Automatic Paper Analysis and Categorisation for Systematic Reviews with Combined Reasoning-Augmented SFT and DAPO RL

Michela Lorandi **Anya Belz** **Simon Mille** **Craig Thomson**
DCU Natural Language Generation Research Group
ADAPT Research Centre, Dublin City University
Dublin, Ireland
{michela.lorandi,anya.belz}@adaptcentre.ie

Abstract

Systematic reviews are a cornerstone of modern science, synthesising evidence from published research to provide the highest level of research evidence in a field. The process includes categorising studies on a number of different dimensions which is laborious and time consuming. Automatic approaches are beginning to be explored but the complexity of the task means we are currently far from a satisfactory solution. In this paper, we test different annotation scheme-agnostic methods for automatic NLP paper categorisation for systematic reviews, and test them on two tasks: (i) annotating NLP papers for categories of reported controlled-text generation methods, and (ii) annotating NLP papers for categories of reported human evaluations. We find that reasoning-enhanced fine-tuning combined with DAPO reinforcement learning rewarding both correctness and output format substantially improves the performance of LLMs (by up to +53.8 points), even when they have been pre-trained to perform reasoning, and cuts time required for annotation by around 80% in a human-in-the-loop setting.

1 Introduction

Scientific reviews are crucial to modern science, providing the highest level of research evidence across many fields. They are, however, notoriously expensive and time-consuming, almost inevitably being out of date by the time they are published, especially for fast moving fields like computer science. The ability to automate (parts of) the systematic review process would substantially reduce the cost involved and make it easier for reviews to stay up to date.

As in many other contexts, the ever increasing capabilities of LLM-based systems offer the possibility of automating the task. However, so far it has proved too complex to achieve satisfactory levels of performance. In this paper, we test some of the most recent LLM techniques on the paper categorisation task in the context of conducting systematic reviews in computer science, more specifically of (i) the efficacy of alternative methods, and (ii) prevailing properties of system evaluations. We used the former during method development, while the latter served as a held-out test scenario.

Our main contributions are: (1) we formulate automatic paper categorisation as a reasoning-based classification problem agnostic to annotation scheme; (2) we introduce a low-resource setup (100 training samples) to test data-efficient fine-tuning with two types of categorisation tasks: (i) classifying controlled text generation (CTG) systems in terms of different types of techniques, and (ii) classifying evaluation experiments in terms of properties of evaluation methods; and (3) we evaluate reasoning-augmented supervised fine-tuning (SFT) (Yeo et al., 2025) and Decoupled Clip and Dynamic sAmpling Policy Optimisation (DAPO) (Yu et al., 2025) for structured category-label generation across different categorisation schemes.

The paper is structured as follows. We start with related research in Section 2, provide task definitions in Section 3, describe our methodology in Section 4 and the experimental set up in Section 5, before presenting results (Section 6) and concluding with some discussion (Section 8).

All code and results are available on GitHub: https://github.com/michelalorandi/sft_dapo_paper_annotation.

2 Related Research

Data annotation with LLMs. The ever-increasing performance of LLMs across language-based tasks (Wang et al., 2023; Liang et al., 2023; Zhang et al., 2024; Bavaresco et al., 2025; Zhang et al., 2025) have prompted researchers to explore their use as data annotators to reduce the high cost of manual annotation (Tan et al.,

Findings of the Association for Computational Linguistics: ACL 2026, pages 40676–40699

2024). Ding et al. (2023) evaluated GPT-3 as an annotator across multiple tasks, finding that while LLM-generated annotations perform worse than human annotations. Recent work has moved beyond simple label generation, e.g. Horych et al. (2025) showed that classifiers trained on LLM annotations for media bias detection can match models trained on human annotations, albeit with reduced robustness. Wang et al. (2024) proposed a hybrid approach where LLMs generate initial annotations with explanations, verifiers assess quality, and humans re-annotate uncertain cases.

Most relevant to our work are recent efforts applying LLMs to the task of annotating scientific documents. Ruan et al. (2024) demonstrated LLM-based edit intent classification of scientific papers as a 5-class sentence-level task. Li et al. (2024) explored continual pre-training for scientific paper parsing and multi-task classification, covering entity extraction, table extraction, and question answering across multiple scientific domains. Similar to our work, their tasks require structured JSON outputs, though they rely solely on SFT without RL refinement. We extend this work by addressing structured annotation generation with LLMs using explicit reasoning traces and DAPO reinforcement learning for multi-field paper categorisation.

RL for reasoning tasks. RL has become a key technique for optimising LLMs beyond human feedback alignment, with recent applications to structured prediction tasks. A key advance was Group Relative Policy Optimisation (GRPO) (Shao et al., 2024) introducing group-level reward statistics to achieve improved performance on reasoning tasks. However, as published it exhibits considerable shortcomings which Dynamic sAmpling Policy Optimisation (DAPO) (Yu et al., 2025) set out to address, proposing the Clip-Higher strategy to promote diversity and avoid entropy collapse; dynamic sampling for training efficiency and stability; a token-Level policy gradient loss for long-CoT RL; and overlong reward shaping to reduce reward noise and stabilise training.

RL for classification. Mao et al. (2019) formulated hierarchical text classification as a Markov decision process, learning a label assignment policy via deep RL to determine where to place objects in the label hierarchy and when to stop. Chai et al. (2020) proposed generating task-specific category descriptions via RL to improve text classification, formulating classification as question answering and optimising descriptions with REINFORCE. Ye et al. (2020) introduced reinforced self-training for zero-shot text classification, using RL to learn data selection policies that automatically identify high-quality pseudo-labeled instances from unlabeled data. Jia et al. (2024) demonstrated that combining RL with supervised learning effectively handles multi-label positive-unlabelled learning, using local and global rewards to address incomplete annotations in document relation extraction.

More recently, Zhao et al. (2025) introduced 'Trustworthiness-as-Reward,' combining SFT and RL with reasoning process evaluation to improve multi-label classification performance. Our work extends this paradigm to complex structured annotation tasks with hierarchical schemes and variable-length outputs, using reasoning traces and multi-component reward functions to optimise for both accuracy and structural validity.

3 Task Definition

Our goal is to automate the annotation of complex documents for properties of processes and methods reported in them. Our intended application is the annotation of scientific papers for systematic surveys. We aim to train models capable of analysing papers and producing structured annotations according to a given categorisation scheme. Achieving this would enable scalable and up-to-date meta-analyses of the research landscape, and substantially reduce the manual effort involved.

We frame the problem as a low-resource annotation task, a realistic setting given that if a large manually annotated dataset has to first be constructed for each new annotation scheme, the value of automation is reduced. This setup also allows us to study the data efficiency of reasoning-augmented fine-tuning and Reinforcement Learning (RL).

We explore two distinct paper annotation tasks with fundamentally different structural properties: (1) **CTG system categorisation** of different techniques in systems (used during development of our approach); and (2) **evaluation method categorisation**, used for held-out testing of generalisation. The CTG categorisation task has a fixed 4-field structure where most fields involve multi-label predictions, while the evaluation categorisation task involves variable-length structured prediction where the model must detect an unknown number of measures and classify each on multiple dimensions.

To characterise the complexity of this task, we

Task	Field	Class	Label	Set
CTG	Control Type	Single	Binary	Closed
	Control Implementation	Multi	Multi	Closed
	Control Integration	Multi	Multi	Closed
	Control Attributes	Multi	Multi	Open
Eval	Evaluation Type paper	Multi	Multi	Closed
	Evaluation Type measure	Multi	Multi	Closed
	QCET node	Multi	Multi	Closed

Table 1: Characteristics of annotation tasks. **Class**: Single=single-label (one value per instance), Multi= multi-label (multiple values per instance). **Label**: Binary=2 possible values, Multi=3+ possible values. **Set**: Closed=fixed taxonomy, Open=unbounded vocabulary.

analyse each annotation field along three standard dimensions of classification problems. Fields may involve *single-label* classification (assigning one value per paper) or *multi-label* classification (assigning multiple values per paper). They may be *binary* (two possible classes) or *multi-class* (three or more possible classes). Finally, they may involve *closed-set* classification (selecting from a fixed, predefined taxonomy) or *open-set* classification (extracting labels from an unbounded vocabulary within the paper, requiring normalisation).

3.1 CTG categorisation scheme

We adopt a four-dimensional categorisation scheme to characterise CTG systems:

1. **Control Type**: whether the system controls a single attribute or multiple attributes.
2. **Control Implementation**: the method used to enable control in the CTG system.
3. **Control Integration**: the level at which the control signal is applied in the architecture.
4. **Control Attributes**: the semantic or stylistic properties of text controlled by the system.

Table 1 shows the annotation characteristics of each field. More details in Appendix A.1.

3.2 Task formulation

Each annotation task is formulated as a conditional text generation problem. Given the text of a paper x, a property-value question q about the system proposed in the paper, and the task description t, the model must generate a reasoning trace r, that provides step-by-step justification by referencing evidence within the paper, and structured annotations a in JSON format following the scheme given in q (Section 3.1), as shown in Figure 1. The reasoning trace provides a step-by-step justification

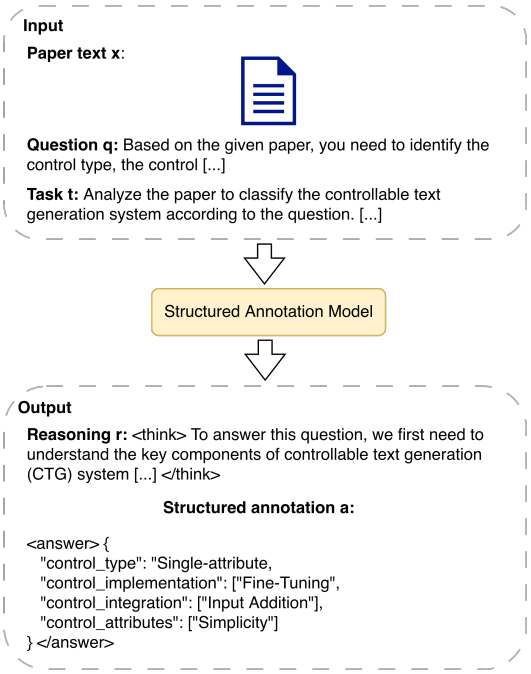


Figure 1: Task formulation for automatic paper categorisation. The model receives the paper text x, a property-value question q, and task description t as input, and generates a reasoning trace r explaining the decision process followed by structured annotations a in JSON format containing categorisation labels for all fields.

of the final decision, supporting transparency and interpretability. By explicitly referencing evidence within the paper, the reasoning traces help human evaluation verification of annotation correctness and provide pointers to the paper regions that support the final decision.

4 Methodology

Our approach consists of the three phases: (i) dataset creation with manual annotation of gold labels and synthetic reasoning traces, (ii) Supervised Fine-Tuning (SFT) with long Chain-of-Thought (CoT), and (iii) Reinforcement Learning (RL) refinement using DAPO.

4.1 Dataset creation

To train and evaluate our proposed annotation system, we first construct a high-quality, manually annotated dataset of CTG papers. The research context for this and the evaluation categorisation effort (Section 7) is to conduct a systematic review of controlled text generation methods and prevailing evaluation methods in NLP, respectively.

Paper collection. We search the ACL Anthology to identify all papers related to CTG published up to

December 2023. Following a systematic screening process, we keep only those papers that propose a new CTG system. This results in a corpus of 140 papers in total. Among these, 110 papers published up to August 2023 are used as training set, while the remaining 30 papers published between September and December 2023 serve as test set. More details can be found in Appendix A.2.

Manual annotation. All selected papers are manually annotated according to the categorisation scheme from Section 3.1 on four dimensions: control type, control implementation, control integration, and control attributes.

Paper preprocessing. To enable models to focus on core paper content and to reduce input size, each paper is converted to text using Docling[1] and cleaned in the following sense: we remove non-informative sections such as appendices, references, ethical checklists, limitations, and the list of authors. We then clean LaTeX artefacts such as formulas or images not decoded. The resulting corpus contains only the title, abstract, and main body of each paper. This preprocessing assumes papers contain all relevant methodological information in the main body. Resource constraints prevented using full papers (up to 87k tokens): preprocessed papers already average 9k tokens (maximum 14k), and RL training required three NVIDIA A100 GPUs (80GB each). Alternative retrieval-based approaches have shown inferior performance on related tasks (Pronesti et al., 2025). Two papers were excluded due to extraction errors, leaving 99 training papers, 10 validation papers, and 29 test papers.

Synthetic reasoning trace generation. To fine-tune the model with SFT, we augment each annotated instance with a *reasoning trace* that explains how the correct answer can be derived from the paper. We use LLaMa 3.3 70B Instruct (Grattafiori et al., 2024) to generate synthetic reasoning traces for every paper-question pair. The generation process uses temperature 0.6, maximum length 4,096 tokens, and the prompt shown in Figure 4 in the Appendix. Each generated trace is automatically validated before inclusion in the dataset: (i) it must contain properly formatted `<think>...</think>` and `<answer>...</answer>` tags; (ii) it must produce a valid JSON output; and (iii) the predicted labels must match the expected gold annotations. Traces failing any check are regenerated until a valid instance is produced (all instances successfully produced valid reasoning traces within a maximum 6 attempts). For an example of a generated reasoning trace, see Appendix, Figure 5. Compute statistics for the generated reasoning traces are reported in Appendix A.4.

[1] https://www.docling.ai/

4.2 Model architecture

We train a multi-field model that predicts the four annotation dimensions jointly, rather than separate single-field models. This enables the model to capture inter-dependencies between annotation dimensions and provides computational efficiency.

The multi-field formulation is particularly suited to RL fine-tuning because it naturally provides dense reward signals. In the single-field setting, correctness rewards are necessarily binary (correct or incorrect), leading to sparse gradients. However, with multi-field outputs, even partially correct predictions (e.g., correct on 2 out of 4 dimensions) receive intermediate rewards, enabling more stable and sample-efficient RL training.

The model receives as input the concatenation of paper text x and a composite question q describing all four dimensions. The output is a structured JSON object containing all four fields, as shown in Figure 1.

4.3 SFT with long CoT

We first fine-tune the given pre-trained model with LoRA using a supervised objective on the reasoning-augmented samples from the last section. Each training instance is represented as a tuple (x, q, y), where x is the text content of the research paper, q is the question defining the annotation task, and y is the target output structured as follows:

$$y = \texttt{<think>}r\texttt{</think><answer>}a\texttt{</answer>}$$

where r is the reasoning trace from Section 4.1, and a is the structured JSON annotation containing the target labels.

Given model $p_\theta(y \mid x, q)$ parametrised by θ, we optimise the standard autoregressive supervised loss:

$$\mathcal{L}_{\text{SFT}}(\theta) = -\sum_{t=1}^{T} \log p_\theta(y_t \mid y_{<t}, x, q)$$

where T is the number of tokens in the output. This objective encourages the model to condition on both input paper x and question q to produce coherent reasoning and structured annotations.

This *long chain-of-thought* SFT (Yeo et al., 2025; Chen et al., 2025) trains the model to generate extended reasoning traces before structured annotations. Each training instance includes a synthetic reasoning trace (see Figure 5 in Appendix) that systematically analyses the paper, identifies supporting evidence, and justifies classification decisions step-by-step. The model learns to replicate this process.

4.4 RL for structured annotation generation

To further improve structured annotation quality, we apply RL on top of both the SFT checkpoint or the base model. We use Group Relative Policy Optimisation (GRPO) (Shao et al., 2024) with the **DAPO** (Decoupled Clip and Dynamic sAmpling Policy Optimisation) (Yu et al., 2025) loss formulation, which enables stable policy optimisation using group-relative rewards without requiring a learned value function.

The model acts as a policy $\pi_\theta(y \mid x, q)$ that generates a structured output y given a paper x and query q. Each output consists of a reasoning trace enclosed in a `<think>...</think>` block, followed by a JSON object encoding the annotations using the same scheme as in SFT. For each input (x, q), we sample G candidate completions $\{y_i\}_{i=1}^{G}$ from the current policy (with G=8). Each completion is scored using rule-based reward functions measuring annotation correctness and structural validity (see below).

Rewards are normalised within each group of G completions to obtain a relative advantage:

$$\hat{A}_{i,t} = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^{G})}{\text{std}(\{R_j\}_{j=1}^{G}) + \epsilon}.$$

The DAPO loss formulation[2] applies token-level normalisation to handle variable-length sequences, assigning balanced rewards to individual tokens:

$$\mathcal{L}_{\text{DAPO}}(\theta) = -\frac{1}{\sum_{i=1}^{G}|o_i|}\sum_{i=1}^{G}\sum_{t=1}^{|o_i|}\Big(p_{i,t}(\theta)\,\hat{A}_{i,t}\Big),$$

where o_i is the i-th completion, $\hat{A}_{i,t}$ is the advantage for token t, and $p_{i,t}(\theta)$ is the token-level probability ratio defined as:

$$p_{i,t}(\theta) = \frac{\pi_\theta(o_{i,t}|q, o_{i,<t})}{[\pi_\theta(o_{i,t}|q, o_{i,<t}))]_{\text{no grad}}}.$$

[2] https://huggingface.co/docs/trl/main/grpo_trainer

Reward function. We design a composite reward function that evaluates three aspects of the generated output. The **correctness reward** measures annotation accuracy using either exact match or Jaccard similarity averaged across all fields F:

$$R_{\text{correct}} = \frac{1}{|F|}\sum_{f\in F}\begin{cases}\mathbb{1}\{A_f^{\text{pred}} = A_f^{\text{gold}}\} & \text{(exact match)}\\ \frac{|A_f^{\text{pred}}\cap A_f^{\text{gold}}|}{|A_f^{\text{pred}}\cup A_f^{\text{gold}}|} & \text{(Jaccard)}\end{cases}$$

where A_f^{pred} is the predicted, and A_f^{gold} the gold label set. The **format reward** R_{format} checks for the presence and correct ordering of structural tags (`<think>`, `</think>`, `<answer>`, `</answer>`): $R_{\text{format}} = \frac{n_{\text{correct}}}{n_{\text{total}}}$.

The **JSON reward** R_{json} verifies that the answer contains valid, parsable JSON with all expected fields. Both are normalised to $[0, 1]$ by counting correct components: $R_{\text{json}} = \frac{n_{\text{valid}}}{n_{\text{expected}}}$.

The final reward is a weighted combination: $R = w_{\text{correct}}R_{\text{correct}} + w_{\text{format}}R_{\text{format}} + w_{\text{json}}R_{\text{json}}$, where we set $w_{\text{correct}} = 0.7$, $w_{\text{format}} = 0.05$, and $w_{\text{json}} = 0.25$ to prioritise prediction accuracy while maintaining structural validity. We experimented with both Jaccard similarity and exact match for the correctness reward, finding that exact match provides better training signal (see Section 6.2).

5 Experimental Setup

Evaluation metrics. We use standard evaluation methods depending on task type. For **Binary classification** (Control Type), we report overall Accuracy and F1; for **Multi-label classification** (Control Implementation, Control Integration, Control Attributes), we report Exact Match (strict correctness), Jaccard score (intersection-over-union between gold and predicted label sets) and Micro F1. All metrics use strict string matching without normalisation (e.g., "sentiment" and "sentiments" are treated as distinct). For the open-vocabulary Control Attributes field, we do not apply ontology mapping or embeddings-based semantic matching, opting instead for conservative exact-match evaluation to avoid overestimating performance through automatic alignment errors.

Training Setup. All training runs are performed on two NVIDIA A100 GPUs (80 GB each). We use Qwen3-8B (Team, 2025) as the base model, fine-tuned with LoRA, which is configured with a rank of 32, and scaling factor $\alpha = 64$. Three training regimes are explored: (i) SFT without reasoning traces, where the model is trained to

generate only the final structured annotation; (ii) reasoning-augmented SFT using the synthetic reasoning traces described in Section 4.1; and (iii) RL with DAPO applied on top of the reasoning-augmented SFT checkpoint. This comparison allows us to assess the value of explicit reasoning traces for structured annotation tasks. Both SFT and RL are trained with batch size 1, gradient accumulation 8. For SFT, we train for up to 30 steps with learning rate 2×10^{-4}. For RL, we train for up to 3 epochs with number of generations per sample 8, and learning rate 1×10^{-5}. All hyperparameters and training details are provided in Appendix C.

Model Baselines. We compare our approach against a range of pre-trained open-source LLMs evaluated in a zero-shot setting due to the long input lengths of the papers. Specifically, we test Qwen3 (8B, 14B, 70B), and LLaMA 3.3 (8B, 70B). We further distinguish between models prompted to produce explicit reasoning traces (*reasoning models*) and those prompted for direct answers (*non-reasoning models*). All baseline hyperparameters and prompting templates are detailed in Appendix C.1.

6 CTG Categorisation Results

6.1 Main results

Table 2 presents results for the different models on the CTG annotation task, separately for the four annotation dimensions: Control Type (binary classification), Control Attributes, Control Integration, and Control Implementation (all multi-label).

Among off-the-shelf models, larger models generally achieve better performance. LLaMA 3.3 70B Instruct achieves the strongest baseline results. Performance on the more challenging multi-label fields is poor, particularly on Control Attributes.

Fine-tuning Qwen3 8B yields substantial improvements over both the base model and larger pre-trained alternatives. Non-reasoning SFT achieves 68.3% accuracy on Control Type, but reasoning-augmented SFT provides significant additional gains: +3.4 points on Control Type (71.7%), +9.0 points EM on Control Attributes (32.4%), and +10.3 points EM on Control Implementation (40.0%). These improvements are particularly pronounced on multi-label fields, validating that explicit reasoning traces provide stronger training signal for complex structured annotation tasks.

Training with RL alone (starting from the base model) achieves comparable performance to non-reasoning SFT. However, RL-only training performs worse than reasoning-augmented SFT on Control Attributes, suggesting that the learned reward signal alone is insufficient to match supervised learning from synthetically annotated reasoning traces.

Our best results come from the SFT+RL approach, which applies RL on top of the reasoning-augmented SFT checkpoint, achieving 84.1% accuracy on Control Type (+12.4 points over reasoning-augmented SFT alone), and substantial gains on multi-label fields. Improvement is particularly pronounced on Exact Match metrics, indicating that RL helps the model generate more completely correct multi-label predictions rather than just partial matches. The Jaccard scores show similar trends, with SFT+RL achieving the best performance across all multi-label fields. These results demonstrate that reasoning-augmented fine-tuning followed by RL refinement enables effective structured prediction even with severely limited training data, substantially outperforming both pre-trained baselines and single-stage training approaches.

6.2 Ablation study

Table 7 in the Appendix compares different reward formulations: Exact Match (EM) vs. Jaccard (J) for correctness, with optional format (F) and JSON validity (JSON) rewards. We also compare RL-only (training from base model) and SFT+RL (training from SFT checkpoint) settings.

The results show that first performing SFT is essential: SFT+RL consistently outperforms RL-only by 14–17% on Control Type across all reward configurations. EM and Jaccard rewards show clear trade-offs: EM yields higher Exact Match scores (63.4% vs. 60.7% on Control Integration), while Jaccard excels on partial-credit metrics (64.4% vs. 63.2% on Control Implementation). Adding format and JSON rewards reduces reasoning length from 656 to 563 tokens while maintaining performance, indicating RL learns more focused explanations. However, multi-component rewards work well after SFT but cause interference when training from scratch (compare rows 5 and 8 in Table 7).

We adopt EM+F+JSON as our final configuration, achieving 84.1% on Control Type and the shortest reasoning traces (563 tokens) while maintaining strong multi-label performance.

Single-field vs. Multi-field models. To validate our multi-field architecture, we compare it against

Model	R	Control		Attributes			Integration			Implementation		
		Acc	F1	EM	F1	J	EM	F1	J	EM	F1	J
Pre-trained LLMs												
Qwen3 8B	✓	30.3	30.3	8.3	29.7	13.8	16.6	36.5	23.3	15.2	46.7	26.7
Qwen3 14B	✓	62.1	62.1	24.8	49.2	36.6	29.0	51.8	42.4	23.4	63.2	49.4
Qwen3 32B	✓	68.3	68.3	21.4	43.5	31.7	32.4	54.7	45.3	21.4	57.7	44.5
LLaMa3.1 8B I	×	34.5	34.5	10.3	37.6	23.6	25.5	35.6	30.0	11.0	33.8	25.7
LLaMa3.3 70B I	×	*74.5*	*74.5*	20.0	44.9	35.2	*47.6*	*62.1*	*58.3*	23.4	62.9	51.6
Our systems												
Qwen3 8B SFT only	×	68.3	68.3	23.4	49.4	35.5	35.9	49.5	42.9	29.7	55.8	46.8
Qwen3 8B SFT only	✓	71.7	71.7	*32.4*	*53.7*	*41.2*	36.6	46.2	44.0	*40.0*	65.4	*59.3*
Qwen3 8B RL only	✓	67.6	67.6	24.1	51.8	35.1	40.7	56.0	50.0	37.2	**69.8**	58.7
Qwen3 8B SFT+RL	✓	**84.1**	**84.1**	**34.5**	**54.5**	**42.4**	**62.8**	**66.7**	**65.9**	**46.2**	*67.1*	**62.0**

Table 2: Performance on CTG system annotation across four dimensions: Control Type (binary classification), Control Attributes, Control Integration, and Control Implementation (multi-label). Results are averaged over 5 runs. **Bold** indicates best result, *italics* second-best. R=reasoning trace; EM=Exact Match; J=Jaccard score (intersection-over-union).

training separate single-field models for each annotation dimension. The multi-field approach consistently outperforms single-field alternatives while achieving 3x faster inference (39.9s vs 120.4s per paper) by producing all annotations in a single forward pass. This efficiency stems from shared reasoning context and learned inter-field dependencies. See Appendix D for detailed results.

	✓	Partial	×
Control Type	25	-	5
Control Attributes	19	6	5
Control Implementation	21	7	2
Control Integration	27	0	3

Table 3: Manual evaluation results on 30 unseen papers from 2024 using our best model (SFT+RL with EM+F+JSON). Partial=some labels correct. No partial results for Control Type, as it is binary classification.

Model	R	Evaluation Type		
		EM	F1	J
Pre-trained LLMs				
Qwen3 8B	✓	68.7	80.3	75.7
Qwen3 14B	✓	71.3	81.9	78.7
Qwen3 32B	✓	6.7	20.7	10.1
LLaMa3.1 8B I	×	4.0	48.7	30.1
LLaMa3.3 70B I	×	53.3	74.3	70.4
Ours				
Qwen3 8B SFT only	✓	*75.3*	*84.7*	*81.7*
Qwen3 8B SFT+RL	✓	**78.7**	**85.4**	**83.0**

Table 4: Paper-level evaluation type classification results, averaged over 5 runs. **Bold** indicates best result, *italics* second-best. R=Reasoning trace; EM=Exact Match; J=Jaccard score (intersection-over-union).

6.3 Manual evaluation

To further validate our approach on completely unseen data, we evaluate our best-performing model (SFT+RL with EM+F+JSON rewards) on 30 additional papers published in 2024. These papers are sampled following the same selection criteria used for the main dataset (Section 4.1). Rather than creating gold annotations independently, we leverage the model-generated reasoning traces followed by human validation: the model produces structured annotations with step-by-step justifications, which we then verify for correctness. This approach allows us to assess model performance on recent work while using the generated reasoning traces as an annotation aid, demonstrating a practical use case where the model assists rather than replaces human annotators.

The results in Table 3 show that the model achieves strong performance on Control Type and Control Integration with 25/3 (83.3%) and 27/30 (90%) of outputs completely correct, respectively. For multi-label fields, it produces fully correct predictions in 19/30 (Attributes) and 21/30 (Implementation) cases, with partially correct predictions (some labels identified) in 6/30 and 7/30 cases, respectively. This pattern indicates the model reliably captures primary characteristics while occasionally missing secondary details. The reasoning traces facilitated validation by enabling rapid error identification in partially correct cases. Overall performance on 2024 papers closely matches test set results, confirming good generalisation to more recent publications, and demonstrating practical utility for semi-automated annotation workflows. Validating model-generated annotations required $\sim$10 minutes per paper (5 hours for 30 papers) compared to 45-60 minutes for manual annotation from scratch, achieving a 5-6x speed-up.

Model	R	Evaluation Type		
		EM	F1	J
Pre-trained LLMs				
Qwen3 8B	✓	13.3	55.4	41.8
Qwen3 14B	✓	15.3	46.2	35.6
Qwen3 32B	✓	17.3	44.8	34.3
LLaMa3.1 8B I	×	3.3	31.1	17.9
LLaMa3.3 70B I	×	19.3	*66.8*	54.7
Ours				
Qwen3 8B SFT only	✓	*28.0*	**67.3**	**58.2**
Qwen3 8B SFT+RL	✓	**30.7**	65.4	*57.8*

Table 5: Measure-level evaluation type classification results, averaged over 5 runs. **Bold** indicates best result, *italics* second-best. R=Reasoning trace; EM=Exact Match; J=Jaccard score (intersection-over-union).

7 Evaluation Categorisation

To demonstrate generalisation, we test our approach to new task types not seen during development. We apply it to a second annotation task type focused on evaluation methods.

7.1 Evaluation categorisation scheme

This scheme categorises papers based on two aspects. **Evaluation type** identifies the methodology used to assess system output quality. Evaluation type can be annotated at two levels of granularity: (i) paper-level, where a multi-label classification identifies the set of evaluation types used in the paper; (ii) measure-level, where each distinct evaluation measure or criterion used in the paper is identified and individually classified.

At the measure level, we additionally annotate evaluation measures using the Quality Criteria for Evaluation Taxonomy (**QCET**) (Belz et al., 2025), which provides a hierarchical framework for characterising what quality aspects evaluation measures assess. QCET consists of three main hierarchy levels that progressively narrow the scope of the quality criterion. More details in Appendix B.1.

We evaluate three classification settings of increasing complexity: (1) **paper-level evaluation type**; (2) **measure-level evaluation type**; and (3) **measure-level QCET classes**. Combining evaluation type and QCET level annotations is unlikely to be helpful for the evaluation method annotation tasks, as they are entirely orthogonal.

7.2 Experimental setup

Following the same procedure as in Section 4.1, we created an evaluation method annotation dataset with 109 training and 30 test papers. Papers are sampled by a systematic process, annotated with gold labels for evaluation type and QCET categories, and augmented with synthetic reasoning traces. See Appendix B for paper selection process, annotation process, and dataset statistics. See Appendix C for hyperparameters used.

7.3 Results

Table 4 shows results for paper-level evaluation type, the least complex of the evaluation method categorisation tasks. Interestingly, larger pre-trained models do not always show better instruction-following: Qwen3 32B achieves only 6.7% EM due to frequent failures to generate outputs in the required format, while the smaller Qwen3 8B and 14B variants successfully follow formatting instructions. This highlights a key challenge in deploying larger models for structured output tasks without task-specific training. Our SFT approach addresses this issue, boosting Qwen3 8B to 75.3% EM with perfect format compliance, and adding RL further improves performance to 78.7% EM.

Table 5 shows results for measure-level evaluation type classification. This variable-length task proves substantially harder: pre-trained models achieve only 13.3-19.3% EM compared to 53.3-71.3% for the paper-level version of the task. Our SFT approach reaches 28.0% EM, surpassing LLaMA3.3 70B by 8.7 points, with strong partial-credit metrics indicating correct classification of many individual measures. RL refinement further improves performance to 30.7% EM, achieving the highest exact match while maintaining competitive F1 and Jaccard scores.

Table 6 shows results for measure-level QCET classification across three hierarchical levels. The task's complexity is evident: none of the pre-trained models achieve any exact matches (all three levels correct for all measures). Even LLaMA3.3 70B, despite reasonable Level 1 performance, fails completely on Level 3, indicating particular difficulty with fine-grained quality aspect classification. Our SFT approach achieves 14.0% complete EM, the only model to achieve non-zero EM, and consistently outperforms baselines across the hierarchy: +8.0 points on Level 1, +10.7 points on Level 2, and +21.3 points on Level 3. Partial-credit metrics (51.7-55.0% F1) demonstrate strong performance on individual measures despite the challenging multi-level structure. RL refinement substantially improves performance to 18.0% complete EM (+4.0 points), with particularly strong gains on

Model	R	QCET node											
		Complete			Level 1			Level 2			Level 3		
		EM	F1	J	EM	F1	J	EM	F1	J	EM	F1	J
Pre-trained LLMs													
Qwen3 8B	✓	0.0	0.3	0.0	12.0	25.4	20.1	6.7	21.9	15.7	*0.0*	2.3	0.7
Qwen3 14B	✓	0.0	0.0	0.0	2.7	4.1	3.6	2.0	3.8	2.8	*0.0*	0.3	0.1
Qwen3 32B	✓	0.0	0.3	0.0	14.7	34.0	28.5	7.3	30.6	22.9	*0.0*	4.1	0.9
LLaMa3.1 8B I	×	0.0	0.5	0.2	3.3	17.5	10.8	4.0	24.9	16.1	*0.0*	2.9	0.9
LLaMa3.3 70B I	×	0.0	0.9	0.2	18.0	52.1	43.3	8.0	*53.0*	40.4	0.0	7.5	2.1
Ours													
Qwen3 8B SFT only	✓	*14.0*	*33.5*	*29.8*	*26.0*	*55.0*	*48.7*	*18.7*	51.7	*42.1*	*21.3*	*53.8*	*45.8*
Qwen3 8B SFT+RL	✓	**18.0**	**40.8**	**35.9**	**28.0**	**58.8**	**51.9**	**19.3**	**54.1**	**43.2**	**33.3**	**68.8**	**61.2**

Table 6: Measure-level QCET node classification results, averaged over 5 runs. **Bold** indicates best result, *italics* second-best. R=Reasoning trace; EM=Exact Match; J=Jaccard score (intersection over union).

Level 3 (+12.0 points to 33.3% EM). Unlike Tables 4 and 5, RL optimisation improves both exact match and partial-credit metrics on this hierarchical task (40.8% F1 complete, 58.8-68.8% F1 across levels), demonstrating effective learning across the full structure.

8 Conclusion

We have presented a reasoning-augmented approach combining SFT and RL with DAPO to automate paper annotation for systematic reviews. Our method addresses the critical bottleneck of manual annotation, achieving high quality through explicit reasoning traces that provide both performance improvements and human-verifiable transparency.

Across two annotation tasks with varying complexity, our SFT+RL approach substantially outperforms strong pre-trained baselines including 10x larger models, and uniquely succeeds on challenging hierarchical tasks. The multi-field architecture captures inter-field dependencies while providing 3x computational speed-up compared to separate single-field models. Our results demonstrate that SFT initialisation is essential for effective RL training, consistently outperforming RL-only approaches. The generated reasoning traces prove valuable beyond model training, enabling rapid human validation in practical semi-automated workflows demonstrated on 30 recent papers.

Our approach generalises across different annotation schemes and task complexities, requiring only $\sim$100 training examples to achieve strong performance. An avenue for future work is to aim for further practical utility by extending reasoning traces with pointers to exact locations of supporting evidence in papers, enabling even more time savings in human-verified systematic review workflows.

Limitations

Our work has some limitations that should be acknowledged and provide directions for future research.

Model coverage: We evaluate only open-source models (Qwen3 and LLaMA3) due to the high costs of running extensive experiments with closed-source LLMs such as GPT-4 or Claude. While our open-source baselines demonstrate the effectiveness of our approach, closed-source models might achieve different performance levels or respond differently to our training methodology. Future work with larger budgets could explore whether similar improvements hold for proprietary models.

Parameter-efficient fine-tuning: We use LoRA as our parameter-efficient fine-tuning (PEFT) technique to reduce computational requirements and resource consumption. LoRA is currently the de facto standard for PEFT, used far more widely than alternatives such as Adapters or Prefix-Tuning, and proves effective in our experiments. While LoRA proves effective in our experiments, other PEFT methods such as Adapters, or Prefix-Tuning might yield different performance trade-offs. Additionally, full fine-tuning of all model parameters, rather than just LoRA adapters, could potentially achieve better results but was not feasible within our computational budget. The relative benefits of different PEFT approaches and full fine-tuning remain open questions for future investigation.

Dataset size and domain: Our low-resource setting uses approximately 100 training papers per task, reflecting realistic annotation budgets in academic research. However, the performance of the approach with larger training sets remains unexplored. Additionally, while we demonstrate generalisation across two distinct annotation tasks within

NLP, applicability to other scientific domains (e.g., medicine, social sciences) has not been tested.

Reasoning trace quality: Our synthetic reasoning traces are generated using LLaMA3.3 70B and validated automatically against gold annotations. While this approach produces traces that improve model performance and facilitate human validation, the quality and diversity of reasoning patterns may be limited by the capabilities of the trace generation model. Human-written reasoning traces or traces from stronger models might yield further improvements.

Computational requirements: Despite using LoRA to reduce resource consumption, our approach still requires multiple high-end GPUs for training. This may limit accessibility for researchers with more constrained computational resources.

Ethical Considerations

Our work uses publicly available papers from the ACL Anthology (Creative Commons Attribution 4.0 License) and open-source models in accordance with their licenses. The system is designed to assist, not replace, human researchers in systematic reviews. Our semi-automated workflow explicitly requires human validation of model outputs, which remains essential for quality and reliability.

Despite requiring substantial computational resources for training, our approach is significantly more efficient than full model training (using LoRA with $\sim$100 examples) and dramatically reduces annotation time compared to purely manual approaches. We provide explicit reasoning traces for transparency and detailed experimental documentation to support reproducibility.

The potential broader impact includes democratising systematic review capabilities and enabling more frequent meta-analyses, which could particularly benefit researchers with limited resources. However, responsible deployment requires maintaining human oversight to ensure annotation quality and diverse perspectives in literature analysis.

References

Anna Bavaresco, Raffaella Bernardi, Leonardo Bertolazzi, Desmond Elliott, Raquel Fernández, Albert Gatt, Esam Ghaleb, Mario Giulianelli, Michael Hanna, Alexander Koller, Andre Martins, Philipp Mondorf, Vera Neplenbroek, Sandro Pezzelle, Barbara Plank, David Schlangen, Alessandro Suglia, Aditya K Surikuchi, Ece Takmaz, and Alberto Testoni. 2025. LLMs instead of human judges? a large scale empirical study across 20 NLP evaluation tasks. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 238–255, Vienna, Austria. Association for Computational Linguistics.

Anya Belz, Simon Mille, and Craig Thomson. 2025. The qcet taxonomy of standard quality criterion names and definitions for the evaluation of nlp systems. *arXiv preprint arXiv:2509.22064*.

Duo Chai, Wei Wu, Qinghong Han, Fei Wu, and Jiwei Li. 2020. Description based text classification with reinforcement learning. In *International conference on machine learning*, pages 1371–1382. PMLR.

Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wanxiang Che. 2025. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *CoRR*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Bosheng Ding, Chengwei Qin, Linlin Liu, Yew Ken Chia, Boyang Li, Shafiq Joty, and Lidong Bing. 2023. Is GPT-3 a good data annotator? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11173–11195, Toronto, Canada. Association for Computational Linguistics.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, and 1 others. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Tomáš Horych, Christoph Mandl, Terry Ruas, Andre Greiner-Petter, Bela Gipp, Akiko Aizawa, and Timo Spinde. 2025. The promises and pitfalls of LLM annotations in dataset labeling: a case study on media bias detection. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 1370–1386, Albuquerque, New Mexico. Association for Computational Linguistics.

David Howcroft, Anya Belz, Miruna Clinciu, Dimitra Gkatzia, Sadid Hasan, Saad Mahamood, Simon Mille, Sashank Santhanam, Emiel van Miltenburg, and Verena Rieser. 2020. Twenty years of confusion in human evaluation: NLG needs evaluation sheets and standardised definitions. In *Proceedings of the 13th International Natural Language Generation Conference*.

Zixia Jia, Junpeng Li, Shichuan Zhang, Anji Liu, and Zilong Zheng. 2024. Combining supervised learning

and reinforcement learning for multi-label classification tasks with partial labels. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13553–13569, Bangkok, Thailand. Association for Computational Linguistics.

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

Sihang Li, Jin Huang, Jiaxi Zhuang, Yaorui Shi, Xiaochen Cai, Mingjun Xu, Xiang Wang, Linfeng Zhang, Guolin Ke, and Hengxing Cai. 2024. SciLitLLM: How to adapt LLMs for scientific literature understanding. In *Neurips 2024 Workshop Foundation Models for Science: Progress, Opportunities, and Challenges*.

Percy Liang, Rishi Bommasani, Tony Lee, Dimitris Tsipras, Dilara Soylu, Michihiro Yasunaga, Yian Zhang, Deepak Narayanan, Yuhuai Wu, Ananya Kumar, Benjamin Newman, Binhang Yuan, Bobby Yan, Ce Zhang, Christian Cosgrove, Christopher D. Manning, Christopher Ré, Diana Acosta-Navas, Drew A. Hudson, and 31 others. 2023. Holistic evaluation of language models. *Trans. Mach. Learn. Res.*, 2023.

Yuning Mao, Jingjing Tian, Jiawei Han, and Xiang Ren. 2019. Hierarchical text classification with reinforced label assignment. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 445–455, Hong Kong, China. Association for Computational Linguistics.

Massimiliano Pronesti, Michela Lorandi, Paul Flanagan, Oisín Redmond, Anya Belz, and Yufang Hou. 2025. Enhancing study-level inference from clinical trial papers via reinforcement learning-based numeric reasoning. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 30357–30373, Suzhou, China. Association for Computational Linguistics.

Qian Ruan, Ilia Kuznetsov, and Iryna Gurevych. 2024. Are large language models good classifiers? a study on edit intent classification in scientific document revisions. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 15049–15067, Miami, Florida, USA. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Zhen Tan, Dawei Li, Song Wang, Alimohammad Beigi, Bohan Jiang, Amrita Bhattacharjee, Mansooreh Karami, Jundong Li, Lu Cheng, and Huan Liu. 2024. Large language models for data annotation and synthesis: A survey. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 930–957, Miami, Florida, USA. Association for Computational Linguistics.

Qwen Team. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Jiaan Wang, Yunlong Liang, Fandong Meng, Zengkui Sun, Haoxiang Shi, Zhixu Li, Jinan Xu, Jianfeng Qu, and Jie Zhou. 2023. Is ChatGPT a good NLG evaluator? a preliminary study. In *Proceedings of the 4th New Frontiers in Summarization Workshop*, pages 1–11, Singapore. Association for Computational Linguistics.

Xinru Wang, Hannah Kim, Sajjadur Rahman, Kushan Mitra, and Zhengjie Miao. 2024. Human-llm collaborative annotation through effective verification of llm labels. In *Proceedings of the 2024 CHI Conference on Human Factors in Computing Systems*, CHI '24, New York, NY, USA. Association for Computing Machinery.

Zhiquan Ye, Yuxia Geng, Jiaoyan Chen, Jingmin Chen, Xiaoxiao Xu, SuHang Zheng, Feng Wang, Jun Zhang, and Huajun Chen. 2020. Zero-shot text classification via reinforced self-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3014–3024, Online. Association for Computational Linguistics.

Edward Yeo, Yuxuan Tong, Xinyao Niu, Graham Neubig, and Xiang Yue. 2025. Demystifying long chain-of-thought reasoning in llms. In *ICLR 2025 Workshop on Deep Generative Model in Machine Learning: Theory, Principle and Efficacy*.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Tiantian Fan, Gaohong Liu, Lingjun Liu, Xin Liu, and 1 others. 2025. Dapo: An open-source llm reinforcement learning system at scale. *CoRR*.

Bing Zhang, Mikio Takeuchi, Ryo Kawahara, Shubhi Asthana, Md. Maruf Hossain, Guang-Jie Ren, Kate Soule, Yifan Mai, and Yada Zhu. 2025. Evaluating large language models with enterprise benchmarks. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 3: Industry Track)*, pages 485–505, Albuquerque, New Mexico. Association for Computational Linguistics.

Tianyi Zhang, Faisal Ladhak, Esin Durmus, Percy Liang, Kathleen McKeown, and Tatsunori B. Hashimoto. 2024. Benchmarking large language models for news summarization. *Transactions of the Association for Computational Linguistics*, 12:39–57.

Yiqing Zhao, Xiaohui Shen, and Lanfeng Pan. 2025. Trustworthiness-as-reward: Improving llm performance on text classification through reinforcement learning.

Model	Control		Attributes		Integration		Implement.		# Avg
	Acc	F1	Acc	J	Acc	J	Acc	J	R tokens
Qwen3 8B	30.3	30.3	8.3	13.8	16.6	23.3	15.2	26.7	338.0
+ SFT only	71.7	71.7	32.4	41.2	36.6	44.0	40.0	59.3	574.8
+ RL only EM	69.7	69.7	22.1	33.8	46.9	54.9	37.2	60.2	992.2
+ RL only EM+F	69.7	69.7	22.1	33.9	36.6	49.3	29.0	54.3	975.7
+ RL only EM+F+JSON (Ours)	67.6	67.6	24.1	35.1	40.7	50.0	37.2	58.7	879.7
+ RL only J	66.9	66.9	24.1	35.6	35.9	49.3	40.7	63.1	979.3
+ RL only J+F	66.9	66.9	24.8	35.7	37.9	46.9	35.2	55.3	942.6
+ RL only J+F+JSON	66.2	66.2	21.4	33.8	28.3	45.6	30.3	53.6	955.8
+ SFT+RL EM	81.4	81.4	34.5	43.7	**63.4**	**66.9**	44.1	61.1	656.4
+ SFT+RL EM+F	**84.1**	**84.1**	**35.9**	**45.4**	57.9	61.0	**46.9**	*63.2*	615.4
+ SFT+RL EM+F+JSON (Ours)	**84.1**	**84.1**	34.5	42.4	*62.8*	*65.9*	46.2	62.0	563.4
+ SFT+RL J	82.8	82.8	33.1	43.4	60.7	64.5	44.8	62.7	580.1
+ SFT+RL J+F	81.4	81.4	35.2	*44.7*	62.1	*65.9*	45.5	63.1	675.4
+ SFT+RL J+F+JSON	78.6	78.6	**35.9**	44.6	51.0	55.2	**46.9**	**64.4**	651.8

Table 7: Ablation study on reward components for RL training. We compare training with RL only (from base model) vs. SFT+RL (from SFT checkpoint), using different reward combinations: Exact Match (EM) vs. Jaccard (J) for correctness, with optional Format (F) and JSON format rewards. # Avg R tokens=average reasoning trace length. Results averaged over 5 runs. **Bold** indicates best results; *italic*: second-best.

A CTG Papers Annotation

A.1 CTG categorisation scheme

We adopt a four-dimensional categorisation scheme to comprehensively characterise CTG systems. This scheme captures both the *what* (control type and attributes) and the *how* (control implementation and integration) of controllability in text generation systems.

Control Type. This binary field indicates whether a CTG system is designed to control a single attribute or multiple attributes simultaneously during text generation. *Single-attribute* systems control only one aspect at a time (e.g., sentiment only), while *Multiple-attribute* systems control multiple aspects simultaneously (e.g., both sentiment and topic).

Control Attributes. This multi-label field identifies the high-level semantic or structural properties that the system aims to influence. Attributes define the target aspects of control, such as *sentiment*, *topic*, *length*, *style*, *formality*, *toxicity*, or *factuality*. Unlike other fields, the attribute space is open-ended, as new controllable aspects continue to emerge depending on the addressed task.

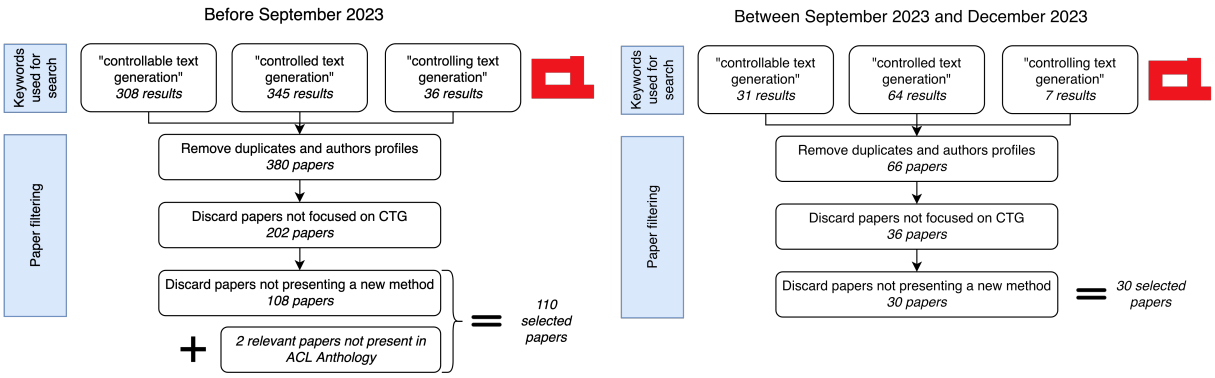
Control Implementation. This multi-label field specifies the method(s) by which control capabilities are enabled in the system. A system may employ multiple implementation strategies, including: *LM Prompting* (using input prompts or instructions to steer generation), *Text Selection* (selecting or filtering outputs that match the desired attribute), *Modification of Token Distribution* (adjusting output probabilities during decoding, e.g., PPLM, FUDGE), *Complete Training* (training a model from scratch with built-in control mechanisms), and *Fine-Tuning* (adapting a pre-trained model on control-specific data or objectives).

Control Integration. This multi-label field describes the architectural level(s) at which control signals are applied. Systems may integrate control through *Input Addition* (control signals provided at the input level, such as control codes or attribute tokens), *Model Injection* (control signals embedded within the model architecture, such as adapter modules or control-specific layers), or *System Addition* (control applied through external components, such as post-processing classifiers or plug-and-play modules). Many systems combine multiple integration strategies.

Table 1 summarises the annotation characteristics of each field.

A.2 Papers selection process

We conducted an extensive search on the ACL Anthology up to December 2023 using the keywords "controllable text generation," "controlled text gen-



(a) Selection process of the papers used as train and val sets.

(b) Selection process of the papers used as test set.

Figure 2: Systematic papers selection process for CTG papers on ACL Anthology.

	# samples	Paper text			Reasoning trace		
		Max token	Min token	Avg token	Max token	Min token	Avg token
Train	99	13673	3902	8905.96	900	200	564.8
Val	10	11118	6785	9151.4	684	337	548.2
Test	29	12141	2134	8116.6	-	-	-

Table 8: Dataset statistics for the CTG annotation task. Paper text and multi-field reasoning trace lengths are measured in tokens using the Qwen3 8B tokeniser. Test set reasoning traces are not used (model generates them during inference).

eration," and "controlling text generation," as illustrated in Figure 2. After eliminating duplicates, authors' profiles, and non-paper resources, we identified 446 papers. From this initial pool, we excluded papers not directly related to CTG, such as those merely mentioning CTG without exploring the task. Following this filtration process, we were left with 238 papers. Among these, we retained only those presenting novel models or control methods, excluding papers that solely introduced new datasets or conducted comparative studies. This selection yielded 140 papers. Additionally, we included two relevant papers (Dathathri et al., 2019; Keskar et al., 2019) not found in the ACL Anthology but cited in our pool of papers.

We consider all the papers published up to August 2023 as training set (Figure 2a), while the papers published between September and December 2023 as test set (Figure 2b).

A.3 Prompts and questions definition

This section presents the prompts used for reasoning trace generation, model training and inference, along with annotated examples illustrating the output quality of different approaches.

Question definition. Based on the categorisation scheme described in Appendix A.1, we define the questions that specify the annotation task for the model. Figure 7 shows the complete multi-field question that defines all four annotation dimensions (Control Type, Control Attributes, Control Implementation, and Control Integration) jointly, including the possible categories for each field and the expected JSON output format.

Reasoning trace generation. Figure 4 shows the prompt used to generate reasoning traces for training data. As described in Section 4.1, we use LLaMa3.3 70B Instruct to produce detailed step-by-step reasoning that leads to the gold annotation for each paper in our dataset. An example of a complete generated reasoning trace is shown in Figure 5.

Model training and inference. During both SFT training and inference, we use the prompt shown in Figure 6. The model is trained to generate outputs in a structured format containing both reasoning and annotation, enclosed within `<think>` and `<answer>` tags, respectively.

Output format and parsing. Figure 8 shows an example output from our best multi-field model (SFT+RL with EM+F+JSON rewards). The output contains both the reasoning trace and the final annotation expressed as a JSON object. To extract

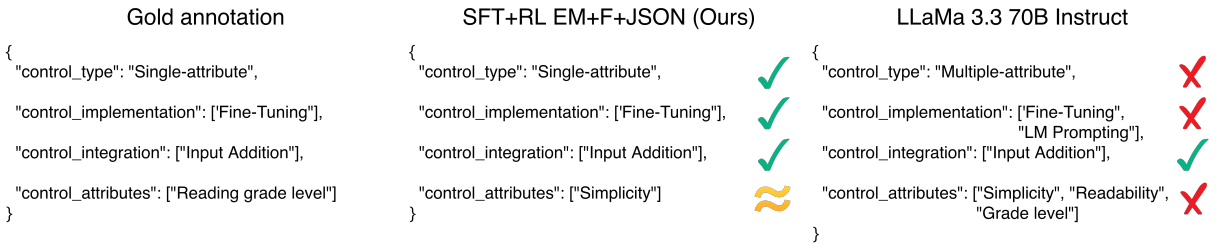


Figure 3: Annotation comparison for the same paper. Gold annotation (left), our best multi-field model (middle: Qwen3 8B SFT+RL with EM+F+JSON rewards), and LLaMa3.3 70B Instruct baseline (right).

the structured annotation, we parse the content between <answer> and </answer> tags as JSON. For evaluation, we extract all expected fields from the parsed JSON and compare them against the gold annotation using the metrics described in Section 5.

Figure 3 shows an example of generated annotations comparing the gold annotation, the annotation generated using our best model and the annotation generated using LLaMa3.3 70B Instruct.

Comparative example. Figure 3 compares annotations for the same paper across three approaches: gold annotation (human expert), our best model (Qwen3 8B SFT+RL), and LLaMA3.3 70B Instruct (zero-shot).

A.4 CTG papers dataset

Data collection and annotation. Following the procedure described in Section 4.1, we construct a dataset of 138 papers on controllable text generation from ACL Anthology published up to 2023. Each paper is annotated by one of the co-authors following the categorisation scheme detailed in Appendix A.1.

Reasoning trace generation. For each annotated paper, we generate synthetic reasoning traces that demonstrate the step-by-step annotation process. We create two sets of reasoning traces: (1) *single-field traces*, where each trace focuses on annotating one dimension independently, used to train separate single-field models for the ablation study in Section D, and (2) *multi-field traces*, where each trace annotates all four dimensions jointly in a single reasoning chain, used to train the unified multi-field model. Both trace types follow the same reasoning structure, systematically analysing paper content to justify label assignments, but differ in scope and inter-field reasoning dependencies.

Dataset statistics. Table 8 presents statistics for our CTG annotation dataset. The dataset is split into 99 training papers, 10 validation papers, and 29 test papers. Paper lengths range from approximately 2,100 to 13,700 tokens (mean: ∼8,900 tokens for training), measured using the Qwen3 8B tokeniser. Multi-field reasoning traces in the training and validation sets average ∼565 tokens, with lengths ranging from 200 to 900 tokens depending on the complexity of the CTG system being annotated. Test set papers do not include reasoning traces, as the model generates these during inference.

B Evaluation Papers Annotation

B.1 Evaluation categorisation scheme

Our evaluation annotation scheme categorises papers along two dimensions: **evaluation type**, which identifies the methodology used to assess system output quality, and **QCET node classification**, which characterises what quality aspects are being assessed.

Evaluation type. This dimension classifies evaluation measures according to the methodology used to produce assessments. We distinguish four types:

- **None found**: No evaluation of system output quality is reported in the paper;
- **Human**: Evaluations where scores, annotations, or other assessments are assigned by human participants, or where human participants interact with a system (other aspects may be automated);
- **Metric**: Fully automatic evaluations, such as BLEU, perplexity, or other metrics calculated automatically;
- **Hybrid**: Evaluations where scores, annotations, or assessments are determined by a combination of human and metric techniques (other aspects may be either manual or automated).

Evaluation type can be annotated at two levels of granularity: (i) *paper-level*, where multi-label classification identifies all evaluation approaches used anywhere in the paper (e.g., a paper might use both Human and Metric evaluations); (ii) *measure-level*, where each distinct evaluation measure or criterion is identified and individually classified (e.g., "BLEU score" → Metric, "human fluency ratings" → Human).

We exclude from annotation any evaluations of computational efficiency (e.g., inference time, memory usage) or evaluations of human-created texts rather than system outputs.

QCET node classification. At the measure level, we additionally annotate each evaluation measure using the Quality Criteria for Evaluation Taxonomy (QCET) (Belz et al., 2025). QCET provides a hierarchical framework with three levels that progressively narrow the characterisation of what quality aspect an evaluation measure assesses:

QCET Level 1 (Frame of Reference) captures the frame of reference relative to which system output quality is assessed. The four classes are:

- **Output in its own right**: Quality criteria that assess outputs based solely on the output itself, without reference to inputs or gold standards (e.g., fluency ratings, readability scores);
- **Output relative to input**: Quality criteria that assess outputs relative to both, and only, the output and the input (e.g., faithfulness to source, relevance to query);
- **Output relative to target outputs**: Quality criteria that assess outputs relative to in-distribution gold-standard reference outputs, optionally also considering the input (e.g., BLEU, ROUGE, exact match against gold answers);
- **Output relative to external frame**: Quality criteria that assess outputs relative to a system-external frame of reference beyond input and target outputs (e.g., factual accuracy verified against knowledge bases, alignment with external guidelines).

QCET Level 2 (Type of Quality) captures the type of quality being assessed. The three classes are:

- **Correctness**: Quality criteria based on countable errors or deviations from a standard (e.g., word error rate, grammatical error count, exact match);
- **Goodness**: Quality criteria that assess overall quality holistically, typically considering multiple factors without distinguishing them (e.g., overall quality ratings, coherence scores, helpfulness);
- **Features**: Quality criteria that measure properties without an inherent good/bad orientation; either end of the scale can be preferred depending on context (e.g., length, diversity, specificity).

QCET Level 3 (Aspect of Outputs) captures what aspect of system outputs is being assessed. The three classes are:

- **Form**: Quality criteria that assess only the surface form of outputs (e.g., spelling, grammar, formatting);
- **Content**: Quality criteria that assess only the meaning or content of outputs (e.g., semantic similarity, factual correctness, relevance);
- **Outputs as a whole**: Quality criteria that assess outputs holistically without distinguishing form from content (e.g., overall quality, adequacy, naturalness).

Each evaluation measure is classified along all three QCET levels, producing a three-dimensional characterisation.

B.2 Paper selection process

The QCET dataset consists of three samples of NLP papers. For the first sample, 60 ACL 2022–2024 main conference papers were selected by downloading the ACL Anthology as BibTex from `https://aclanthology.org` in Feb 2025, extracting all ACL main conference items for 2022, 2023, and 2024, then removing any entries which were not papers, e.g. frontmatter, before randomly sampling 20 papers from each year.

The second sample is another 60 papers, also randomly, but independently, sampled using the sample method, and ensuring no overlap.

The third sample consists of the first 20 papers from a reannotation of the 190 papers from the 20Years Survey (Howcroft et al., 2020). Full details of the sampling methods and annotation scheme can be found in Belz et al. (2025).

Due to extraction errors in our preprocessing phase, 1 paper was discarded. This resulted in 139 papers in our dataset.

Altogether, the three samples had 569 individual evaluation measures in them, corresponding to an average of 4.09 per paper. When we refer to a measure-level classification task in the main part of the paper, this is what we refer to: the task is to identify all evaluation measures for which results are reported in a paper and then categorise them according to evaluation type and QCET Levels 1–3.

B.3 Prompts and questions definition

We adapt the prompts from CTG annotation (Appendix A.3) by modifying task and question definitions to reflect the evaluation categorisation scheme (Appendix B.2).

Question definition. Figure 9 shows the measure-level evaluation type question, which asks the model to identify and classify each evaluation measure. Figure 10 shows the measure-level QCET node question, requiring classification across three hierarchy levels.

Reasoning trace generation. We use LLaMA3.3 70B Instruct with the same prompt structure as Figure 4, substituting evaluation-specific questions and task description.

Model training and inference. We use the same prompt structure as Figure 6 with evaluation-specific questions. Outputs contain `<think>` blocks (reasoning) and `<answer>` blocks (JSON annotations). For measure-level tasks, JSON outputs are arrays of measures with their classifications.

B.4 Evaluation papers dataset

Data collection and annotation. Following the same procedure used for our CTG dataset (Section 4.1) and the papers selection process described in Section B.2, we construct a dataset of 139 papers. Each paper is annotated by three of the co-authors following the categorisation scheme detailed in Appendix B.1.

Reasoning trace generation. For each annotated paper, we generate synthetic reasoning traces that demonstrate the step-by-step annotation process. We create one set of reasoning traces for each setting: (1) paper-level evaluation type classification, (2) measure-level evaluation type classification, (3) measure-level QCET classification, and (4) measure-level evaluation type and QCET classification. All trace types follow the same reasoning structure, systematically analysing paper content to justify label assignments, but differ in scope.

Dataset statistics. Table 9 shows statistics for our evaluation annotation dataset. The dataset is split into 99 training papers, 10 validation papers, and 30 test papers. Paper lengths range from approximately 400 to 15,900 tokens (mean: ∼8,500 tokens for training), measured using Qwen3 8B tokeniser. Reasoning traces in the training and validation sets average ∼405 and ∼449 tokens respectively, with lengths ranging from 171 to 838 tokens depending on the number and complexity of evaluation measures in each paper. Test set papers do not include reasoning traces, as the model generates these during inference.

C Hyperparameters

We use LLaMa3.3 70B Instruct (Grattafiori et al., 2024) to generate the reasoning traces of our datasets using three NVIDIA A100 GPUs (80 GB each). We use 2048 maximum generated tokens, temperature 0.6, and top p 0.95.

All training runs are performed on two NVIDIA A100 GPUs (80 GB each). Only for RL training for all the evaluation categorisation settings we use three NVIDIA A100 GPUs (80 GB each). We use Qwen3-8B[3] (Team, 2025) as the base model, fine-tuned with LoRA, which is configured with a rank of 32, scaling factor $\alpha = 64$, dropout 0.05, and applied to all attention projection matrices.

For SFT, we train for up to 30 steps with batch size 1, gradient accumulation 8, learning rate 2×10^{-4}, `paged_adamw_32bit` optimiser, a constant learning-rate scheduler, and weight decay 0.01.

For RL, we train for up to 3 epochs with batch size 1, gradient accumulation 8, number of generations per sample 8, learning rate 1×10^{-5}, `paged_adamw_32bit` optimiser, a constant learning-rate scheduler, and DAPO loss type. When using the combination of 3 reward functions, we use weights 0.7, 0.25, 0.05 for correctness, JSON format, and format reward, respectively. When using the combination of 2 reward functions, we use weights 0.7 and 0.3 for correctness and format reward, respectively.

For inference of all our trained models, we use one NVIDIA RTXA6000 with 48 GB. We use 2048

[3] https://huggingface.co/Qwen/Qwen3-8B

	# samples	Paper text			Reasoning trace		
		Max token	Min token	Avg token	Max token	Min token	Avg token
Train	99	15911	4121	8528.8	838	171	404.7
Val	10	11394	3278	8267.4	770	260	448.6
Test	30	13924	418	9117	-	-	-

Table 9: Dataset statistics for the evaluation annotation task. Paper text and reasoning trace lengths are measured in tokens using the Qwen3 8B tokeniser. Test set reasoning traces are not used (model generates them during inference).

maximum generated tokens, temperature 0.6, top p 0.95, top k 20, and min p 0, as recommended by Qwen3 8B.[4]

C.1 Baselines

Regarding the baselines, we tested two families of models: LLaMa3 (8B and 70B) without reasoning training (Grattafiori et al., 2024) and Qwen 3 (8B, 14B, and 32B) with reasoning training (Team, 2025). More specifically, we tested LLaMa3.1 8B Instruct[5], LLaMa3.3 70B Instruct,[6] Qwen3 8B,[7] Qwen3 14B,[8] and Qwen3 32B.[9] For all models, we use 1024 maximum generated tokens, temperature 0.6, top p 0.95, top k 20, and min p 0.

D Additional Results Comparing Single-field vs. Multi-field Models

To validate our architectural choice of training a multi-field model, we compare it against an alternative approach of training four separate single-field models (one per annotation dimension). Table 10 presents the results along with inference time per paper. The multi-field model consistently outperforms single-field models after SFT with only a minor disadvantage on Control Implementation Jaccard. After RL training, the multi-field model achieves strong improvements across all dimensions. Regarding inference efficiency, the multi-field model provides substantial computational advantages. Single-field models require four separate forward passes to annotate a complete paper (one per dimension). In contrast, the multi-field model produces all four annotations in a single forward pass, likely due to the more concise reasoning traces learned during RL. The multi-field approach benefits from: (1) shared reasoning context that analyses the paper once for all dimensions, (2) learned inter-field dependencies (e.g., multiple control implementations often correlate with multiple integration levels), and (3) denser RL reward signals from partial credit across fields. These advantages outweigh potential risks like error propagation across dimensions.

E Scientific Artefacts and Licensing

LLaMa 3.1 8B Instruct[10] and LLaMa 3.3 70B Instruct[11] are licensed under a commercial license. Qwen3 8B, Qwen3 14B, and Qwen3 32B are licensed under the Apache-2.0 license. ACL Antholoy[12] material is licensed under the Creative Commons Attribution 4.0 International License.[13] The usage of all listed artifacts is consistent with their licenses.

[4] https://huggingface.co/Qwen/Qwen3-8B

[5] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct

[6] https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct

[7] https://huggingface.co/Qwen/Qwen3-8B

[8] https://huggingface.co/Qwen/Qwen3-14B

[9] https://huggingface.co/Qwen/Qwen3-32B

[10] https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/LICENSE

[11] https://github.com/meta-llama/llama-models/blob/main/models/llama3_3/LICENSE

[12] https://aclanthology.org/

[13] https://creativecommons.org/licenses/by/4.0/

Model	Control		Attributes		Integration		Implementation		Time (s) ↓
	Acc	F1	Acc	J	Acc	J	Acc	J	
SFT Single-field	69.7	69.7	29.7	41.8	33.8	45.5	38.6	**62.4**	120.4
SFT Multi-field	71.7	71.7	32.4	41.2	36.6	44.0	40.0	59.3	39.9
SFT+RL Multi-field	**84.1**	**84.1**	**34.5**	**42.4**	**62.8**	**65.9**	**46.2**	62.0	**36.8**

Table 10: Comparison of single-field vs. multi-field models on CTG annotation. Single-field trains four drivemodels (one per dimension), requiring four sequential inferences. Multi-field predicts all dimensions jointly in a single forward pass. Time=average inference time per paper (lower is better). Results averaged over 5 runs. **Bold** = best result.

Think Traces Generation prompt

```
<paper>
{paper_text}
</paper>

<question>
{question}
</question>

Your task is to produce a reasoning to explain how to extract the {question_type} from the paper text.

The expected answer is:
<answer>
{answer}
</answer>

Instructions for reasoning:
1. Initial Analysis: First, identify what specific aspect of the CTG system the question is asking about and what the possible answer categories are.
2. Systematic Paper Review:
    - Start with the abstract and introduction to understand the proposed system
    - Examine the methodology/approach section for technical details
    - Review results/experiments to see how the system works in practice
    - Check any architectural diagrams or system descriptions
3. Evidence Collection:
    - Quote specific sentences or phrases that directly support your answer
    - Note section names/numbers where you found key information
    - If information is implicit, explain your inference process
4. Category Matching:
    - For each possible category, explain why it does or doesn't apply
    - Pay special attention to edge cases and hybrid approaches
    - If the answer involves multiple components (like "Hybrid"), identify all relevant parts
5. Verification:
    - Double-check that your chosen category aligns with all the evidence
    - Consider if there are any contradictory statements in the paper
    - Assess your confidence level and note any ambiguities

Special considerations:
- For Hybrid categories: Clearly identify each component and provide evidence for all parts
- For attribute identification: Look beyond obvious mentions, consider implicit attributes controlled through the methodology
- For technical distinctions: Be precise about implementation details (e.g., distinguishing between fine-tuning vs. prompting)

Encapsulate your complete reasoning process between <think> and </think> tags, following the structure above.
<think>
[Your detailed reasoning here]
</think>
<answer>
[Your final answer here aligned to the answer format and reasoning provided]
</answer>
```

Figure 4: Prompt used to generate reasoning traces with LLaMa 3.3 70B Instruct. Placeholders are shown in {braces}.

Example of generated reasoning trace

```
To extract the control type, control attributes, control implementation, and control integration from the given paper, let's follow the steps outlined in the instructions.

1. **Initial Analysis**: The question asks for the control type, control attributes, control implementation, and control integration used in the proposed system. The possible answer categories for control type are Single-attribute and Multiple-attribute. Control attributes could be various, such as sentiment, topic, length, etc. Control implementation could be LM Prompting, Text Selection, Modification of Token Distribution, Complete Training, Fine-Tuning, or Hybrid. Control integration could be Input Addition, Model Injection, System Addition, or Hybrid.

2. **Systematic Paper Review**:
    - The abstract and introduction indicate that the paper proposes a joint framework with a topic refinement mechanism for topic-aware dialogue response generation. This suggests that the system is focused on controlling the topic attribute.
    - The methodology section describes a three-pass mechanism involving response generation, topic prediction, and topic refinement, which implies that the system is designed to control the topic attribute explicitly.
    - The results and experiments show that the system outperforms previous state-of-the-art models in terms of topic-related metrics, further supporting the focus on topic control.

3. **Evidence Collection**:
    - The sentence "We propose a joint framework with a topic refinement mechanism to solve the topic-aware multi-turn end-to-end dialogue generation problem" supports the idea that the system controls the topic attribute.
    - The description of the three-pass mechanism (response generation, topic prediction, and topic refinement) in the methodology section provides evidence for how the topic attribute is controlled.
    - The use of specific models like GPT2 and BERT for topic prediction and response generation implies fine-tuning as the control implementation method.

4. **Category Matching**:
    - **Control Type**: Since the system is primarily focused on controlling the topic attribute, the control type is Single-attribute.
    - **Control Attributes**: The primary attribute controlled by the system is the topic.
    - **Control Implementation**: The system fine-tunes pre-trained models (GPT2 and BERT) for topic prediction and response generation, indicating Fine-Tuning as the control implementation.
    - **Control Integration**: The control signal (topic information) is integrated into the system through the input, as the topic prediction and refinement stages are directly influenced by the input dialogue context and previous topics, suggesting Input Addition.

5. **Verification**:
    - The evidence supports that the system is designed to control the topic attribute, making Single-attribute the correct control type.
    - Topic is the primary attribute being controlled.
    - Fine-Tuning is the correct implementation, as the system adapts pre-trained models for its tasks.
    - Input Addition is the correct integration method, as the topic information is used to influence the generation process through the input stages of the system.
```

Figure 5: Example of a generated reasoning trace in the training set using LLaMa3.3 70B Instruct.

Training and Inference prompt

```
<paper> {paper_text} </paper>

<question> {question} </question>

Analyze the paper to classify the controllable text generation system according to the question.

Think step by step: identify the relevant information in the paper, cite specific sections or quotes that support your answer, and explain how they lead to your classification choice.

<think> [Your reasoning process] </think>
<answer> [Your final classification] </answer>
```

Figure 6: Prompt used for training and inference of CTG papers annotation. Placeholders are shown in {braces}.

Multi-field question for CTG categorisation

```
Based on the given paper, you need to identify the control type, the control implementation, the control integration, and the control attributes used in the proposed system.

Control type is whether the implemented system is able to control only one attribute at a time, or multiple attributes at once and it must be one of the following categories:
- Single-attribute, i.e. the implemented system controls only one attribute type at a time;
- Multiple-attribute, i.e. the implemented system controls multiple attribute types at once, for example sentiment and topic.

Control attributes are the high-level semantic or structural properties of the generated text that a controllable text generation system aims to influence or constrain or control. They define the target aspect of control, such as sentiment, topic, length, style, knowledge content, or discourse function.
You need to return a list of all the control attributes used, for example: {"control_attributes": ["Sentiment", "Keywords", "Length", "Dialogue act", "Knowledge elements"]}.

Control implementation is the method by which control capabilities are enabled in a CTG system and it must be one or more of the following categories:
- LM Prompting, i.e. use input prompts to steer the generation of the model;
- Text Selection, i.e. select the text that most represents the given control attribute;
- Modification of Token Distribution, i.e. modify the token distribution of the pre-trained model to control the generated text;
- Complete Training, i.e. train the model from scratch;
- Fine-Tuning, i.e. fine-tune the model on a specific dataset.
In case of multiple implementations, you need to specify the combination of implementations as list, for example: {"control_implementation": ["LM Prompting", "Modification of Token Distribution"]}.

Control integration is the level at which the control signal is applied within the system architecture: at the input (Input Addition), within the model (Model Injection), or as an external module (System Addition). Different combinations of integrations are possible. It must be one or more of the following categories:
- Input Addition, i.e. the control signal is added at the input of the system, for example adding a control code to the input;
- Model Injection, i.e. the control signal is injected within the model, for example injecting a PEFT module encapsulating the control signal inside the model;
- System Addition, i.e. the control signal is added as an external module, for example adding a classifier that modifies the output of the model;
In case of multiple integrations, you need to specify the combination of integrations as list, for example: {"control_integration": ["Input Addition", "Model Injection"]}.

You need to return the control type, the control attributes, the control implementation, and the control integration categories as JSON object between <answer> and </answer> tags, for example: <answer>{"control_type": "Single-attribute", "control_attributes": ["Sentiment", "Keywords", "Length", "Dialogue act", "Knowledge elements"], "control_implementation": "LM Prompting", "control_integration": "Input Addition"}</answer>.
```

Figure 7: Definition of the multi-field question for CTG categorisation. The question defines all 4 fields for the CTG categorisation: *Control Type*, *Control Attributes*, *Control Implementation*, and *Control Integration*.

Output example using our multi-field SFT+RL best model

```
<think>
To answer this question, we first need to understand the key components of a controllable text generation (CTG) system and how they relate to the control type, control attributes, control implementation, and control integration.

1. **Initial Analysis**: The question asks us to identify the control type, control attributes, control implementation, and control integration used in the proposed system based on the given paper. The control type can be either single-attribute or multiple-attribute. Control attributes are the aspects of the text that the system aims to control, such as sentiment, topic, or length. Control implementation refers to how the control capabilities are enabled, which could be through LM prompting, text selection, modification of token distribution, complete training, fine-tuning, or a hybrid approach. Control integration pertains to where the control signal is applied, which could be through input addition, model injection, system addition, or a hybrid method.

2. **Systematic Paper Review**:
    - The abstract and introduction indicate that the system focuses on controlling the simplicity of text, specifically for grade-specific text simplification. This suggests that the primary control attribute is "Simplicity".
    - The methodology section explains that the system predicts control tokens based on source text features and the desired target grade level, indicating a fine-tuning approach for implementation.
    - The system description and results show that the control signal is integrated at the input level, as the control tokens are added to the input sequence to guide the generation of the output.

3. **Evidence Collection**:
    - "We introduce a Control Predictor module (CP) that predicts a vector of control token values V for each input X at inference time." This supports the idea of fine-tuning and input addition for control integration.
    - "The control tokens are added to the source text as a side constraint in the format, W_{} C_{} L_{} WR_{} DTD_{} {Source_ text}." This confirms that control attributes are integrated at the input level.
    - The system aims to control the simplicity of the text, as indicated by the focus on grade-specific simplification and the use of metrics like SARI.

4. **Category Matching**:
    - **Control Type**: Given that the primary focus is on controlling simplicity, it appears to be a single-attribute control, as the system does not explicitly mention controlling multiple attributes simultaneously.
    - **Control Attributes**: The primary attribute controlled is "Simplicity".
    - **Control Implementation**: The system is implemented through fine-tuning, as it involves training a control predictor to predict control tokens and using these to guide the generation.
    - **Control Integration**: The control signal is integrated through input addition, as control tokens are added to the input sequence to influence the output.

5. **Verification**:
    - The evidence supports a fine-tuning implementation and input addition for control integration.
    - The control attribute is clearly simplicity, and the system is designed to control this attribute effectively.

</think> <answer> {"control_type": "Single-attribute", "control_implementation": "Fine-Tuning", "control_integration": "Input Addition", "control_attributes": ["Simplicity"]} </answer><|im_end|>
```

Figure 8: Example of generated output using our best model (multi-field SFT+RL with EM+F+JSON rewards). The output contains the reasoning trace and the final answer expressed as JSON object.

Measure-level evaluation type question for evaluation categorisation

```
Based on the given paper, you need to identify each evaluation measure used in the research reported in the paper in order to assess the output quality of the proposed systems. For each identified evaluation measure, you further need to identify its type. Include only evaluations of system output quality; do not include evaluations of efficiency, or evaluations of human-created texts.

More specifically, your task is to list each distinct evaluation measure used in the paper, and for each one, classify it into one of the following types:
- None found, i.e. no evaluation of system output quality is found in the paper;
- Human, i.e. evaluations where scores, annotations or other assessments are assigned by human participants, our where human participants interact with a system (other aspects may be automated);
- Metric, i.e. fully automatic evaluations, such as BLEU or perplexity calculated with a given model;
- Hybrid, i.e. evaluations where scores, annotations or other assessments are determined by a combination of human and metric techniques (other aspects may be either manual or automated).

For each evaluation measure identified, create one item in the output list. You need to return the results as a JSON array of objects where each object (key-value pair) represents one evaluation measure, for example [{"evaluation_type": "Metric"}, {"evaluation_type": "Human"}, {"evaluation_type": "Metric"}], or [{"evaluation_type": "None found"}] if no evaluation measures are found. The array must contain one key-value pair for each evaluation measure identified in the paper, and each value must be exactly one label (one of "Metric", "Human", "Hybrid", "None found").
```

Figure 9: Definition of the measure-level evaluation type question for evaluation categorisation.

Measure-level QCET node question for evaluation categorisation

```
Based on the given paper, you need to identify each evaluation measure used in the research reported in the paper in order to assess the output quality of the proposed systems. For each identified evaluation measure, you further need to classify the evaluation measure in terms of its frame of reference, the type of quality it assesses, and the aspect of system outputs it assesses. Include only evaluations of system output quality; do not include evaluations of efficiency, or evaluations of human-created texts.

More specifically, your task is to list each distinct evaluation measure used in the paper, and for each one, to classify the quality criterion the evaluation assesses in terms of the following three dimensions:

1. QCET Level 1, consisting of four Frame-of-Reference classes each capturing one of four possible frames of reference relative to which system output quality can be assessed. The four classes are the following:

- Output in its own right (output_in_its_own_right): choose this class for quality criteria that assess system output quality relative to just the output; i.e. the measured values these quality criteria produce are obtained on the basis of the system outputs alone;
- Output relative to input (output_relative_to_input): choose this class for quality criteria that assess system output quality relative to both, and only, the output and the input; i.e. the measured values these quality criteria produce are obtained on the basis of system outputs and system inputs alone;
- Output relative to in-distribution target outputs +/-input (output_relative_to_target_outputs): choose this class for quality criteria that assess system output quality relative to target outputs sampled from the same distribution as the training data, and optionally also relative to the input; i.e. the measured values these quality criteria produce are obtained on the basis of system outputs and gold-standard reference outputs (plus, optionally, system inputs);
- Output relative to a system-external frame of reference +/-input (output_relative_to_external_frame): choose this class for quality criteria that assess system output quality relative to a system-external frame of reference, and optionally also the input; i.e. the measured values produced by an evaluation with this type of quality criterion are obtained on the basis of system outputs and gold-standard reference outputs (plus, optionally, system inputs).

2. QCET Level 2, consisting of three Type-of-Quality classes each capturing one of three possible types of system output quality assessed by a quality criterion. The three classes are the following:

- Correctness (correctness): choose this class for quality criteria that assess the correctness of system outputs; i.e. the measured values these quality criteria produce are based on countable errors or deviations;
- Goodness (goodness): choose this class for quality criteria that assess the goodness of system outputs; i.e. the measured values these quality criteria produce are not primarily based on notions of countable errors, instead typically taking multiple factors into account without naming or distinguishing them;
- Features (features): choose this class for quality criteria that assess features of system outputs; i.e. quality criteria of this type have no natural good end of the scale; while Correctness and Goodness criteria align with (i) what systems are trained to be good at, and/or (ii) common-sense notions of desirable properties in systems; in Feature-type criteria, instead, either end of the scale can be the preferred end depending on evaluation context.

3. QCET Level 3, consisting of three Aspect-of-Outputs classes each capturing one of three possible aspects of system outputs assessed by a quality criterion. The three classes are the following:
- Form (form): choose this class for quality criteria that assess the form of system outputs alone;
- Content (content): choose this class for quality criteria that assess the content/meaning of system outputs alone;
- Outputs as a whole (outputs_as_a_whole): choose this class for quality criteria that assess system outputs without distinguishing form from content.

In choosing the above classes, bear in mind that the name a paper gives an evaluation measure may not be fully aligned with what it actually assesses. If the paper provides a definition, gloss, or the question put to evaluators for an evaluation measure, then use that as the basis for choosing classes, rather than the name.

For each evaluation measure identified, create one item in the output list. You need to return the results as a JSON array of objects where each object (key-value pair) represents one evaluation measure, for example [{"qcet_level_1": "output_relative_to_input", "qcet_level_2": "goodness", "qcet_level_3": "content"}, {"qcet_level_1": "output_relative_to_target_outputs", "qcet_level_2": "correctness", "qcet_level_3": "outputs_as_a_whole"}].
```

Figure 10: Definition of the measure-level QCET node question for evaluation categorisation. The question defines all 3 fields for the QCET node categorisation: QCET node level 1, QCET node level 2, and QCET node level 3.