Benchmarking and Improving Compositional Generalization of Multi-aspect Controllable Text Generation

Tianqi Zhong[1]*, **Zhaoyi Li**[1,2]*, **Quan Wang**[3], **Linqi Song**[2]
Ying Wei[4], **Defu Lian**[1], **Zhendong Mao**[1]†
[1]University of Science and Technology of China, [2]City University of Hong Kong
[3]Beijing University of Posts and Telecommunications, [4]Nanyang Technological University
{ztq602656097, lizhaoyi777}@mail.ustc.edu.cn
wangquan@bupt.edu.cn, linqi.song@cityu.edu.hk
ying.wei@ntu.edu.sg, {liandefu, zdmao}@ustc.edu.cn

Abstract

Compositional generalization, representing the model's ability to generate text with new attribute combinations obtained by recombining single attributes from the training data, is a crucial property for multi-aspect controllable text generation (MCTG) methods. Nonetheless, a comprehensive compositional generalization evaluation benchmark of MCTG is still lacking. We propose CompMCTG, a benchmark encompassing diverse multi-aspect labeled datasets and a crafted three-dimensional evaluation protocol, to holistically evaluate the compositional generalization of MCTG approaches. We observe that existing MCTG works generally confront a noticeable performance drop in compositional testing. To mitigate this issue, we introduce Meta-MCTG, a training framework incorporating meta-learning, where we enable models to learn how to generalize by simulating compositional generalization scenarios in the training phase. We demonstrate the effectiveness of Meta-MCTG through achieving obvious improvement (by at most 3.64%) for compositional testing performance in 94.4% cases[1].

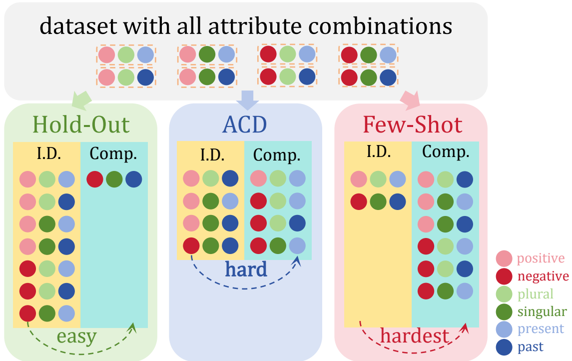

Figure 1: Three evaluation protocols in CompMCTG benchmark, where each set of three colored balls represents texts with these three attribute labels (e.g., positive, plural, and present). "I.D." denotes the *In-Distribution* set and "Comp." denotes the *Compositional* set.

1 Introduction

Multi-aspect Controllable Text Generation aims to generate fluent text with a combination of attributes from diverse aspects (e.g. sentiment, topic, tense, person, and stuff). In comparison with single-aspect controllable text generation (Zhang and Song, 2022), it is more challenging and calls for increasing attention in recent years (Gu et al., 2022; Yang et al., 2023).

Current MCTG methods involve decoding-time-based (Dathathri et al., 2019; Yang and Klein, 2021) that modulate output distribution by a well-trained classifier, separate-training-based (Gu et al., 2022; Huang et al., 2023; Gu et al., 2023; Yang et al., 2023) that train multiple single-aspect modules in turn with single-aspect data and generating multi-aspect text by fusing them, and joint-training-based (Keskar et al., 2019; Qian et al., 2022a; Zeng et al., 2023), which train multiple single-aspect modules simultaneously or multi-aspect modules with multi-aspect data. These methods based on pre-trained language models (Radford et al., 2019) have achieved promising results on this task.

However, seldom works focus on the investigation of compositional generalization, a crucial property of MCTG approaches, which refers to the model's ability to generate text with new attribute combinations obtained by recombining single attributes from the training data. For example, we aim for the model to generate text with the attribute combination (negative, male) after training on data with (positive, male) and (negative, female). Due to the difficulties in collecting data with all possible attribute combinations in most real-world scenarios, the capability for compositional generalization is paramount.

To this end, We propose CompMCTG, a compre-

*The first two authors contributed equally to this work.
†Corresponding author: Zhendong Mao.
[1]The code implementation is available at https://github.com/tqzhong/CG4MCTG.

Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 6486–6517
August 11-16, 2024

hensive benchmark to evaluate the compositional generalization of MCTG approaches (Section 3.1). We first collect four popular datasets (from a minimum of two-aspect, eight attribute combinations to a maximum of four-aspect, forty attribute combinations) in the MCTG field to comprise CompMCTG. The next crucial issue is how to split the dataset to better unveil the compositional generalization risk of MCTG methods. Generally, we split the whole dataset $\mathcal{C}$ into two disjoints sets: in-distribution set $\mathcal{C}_{i.d.}$ and compositional set $\mathcal{C}_{comp}$, where the MCTG model is trained on $\mathcal{C}_{i.d.}$ and tested on both $\mathcal{C}_{i.d.}$ (**in-distribution testing**) and $\mathcal{C}_{comp}$ (**compositional testing**). For an all-sided evaluation, we propose a three-dimensional evaluation protocol containing *Hold-Out*, *ACD*, and *Few-Shot*, which is depicted in Figure 1. Among them, *Hold-Out* is an easy protocol, which holds a few attribute combinations out from $\mathcal{C}$ as $\mathcal{C}_{comp}$ and uses the remaining combinations as $\mathcal{C}_{i.d.}$. *Few-Shot* is the hardest protocol, in which we guarantee every single attribute appears in the $\mathcal{C}_{i.d.}$ while minimizing $|\mathcal{C}_{i.d.}|$[2]. To better reflect the capacity of models in cases that $|\mathcal{C}_{comp}|$ is comparable to $|\mathcal{C}_{i.d.}|$, which are closer to real-world scenarios, we design ***A****ttribute* ***C****ompound* ***D****ivergence* (*ACD*), where we make $|\mathcal{C}_{i.d.}| = |\mathcal{C}_{comp}|$. The core idea of *ACD* is to maximize the distributional divergence between $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$. Compared with random sampling that contributes to similar distributions between $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$ easily (Zeng et al., 2023), *ACD* can better amplify the compositional generalization risk while random-based splits often lead to gross under-estimation (Section 3.4).

Through the results on CompMCTG (Section 3.3), we observe that all of the evaluated MCTG baseline approaches are faced with a noticeable performance drop between in-distribution and compositional testing. To further enhance the compositional generalization performance of joint-training-based methods which generally perform the best among all baselines, we propose Meta-MCTG (Section 4), a training framework incorporating meta-learning (Finn et al., 2017), in which we enable models to learn how to generalize by simulating compositional generalization scenarios in the training phase. Firstly, we train the original model on a training batch $\mathcal{B}_{train}$, perform one step of gradient descent, and save the updated parameters to a backup model without updating the original model's parameters. Secondly, we sample a "pseudo compositional" batch $\mathcal{B}_{pcomp}$ from the training set where the attribute combinations are the re-combination of those in $\mathcal{B}_{train}$ and train the backup model on $\mathcal{B}_{pcomp}$. Finally, we combine the losses from both steps and perform one step of gradient descent to update the original model's parameters. Compared with solely training the model on $\mathcal{B}_{train}$, introducing $\mathcal{B}_{pcomp}$ enables the model's parameters to update in a direction that not only focuses on fitting the training data but also takes out-of-distribution data into account, which helps to elevate model's capability of compositional generalization. We implement Meta-MCTG on three top-performing joint-training-based MCTG baselines and conduct extensive experiments on CompMCTG, demonstrating the effectiveness of Meta-MCTG through achieving obvious improvement (by at most 3.64%) for compositional testing in 94.4% cases.

Our main contributions are three-fold: (1) We propose CompMCTG, the first holistic benchmark targeting compositional generalization for MCTG, incorporating four popular datasets and a crafted three-dimensional evaluation protocol. (2) We conduct extensive experiments on CompMCTG with eight representative MCTG baselines and two additional LLMs, unveiling noticeable compositional generalization risk in them and demonstrating the necessity of designs in CompMCTG. (3) We propose Meta-MCTG, incorporating meta-learning into the MCTG training process, to mitigate MCTG models' over-fitting to attribute combinations seen in the training phase and improve their capacity for compositional generalization. To the best of our knowledge, we are the first to comprehensively evaluate MCTG on compositional generalization and introduce meta-learning into MCTG to improve composition generalization.

2 Related Work

Multi-aspect Controllable Text Generation Existing works on MCTG primarily fall into the following three categories: The first is **decoding-time-based** (Dathathri et al., 2019; Yang and Klein, 2021; Krause et al., 2021), which uses a well-trained classifier or conditional language model to adjust the output probability distribution of a frozen causal language model. The second is **separate-training-based**, which trains single-attribute mod-

[2]We define $|\mathcal{C}|$ as the number of attribute combinations in $\mathcal{C}$

ules (Yang et al., 2023; Huang et al., 2023), Energy-based Models (Mireshghallah et al., 2022; Qin et al., 2022) or latent space representations (Gu et al., 2022, 2023) using single-attribute label data, and controls the generation by concatenating individual modules, Energy-based Models or seeking the intersection of different attribute representations in the latent space. The third is **joint-training-based**, which trains multi-attribute modules (Keskar et al., 2019; Zeng et al., 2023; Qian et al., 2022b) simultaneously using multi-attribute label data. Qian et al. (2022b) add a prefix (Li and Liang, 2021) for each attribute and train these prefixes using a contrastive loss. Zeng et al. (2023) encode different control codes (word embedding of attribute tokens) into prompts (Lester et al., 2021) using a fully connected layer and train this layer using a contrastive loss similar to Qian et al. (2022b).

Compositional Generalization Existing works on compositional generalization involve various NLP topics: Semantic Parsing (Herzig and Berant, 2021; Ontanon et al., 2022; Drozdov et al., 2023; Li et al., 2023), Machine Translation (Li et al., 2021; Zheng and Lapata, 2022; Lin et al., 2023), Text Classification (Kim et al., 2021; Chai et al., 2023), Complex Reasoning (Zhou et al., 2023a; Press et al., 2023; Li et al., 2024; Lin et al., 2024) and stuff. Nonetheless, in the field of open-domain controllable text generation, compositional generalization, which we target and reveal as the necessity for the robustness of neural language generators in this paper, remains under-explored. (Zeng et al., 2023) investigates compositional generalization focusing on a neighboring topic, controllable dialogue generation. We regard their work as a starting point of our research and further depict the deficiency of its naive evaluation protocol, for the underestimation of the compositionality gap in more realistic scenarios (Keysers et al., 2020).

3 Benchmark: CompMCTG

We propose CompMCTG, a novel benchmark to comprehensively evaluate the compositional generalization capacity of MCTG approaches. The superiority and novelty of CompMCTG are out of its scale of dataset and its three-dimensional evaluation protocol (Section 3.1). We select eight representative baseline approaches (Section 3.2), evaluate their performance on our CompMCTG benchmark, and unveil their struggling on compositional testing (Section 3.3). Moreover, systematic analysis towards exploring the behaviors of baseline approaches under different evaluation protocols of CompMCTG is provided in Section 3.4, which highlights: 1) its capacity to dig out the potential generalization risk of evaluated approaches and 2) the undervalued compositionality gap in the previous work (Zeng et al., 2023) as well.

3.1 On the Construction of CompMCTG

Data Source We collect commonly used and open-sourced datasets for our usage. Consequently, we select a shopping review dataset: *Amazon Review* (He and McAuley, 2016), a mixture of movie(IMDB (Maas et al., 2011)), tablet, automobile(Sentube (Uryupina et al., 2014)) and hotel(OpenNER (Agerri et al., 2013)) review dataset: *Mixture* (Liu et al., 2022), and two restaurant review datasets: *YELP* (Shen et al., 2017; YELP, 2014) and *Fyelp* (Lample et al., 2019). Details of these datasets are concluded in Appendix A.

Three-Dimensional evaluation Protocol We design a three-dimensional(*Hold-Out*, *ACD* and *Few-Shot*) evaluation protocol, aiming to sufficiently explore the compositional generalization capacity of existing approaches. Supposing that dataset $\mathcal{D}$[3] contains m distinct aspect sets: $\mathcal{A}_1$, $\mathcal{A}_2$, ..., $\mathcal{A}_m$ and a specific aspect $\mathcal{A}_i$ ($1 \leq i \leq m$) has a_i kinds of different attribute values in its set: $\mathcal{A}_i = \{A_i^1, A_i^2, ..., A_i^{a_i}\}$, we denote the whole attribute combination set as the continued Cartesian product $\mathcal{C} = \mathcal{A}_1 \times \mathcal{A}_2 \times ... \times \mathcal{A}_m = \{(A_i^{t_i})_{1 \leq i \leq m} | 1 \leq t_i \leq a_i\}$. The core of constructing CompMCTG is to **split** the attribute combination set $\mathcal{C}$ into *in-distribution* set $\mathcal{C}_{i.d.}$ and *compositional* set $\mathcal{C}_{comp}$. Basically, $\mathcal{C}_{comp}$ has no intersection with $\mathcal{C}_{i.d.}$ and any attribute combination in $\mathcal{C}_{comp}$ can be derived through recombining single attributes in $\mathcal{C}_{i.d.}$. Hence we have the formal definition of **an eligible split** $s(\mathcal{C}) = \mathcal{C}_{i.d.}, \mathcal{C}_{comp}$ as following:

$$\begin{aligned} &\mathcal{C} = \mathcal{C}_{i.d.} \cup \mathcal{C}_{comp},\ \mathcal{C}_{i.d.} \cap \mathcal{C}_{comp} = \emptyset \\ &\{attribute | \exists c \in \mathcal{C}_{comp}, attribute \in c\} \subseteq \\ &\{attribute | \exists c \in \mathcal{C}_{i.d.}, attribute \in c\} \end{aligned} \tag{1}$$

Hold-Out is an easy evaluation protocol, which holds a few attribute combinations out from $\mathcal{C}$ as

[3] Each datum in $\mathcal{D}$ consists of two components: (c, x), where c denotes the *condition part*, a combination of several attributes of different aspects (e.g., sentiment:"positive", tense:"past", and topic:"basketball") and x denotes the *text part*, a span of text corresponding to these conditions. For brevity, we omit the text part and use the *condition part* to represent the data in this section.

Method	*Original*		*Hold-Out*				*ACD*				*Average*		
	$A_{i.d.}(\uparrow)$	$P_{i.d.}$ $(\downarrow)$	$A_{i.d.}$ $(\uparrow)$	$P_{i.d.}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$	$A_{i.d.}$ $(\uparrow)$	$P_{i.d.}$ $(\downarrow)$	A_{comp} $(\uparrow)$	$P_{comp}(\downarrow)$	A_{avg} $(\uparrow)$	P_{avg} $(\downarrow)$	G_{avg} $(\downarrow)$
LLM+In-context Learning													
LLaMA-2 (Touvron et al., 2023)	61.53%	27.30	62.61%	25.55	40.82%	23.80	62.98%	28.31	42.11%	24.63	54.01%	25.92	33.97%
ChatGPT (OpenAI, 2023)	57.51%	18.03	56.62%	18.29	49.21%	18.49	57.13%	18.27	49.75%	18.22	54.04%	18.26	13.00%
Decoding-Time based													
PPLM (Dathathri et al., 2019)	40.91%	322.59	41.05%	325.09	40.62%	340.76	42.25%	328.07	39.60%	325.74	40.89%	328.45	3.66%
Fudge (Yang and Klein, 2021)	60.12%	178.51	59.35%	179.47	42.10%	252.08	57.17%	175.66	41.49%	223.08	52.05%	201.76	28.25%
Separate-Training based													
Dis-Lens (Gu et al., 2022)	85.46%	123.72	84.84%	95.84	55.58%	104.89	85.54%	90.87	49.52%	112.60	72.19%	105.58	22.30%
Prior (Gu et al., 2023)	73.85%	119.91	73.64%	108.58	49.93%	97.64	78.24%	113.73	50.05%	97.63	65.14%	107.50	34.11%
Joint-Training based													
CTRL (Keskar et al., 2019)	79.10%	54.17	78.89%	51.20	75.09%	51.22	77.83%	51.71	69.96%	51.28	**76.17%**	51.92	**7.46%**
CatPrompt (Yang et al., 2023)	63.91%	74.53	63.95%	73.24	60.32%	69.13	60.53%	98.08	48.25%	68.45	59.39%	76.69	12.98%
Con.Prefix (Qian et al., 2022a)	83.99%	79.29	83.75%	80.49	80.36%	87.19	81.15%	80.71	69.84%	83.90	**79.82%**	82.32	**8.99%**
DCG (Zeng et al., 2023)	79.93%	56.37	79.72%	62.05	76.66%	64.40	78.43%	57.97	67.7%	61.11	**76.49%**	60.38	**8.76%**

Table 1: Averaged overall evaluation results for state-of-the-art baseline approaches on our CompMCTG benchmark (*Hold-Out* testing and *ACD* testing). *A*, *P* and *G* are the abbreviations of accuracy, perplexity, and gap (we explain the meaning of "gap" in Section 3.3.) respectively. Subscript *i.d.* and *comp* refer to in-distribution and compositional generalization performance. Each value in this table is the average (Please find the detailed results for each dataset in Appendix I.3) of testing performances on four component datasets of CompMCTG: Amazon Review (He and McAuley, 2016), Fyelp (Lample et al., 2019), YELP (Shen et al., 2017; YELP, 2014) and Mixture (Liu et al., 2022).

Method	*Few-Shot*			
	$A_{i.d.}(\uparrow)$	$P_{i.d.}$ $(\downarrow)$	A_{comp} $(\uparrow)$	$P_{comp}(\downarrow)$
LLM+In-context Learning				
LLaMA-2 (Touvron et al., 2023)	62.78%	26.08	42.99%	23.90
ChatGPT (OpenAI, 2023)	56.64%	18.62	49.50%	17.71
Decoding-Time based				
PPLM (Dathathri et al., 2019)	43.07%	361.60	40.21%	330.94
Fudge (Yang and Klein, 2021)	58.00%	167.31	40.90%	224.91
Separate-Training based				
Dis-Lens (Gu et al., 2022)	87.81%	95.05	51.47%	116.68
Prior (Gu et al., 2023)	85.19%	118.97	51.71%	104.16
Joint-Training based				
CTRL (Keskar et al., 2019)	77.87%	48.48	65.94%	48.28
CatPrompt (Yang et al., 2023)	62.47%	163.66	46.23%	130.50
Con.Prefix (Qian et al., 2022a)	79.89%	88.34	57.56%	93.31
DCG (Zeng et al., 2023)	78.89%	63.22	59.27%	68.14

Table 2: Averaged overall evaluation results for state-of-the-art baseline approaches on our CompMCTG benchmark (*Few-Shot* testing). Each value in this table is the average of testing performances on four component datasets of CompMCTG: Amazon Review (He and McAuley, 2016), Fyelp (Lample et al., 2019), YELP (Shen et al., 2017; YELP, 2014) and Mixture (Liu et al., 2022).

$\mathcal{C}_{comp}$ and uses the remaining attribute combinations as $\mathcal{C}_{i.d.}$. Supposing $|\mathcal{C}_{comp}|$ equals to k (k is relatively small so that the split is eligible), there are $\binom{|\mathcal{C}|}{k}$ different kinds of splits. In our benchmark, we set $k = 1$, and the final result is the average across $\binom{|\mathcal{C}|}{k}$ scenarios to eliminate bias.

Few-Shot is the hardest evaluation protocol, in which we guarantee every single attribute appears in the $\mathcal{C}_{i.d.}$ while minimizing $|\mathcal{C}_{i.d.}|$, which simulate the scenarios of the low-data regime.

While in most real-world scenarios, $|\mathcal{C}_{comp}|$ is comparable to $|\mathcal{C}_{i.d.}|$. A crucial issue to this situation is how we divide $\mathcal{C}$ into $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$ as the exponential complexity of sweeping over all of the eligible possibilities (We discuss this point in Appendix C). Thus focusing on a representative subset of them is a feasible solution. Inspired by (Keysers et al., 2020), we propose *ACD*, where we keep $|\mathcal{C}_{i.d.}| = |\mathcal{C}_{comp}|$ and construct representative splits by maximizing the ***A****ttribute* ***C****ompound* ***D****ivergence* between $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$. The term *attribute compound* refers to a specific tuple of two attributes: $(A_i^{t_i}, A_j^{t_j}), i \leq j, 1 \leq t_i \leq a_i, 1 \leq t_j \leq a_j$, which characterizes the co-occurrence of two attributes in one attribute combination $c \in \mathcal{C}$. Firstly, we calculate the frequency density of the *attribute compound* $(A_i^{t_i}, A_j^{t_j})$ in the combination sets $\mathcal{C} \in \{\mathcal{C}_{i.d.}, \mathcal{C}_{comp}\}$ and obtain two frequency distributions $(f_{\mathcal{C}_{i.d.}}((A_i^{t_i}, A_j^{t_j})))_{i,j,t_i,t_j}$ and $(f_{\mathcal{C}_{comp}}((A_i^{t_i}, A_j^{t_j})))_{i,j,t_i,t_j}$:

$$
\begin{aligned}
f_{\mathcal{C}}((A_i^{t_i}, A_j^{t_j})) &= \frac{\sum_{c \in \mathcal{C}} \mathbb{I}(A_i^{t_i} \in c \wedge A_j^{t_j} \in c)}{\sum_{c \in \mathcal{C}} \sum_{x \in c, y \in c, x \neq y} 1} \\
&= \frac{2 \sum_{c \in \mathcal{C}} \mathbb{I}(A_i^{t_i} \in c \wedge A_j^{t_j} \in c)}{m(m-1)|\mathcal{C}|}
\end{aligned}
\tag{2}
$$

Then we introduce the Chernoff Coefficient $S(P, Q)$ (Chung et al., 1989) to measure the scale of similarity between two probability distributions P and Q (i.e., $P = (p_1, p_2, ..., p_n)$ and $Q = (q_1, q_2, ..., q_n)$, $S(P, Q) = \sum_{i=1}^{n} p_i^{\alpha} q_i^{1-\alpha} \in [0, 1]$)[4]. Finally, we define the *Attribute Compound Divergence* as $D(P_{i.d.}, P_{comp}) = 1 - S(P_{i.d.}, P_{comp}) \in [0, 1]$ to measure the divergence between $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$, where distribution $P_{i.d.}$

[4] $\alpha \in [0, 1]$ is a hyperparameter to control our tolerance on the difference between P and Q:

and P_{comp} represent $(f_{\mathcal{C}_{i.d.}}((A_i^{t_i}, A_j^{t_j})))_{i,j,t_i,t_j}$ and $(f_{\mathcal{C}_{comp}}((A_i^{t_i}, A_j^{t_j})))_{i,j,t_i,t_j}$, respectively. In the real construction of *ACD* splits, we adopt a greedy-based hill climbing algorithm (Russell and Norvig, 2010)[5] to **sample satisfactory splits which maximize** $D(P_{i.d.}, P_{comp})$.

Note that for *Amazon Review* and *Mixture* datasets, *ACD* protocol degenerates to *Few-Shot* protocol as these datasets only contain two aspects and we can not optimize the attribute compound divergence in that situation.

3.2 Baseline and Evaluation Metric

We select eight representative baseline methods to study: 1) for **Joint-Training based** methods, we choose *CTRL* (Keskar et al., 2019), a classic and powerful baseline, *Contrastive Prefix* (*Con.Prefix*) (Qian et al., 2022a), *CatPrompt* (Yang et al., 2023), and *DCG* (Zeng et al., 2023), a related work targeting on reducing the compositionality gap, as our baseline methods, 2) for **Seperate-Training based**, we select two state-of-the-art baselines: *Distribution-Lens* (Gu et al., 2022) and *Prior* (Gu et al., 2023), 3) for **Decoding-Time based** methods, we choose *PPLM* (Dathathri et al., 2019) and *Fudge* (Yang and Klein, 2021). In addition, we adopt *LLaMA-2* (Touvron et al., 2023) and *ChatGPT* (OpenAI, 2023) to study the compositional generalization of large language models (LLMs) with In-context Learning paradigm (Brown et al., 2020). Following (Sun et al., 2023), we attach five demonstrations in the input prompt for LLMs to follow. One can find more details about our implementations in Appendix D.

Grounded on the MCTG task, we adopt the evaluation metrics (note that the subfixes "*i.d.*" and "*comp*" refer to the in-distribution and compositional testing respectively.) of 1) $ACC_{i.d.}$ and ACC_{comp}: the averaged prediction accuracies[6] for all of the control aspects to measure the **controllability** of generated text, 2) $PPL_{i.d.}$ and PPL_{comp}: perplexity calculated by GPT-2 Large to measure the **fluency** of generated text in all of our experiments, and 3) *Dist-3*: 3-gram distinctness to evaluate the **diversity** of the text generated by approaches mentioned above. We also adopt ***Human-evaluation*** to measure the relevance and fluency of the generated text for each approach[7].

3.3 Evaluation Result

The main evaluation results on CompMCTG benchmark are shown in Table 1, where values in "*Original*" column refer the performance where text data of all attribute combinations are available in the training set and hence there is no compositional testing; values in "*Hold-Out*" and "*ACD*" columns refer to in-distribution and compositional testing performance through the evaluation protocols of "*Hold-Out*" and "*ACD*" mentioned in Section 3.1 respectively; values in "A_{avg}" and "P_{avg}" column refer to overall performance which is the arithmetic mean of results under different evaluation protocols mentioned here (*Original*$_{i.d.}$, *Hold-Out*$_{i.d.}$, *Hold-Out*$_{comp}$, *ACD*$_{i.d.}$ and *ACD*$_{comp}$), which are formulated as:

$$A_{avg} = \frac{1}{5}(A_{i.d.}^{original} + A_{i.d.}^{holdout} + A_{comp}^{holdout} + A_{i.d.}^{acd} + A_{comp}^{acd})$$
$$P_{avg} = \frac{1}{5}(P_{i.d.}^{original} + P_{i.d.}^{holdout} + P_{comp}^{holdout} + P_{i.d.}^{acd} + P_{comp}^{acd}) \quad (3)$$

The "gap" (G_{avg}) is used to assess the average compositional generalization risk and a lower G_{avg} indicates better robustness under compositional testing, which is formulated as:

$$\begin{aligned} G_{avg} &= \frac{1}{2}(G_{holdout} + G_{acd}) \\ &= \frac{1}{2}(\frac{A_{i.d.}^{holdout} - A_{comp}^{holdout}}{A_{i.d.}^{holdout}} + \frac{A_{i.d.}^{acd} - A_{comp}^{acd}}{A_{i.d.}^{acd}}) \end{aligned} \quad (4)$$

Among all the evaluated baselines, **joint-training-based** approaches generally exhibit higher attribute accuracy, better fluency (lower perplexity, only inferior to LLM+ICL), and better robustness to compositional testing (lower G_{avg}). Though seperate-training-based methods perform acceptably in in-distribution testing, their performance drops drastically in compositional testing and we discuss the inherent reason for their failures in Appendix I.1. Decoding-time-based methods perform poorly overall, despite PPLM owning the lowest G_{avg}, both its average accuracy and perplexity are unusable. LLMs can generate more fluent text while the controllability of the generated text (54.04%) falls behind joint-training-based methods (79.82%). At the same time, LLMs (+ICL) also suffer from a large performance drop in compositional testing (G_{avg} is 23.5% for LLaMA and ChatGPT).

[5]The algorithm pseudo-code is available in Appendix H.

[6]For each aspect in each dataset, we train a Roberta classifier (Liu et al., 2019) to evaluate its accuracy (details in Appendix D.3).

[7]Due to the page limit, please find the result of *Dist-3* and *Human-evaluation* in Appendix E and F.

Additionally, We evaluate all of the baseline approaches with *Few-Shot* evaluation protocol in Table 2, to reflect their performance when only limited attribute combinations are available. Again, **joint-training-based** approaches hold the best average performance and compositional generalization capacity among them. We provide the details of our benchmark in Appendix B.

3.4 Insight

In this section, we conduct analysis experiments to show the effect of our key designs in CompMCTG: 1) the three-dimensional evaluation protocol (*Hold-Out*, *ACD* and *Few-Shot*) and 2) the effectiveness of *ACD* in amplifying the compositional generalization gap.

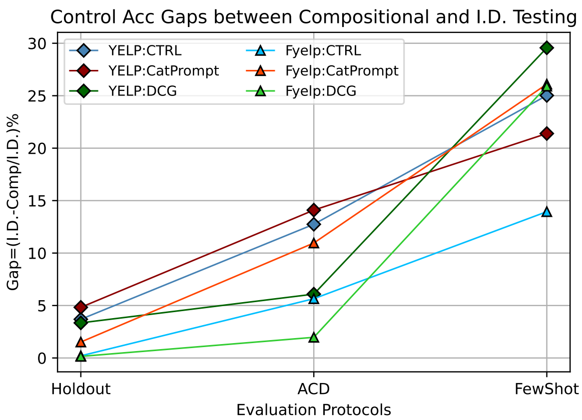

Figure 2: Compositional generalization gap with different evaluation protocols.

Compositional gaps with different evaluation protocols. In Figure 2, we show compositional gaps ($G = \frac{A_{i.d.} - A_{comp}}{A_{i.d.}}$) for approaches: *CTRL*, *CatPrompt* and *DCG*, with three evaluation protocols on *YELP* and *Fyelp* datasets. We observe that the compositional gaps on the same approach and dataset vary a lot with different evaluation protocols: $G_{holdout} < G_{acd} < G_{fewshot}$ generally holds. Notably, *Hold-Out* can not properly unveil the compositional generalization gap for a specific approach. For instance: On *Fyelp* dataset, *CatPrompt* has the compositional gap of 0.91% on *Hold-Out* protocol, while it drastically increases to 10.96% on *ACD* protocol. Moreover, different approaches have different preferences for these protocols. By way of example, The compositional gap (e.g., on *Fyelp*) of *DCG* with *ACD* (1.97%) is lower than *CTRL* (5.95%) while its gap with *Few-Shot* (25.91%) is much higher than *CTRL* (13.95%), demonstrating that the deficiency of *DCG* in low-data regime. Hence jointly leveraging these three evaluation protocols evaluates MCTG approaches more comprehensively.

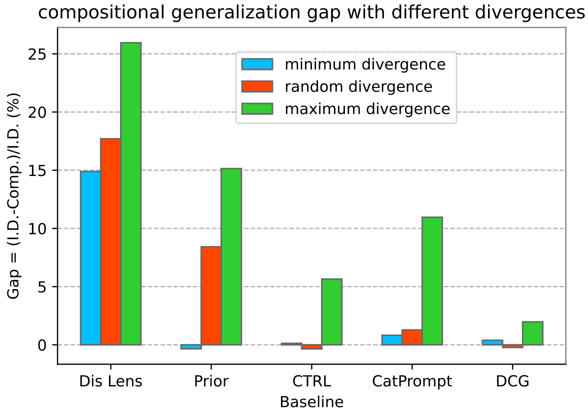

Figure 3: Comparison of compositional gaps between *ACD* (green bars) and two other splitting methods: *Random Sampling* (red bars) and *minimizing the divergence* (blue bars) on five baselines.

Does the ACD better unveil the compositional generalization risk in comparison with Random Sampling? To demonstrate the effectiveness of *ACD*, where we maximize the divergence of *attribute compound distributions* between in-distribution and compositional sets, we design two other protocols in which we still keep $|\mathcal{C}_{i.d.}| = |\mathcal{C}_{comp}|$: *Random Sampling* (random divergence) and *minimizing the divergence* (minimum divergence). We compare the compositional gaps among the three protocols (on *Fyelp* dataset) in Figure 3. We observe that gaps of *ACD* are consistently higher than two comparison protocols by large margins. Notably, using baseline approaches of *CTRL* and *DCG*, compositional gaps with *Random Sampling* are **near zero** while they are 5.65% and 1.97% with *ACD*. Hence we conclude that *ACD* generally better unveils the compositional generalization risk while *Random Sampling* often causes gross under-estimation of such risk.

4 Methodlogy: Meta-MCTG

In Section 3.4, we observe that joint-training-based (both parameter-efficient fine-tuning based and all-parameter fine-tuning based) baselines generally achieve better overall performance. Nonetheless, there still exist non-negligible compositional generalization gaps for all these baselines, which highly calls for our attention. To this end, we propose Meta-MCTG, a novel **Meta**-learning (Finn et al.,

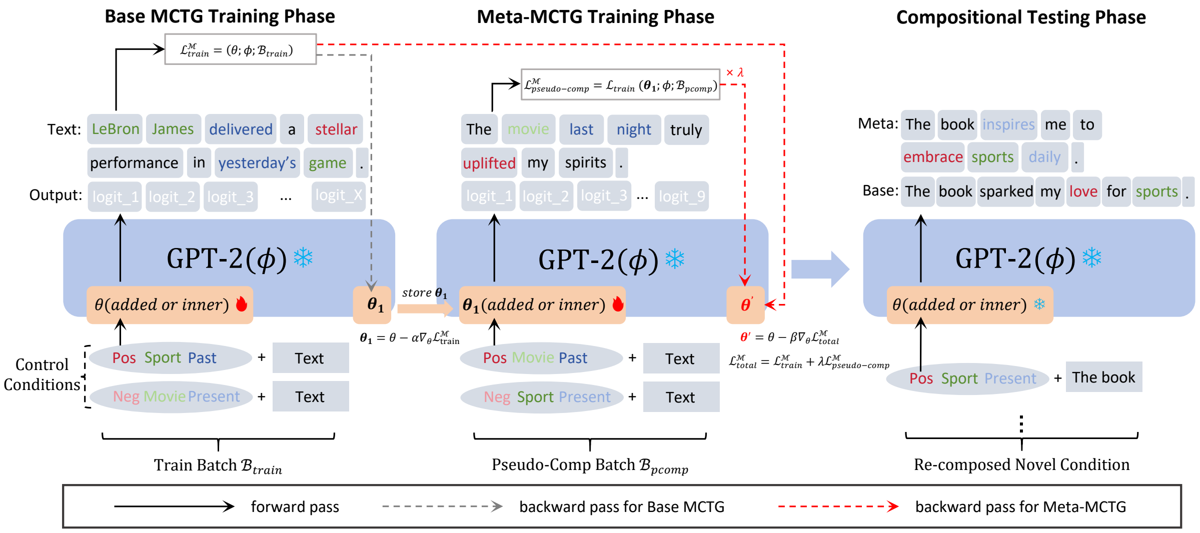

Figure 4: Meta-MCTG: θ refers to the learnable parameters for encoding control conditions, which could be inner (*CTRL*) or added (*DCG* and *ContraPrefix*). ϕ, the parameters of LMs, are usually frozen during training (PEFT).

2017) based **MCTG** training framework, to further improve compositional generalization capabilities of existing joint-training baselines. The framework is easy to implement and can be directly combined with any joint-training-based methods. We discuss the design of Meta-MCTG in Section 4.1 and demonstrate its effectiveness through experiment results for Meta-MCTG in combination with three competitive joint-training baselines (CTRL (Keskar et al., 2019), ContrastivePrefix (Qian et al., 2022a) and DCG (Zeng et al., 2023)) in Section 4.2.

4.1 Design

Overall Motivation The overall framework of Meta-MCTG is depicted in Figure 4. We analyze that the failure of generating text satisfying control conditions in compositional testing can be attributed to the over-fitting of language models to local optima of control conditions in the training set. Thus when trained language models are fed with recomposed attribute combinations as the control conditions in the compositional testing (e.g., In Figure 4, "positive-sport-**present**"), it will potentially encode and distribute those new attribute combinations in the neighbor area of similar ones (e.g., "positive-sport-**past**") that they have seen in the training phase. In this way, previous MCTG approaches fail to generate text that perfectly meets the requirements of all given conditions. As depicted in Figure 4, when given the recomposed attribute combination of "positive-sport-**present**", models may generate text like "The book **sparked** my love for sports.", neglecting the "**present**" condition (As models only sees "positive-sport-**past**" attribute combination in the training phase).

Meta-MCTG training procedure Inspired by previous meta-learning works targeting generalization (Li et al., 2018; Wang et al., 2021; Conklin et al., 2021), we aim to leverage Model-Agnostic Meta Learning (MAML) (Finn et al., 2017) to mitigate the overfitting problem.

First of all, given a specific joint-training-based approach $\mathcal{M}$, we denote its training objective as $\mathcal{L}^{\mathcal{M}}_{train}(\theta; \phi; \mathcal{B})$ where θ represents the learnable parameters of encoding control conditions, ϕ represents the parameters of the language model (e.g., GPT-2), which are frozen during training (Note that in CTRL, ϕ is also updated while it still suits for the Meta-MCTG.), and $\mathcal{B}$ denotes a batch of data. In general, the training objective can be derived as:

$$\min_\theta \mathcal{L}^{\mathcal{M}}_{train}(\theta; \phi; \mathcal{B}) = \min_\theta \sum_{(c_i, x_i)\in\mathcal{B}} [-\log p(x_i|c_i; \theta; \phi)] + \mathcal{L}_{\mathcal{M}}(\theta; \phi; \mathcal{B}) \quad (5)$$

The first term refers to the basic LM loss (Radford et al., 2018) which maximizes the likelihood of generating target text x_i and the second term refers to the auxiliary loss added by baseline $\mathcal{M}$ (e.g., contrastive loss (Qian et al., 2022a)).

In the Meta-MCTG framework, we first sample a batch of training data, denoted as $\mathcal{B}_{train} = (c_i^{train}, x_i^{train})_{i=1}^m$ and a batch of pseudo-comp data, denoted as $\mathcal{B}_{pcomp} = (c_i^{pcomp}, x_i^{pcomp})_{i=1}^m$ where $\{c_i^{train}\}_{i=1}^m \cap \{c_i^{pcomp}\}_{i=1}^m = \emptyset$ and each attribute combination of $\{c_i^{pcomp}\}_{i=1}^m$ must be the recombination of single attributes appearing in the $\{c_i^{train}\}_{i=1}^{m}$. For instance, in Figure 4 the pseudo-comp conditions "positive-movie-past" and "negative-sport-present" are the recombinations of conditions "positive-sport-past" and "negative-movie-present" in the training batch.

We train model on $\mathcal{B}_{train}$ and perform one step of gradient descent to update θ with Objective 5 (α is the learning-rate):

$$\theta_1 = \theta - \alpha \nabla_\theta \mathcal{L}_{train}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{train}) \tag{6}$$

Then we maintain θ unchanged in the original model, temporarily store θ_1 to a backup model, and feed $\mathcal{B}_{pcomp}$ to the backup model to obtain the loss on pseudo-comp data:

$$\begin{aligned} &\mathcal{L}_{pseudo-comp}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{pcomp}) = \mathcal{L}_{train}^{\mathcal{M}}(\theta_1; \phi; \mathcal{B}_{pcomp}) \\ &= \mathcal{L}_{train}^{\mathcal{M}}(\theta - \alpha \nabla_\theta \mathcal{L}_{train}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{train}); \phi; \mathcal{B}_{pcomp}) \end{aligned} \tag{7}$$

According to the construction of $\mathcal{B}_{pcomp}$, we use $\mathcal{L}_{pseudo-comp}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{pcomp})$ to simulate the compositional generalization scenario, evaluating the compositional generalization capacity of model updated by Eq 6. We hope the updated model (with θ_1) performs as well as possible on these pseudo-comp data rather than merely overfitting $\mathcal{B}_{train}$. Taking both the original training Objective 5 and the compositional generalization Objective 7 into consideration, Meta-MCTG is to minimize the following objective:

$$\begin{aligned} &\mathcal{L}_{total}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{train}; \mathcal{B}_{pcomp}) = \\ &\mathcal{L}_{train}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{train}) + \lambda \mathcal{L}_{pseudo-comp}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{pcomp}) \end{aligned} \tag{8}$$

Where λ is a hyper-parameter to make a trade-off between the above two terms. Finally, we perform one step of gradient descent to update θ in the original model with Objective 8:

$$\theta' = \theta - \beta \nabla_\theta \mathcal{L}_{total}^{\mathcal{M}}(\theta; \phi; \mathcal{B}_{train}; \mathcal{B}_{pcomp}) \tag{9}$$

Where β is the learning rate. We summarize the pseudo-code of the Meta-MCTG training procedure in Algorithm 2 in Appendix H.

4.2 Experiment Results and Analysis

Experiment Results of Meta-MCTG We train *CTRL*, *ContrastivePrefix* and *DCG* with the Meta-MCTG algorithm and aim to demonstrate that Meta-MCTG can generally improve their compositional generalization capacity. The compositional testing results for all four datasets are shown in Table 3[8]. For most cases (94.4% of the total), we can observe that baseline approaches trained with Meta-MCTG have an obvious improvement in compositional testing performance on controllability of generated text (i.e., attribute accuracy) over the original versions (by at most 3.64%). Besides, the introduction of the Meta-MCTG framework has almost no impact on text fluency (i.e., perplexity). We additionally show the in-distribution testing results in Table 4, demonstrating that Meta-MCTG nearly has no negative effect on in-distribution testing. Instead, it improves the in-distribution testing over the original baselines on 72.2% cases. We also provide a separate analysis of the compositional generalization gap variations for each dataset and protocol before and after incorporating the Meta-MCTG framework in Table 5, where the gap is calculated by $gap = \frac{A_{i.d.} - A_{comp}}{A_{i.d.}}$. From the results, it can be observed that in the majority of cases, the Meta-MCTG framework is able to reduce the compositional generalization gap.

Visualization and Case Study Previously we hypothesize that Meta-MCTG mitigates the problem that overfitted baseline approaches distribute recomposed novel attribute combinations in the neighbor of in-distribution ones in the representation space. We now calculate the difference in the distance of any two attribute combinations of the original version of baselines and baselines trained with Meta-MCTG. An example result for CTRL is shown in Figure 5. We observe that nearly all

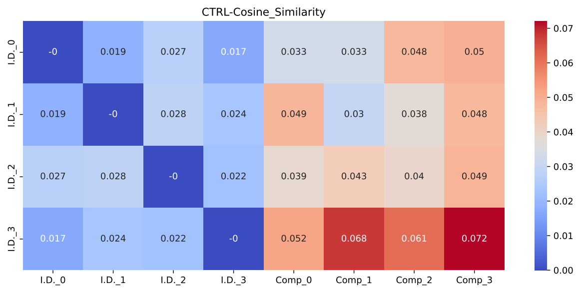

Figure 5: Difference of the distances ($d = 1 - cos < h_1, h_2 >$) between attribute combinations in the representation space (h_1, h_2) with Meta-CTRL and the origin version of CTRL.

of the distances between $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$ increase with Meta-MCTG and are notably larger than the

[8]We do not apply Meta-MCTG to *Few-Shot* settings, for we can not construct $\mathcal{B}_{pseudo-comp}$ when each attribute only appears once in $\mathcal{C}_{i.d.}$.

Method	Fyelp				Amazon		YELP				Mixture	
	Hold-Out		*ACD*		*Hold-Out*		*Hold-Out*		*ACD*		*Hold-Out*	
	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{comp}(\downarrow)$
CTRL (Keskar et al., 2019)	68.29%	45.61	65.31%	45.86	77.89%	37.02	82.02%	73.74	74.63%	75.46	71.82%	47.46
Meta-CTRL **(Ours)**	**68.69%**	46.42	**65.77%**	46.01	**78.78%**	37.30	**83.85%**	68.94	**78.27%**	78.11	**72.83%**	46.20
Con.Prefix (Qian et al., 2022a)	67.50%	52.32	63.93%	49.78	87.58%	44.36	92.79%	132.21	88.84%	128.87	71.91%	138.93
Meta-Con.Prefix **(Ours)**	**67.75%**	52.62	**64.06%**	49.12	**87.69%**	43.89	**94.06%**	130.66	**90.40%**	132.19	**73.11%**	140.53
DCG (Zeng et al., 2023)	66.39%	53.52	64.71%	53.67	84.51%	47.09	80.61%	69.87	75.72%	82.08	76.32%	71.20
Meta-DCG **(Ours)**	66.36%	53.04	**64.84%**	53.58	**85.11%**	47.77	**81.15%**	72.32	**75.88%**	84.58	**79.15%**	65.68

Table 3: Experiment results of *CTRL*, *ContraPrefix*, and *DCG* with Meta-MCTG training in compositional testing.

Method	Fyelp				Amazon		YELP				Mixture	
	Hold-Out		*ACD*		*Hold-Out*		*Hold-Out*		*ACD*		*Hold-Out*	
	$A_{i.d.}(\uparrow)$	$P_{i.d.}(\downarrow)$	$A_{comp}(\uparrow)$	$P_{i.d.}(\downarrow)$	$A_{i.d.}(\uparrow)$	$P_{i.d.}(\downarrow)$	$A_{i.d.}(\uparrow)$	$P_{i.d.}(\downarrow)$	$A_{i.d.}(\uparrow)$	$P_{i.d.}(\downarrow)$	$A_{i.d.}(\uparrow)$	$P_{i.d.}(\downarrow)$
CTRL (Keskar et al., 2019)	69.43%	45.95	69.22%	45.60	80.52%	37.43	85.16%	72.20	85.52%	76.06	80.56%	48.82
Meta-CTRL **(Ours)**	**69.51%**	46.16	**69.45%**	45.50	80.26%	37.31	**85.76%**	69.05	**86.11%**	70.95	80.08%	46.42
Con.Prefix (Qian et al., 2022a)	67.84%	52.48	63.40%	53.11	87.56%	43.97	94.40%	136.04	91.82%	141.15	83.88%	96.46
Meta-Con.Prefix **(Ours)**	**67.90%**	52.40	**64.19%**	52.84	87.43%	43.93	**94.42%**	136.42	**91.86%**	136.39	**84.24%**	97.66
DCG (Zeng et al., 2023)	66.49%	53.50	66.01%	53.29	84.71%	47.20	82.43%	70.28	80.12%	82.96	83.69%	91.80
Meta-DCG **(Ours)**	**66.50%**	53.16	**66.23%**	52.92	**84.78%**	47.55	82.07%	70.01	**80.57%**	82.04	83.50%	83.39

Table 4: Experiment results of *CTRL*, *ContraPrefix* and *DCG* with Meta-MCTG training in in-distribution testing.

Method	Fyelp		Amazon	YELP		Mixture
	Hold-Out	*ACD*	*Hold-Out*	*Hold-Out*	*ACD*	*Hold-Out*
CTRL (Keskar et al., 2019)	1.64%	5.65%	3.27%	3.69%	12.73%	10.85%
Meta-CTRL **(Ours)**	**0.89%**	**5.3%**	**1.84%**	**2.23%**	**9.1%**	**9.05%**
Con.Prefix (Qian et al., 2022a)	0.5%	−0.84%	−0.02%	1.71%	3.25%	14.27%
Meta-Con.Prefix **(Ours)**	**0.22%**	0.2%	**−0.3%**	**0.38%**	**1.59%**	**13.21%**
DCG (Zeng et al., 2023)	0.15%	1.97%	0.24%	2.21%	5.49%	8.81%
Meta-DCG **(Ours)**	0.21%	2.1%	**−0.39%**	**1.12%**	5.82%	**5.21%**

Table 5: Compositional generalization gap of *CTRL*, *ContraPrefix* and *DCG* with Meta-MCTG training.

distances within $\mathcal{C}_{i.d.}$. The results demonstrate that Meta-MCTG can distribute the hidden representations of attribute combinations more sparsely and thus possibly make them more distinguishable. Calculation details and more relevant results are available in Appendix I.2. Besides, we also present **case study** to compare the generation results of the original version of baselines and baselines trained with Meta-MCTG in Appendix G, highlighting the better controllability of the latter ones.

5 Conclusion

We propose CompMCTG, the first holistic benchmark targeting compositional generalization for Multi-Aspect Controllable Text Generation (MCTG), and conduct extensive experiments on CompMCTG with eight representative MCTG baselines and two LLM baselines, unveiling noticeable compositional generalization risk in them and demonstrating the effectiveness of CompMCTG. In addition, we propose Meta-MCTG, a framework incorporating meta-learning into the MCTG training process to improve its compositional generalization ability, which can be combined with any joint-training-based MCTG methods.

Limitations

Our proposed Meta-MCTG framework improves the compositional generalization performance of MCTG methods in most scenarios. However, when attribute combinations of data in the training set are extremely scarce (e.g., the *Few-Shot* protocol in CompMCTG), we cannot build the pseudo-comp batch to utilize the Meta-MCTG framework. Besides, though Meta-MCTG is generally effective, current MCTG methods still have considerable room for improvement in compositional generalization. Both of these limitations will be areas for our future research.

Frankly speaking, the experimental workload of *Hold-Out* protocol in the CompMCTG benchmark is overly cumbersome, and the average results in our main table do not include *Few-Shot*, which we believe are discrepancies. For researchers with limited resources who want to follow our work, we recommend focusing on the performance of models under the *ACD* and *Few-Shot* protocols. These protocols are relatively more challenging and facilitate distinguishing models based on different capabilities.

Ethics Statement

Multi-aspect controllable text generation is widely used in social media. However, improper use can cause serious negative effects, such as using this technology to spread inappropriate remarks (political attributes) or create rumors. Therefore this kind of technology should be subject to certain regulations.

Acknowledgements

We sincerely thank all the anonymous reviewers for their constructive comments. This work is supported by the National Science Fund for Excellent Young Scholars under Grant 62222212, the National Key R&D Program of China (No.2021ZD0111801), the Research Grants Council of the Hong Kong SAR under Grant GRF 11217823 and Collaborative Research Fund C1042-23GF, the National Natural Science Foundation of China under Grant 62371411 and InnoHK initiative, the Government of the HKSAR, Laboratory for AI-Powered Financial Technologies.

References

Rodrigo Agerri, Montse Cuadros, Seán Gaines, and German Rigau. 2013. Opener: Open polarity enhanced named entity recognition. *Procesamiento de Lenguaje Natural*, 51:215–218.

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

Yuyang Chai, Zhuang Li, Jiahui Liu, Lei Chen, Fei Li, Donghong Ji, and Chong Teng. 2023. Compositional generalization for multi-label text classification: A data-augmentation approach.

J.K Chung, P.L Kannappan, C.T Ng, and P.K Sahoo. 1989. Measures of distance between probability distributions. *Journal of Mathematical Analysis and Applications*, 138(1):280–292.

Henry Conklin, Bailin Wang, Kenny Smith, and Ivan Titov. 2021. Meta-learning to compositionally generalize. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3322–3335, Online. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. In *International Conference on Learning Representations*.

Andrew Drozdov, Nathanael Schärli, Ekin Akyürek, Nathan Scales, Xinying Song, Xinyun Chen, Olivier Bousquet, and Denny Zhou. 2023. Compositional semantic parsing with large language models. In *The Eleventh International Conference on Learning Representations*.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104, Copenhagen, Denmark. Association for Computational Linguistics.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Yuxuan Gu, Xiaocheng Feng, Sicheng Ma, Lingyuan Zhang, Heng Gong, and Bing Qin. 2022. A distributional lens for multi-aspect controllable text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 1023–1043, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Yuxuan Gu, Xiaocheng Feng, Sicheng Ma, Lingyuan Zhang, Heng Gong, Weihong Zhong, and Bing Qin. 2023. Controllable text generation via probability density estimation in the latent space. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12590–12616, Toronto, Canada. Association for Computational Linguistics.

Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 507–517, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Jonathan Herzig and Jonathan Berant. 2021. Span-based semantic parsing for compositional generalization. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 908–921, Online. Association for Computational Linguistics.

Xuancheng Huang, Zijun Liu, Peng Li, Tao Li, Maosong Sun, and Yang Liu. 2023. An extensible plug-and-play method for multi-aspect controllable text generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15233–15256, Toronto, Canada. Association for Computational Linguistics.

Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434, Florence, Italy. Association for Computational Linguistics.

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

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. 2020. Measuring compositional generalization: A comprehensive method on realistic data. In *International Conference on Learning Representations*.

Juyong Kim, Pradeep Ravikumar, Joshua Ainslie, and Santiago Ontanon. 2021. Improving compositional generalization in classification tasks via structure annotations. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 637–645, Online. Association for Computational Linguistics.

Ben Krause, Akhilesh Deepak Gotmare, Bryan McCann, Nitish Shirish Keskar, Shafiq Joty, Richard Socher, and Nazneen Fatema Rajani. 2021. Gedi: Generative discriminator guided sequence generation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4929–4952.

Guillaume Lample, Sandeep Subramanian, Eric Smith, Ludovic Denoyer, Marc'Aurelio Ranzato, and Y-Lan Boureau. 2019. Multiple-attribute text rewriting. In *International Conference on Learning Representations*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy Hospedales. 2018. Learning to generalize: Meta-learning for domain generalization. In *AAAI Conference on Artificial Intelligence*.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 110–119, San Diego, California. Association for Computational Linguistics.

Pan Li and Alexander Tuzhilin. 2019. Towards controllable and personalized review generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3237–3245, Hong Kong, China. Association for Computational Linguistics.

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

Yafu Li, Yongjing Yin, Yulong Chen, and Yue Zhang. 2021. On compositional generalization of neural machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4767–4780, Online. Association for Computational Linguistics.

Zhaoyi Li, Gangwei Jiang, Hong Xie, Linqi Song, Defu Lian, and Ying Wei. 2024. Understanding and patching compositional reasoning in llms.

Zhaoyi Li, Ying Wei, and Defu Lian. 2023. Learning to substitute spans towards improving compositional generalization. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2791–2811, Toronto, Canada. Association for Computational Linguistics.

Lei Lin, Jiayi Fu, Pengli Liu, Qingyang Li, Yan Gong, Junchen Wan, Fuzheng Zhang, Zhongyuan Wang, Di Zhang, and Kun Gai. 2024. Ask one more time: Self-agreement improves reasoning of language models in (almost) all scenarios.

Lei Lin, Shuangtao Li, Yafang Zheng, Biao Fu, Shan Liu, Yidong Chen, and Xiaodong Shi. 2023. Learning to compose representations of different encoder layers towards improving compositional generalization.

Guisheng Liu, Yi Li, Yanqing Guo, Xiangyang Luo, and Bo Wang. 2022. Multi-attribute controlled text generation with contrastive-generator and external-discriminator. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5904–5913, Gyeongju, Republic of Korea. International Committee on 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.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA. Association for Computational Linguistics.

Fatemehsadat Mireshghallah, Kartik Goyal, and Taylor Berg-Kirkpatrick. 2022. Mix and match: Learning-free controllable text generationusing energy language models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 401–415, Dublin, Ireland. Association for Computational Linguistics.

Santiago Ontanon, Joshua Ainslie, Zachary Fisher, and Vaclav Cvicek. 2022. Making transformers solve compositional tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3591–3607, Dublin, Ireland. Association for Computational Linguistics.

OpenAI. 2023. ChatGPT — openai.com. `https://openai.com/chatgpt`.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5687–5711, Singapore. Association for Computational Linguistics.

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022a. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, Dublin, Ireland. Association for Computational Linguistics.

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022b. Controllable natural language generation with contrastive prefixes. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2912–2924, Dublin, Ireland. Association for Computational Linguistics.

Lianhui Qin, Sean Welleck, Daniel Khashabi, and Yejin Choi. 2022. Cold decoding: Energy-based constrained text generation with langevin dynamics. In *Advances in Neural Information Processing Systems*, volume 35, pages 9538–9551. Curran Associates, Inc.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Herbert E. Robbins. 1955. A remark on stirling's formula. *American Mathematical Monthly*, 62:402–405.

Stuart Russell and Peter Norvig. 2010. *Artificial Intelligence: A Modern Approach*, 3 edition. Prentice Hall.

Giuseppe Russo, Nora Hollenstein, Claudiu Cristian Musat, and Ce Zhang. 2020. Control, generate, augment: A scalable framework for multi-attribute text generation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 351–366, Online. Association for Computational Linguistics.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Jiao Sun, Yufei Tian, Wangchunshu Zhou, Nan Xu, Qian Hu, Rahul Gupta, John Wieting, Nanyun Peng, and Xuezhe Ma. 2023. Evaluating large language models on controlled generation tasks. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 3155–3168, Singapore. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Olga Uryupina, Barbara Plank, Aliaksei Severyn, Agata Rotondi, and Alessandro Moschitti. 2014. Sentube: A corpus for sentiment analysis on youtube social media. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Bailin Wang, Mirella Lapata, and Ivan Titov. 2021. Meta-learning for domain generalization in semantic parsing. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 366–379, Online. Association for Computational Linguistics.

Kevin Yang and Dan Klein. 2021. FUDGE: Controlled text generation with future discriminators. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, pages 3511–3535, Online. Association for Computational Linguistics.

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

YELP. 2014. Yelp dataset. `https://www.yelp.com/dataset/challenge`.

Weihao Zeng, Lulu Zhao, Keqing He, Ruotong Geng, Jingang Wang, Wei Wu, and Weiran Xu. 2023. Seen to unseen: Exploring compositional generalization of multi-attribute controllable dialogue generation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14179–14196, 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.

Hao Zheng and Mirella Lapata. 2022. Disentangled sequence to sequence learning for compositional generalization. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4256–4268, Dublin, Ireland. Association for Computational Linguistics.

Tianqi Zhong, Quan Wang, Jingxuan Han, Yongdong Zhang, and Zhendong Mao. 2023. Air-decoding: Attribute distribution reconstruction for decoding-time controllable text generation. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc V Le, and Ed H. Chi. 2023a. Least-to-most prompting enables complex reasoning in large language models. In *The Eleventh International Conference on Learning Representations*.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Ethan Wilcox, Ryan Cotterell, and Mrinmaya Sachan. 2023b. Controlled text generation with natural language instructions. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 42602–42613. PMLR.

Appendix

A Datasets

We select a shopping review dataset: *Amazon Review* (He and McAuley, 2016), a mixture of movie(IMDB (Maas et al., 2011)), tablet, automobile(Sentube (Uryupina et al., 2014)) and hotel(OpenNER (Agerri et al., 2013)) review dataset: *Mixture* (Liu et al., 2022), and two restaurant review datasets: *YELP* (Shen et al., 2017; YELP, 2014) and *FYelp* (Lample et al., 2019). In this section, we mainly introduce the four datasets that make up our benchmark as mentioned above.

Fyelp Following previous work (Yang et al., 2023; Huang et al., 2023; Lample et al., 2019), we adopt the widely used *Fyelp* dataset, which contains restaurant reviews with the sentiment (positive and negative), the cuisine (American, Mexican, Asian, Bar, and dessert), and the gender (Male and Female). To evaluate the extensibility of methods, we add one additional aspect of constraints: the tense (Past and Present) (Ficler and Goldberg, 2017), where its label is automatically extracted from the reviews. Thus far, the *Fyelp* dataset is the one with the largest scale of attribute combinations in our benchmark. In total, there are $2 \times 2 \times 5 \times 2 = 40$ possible attribute combinations.

Amazon Review Amazon Review (He and McAuley, 2016) is a dataset containing reviews for Amazon products, which is widely used in previous academic works around text rewriting, controllable text generation, and stuff (Li and Tuzhilin, 2019; Lample et al., 2019; Zhou et al., 2023b). Following (Lample et al., 2019), we process the dataset and label the data with two aspects: the sentiment (positive and negative) and the topic (Books, Clothing, Music, Electronics, Movies and Sports) with the meta-data in the original Amazon Review[9] dataset. Hence there are $2 \times 6 = 12$ different attribute combinations.

YELP YELP business reviews dataset (YELP, 2014) contains the three aspects of attributes: the tense (Past and Present), the sentiment (positive and negative), and the person (singular and plural). We process the dataset in alignment with (John et al., 2019) and (Russo et al., 2020) and randomly re-split the whole dataset for our usage. There are $2 \times 2 \times 2 = 8$ different attribute combinations in this dataset.

Mixture Mixture is the combination of three individual datasets: IMDb (Maas et al., 2011) (movie reviews) OpenNER (Agerri et al., 2013) (hotel reviews) and SenTube (Uryupina et al., 2014) (tablet and automobile reviews), constructed by (Liu et al., 2022). Hence each datum in Mixture has two aspects of attributes: sentiment (positive and negative) and topic (movie, hotel, tablet, and automobile) and there are in total $2 \times 4 = 8$ possible attribute combinations.

We summarize all details and statistics of these datasets in Table 6.

B Details of CompMCTG Benchmark

B.1 Two Types of Testing

Our CompMCTG benchmark contains four datasets: *Fyelp*, *Amazon*, *Yelp*, and *Mixture*. For each dataset, we divide it into two disjoint subsets: in-distribution set and the compositional set. The in-distribution set contains the data that is visible during training, while the compositional set contains the data that is not visible during training. The sets of attribute combinations contained in the in-distribution set and compositional set are defined as $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$, respectively. We first train the model on the in-distribution set and then there are two types of testing. The first type that involves generating text attribute combinations from $\mathcal{C}_{i.d.}$ is referred to as ***in-distribution testing***, which tests the model's learning ability within the distribution of the training data. The second type that involves generating text with attribute combinations from $\mathcal{C}_{comp}$ is referred to as ***compositional testing***, which tests the model's compositional generalization ability beyond the distribution of the training data.

B.2 Datasets Details

For each dataset, the total number of data points N, the number of attribute combinations $|\mathcal{C}|$, and the number of data points per attribute combinations N_i are related as $N = |\mathcal{C}| \times N_i$ (the data points per attribute combination are equal for all datasets).

For the *Hold-Out* protocol, we define the in-distribution set as the subset obtained by removing one attribute combination from the total dataset. Therefore, the size of the in-distribution set for the *Hold-Out* protocol is given by $N \times (|\mathcal{C}| - 1)/|\mathcal{C}|$.

[9] https://jmcauley.ucsd.edu/data/amazon/

Dataset	m	$\|\mathcal{C}\|$	**Classifier**		**Generator**
			Train	Development	Train
Fyelp	4	40	34000	6000	70000
Amazon	2	12	153000	27000	120000
Yelp	3	8	20400	3600	24000
Mixture	2	8	3624	640	4800

Table 6: Information of the datasets in our CompMCTG Benchmark. m is the number of aspects (e.g., sentiment, topic, tense, and stuff); $|\mathcal{C}|$ is the number of attribute combinations. "Classifier" refers to the size of the data used for training the classifier. We split the data into training and development sets at a ratio of 8.5:1.5 based on this. "Generator" refers to the size of the data used for training the generative model. The data for each attribute combination is uniformly distributed across all sub-datasets (i.e., Train and Development of "Classifier" and Train of "Generator").

For the ACD protocol, we designed it such that the ratio between the in-distribution set and the compositional set is 1:1. Therefore, the size of the in-distribution set for the *ACD* protocol is $N/2$.

For the *Few-Shot* protocol, our requirement for the in-distribution set is: 1) Each individual attribute must appear at least once, and 2) The total number of attribute combinations should be minimal. Therefore, for the *Few-Shot* protocol, the number attribute combinations in the in-distribution set is equal to the number of attributes in the aspect with the most attributes. Let's assume that the aspect with the most attributes in the dataset contains M attributes. In this case, the size of the in-distribution set for the *Few-Shot* protocol is $N \times M/|\mathcal{C}|$.

Take the *Fyelp* dataset as an example. The total number of data points for training generator is 70000, and the number of attribute combinations is $2 \times 2 \times 2 \times 5 = 40$. Therefore, $N = 70000$, $|\mathcal{C}| = 40$, $M = 5$. Hence, the size of the in-distribution set for the *Hold-Out* protocol is $70000 \times (40 - 1)/40 = 68250$, for the *ACD* protocol is $70000/2 = 35000$, and for the *Few-Shot* protocol is $70000 \times 5/40 = 8750$. Similarly, we can calculate the corresponding sizes of the in-distribution sets for the other three datasets.

B.3 Why Few-Shot not in Average Results?

In Table 1, the calculation of the Average does not include results from the *Few-Shot* protocol. There are two reasons for this approach: 1) According to the design principles of our *ACD* and *Few-Shot* protocols, the partitioning results for the datasets *Mixture* and *Amazon* are consistent between *ACD* and *Few-Shot*; 2) The difficult level of the *Few-Shot* protocol is relatively high for current models, and we aim to present this category as a direction for future research within the community.

Dataset	*Original*	*Hold-Out*	*ACD*	*Few-Shot*
Fyelp	1	40	10	2
Amazon	1	12	–	10
YELP	1	8	10	8
Mixture	1	8	–	8

Table 7: The number of partitioning methods included in different protocols across four datasets in CompMCTG Benchmark.

B.4 Results in CompMCTG

As previously mentioned, the results in Table 1 and Table 2 represent the average outcomes across four datasets. In fact, for each dataset, the results for each protocol are derived from the average of multiple experiments.

For the *Hold-Out* protocol, we define it as randomly selecting one attribute combination from the complete dataset. In order to eliminate bias during the experiments, we iterate over all attribute combinations, and the final result for each dataset in the *Hold-Out* protocol is the average of all these results.

For the *ACD* protocol, we maximize attribute divergence to partition the datasets. In our experiments, there are usually multiple optimal partitioning methods, hence we also average over all cases for the final results.

Similarly, for the *Few-Shot* protocol, we partition the datasets by maximizing attribute divergence and take the average of all optimal partitioning results.

We present the number of partitioning methods included in different protocols across four datasets in Table 7.

C Complexity discussion

In this section, we discuss the complexity of sweeping over all possibilities for "Half&Half" splitting (i.e.,$|\mathcal{C}_{i.d.}| = |\mathcal{C}_{comp}|$) in Section 3.1. Following the denotations in Section 3.1: m refers to the number of different aspects; $\mathcal{A}_i, (1 \le i \le m)$ is the set of attribute values for the i-th aspect; $\min_{1\le i\le m} |\mathcal{A}_i| = a$; the total number of attribute combinations is $\mathcal{O}(a^m)$.

Sweeping over all possible "Half&Half" splitting methods requires $\mathcal{O}(\binom{a^m}{a^m/2})$ kinds of situations, which can be derived as follows (using Stirling's formula (Robbins, 1955)):

$$\binom{a^m}{a^m/2} = \frac{(a^m)!}{(\frac{a^m}{2})! \cdot (\frac{a^m}{2})!} \approx \frac{\sqrt{2\pi a^m} \cdot (\frac{a^m}{e})^{a^m}}{\pi a^m \cdot (\frac{a^m}{2e})^{a^m}}$$
$$= \frac{\sqrt{2\pi a^m} \cdot 2^{a^m}}{\pi a^m}$$

Hence $\mathcal{O}(\binom{a^m}{a^m/2}) \approx \mathcal{O}(\frac{\sqrt{2\pi a^m} \cdot 2^{a^m}}{\pi a^m}) = \mathcal{O}((2-\eta)^{a^m})$ where $\eta \to 0$. This complexity is exponential to a^m and thus unacceptable, which highly calls for an effective sampling strategy (i.e., *ACD* in Section 3.1).

D Implementation Details

Our implementation is based on Hugging Face Transformer models[10] and we use GPT-2 Medium as our backbone for all baselines (except two LLM baselines). In this section, we provide all the hyperparameters for the baselines and our Meta-MCTG method, as well as the training hyperparameters for the classifiers used for evaluation.

First of all, we unify the settings for all experiments during the generation phase. Following previous work (Gu et al., 2022, 2023), we use the 35 prompts from PPLM (Dathathri et al., 2019) for testing. For all MCTG baselines, we generate 10 texts for each prompt and each attribute combination, each text with a length of 50, and we adopt topk=200, topp=1.0, and temperature=1.0. For two LLM baselines, due to time and financial costs, we generate only one text for each prompt and each attribute combination. All experiments are completed on an NVIDIA A100 (80G) GPU.

D.1 MCTG Baselines

Fudge Fudge (Yang and Klein, 2021) uses a future discriminator to guide the GPT-2 for the generation. Following previous work (Zeng et al., 2023), for each dataset, we train a Multilayer Perceptron (MLP) of dimension $d_{embd} \times m$ as the future discriminator, where d_{embd} is the embedding dimension of GPT-2 Medium, and m is the number of all attribute combinations in the dataset. We set batch size to 8, epoch to 5, and learning rate to 3e-5 in the training phase for all datasets and all protocols. As for the generation, we set control strength α to 20 for all datasets and all settings.

[10] https://github.com/huggingface/transformers

Dataset	*Original*	*Hold-Out*	*ACD*	*Few-Shot*
Fyelp	8000	8000	4000	4000
Amazon	6000	6000	–	4000
YELP	4000	4000	6000	8000
Mixture	10000	10000	–	10000

Table 8: Training steps of different datasets and different protocols in Distributional Lens (Gu et al., 2022).

PPLM PPLM (Dathathri et al., 2019) uses a discriminator to calculate gradient to update the states of a language model and guide the model to generate texts with a certain attribute. We train a Multilayer Perceptron of dimension $d_{embd} \times m$ as the discriminator-like fudge to guide the model. For each dataset and each protocol, we set the batch size to 8, epoch to 5, and learning rate to 3e-5 in the training phase. As for the generation, we followed the hyperparameters in Dathathri et al. (2019). We set γ to 1.5, num-iterations to 3, num-samples to 10, stepsize to 0.03, window-length to 5, fusion-kl-scale to 0.01, and fusion-gm-scale to 0.99.

Distributional Lens During the training phase, we follow all the hyperparameters of the original work (Gu et al., 2022), with the only change made to the number of training steps. We sweep across training steps from {2000,4000,6000, ...,30000} and select the minimum number of steps for convergence as our experimental setup. We summarize it in the Table 8. In the generation phase, for simplicity and fairness, we set all aspect weights to 1, and all other settings are consistent with the original paper.

Prior Proposed by (Gu et al., 2023), this method is based on the model trained in Gu et al. (2022), with the training loss of the Normalizing Flows added for further training. Therefore, during the training phase, we further train based on all models trained by method Gu et al. (2022), with the hyperparameters consistent with the original work and

Dataset	*Original*	*Hold-Out*	*ACD*	*Few-Shot*
Fyelp	30000	30000	30000	30000
Amazon	30000	30000	–	30000
YELP	5000	5000	5000	5000
Mixture	30000	30000	–	30000

Table 9: Training steps of different datasets and different protocols in Prior Control (Gu et al., 2023).

only a change made to the number of training steps. Like experiments in Gu et al. (2022), we sweep across training steps from {5000, 10000, ..., 50000} and select the minimum number of steps for convergence as our experimental setup. We summarize it in the Table 9. In the generation phase, we find that aspect weights setting to 1 for the *Fyelp* dataset do not yield satisfactory results. Therefore, we attempt to adjust the aspect weights on this dataset and finally set weights to [12,4,24,12] corresponding to aspect ["sentiment", "gender", "cuisine", "tense"] and std to 0.1. For the other three datasets, we set weight to 1 for all aspects and set std to 1.

Catprompt As this is a naive method derived from Yang et al. (2023), there is no clear experiment setup for reference. We sweep across prompt length from {10,20,40,60,80,100,120}, selecting the length with the best test results for each attribute as our experimental hyperparameters. The specific results are as follows. For the *Fyelp* dataset, in the non-FewShot protocols, we set prompt length to 120, batch size to 16, epochs to 20, and learning rate to 5e-5, and in the FewShot protocol, we set prompt length to 100, batch size to 16, epochs to 40, and learning rate to 5e-5. For the *Amazon* dataset, we set prompt length to 10, batch size to 16, epochs to 5, and learning rate to 5e-5 for all settings. For the *YELP* dataset, in the non-FewShot protocols, we set prompt length to 20, batch size to 16, epochs to 20, and learning rate to 5e-5, and in the FewShot protocol, we set prompt length to 20, batch size to 16, epochs to 40, and learning rate to 5e-5. For the *Mixture* dataset, we set prompt length to 10, batch size to 16, epochs to 50, and learning rate to 5e-5 for all settings.

DCG Following previous work (Zeng et al., 2023), for all settings across all datasets, prompt length is set to 50 (where attribute prompt length is set to 6 and task prompt length is set to 44), the disentanglement loss weight is set to 0.1, the batch size is set to 8, and the number of Pseudo Combinations is set to 7. For the setting of epochs, we set epochs to 3 for dataset *Fyelp* and *Amazon*, epochs to 8 for dataset *YELP*, and epochs to 7 for dataset *Mixture*. And for all datasets and protocols, we set the learning rate to 7.5e-5.

CTRL Following previous work (Zeng et al., 2023), we concatenate multi-attribute control codes with training datasets to fine-tune the GPT-2. Since we find that *CTRL* is not sensitive to hyperparameters, we set the batch size to 8, epochs to 5, and learning rate to 3e-5, which converges well for all datasets and protocols.

Contrastive Prefix-Tuning Following previous work (Qian et al., 2022a), we set each attribute's prefix length to 10. For the dataset *Fyelp* and *Amazon*, we set the batch size to 8 and epochs to 2 for all protocols. For the dataset *YELP*, we set the batch size to 8 and epochs to 5 for all protocols. For the dataset *Mixture*, we set the batch size to 8 and epochs to 5 for non-FewShot protocols. For the FewShot protocol of the dataset *Mixture*, we set the batch size to 8 and the epoch to 10. And for all datasets and protocols, we set the learning rate to 3e-5.

D.2 LLM Baselines and Prompts

In this section, we introduce the LLMs we use in Section 3.3 and the prompt template we used for In-Context Learning.

Prompt Following (Sun et al., 2023), we use *5-shot* in context learning prompt template to evaluate the compositional generalization capacity of LLMs regarding ICL. Namely, we insert five demonstrations (Input, Output) for each time of controllable generation. Here is our prompt template:

```
\\5-shot in-context-learning
\\prompt template
"Task: write a sentence that meets the
    requirement of input control
    conditions.
Below are some examples (Input, Output)
    for the task:
Input: <attribute combination 1>.
Output: <text 1> # demonstration_1
Input: <attribute combination 2>.
Output: <text 2> # demonstration_2
Input: <attribute combination 3>.
Output: <text 3> # demonstration_3
Input: <attribute combination 4>.
Output: <text 4> # demonstration_4
Input: <attribute combination 5>.
Output: <text 5> # demonstration_5
Input: <testing attribute combination>.
Output: <a head of text>" \\ generation
```

For in-distribution testing, we insert five demonstrations that share the control conditions (in the attribute combinations) with the testing one. For compositional testing, we uniformly sample five demonstrations (of different attribute combinations) from the whole training set.

Another point that is worth noting is that we encode the control conditions in a standard format (e.g., in Yelp we use "cuisine-0" to represent Asian cuisine, "cuisine-1" to represent Mexican cuisine, "gender-0" to represent gender Male, "gender-1" to represent gender Female and so on). The underlying reason is that we aim to test the LLM's ability to understand the relationship between control attributes and target text content, as well as their capacity to generalize to new combinations of previously seen control attributes.

LLM For LLaMA-2 (Touvron et al., 2023), we adopt the version of "LLaMA-2-7B-hf"[11]. Our generation configuration is following the default configuration provided by Meta:

```
\\LLaMA-2-7B generation configuration
GEN_CONFIGS["llama2-7b"]={
 "bos_token_id": 1,
 "do_sample": True,
 "eos_token_id": 2,
 "pad_token_id": 0,
 "temperature": 0.6,
 "max_length": 50,
 "top_p": 0.9,
 "transformers_version": "4.31.0.dev0"
}
```

For ChatGPT (OpenAI, 2023), we use the OpenAI-api[12] and adpot the version of "gpt-3.5-turbo-0613". The default generation configuration is as follows:

```
\\gpt-3.5 generation configuration
GEN_CONFIGS["gpt-3.5-turbo-0613"]={
 "temperature": 1.0,
 "max_length": 50,
 "top_p": 0.9,
 "openai_version": "0.28.0"
}
```

Cost For the evaluation of LLaMA-2-7B, we do experiments on a NVIDIA A100 GPU for around 60 hours. For the evaluation of ChatGPT, we spend around 3.5e7 tokens in total, costing 70 dollars.

[11]https://huggingface.co/meta-llama/Llama-2-7b-hf

[12]https://openai.com/blog/openai-api

Dataset	Aspect	Batch	Epochs	Accuracy
Fyelp	*Sentiment*	512	5	98.68%
	Gender	512	3	70.68%
	Cuisine	64	4	77.97%
	Tense	32	4	88.57%
Amazon	*Sentiment*	128	5	98.41%
	Topic	64	5	92.84%
YELP	*Sentiment*	1024	5	97.11%
	Person	32	8	99.42%
	Tense	256	3	99.78%
Mixture	*Sentiment*	128	4	84.37%
	Topic	512	8	98.59%

Table 10: The specific configuration and the performance of the classifiers used in our benchmark.

D.3 Classifiers

To avoid the impact of domain differences among different datasets on the accuracy of the classifier, we train a classifier using Roberta-Large (Liu et al., 2019) for each aspect of each dataset. We sweep over batch sizes from {4,8,16,32,64,128,256,512,1024} and epochs from {1,2,3,4,5,6,7,8,9,10}, choosing the settings that yield the highest accuracy on the test set as our experimental configuration. The specific configuration results and the performance of the classifiers on the test set for all datasets and all attribute aspects are shown in Table 10.

D.4 Meta-MCTG

In the experiments of Meta-MCTG, we select the three best-performing joint-training-based methods from the baselines, namely *CTRL* (Keskar et al., 2019) , *DCG* (Zeng et al., 2023), and *Contrastive Prefix* (Qian et al., 2022b). For different datasets and protocols in our benchmark, we search λ from {0.01,0.05,0.1,0.2} based on the original experimental hyperparameters, and further refine the value of λ based on the results. For the majority of cases, we opt for λ to be 0.01. For the learning rate β in all MCTG experiments, we set β to be the same as the learning rate α of each baseline.

E Evaluation on diversity

Following previous work (Li et al., 2016), we use distinctness to measure the generated text's diversity. For each text, we calculate 3-grams named Dist-3 to evaluate distinctness. We choose to conduct diversity evaluation on the data under the three protocols of *Original*, *Hold-Out*, and *ACD*. The whole results are shown in Table 11.

Method	*Original* $Dist\text{-}3_{i.d.}$ (↑)	*Hold-Out* $Dist\text{-}3_{i.d.}$	$Dist\text{-}3_{comp}$	*ACD* $Dist\text{-}3_{i.d.}$	$Dist\text{-}3_{comp}$	*Average* $Dist\text{-}3_{avg}$
LLM+In-context Learning						
LLaMA-2 (Touvron et al., 2023)	0.587	0.430	0.577	0.456	0.451	0.500
ChatGPT (OpenAI, 2023)	0.611	0.408	0.660	0.451	0.457	0.517
Decoding-Time based						
Fudge (Yang and Klein, 2021)	0.656	0.652	0.621	0.625	0.587	0.628
PPLM (Dathathri et al., 2019)	0.697	0.622	0.694	0.621	0.617	0.650
Separate-Training based						
Dis-Lens (Gu et al., 2022)	0.473	0.466	0.462	0.454	0.427	0.456
Prior (Gu et al., 2023)	0.573	0.547	0.548	0.539	0.540	0.549
Joint-Training based						
CTRL (Keskar et al., 2019)	0.625	0.623	0.634	0.616	0.622	0.624
CatPrompt (Yang et al., 2023)	0.642	0.636	0.656	0.677	0.688	0.660
Con.Prefix (Qian et al., 2022b)	0.701	0.696	0.727	0.682	0.717	0.705
DCG (Zeng et al., 2023)	0.677	0.694	0.716	0.675	0.695	0.691

Table 11: Averaged overall evaluation results of **diversity** for state-of-the-art baseline approaches on our CompM-CTG benchmark (*Hold-Out* testing and *ACD testing*). Subscript *i.d.* and *comp* refer to in-distribution and compositional generalization performance.

F Human Evaluation

Following previous work (Zhang and Song, 2022; Zhong et al., 2023), we evaluate generated texts from two aspects: **Relevance** (**R**) which reflects the degree of achievement for the desired control attribute combination and **Fluency** (**F**) which evaluates the text's fluency. Unlike automated evaluation, where the accuracy of individual attributes is measured and averaged, human evaluation directly scores the satisfaction of the given control condition (attribute combination). For each dataset and baseline in each protocol (*Original, HoldOut, and ACD*), we randomly sample 10 texts (for *HoldOut* and *ACD*, we sample 10 texts from in-distribution result and 10 texts from compositional result) and employ three annotators to score them on the two metrics on a scale from 1 (very bad) to 5 (very good). Finally, we calculate the average of these scores and get the final result shown in Table 12. We can find that the results of human evaluation are consistent with the results of automated evaluation.

F.1 Specific Scoring Guidelines

In this subsection, we provide specific scoring guidelines for each human evaluation metric.

Relevance

- 5: The generated texts are perfectly aligned with the desired attribute combination.
- 4: The generated texts are very related to the desired attribute combination.
- 3: The generated texts are related to the desired attribute combination. At most one attribute does not match.
- 2: The generated texts are less related to the desired attribute combination. At most two attributes do not match.
- 1: The generated texts are not aligned with the desired attribute combination. None of the attributes meet the requirements.

Fluency

- 5: The generated texts are grammatically correct, fluent, and easy to understand.
- 4: The generated texts are grammatically correct, but slightly less smooth, yet still easily understandable.
- 3: The generated texts have a few grammar errors, but do not hinder understanding.
- 2: The generated texts have a few grammar errors and are not very easy to understand.
- 1: The generated texts have many grammar errors, lack coherence, and are difficult to understand.

Method	*Original*		*Hold-Out*				*ACD*				*Average*	
	$R_{i.d.}(\uparrow)$	$F_{i.d.}(\uparrow)$	$R_{i.d.}(\uparrow)$	$F_{i.d.}(\uparrow)$	$R_{comp}(\uparrow)$	$F_{comp}(\uparrow)$	$R_{i.d.}(\uparrow)$	$F_{i.d.}(\uparrow)$	$R_{comp}(\uparrow)$	$F_{comp}(\uparrow)$	$R_{avg}(\uparrow)$	$F_{avg}(\uparrow)$
LLM+In-Context Learning												
LLaMA-2 (Touvron et al., 2023)	3.12	4.56	3.23	4.48	2.37	4.43	3.31	4.60	2.22	4.59	2.85	4.53
ChatGPT (OpenAI, 2023)	2.89	4.78	2.86	4.75	2.47	4.81	2.75	4.88	2.57	4.74	2.71	**4.79**
Decoding-Time based												
PPLM (Dathathri et al., 2019)	2.07	1.12	2.22	1.07	2.01	1.09	2.16	1.14	1.82	1.03	2.06	1.09
Fudge (Yang and Klein, 2021)	2.88	2.35	2.68	2.13	2.07	1.87	2.59	1.90	1.97	2.24	2.44	2.10
Separate-Training based												
Dis-Lens (Gu et al., 2022)	4.24	2.86	4.10	3.12	2.55	3.01	4.44	3.21	2.42	2.91	3.55	3.02
Prior (Gu et al., 2023)	3.67	2.96	3.53	3.04	2.42	3.20	3.78	3.03	2.39	3.24	3.16	3.09
Joint-Training based												
CTRL (Keskar et al., 2019)	3.98	3.87	3.78	3.92	3.75	3.94	3.80	3.81	3.55	3.84	3.77	3.88
CatPrompt (Yang et al., 2023)	3.23	3.52	3.27	3.49	3.04	3.58	3.01	3.07	2.45	3.61	3.00	3.45
Con.Prefix (Qian et al., 2022a)	4.22	3.44	4.19	3.40	4.01	3.13	4.15	3.23	3.52	3.12	**4.02**	3.26
DCG (Zeng et al., 2023)	3.92	3.80	3.90	3.68	3.84	3.64	3.88	3.83	3.39	3.73	3.79	3.74

Table 12: Averaged overall **human evaluation** results for state-of-the-art baseline approaches on our CompMCTG benchmark (*Hold-Out* testing and *ACD* testing). "R" refers to metric "Relevance" and "F" refers to metric "Fluency". Subscript *i.d.* and *comp* refer to in-distribution and compositional generalization performance.

Method	*Original*		*Hold-Out*				*ACD*			
	$R_{i.d.}(\uparrow)$	$F_{i.d.}(\uparrow)$	$R_{i.d.}(\uparrow)$	$F_{i.d.}(\uparrow)$	$R_{comp}(\uparrow)$	$F_{comp}(\uparrow)$	$R_{i.d.}(\uparrow)$	$F_{i.d.}(\uparrow)$	$R_{comp}(\uparrow)$	$F_{comp}(\uparrow)$
LLM+In-context Learning										
LLaMA-2 (Touvron et al., 2023)	0.823	0.805	0.834	0.816	0.840	0.809	0.825	0.833	0.836	0.824
ChatGPT (OpenAI, 2023)	0.811	0.814	0.805	0.843	0.827	0.840	0.829	0.860	0.851	0.837
Decoding-Time based										
PPLM (Dathathri et al., 2019)	0.910	0.908	0.887	0.893	0.828	0.839	0.834	0.890	0.887	0.836
Fudge (Yang and Klein, 2021)	0.845	0.814	0.838	0.829	0.845	0.789	0.830	0.892	0.846	0.837
Separate-Training based										
Dis-Lens (Gu et al., 2022)	0.923	0.898	0.914	0.887	0.791	0.867	0.910	0.879	0.801	0.882
Prior (Gu et al., 2023)	0.858	0.838	0.835	0.846	0.837	0.821	0.845	0.883	0.826	0.818
Joint-Training based										
CTRL (Keskar et al., 2019)	0.830	0.808	0.845	0.794	0.815	0.829	0.810	0.822	0.816	0.815
CatPrompt (Yang et al., 2023)	0.782	0.804	0.793	0.811	0.824	0.815	0.806	0.785	0.823	0.836
Con.Prefix (Qian et al., 2022a)	0.898	0.843	0.904	0.826	0.876	0.837	0.879	0.841	0.844	0.820
DCG (Zeng et al., 2023)	0.857	0.886	0.854	0.874	0.818	0.825	0.857	0.867	0.834	0.826

Table 13: Averaged overall **Fleiss'Kappa coefficient** of human evaluation results for state-of-the-art baseline approaches on our CompMCTG benchmark (*Hold-Out* testing and *ACD* testing). "R" refers to the Kappa coefficient of metric "Relevance" and "F" refers to the Kappa coefficient of metric "Fluency". Subscript *i.d.* and *comp* refer to in-distribution and compositional generalization performance.

F.2 Inter-Annotator Agreement Score

We also use **Fleiss'Kappa coefficient** (Fleiss, 1971) to measure the inter-annotator agreement score for each human evaluation metric. The result is shown in Table 13.

G Case Study

In this section, we show some specific generation examples, primarily to compare the difference in generation results before and after using the Meta-MCTG framework. Cases in this section are from the compositional result of *ACD* protocol of dataset *Fyelp*. The specific results are shown in Table 14.

H Algorithm Pseudo-Code

We conclude the pseudo-code of constructing ACD splits in Algorithm 1 and the pseudo-code of Meta-MCTG training in Algorithm 2.

Following the denotations in Section 3.1: m refers to the number of different aspects; $\mathcal{A}_i, (1 \le i \le m)$ is the set of attribute values for the i-th aspect; $\min_{1 \le i \le m} |\mathcal{A}_i| = a$; the total number of attribute combinations is $\mathcal{O}(a^m)$. The time complexity of Algorithm 1 (Greedily constructing ACD splits) is $\mathcal{O}(T_1 \cdot T_2 \cdot a^m)$ (linearly increasing with a^m) which is much better than $\mathcal{O}((2-\epsilon)^{a^m}), (\epsilon \leftarrow 0)$ (exponentially increasing with a^m) in Appendix C.

I Additional Results

I.1 Why do Separate-Training-based methods perform badly in compositional testing?

In this section, we briefly discuss the reasons why the seperate-training-based MCTG methods fail in compositional testing. We take *Dis-Lens* (Gu et al., 2022) as an example to illustrate. This type

Algorithm 1 Constructing ACD splits

Require: Attribute combination set $\mathcal{C}$.
Require: Divergence function $D(\cdot,\cdot)$.
Require: Maximum step T_1, T_2, maximum divergence threshold $\eta \in (0,1)$.

Initialization: current step $t_1 = 0$; maximum divergence $d_m = 0$.
A set of ACD splits *result* = $\emptyset$.
while $t_1 < T_1$ **do**
  $t_1 = t_1 + 1$
  Randomly split $\mathcal{C}$ into $\mathcal{C}_{i.d.}$ and $\mathcal{C}_{comp}$ where $|\mathcal{C}_{i.d.}| = |\mathcal{C}_{comp}|$.
  $t_2 = 0$
  Compute current divergence d: $d = D(\mathcal{C}_{i.d.}, \mathcal{C}_{comp})$.
  Update maximum divergence: $d_m = d$.
  while $t_2 < T_2$ **do**
    $t_2 = t_2 + 1$
    $c_1 = None$.
    for $c \in \mathcal{C}_{i.d.}$ **do**
      if $d_m < D(\mathcal{C}_{i.d.} - \{c\}, \mathcal{C}_{comp} + \{c\})$ **then**
        $c_1 = c$.
        $d_m = D(\mathcal{C}_{i.d.} - \{c\}, \mathcal{C}_{comp} + \{c\})$.
        break
      end if
    end for
    if $c_1 == None$ **then**
      continue
    end if
    $\mathcal{C}_{i.d.} = \mathcal{C}_{i.d.} - \{c_1\}$.
    $\mathcal{C}_{comp} = \mathcal{C}_{comp} + \{c_1\}$.
    for $c \in \mathcal{C}_{comp}$ **do**
      if $d_m < D(\mathcal{C}_{i.d.} + \{c\}, \mathcal{C}_{comp} - \{c\})$ **then**
        $d_m = D(\mathcal{C}_{i.d.} + \{c\}, \mathcal{C}_{comp} - \{c\})$.
        $\mathcal{C}_{i.d.} = \mathcal{C}_{i.d.} + \{c_1\}$.
        $\mathcal{C}_{comp} = \mathcal{C}_{comp} - \{c_1\}$.
        break
      end if
    end for
  end while
  for $d_m \geq \eta$ **do**
    Add ($\mathcal{C}_{i.d.}$,$\mathcal{C}_{comp}$) into *result*.
  end for
end while
return *result*

Algorithm 2 Meta-MCTG

Require: Training set $\mathcal{D}_{train}$
Require: Base Method $\mathcal{M}$
Require: Learning rate α, β, batch size m

while not done **do**
  Sample m data as the training batch $\mathcal{B}_{train} = (c_i^{train}, x_i^{train})_{i=1}^m$ from $\mathcal{D}_{train}$.
  Construct pseudo-compositional batch $\mathcal{B}_{pcomp} = (c_i^{pcomp}, x_i^{comp})_{i=1}^m$ by sampling another m data from $\mathcal{D}_{train}$, where $\{c_i^{train}\}_{i=1}^m \cap \{c_i^{pcomp}\}_{i=1}^m = \emptyset$ while each single attribute condition in $\mathcal{B}_{pseudo-comp}$ must appear in the $\mathcal{B}_{train}$.
  Compute training loss $\mathcal{L}_{train}^{\mathcal{M}}$ through Objective 5.
  Compute θ_1 through Equation 6. (while not really update θ to θ_1)
  Temporarily use θ_1 in the language model.
  Compute pseduo compositional generalization loss $\mathcal{L}_{p-comp}^{\mathcal{M}}$ through Objective 7.
  Compute total loss $\mathcal{L}_{total}^{\mathcal{M}}$ through Objective 8.
  Update θ to θ' through Equation 9
end while

of method encodes each single attribute data into a latent vector space, and then constructs the intersection of different attribute latent vector areas through loss function constraints, and finally guides GPT-2 to generate multi-aspect text by searching for the intersection of different attribute spaces. The essential reason why this method can work is that the training dataset itself has multiple attributes. For example, the data corresponding to the latent space intersection constructed with positive emotion data and sports theme data actually has these two attributes. Therefore, when using a multi-attribute dataset to train the latent vector space, the attribute combinations corresponding to the constrained intersection space are the attribute combinations contained in the training set, and will not produce attribute combinations that do not exist in the training set.

Specifically, we use a *Few-Shot* split of the dataset *Mixture* to conduct experiments, reducing the dimensionality of hidden vectors to a two-dimensional plane through PCA and performing visualization processing. There are four attribute combinations in the training set which are "Negative-movies", "Negative-opener", "Negative-tablets", and "Positive-auto". Figure 6, 7 depict

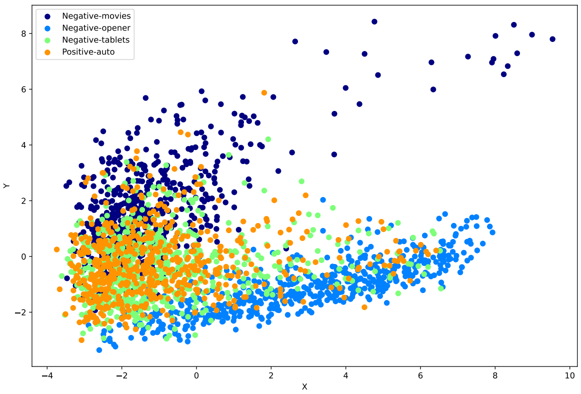

Figure 6: Visualization of *Dis-lens* in *Mixture* dataset before training with multi-aspect label.

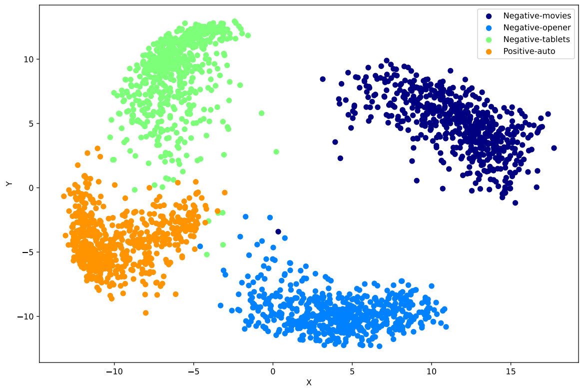

Figure 8: Visualization of *Dis-lens* in *Mixture* dataset after training with multi-aspect label.

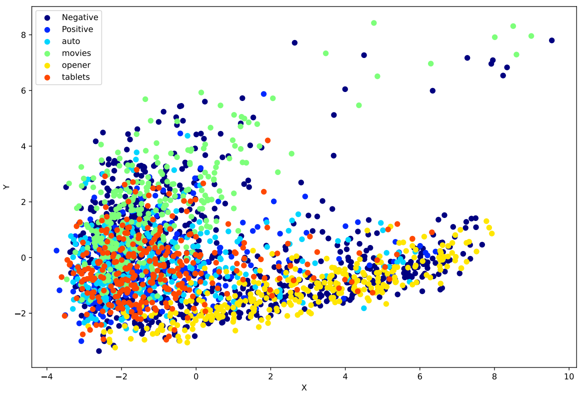

Figure 7: Visualization of *Dis-lens* in *Mixture* dataset before training with single-aspect label.

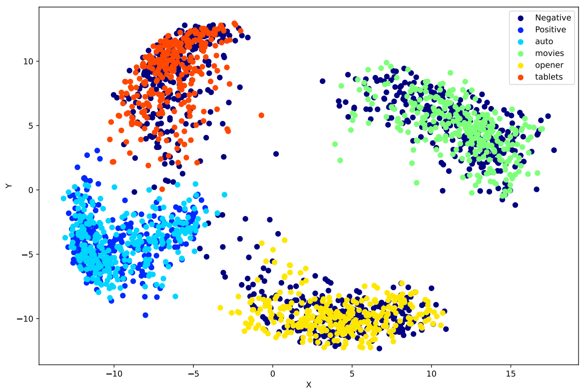

Figure 9: Visualization of *Dis-lens* in *Mixture* dataset after training with single-aspect label.

the results of pre-training visualizations, while Figure 8, 9 show the results of post-training's counterpart. Figure 6, 8 are annotated with multi-aspect labels, whereas Figure 7, 9 are annotated with single-aspect labels. From these four figures, we can find that after training, the hidden vector spaces corresponding to different single attributes have converged, and the intersection of four multi-attribute latent vector spaces has been formed. However, through Figure 8, it can be found that these four intersections exactly correspond to the four attribute combinations contained in the training set, and the intersection of the latent vector spaces of the four compositional attribute combinations ("Negative-auto", "Positive-movies", "Positive-opener", and "Positive-tablets") in Figure 9 basically does not exist. This explains why such methods fail in compositional testing.

I.2 Analysis Experiments

In this section, we conduct visualization experiments on the Meta-MCTG framework we proposed, indirectly verifying its effectiveness. Considering that the joint-training-based MCTG methods tend to overfit the control parameters to the *in-distribution* (I.D.) attribute combinations, this implies that for *compositional* (Comp.) attribute combinations, their control parameters are relatively close to those of *in-distribution*. Therefore, we approach this from the perspective of control parameters, calculating the $L1$ distance $L1_{base}, L1_{meta}$ and cosine similarity Cos_{base}, Cos_{meta} between the control parameters before and after the introduction of the Meta-MCTG framework, and use the difference $diff_{L1} = \frac{L1_{meta} - L1_{base}}{L1_{meta}} \times 100$, $diff_{Cos} = -\frac{Cos_{meta} - Cos_{base}}{Cos_{meta}} \times 100$ between the two as the data for visualization.

We select *CTRL* (Keskar et al., 2019) , *DCG* (Zeng et al., 2023), and *Contrastive Prefix* (Qian et al., 2022b) and conduct our visualization experiments on *ACD* protocol of *YELP* (YELP, 2014) and *Fyelp* (Lample et al., 2019) datasets. For *CTRL*, we use the mean embeddings of its attribute tokens (i.e., control codes) as the control parameters. For *DCG*, we use the mean embedding obtained by encoding the attribute tokens through a

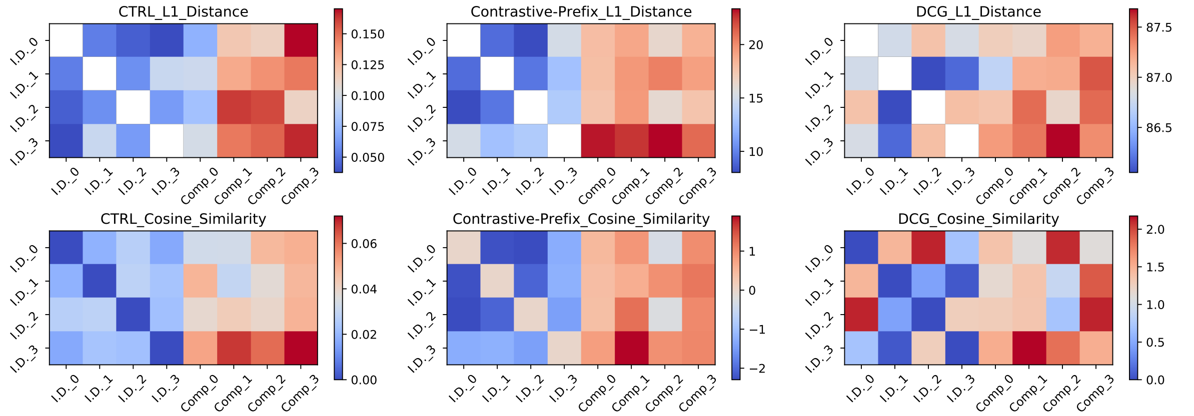

Figure 10: Difference of the distances ($d_{cos} = 1 - cos < h_1, h_2 >$, $d_{l1} = |h_1 - h_2|$) between attribute combinations in the representation space (h_1, h_2) with *Meta-CTRL Meta Contrastive Prefix*, *Meta-DCG* and the origin version of *CTRL, Contrastive Prefix, DCG* in dataset *YELP*.

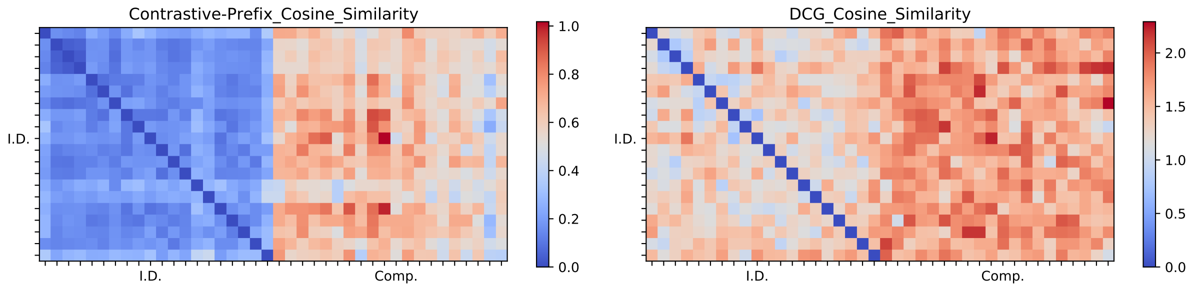

Figure 11: Difference of the distances ($d_{cos} = 1 - cos < h_1, h_2 >$) between attribute combinations in the representation space (h_1, h_2) with *Meta Contrastive Prefix*, *Meta-DCG* and the origin version of *Contrastive Prefix, DCG* in dataset *Fyelp*.

fully connected layer as the control parameters. For *Contrastive Prefix*, we use the mean embedding of the prefix keys and prefix values of the corresponding attributes in the last layer of the GPT-2 as the control parameters. On the *YELP* dataset, there are a total of 8 attribute combinations, including 4 in-distribution and 4 compositional. For the control parameters under 8 control conditions, we compute the difference $diff_{L1}$ and $diff_{Cos}$ between each pair and obtain two 4×8 heatmaps for each baseline. Similarly, for the *Fyelp* dataset, we can get two 20×40 heatmaps for each baseline. The results are shown in Figure 10 and Figure 11. The visual results show that the control parameters after the Meta-MCTG training framework can better distinguish between the in-distribution and compositional parts, thus confirming the effectiveness of the Meta-MCTG framework.

I.3 Detailed Results on the Single Dataset

In this section, we provide detailed experimental results of all baselines (eight MCTG baselines and two LLMs) in CompMCTG Benchmark in 4 datasets. In these tables, the first column contains the protocol, including *Original*, *HoldOut*, *ACD*, and *FewShot* (*Amazon* and *Mixture* datasets do not have *ACD* protocol). *Holdout*, *ACD*, and *FewShot* respectively divide the *in-distribution* (*I.D.*) results and *compositional* (*Comp.*) results. The second column is the method name and the next two to four columns are the accuracy of the corresponding attributes. Specifically, $Acc_s, Acc_g, Acc_c, Acc_t$ in *Fyelp* are the accuracy of sentiment, gender, cuisine, and tense respectively. Acc_s, Acc_t in *Amazon* are the accuracy of sentiment and topic respectively. Acc_s, Acc_p, Acc_t in *YELP* are the accuracy of sentiment, person, and tense respectively. Acc_s, Acc_t in *Mixture* are the accuracy of sentiment and topic respectively. Acc_{avg} is the average of accuracy in previous columns. Acc_{gap} is calculated from the *I.D.* and *Comp.* of the corresponding protocol. The specific formula is $Acc_{gap} = \frac{Acc_{I.D.} - Acc_{Comp.}}{Acc_{I.D.}} \times 100\%$. PPL represents perplexity and $Dist3$ is the value of 3-ngrams. All the detailed experimental results can be found in Table 15 to Table 54.

Method	Comb.	Generated Texts
CTRL	*Negative* *Female* *Bar* *Past*	**Furthermore**, where's the shrimp? there are currently seven shrimp clusters. thank god there are still two clusters and that's where we ordered our food... at this location. despite this, we waited over 45 minutes. the server put me on
CTRL+Meta	*Negative* *Female* *Bar* *Past*	**Furthermore**, this tiny place fell short of its potential. The food was enormous but not up to par. I ordered an oyster benedict which was disappointing, and the wine was subpar. I still find myself having nightmares about that dessert, which was made with stale
CTRL	*Positive* *Female* *Asian* *Past*	**The movie** i saw called därk star änd it really impressed me. the chef in the movie is so nice and did his chef impression. this restaurant was amazing to watch. we had the eggplant korma and beef teriyaki.
CTRL+Meta	*Positive* *Female* *Asian* *Past*	**The movie** started and we were getting ready for bed but the korean guys got us a lot of food. my husband and i both got the delicious beef kimchi rice dish and both really liked it and were very happy with it. we ended
DCG	*Positive* *Female* *dessert* *Past*	**In brief**, we are so glad we went here. it's always as cold as winter to get the fruit or veg for me (some frozen squash, maybe a banana, too). They have everything we go for here, and they
DCG+Meta	*Positive* *Female* *dessert* *Past*	**In brief**, this was an awesome place. Forget the size of it, which i really found to be little too large, this was SO GOOD. We stopped in for breakfast and decided to try the sweet omelet pancakes. My husband and
DCG	*Negative* *Male* *Mexican* *Present*	**More importantly**, they have no toilet paper. would NEVER EVER order coffee or soda here.! they also give you a coupon for soup to go. not the best. everyone is rude. it is a crowded place. what gives there drive is that
DCG+Meta	*Negative* *Male* *Mexican* *Present*	**More importantly**, the food isn 't good enough for me. my girlfriend's favorite taco out of the bunch, Taco Linguini, is supposed to be good but she never saw it ; dang there you guys. my salsa is really a letdown. It's too bland and lacks the right kick
Con.P.	*Negative* *Female* *Asian* *Past*	**The last time** I went to a restaurant in town for sushi I was happy with the time but was disappointed the broth was chalky with soy sauce and rice. The temperature was extreme and the restaurant had no food prepared that looked appealing even when I
Con.P.+Meta	*Negative* *Female* *Asian* *Past*	**The last time** I was to see the sushi place here I felt poor. My boyfriend and I felt uneasy entering our table, so we were at all to begin with and he waited outside to eat lunch all the way until we were seated. The food was bad
Con.P.	*Positive* *Male* *American* *Past*	**The book** is well written and well planned with lots of really delicious-to-and-simple recipes and an in depth look at the last few years in the region with some wonderful photos and interesting twists on local food. Many thanks to my husband for
Con.P.+Meta	*Positive* *Male* *American* *Past*	**The book** commenced with the account of a baseball-loving American daycare worker in a center for immigrant families on Thanksgiving. "Every day, this gentle man, with his warm smile, taught the children that their most vital abilities resided within them

Table 14: A case study of the state-of-the-art baselines before and after incorporating the Meta-MCTG training framework. Different attribute words are marked with their corresponding colors. The text in bold represents the prompt. "Comb." means attribute combination and "Con.P." represents the baseline ContrastivePrefix.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CTRL*	88.28	60.13	60.38	67.29	69.02	-	45.69	0.675
HoldOut-I.D.	*CTRL*	88.42	60.88	60.53	67.89	69.43	1.64	45.95	0.675
HoldOut-Comp.	*CTRL*	87.88	59.65	59.02	66.61	68.29		45.61	0.676
ACD-I.D.	*CTRL*	87.83	60.25	59.45	69.35	69.22	5.65	45.60	0.684
ACD-Comp.	*CTRL*	87.00	55.35	58.93	59.95	65.31		45.86	0.678
FewShot-I.D.	*CTRL*	84.06	70.03	54.71	69.11	69.48	13.95	45.01	0.683
FewShot-Comp.	*CTRL*	82.37	48.35	55.75	52.70	59.79		44.33	0.684

Table 15: The result of baseline *CTRL* (Keskar et al., 2019) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CTRL*	88.43	71.76	80.10	-	37.97	0.731
HoldOut-I.D.	*CTRL*	88.77	72.00	80.39	2.67	37.87	0.734
HoldOut-Comp.	*CTRL*	86.55	69.93	78.24		38.10	0.736
FewShot-I.D.	*CTRL*	88.60	70.29	79.45	9.13	37.40	0.734
FewShot-Comp.	*CTRL*	76.53	67.87	72.20		37.50	0.740

Table 16: The result of baseline *CTRL* (Keskar et al., 2019) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CTRL*	90.07	75.71	89.82	85.20	-	84.94	0.356
HoldOut-I.D.	*CTRL*	91.47	74.28	89.72	85.16	3.69	72.20	0.360
HoldOut-Comp.	*CTRL*	89.89	69.00	87.18	82.02		73.74	0.368
ACD-I.D.	*CTRL*	91.76	74.35	90.46	85.52	12.73	76.06	0.348
ACD-Comp.	*CTRL*	88.06	55.81	80.03	74.63		75.46	0.359
FewShot-I.D.	*CTRL*	90.05	76.55	89.73	85.44	25.02	63.72	0.269
FewShot-Comp.	*CTRL*	81.90	47.54	62.73	64.06		64.74	0.338

Table 17: The result of baseline *CTRL* (Keskar et al., 2019) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CTRL*	76.14	88.04	82.09	-	48.11	0.736
HoldOut-I.D.	*CTRL*	72.45	88.66	80.56	10.85	48.82	0.723
HoldOut-Comp.	*CTRL*	66.46	77.18	71.82		47.46	0.755
FewShot-I.D.	*CTRL*	68.71	85.51	77.11	12.19	47.79	0.699
FewShot-Comp.	*CTRL*	61.21	74.20	67.71		46.31	0.709

Table 18: The result of baseline *CTRL* (Keskar et al., 2019) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CatPro*	84.65	54.43	53.72	63.91	64.18	-	70.58	0.726
HoldOut-I.D.	*CatPro*	84.45	54.76	56.80	64.64	65.16	0.91	69.71	0.726
HoldOut-Comp.	*CatPro*	83.82	54.07	56.04	64.36	64.57		69.48	0.725
ACD-I.D.	*CatPro*	83.45	54.04	47.33	61.21	61.51	10.96	69.30	0.735
ACD-Comp.	*CatPro*	71.26	50.11	35.36	62.35	54.77		63.83	0.750
FewShot-I.D.	*CatPro*	79.31	66.71	37.54	63.00	61.64	26.10	70.94	0.741
FewShot-Comp.	*CatPro*	46.04	48.28	24.11	63.75	45.55		68.16	0.740

Table 19: The result of baseline *CatPrompt* (Yang et al., 2023) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CatPro*	82.31	60.88	71.60	-	55.08	0.734
HoldOut-I.D.	*CatPro*	83.00	56.99	70.00	9.89	57.50	0.701
HoldOut-Comp.	*CatPro*	72.86	53.29	63.08		50.39	0.727
FewShot-I.D.	*CatPro*	77.95	44.64	61.30	35.42	55.63	0.658
FewShot-Comp.	*CatPro*	48.22	30.96	39.59		41.59	0.717

Table 20: The result of baseline *CatPrompt* (Yang et al., 2023) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CatPro*	78.93	51.43	75.43	68.60	-	83.96	0.467
HoldOut-I.D.	*CatPro*	76.04	51.67	74.86	67.52	4.83	86.92	0.462
HoldOut-Comp.	*CatPro*	70.68	50.18	71.93	64.26		86.79	0.467
ACD-I.D.	*CatPro*	72.24	52.88	73.23	66.12	14.10	118.02	0.634
ACD-Comp.	*CatPro*	47.54	49.75	73.12	56.80		105.37	0.657
FewShot-I.D.	*CatPro*	79.86	57.07	84.21	73.71	21.39	378.69	0.448
FewShot-Comp.	*CatPro*	45.43	49.73	78.65	57.94		349.24	0.585

Table 21: The result of baseline *CatPrompt* (Yang et al., 2023) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*CatPro*	51.61	50.86	51.24	-	88.51	0.641
HoldOut-I.D.	*CatPro*	51.53	54.67	53.10	7.01	79.25	0.654
HoldOut-Comp.	*CatPro*	50.36	48.39	49.38		69.87	0.705
FewShot-I.D.	*CatPro*	54.52	51.91	53.22	21.42	149.37	0.679
FewShot-Comp.	*CatPro*	53.11	30.52	41.82		63.00	0.629

Table 22: The result of baseline *CatPrompt* (Yang et al., 2023) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*DCG*	90.18	56.68	56.50	62.34	66.43	-	53.31	0.688
HoldOut-I.D.	*DCG*	90.09	56.33	57.21	62.33	66.49	0.15	53.50	0.702
HoldOut-Comp.	*DCG*	90.29	56.39	57.00	61.88	66.39		53.52	0.704
ACD-I.D.	*DCG*	90.07	55.55	56.44	61.96	66.01	1.97	53.29	0.702
ACD-Comp.	*DCG*	89.73	55.04	54.99	59.07	64.71		53.67	0.704
FewShot-I.D.	*DCG*	89.00	68.26	50.37	65.63	68.32	25.91	53.30	0.704
FewShot-Comp.	DCG	57.34	49.02	41.68	54.42	50.62		52.82	0.695

Table 23: The result of baseline *DCG* (Zeng et al., 2023) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*DCG*	91.00	77.95	84.48	-	46.66	0.723
HoldOut-I.D.	*DCG*	91.13	78.29	84.71	0.24	47.20	0.727
HoldOut-Comp.	*DCG*	91.50	77.52	84.51		47.09	0.723
FewShot-I.D.	*DCG*	91.66	76.63	84.15	18.86	48.05	0.727
FewShot-Comp.	*DCG*	69.86	66.70	68.28		48.36	0.720

Table 24: The result of baseline *DCG* (Zeng et al., 2023) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL\downarrow$	$Dist3$
Original	*DCG*	95.75	66.57	91.07	84.46	-	57.08	0.706
HoldOut-I.D.	*DCG*	94.49	64.33	90.38	83.07	3.35	79.05	0.703
HoldOut-Comp.	*DCG*	94.50	58.75	87.61	80.29		80.58	0.721
ACD-I.D.	*DCG*	92.64	61.59	88.79	81.01	6.09	79.86	0.668
ACD-Comp.	*DCG*	88.06	57.90	82.28	76.08		84.30	0.686
FewShot-I.D.	*DCG*	90.82	62.21	85.93	79.65	29.57	93.66	0.510
FewShot-Comp.	*DCG*	55.15	52.51	60.63	56.10		111.03	0.653

Table 25: The result of baseline *DCG* (Zeng et al., 2023) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL\downarrow$	$Dist3$
Original	*DCG*	72.07	96.61	84.34	-	68.44	0.592
HoldOut-I.D.	*DCG*	73.86	95.35	84.61	10.83	68.45	0.645
HoldOut-Comp.	*DCG*	56.64	94.25	75.45		76.41	0.715
FewShot-I.D.	*DCG*	71.64	95.21	83.43	25.58	57.87	0.603
FewShot-Comp.	*DCG*	40.34	83.83	62.09		60.33	0.670

Table 26: The result of baseline *DCG* (Zeng et al., 2023) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL\downarrow$	$Dist3$
Original	*Fudge*	67.49	51.45	37.07	59.73	53.94	-	223.31	0.732
HoldOut-I.D.	*Fudge*	67.09	51.45	37.15	59.71	53.85	22.54	221.77	0.726
HoldOut-Comp.	*Fudge*	49.61	48.80	20.91	47.50	41.71		269.55	0.728
ACD-I.D.	*Fudge*	67.44	48.58	36.64	60.15	53.20	24.02	213.12	0.705
ACD-Comp.	*Fudge*	51.01	50.34	19.17	41.17	40.42		239.45	0.718
FewShot-I.D.	*Fudge*	70.83	79.46	25.80	45.54	55.41	26.06	208.09	0.666
FewShot-Comp.	*Fudge*	47.87	45.30	20.27	50.44	40.97		282.25	0.490

Table 27: The result of baseline *Fudge* (Yang and Klein, 2021) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL\downarrow$	$Dist3$
Original	*Fudge*	65.40	47.64	56.52	-	185.96	0.743
HoldOut-I.D.	*Fudge*	64.71	47.49	56.10	38.89	192.16	0.738
HoldOut-Comp.	*Fudge*	51.81	16.74	34.28		188.13	0.786
FewShot-I.D.	*Fudge*	64.16	54.30	59.23	41.53	206.58	0.722
FewShot-Comp.	*Fudge*	52.05	17.21	34.63		175.48	0.772

Table 28: The result of baseline *Fudge* (Yang and Klein, 2021) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL\downarrow$	$Dist3$
Original	*Fudge*	63.68	93.79	84.57	80.68	-	104.33	0.667
HoldOut-I.D.	*Fudge*	63.09	93.59	83.55	80.08	34.12	99.90	0.656
HoldOut-Comp.	*Fudge*	50.39	55.25	52.64	52.76		355.48	0.717
ACD-I.D.	*Fudge*	53.24	86.00	74.31	71.18	24.23	86.50	0.609
ACD-Comp.	*Fudge*	55.39	54.55	51.86	53.93		297.18	0.636
FewShot-I.D.	*Fudge*	58.32	87.32	71.32	72.32	29.29	58.13	0.481
FewShot-Comp.	*Fudge*	50.24	51.70	51.48	51.14		261.71	0.578

Table 29: The result of baseline *Fudge* (Yang and Klein, 2021) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Fudge*	56.00	42.64	49.32	-	200.42	0.483
HoldOut-I.D.	*Fudge*	54.22	40.51	47.37	16.34	204.05	0.487
HoldOut-Comp.	*Fudge*	51.96	27.29	39.63		195.15	0.254
FewShot-I.D.	*Fudge*	51.89	38.15	45.02	18.15	196.42	0.465
FewShot-Comp.	*Fudge*	48.65	25.05	36.85		180.19	0.221

Table 30: The result of baseline *Fudge* (Yang and Klein, 2021) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Lens*	96.89	59.31	77.23	70.77	76.05	-	51.09	0.555
HoldOut-I.D.	*Lens*	94.53	60.30	78.33	71.19	76.09	11.87	52.63	0.562
HoldOut-Comp.	*Lens*	77.03	56.05	78.23	56.93	67.06		52.59	0.556
ACD-I.D.	*Lens*	94.15	62.34	76.83	76.22	77.39	25.95	54.63	0.526
ACD-Comp.	*Lens*	60.80	57.27	51.68	59.49	57.31		54.15	0.469
FewShot-I.D.	*Lens*	97.00	70.00	74.29	84.80	81.52	36.73	50.69	0.539
FewShot-Comp.	*Lens*	63.60	50.63	34.18	57.92	51.58		50.25	0.501

Table 31: The result of baseline *Lens* (Gu et al., 2022) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Lens*	91.67	81.52	86.60	-	68.33	0.666
HoldOut-I.D.	*Lens*	91.68	83.31	87.50	47.78	69.95	0.660
HoldOut-Comp.	*Lens*	48.26	43.12	45.69		130.07	0.663
FewShot-I.D.	*Lens*	90.86	81.40	86.13	49.92	71.27	0.650
FewShot-Comp.	*Lens*	48.85	37.40	43.13		198.37	0.587

Table 32: The result of baseline *Lens* (Gu et al., 2022) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Lens*	79.54	96.75	93.36	89.88	-	265.42	0.284
HoldOut-I.D.	*Lens*	71.74	96.77	95.47	87.99	36.73	121.94	0.232
HoldOut-Comp.	*Lens*	51.54	64.75	50.71	55.67		122.77	0.231
ACD-I.D.	*Lens*	83.83	90.26	96.14	90.08	47.59	121.54	0.228
ACD-Comp.	*Lens*	48.78	52.94	39.92	47.21		121.13	0.233
FewShot-I.D.	*Lens*	98.54	89.25	97.25	95.01	36.07	142.18	0.212
FewShot-Comp.	*Lens*	62.87	58.14	61.20	60.74		141.35	0.271

Table 33: The result of baseline *Lens* (Gu et al., 2022) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Lens*	83.11	95.46	89.29	-	110.04	0.387
HoldOut-I.D.	*Lens*	82.14	93.37	87.76	38.58	138.82	0.410
HoldOut-Comp.	*Lens*	52.00	55.79	53.90		114.13	0.397
FewShot-I.D.	*Lens*	81.41	95.72	88.57	43.05	116.04	0.410
FewShot-Comp.	*Lens*	49.36	51.52	50.44		76.73	0.418

Table 34: The result of baseline *Lens* (Gu et al., 2022) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Prior*	72.43	52.02	48.39	63.58	59.11	-	72.14	0.602
HoldOut-I.D.	*Prior*	70.82	51.96	46.51	64.13	58.36	6.37	73.95	0.607
HoldOut-Comp.	*Prior*	63.56	50.79	43.58	60.62	54.64		73.91	0.609
ACD-I.D.	*Prior*	72.96	54.53	47.62	71.36	61.62	15.14	79.37	0.624
ACD.Comp.	*Prior*	68.42	48.29	48.26	44.20	52.29		79.10	0.627
FewShot-I.D.	*Prior*	98.11	73.89	55.83	86.86	78.67	32.54	84.29	0.643
FewShot-Comp.	*Prior*	59.07	47.37	48.67	57.18	53.07		83.13	0.576

Table 35: The result of baseline *Prior* (Gu et al., 2023) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Prior*	82.02	82.90	82.46	-	86.79	0.647
HoldOut-I.D.	*Prior*	83.78	79.46	81.62	40.74	86.93	0.644
HoldOut-Comp.	*Prior*	25.76	70.98	48.37		84.02	0.650
FewShot-I.D.	*Prior*	96.91	78.99	87.95	40.11	93.00	0.643
FewShot-Comp.	*Prior*	54.43	50.90	52.67		93.80	0.648

Table 36: The result of baseline *Prior* (Gu et al., 2023) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Prior*	70.96	65.11	82.93	73.00	-	124.68	0.477
HoldOut-I.D.	*Prior*	73.48	63.91	80.62	72.67	24.99	68.44	0.379
HoldOut-Comp.	*Prior*	55.89	51.18	56.46	54.51		65.61	0.398
ACD-I.D.	*Prior*	79.93	68.35	82.45	76.91	39.11	82.68	0.347
ACD-Comp.	*Prior*	48.45	51.56	40.48	46.83		72.61	0.344
FewShot-I.D.	*Prior*	89.68	77.07	96.21	87.65	39.92	98.73	0.287
FewShot-Comp.	*Prior*	53.36	51.62	53.00	52.66		94.69	0.345

Table 37: The result of baseline *Prior* (Gu et al., 2023) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Prior*	77.79	83.89	80.84	-	196.01	0.565
HoldOut-I.D.	*Prior*	81.69	82.08	81.89	48.49	205.01	0.558
HoldOut-Comp.	*Prior*	41.07	43.29	42.18		167.01	0.535
FewShot-I.D.	*Prior*	85.56	87.42	86.49	44.02	199.85	0.541
FewShot-Comp.	*Prior*	49.40	47.43	48.42		145.01	0.540

Table 38: The result of baseline *Prior* (Gu et al., 2023) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Con.P*	93.47	59.39	50.41	69.11	68.10	-	51.76	0.704
HoldOut-I.D.	*Con.P*	93.67	59.25	49.64	68.79	67.84	0.50	52.48	0.701
HoldOut-Comp.	*Con.P*	93.66	59.24	48.30	68.78	67.50		52.32	0.705
ACD-I.D.	*Con.P*	92.50	57.39	39.04	64.68	63.40	-0.84	53.11	0.704
ACD-Comp.	*Con.P*	93.85	58.24	40.18	63.44	63.93		49.78	0.745
FewShot-I.D.	*Con.P*	81.69	72.09	24.49	60.40	59.67	24.03	76.80	0.744
FewShot-Comp.	*Con.P*	58.89	47.51	22.39	52.51	45.33		86.49	0.745

Table 39: The result of baseline *Contrastive Prefix* (Qian et al., 2022b) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Con.P*	93.76	81.31	87.54	-	43.55	0.716
HoldOut-I.D.	*Con.P*	94.26	81.27	87.77	-0.50	43.84	0.713
HoldOut-Comp.	*Con.P*	94.67	81.74	88.21		44.49	0.716
FewShot-I.D.	*Con.P*	92.93	77.13	85.03	19.45	43.92	0.713
FewShot-Comp.	*Con.P*	82.72	54.26	68.49		43.28	0.727

Table 40: The result of baseline *Contrastive Prefix* (Qian et al., 2022b) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Con.P*	98.21	87.11	99.21	94.84	-	139.13	0.709
HoldOut-I.D.	*Con.P*	98.03	85.91	99.26	94.40	1.71	136.04	0.687
HoldOut-Comp.	*Con.P*	97.36	82.11	98.89	92.79		132.21	0.707
ACD-I.D.	*Con.P*	96.52	80.96	98.66	92.05	3.34	139.71	0.669
ACD-Comp.	*Con.P*	96.27	72.73	97.93	88.98		131.12	0.674
FewShot-I.D.	*Con.P*	96.09	78.25	97.82	90.72	35.53	136.95	0.527
FewShot-Comp.	*Con.P*	60.87	52.94	61.65	58.49		132.02	0.624

Table 41: The result of baseline *Contrastive Prefix* (Qian et al., 2022b) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*Con.P*	75.68	95.25	85.47	-	82.73	0.676
HoldOut-I.D.	*Con.P*	75.87	94.08	84.98	14.16	89.59	0.681
HoldOut-Comp.	*Con.P*	66.82	79.07	72.95		119.74	0.778
FewShot-I.D.	*Con.P*	74.12	94.11	84.12	31.12	86.10	0.642
FewShot-Comp.	*Con.P*	52.47	63.40	57.94		111.43	0.723

Table 42: The result of baseline *Contrastive Prefix* (Qian et al., 2022b) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*PPLM*	49.86	50.00	19.91	49.91	42.42	-	355.27	0.691
HoldOut-I.D.	*PPLM*	50.43	50.03	20.34	50.31	42.78	0.68	351.74	0.687
HoldOut-Comp.	*PPLM*	49.96	50.02	19.93	50.06	42.49		365.57	0.688
ACD-I.D.	*PPLM*	49.30	52.75	20.62	54.55	44.31	8.31	348.59	0.688
ACD-Comp.	*PPLM*	50.57	47.25	19.42	45.27	40.63		329.13	0.688
FewShot-I.D.	*PPLM*	55.11	79.57	19.06	42.14	48.97	15.15	470.44	0.692
FewShot-Comp.	*PPLM*	49.42	45.79	20.09	50.90	41.55		332.87	0.686

Table 43: The result of baseline *PPLM* (Dathathri et al., 2019) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*PPLM*	49.60	16.62	33.11	-	340.99	0.689
HoldOut-I.D.	*PPLM*	50.31	17.24	33.78	1.51	379.86	0.689
HoldOut-Comp.	*PPLM*	49.64	16.89	33.27		346.97	0.691
FewShot-I.D.	*PPLM*	53.04	16.75	34.90	8.51	343.87	0.690
FewShot-Comp.	*PPLM*	47.01	16.85	31.93		355.93	0.686

Table 44: The result of baseline *PPLM* (Dathathri et al., 2019) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*PPLM*	50.43	49.86	49.75	50.01	-	297.53	0.704
HoldOut-I.D.	*PPLM*	50.46	49.43	48.79	49.56	0.93	294.58	0.422
HoldOut-Comp.	*PPLM*	50.32	48.28	48.70	49.10		294.58	0.695
ACD-I.D.	*PPLM*	54.46	50.04	50.42	51.64	5.58	289.95	0.439
ACD-Comp.	*PPLM*	45.54	50.10	50.65	48.76		285.21	0.434
FewShot-I.D.	*PPLM*	49.86	49.71	51.25	50.27	0	302.25	0.492
FewShot-Comp.	*PPLM*	49.86	49.71	51.25	50.27		302.26	0.438

Table 45: The result of baseline *PPLM* (Dathathri et al., 2019) in dataset *YELP*.

Protocol	Method	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*PPLM*	51.71	24.50	38.11	-	296.57	0.704
HoldOut-I.D.	*PPLM*	51.18	24.93	38.06	1.21	274.16	0.690
HoldOut-Comp.	*PPLM*	50.14	25.05	37.60		355.92	0.702
FewShot-I.D.	*PPLM*	50.94	25.35	38.15	2.80	329.85	0.665
FewShot-Comp.	*PPLM*	48.93	25.22	37.08		332.68	0.660

Table 46: The result of baseline *PPLM* (Dathathri et al., 2019) in dataset *Mixture*.

Protocol	Method	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*llama2*	66.57	52.00	32.50	56.07	51.78	-	17.64	0.473
HoldOut-I.D.	*llama2*	66.94	52.72	30.81	55.99	51.61	15.09	17.08	0.387
HoldOut-Comp.	*llama2*	56.43	49.79	20.36	48.71	43.82		16.56	0.449
ACD-I.D.	*llama2*	68.36	51.51	29.50	56.94	51.58	15.99	16.72	0.379
ACD-Comp.	*llama2*	55.31	49.37	20.67	47.96	43.33		17.34	0.371
FewShot-I.D.	*llama2*	65.37	52.17	29.77	56.11	50.86	12.09	17.21	0.444
FewShot-Comp.	*llama2*	57.59	49.17	21.07	50.99	44.71		17.46	0.374

Table 47: The result of baseline *LLaMA-2* (Touvron et al., 2023) in dataset *Fyelp*.

Protocol	Method	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*llama2*	68.10	53.10	60.60	-	15.25	0.633
HoldOut-I.D.	*llama2*	72.03	51.13	61.58	47.22	15.16	0.442
HoldOut-Comp.	*llama2*	47.86	17.14	32.50		15.50	0.622
FewShot-I.D.	*llama2*	75.81	51.10	63.45	49.24	15.14	0.474
FewShot-Comp.	*llama2*	47.86	16.57	32.21		15.23	0.474

Table 48: The result of baseline *LLaMA-2* (Touvron et al., 2023) in dataset *Amazon*.

Protocol	Method	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*llama2*	74.29	51.43	70.36	65.36	-	48.79	0.575
HoldOut-I.D.	*llama2*	70.92	53.06	72.81	65.60	27.59	46.45	0.391
HoldOut-Comp.	*llama2*	49.64	50.00	42.86	47.50		47.49	0.551
ACD-I.D.	*llama2*	68.93	54.64	72.29	65.29	22.81	54.56	0.410
ACD-Comp.	*llama2*	50.86	49.71	50.64	50.40		49.36	0.399
FewShot-I.D.	*llama2*	72.68	52.50	70.36	65.18	19.42	45.17	0.486
FewShot-Comp.	*llama2*	56.61	50.06	50.89	52.52		46.32	0.384

Table 49: The result of baseline *LLaMA-2* (Touvron et al., 2023) in dataset *YELP*.

Protocol	**Method**	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*llama2*	52.14	84.64	68.39	-	27.53	0.667
HoldOut-I.D.	*llama2*	58.78	84.54	71.66	44.92	23.49	0.500
HoldOut-Comp.	*llama2*	51.07	27.86	39.47		15.65	0.686
FewShot-I.D.	*llama2*	56.52	86.70	71.61	40.65	26.81	0.559
FewShot-Comp.	*llama2*	56.79	28.21	42.50		16.57	0.558

Table 50: The result of baseline *LLaMA-2* (Touvron et al., 2023) in dataset *Mixture*.

Protocol	**Method**	Acc_s	Acc_g	Acc_c	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*gpt3.5*	66.29	52.29	28.14	57.00	50.93	-	13.41	0.454
HoldOut-I.D.	*gpt3.5*	67.07	51.10	27.90	56.29	50.59	7.61	13.39	0.347
HoldOut-Comp.	*gpt3.5*	59.05	52.06	31.11	44.76	46.74		12.50	0.652
ACD-I.D.	*gpt3.5*	64.25	50.68	29.34	56.43	50.17	5.74	13.52	0.347
ACD-Comp.	*gpt3.5*	60.12	49.45	27.77	51.80	47.29		13.29	0.369
FewShot-I.D.	*gpt3.5*	49.14	58.00	26.00	62.29	48.86	2.89	13.06	0.627
FewShot-Comp.	*gpt3.5*	68.65	48.08	25.35	47.71	47.45		13.07	0.401

Table 51: The result of baseline *ChatGPT* (gpt-3.5-turbo-0613) (OpenAI, 2023) in dataset *Fyelp*.

Protocol	**Method**	Acc_s	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*gpt3.5*	77.86	33.33	55.59	-	14.13	0.670
HoldOut-I.D.	*gpt3.5*	74.72	36.54	55.63	15.69	14.50	0.417
HoldOut-Comp.	*gpt3.5*	75.71	18.10	46.90		14.94	0.667
FewShot-I.D.	*gpt3.5*	79.29	36.43	57.86	20.26	14.50	0.472
FewShot-Comp.	*gpt3.5*	71.52	20.76	46.14		14.24	0.474

Table 52: The result of baseline *ChatGPT* (gpt-3.5-turbo-0613) (OpenAI, 2023) in dataset *Amazon*.

Protocol	**Method**	Acc_s	Acc_p	Acc_t	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*gpt3.5*	53.57	51.43	66.79	57.26	-	25.58	0.596
HoldOut-I.D.	*gpt3.5*	60.97	50.41	65.77	59.05	6.86	26.43	0.367
HoldOut-Comp.	*gpt3.5*	67.14	50.36	47.50	55.00		26.41	0.614
ACD-I.D.	*gpt3.5*	60.86	51.43	67.71	60.00	4.88	25.76	0.400
ACD-Comp.	*gpt3.5*	71.07	50.71	49.43	57.07		28.81	0.421
FewShot-I.D.	*gpt3.5*	58.75	51.07	68.21	59.34	5.73	27.61	0.498
FewShot-Comp.	*gpt3.5*	65.42	50.54	51.85	55.94		26.98	0.384

Table 53: The result of baseline *ChatGPT* (gpt-3.5-turbo-0613) (OpenAI, 2023) in dataset *YELP*.

Protocol	**Method**	Acc_s	Acc_{tc}	Acc_{avg}	Acc_{gap}	$PPL \downarrow$	$Dist3$
Original	*gpt3.5*	69.64	62.86	66.25	-	19.00	0.722
HoldOut-I.D.	*gpt3.5*	63.47	58.93	61.20	21.23	18.84	0.500
HoldOut-Comp.	*gpt3.5*	66.43	30.00	48.21		20.10	0.707
FewShot-I.D.	*gpt3.5*	60.09	60.89	60.49	19.85	19.31	0.583
FewShot-Comp.	*gpt3.5*	67.41	29.55	48.48		16.54	0.562

Table 54: The result of baseline *ChatGPT* (gpt-3.5-turbo-0613) (OpenAI, 2023) in dataset *Mixture*.