

Unveiling the Art of Heading Design: A Harmonious Blend of Summarization, Neology, and Algorithm

Shaobo Cui^{1*}, Yiyang Feng^{1*}, Yisong Mao¹, Yifan Hou², Boi Faltings¹
¹ EPFL, Switzerland, ²ETH Zürich, Switzerland
¹{firstname.lastname}@epfl.ch, ²{firstname.lastname}@inf.ethz.ch

Abstract

Crafting an appealing heading is crucial for attracting readers and marketing work or products. A popular way is to summarize the main idea with a refined description and a memorable acronym. However, there lacks a systematic study and a formal benchmark including datasets and metrics. Motivated by this absence, we introduce **Logogram** (LOGOGRAM), a novel benchmark comprising 6,653 paper abstracts with corresponding descriptions and acronyms. To measure the quality of heading generation, we propose a set of evaluation metrics from three aspects: summarization, neology, and algorithm. Additionally, we explore three strategies for heading generation (generation ordering, tokenization of acronyms, and framework design) under various prevalent learning paradigms (supervised fine-tuning, in-context learning with Large Language Models (LLMs), and reinforcement learning) on our benchmark. Our experimental results indicate the difficulty in identifying a practice that excels across all summarization, neologistic, and algorithmic aspects.

1 Introduction

Heading design has various applications in news recommendation (Cai et al., 2023), academia (Zhang and Tetreault, 2019), advertising (Yang et al., 2023), and search engine optimization (SEO) (Tsuruoka et al., 2005). Regarding academia, a well-crafted paper heading should accurately encapsulate the paper’s main idea while also being memorable and appealing. A typical approach is to concatenate the following two components as the heading (Tsuruoka et al., 2005): (i) *Description* — the main body of the heading that summarizes the paper’s main idea; (ii) *Acronym* — an appealing and memorable word or word-like string whose letters are usually

*Equal contribution

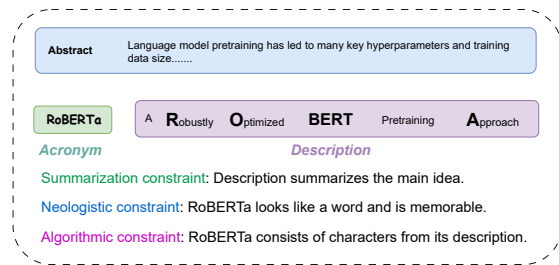


Figure 1: An example of a paper heading consisting of a description and an acronym.

constructed from the description (Appelman, 2020).¹ For the example shown in Figure 1, the description part in the heading, namely “A Robustly Optimized BERT pretraining Approach”, precisely summarizes the main idea from the work (Liu et al., 2019) that it provides a more robust BERT (Devlin et al., 2019) pretraining approach. The acronym “RoBERTa” forms itself from the underlined letters shown in the description. Additionally, it is memorable because it resembles a word and is the name of a character from the television series *Sesame Street*.

Despite growing interest in heading generation, there is still a lack of systematic studies and benchmarks on this topic. Addressing this, we propose a fun generation task predicting the description and acronym of the heading given the *abstract* of a paper as its main idea. This task is challenging as it requires a harmonious blend of multiple constraints: (i) **Summarization** constraint — the predicted description should be a precise summary of the abstract; (ii) **Neologistic** constraint — the predicted acronym should look like a word and be memorable; (iii) **Algorithmic** constraint — The letters in the acronym should be taken sequentially from the words in the description. It is clear from Figure 1 that this constraint is satisfied with the

¹We have updated the definition of “acronym” not to require initial letters, noting that researchers, like in the RoBERTa example, often deviate from tradition.






Aspect	 (our work)	Acronym Generation	Controllable Text Generation	Dual Text Generation
Objectives	Generating engaging headings consisting of acronyms and descriptions.	Generating acronyms for input text.	Generating text that meets specific constraints.	Generating two texts with dual dependency.
Problem definitions	$\{D \rightleftharpoons A\} = f(X, \mathcal{M})$	$A = f(X)$ where X is the input, A is the expected acronym.	$Y = f(X, \mathcal{M})$ where X is the input while \mathcal{M} is the set of controlled elements.	$\begin{cases} Y = f(X) \\ X = f^{-1}(Y) \end{cases}$ where X and Y are two dual outputs.
Challenges	(i) Balancing multiple constraints: summarization, neologistic, and algorithmic; (ii) There are limited techniques for neologistic, and algorithmic requirements.	(i) Involving creative word form, combining words or parts of words in novel ways; (ii) Grammatically correct and pronounceable; (iii) Accurately reflecting the meaning of the original phrase.	(i) Adhering to controlled elements while maintaining text quality and diversity; (ii) Understanding the semantics of the controlled elements and translating them into appropriate linguistic features.	(i) Ensuring dual dependency; (ii) Capturing the intricate relationships between two different types of text; (iii) Maintaining consistency and coherence across dual outputs.
Evaluation metrics	Summarization, neologistic, and algorithmic aspects.	Creativity, relevance, and impact metrics.	Evaluated based on control adherence and text quality.	Evaluated based on accuracy and coherence between dual texts.
Applications	Academic (Zhang and Tetreault, 2019), news recommendation (Cai et al., 2023), SEO (Tsuruoka et al., 2005), etc (App. A).	Search engine optimization (SEO) (Tsuruoka et al., 2005)	Attribute-based generation (Saha et al., 2022), debiasing (Dinan et al., 2020), data to text (Ribeiro et al., 2021), etc.	Machine translation (forward and backward) (He et al., 2016), Question and answer generation (Tang et al., 2017), etc.
Unique contributions of 	 highlights the interplay of summarization requirements, neologistic creativity, and algorithmic precision. Its standout contributions lie in its challenges of novel neologistic and algorithmic demands. The three requirements make it particularly suited for applications where branding, marketing impact, and search engine optimization.			

Table 1: Comparison of related text generation tasks.

acronym ‘‘RoBERTa’’ and its description. Our proposed constraints align well with existing standards of academic heading design (Tsuruoka et al., 2005; Appelman, 2020).


The heading generation task is different from the text generation tasks in the literature. Unlike dual text generation tasks (Xia et al., 2016; Tang et al., 2017) that focus on generating two outputs with a dual relationship, our heading generation task requires a nuanced form of dual dependency between the acronym and the description at the algorithmic level. Besides, this task differs from controllable text generation (Dinan et al., 2020; Saha et al., 2022) as it requires two mutually dependent outputs, i.e., the description and the acronym. Additionally, the neologistic and algorithmic constraints have never been considered in previous controllable generation tasks.

To support our heading generation task, we first establish a pioneering benchmark called . The  dataset consists of the abstracts and headings of 6,653 papers published in academic venues featured in the ACL Anthology. We exclusively collect papers whose headings include both descriptions and acronyms.

Then, we measure the quality of the generated descriptions and acronyms by evaluating their compliance with the summarization, neologistic, and algorithmic constraints: (i) We first evaluate descriptions using the existing summarization met-

rics like ROUGE (Lin, 2004), BERTScore (Zhang et al., 2020a), etc; (ii) We measure the extent of the resemblance of a generated acronym to real words by using our newly proposed neologistic metrics, i.e., *WordLikeness* and *WordOverlap*; (iii) Our newly introduced algorithmic metric, *LCSRratio*, measures the consistency between the sequence of letters in an acronym and the sequence of letters in its corresponding description.

To explore the best practice for heading generation, we propose and empirically study various strategies (generation ordering, tokenization of acronym, and framework design) across summarization, neologistic, and algorithmic dimensions. We test these strategies within learning paradigms including supervised fine-tuning, reinforcement learning, and in-context learning with LLMs. Experimental results from various strategies and learning paradigms indicate the difficulty in identifying a practice that optimally balances the summarization, neologistic, and algorithmic requirements. Our contributions are three-fold:

- To support the task of generating a heading consisting of an acronym and a description, we contribute a valuable benchmark . It poses unique summarization, neologistic, and algorithmic challenges. The benchmark can be accessed via <https://github.com/cui-shaobo/logogram>.

- Apart from the existing summarization metrics, we propose new metrics regarding both neologistic and algorithmic aspects. These various types of metrics enable a comprehensive evaluation of heading generation.
- To explore the best practice for heading generation, we propose and empirically study strategies including generation ordering, tokenization of acronyms, and framework design, under learning paradigms covering supervised fine-tuning, reinforcement learning, and in-context learning with LLMs.

Organization of This Paper. We begin by defining the task and outlining the core research questions of `Ugogram` in § 2. This is followed by a detailed description of dataset construction and its statistics in § 3. In § 4, we introduce the automatic evaluation metrics tailored for this novel task, which integrates aspects of summarization, neology, and algorithm constraints, along with justification evidence for these metrics. We then present an empirical analysis of various strategies across different learning paradigms in § 5. The paper concludes with a review of related work in § 6 and final remarks in § 7.

2 Task of `Ugogram`

In this section, we first formally define our task in § 2.1 and compare it with existing relevant tasks in § 2.2. Then, we propose three focused research questions that motivate our work in § 2.3.

2.1 Task Definition

As depicted in Figure 1, our goal is to generate an engaging paper heading in the fashion of a description prepended with an acronym, given the paper’s main idea, i.e., the abstract. This task has three inherent constraints:

- **Summarization Constraint:** The description should summarize well the paper’s main idea.
- **Neologistic Constraint:** The acronym forms a memorable word/pseudo-word.
- **Algorithmic Constraint:** The acronym catches letters from the description in the correct order.

Mathematically, our task is defined as

$$\{D \rightleftharpoons A\} = f(X, \mathcal{M}) \quad (1)$$

where D and A means the description and the acronym in the heading. \mathcal{M} is the set of constraints covering summarization, neologistic, and algorithmic requirements. The \rightleftharpoons symbol in $\{D \rightleftharpoons A\}$ means that D and A are mutually dependent: A forms itself from characters in D and A in return guides the word choice during the generation of D . This task demands a sophisticated blend of human skills, including summarization capability, algorithmic planning, and creativity in neology. Besides, the interconnected nature between the acronym and the description increases this task’s intricacy.

2.2 Comparison with Other Generation Tasks

As delineated in Table 1, the distinctiveness of `Ugogram` in the landscape of text generation is evident when compared to related works. In addition to the distinctness of problem definition, `Ugogram` distinguishes itself from related works in three aspects:

- **Exclusive Challenges:** Unlike conventional acronym generation which only expects creative words, controllable text generation which requires adhering to controlled elements, and dual text generation that asks for dual dependency, `Ugogram` uniquely integrates the challenges of summarization accuracy for descriptions, neologistic creativity for acronyms, and algorithmic correctness for dependency between descriptions and acronyms.
- **Demand for Novel Evaluation Metrics:** Neologistic and algorithmic requirements of `Ugogram` bring exclusively demand for evaluation metrics for these two aspects, which is unseen in previous text generation tasks.
- **Unique Empowering Application:** the specialty of `Ugogram` uniquely empowers multiple applications like branding, researching, and search engine optimization.

2.3 Three Core Research Questions

Our paper is centered on three key research questions (RQs) based on our task:

- **RQ I:** Considering the unique requirements of heading generation involving both descriptions and acronyms, what steps are essential in constructing a dataset that adequately supports this task?

- **RQ II:** Given the multifaceted nature of heading generation, encompassing summarization, neology, and algorithm, what innovative metrics can be developed to effectively evaluate outcomes in these aspects?
- **RQ III:** Exploring the spectrum of strategies and learning paradigms, which approaches offer promising results for heading generation, particularly in balancing summarization, neology, and algorithmic challenges?

We answer RQ I by showing how we construct a supportive dataset for heading generation in § 3. Then we present our well-designed metrics to answer RQ II in § 4. Finally, we apply different strategies under supervised fine-tuning, reinforcement learning, and in-context learning with LLMs to gauge their effectiveness across the summarization, neologistic, and algorithmic aspects in § 5.

3 Dataset of Lgogram

In this section, we detail the dataset construction process (§ 3.1) and the dataset’s statistics (§ 3.2).

3.1 Dataset Construction

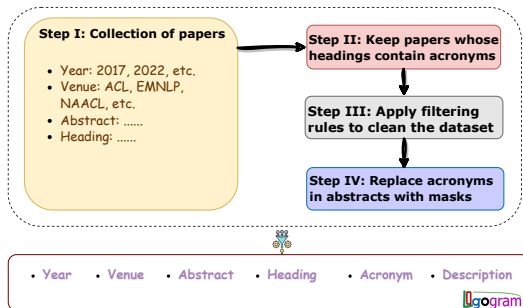


Figure 2: Pipeline of collection and refinement of Lgogram dataset.

Dataset Construction. The construction of Lgogram outlined in Figure 2 involves a four-step process. (i) We crawl the ACL Anthology with 89,047 papers published up to 2023 as our foundational dataset; (ii) The collected dataset was then refined by excluding headings without colons, followed by segmenting each heading at the first colon to separate the acronym from the description; (iii) We further applied a set of tailored filtering rules to eliminate anomalous and low-quality samples; (iv) Acronyms in the abstracts were replaced with a mask to prevent acronym leakage. All processing steps are detailed in App. B.

Dataset Profile. Our final refined Lgogram dataset comprises 6,653 papers from 1989 to 2023 spanning 37 well-recognized venues. The refinement steps and the reliable source ensure the high quality of our collected samples. The dataset is partitioned into training, validation, and test sets with 5,000, 653, and 1,000 papers, respectively. It includes the following metadata: (i) Venue: the conference or journal where the paper was published; (ii) Year: the acceptance year of the paper; (iii) Abstract: the paper’s abstract; (iv) Heading: the paper’s heading; (v) Acronym: the acronym extracted from the paper’s heading; (vi) Description: the description extracted from the paper’s heading. The abstract serves as input, representing the paper’s main idea, for language models to predict the corresponding acronym and description.

3.2 Dataset Statistics

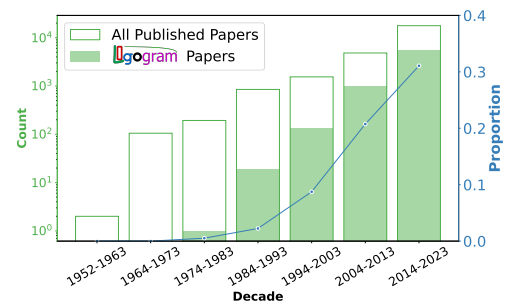


Figure 4: Publication counts across decades. The left y-axis (in green, log-scaled) displays the counts of all ACL Anthology published papers and the counts of papers in Lgogram format. The ratio of Lgogram papers to the total amount of papers is shown in the right y-axis, which is in blue and the linear scale.

w.r.t. Text Length. We plot the distributions of the number of tokens of acronyms, descriptions, and abstracts, along with the number of characters of acronyms in Figure 3. We observe that they are close to normal distributions, indicating that our distributions are symmetric and not skewed. The non-skewed distributions avoid the adverse effects of outliers on model performance, demonstrating the good quality of our dataset.

w.r.t. Publication Counts Across Decades. The counts of all published articles and papers that meet our task requirements (the heading consisting of an acronym and a description) over decades are displayed on the left y-axis of Figure 4 (in green). Though the quantity of published articles is expanding, the number of publications whose titles consist

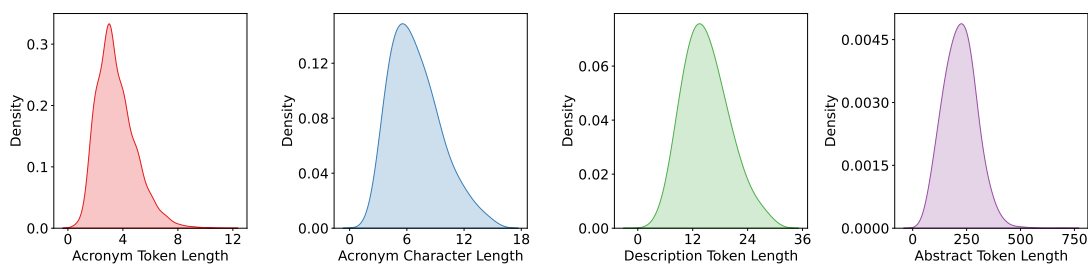


Figure 3: The distributions of acronym token length, acronym character length, description token length, and abstract token length of data samples in **Ugogram**. These curves are plotted with the technique of kernel density estimation (Parzen, 1962), which represents the data using a continuous probability density curve.

of acronyms and descriptions is rising far more quickly. The right y-axis (in blue) of Figure 4 displays the ratio of eligible papers to the total amount of publications over decades. The rising tendency of proportion indicates that more authors are creating more appealing headings by using acronyms and descriptions simultaneously.

4 Automatic Evaluation of **Ugogram**

In this section, we propose automatic evaluation metrics in § 4.1 and discuss their justification evidence in § 4.2. We elaborate on these metrics in detail in App. C.

4.1 Automatic Metrics

Summarization Metrics for Descriptions. To assess if the generated descriptions adequately summarize the abstracts under summarization requirements, we compare them against the ground truth descriptions, using automatic metrics including BLEU (Papineni et al., 2002), METEOR (Banerjee and Lavie, 2005), ROUGE-L (Lin, 2004), CIDEr (Vedantam et al., 2015)), and BERT-Score (Zhang et al., 2020b).

Neologistic Metrics for Acronyms. To measure the resemblance of the generated acronyms to real words, we propose two new metrics: WordLikeness (WL) and WordOverlap (WO).

WordLikeness (WL). We define WordLikeness (WL) as follows:

$$\mathbf{WL} = 1 - \frac{\text{tokenize_num}(A) - 1}{|A|} \quad (2)$$

where A is for the generated lowercase acronym, $\text{tokenize_num}(\ast)$ returns the number of tokenized subwords, and $| \ast |$ denotes the string length of \ast . Our inspiration comes from Byte Pair Encoding (BPE) (Sennrich et al., 2016), which segments words or sequences into most frequently

occurring subword units, akin to morphemes (the smallest grammatical units in a language). To mimic real words, we advocate for segmenting an acronym into longer subword units that resemble morphemes rather than decomposing it into individual letters, thereby minimizing the number of tokens. A high WL value signifies that this acronym consists of morphemes and resembles real words. *WordOverlap (WO).* Our assertion is that an acronym meeting the criteria for a newly coined term should exhibit notable resemblances with pre-existing vocabulary. Thus, we define WordOverlap (WO) as the maximum overlapping similarity between an acronym and any of the existing common words:

$$\mathbf{WO} = \max_{w \in \mathcal{D}} \frac{2 \cdot |\text{LCS}(A, w)|}{|A| + |w|} \quad (3)$$

where \mathcal{D} is a set of collected lowercase common words, detailed in App. C.1. $\text{LCS}(\ast, \ast)$ represents the Longest Common Subsequence (LCS) (Hirschberg, 1977) between two strings. A high WO value shows that the generated acronym overlaps largely with at least one familiar word.

Algorithmic Metric for Acronyms and Descriptions. To assess how much the generated acronym accurately retains the letter sequence from the description in the correct order, we also propose LCSRatio (LR). LCSRatio leverages the Longest Common Subsequence (LCS) to quantify how closely the acronym reflects the sequence of letters found in its corresponding description. A longer LCS means the acronym and the description share a larger portion of its consisting letters. The LR metric is calculated as follows:

$$\mathbf{LR} = \frac{|\text{LCS}(A, D)|}{|A|} \quad (4)$$

where $\text{LCS}(\ast, \ast)$ references the LCS definition provided in Equation (3), and D represents the gen-

erated description in lowercase. A high LR value implies that the acronym successfully encapsulates its description’s letters in the correct order. Further implementation details can be found in App. C.2.

Summarization Metrics for Acronym. Our proposed metrics WL, WO, and LR align well with our neologistic and algorithmic aspects of generated acronyms and descriptions, but they don’t offer ground truth comparison. Therefore, we also compare the generated acronyms with the ground truth acronyms using C-BLEU (from $n = 1$ to 4) (Papineni et al., 2002) and C-ROUGE-L (Lin, 2004), which are detailed in App. C.3.

4.2 Justification of Metrics

Since WL, WO, and LR don’t offer ground truth comparisons, it is intuitive to justify that these metrics are correlated to the quality of headings. We first demonstrate that the gold-standard examples achieve high value in these metrics. Furthermore, we compare WL, WO, and LR values for different manually created acronyms of varying quality. We show that the three metrics are valid for evaluating the neologistic and algorithmic constraints.

Density Estimation of Different Metrics Over *Ugogram*. Figure 5 shows the distributions of WordLikeness (WL), WordOverlap (WO), and LCSRatio (LR) in *Ugogram*. We observe that these three kernel density estimation curves exhibit a pronounced right skew, with values densely clustered near 1 and becoming increasingly sparse as they approach 0. Given that our dataset comprises research works from prestigious venues, the data samples are generally of high quality. The right-skew density estimation curves in turn justify our proposed metrics’ appropriateness in evaluating our task.

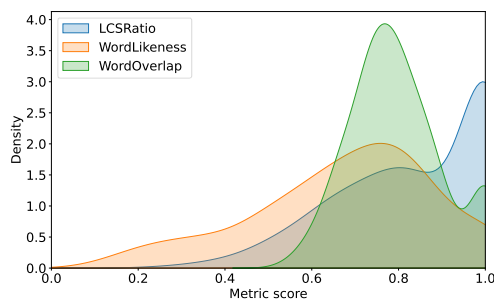


Figure 5: The kernel density estimation curves (Parzen, 1962) of WordLikeness (WL), WordOverlap (WO), and LCSRatio (LR) values of samples in *Ugogram*. The values of WL, WO, and LR all range from 0 to 1.

Joint Distribution in Cubic Space. Figure 6 further displays the joint distribution of WL, WO, and LR within a cubic space, revealing a significant concentration of data points around the (1, 1, 1) coordinate. This indicates that a substantial portion of examples in *Ugogram* simultaneously exhibit high values across all three metrics. Such alignment with the ground truth data underscores the validity of these metrics as measures of quality. If the quality of the generated descriptions and acronyms can be on par with the ground truth, then the corresponding metrics should all be high.

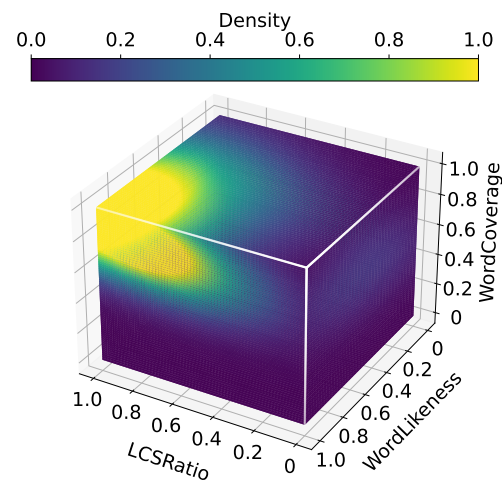


Figure 6: Joint kernel density estimation within a cubic space for *Ugogram*. Axes x , y , and z represent the values of LCSRatio, WordLikeness, and WordCoverage, respectively, each ranging from 0 to 1. Color intensity indicates density levels, with lighter shades denoting higher densities.

Performance of Metrics on Manually Created Examples. We provide several manually created examples, as opposed to the real ones from the dataset, and further clarify how LCSRatio, WordLikeness, and WordOverlap metrics are applied. Table 2 compares the WL, WO, and LR values for various acronyms derived from the same description. The acronym “SVD” has low scores across all metrics, as it fails to meet the neologistic and algorithmic requirements. “Sugar” exhibits relatively high WL and WO values, due to its resemblance as a word. However, its LR value is low because its constituent letters and the description are not in the same alphabetical order. Conversely, “SMfATE” demonstrates a superior LR value, reflecting its derivation from the initial letters of the description. However, the WL and WO values are low because it is not similar to any recognizable

word. “SMARTies” scores highly across all metrics, benefiting from its resemblance to the term “smartie”—connoting intelligence—and its association with the name of a confectionery brand, in addition to its formulation from the letters in the description. From the above examples, we elaborate that the WL, WO, and LR metrics are justified indicators for the neologistic and algorithmic aspects.

Acronym	WL	WO	LR
SVD	0.33	0.80	0.67
Sugar	1.00	1.00	0.60
SMfATe	0.50	0.80	1.00
SMARTies	0.88	0.88	1.00
Description	Sentiment Models for Arabic Target entities		

Table 2: Comparison of WL, WO, and LR values by an example acronym. The higher the better.

5 Empirical Study on

This section addresses RQ III, which focuses on assessing the effectiveness of various strategies and learning paradigms when applied to language models. We specifically examine their performance on summarization, neologistic capabilities, and algorithmic dimensions. We first introduce several potential strategies in § 5.1. Following this, in § 5.2, we demonstrate the usage of these strategies across different learning paradigms. Finally, we thoroughly assess the effectiveness of the strategies and learning paradigms across the summarization, neologistic, and algorithmic performance in 5.3.

5.1 Possible Strategies

Generation Ordering. The rationale behind this strategy is the importance of text generation order for autoregressive models, where the content generated later depends on what is generated first. The decision diverges on whether to first generate the description and followed by the acronym (○) or to generate the acronym and then the description (●).

Tokenization of Acronyms. Since the letter from acronym is derived from letters in the description, adjacent characters in acronyms might not constitute a token found in the dictionary. Thus, beyond the standard tokenization method that processes the entire acronym (□), we explore the potential benefit of tokenizing each separated letter of the acronym (■). This approach is considered as it may

enhance language models’ understanding that an acronym is composed of individual letters from the description.

Framework Design. Since our task involves the generation of descriptions and acronyms, a natural divergence is whether to use a onestop approach — employing a single language model that generates these two outputs separated by a delimiter in a onestop fashion (◇) — or generating one before the other type of output with two language models in a sequential pipeline fashion (◆).

5.2 Learning Paradigms

Supervised Fine-Tuning (SFT). SFT is the most straightforward learning paradigm. Concretely, we fine-tune a pre-trained language model using various strategies with supervised data. The model receives an abstract text as input and produces a description and an acronym as outputs. We investigate the three abovementioned strategies under the SFT learning paradigm: Generation Order, Tokenization, and Framework Design. Specifically, we fine-tune generative pre-trained language models T5 (Raffel et al., 2020), and conduct a set of comparative studies to explore the effectiveness of different strategies (○ vs. ●, □ vs. ■, and ◇ vs. ◆). Details about our experiment setup and fine-tuning baselines are in App. D.

Reinforcement Learning (RL). Building upon the foundation of the SFT paradigm, we extend our exploration to the RL paradigm, which aims to improve the quality of the generated outputs through a reward-based mechanism.

Specifically, we choose the Proximal Policy Optimization (PPO) algorithm (Ziegler et al., 2019) to implement the RL paradigm. This process initiates with a baseline policy π_{ϕ}^{SFT} from the SFT model and progresses to a refined policy π_{ϕ}^{RL} which aims to enhance the overall quality r of the output, measured by a geometric mean of WL, WO, and LR scores. Meanwhile, the optimized model maintains proximity to the original SFT model. The enhancement is guided by optimizing the reward function r_{total} , defined as

$$r_{\text{total}} = r - \eta \text{KL}(\pi_{\phi}^{\text{RL}}, \pi_{\phi}^{\text{SFT}}) \quad (5)$$

where KL denotes the Kullback–Leibler divergence and η is a coefficient regulating the strength of the KL penalty. We elaborate on the details in App. E. We evaluate all strategies with the exception of

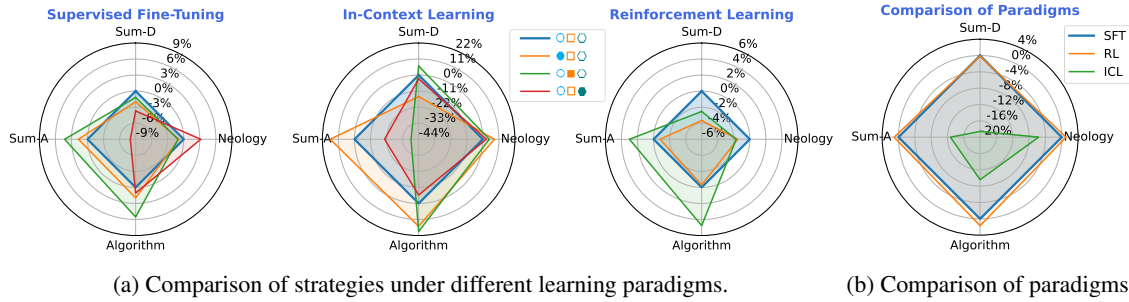


Figure 7: A comparative analysis of different strategies and learning paradigms. The axes labeled *Sum-A* and *Sum-D* quantify the average performance metrics for acronym and description summarization, respectively. The axis of *Algorithm* displays the LR value while the *Neology* axis reflects the average value of WL and WO. Subfigure (a) shows the relative change of each strategy (● □ ◇, ● □ ◇, and ● □ ◇) over the baseline (● □ ◇) under Supervised Fine-tuning (SFT), Reinforcement Learning (RL), and In-Context Learning with LLMs (ICL). Subfigure (b) zooms in on the performance variation within the same strategy (● □ ◇) across learning paradigms. It highlights the improvement or decline compared to the SFT baseline, which is bolded blue with a 0% change.

framework design (○ vs. ● and □ vs. ■). This exception is made due to the relatively unexplored territory of feedback mechanisms within Reinforcement Learning (RL) for pipeline language models.

In-Context Learning with LLMs (ICL). Apart from SFT with massive training data, we also adopt LLMs (OpenAI et al., 2023) with advanced few-shot prompting techniques (Wei et al., 2022). Specifically, we test the one-shot performance of gpt-4-1106-preview version of GPT-4 (OpenAI et al., 2023) on the heading generation task. Detailed setup is in App. F.

5.3 Experiment Results

Comparison of Different Generation Strategies.

Figure 7a illustrates the radar plot for comparing various strategies across different learning paradigms, evaluated from three dimensions: summarization, neologistic, and algorithmic performance. Key takeaways are:

- Certain strategies under different paradigms show similar performance in specific aspects.
- Overall, strategies demonstrate inconsistency across different learning paradigms, with patterns varying significantly.
- Strategies in SFT and RL exhibit similar patterns, differing greatly from those in ICL.
- Not a single strategy excels across all aspects within each learning paradigm, challenging the identification of an optimal strategy.

Table 6 and App. G.1 further detail the performance of various strategies in each learning paradigm.

Comparison of Different Learning Paradigms.

We further compare the performance of different learning paradigms employing the same strategy (○ □ ◇) depicted in Figure 7b. SFT and RL yield comparable results, with RL slightly outperforming in neologistic and algorithmic aspects. Conversely, ICL performs significantly worse than both SFT and RL. This subpar performance is particularly due to the novelty of the task for LLMs and the fact that LLMs are not optimized for the *Logogram* dataset. Table 6 in App. G.1 displays detailed performance.

Case Study.

Table 3 presents a case study comparing various strategies under different learning paradigms, revealing that the ground truth description and acronym meet the requirements for summarization, neology, and algorithmic constraints effectively. However, while most strategies adeptly address the summarization requirement across learning paradigms, they often struggle to meet the neologistic with algorithmic constraints simultaneously. Specifically, the ground-truth description, “Discriminator Cooperative Unlikelihood Prompting for Controllable Text Generation” accurately summarizes the abstract and the main idea of the paper (Zhang and Song, 2022) which investigates controllable text generation. Moreover, the ground-truth acronym, “DisCup”, meets the neologistic (like the variant form of the real word “cup” and beginning with a common prefix of “dis”) and algorithmic criteria (letters all come from the description). On the other hand, generated acronyms fail to satisfy the neologistic constraint (“ATROPT” and “AtticCLM”), the algorithmic constraint (“Ad-

vProbe”), or both constraints (“PCTG”).

Strategies	Descriptions	Acronyms
<i>Supervised Fine-Tuning Paradigm</i>		
○□○	Attribute Knowledge-Attribute-Discriminator for Text Generation	AtticCLM
●□○	Prompt Learning with Attribute Knowledge	P-CTG
○■○	attribute-discriminator for Text Generation	PromptCTG
○□●	Frozen Language Model Control with Attribute Knowledge	AdvProbe
<i>In-Context Learning Paradigm</i>		
○□○	Enhancing Controlled Text Generation in Casual Language Models through Attribute-Discriminator Optimized Prompt Learning	ATROPT
●□○	Attribute-Controlled Text Generation via Discriminator-Optimized Prompts in Frozen Casual Language Models	ACT-Prompt
○■○	Enhanced Attribute-Controlled Text Generation with Discriminator-Optimized Prompts	EACTG
○□●	Enhanced Attribute-Controllable Text Generation with Discriminator-Optimized Control-Prompts in Large Casual Language Models	DisCoGen
<i>Reinforcement Learning Paradigm</i>		
○□○	Controlled Attribute Knowledge for Attribute Controlled Text Generation	AttCraft
●□○	Prompt Learning with Attributes	P-CTG
○■○	attribute-discriminator for Text Generation	PromptCTG
○□●	N/A	N/A
Ground Truth	Discriminator Cooperative Unlikelihood Prompt-tuning for Controllable Text Generation	DisCup

Table 3: A case study for generating a heading that consists of an acronym and a description given the abstract of (Zhang and Song, 2022).

6 Related Work

Acronym Construction. The generation of acronyms involves creating memorable and communicative phrases. Tsuruoka et al. (2005) proposed a generative approach for acronym construction, wherein the acronym generation is formalized as a sequence labeling problem like part-of-speech tagging, and they adopt conventional Markov modeling methods for the acronym generation. Appelman (2020) focuses on the importance of acronyms in news headlines and defines acronyms as word-like strings from letters in the headlines. Li et al. (2022) presented a method for acronym extraction from definition text using a prompt-based sequence generation approach. Existing tasks in acronym construction do not address the simultaneous generation of both acronyms and their descriptions. We introduce **Ugogram** to pioneer the field of jointly generating acronyms and descriptions.

Controllable Text Generation. Controllable Text Generation (CTG) (Zhang et al., 2022) is the process of producing text that meets specific criteria. Research in this area, leveraging Pre-trained Language Models (PLMs) for CTG, can generally be categorized into three distinct methodologies. The first approach is to fine-tune the PLM

with supervised data (Nan et al., 2021; Zhang and Song, 2022), which demands many supervised data samples. The second one is to re-train the PLM from scratch, which is very costly. Typical examples include CTRL (Keskar et al., 2019), POINTER (Zhang et al., 2020c), etc. The third approach (Holtzman et al., 2020; Scialom et al., 2022) is to re-rank the generated outputs in a post-processing manner based on the criterion of the controllable task.

Dual Text Generation. Dual Text Generation (DTG) involves the simultaneous creation of two interdependent text outputs. This field concentrates on generating parallel text formats, such as translation and simultaneous summarization. Xia et al. (2016) pioneered the concept of duality in machine translation by connecting both backward and forward translation processes, showing the potential for enhancing translation accuracy and coherence through DTG approaches. Rashid et al. (2019) proposed Bilingual-GAN, a bilingual generative adversarial network (GAN) architecture, to tackle the dual text generation task. Bao et al. (2022) further explored this problem by developing Latent-GLAT, a latent variable model that uses discrete latent variables to capture word categorical information.

7 Conclusions

In this paper, we introduce a novel task that aims at generating headings that comprise both descriptions and acronyms, supported by a dedicated dataset, **Ugogram**, and tailored evaluation metrics. This task poses challenges in summarization, neologistic, and algorithmic aspects. Furthermore, we propose different generation strategies, including generation ordering, tokenization of acronyms, and framework design, and assess their effectiveness across learning paradigms such as supervised fine-tuning, reinforcement learning, and in-context learning with LLMs. Our findings highlight the complexities involved in striking an optimal balance between summarization, neologism, and algorithmic constraints, demonstrating the intricate nature of this task.

Acknowledgements

We are greatly grateful to Ping and Zongyang for their help and discussion of this work and for EPFL’s financial and IT support. Yifan Hou is supported by the Swiss Data Science Center PhD Grant (P22-05).

Limitations

Despite the value of the contributed task, the supportive dataset, the evaluation metrics, and the introduced strategies, some limitations require special attention. Firstly, the scope of the dataset is relatively limited, mainly to academic heading generation, though our task can be applied to various scenarios. It is a very tricky balance between contributing a comprehensive dataset and not violating any copyright usage policies. Secondly, though the currently proposed metrics meet our task’s requirements quite well, some other possible evaluation aspects could also be considered, such as engagement and clarity. We encourage more investigation on the evaluation of heading generation from different perspectives. In summary, we acknowledge the limitations of our work. However, our work sheds light on a novel promising generation topic and contributes its supportive dataset and evaluation metrics.

Ethical Considerations

The dataset we collect is under the MIT License for research objectives. The licenses of packages that we use are listed in Table 5. All models (App. D and E) and the LLMs’ API we invoke (App. F) are revealed to the public. Another ethical concern requiring special attention is the potential for language models to generate acronyms with negative connotations, such as “HARD” or “MEAN”. Users should be particularly careful when designing acronyms to avoid negative connotations.

References

- Somayeh Abdi and Abdollah Irandoust. 2013. [The importance of advertising slogans and their proper designing in brand equity](#). *International Journal of Organizational Leadership*, 2(2):62–69.
- Alyssa Appelman. 2020. “alphabet soup”: Examining acronym and abbreviation style in headlines. *Journalism Practice*, 14(7):880–895.
- Satanjeev Banerjee and Alon Lavie. 2005. [METEOR: An automatic metric for MT evaluation with improved correlation with human judgments](#). In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.
- Yu Bao, Hao Zhou, Shujian Huang, Dongqi Wang, Lihua Qian, Xinyu Dai, Jiajun Chen, and Lei Li. 2022. [latent-GLAT: Glancing at latent variables for parallel text generation](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8398–8409, Dublin, Ireland. Association for Computational Linguistics.
- Steven Bird and Edward Loper. 2004. [NLTK: The natural language toolkit](#). In *Proceedings of the ACL Interactive Poster and Demonstration Sessions*, pages 214–217, Barcelona, Spain. Association for Computational Linguistics.
- Claire Brierley and Eric Atwell. 2008. [ProPOSEL: A prosody and POS English lexicon for language engineering](#). In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC’08)*, Marrakech, Morocco. European Language Resources Association (ELRA).
- Pengshan Cai, Kaiqiang Song, Sangwoo Cho, Hongwei Wang, Xiaoyang Wang, Hong Yu, Fei Liu, and Dong Yu. 2023. [Generating user-engaging news headlines](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3265–3280, Toronto, Canada. Association for Computational Linguistics.
- Lihu Chen, Gael Varoquaux, and Fabian M. Suchanek. 2023. [GLADIS: A general and large acronym disambiguation benchmark](#). In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2073–2088, Dubrovnik, Croatia. Association for Computational Linguistics.
- Michal Danilák. 2023. [langdetect](#). Version information, Accessed: 2023-12-07.
- Mayukh Dass, Chiranjeev Kohli, and Manaswini Acharya. 2023. [An investigation into slogan design on creating slogan-brand alignment: Message clarity and creativity enhance while jingles and rhymes weaken alignment](#). *Journal of Advertising Research*, 63(1):43–60.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. [BERT: Pre-training of deep bidirectional transformers for language understanding](#). In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.
- Emily Dinan, Angela Fan, Adina Williams, Jack Urbanek, Douwe Kiela, and Jason Weston. 2020. [Queens are powerful too: Mitigating gender bias in dialogue generation](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8173–8188, Online. Association for Computational Linguistics.
- Dirk Goldhahn, Thomas Eckart, and Uwe Quasthoff. 2012. [Building large monolingual dictionaries at the](#)

- Leipzig corpora collection: From 100 to 200 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 759–765, Istanbul, Turkey. European Language Resources Association (ELRA).
- Sylvain Gugger, Lysandre Debut, Thomas Wolf, Philipp Schmid, Zachary Mueller, Sourab Mangrulkar, Marc Sun, and Benjamin Bossan. 2022. Accelerate: Training and inference at scale made simple, efficient and adaptable. <https://github.com/huggingface/accelerate>.
- Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. *Array programming with NumPy. Nature*, 585(7825):357–362.
- Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. *Dual learning for machine translation*. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 820–828.
- Daniel S. Hirschberg. 1977. *Algorithms for the longest common subsequence problem*. *J. ACM*, 24(4):664–675.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. *The curious case of neural text degeneration*. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- John D. Hunter. 2007. *Matplotlib: A 2d graphics environment*. *Comput. Sci. Eng.*, 9(3):90–95.
- Nitish Shirish Keskar, Bryan McCann, Lav R. Varshney, Caiming Xiong, and Richard Socher. 2019. *CTRL: A conditional transformer language model for controllable generation*. *CoRR*, abs/1909.05858.
- Diederik P. Kingma and Jimmy Ba. 2017. *Adam: A method for stochastic optimization*.
- Bin Li, Fei Xia, Yixuan Weng, Bin Sun, Shutao Li, and Xiusheng Huang. 2022. *PSG: prompt-based sequence generation for acronym extraction*. In *Proceedings of the Workshop on Scientific Document Understanding co-located with 36th AAAI Conference on Artificial Intelligence, SDU@AAAI 2022, Virtual Event, March 1, 2022*, volume 3164 of *CEUR Workshop Proceedings*. CEUR-WS.org.
- Chin-Yew Lin. 2004. *ROUGE: A package for automatic evaluation of summaries*. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. *Roberta: A robustly optimized BERT pretraining approach*. *CoRR*, abs/1907.11692.
- Prakhar Mishra, Chaitali Diwan, Srinath Srinivasa, and G. Srinivasaraghavan. 2021. *Automatic title generation for text with pre-trained transformer language model*. In *15th IEEE International Conference on Semantic Computing, ICSC 2021, Laguna Hills, CA, USA, January 27-29, 2021*, pages 17–24. IEEE.
- Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangu Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiyaz Rahman, Ahmad Zaidi, Mutethia Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021. *DART: Open-domain structured data record to text generation*. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 432–447, Online. Association for Computational Linguistics.
- OpenAI, :, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook Kim, Christina Kim, Yongjik Kim, Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo,

- Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O’Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lillian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2023. [Gpt-4 technical report](#).
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. [Bleu: a method for automatic evaluation of machine translation](#). In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.
- Emanuel Parzen. 1962. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. [Pytorch: An imperative style, high-performance deep learning library](#). In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *J. Mach. Learn. Res.*, 21:140:1–140:67.
- Ahmad Rashid, Alan Do-Omri, Md. Akmal Haidar, Qun Liu, and Mehdi Rezagholizadeh. 2019. [Bilingual-GAN: A step towards parallel text generation](#). In *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, pages 55–64, Minneapolis, Minnesota. Association for Computational Linguistics.
- Leonardo F. R. Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. [Structural adapters in pretrained language models for AMR-to-Text generation](#). In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4269–4282, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.
- Leonard Richardson. 2017. [Beautiful soup documentation](#).
- Punyajoy Saha, Kanishk Singh, Adarsh Kumar, Binny Mathew, and Animesh Mukherjee. 2022. [Countergedi: A controllable approach to generate polite, detoxified and emotional counterspeech](#). In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 5157–5163. ijcai.org.
- Thomas Scialom, Tuhin Chakrabarty, and Smaranda Muresan. 2022. [Fine-tuned language models are continual learners](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 6107–6122, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. [Neural machine translation of rare words with subword units](#). In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.
- Noam Shazeer and Mitchell Stern. 2018. [Adafactor: Adaptive learning rates with sublinear memory cost](#). In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, July 10-15, 2018*, volume 80

- of *Proceedings of Machine Learning Research*, pages 4603–4611. PMLR.
- Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. 2017. [Question answering and question generation as dual tasks](#). *CoRR*, abs/1706.02027.
- Yoshimasa Tsuruoka, Sophia Ananiadou, and Jun’ichi Tsujii. 2005. [A machine learning approach to acronym generation](#). In *Proceedings of the ACL-ISMB Workshop on Linking Biological Literature, Ontologies and Databases: Mining Biological Semantics*, pages 25–31, Detroit. Association for Computational Linguistics.
- Ramakrishna Vedantam, C. Lawrence Zitnick, and Devi Parikh. 2015. [Cider: Consensus-based image description evaluation](#). In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 4566–4575. IEEE Computer Society.
- Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, and Shengyi Huang. 2020. [trl: Transformer reinforcement learning](#). <https://github.com/huggingface/trl>.
- Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2022. [Chain-of-thought prompting elicits reasoning in large language models](#). In *NeurIPS*.
- Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. [Transformers: State-of-the-art natural language processing](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.
- Nuwa Xi, Sendong Zhao, Haochun Wang, Chi Liu, Bing Qin, and Ting Liu. 2023. [UniCoRN: Unified cognitive signal Reconstruction bridging cognitive signals and human language](#). In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13277–13291, Toronto, Canada. Association for Computational Linguistics.
- Yingce Xia, Di He, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. [Dual learning for machine translation](#). *CoRR*, abs/1611.00179.
- Bang Yang, Fenglin Liu, Zheng Li, Qingyu Yin, Chenyu You, Bing Yin, and Yuexian Zou. 2023. [Multimodal prompt learning for product title generation with extremely limited labels](#). In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 2652–2665, Toronto, Canada. Association for Computational Linguistics.
- Hanqing Zhang and Dawei Song. 2022. [DisCup: Discriminator cooperative unlikelihood prompt-tuning for controllable text generation](#). In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3392–3406, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.
- Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. [A survey of controllable text generation using transformer-based pre-trained language models](#). *CoRR*, abs/2201.05337.
- Rui Zhang and Joel Tetreault. 2019. [This email could save your life: Introducing the task of email subject line generation](#). In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 446–456, Florence, Italy. Association for Computational Linguistics.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020a. [Bertscore: Evaluating text generation with BERT](#). In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020b. [Bertscore: Evaluating text generation with BERT](#). In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.
- Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020c. [POINTER: Constrained progressive text generation via insertion-based generative pre-training](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8649–8670, Online. Association for Computational Linguistics.
- Rui Zheng, Shihan Dou, Songyang Gao, Yuan Hua, Wei Shen, Binghai Wang, Yan Liu, Senjie Jin, Qin Liu, Yuhao Zhou, Limao Xiong, Lu Chen, Zhiheng Xi, Nuo Xu, Wenbin Lai, Minghao Zhu, Cheng Chang, Zhangyue Yin, Rongxiang Weng, Wensen Cheng, Haoran Huang, Tianxiang Sun, Hang Yan, Tao Gui, Qi Zhang, Xipeng Qiu, and Xuanjing Huang. 2023. [Secrets of rlhf in large language models part i: Ppo](#).
- Daniel M. Ziegler, Nisan Stiennon, Jeffrey Wu, Tom B. Brown, Alec Radford, Dario Amodei, Paul F. Christiano, and Geoffrey Irving. 2019. [Fine-tuning language models from human preferences](#). *CoRR*, abs/1909.08593.

A Applications of Heading Generation

In this section, we explore the various applications of heading generation.

Branding and Marketing. An engaging slogan plays a crucial role in the marketing and advertising of commercial companies. It helps to craft appealing headings for product descriptions and promotion policies to attract customers' interest (Abdi and Irandoust, 2013; Dass et al., 2023). Additionally, it assists in creating catchy and memorable brand names that enhance market recognition.

Academia and Research. To attract a broader audience to research papers, it is essential to construct compelling titles (Mishra et al., 2021). Besides, for complex scientific terms or project topics, the emergence of acronyms makes the work more memorable and easier to refer to. It also facilitates the understanding of the research.

Search Engine Optimization. An algorithm-friendly acronym is beneficial for content to be retrieved by search engines (Tsuruoka et al., 2005; Li et al., 2022; Chen et al., 2023). Our work can not only generate a description that is accessible to human readers but also to create an algorithm-friendly acronym for search engines.

B Details of Construction

The construction process of the dataset, as outlined in the main text, involves several critical steps. These steps, detailed from App. B.1 through to App. B.4, correspond sequentially to steps (i) through (iv) as discussed in § 3.1.

B.1 Crawling Process

All samples are collected from the root page of ACL Anthology². We iterate all conferences listed in the root page of ACL Anthology. For each conference, we first fetch the content of the main page. Then, we parse the HTML content using BeautifulSoup³ to obtain papers' webpage links. With the link, we iterate through all papers to extract details like headings and abstracts.

²Based on the claim of ACL, materials prior to 2016 are under the license of Creative Commons Attribution-NonCommercial-ShareAlike 3.0 International License (CC BY-NC-SA 3.0 DEED). Materials after 2016 are under the license of Creative Commons Attribution 4.0 International License (CC BY 4.0 DEED).

³<https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

However, for quite a large portion of papers, there are blanks for papers' abstracts. To overcome this knotty cases, we downloaded the pdf files of these papers that miss abstracts and extract the abstract from these PDF files using PyMuPDF⁴. We suppose that the abstract is the content between the section of Abstract and Introduction.

B.2 Extraction of Acronyms and Descriptions

When crawling from ACL anthology, we also extract acronyms A and descriptions D from headings in the crawled dataset. We first discard examples whose heading does not have an acronym or a description. We achieve this by applying a formatting pattern $\langle A \rangle : \langle D \rangle$ to match each heading in the data. If the data does not have any colon, we drop it. Otherwise, we split on the first colon and regard the acronym as strings before the colon, and the description as strings after the colon. This process resulted in 38,784 headings with pairs of an acronym and a description.

B.3 Removing Low Quality Data

After keeping those examples with acronyms, we clean our data and filter out anomalies by applying a set of tailored filtering rules as follows:

1. Basic Cleaning:
 - (a) Replace all newline characters with spaces, and all "`-\n`" with an empty string. We do that because PDF files have actual line breaks at the end of every line in the PDF.
 - (b) Strip whitespace from all string columns.
 - (c) Drop duplicates and rows with missing values or empty string.
2. Special Processing for Acronyms:
 - (a) Replace underscores with spaces in the Acronym column.
 - (b) Remove rows where the Acronym column contains more than one word.
 - (c) Remove rows where the Acronym column contains characters other than letters and hyphens.
3. Special Processing for Abstracts:
 - (a) Replace the first "Abstract" string with an empty string in the Abstract column,

⁴<https://pymupdf.readthedocs.io/>

since many crawled abstracts start with this string.

- (b) Remove strings after the last period in the Abstract column.
- (c) Remove non-English abstracts with the langdetect (Danilák, 2023) package.

4. Outlier Removal:

- (a) Get distributions of lengths in acronyms (char-level), descriptions (token-level)⁵, and abstracts (token-level).
- (b) Remove outliers that are more than 3 standard deviations away from the mean.

To summarize, we employ some basic text cleaning and make sure that 1) the acronym is a single English word and 2) the abstract is in English and ends with a period. After the cleaning, 6,653 examples are left.

B.4 Preventing Acronym Leakage

There is still one issue in our cleaned data that the abstract will reveal the acronym in its texts. The language therefore can easily cheat and simply copy the acronym in the abstract, instead of deriving from the description or abstract. To avoid that, we mask acronyms with the abstract and replace all matched ones with <MASKED_ACRONYM>.

For the description, we don't mask them mentioned in the abstract. We think hiding the description is unnecessary because it is very common for researchers to use a part of the abstract as the description part of the heading. Besides, if the abstract contains the description, the description serves the purpose of the paper, and removing it will reduce the clarity of the abstract. Therefore, we allow language models to copy words from the abstract as the description.

C Details of Automatic Metrics

C.1 Neologistic Evaluation Metrics

WordLikeness (WL). The code below shows the detailed implementation of WL:

```
1 def calculate_wordlikeness(self, acronym):
2     if acronym == "":
3         return 0
4     else:
5         acronym = acronym.lower()
6         return 1 - \
7             (len(self._tokenizer.tokenize([
8                 e for e in
9                 ↪ self.tokenizer.tokenize(acronym) if
10                ↪ e != ' '))
```

⁵We use the T5 tokenizer from Huggingface.

```
9         ))-1) / len(acronym)
```

We utilize the T5 Tokenizer as our tokenizer. Since the tokens contain meaningless underlined dash symbols ' _ ', we need to remove them when counting the number of tokens of the acronym. All of the acronyms are lowercase in WL and the same is true for the other metrics mentioned later.

WordOverlap (WO). We implement the WO metric in the code below:

```
1 import difflib
2
3 def calculate_wordoverlap(self, acronym):
4     acronym = acronym.lower()
5     return max(difflib.SequenceMatcher(None, acronym,
6     ↪ word).ratio() for word in self._dictionary)
```

We employ the Python difflib library and use the class SequenceMatcher to compare pairs of sequences of any type. The ratio() method returns a measure of the sequences' similarity:

$$\text{similarity} = \frac{2 \cdot |\text{LCS}(s_1, s_2)|}{|s_1| + |s_2|} \quad (6)$$

where $\text{LCS}(*, *)$ represents the Longest Common Subsequence (LCS) (Hirschberg, 1977) between two sequences. s_1 and s_2 are two lowercase strings. $|*|$ denotes the length of a string.

In the WO metric, we measure the sequences' similarity between the acronym and every common English word in a predefined dictionary. To collect common English words, we source from the corpora eng_news_2020_1M⁶ in Leipzig Corpora (Goldhahn et al., 2012) under CC BY-NC 4.0, including words and their frequencies. We then drop out empty strings and non-English words. After that, we lowercase all words and merge all duplicate words and their counts. Finally, we remove all NLTK stopwords⁷ from the dictionary and keep the top 10% frequent words, resulting in 21,610 words in our final dictionary.

C.2 Algorithmic Evaluation Metrics

LCSRatio (LR). The LR metric measures how well the generated acronym catches the letters from the description in the right sequence. Specifically, we use dynamic programming to derive the LCS between the acronym and the description and quantify the proportion of the acronym in the LCS. We implement the LR metric by the following code:

⁶https://wortschatz.uni-leipzig.de/en/download/English#eng_news_2020

⁷<https://www.nltk.org/search.html?q=stopwords>

```

1 def calculate_lcsratio(self, acronym, description):
2     if acronym == "" or description == "":
3         return 0
4     else:
5         acronym = acronym.lower()
6         description = description.lower()
7         lcs = self._lcs(acronym, description)
8         return len(lcs) / len(acronym)

```

C.3 Other Acronym Evaluation Metrics

C-BLEU. C-BLEU is a weighted geometric mean of all the modified n -gram character-level precisions, multiplied by the brevity penalty.

$$\text{C-BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^N w_n \log p_n\right), \quad (7)$$

where w_n is the weight for character-level n -gram. BP is the brevity penalty term

$$\text{BP} = \begin{cases} 1 \\ \exp(1 - \text{len}(S)/\text{len}(C)) \end{cases}, \quad (8)$$

where C and S are the generated acronym and the reference, respectively. p_n is the modified n -gram precision that is calculated as

$$p_n = \frac{\sum_{n\text{-gram} \in C} \text{Count}_{\text{clip}}(n\text{-gram})}{\sum_{n\text{-gram} \in C} \text{Count}(n\text{-gram})} \quad (9)$$

where C is the reference acronym and $\text{Count}(n\text{-gram})$ is the number of this letter-level n -gram in C . For the clip term in the numerator of Equation (9) $\text{Count}_{\text{clip}}(n\text{-gram}) = \min(\text{Count}, \text{Max_Ref_Count})$. In other words, each character-level gram should not exceed the largest count observed in any single reference for that character-level gram, i.e., MAX_Ref_Count .

C-ROUGE-L. C-ROUGE-L is based on the longest common substring (LCS) between the generated acronym and reference. A longer shared sequence indicates more similarity between two words.

$$\begin{cases} R_{LCS} = \frac{LCS(C,S)}{\text{len}(S)} \\ P_{LCS} = \frac{LCS(C,S)}{\text{len}(C)} \\ F_{LCS} = \frac{(1+\beta^2)R_{LCS}P_{LCS}}{R+\beta^2P_{LCS}} \end{cases} \quad (10)$$

where C and S are the generated acronym and the reference, respectively.

C.4 Implementation of ROUGE-L and C-ROUGE-L Metrics

We employed the HuggingFace’s evaluate library to implement the ROUGE metrics, as detailed in

Table 5. Our reports use the “rougeL” value for scoring. The parameters are set to default with `use_aggregator=True` and `use_stemmer=False`, which means that we return aggregate scores and do not employ the Porter stemmer for suffix stripping. This implementation serves as a wrapper around Google Research’s reimplementation of ROUGE⁸, which differs slightly from the original Perl script implementation. The original version splits text at `\n` for processing multiple sentences, while our approach does not recognize the `\n` symbol. However, as our task focuses on heading generation without producing the `\n` symbol, both implementations function equivalently in our context.

D Supervised Fine-Tuning

D.1 Training Details

We finetune T5-base models (Raffel et al., 2020) (220M parameters) for all fine-tuning strategies. For the pipeline strategy, we train two separate T5 models for the two steps. All models are under the same training configuration. Specifically, we optimize each model on four NVIDIA TITAN Xp GPUs in parallel with a batch size of 8 per device. We fine-tune these models for 5 epochs and use Huggingface Adafactor (Shazeer and Stern, 2018), the default optimizer for the T5-base model with a default learning rate of 1e-3. We set the `max_length` to 512 and the `max_decode_step` to 64. The configurations of inference remain the same as fine-tuning.

D.2 Input and Output Format for Each Strategy

The input format for each strategy is shown in Table 4. There are a few points that need to be clarified: (i) we use the colon symbol (:) as the separator for the acronym and the description. If the model fails to generate a colon symbol during inference, we treat the entire output as the description and the acronym becomes an empty string. All empty acronyms have zero values for all metrics related to the acronym evaluation; (ii) to implement the char-level tokenization, we insert spaces into every two adjacent letters of the acronym. We ensure the tokenizer will split the acronym into letters instead of subtokens. During inference, we split the acronym by spaces and join the letters together; (iii) we have two models for the pipeline strategy

⁸<https://github.com/google-research/google-research/tree/master/rouge>

and thus two formats; (iv) we prepend a prefix to every piece of information to indicate its meaning to follow the T5 fashion.

E Reinforcement Learning

E.1 PPO Algorithm

The Proximal Policy Optimization (PPO) algorithm is a reinforcement learning approach that could be used to fine-tune language models towards human preferences (Ziegler et al., 2019). It starts with an original policy π_{ϕ}^{SFT} and evolves to a new policy π_{ϕ}^{RL} .

To quantify how close the output is to the human requirements, we measure with a reward model $r(x, y)$, where x is the model input and y is the output. In our task, $r(x, y)$ is the geometric average of the WL, WO, and LR values (defined in § 4.1) of the acronym from the output y . An extra term is introduced into $r(x, y)$, which serves as a penalty based on the Kullback-Leibler (KL) divergence between the learned RL policy π_{ϕ}^{RL} and initial supervised model π_{ϕ}^{SFT} (Zheng et al., 2023). The total reward is calculated as follows:

$$r_{\text{total}} = r(x, y) - \eta \text{KL}(\pi_{\phi}^{\text{RL}}(y|x), \pi_{\phi}^{\text{SFT}}(y|x)), \quad (11)$$

where η is the KL coefficient and controls the strength of the KL penalty. The total reward not only fosters the language model to generate better acronyms that increase $r(x, y)$, but also prevents the model from deviating drastically from the original model.

Additionally, the PPO algorithm employs a clipping mechanism in the policy objective function to limit policy updates, ensuring they don't diverge too far from the old policy, thus maintaining training stability. The value function in PPO, a critical component, estimates the expected return of a state, guiding the policy towards more rewarding states. This combination of reward optimization, clipping, and value estimation, allows PPO to fine-tune language models effectively while balancing between exploration and exploitation. The details of the PPO algorithm can be found in (Zheng et al., 2023).

E.2 Training Details

We use the Huggingface Transformer Reinforcement Learning (TRL) library for the implementation of the PPO Trainer. The PPO models originate

from the SFT models trained with different strategies ($\circ \square \diamond$, $\bullet \square \diamond$, $\circ \blacksquare \diamond$). Therefore, the input format is the same as SFT, shown in Table 4. We do not consider $\circ \square \blacklozenge$ because the feedback mechanism in RL for pipelined language models is still unexplored. The reward value is the geometric average of the values of WL, WO, and LR. We further fine-tune our model with PPO for 1 epoch with a batch size of 8 per GPU device, the same device as the supervised fine-tuning. For each batch, we sample with a minibatch size of 2 per GPU device and train 100 mini epochs for the PPO update. We utilize the Adam optimizer (Kingma and Ba, 2017) with a learning rate of 1.41e-7. We use the Adaptive KL controller (Ziegler et al., 2019) and tune the KL coefficient η from 0.002 to 0.001. We set the threshold of PPO ratios to 10 to skip mini-batches with high PPO ratios that can cause loss spikes. All other hyperparameters follow the default settings in the TRL Trainer.

F In-Context Learning

We implement four methods for the in-context learning paradigm.

- **Prompts for $\circ \square \diamond$:** For this method, we first generate the description and then the acronym via a single prompt to the LLM. The prompt we provide is as shown in Figure 8. The prompt consists of a detailed **instruction** explaining what the task is, a 1-shot **demonstration** as an example, and a **query** for acquiring outputs.
- **Prompts for $\bullet \square \diamond$:** The prompt is almost the same as $\bullet \square \diamond$, but since we switch the generation ordering, we modify the instruction and the demonstration part so that the acronym comes first then the description. The prompt for this method is given in Figure 9.
- **Prompts for $\circ \blacksquare \diamond$:** For this strategy, similarly, we choose to generate the description and acronym via a single prompt. The difference is that we explicitly request that the abbreviation be in the character by character. The prompt we provide is as shown in Figure 10. The prompt consists of a detailed **instruction**, a 1-shot **demonstration**, and a **query** for prompting outputs.
- **Prompts for $\circ \square \blacklozenge$:** Unlike the one-stop strategy, the pipeline one prompts the LLM

Strategies	Input	Output
<i>Supervised Fine-Tuning</i>		
○ □ ○	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Title & Abbreviation:	A Robustly Optimized BERT Pretraining Approach: RoBERTa
● □ ○	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Abbreviation & Title:	RoBERTa: A Robustly Optimized BERT Pretraining Approach
○ ■ ○	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Title & Abbreviation:	A Robustly Optimized BERT Pretraining Approach: R o B E R T a
○ □ ● _{1st}	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Title:	A Robustly Optimized BERT Pretraining Approach
○ □ ● _{2nd}	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Title: A Robustly Optimized BERT Pretraining Approach Abbreviation:	RoBERTa
<i>Reinforcement Learning</i>		
○ □ ○	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Title & Abbreviation:	A Robustly Optimized BERT Pretraining Approach: RoBERTa
● □ ○	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Abbreviation & Title:	RoBERTa: A Robustly Optimized BERT Pretraining Approach
○ ■ ○	Abstract: Language model pretraining has led to many key hyperparameters and training data size... Title & Abbreviation:	A Robustly Optimized BERT Pretraining Approach: R o B E R T a

Table 4: The input and output format for each strategy under the Supervised Fine-Tuning and Reinforcement Learning paradigms. ○ means that first generate the description and then the acronym autoregressively. ● means first generate the acronym and then the description in an autoregressive way. □ means to tokenize the acronym into subwords while ■ means to tokenize it character-by-character. ○ means generate the description and the acronym in a one-stop approach. ● adapts a pipeline approach wherein the model first generates a description and then uses the generated description and the input to further generate the acronym with another model. Note that the pipeline strategy (●) involves two models, and the subscript _{1st} and _{2nd} represents the first and second step for the pipeline.

Artifacts	Citation	Link	License
<i>Datasets</i>			
ACL Anthology	N/A	https://aclanthology.org/	CC BY-NC-SA 3.0 License (prior to 2016) CC BY 4.0 License (in or after 2016)
Leipzig Corpora	(Goldhahn et al., 2012)	https://wortschatz.uni-leipzig.de	CC BY-NC 4.0 License
<i>Packages</i>			
PyTorch	(Paszke et al., 2019)	https://pytorch.org/	BSD-3 License
transformers	(Wolf et al., 2020)	https://huggingface.co/docs/transformers/index	Apache License 2.0
Accelerate	(Gugger et al., 2022)	https://huggingface.co/docs/accelerate/index	Apache License 2.0
nlTK	(Bird and Loper, 2004)	https://www.nltk.org/	Apache License 2.0
numpy	(Harris et al., 2020)	https://numpy.org/	BSD License
matplotlib	(Hunter, 2007)	https://matplotlib.org/	BSD compatible License
rouge	(Lin, 2004)	https://github.com/huggingface/evaluate	Apache License 2.0
BERTScore	(Zhang et al., 2020a)	https://pytorch.org/project/bert-score/	MIT License
BeautifulSoup	(Richardson, 2017)	https://beautiful-soup-4.readthedocs.io/en/latest/	MIT License
PyMuPDF	(Richardson, 2017)	https://pymupdf.readthedocs.io/en/latest/	Open-source AGPL
TRL	(von Werra et al., 2020)	http://hf.co/docs/trl	Apache License 2.0
langdetect	(Danilák, 2023)	https://pytorch.org/project/langdetect/	Apache License 2.0
OpenAI API	N/A	https://platform.openai.com/docs/api-reference	MIT License

Table 5: Details of the artifacts we use, including datasets and major packages. The dataset we constructed and the software we provided are under the MIT License. N/A means not applicable.

twice with different inputs. The first one produces the description by prompting with the abstract, shown in Figure 11. Then, the second one generates the acronym by providing prompts including the description from the first LLM and the abstract. One prompt example is displayed in Figure 12. The prompts also follow the **instruction**-**demonstration**-**query** structure.

For all methods, we use gpt-4-1106-preview as the base model for LLM prompting. We have set the temperature to 0.7 and max_tokens to 30, while the other parameters follow the default settings of the OpenAI ChatCompletion model.

G Detailed Experimental Results

G.1 Detailed Results

We list the detailed experimental results in Table 6. Specifically, we present the value of the evaluation metrics. \circ means that first generate descriptions and then acronyms in an autoregressive way. \bullet means first generate acronyms and then descriptions in an autoregressive way. \square means to tokenize the acronym with standard tokenizers like T5Tokenizer or BERTTokenizer or BertTokenizer while \blacksquare means to tokenize it character-by-character. \blacklozenge adapts a pipeline approach that first generates a description and then uses the generated description and the input abstract to further generate acronym. \diamond means generate description and acronym in a one-stop approach. Figure 7 is the visualization of the results in Table 6. From the results, we have four key observations:

- Strategies like generation ordering, tokenization, and framework design have a nuanced impact, with some showing potential for specific improvements but not universally enhancing all aspects of performance. This suggests that optimization of these strategies is highly context-dependent.
- Certain strategies exhibit commonalities across different learning paradigms in specific aspects. For instance, the first-acronym-then-description ($\bullet \square \diamond$) strategy decreases summarization performance for descriptions (Sum-D), the letter-level tokenization strategy ($\circ \blacksquare \diamond$) improves algorithmic metrics, and the pipeline strategy ($\circ \square \blacklozenge$) negatively impacts the summarization aspects (Sum-A and Sum-D), while enhancing neology.
- Overall, strategies demonstrate inconsistency across different learning paradigms, with patterns varying significantly. $\bullet \square \diamond$, for example, enhances summarization for acronyms (Sum-A) and algorithmic aspects in SFT and ICL but has negligible effects on RL. It diminishes neologistic performance in RL but enhances it in SFT and ICL. Similar patterns are observed for summarization and neologistic aspects with $\circ \blacksquare \diamond$, and the algorithmic aspect with $\circ \square \blacklozenge$.
- Strategies in SFT and RL exhibit similar patterns, differing substantially from those in ICL. Except for $\circ \blacksquare \diamond$, which slightly varies in acronym summarization and algorithmic aspects, the patterns of other strategies remain consistent in SFT and RL. The explanation may be that RL models is originated from the SFT model (T5-base) with further training, while ICL uses a different model (GPT-4) with no explicit training on our dataset.
- No single strategy excels across all aspects within each learning paradigm, challenging the identification of an optimal strategy. Almost all strategies, except for $\circ \blacksquare \diamond$ under ICL, reduce performance in description summarization. Although $\circ \blacksquare \diamond$ under ICL benefits the description summarization aspect, it concurrently weakens acronym summarization performance.

G.2 More Case Study

More cases generated by the involved methods are listed in Table 7. First, the ground truth description and acronym satisfy the summarization, neology, and algorithm requirements well. The description, “Unified Cognitive Signal Reconstruction bridging cognitive signals and human language”, accurately summarizes the abstract of the paper (Xi et al., 2023) that investigates cognitive signals and human language. “A Prosody and POS English Lexicon for Language Engineering” encapsulates the main idea of studying prosody and POS from the paper (Brierley and Atwell, 2008). Besides, “UniCoRN” is exactly a common word, and “PROPOSEL” resembles a word, which means they are satisfying in the neologistic aspect. Furthermore, the acronyms’ letters all come from descriptions, meeting the algorithmic requirements well.

Aspect Metrics	Summarization for Description					Neology		Algorithm	Summarization for Acronym	
	BLEU-4	METEOR	ROUGE-L	CIDEr	BERT-Score	WL	WO	LR	C-BLEU-4	G-ROUGE-L
Strategies	Generation of descriptions: D.					Generation of acronyms: A.				
	<i>Supervised Fine-Tuning Paradigm</i>									
○ □ ○	13.47 Baseline	21.29 Baseline	40.60 Baseline	1.50 Baseline	66.95 Baseline	67.73 Baseline	78.18 Baseline	43.92 Baseline	14.91 Baseline	43.27 Baseline
● □ ○	12.82 (-4.83%)	20.66 (-2.96%)	40.37 (-0.57%)	1.48 (-1.33%)	66.75 (-0.30%)	66.81 (-1.36%)	77.82 (-0.46%)	44.78 (+1.96%)	15.33 (+2.82%)	43.52 (+0.58%)
○ □ ○	13.13 (-2.52%)	20.93 (-1.69%)	40.32 (-0.69%)	1.49 (-0.67%)	66.75 (-0.30%)	68.25 (+0.77%)	75.91 (-2.90%)	46.35 (+5.53%)	14.96 (+0.34%)	46.83 (+8.23%)
○ □ ●	12.76 (-5.27%)	20.18 (-5.21%)	38.95 (-4.06%)	1.46 (-2.67%)	66.06 (-1.33%)	69.70 (+2.91%)	80.88 (+3.45%)	44.39 (+1.07%)	12.99 (-12.88%)	41.92 (-3.12%)
	<i>In-Context Learning Paradigm</i>									
○ □ ○	8.84 Baseline	21.36 Baseline	29.37 Baseline	1.10 Baseline	63.88 Baseline	63.83 Baseline	73.81 Baseline	39.71 Baseline	12.12 Baseline	40.36 Baseline
● □ ○	8.10 (-8.37%)	16.53 (-22.61%)	24.62 (-16.17%)	0.92 (-16.36%)	57.33 (-10.25%)	69.81 (+9.37%)	78.65 (+6.56%)	45.94 (+15.69%)	14.50 (+21.95%)	45.87 (+13.65%)
○ □ ○	9.47 (+7.13%)	22.45 (+5.10%)	31.43 (+7.01%)	1.21 (+10.00%)	65.04 (+1.82%)	61.42 (-3.78%)	83.16 (+12.67%)	47.35 (+19.24%)	4.82 (-60.23%)	33.39 (-17.27%)
○ □ ●	8.29 (-6.22%)	21.21 (-0.70%)	31.06 (+5.75%)	0.98 (-10.91%)	63.43 (-0.70%)	66.34 (+3.93%)	74.39 (+0.79%)	37.43 (-5.74%)	7.49 (-38.20%)	39.11 (-3.10%)
	<i>Reinforcement Learning Paradigm</i>									
○ □ ○	13.34 Baseline	21.16 Baseline	40.72 Baseline	1.51 Baseline	66.84 Baseline	68.38 Baseline	79.07 Baseline	44.66 Baseline	15.23 Baseline	43.28 Baseline
● □ ○	12.27 (-8.02%)	20.39 (-3.64%)	40.06 (-1.62%)	1.44 (-4.64%)	66.57 (-0.40%)	67.17 (-1.77%)	78.11 (-1.21%)	44.52 (-0.31%)	14.99 (-1.58%)	43.26 (-0.05%)
○ □ ○	12.75 (-4.42%)	20.64 (-2.46%)	39.86 (-2.11%)	1.46 (-3.31%)	66.58 (-0.39%)	68.34 (-0.06%)	76.43 (-3.34%)	46.79 (+4.77%)	14.85 (-2.50%)	46.96 (+8.50%)
○ □ ●	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A

Table 6: Experimental results of different strategies in learning paradigms including supervised fine-tuning, reinforcement learning, and in-context learning. The **increase(+)/decrease(-)** ratio aside from the value is the improvement/deterioration of the model over the baseline ○ □ ○. N/A means not applicable.

On the other hand, the generated acronyms and descriptions are not satisfactory. Most strategies handle well the summarization aspect across learning paradigms, as their generated descriptions include the key concepts related to “cognitive signal” (the first example) or “prosody” and “POS” (the second example). However, there’s an evident discrepancy in balancing neologistic with algorithmic aspects. For example, “UCBD” and “UCS-BD” (the first example), or “ProSPL” and “IPPE” achieve algorithmic accuracy with their descriptions, but fall short of the neologistic performance, as they do not resemble real words. Conversely, “BLASER” (the second example) stands as a genuine word, yet it falls short of fulfilling algorithmic criteria. Other strategies fail to meet either criterion satisfactorily.

Strategies	Descriptions	Acronyms
<i>Supervised Fine-Tuning Paradigm</i>		
○□◇	Unifying Cognitive Signal Representation for Brain Decoding	UniCSE
●□◇	the Unified Cognitive Signal Reconstruction for Brain Decoding	fMRI2Text
○■◇	The Unified Cognitive Signal Reconstruction for Brain Decoding	UCBD
○□●	The Unified Cognitive Signal Reconstruction for Brain Decoding	CoS-MeR
<i>In-Context Learning Paradigm</i>		
○□◇	fMRI2text	N/A (Fail to generate)
●□◇	Unified Cognitive Signal Reconstruction for Open-Vocabulary Brain Decoding	UCS-BD
○■◇	fMRI2text	UCSR
○□●	A Unified Approach for Open-Vocabulary Decoding of Cognitive Signals into Text	UniCogniD
<i>Reinforcement Learning Paradigm</i>		
○□◇	Unifying Cognitive Signal Representation for Brain Decoding	UniCSE
●□◇	the Unified Cognitive Signal Reconstruction for Brain Decoding	fMRI2Text
○■◇	The Unified Cognitive Signal Reconstruction for Brain Decoding	UCBD
○□●	N/A	N/A
Ground Truth	Unified Cognitive Signal Reconstruction bridging cognitive signals and human language	UniCoRN
<i>Supervised Fine-Tuning Paradigm</i>		
○□◇	Prosody and PoS for Language Engineering	ProSPL
●□◇	Prosody and PoS Annotated Lexicon for Language Engineering	BLCULex
○■◇	Prosody and PoS - English Lexicon for Language Engineering	ProSpeech
○□●	Prosodic and PoS an English Lexicon for Language Engineering	BLASER
<i>In-Context Learning Paradigm</i>		
○□◇	Integrating Prosody and Part-of-Speech Annotations in the Enhanced English Lexicon for Language Engineering	ProSPEL
●□◇	ProPoSLEX	ProPoSLEX
○■◇	Integrated Prosody and Part-of-Speech English Lexicon for Enhanced Language Engineering	IPPE
○□●	Integrating Prosody and Part-of-Speech for Enhanced English Lexicon in Language Engineering Applications	ProSPEL
<i>Reinforcement Learning Paradigm</i>		
○□◇	Prosody and PoS for Language Engineering	ProSPL
●□◇	Prosody and PoS Annotated Lexicon for Language Engineering	BLCULex
○■◇	Prosody and PoS - English Lexicon for Language Engineering	ProSpeech
○□●	N/A	N/A
Ground Truth	A Prosody and POS English Lexicon for Language Engineering	ProPOSEL

Table 7: Case studies of the acronyms and descriptions generated by various strategies within the learning paradigms of supervised fine-tuning, reinforcement learning, and in-context learning. The examples are from (Xi et al., 2023; Brierley and Atwell, 2008).

Instruction

Given an abstract of a paper, generate a title and an abbreviation for it. The title should capture the main idea of the paper. The abbreviation works as a shorthand name for the title. The abbreviation is usually composed of characters from the title. The abbreviation should look like a real word and be easy to remember.

Demonstration

One example is as follows:

Abstract: With the ever-growing size of pretrained models (PMs), fine-tuning them has become more expensive and resource-hungry. As a remedy, low-rank adapters (LoRA) keep the main pretrained weights of the model frozen and just introduce some learnable truncated SVD modules (so-called LoRA blocks) to the model. While LoRA blocks are parameter-efficient, they suffer from two major problems: first, the size of these blocks is fixed and cannot be modified after training (for example, if we need to change the rank of LoRA blocks, then we need to re-train them from scratch); second, optimizing their rank requires an exhaustive search and effort. In this work, we introduce a dynamic low-rank adaptation (<MASKED_ACRONYM>) technique to address these two problems together. Our <MASKED_ACRONYM> method trains LoRA blocks for a range of ranks instead of a single rank by sorting the representation learned by the adapter module at different ranks during training. We evaluate our solution on different natural language understanding (GLUE benchmark) and language generation tasks (E2E, DART and WebNLG) using different pretrained models such as RoBERTa and GPT with different sizes. Our results show that we can train dynamic search-free models with <MASKED_ACRONYM> at least 4 to 7 times (depending to the task) faster than LoRA without significantly compromising performance. Moreover, our models can perform consistently well on a much larger range of ranks compared to LoRA.

Title: Parameter-Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation | Abbreviation: DyLoRA

Query

Given the following abstract, generated a title and an abbreviation for it. The title and abbreviation are separated by |.

Abstract: Prompt learning with immensely large Casual Language Models (CLMs) has been shown promising for attribute-controllable text generation (CTG). However, vanilla prompt tuning tends to imitate training corpus characteristics beyond the control attributes, resulting in a poor generalization ability. Moreover, it is less able to capture the relationship between different attributes, further limiting the control performance. In this paper, we propose a new CTG approach, namely <MASKED_ACRONYM>, which incorporates the attribute knowledge of discriminator to optimize the control prompts, steering a frozen CLM to produce attribute-specific texts. Specifically, the frozen CLM model, capable of producing multitudinous texts, is first used to generate the next-token candidates based on the context, so as to ensure the diversity of tokens to be predicted. Then, we leverage an attribute-discriminator to select desired/undesired tokens from those candidates, providing the inter-attribute knowledge. Finally, we bridge the above two traits by an unlikelihood objective for prompt-tuning. Extensive experimental results show that <MASKED_ACRONYM> can achieve a new state-of-the-art control performance while maintaining an efficient and high-quality text generation, only relying on around 10 virtual tokens.

Figure 8: Prompts for ○ □ ◇ in the in-context learning paradigm.

Instruction

Given an abstract of a paper, generate an abbreviation and a title for it. The abbreviation works as a shorthand name for the title. The title should capture the main idea of the paper. The abbreviation is usually composed of characters from the title. The abbreviation should look like a real word and be easy to remember.

Demonstration

One example is as follows:

Abstract: With the ever-growing size of pretrained models (PMs), fine-tuning them has become more expensive and resource-hungry. As a remedy, low-rank adapters (LoRA) keep the main pretrained weights of the model frozen and just introduce some learnable truncated SVD modules (so-called LoRA blocks) to the model. While LoRA blocks are parameter-efficient, they suffer from two major problems: first, the size of these blocks is fixed and cannot be modified after training (for example, if we need to change the rank of LoRA blocks, then we need to re-train them from scratch); second, optimizing their rank requires an exhaustive search and effort. In this work, we introduce a dynamic low-rank adaptation (<MASKED_ACRONYM>) technique to address these two problems together. Our <MASKED_ACRONYM> method trains LoRA blocks for a range of ranks instead of a single rank by sorting the representation learned by the adapter module at different ranks during training. We evaluate our solution on different natural language understanding (GLUE benchmark) and language generation tasks (E2E, DART and WebNLG) using different pretrained models such as RoBERTa and GPT with different sizes. Our results show that we can train dynamic search-free models with <MASKED_ACRONYM> at least 4 to 7 times (depending to the task) faster than LoRA without significantly compromising performance. Moreover, our models can perform consistently well on a much larger range of ranks compared to LoRA.

Abbreviation: DyLoRA | Title: Parameter-Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation

Query

Given the following abstract, generated an abbreviation and a title for it. The abbreviation and the title are separated by |.

Abstract: Prompt learning with immensely large Casual Language Models (CLMs) has been shown promising for attribute-controllable text generation (CTG). However, vanilla prompt tuning tends to imitate training corpus characteristics beyond the control attributes, resulting in a poor generalization ability. Moreover, it is less able to capture the relationship between different attributes, further limiting the control performance. In this paper, we propose a new CTG approach, namely <MASKED_ACRONYM>, which incorporates the attribute knowledge of discriminator to optimize the control prompts, steering a frozen CLM to produce attribute-specific texts. Specifically, the frozen CLM model, capable of producing multitudinous texts, is first used to generate the next-token candidates based on the context, so as to ensure the diversity of tokens to be predicted. Then, we leverage an attribute-discriminator to select desired/undesired tokens from those candidates, providing the inter-attribute knowledge. Finally, we bridge the above two traits by an unlikelihood objective for prompt-tuning. Extensive experimental results show that <MASKED_ACRONYM> can achieve a new state-of-the-art control performance while maintaining an efficient and high-quality text generation, only relying on around 10 virtual tokens.

Figure 9: Prompts for ● □ ◇ in the in-context learning paradigm.

Instruction

Given an abstract of a paper, generate a title and an abbreviation for it. The title should capture the main idea of the paper. The abbreviation works as a shorthand name for the title. The abbreviation is usually composed of characters from the title. The abbreviation should look like a real word and be easy to remember. Besides, the abbreviation should be given character by character.

Demonstration

One example is as follows:

Abstract: With the ever-growing size of pretrained models (PMs), fine-tuning them has become more expensive and resource-hungry. As a remedy, low-rank adapters (LoRA) keep the main pretrained weights of the model frozen and just introduce some learnable truncated SVD modules (so-called LoRA blocks) to the model. While LoRA blocks are parameter-efficient, they suffer from two major problems: first, the size of these blocks is fixed and cannot be modified after training (for example, if we need to change the rank of LoRA blocks, then we need to re-train them from scratch); second, optimizing their rank requires an exhaustive search and effort. In this work, we introduce a dynamic low-rank adaptation (<MASKED_ACRONYM>) technique to address these two problems together. Our <MASKED_ACRONYM> method trains LoRA blocks for a range of ranks instead of a single rank by sorting the representation learned by the adapter module at different ranks during training. We evaluate our solution on different natural language understanding (GLUE benchmark) and language generation tasks (E2E, DART and WebNLG) using different pretrained models such as RoBERTa and GPT with different sizes. Our results show that we can train dynamic search-free models with <MASKED_ACRONYM> at least 4 to 7 times (depending to the task) faster than LoRA without significantly compromising performance. Moreover, our models can perform consistently well on a much larger range of ranks compared to LoRA.

Title: Parameter-Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation | Abbreviation: D y L o R A

Query

Given the following abstract, generated a title and an abbreviation for it. The abbreviation should be given character by character. The title and abbreviation are separated by |.

Abstract: Prompt learning with immensely large Casual Language Models (CLMs) has been shown promising for attribute-controllable text generation (CTG). However, vanilla prompt tuning tends to imitate training corpus characteristics beyond the control attributes, resulting in a poor generalization ability. Moreover, it is less able to capture the relationship between different attributes, further limiting the control performance. In this paper, we propose a new CTG approach, namely <MASKED_ACRONYM>, which incorporates the attribute knowledge of discriminator to optimize the control prompts, steering a frozen CLM to produce attribute-specific texts. Specifically, the frozen CLM model, capable of producing multitudinous texts, is first used to generate the next-token candidates based on the context, so as to ensure the diversity of tokens to be predicted. Then, we leverage an attribute-discriminator to select desired/undesired tokens from those candidates, providing the inter-attribute knowledge. Finally, we bridge the above two traits by an unlikelihood objective for prompt-tuning. Extensive experimental results show that <MASKED_ACRONYM> can achieve a new state-of-the-art control performance while maintaining an efficient and high-quality text generation, only relying on around 10 virtual tokens.

Figure 10: Prompts for ○ ■ ◇ in the in-context learning paradigm.

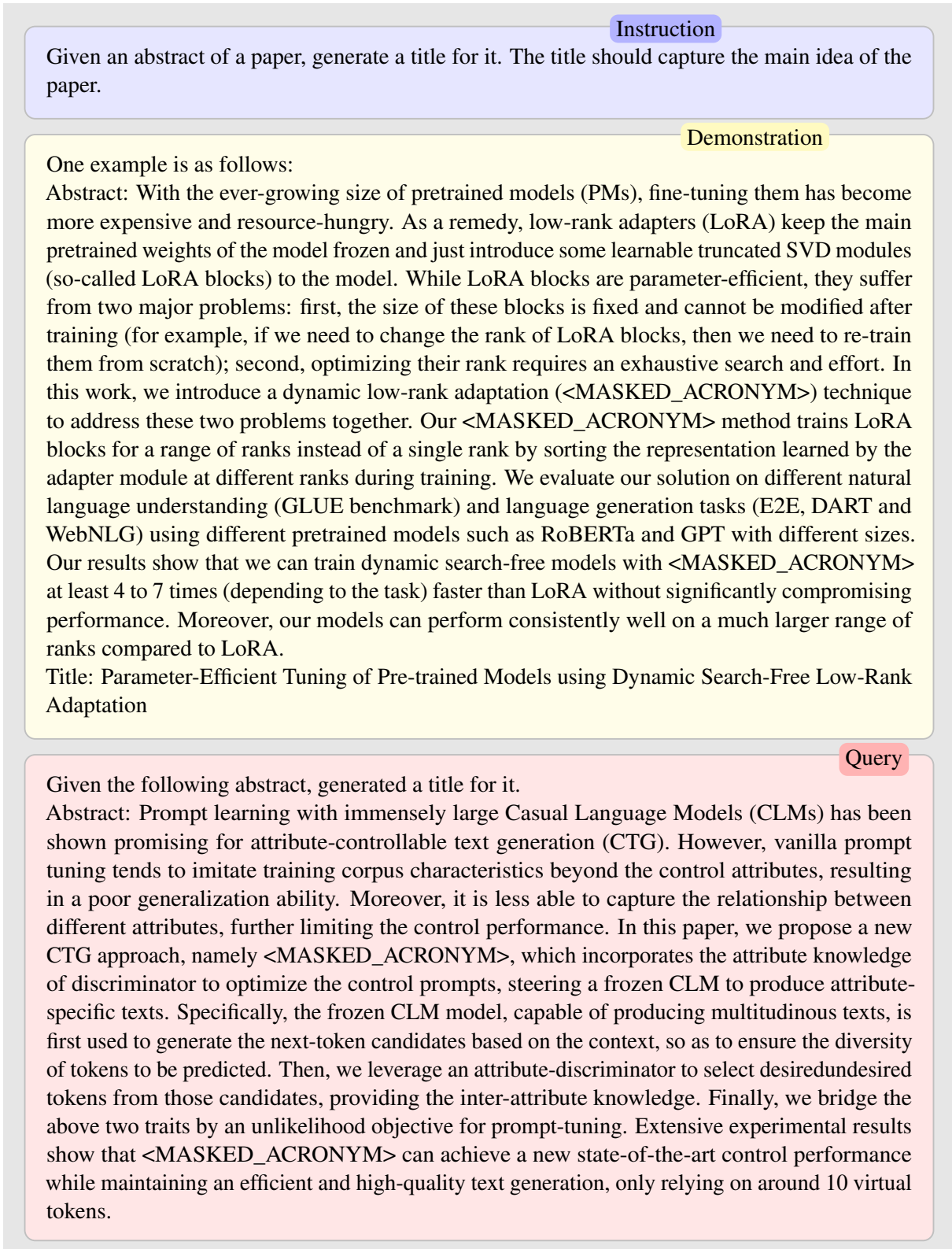


Figure 11: Prompts for generating descriptions in ○ □ ● in the in-context learning setting.

Instruction

Given an abstract of a paper and its title, generate an abbreviation for it. The abbreviation works as a shorthand name for the title. The abbreviation is usually composed of characters from the title. The abbreviation should look like a real word and be easy to remember.

Demonstration

One example is as follows:

Abstract: With the ever-growing size of pretrained models (PMs), fine-tuning them has become more expensive and resource-hungry. As a remedy, low-rank adapters (LoRA) keep the main pretrained weights of the model frozen and just introduce some learnable truncated SVD modules (so-called LoRA blocks) to the model. While LoRA blocks are parameter-efficient, they suffer from two major problems: first, the size of these blocks is fixed and cannot be modified after training (for example, if we need to change the rank of LoRA blocks, then we need to re-train them from scratch); second, optimizing their rank requires an exhaustive search and effort. In this work, we introduce a dynamic low-rank adaptation (<MASKED_ACRONYM>) technique to address these two problems together. Our <MASKED_ACRONYM> method trains LoRA blocks for a range of ranks instead of a single rank by sorting the representation learned by the adapter module at different ranks during training. We evaluate our solution on different natural language understanding (GLUE benchmark) and language generation tasks (E2E, DART and WebNLG) using different pretrained models such as RoBERTa and GPT with different sizes. Our results show that we can train dynamic search-free models with <MASKED_ACRONYM> at least 4 to 7 times (depending to the task) faster than LoRA without significantly compromising performance. Moreover, our models can perform consistently well on a much larger range of ranks compared to LoRA.

Title: Parameter-Efficient Tuning of Pre-trained Models using Dynamic Search-Free Low-Rank Adaptation




Abbreviation: DyLoRA

Query

Given the following abstract and its title, generate an abbreviation for the abstract.

Abstract: Prompt learning with immensely large Casual Language Models (CLMs) has been shown promising for attribute-controllable text generation (CTG). However, vanilla prompt tuning tends to imitate training corpus characteristics beyond the control attributes, resulting in a poor generalization ability. Moreover, it is less able to capture the relationship between different attributes, further limiting the control performance. In this paper, we propose a new CTG approach, namely <MASKED_ACRONYM>, which incorporates the attribute knowledge of discriminator to optimize the control prompts, steering a frozen CLM to produce attribute-specific texts. Specifically, the frozen CLM model, capable of producing multitudinous texts, is first used to generate the next-token candidates based on the context, so as to ensure the diversity of tokens to be predicted. Then, we leverage an attribute-discriminator to select desired/undesired tokens from those candidates, providing the inter-attribute knowledge. Finally, we bridge the above two traits by an unlikelihood objective for prompt-tuning. Extensive experimental results show that <MASKED_ACRONYM> can achieve a new state-of-the-art control performance while maintaining an efficient and high-quality text generation, only relying on around 10 virtual tokens.

Title: Discriminator Cooperative Unlikelihood Prompt-tuning for Controllable Text Generation

Figure 12: Prompts for generating acronyms in    in the in-context learning setting.