, Börje F Karlsson, Abinaya Mahendiran, Wei-Yin Ko, Herumb Shandilya, Jay Patel, Deividas Mataciunas, Laura OMahony, et al. Aya dataset: An open-access collection for multilingual instruction tuning. *arXiv preprint arXiv:2402.06619*, 2024.

[49] Charlie Victor Snell, Jaehoon Lee, Kelvin Xu, and Aviral Kumar. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *The Thirteenth International Conference on Learning Representations*, 2025.

[50] Emmanouil Stergiadis, Satendra Kumar, Fedor Kovalev, and Pavel Levin. Multi-domain adaptation in neural machine translation through multidimensional tagging. In Janice Campbell, Ben Huyck, Stephen Larocca, Jay Marciano, Konstantin Savenkov, and Alex Yanishevsky, editors, *Proceedings of Machine Translation Summit XVIII: Users and Providers Track*, pages 396–420, Virtual, August 2021. Association for Machine Translation in the Americas.

[51] Gemma Team, Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[52] Lei Wang, Wanyu Xu, Yihuai Lan, Zhiqiang Hu, Yunshi Lan, Roy Ka-Wei Lee, and Ee-Peng Lim. Plan-and-solve prompting: Improving zero-shot chain-of-thought reasoning by large language models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2609–2634, Toronto, Canada, July 2023. Association for Computational Linguistics.

[53] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed H. Chi, Quoc V Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[54] Han Wenjuan, Wei Xiang, Cui Xingyu, Cheng Ning, Jiang Guangyuan, Qian Weinan, and Zhang Chi. Prompt engineering 101 prompt engineering guidelines from a linguistic perspective. In Maosong Sun, Jiye Liang, Xianpei Han, Zhiyuan Liu, and Yulan He, editors, *Proceedings of the 23rd Chinese National Conference on Computational Linguistics (Volume 1: Main Conference)*, pages 1408–1426, Taiyuan, China, July 2024. Chinese Information Processing Society of China.

[55] Genta Winata, Shijie Wu, Mayank Kulkarni, Thamar Solorio, and Daniel Preotiuc-Pietro. Cross-lingual few-shot learning on unseen languages. In Yulan He, Heng Ji, Sujian Li, Yang Liu, and Chua-Hui Chang, editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 777–791, Online only, November 2022. Association for Computational Linguistics.

[56] Liwei Wu, Shanbo Cheng, Mingxuan Wang, and Lei Li. Language tags matter for zero-shot neural machine translation. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 3001–3007, Online, August 2021. Association for Computational Linguistics.

[57] Yangjian Wu and Gang Hu. Exploring prompt engineering with GPT language models for document-level machine translation: Insights and findings. In Philipp Koehn, Barry Haddow, Tom Kocmi, and Christof Monz, editors, *Proceedings of the Eighth Conference on Machine Translation*, pages 166–169, Singapore, December 2023. Association for Computational Linguistics.

[58] Hayahide Yamagishi, Shin Kanouchi, Takayuki Sato, and Mamoru Komachi. Controlling the voice of a sentence in Japanese-to-English neural machine translation. In Toshiaki Nakazawa, Hideya Mino, Chenchen Ding, Isao Goto, Graham Neubig, Sadao Kurohashi, Ir. Hammam Riza, and Pushpak Bhattacharyya, editors, *Proceedings of the 3rd Workshop on Asian Translation (WAT2016)*, pages 203–210, Osaka, Japan, December 2016. The COLING 2016 Organizing Committee.

[59] Kexin Yang, Dayiheng Liu, Wenqiang Lei, Baosong Yang, Mingfeng Xue, Boxing Chen, and Jun Xie. Tailor: A soft-prompt-based approach to attribute-based controlled text generation. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 410–427, Toronto, Canada, July 2023. Association for Computational Linguistics.

[60] Fangyi Yu, Lee Quartey, and Frank Schilder. Exploring the effectiveness of prompt engineering for legal reasoning tasks. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023*, pages 13582–13596, Toronto, Canada, July 2023. Association for Computational Linguistics.

[61] Weizhe Yuan, Ilia Kulikov, Ping Yu, Kyunghyun Cho, Sainbayar Sukhbaatar, Jason Weston, and Jing Xu. Following length constraints in instructions. *arXiv preprint arXiv:2406.17744*, 2024.

[62] Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. A survey of controllable text generation using transformer-based pre-trained language models. *ACM Comput. Surv.*, 56(3), October 2023.

[63] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017.

# A Taxonomy for training time markers

## A.1 Categories for Training Markers

**Length**: <length_tokens>, <length_sentences>, and <length_paragraphs> models granular control in generation length. We tokenize using language-specific Spacy models[15] to obtain token and sentence counts. For paragraph counts, we use the "\n\n" delimiter. <length_bucket> categorizes generations into broader categories such as *concise* (under 300 tokens), *medium* (between 300 and 1,000 tokens) or *long* (over 1,000 tokens), providing a more general level of control when needed.

**Format**: <format> is used to describe generations with specific output structures such as: JSON, Markdown, or tabular formats. This is particularly useful to condition stricter format requirements needed for real world use cases.

**Style**: <style> captures tone and manner of communication, distinguishing between different modes of expression such as "*Formal*" and "*Informal*". We also add a "*Custom*" value to model for training examples where the user specifies a particular format. For instance, at inference if a user asks, "Respond like Yoda you will" this marker will allow the model to adapt its response to match the requested style. We annotate this marker by using dataset-related information.

**Language**: <language> describes the natural language of the generation, enabling us to model responses in specific languages during inference. We provide detailed markers across the 23 languages covered by our model. Our goal with this tag is to improve language-specific generations and reduce language switching where a prompt specified by a user in one language is not responded to in the same language in the completion. <code_type> is specifically used to model programming languages for coding-related tasks. We annotate this marker by using dataset-related information.

**Quality**: <quality> provides a measurable score indicating the quality of a sample, often derived from human annotations or a Reward Model (RM). We utilize a proprietary reward model[5] to assign rewards to a subset of our training data. We also use these rewards to create a categorical marker <quality_bucket> by using quartiles within language-specific subsets into {1,2,3,4}, offering a broader description of quality.

**Source**: <source> describes the origin of the data, distinguishing between human-generated content and other methods of data creation like synthetic and translation. We annotate this marker by using dataset-related information.

**Domain**: <domain> ensures that domain-specific knowledge is captured from training subsets, which can then be leveraged at inference to generate content that is relevant and accurate within a particular field. This is particularly crucial for inputs that could belong to multiple fields. For instance, when a user asks, "How do I calculate a factorial?", specifying the <domain> as either Code or Math provides valuable context for modeling the interaction. In cases where this marker cannot be obtained from the dataset information, we employ an LLM to annotate and provide our detailed prompt in D

**Task**: <task> defines the overall objective of the generation and helps capture task-specific behaviors, especially when outputs involve complex combinations of formats or actions. This marker is useful to model dataset-wise characteristics. We hypothesize this is particularly helpful for indicating complex workflows during inference. For example, distinguishing between Translation and CodeTranslation, or Rewrite and CodeFix, enhances the descriptiveness of datapoints within the same training pool. In cases where this marker cannot be obtained from the dataset information, we employ an LLM to annotate and provide our detailed prompt in D

# B Additional details about experimental sections and key ablations.

**Languages covered by Markers** We cover 23 languages: *Arabic, Chinese (simplified & traditional), Czech, Dutch, English, French, German, Greek, Hebrew, Hindi, Indonesian, Italian, Japanese, Korean, Persian, Polish, Portuguese, Romanian, Russian, Spanish, Turkish, Ukrainian and Vietnamese*.

**Training protocol** Training for each variant spanned 8,000 steps, employed a cosine learning rate schedule with a warm-up phase, using a batch size of 32 and an evaluation batch size of 64. We train

[5]The RM is competitive with leading reward models on the RewardBench Leaderboard [24](https://huggingface.co/spaces/allenai/reward-bench)

| category | definition | values |
|---|---|---|
| <quality> | Score indicating the quality assigned to a sample as annotated by a human or a RewardModel(RM). | *float* |
| <quality_bucket> | <quality> bucketed into quartiles (calculated by language).<br>1 indicating *highest* quality and 4 indicating *lowest* quality | {1,2,3,4} |
| <length_tokens> | # of tokens | *int* |
| <length_sentences> | # of sentences | *int* |
| <length_paragraphs> | # of paragraphs | *int* |
| <length_bucket> | <length_tokens> bucketed by defined response length ranges | concise, medium, long |
| <task> | Task-related information | {OpenEnded, Explanation, Translation, Classification, CreativeWriting, QuestionAnswering, InformationExtraction, Summarization, Rewrite, Reasoning, CodeGeneration, CodeFix, CodeTranslation, CodeExplanation} |
| <domain> | Domain-related information | {Sciences, Technology, SocialSciences, Culture, Medical, Legal, Unspecified, Conversation, Code, Math} |
| <code_type> | *(coding tasks)* Programming languages | {python, javascript, cpp, cobol, java, go, rust, swift, csharp, php, typescript, shell, c, kotlin, ruby, haskell, sql} |
| <format> | To model desired generation format | {MCQAnswer, ChainOfThought, XML, JSON, Enumeration, Tabular, Markdown, Latex} |
| <source> | To model the data-generation/annotation source | {Human, Translation, Synthetic, Others} |
| <style> | To model tone and style of the generation | {Formal, Informal, Custom} |
| <lang> | To model the 23 languages present in the training data | {English, French, Spanish, Italian, German, Portuguese, Japanese, Korean, Arabic, Chinese, Russian, Polish, Turkish, Vietnamese, Dutch, Czech, Indonesian, Ukrainian, Romanian, Greek, Hindi, Hebrew, Persian} |

Table 8: **Comprehensive taxonomy for training time markers:** Our taxonomy contains 13 categories shown with their descriptions and values.

for 2 epochs with a peak learning rate of at $2.5 \times 10^{-4}$, achieved through 10 warm-up steps starting from a learning rate of 0.0, and then decay back to $1.25 \times 10^{-4}$. One fine-tuning run using 8,000 steps on 128 Nvidia H100 GPUs takes around 6 hours.

**Length Evaluations** Given the original instruction in the AlpacaEval-LI dataset [61] contains the exact constraint, our **TreasureMarked** and **TreasureMarked** (*fixed*) both contain explicit reference to the contraint. For **TreasureMarked**, we present the original length-instructed prompt, allowing the model to deduce the associated tags. This approach evaluates the model's ability to extrapolate tags from instructions. in contrast, for **TreasureMarked** (*fixed*), since the original instruction contains the exact constraint, we investigate an additional control strategy where we provide the constraint in the marker template if the taxonomy directly supports it. We remove the length instruction and append the corresponding `<length_tokens>` tag with the appropriate value. Table 3 provides an example of an edited prompt. This strategy assesses the model's adherence to known templates and its ability to follow explicit length requirements that are only provided via the marker template.

**Language Control** We insert training markers present in the data into the prompt, but leave out the `<lang>` marker, since it is already present in the prompt.

# C Extended Related Work

**From one- to multi-dimensional training data tagging** The idea of tagging inputs in neural sequence prediction goes back to early applications in machine translation and language modeling. The motivation there was to leverage discrete features to overcome data sparsity or imbalance and introduce levers of control. In early neural LMs, often tokens were added to target a very specific attribute such as topic [37] or auxiliary features [3] including genre and length. A specific token for example was added at inference time to control a feature in translation like the target language [17], or desired output quality [6, 42, 36, 25, 13] and text complexity [1, 34] but also language-specific nuanced attributes like politeness [44, 11], the voice [58], gender [23]. domains [19, 4], or diversity [47] of translations. Other works enriched the input representation during training with discrete linguistic features [43] or document information [16] for a better contextualization at inference time. Where and how tags should be placed best differed across applications [16, 56]. Some tags were combined into multidimensional tagging [50, 40] All of these were individual efforts that target one or two dimensions at a time, highly specialized for one trained target model and with training data for one particular task. In contrast, our focus is on a much more general framework with a vast training corpus that targets general performance. Our approach is similarly general, where instead of a single feature, we want to enable a flexible approach that can be used for any text generation task. Furthermore, our goal is to explicitly target improving performance on the long-tail of underrepresented features.

**From control in pretraining to control in instruction finetuning** In LLM research, there are several related works that experiment with adding prefixes for *control in pretraining*: [18] add control codes for desired text features in pretraining of a LLM derived from the structure of their source, i.e., subdomains or links of online texts and specific task labels for translation and QA. At inference time, values for these control codes are specific to steer the generation. [14] further propose a cooldown schedule in pretraining going from tagged data to untagged data in order to not require prefixes at inference. [61] focus on length control by adding natural language length specification templates to *fine-tuning* data for DPO. In our work we focus on the instruction finetuning stage and incorporate nuanced multi-dimensional tags (i.e. the user can specify length *and* domain *and* format). We circumvent a cooldown schedule by simply introducing tag dropout, hence requiring a much smaller volume of tagged data at training time, and not a complete population of tags at inference time.

**From encododed to inferred meta-information** Related prefix and prompt tuning methods [28, 26] use continuous embeddings learned for special tokens representing tags in training to condition predictions for specific tasks at inference time. [45] further break those into separate tags for domain and function tags. In our case, we directly embed prefixes with the same vocabulary as the LLM, smoothly integrating them into the sequence. In our experiments, we find that this helps format following even when specified in natural language and not tags.

Attribute-based control in LLM generations has also been pursued with other methods, such as attribute classifiers [9] or learned attribute vectors [59] — see [62] for a comprehensive survey.

# D LLM Annotation

For the following training markers : <domain>, <task>, <format> we annotate using the multilingual open-weights model Command R+.

We provide definitions and multilingual few-shot examples (except for <format>) to obtain high-quality annotations from the LLM. The prompt used for tagging is as follows:

### D.0.1 <domain>

```
You are a helpful assistant whose goal is to classify the given prompt into
↪  a single class given the following definitions

`Sciences` : Topics related to the broad area of knowledge encompassing all
↪  scientific disciplines, including biology, chemistry, physics, earth
↪  sciences, and astronomy, which study the natural world through
↪  observation, experimentation, and analysis, aiming to understand
↪  fundamental principles and phenomena across various scales and aspects
↪  of the universe
`Technology` : Topics related to the broad area of knowledge encompassing
↪  all engineering and technical disciplines, including Computer Science,
↪  Software Engineering, Internet of Things(IoT), Cybersecurity, Data
↪  Science, Artificial Intelligence, Machine Learning and various
↪  engineering disciplines like Mechanical Engineering, Civil Engineering
↪  and Biotechnology
`SocialSciences` : Topics related to the broad area of knowledge
↪  encompassing all academic disciplines dedicated to the systematic study
↪  of human society, social relationships, and the structures that shape
↪  them, including fields like anthropology, economics, political science,
↪  psychology, and sociology, all focused on understanding how individuals
↪  and groups interact within a society and the factors influencing their
↪  behavior, cultural norms, and societal institutions
`Culture` : Topics related to the broad area of knowledge encompassing all
↪  cultural practices or beliefs within societies, including related
↪  concepts or behaviors that people within a culture group share and
↪  understand as belonging together, like food, art, language, family
↪  structure, societal norms or religious rituals
`Medical` : Topics related to the broad area of knowledge and practice
↪  encompassing all medicine and healthcare, including diagnosing and
↪  treating diseases, preventative measures, specialties like surgery,
↪  cardiology, oncology, pediatrics, and more, all built upon the
↪  foundation of basic medical sciences and patient care principles
`Finance` : Topics related to the broad area of knowledge encompassing
↪  activities like managing money, business ethics, investing, borrowing,
↪  lending, trading, budgeting, saving, and forecasting, essentially
↪  focusing on the acquisition, allocation, and management of capital
↪  within businesses
, individuals, and governments across various financial markets and
↪  instruments
`Legal` : Topics related to the broad area of knowledge encompassing
↪  Private, Public and Criminal Law, Criminal Justice, Law Enforcement,
↪  Policing, Justice Systems or Crime
`Conversation` : Topics related to Conversation, Chit-Chat or Roleplay
```

```
`Code` : Topics related to a specific subject/field within computer
↪  programming where software is designed and developed to solve problems
↪  related to a particular industry, business function, or area of
↪  expertise, essentially defining the target audience and unique
↪  requirements for the code being written including tasks like Code
↪  Generation, Code Fix and Code Explanation
`Math` : Topics related to the broad field of study that uses numbers,
↪  shapes, and formulas to describe and quantify the world, including
↪  areas like Logical Reasoning, Quantitative Calculation, Pattern
↪  Recognition, Formulating Conjectures, Arithmetic, Algebra, Geometry,
↪  Number Theory, Set Theory and Analysis

If you are unable to confidently assign one of the above classes, you will
↪  simply respond with `Unspecified` and nothing else.


Note:
- You are only to respond with the name of the class you believe best
↪  matches the domain of the example.
- You are only allowed to classify the example into one of the following
↪  tags :
[`Sciences`, `Technology`, `SocialSciences`, `Culture`, `Medical`,
↪  `Finance`, `Legal`, `Conversation`, `Code`, `Math`, `Unspecified`]

Here are a few examples :

Prompt : What is photosynthesis?
Answer : `Sciences`

Prompt : What is the TCP/IP protocol and how does it work ?
Answer : `Technology`

Prompt : How has globalization affected social cohesion ?
Answer : `Social Sciences`

Prompt : What is an example of a popular dish that is available in multiple
↪  communities but known under different names?
Answer : `Culture`

Prompt : How long does one have to fast before a fasting sugar blood test?
Answer : `Medical`

Prompt : Analyze the impact of microfinance initiatives on poverty
↪  alleviation in developing countries.
Answer : `Finance`

Prompt : What is the difference between a first-degree crime and a
↪  second-degree crime?
Answer : `Legal`

Prompt : Hey! How are you?
Answer : `Conversation`


Prompt : Given a variable x=3.142 in Python, how would I use an f-string to
↪  show just 1 decimal value?
Answer : `Code`

Prompt : Solve the quadratic equation: x² + 5x - 6 = 0
```

```
Answer : `Math`

Prompt : Use ABC notation to write a melody in the style of a folk tune.
Answer :
```

#### D.0.2 <task>

```
You are a helpful assistant whose goal is to classify the given prompt into
↪  a single class given the following definitions


`CodeTranslation` : Tasks related to the process of converting source code
↪  from one programming language to another while preserving the code's
↪  functionality
`CodeExplanation` : Tasks related to the specific process of explaining a
↪  snippet of code in a programming language
`CodeGeneration` : Tasks related to the specific process of generating a
↪  snippet of code in a programming language
`Explanation` : Tasks related to explaining a concept in any domain
`CreativeWriting` : Tasks related to any form of writing that employs
↪  creative, literary or poetic techniques that displays imagination or
↪  invention including role-play
`QuestionAnswering` : Tasks related to any form of question answering,
↪  including open-ended questions, closed-ended questions about a given
↪  context and requests for information about a particular entity. This
↪  will also generally include your `what`, 'which', 'who', 'when' type
↪  questions
`OpenEnded` : Tasks related to any form of open-ended text generation like
↪  chat, conversation or chit-chat
`InformationExtraction` : Tasks related to any form of information
↪  extraction usually involving some context
`Summarization` : Tasks related to any form of summarization including but
↪  not limited to abstractive summarization, extractive summarization or
↪  concise descriptions of content
`CodeFix` : Tasks related to the specific process of correcting/fixing a
↪  piece of code to achieve the desired result.
`Reasoning` : Tasks involving any form of Ideation, Reasoning, Problem
↪  Solving, Instruction Following or Chain-of-Thought(CoT) in order to
↪  achieve the desired result.
`Rewrite` : Tasks involving any form of
↪  re-writing/re-phasing/re-wording/re-framing in order to achieve the
↪  desired result.
`Classification` : Tasks related to specific request of classification
↪  where you are required to assign a thing to one of several groups
`Translation` : Tasks related to specific request of translating a given
↪  piece of text from one language to another language
If you are unable to confidently assign one of the above classes, you will
↪  simply respond with `Unspecified` and nothing else.
Note:
- You are only to respond with the name of the class you believe best
↪  matches the domain of the example.
- You are only allowed to classify the example into one of the following
↪  tags :
[`CodeTranslation`, `CodeExplanation`, `CodeGeneration`, `Explanation`,
↪  `CreativeWriting`, `QuestionAnswering`, `OpenEnded`,
↪  `InformationExtraction`, `Summarization`, `CodeFix`, `Reasoning`,
↪  `Rewrite`, `Classification`, `Translation`, `Unspecified`]

Here are a few examples :
```

```
Prompt : Translate the following Python function to equivalent JavaScript
↪  code that checks if a string is a palindrome.
            def is_palindrome(str):
                return str == str[::-1]
Answer : `CodeTranslation`

Prompt : Explain the following python function :
            def is_palindrome(str):
                return str == str[::-1]
Answer : `CodeExplanation`

Prompt : Generate a Python function to check whether a string is a
↪  palindrome.
Answer : `CodeGeneration`

Prompt : Explain briefly how the water cycle works
Answer : `Explanation`

Prompt : Translate the following sentence from English to Spanish, using a
↪  formal tone: 'We are pleased to announce the new partnership with our
↪  company.'
Answer : `Translation`

Prompt : You're a talk show host. Pick two guests that are wildly different
↪  from each other. Briefly introduce them
Answer : `CreativeWriting`

Prompt : Classify the sentiment of the following review as positive,
↪  negative, or neutral: 'The product exceeded my expectations!'
Answer : `Classification`

Prompt : What is the capital city of France?
Answer : `QuestionAnswering`

Prompt : Describe your ideal work environment
Answer : `OpenEnded`

Prompt : From the following news article, extract the names of the
↪  companies involved in the recent merger, along with the date the merger
↪  was announced.
Context:In a significant development in the tech industry, two leading
↪  companies have announced their merger, marking a new era of innovation
↪  and collaboration. The merger, which was officially announced on March
↪  15, 2025, brings together TechInnovate Inc. and DigitalSolutions Corp,
↪  two giants in their respective fields. TechInnovate Inc, known for its
↪  cutting-edge research and development in artificial intelligence and
↪  machine learning, has been at the forefront of technological
↪  advancements. With a team of over 5,000 engineers and scientists, the
↪  company has consistently delivered groundbreaking solutions that have
↪  transformed various industries. DigitalSolutions Corp, on the other
↪  hand, is renowned for its expertise in software development and digital
↪  transformation. The company has a proven track record of helping
↪  businesses across the globe to modernize their
```

```
 operations and enhance their digital capabilities. With a workforce of
 ↪  over 10,000 professionals, DigitalSolutions Corp. has been a key
 ↪  player in driving digital innovation. The merger is expected to create
 ↪  a powerhouse in the tech industry, combining the strengths of both
 ↪  companies to offer comprehensive solutions that address the evolving
 ↪  needs of businesses and consumers. The combined entity will leverage
 ↪  TechInnovate's AI and machine learning capabilities with
 ↪  DigitalSolutions' software development expertise to develop
 ↪  next-generation technologies. Industry analysts predict that this
 ↪  merger will lead to significant advancements in areas such as
 ↪  autonomous systems, smart cities, and personalized healthcare. The
 ↪  synergy between the two companies is anticipated to drive innovation,
 ↪  improve efficiency, and create new opportunities for growth.
Answer : `InformationExtraction`

Prompt : Given the following story, provide a title that summarizes the
↪  idea behind the story:
Context:
There once was a girl who was frustrated with life and asked her father for
↪  advice. He asked her to bring an egg, two tea leaves, and a potato. He
↪  then started boiling water in three separate vessels. He put the egg,
↪  potato, and tea leaves in one vessel each. After a few minutes, he
↪  asked her to peel the egg and potato and strain the leaves. He
↪  explained to his daughter that:
The soft egg was now hard.
The hard potato was now soft.
The tea had changed the water itself.
When adversity is at our door, we can respond to it in different ways.

Moral: We decide how to respond to difficult situations.
Answer : `Summarization`

Prompt : Fix the code below to correctly identify a palindrome:
def is_palindrome(str):
        return str == str[-1]
Answer : `CodeFix`

Prompt : John has one pizza, cut into eight equal slices. John eats three
↪  slices, and his friend eats two slices. How many slices are left?
↪  Explain your reasoning step by step.
Answer : `Reasoning`

Prompt : Exaggerate this product description : 'Our new sneakers are
↪  comfortable, lightweight, and stylish.' to a paragraph that can be used
↪  by the marketing team
Answer : `Rewrite`

Prompt : Use ABC notation to write a melody in the style of a folk tune.
Answer :
```

#### D.0.3 <format>

```
You are a helpful assistant whose goal is to classify the given prompt into
↪  a single class given the following definitions
```

```
`MCQAnswer` : Tasks related to multiple-choice type question answering.
↪  These prompts will typically contain multiple choices provided either
↪  in bullet form or eumerated numerically/alphabetically. This also
↪  contains multiple-choice question answer generation tasks.
`ChainOfThought` : Tasks related to Chain-of-Thought(CoT) style question
↪  answering. This also contains CoT style question answer generation
↪  tasks
`Enumeration` : Tasks that involve enumeration, bullet points, lists or
↪  itemization of any form
`XML` : Tasks that involve XML generation, validation or processing in any
↪  form
`Tabular` : Tasks that involve table generation, validation or processing
↪  in any form
`JSON` : Tasks that involve JSON generation, validation or processing in
↪  any form
`Markdown` : Tasks that involve Markdown generation, validation or
↪  processing in any form

If you are unable to confidently assign one of the above classes, you will
↪  simply respond with `Unspecified` and nothing else.


Note:
- You are only to respond with the name of the class you believe best
↪  matches the domain of the example.
- You are only allowed to classify the example into one of the following
↪  tags :
[`MCQAnswer`, `ChainOfThought`, `Enumeration`, `XML`, `Tabular`, `JSON`,
↪  `Markdown`, `Unspecified`]

Prompt : Use ABC notation to write a melody in the style of a folk tune.
Answer :
```

## NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

**The checklist answers are an integral part of your paper submission.** They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification:

   - Framework for controllability: "We create a detailed taxonomy of data characteristics and task provenance to explicitly control generation attributes and implicitly condition generations at inference time.": The taxonomy is provided in section 2.2, with the definition of training objective in section 2
   - Experimental scope: "We fine-tune a base model to infer these markers automatically, which makes them optional at inference time.": The method is described in section 2, the process of finetuning in section 2.3, and the empirical evidence for not requiring the markers in training time is provided in section 3.1
   - Long tail lift: "This principled approach yields pronounced improvements in performance on examples from the long tail of the training distribution. ": Experimental evidence is reported in section 3.2.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.

- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section 6 discusses the limitations.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: No theoretical result.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.

- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Finetuning settings are reported in section 2.3, base and LLM judge models are also specified including version. Evaluation paradigms are explained with each experiment section 2.3 and follow standard metrics of the respective benchmarks.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Justification: Code, model and data are not released.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).

- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: section 2.3 specify hyperparameters, with additional details about the evaluations in appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We only report significance tests for the machine translation experiments, in section 3.4.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We report the required compute and estimated time for each training run in Appendix B.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: There is no violation with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The broader impact is presented in the introduction in section 1, potential negative impacts are discussed with the limitations in section 6.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: No release of models or data.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All models and datasets are credited in section 2.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: No new assets introduced.

    Guidelines:

    - The answer NA means that the paper does not release new assets.

- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing or research with human subjects conducted.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No such studies conducted.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: We use LLMs for annotating the training markers used in our framework (§ 2.2) and during evaluation (§ 2.3.1).

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.

- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.