Controllable Text Generation with Residual Memory Transformer

Hanqing Zhang[1*], **Si Sun**[2*], **Haiming Wu**[1], **Dawei Song** [1†]
[1]School of Computer Science & Technology, Beijing Institute of Technology
[2]China Academy of Launch Vehicle Technology
zhanghanqing@bit.edu.cn sunsi.shining@gmail.com
haimming@bit.edu.cn dwsong@bit.edu.cn
[*]Equal contributions [†]Corresponding author

Abstract

Large-scale Causal Language Models (CLMs), e.g., GPT3 and ChatGPT, have brought great success in text generation. However, it is still an open challenge to effectively control the generation process of a CLM while balancing the flexibility, control granularity, and generation efficiency. In this paper, we provide a new alternative for controllable text generation (CTG), by designing a non-intrusive, lightweight control plugin, namely Residual Memory Transformer (RMT), to accompany the generation of CLM at arbitrary time steps. With an encoder-decoder setup, RMT can accept any types of control conditions and cooperate with the base CLM through a residual learning paradigm, to achieve a more flexible, general, and efficient CTG. Extensive experiments are carried out on various control tasks, in the form of both automatic and human evaluations. The results demonstrate the superiority of RMT over a wide range of state-of-the-art CTG approaches. The code implementation of our work is available at: https://github.com/Residual_Memory_Transformer.

1 Introduction

Controllable text generation (CTG) focuses on generating text while adhering to specific constraints (Hu and Li, 2021; Zhang et al., 2022). These constraints can range from high-level semantic elements, such as emotions, topics, and toxicity avoidance, to finer-grained content, e.g., inclusion of specific concepts or key elements in the generated text. Recently, CTG has been regarded as critical for establishing safer, more reliable and practical (Krause et al., 2021; Liu et al., 2021a; Zhang et al., 2022) AI applications in the real world.

The current state-of-the-art CTG methods are based on Large Language Models (LLMs) that build upon the Transformer structure (Vaswani et al., 2017) and have gained significant attention due to their remarkable ability to understand and generate text. Recently, large-scale causal Language Models (CLMs), i.e., decoder-only language models, show particular advantages in zero/few-shot scenarios (Brown et al., 2020), resulting in a series of successors such as GPT3 and GPT4. This has been seen as a milestone towards the realization of Artificial General Intelligence.

Considering the significant scale of CLMs and the substantial cost of training such models, current mainstream CLM-based CTG methods fall into two categories, i.e., prompt-based and post-processing approaches. The prompt-based methods (Zhang and Song, 2022; Yang et al., 2022; Qian et al., 2022; Lu et al., 2022a; Zhou et al., 2023) concatenate the control-prompts or demonstrations with the input head of the generative model to instruct more controllable text generation. As previous studies have revealed (Zou et al., 2021; Carlsson et al., 2022), the control effectiveness tends to deteriorate with the increasing distance from the prompt. Additionally, inserting a control-prompt into a well-trained model may harm the model's original generative stream, thus losing the flexibility of control (Carlsson et al., 2022). On the other hand, most post-processing methods leverage an auxiliary module to adjust the probability of naturally producing a token by the generative model at the decoding phase (Liu et al., 2021a; Yang and Klein, 2021; Lu et al., 2022b; Zhong et al., 2023), hindering the model's capacity of content planning and thus limiting the fine-grained control. More recent decode-time methods (Li et al., 2022; Mireshghallah et al., 2022; Qin et al., 2022) improve the control granularity through iterative sampling or editing, but at the expense of generation efficiency. Therefore, the flexibility, control granularity and generation efficiency need to be better balanced, which demands a more versatile CLM-oriented CTG framework[1].

[1]We compare the key features of different CTG methods in Appendix C.

Findings of the Association for Computational Linguistics: ACL 2024, pages 1048–1066

In this paper, we propose a novel CTG plugin named Residual Memory Transformer (RMT), which borrows the paradigm of residual learning (He et al., 2016; Zhang et al., 2020a) and only makes late fusion with a frozen CLM to noninvasively steer the generation process. Unlike the prompt-based approaches, this paradigm does not disturb the original generative stream of the base CLM model, allowing for a better flexibility of CTG (i.e., control with a plug-and-play manner). In addition, the RMT architecture consists of an encoder-decoder structure, where the encoder handles different types of control information and influences the generation process, so as to achieve fine-grained control. Meanwhile, RMT utilizes cross-attention to uniformly apply control conditions to each generated token, avoiding the negative effect varying with context length. In particular, different from the vanilla decoder of Transformer, an additional CLM causal attention is introduced to extract the prior knowledge from the generative stream of CLM, allowing RMT not to deviate too far away from the original generative model while its implied high-level semantics is leveraged. The reuse of the base-CLM's output allows RMT to achieve effective control with a tiny network, resulting in an improved generation efficiency.

The training of RMT includes two stages: pre-training and fine-tuning. The pre-training of RMT aims to reconstruct noisy text into a complete sentence, facilitating RMT's understanding of the semantics of various control conditions, while aligning with the generative process of CLM. During the fine-tuning stage (i.e., residual learning), the logit of RMT is directly added to that of the fixed CLM, and the goal is to learn the parameters of RMT such that the joint model distribution gets close to the desired text distribution. Since the gradient does not need to be backpropagated to base-CLM, the training of RMT is efficient. For instance, in our experiments based on GPT2-large, the entire pre-training stage of RMT with 4M samples was completed in approximately 30 hours, using a single NVIDIA A6000 GPU. For comparison, the computational cost of RMT is approximately 1/10 of NRP citecarlsson-etal-2022-fine, a baseline CTG model used in our experiments.

We conduct extensive experiments to explore the superiority of our approach in three aspects. **(1) Flexibility:** Theoretically, the proposed RMT has the capability to intervene in the generation process at any step with a non-intrusive manner. Experimentally, it maintains the same level of control effectiveness in both with-context and without-context settings, showing that RMT enables long-distance control throughout the generation process. **(2) Control Granularity:** We test our approach on a range of CTG tasks of different granularity levels (i.e., fine-grained control tasks including word inclusion and sentence length control; and attribute control based on sentiment). RMT achieves a control effectiveness comparable to the state-of-the-art approaches, while guaranteeing the text quality. **(3) Efficiency:** The results show that three-layer blocks of RMT is enough to achieve a competitive CTG effectiveness, approving the parameter-efficient traits. The time-cost of text generation is almost the same to the original CLM, significantly outperforms the baselines.

2 Methodology

2.1 CTG Framework with RMT

As the dashed diagram in Figure 1 illustrates, RMT operates in a non-intrusive control mode as a CLM plug-in, where the control signal passes through RMT independently and affects the output distribution of a frozen CLM through residual learning ($y_t = y_t^g + y_t^c$) without interfering with the CLM's free-generation passageway.

Compared to intrusive controllable paradigms, e.g., prompt-based approaches or the attention-based CTG (e.g., CoCon (Chan et al., 2021) inserts a control block into the middle layer of GPT in an intrusive manner), RMT allows for more flexibility in switching between freestyle generation and controllable generation modes. This trait is very important in the LLM era. On the one hand, it avoids harm to the generality of LLM, keeping the powerful capacity to multi-task. On the other hand, the gradient does not need backpropagation into CLM with large-scale parameters, bringing significant training efficiency. Hence, the non-intrusive fashion for CTG is challenging, but also more promising. More explanation can be seen in Appendix C.

Formally, given a partial generated text $x_{<t}$ and a sequence of control instruction C, the proposed framework aims to generate eligible text $X_n = \{x_1, ..., x_n\}$ that meets the control conditions:

$$P_\Theta\left(X_n\right) = \prod_{t=1}^{n} P_\Theta\left(x_t \mid x_{<t}; C\right), \qquad (1)$$

where $\Theta = \{\tilde{\theta}; \phi\}$ represents the CTG model's parameters, which comprise both the frozen param-

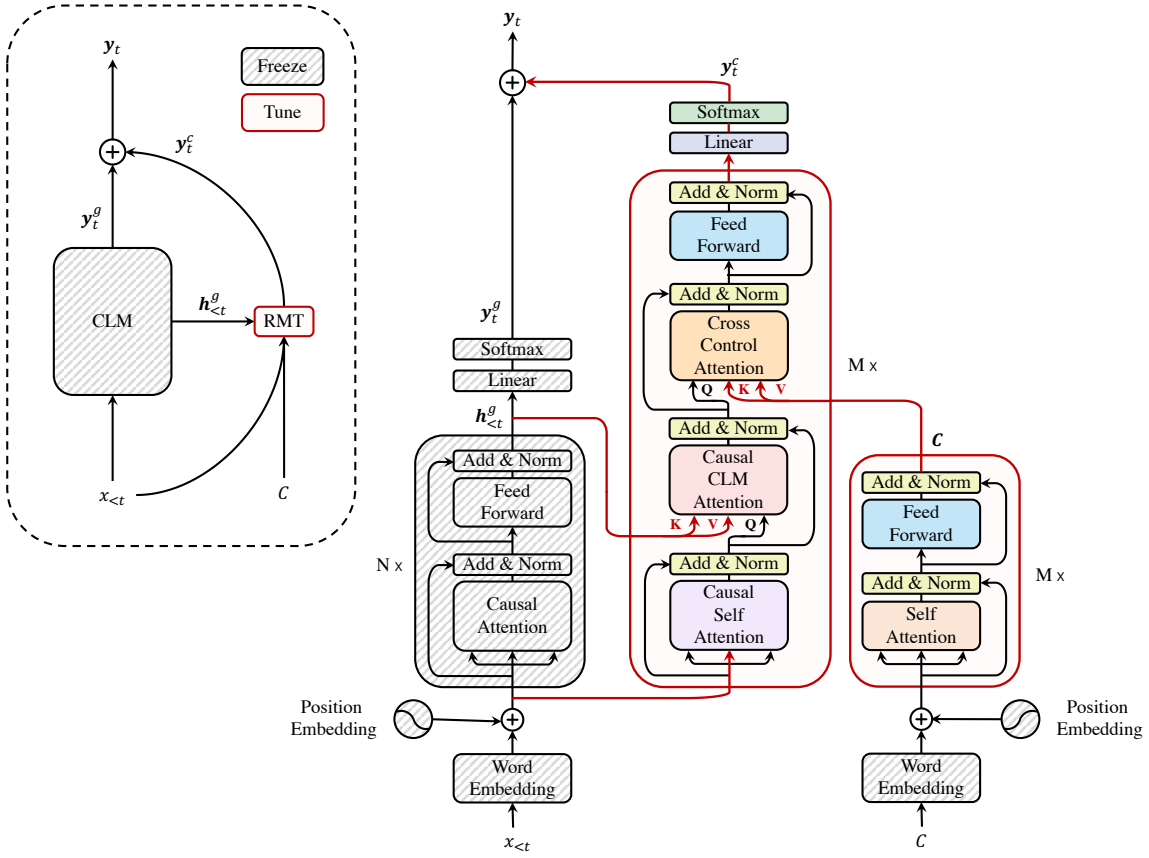

Figure 1: Illustration of controllable text generation with Residual Memory Transformer (RMT). In the top left, we present a miniaturization of our method framework, where the gray sector represents the frozen CLM, and the red box symbolizes our plug-in control module, i.e., RMT. The right part illustrates the embodiment of each detailed component.

eters inherited from the CLM ($\tilde{\theta}$) and the tunable parameters derived from the RMT (ϕ). The entire generation process of our approach consists of the following four steps:

CLM's Raw Generation. At the generation step t, CLM first maps the generated text $x_{<t} = \{x_1, ..., x_{t-1}\}$ to hidden states $\mathbf{h}^g_{<t} = \{\mathbf{h}^g_1, ..., \mathbf{h}^g_{t-1}\}$. Afterward, a linear layer and softmax normalization are used to project the hidden state of the last token $\mathbf{h}^g_{t-1}$ to the probability distribution $\mathbf{y}^g_t$ over the vocabulary:

$$\mathbf{y}^g_t = P_{\tilde{\theta}}(x_t|x_{<t}) = \text{softmax}(\text{Linear}(\mathbf{h}^g_{t-1})), \tag{2}$$

where the CLM naturally generates the next token x_t given the context $x_{<t}$ as per its training.

RMT's Control Encoding. Next, the RMT's encoder is responsible for encoding the control instruction $C = \{c_1, ..., c_m\}$ into the control memory $\mathbf{C} = \{\mathbf{c_1}, ..., \mathbf{c_m}\}$, which is used to guide the controllable text generation in the RMT's decoder:

$$\mathbf{C} = \text{RMT-Enc}(C; \phi_{\text{enc}}). \tag{3}$$

RMT's Control Decoding. After encoding the control signal, the RMT's decoder maps the generated text $x_{<t}$ to new hidden states $\mathbf{h}^c_{<t} = \{\mathbf{h}^c_1, ..., \mathbf{h}^c_{t-1}\}$ by considering both the control memory $\mathbf{C}$ and CLM's vanilla hidden states $\mathbf{h}^g_{<t}$, and synthetically predicting next token's probability distribution $\mathbf{y}^c_t$ over the vocabulary:

$$\begin{aligned}\mathbf{h}^c_{<t} &= \{\mathbf{h}^c_1 ..., \mathbf{h}^c_{t-1}\} = \text{RMT-Dec}(x_{<t}, \mathbf{h}^g_{<t}, \mathbf{C}; \phi_{\text{dec}}), \\ \mathbf{y}^c_t &= P_\phi(x_t|x_{<t}; C) = \text{softmax}(\text{Linear}(\mathbf{h}^c_{t-1})).\end{aligned} \tag{4}$$

Residual Learning to Generate. We employ residual learning to fuse the output distributions from the CLM (Eq. 2) and the RMT (Eq. 4), allowing the framework to obtain the joint predictions for the next token and achieve a noninvasive CTG:

$$\mathbf{y}_t = P_\Theta(x_t|x_{<t}; C) = \mathbf{y}^g_t + \mathbf{y}^c_t. \tag{5}$$

2.2 Residual Memory Transformer (RMT)

The detailed structure of RMT is shown in Figure 1. Specifically, RMT adopts an encoder-decoder architecture and reuses the CLM's suite of word and position embeddings.

RMT Encoder. The RMT encoder is responsible for encoding the control description C into the control memory $\mathbf{C}$ (Eq. 3). The encoder is composed of a stack of M identical blocks. Each block comprises a self-attention layer (Eq. 6) and a fully connected feed-forward layer. Additionally, it incorporates residual connections around above each layer, and is followed by a layer normalization.

RMT Decoder. The RMT decoder aims to predict the probability distribution of the next token $\mathbf{y}_t^c$ (Eq. 4) in the control mode. The decoder is also composed of a stack of M identical blocks. Each block contains three carefully-designed attention layers and a fully connected feed-forward network. Similar to the encoder, residual connections and layer normalization are also applied. The three attention layers are directed at the already generated text $x_{<t}$, the CLM's output hidden states $\mathbf{h}_{<t}^g$, and the control memory $\mathbf{C}$, respectively:

Causal Self Attention. The first attention layer utilizes a causal attention operation (Eq. 7 in Appendix A), whereby Q, K, and V are all firstly mapped from the original generated text $x_{<t}$ itself. This attention mechanism facilitates the identification and capturing contextual features of generated sequence from scratch.

Causal CLM Attention. The second attention layer also employs causal attention, but with a key difference: Q is sourced from the previous causal self-attention layer's output, while K and V are obtained from the CLM's last hidden states $\mathbf{h}_{<t}^g$. This design establishes an inner residual connection with CLM, enabling RMT to consider the high-level contextual features and maximally interact with the generative stream of CLM.

Cross Control Attention. The third attention layer is a cross-attention (Refer to the Appendix A) for the control memory from the RMT encoder. Specifically, Q is sourced from the previous causal CLM attention layer, while K and V are derived from the control memory $\mathbf{C}$. This cross-attention layer bridges the RMT's encoder and decoder, introducing the control signals to the generation. The cross-attention allows that every hidden vector of the current token can equally query the control instruction sequence encoded by RMT's encoder, avoiding the negative effect of the control instruction's token position.

2.3 Model Training

Pre-training. To enable the RMT's semantic understanding capability and allow it align with the

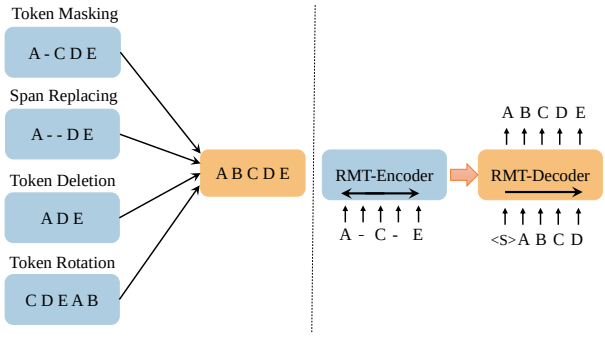

Figure 2: The schematic diagram for the denoising pre-training. The left hand shows transformations for noising the text, and the right hand shows the inputs and outputs of RMT.

CLM's generation process, we utilize the denoising auto-encoder pre-training method, whereby the encoder processes the corrupted text $\hat{X}$ regarded as control condition, and the decoder reconstructs the original text X.

The denoising pre-training schematic diagram could be seen in Figure2. Specifically, we corrupt the pretraining text ($X \rightarrow \hat{X}$) in four ways referring to (Lewis et al., 2020): *(1) Token Masking:* randomly masking tokens with special tokens. *(2) Token Deletion:* randomly deleting tokens in the text. *(3) Span Replacing:* randomly replacing the text spans of different lengths with special tokens. Different from token masking, each span is replaced with a single special token. *(4) Text Rotation:* randomly selecting a token and rotating the text around it, i.e., the text begins with that token and ends with its previous token. More details are supplied in Appendix B.1. ***Once the pre-training is complete, RMT can be fine-tuned to perform various controllable text generation tasks, without any need for re-pretraining.***

Fine-tuning. This allows the CLM to generate sequences that meet specific control requirements in an auto-regressive manner. The specific objective can vary according to different tasks: we use the *Maximum Likelihood Estimation (MLE)* objective (please see details in Appendix A.3) for the word inclusion and length control tasks; and use the unlikelihood training strategy (Zhang and Song, 2022) for the attribute control task.

It is worth noting that RMT shares the trait of training efficiency. Firstly, RMT efficiently reuses the output of the base CLM, which makes RMT requiring fewer pre-training datasets and expediting the pre-training process. Additionally, RMT is lightweight and is built on the top layer of the

base CLM. Therefore, gradients do not need to propagate into the base CLM during the backpropagation process, which can save a significant amount of training time and GPU memory.

3 Experiments

3.1 Word Inclusion Experiment

Two different settings are experimented in this subsection. The first one is the without-context setting, i.e., steering the CLM to generate a single sentence containing the keywords without the context. The instruction is as follows:*Continue to write a sentence, and include [$word_1$, $word_2$, ..., $word_n$].* The second is the with-context generation setting, which requires the CLM to continue to generate target text under an extended context. The instruction is:*Continue to write a sentence, and include [$word_1$, $word_2$, ..., $word_n$]. Context: [context text].* The consideration of these two modes aims to test the CTG's flexibility, capable of controlling the generation of CLM in a position-independent way.

Experimental Setting. Following the NRP (Carlsson et al., 2022), we use the GPT2-large as the backbone CLM, and the training data for pre-training and fine-tuning also comes from Wikipedia. More details on training and inference are provided in Appendix B.2. We test our approach on CommonGen (Lin et al., 2020) (for without-context setting), and C2Gen (Carlsson et al., 2022)(for with-context setting).

Baselines. As for the without-context setting, we firstly choose those methods that are specifically trained for lexical constraints tasks. They are KG-BART (Liu et al., 2021b), which fine-tunes the whole parameters of BART while it is augmented by knowledge graph; and POINT (Zhang et al., 2020b), which is an insertion-based method and iteratively injects words around the target words. Finally, we also employ a pure decode-time approaches for lexical constraint, namely NeuroLogic (Lu et al., 2022b), as a baselines. The general CTG methods are also compared, including Non-Residual Prompt (Carlsson et al., 2022)(i.e., a recent state-of-the-art method using position-independent prompting model to guide the generation of GPT2-large), and approaches directly instructing the GPT2-large and ChatGPT to generate target sentences. As for the with-context setting, KG-BART and POINT could not be applied to this scene and thus they are removed. The test results of POINT, KG-BART, and NRP are adopted from the open source code[3].

Evaluation. Following the NRP (Carlsson et al., 2022), We employ coverage (Cov), measuring the average coverage rate of the generated text with respect to the target words, to assess the controllability. Perplexity (PPL) is used to evaluate the fluency. We calculate PPL using an off-the-shelf GPT2-large model. Additionally, we utilize Self-Bleu-5 to evaluate the diversity of the generated text, with lower Self-Bleu values indicating greater syntactic diversity. Considering the non-intrusive setting (i.e., the CLM is fixed and the task is conducted on the open-ended text generation setting), our experiments omit that task-specific metrics like BLEU, METEOR, CIDEr, and SPICE for CommonGen dataset. Additionally, we conduct a human evaluation to assess the conformity of the generated text to common sense (**CS**) and text **Fluency** from human perspective. For the with-context setting, we introduce an additional manual evaluation metric called Relevance (Rel), where human evaluators scored the relevance of the generated text to the given context. More detailed information can be found in the Appendix D.

Result and Analysis. As is shown in Table 1, in the without-context setting, the coverage of RMT significantly outperforms the vanilla prompt method (i.e., Prompt+GPT2) and pure decode-time approach (i.e, NeuroLogit) with the same decode algorithm using by RMT. It shows slight advantages over ChatGPT and NRP. Compared to task-specific models (i.e., POINT, KG-BART), the coverage of RMT is close to them, yet with lower PPL. It is worth noting that, in the with-context setting, the strong baselines, including ChatGPT and NRP, respectively suffer from a 12.0% and 8.0% decline of control ability compared to the no-context setting. However, RMT maintains the same-level of control ability with the no-context setting, and outperforms NRP and ChatGPT, which shows the superiority of RMT in that the control ability will not degrade with the context length.

In term of human evaluation in CommonGen, POINT is an insertion-based generation framework that naturally suffered from worse text quality; thus the result of RMT is much better than POINT on terms of commonsense and fluency. Like NRP and Promp+GPT2, we all use frozen GPT-2-large as the backbone and do not introduce external knowledge.

[3] https://github.com/FreddeFrallan/Non-Residual-Prompting

Method	Cov (↑)	PPL (↓)	Self-Bleu (↓)	CS (↑)	Fluency (↑)
CommonGen (without-context setting)					
POINT (2020b)	**98.0**	65.6	27.7	4.8	4.0
KG-BART (2021b)	97.2	51.3	33.0	7.4	7.3
NeuroLogit (2022b)	77.7	**23.3**	56.2	4.6	4.5
NRP (2022)	93.0	42.3	28.4	6.1	7.0
GPT2+Prompt	70.9	61.3	48.3	6.3	7.5
GPT-3.5-turbo+Prompt	90.2	59.1	-	**7.9**	**8.2**
RMT (w/o CL)	93.1	56.4	**20.9**	6.5	6.7
RMT *(CL=15)*	93.9	60.3	22.5	-	-
RMT *(CL=18)*	93.7	47.9	23.7	-	-
RMT *(CL=20)*	93.8	44.1	23.2	-	-
C2Gen (with-context setting)					CS / Rel
NRP (2022)	81.0 (12% ↓)	-	-	-	-
GPT-3.5-turbo+Prompt	82.2 (8.0% ↓)	46.7	**5.2**	**9.0 / 8.5**	**8.8**
RMT (w/o CL)	**91.8 (1.3% ↓)**	**46.2**	5.8	7.1 / 8.3	6.4

Table 1: The experiment results on word inclusion. *CL* represent the external control length setting in our experiment. We set the number of RMT block-layers to three ($M = 3$). The result of NRP* in C2Gen is taken from what was reported in the original paper[2]. For the ChatGPT baseline in CommonGen, we test it on a subset of 500 samples, due to the problem of API access. For the with-context setting (C2Gen), KG-BART and POINT could not be applied to this scene and thus they are removed.

RMT achieves comparable results, which prove that the non-intrusive paradigm with residual learning would maintain the language model's ability, thereby guaranteeing the text quality. Our approach falls behind KG-BART enhanced by knowledge graph and ChatGPT with hundred-billion level parameters. This is within exception. If inserting RMT into larger-scale CLM, e.g., GPT3, we believe the text quality of our approach will also be improved. As for C2Gen, RMT produces comparable results in contextual relevance with ChatGPT, but there is still a gap in common sense. ***More qualitative examples of different CTG approaches can be seen in Appendix E.***

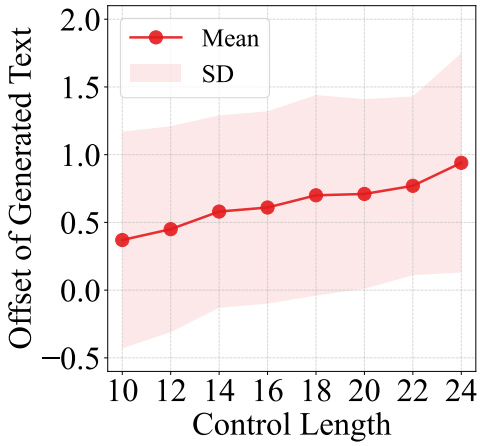

Figure 3: Control length and control performance.

3.2 Sentence Length Control Experiment

We test the length-control ability of RMT, i.e., instructing the CLM to generate a sentence which satisfies the word inclusion objective under an exact number of words. The instruction is as follows:*Continue to write a sentence, and include [word$_1$, word$_2$, ..., word$_n$], the sentence length should be within CL words.* This setting increases the challenge of word inclusion task, effectively approving the fine-grained control ability.

We report control performance for $CL = 15, 18, 20$. As shown in Table 1, RMT can maintain same level word coverage under different control-length. The longer the target length, the lower the generated text's PPL. This result is quite intuitive since it is harder to produce fluent text within the shorter required length.

To quantitatively measure the controllability of sentence length, we conduct tests on the CommonGen validation dataset to analyze the control effect of our proposed method. Some examples are presented in Table 2, which demonstrate that RMT effectively steers the CLM to generate texts with specific keywords and required sentence lengths. Furthermore, we calculated the mean offset (Mean) and the standard deviation (SD) for different con-

Target Words	Control Length	RMT's Generated Text
circle, sit, talk	13	The group of people sitting around talking in a circle around them.
	17	The woman is sitting in a circle and talking to a man in a white shirt.
	25	The group of people are sitting in a circle, talking about what they're going to do with their lives.
drink, sit, table, wine	13	The man is sitting on a table and drinking wine from a glass.
	17	The man is sitting on a table with a drink in his hand and drinking wine.
	25	The man is sitting at a table with a drink in his hand, while another man is drinking wine from a glass.

Table 2: The examples of generated text under different control length. The generated length is counted with GPT2's Tokenizer, thus it will make a little difference with the number of actual words.

trol lengths. The results are visualized in Figure 3. We observe that the offset and standard deviation both remain within one, indicating that RMT successfully achieves the sentence generation with desired sentence lengths as instructed.

3.3 Attribute Control Experiment

Setting and Baselines. We follow DisCup (Zhang and Song, 2022), and use a discriminator-involved training strategy to fine-tune RMT, and the checkpoint of attribute-classifier comes from the open source code[4]. More details could be seen in Appendix B.3. Different from DisCup, which optimizes unique prompt for every attribute, we use the RMT module to directly encode different attribute instructions uniformly. The instructions for sentiment control are *"The sentiment: positive"* and *"The sentiment: negative"*, respectively. GPT2-large is used as the backbone CLM, and the training data is the widely used Stanford Sentiment Tree (SST-5) (Socher et al., 2013) collected from movie reviews. We follow the commonly used setting in previous work (Krause et al., 2021; Zhang and Song, 2022; Lu et al., 2022a). Specifically we use 5K neutral prompts, 2.5K positive prompts, and 2.5K negative prompts as test dataset (provided by DEXPERT (Liu et al., 2021a)), and every CTG approach generates another 20 continuations based on given prompts, to achieve sentiment (i.e., positive and negative) controllable text generation. We collect the primary baselines reported by DisCup (Zhang and Song, 2022) and DEXPERT (Liu et al., 2021a), including the decode-time approaches with GPT2-large as base-CLM (i.e., PPLM (Dathathri et al., 2020),GEDI (Krause et al., 2021),and DEXPERT (Liu et al., 2021a)), and the training approaches (i.e., CTRL (Keskar et al., 2019) and DisCup (Zhang and Song, 2022)).

Evaluation. Following previous work (Krause et al., 2021; Zhang and Song, 2022; Lu et al., 2022a), we use an external sentiment classifier provided by `Huggingface.co` to classify the generated texts, and get sentiment control accuracy (i.e., Correctness). PPL and Dist-1/2/3 are reported to test their fluency and diversity, respectively. For human evaluation, we introduce *Relevance*, i.e., how the text conforms to required sentiment; *Topicality*, i.e., how the generated continuations are context-consistent with the given prompts, and *Fluency*, i.e., the text's fluency evaluated from the human perspective. More details can be seen in Appendix D.

Result and Analysis. As shown in Table 3, RMT in top-k setting demonstrates a superior control performance compared to all baselines, while maintaining comparable text quality and diversity. And RMT in the top-p setting shows better text quality yet weaker control performance. The closest performers to our approach are DEXPERT and DisCup. However, DEXPERT requires fine-tuning an external pair of CLM, resulting in tuning two time more parameters of the base CLM. In contrast, RMT is parameter-efficient, requiring tuning relevant parameters which are only around 16% of the base CLM's parameters. Regarding DisCup, both approaches employ the same loss function. However, DisCup optimizes different continuous prompts for each attribute, while we utilize the RMT to steer the CLM using residual learning. DisCup requires fine-tuning fewer parameters, while RMT is completely decoupled from the CLM and controls different attributes using a unified model,

[4] https://github.com/littlehacker26/Discriminator-Cooperative-Unlikelihood-Prompt-Tuning

Target Sentiment	Method	Correctness (↑)			Fluency (↓)	Diversity (↑)
		Positive	Neutral	Negative	PPL	Dist-1 / Dist-2 / Dist-3
Positive	PPLM (2020)		52.68	8.72	113.54	0.39 / 0.83 / 0.89
	CTRL (Keskar et al., 2019)		77.24	18.88	48.24	0.13 / 0.53 / 0.79
	GEDI (Krause et al., 2021)		86.01	26.80	123.56	0.20 / 0.66 / 0.85
	DEXPERT (Liu et al., 2021a)		94.46	36.42	60.64	0.18 / 0.63 / 0.84
	DisCup (2022)		94.20	60.40	46.6	0.14 / 0.51 / 0.78
	RMT+top-k		**97.62**	**67.20**	46.0	0.14 / 0.56 / 0.79
	RMT+top-p		94.50	42.60	**17.3**	0.13 / 0.45 / 0.65
Negative	PPLM (2020)	10.26	60.95		122.41	0.40 / 0.83 / 0.90
	CTRL (2019)	20.95	62.37		45.27	0.13 / 0.51 / 0.78
	GEDI (2021)	60.43	91.27		138.93	0.19 / 0.66 / 0.86
	DEXPERT (2021a)	64.01	**96.23**		67.12	0.20 / 0.64 / 0.83
	DisCup (2022)	62.80	91.40		47.90	0.13 / 0.50 / 0.77
	RMT+top-k	**77.16**	95.92		49.15	0.15 / 0.60 / 0.82
	RMT+top-p	56.4	92.7		**19.0**	0.13 / 0.48 / 0.70

Table 3: The experimental results of sentiment controllable text generation. Among the table, ↑ indicates that the higher corresponding value is better, and ↓ is the opposite.

Method	Relevance (↑)	Fluency (↑)	Topicality (↑)
DisCup (2022)	6.3	6.6	6.5
DEXPERT (2021a)	6.2	**7.1**	**7.4**
RMT+top-k	**7.8**	7.0	7.2

Table 4: Human evaluation results of sentiment control.

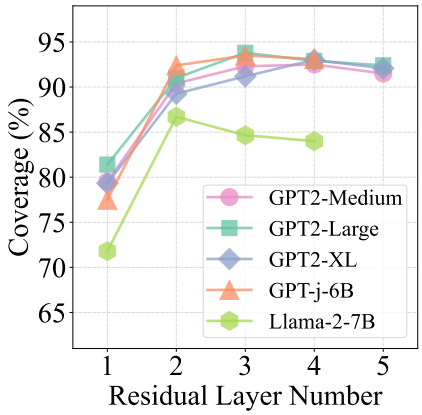

Figure 4: RMT block layers across different base-CLM and control performance.

providing a greater flexibility and a more powerful control capability.

The human evaluation presented in Table 4 indicates that DEXPERT performs slightly better in terms of fluency and contextual relevance, but exhibits lower control ability, and RMT excels in attribute relevance. We speculate that the use of a classifier-based objective in RMT and DisCup might result in overly strong control, leading to the generation of abrupt turning points that may compromise fluency and topicality to some extent. DisCup performs relatively poorly, as too few control parameters limit its expression ability. ***More detailed examples are presented in Appendix E.***

3.4 Further Analysis

Block Layers and different base-CLMs. we investigate the influence of the number of block layers on the RMT's control performance over different base-CLMs, i.e., GPT2, GPT-j-6b (Wang and Komatsuzaki, 2021) and Llama2-7b (Touvron et al., 2023). The results shown in Figure 4 indicate that a value of $M = 3$ is deemed appropriate for achieving effective word inclusion and sentence length control. Moreover, RMT achieves a consistently high and stable control performance for different CLMs, showing its model-agnostic nature and adaptability across different CLMs.

Ablation Study. To assess the effectiveness of the Residual Memory Transformer (RMT), we conduct an ablation study. The study involves three settings: removing the causal self-attention, excluding the causal CLM attention, and evaluating RMT's performance without pre-training. The results are presented in Table 6. Interestingly, under all three settings, we observe a significant decline in performance. This finding serves as compelling evidence that every component in RMT, as well as the pre-training stage, is crucial and indispensable for achieving optimal results.

Generation Efficiency. RMT is a lightweight plugin, allowing for efficient inference speeds comparable to the original CLM. Table 5 demonstrates that RMT outperforms typical CTG approaches in term of efficient generation speed, approaching the speed of the pure CLM (GPT2-large).

Method	Time (second)
PPLM (Dathathri et al., 2020)	37.39
Mix-Match* (Mireshghallah et al., 2022)	33.5
COLD* (Qin et al., 2022)	33.6
DEXPERT (Liu et al., 2021a)	2.54
NRP (Carlsson et al., 2022)	2.93
DisCup (Zhang and Song, 2022)	0.94
GPT2-large	0.78
RMT	0.88

Table 5: The generation efficiency of RMT. The cost (time) for generating 20 tokens based on GPT2-large. The results of methods marked with $*$ are reported from (Qin et al., 2022).

Models	Cov (%)
RMT	**93.8**
w/o causal Self-Att	73.2
w/o causal CLM-Att	75.1
w/o Pre-training	74.5

Table 6: Ablation study results of RMT.

4 Related Work

As summarized by (Zhang et al., 2022), CTG approaches could be divided into three categories, i.e., retraining/refactoring, fine-tuning, and post-process. The first two categories refer to the training approaches, which aim to fine-tune (e.g., reinforcement learning is used to fine-tune pre-trained language model(PLM) (Ouyang et al., 2022; Lu et al., 2022a), optimizing continuous prompts (Yang et al., 2022; Zhang and Song, 2022), and prompting model (Carlsson et al., 2022) to steer text generation) or retrain/refactor (Chan et al., 2021; Yu et al., 2021; Keskar et al., 2019; Gururangan et al., 2020) a PLM to generate texts that meet the desired control condition. These methods have shown significant performance improvements in the field. However, with the increasing size of PLM, they have become resource-intensive to fine-tune or retrain. As a result, post-process approaches have become more popular in the research community.

The post-process methods are dedicated to guiding language model toward desired texts in the decode-time stage using an auxiliary module. Achieving the goal of generating attribute-specific texts, PPLM (Dathathri et al., 2020) leverages a simple attribute classifier to update the head hidden layer of LM by gradient feedback. Then, Diffusion-LM (Li et al., 2022) combines Diffusion-LM and classifier-guided gradient updates to the continuous sequence of latent variables, achieving plug-and-play controllable generation. COLD (Qin et al., 2022) proposes a novel decoding framework, by combining energy-based model and gradient-based sampling. Fudge (Yang and Klein, 2021) uses the discriminator that contains future attribute information to re-rank the token distribution produced by a frozen GPT2 model. NeuroLogits (Lu et al., 2022b, 2021) incorporates the lexical constraints into the decode-time algorithm. In order to accelerate the generation process, GeDi (Krause et al., 2021) and DEXPERT (Liu et al., 2021a) train another smaller language model as generative discriminators to guide the generation from a base CLM. Recently, more subtle and efficient decoding approaches (Zhong et al., 2023; Deng and Raffel, 2023) are also explored to achieve the better controllability and text quality. Moreover, controlling multiple control elements is also expolred (Kumar et al., 2021; Yang et al., 2022).

Most decoder-time CTG either solely intervene with the LM during the token selection phase of generation, lacking planning capabilities, or require multiple iterations, resulting in excessive generation times. RMT shares a plug-and-play trait with decoder-time approaches, while the key distinction is that we use a lightweight residual model to integrates control information and multi-level contextual streams from the CLM, enabling fine-grained content planning and efficient text generation.

5 Conclusions

We have proposed a new CTG approach, which leverages a residual model to steer the CLM to generate desired text noninvasively. Additionally, we propose a residual memory transformer, a novel encoder-decoder architecture, to fuse the raw contextual text, the generative stream of CLM, and the control information in one shot, thus better collaborating and controlling the text generation. Experimental results show that RMT exhibits a better performance in flexibility, control granularity and efficiency over a range of state-of-the-art baselines, making it a compelling solution for CTG.

Acknowledgments

This work is funded in part by the Natural Science Foundation of China (grant no: 62376027) and Beijing Municipal Natural Science Foundation (grant no: 4222036 and IS23061).

Limitations

RMT still suffers from the following limitations. (1) Challenges in applying RMT to close-ended CLMs. Presently, the application of RMT needs to obtain the last hidden states or the logits of CLM, thus directly applying RMT to certain commercial CLMs, e,g., GPT-4, is a problem that needs further investigation. This is indeed a common problem of all plugin-style CTG approaches. (2) RMT does not focus on commonsense and may result in the generation of texts that do not conform to commonsense. This issue could be relieved in the future by incorporating some external knowledge graphs.

Up to now, it is still challenging for large-scale causal language models to avoid the factual errors in the generated text, which RMT does not address either. A promising future work is to combine RMT and information retrieval techniques to enhance the factual accuracy of those generative models. Moreover, RMT could also be used to encode personal profiles and build personalized chatbots, or fusion with image information so as to be applied to multimodal scenes.

References

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Fredrik Carlsson, Joey Öhman, Fangyu Liu, Severine Verlinden, Joakim Nivre, and Magnus Sahlgren. 2022. Fine-grained controllable text generation using non-residual prompting. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6837–6857, Dublin, Ireland. Association for Computational Linguistics.

Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. 2021. Cocon: A self-supervised approach for controlled text generation. In *International Conference on Learning Representations*.

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

Haikang Deng and Colin Raffel. 2023. Reward-augmented decoding: Efficient controlled text generation with a unidirectional reward model. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 11781–11791, Singapore. Association for Computational Linguistics.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020. Don't stop pretraining: Adapt language models to domains and tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8342–8360, Online. Association for Computational Linguistics.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society.

Zhiting Hu and Li Erran Li. 2021. A causal lens for controllable text generation. In *Advances in Neural Information Processing Systems*, volume 34, pages 24941–24955. Curran Associates, Inc.

Nitish Shirish Keskar, Bryan McCann, Lav Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL - A Conditional Transformer Language Model for Controllable Generation. *arXiv preprint arXiv:1909.05858*.

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, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Sachin Kumar, Eric Malmi, Aliaksei Severyn, and Yulia Tsvetkov. 2021. Controlled text generation as continuous optimization with multiple constraints. In *Advances in Neural Information Processing Systems*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances*

in Neural Information Processing Systems, 35:4328–4343.

Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. 2020. CommonGen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, Online. Association for Computational Linguistics.

Alisa Liu, Maarten Sap, Ximing Lu, Swabha Swayamdipta, Chandra Bhagavatula, Noah A. Smith, and Yejin Choi. 2021a. DExperts: Decoding-time controlled text generation with experts and anti-experts. 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 6691–6706, Online. Association for Computational Linguistics.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and S Yu Philip. 2021b. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6418–6425.

Ximing Lu, Sean Welleck, Jack Hessel, Liwei Jiang, Lianhui Qin, Peter West, Prithviraj Ammanabrolu, and Yejin Choi. 2022a. Quark: Controllable text generation with reinforced unlearning. *Advances in neural information processing systems*, 35:27591–27609.

Ximing Lu, Sean Welleck, Peter West, Liwei Jiang, Jungo Kasai, Daniel Khashabi, Ronan Le Bras, Lianhui Qin, Youngjae Yu, Rowan Zellers, Noah A. Smith, and Yejin Choi. 2022b. NeuroLogic a*esque decoding: Constrained text generation with lookahead heuristics. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 780–799, Seattle, United States. Association for Computational Linguistics.

Ximing Lu, Peter West, Rowan Zellers, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. NeuroLogic decoding: (un)supervised neural text generation with predicate logic constraints. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4288–4299, Online. 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.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Jing Qian, Li Dong, Yelong Shen, Furu Wei, and Weizhu Chen. 2022. 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. *arXiv preprint arXiv:2202.11705*.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, and et al. Nikolay Bashlykov. 2023. Llama 2: Open foundation and fine-tuned chat models.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Ben Wang and Aran Komatsuzaki. 2021. GPT-J-6B: A 6 Billion Parameter Autoregressive Language Model. `https://github.com/kingoflolz/mesh-transformer-jax`.

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. 2022. Tailor: A prompt-based approach to attribute-based controlled text generation. *CoRR*, abs/2204.13362.

Dian Yu, Zhou Yu, and Kenji Sagae. 2021. Attribute alignment: Controlling text generation from pre-trained language models. In *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2251–2268. Association for Computational Linguistics.

Hanqing Zhang and Dawei Song. 2022. DisCup: Discriminator cooperative unlikelihood prompt-tuning

for controllable text generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3392–3406, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models.

Jeffrey O Zhang, Alexander Sax, Amir Zamir, Leonidas Guibas, and Jitendra Malik. 2020a. Side-tuning: a baseline for network adaptation via additive side networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pages 698–714. Springer.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020b. POINTER: Constrained progressive text generation via insertion-based generative pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8649–8670, Online. Association for Computational Linguistics.

Tianqi Zhong, Quan Wang, Jingxuan Han, Yongdong Zhang, and Zhendong Mao. 2023. Air-decoding: Attribute distribution reconstruction for decoding-time controllable text generation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 8233–8248, Singapore. Association for Computational Linguistics.

Wangchunshu Zhou, Yuchen Eleanor Jiang, Ethan Wilcox, Ryan Cotterell, and Mrinmaya Sachan. 2023. 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.

Xu Zou, Da Yin, Qingyang Zhong, Hongxia Yang, Zhilin Yang, and Jie Tang. 2021. Controllable generation from pre-trained language models via inverse prompting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, page 2450–2460, New York, NY, USA. Association for Computing Machinery.

Appendices

In order to enhance the reproducibility of the experimental results presented in our paper and provide additional supporting details for the conclusions outlined on the main page, we have included supplementary materials in this section.

A Preliminary

A.1 Attention in Transformer

Self-Attention mechanism is a critical component in Transformer (Vaswani et al., 2017), which is used to effectively capture long-range dependencies between words in the input sequence. Specifically, it is defined as a mapping between a query and a collection of key-value pairs. The values are weighted according to a compatibility function between the query and each corresponding key, and then summed up, eventually obtaining the output vector. The self-attention mechanism can be formatted as follows:

$$\mathrm{Att}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (6)$$

where Q, K, V represent the transformed representations of the sequence using the corresponding learnable matrix, and $\sqrt{d_k}$ is the scaling factor. In the Transformer model, the key, query and value vectors are always derived from a shared sequence.

Causal Attention is a particular branch of self-attention, also called masked self-attention, which is usually applied in the decoder module of Transformer. Different from normal self-attention, the queries in causal attention are only confined to their preceding key-value pairs' positions and their current position to maintain the auto-regressive property. Usually, it can be implemented by a mechanism that masks the invalid positions and sets them to negative infinite:

$$\mathrm{Att}(Q, K, V) = \mathrm{softmax}\left(\frac{QAK^T}{\sqrt{d_k}}\right)V, \quad (7)$$

$$A_{ij} = \begin{cases} 1 \text{ if } i \geq j, \\ -\infty \text{ else.} \end{cases} \quad (8)$$

Cross Attention is another branch of self-attention in the decoder part of Transformer, which aims to capture the interaction between the encoder and decoder. Specifically, The key and value parts are obtained from the previous step outputs of the

encoder, and the query is from the decoder, enabling the encoder to attend to each position in the decoder module at the same time.

A.2 Causal Language Model

Causal Language Model (CLM) eliminates the need for a separate encoder component and typically leverages a decoder-only Transformer (Vaswani et al., 2017), in which only the causal attention is used to perform step-wise density estimation, i.e., predicting the next token. Suppose a CLM is parameterized with θ. Given a partial sequence $x_{<t}$, it assigns a probability $P_\theta(x_t|x_{<t})$ over a vocabulary $\mathcal{V}$ for next-token x_t generation. When generating a sequence of text $X_n = \{x_1, x_2, \ldots, x_n\}$, it can be formulated by the chains rule as below:

$$P_\theta\left(X_n\right) = \prod_{t=1}^{n} P_\theta\left(x_t \mid x_{<t}\right). \tag{9}$$

The entire generation process is carried out iteratively. First, a token is sampled from the distribution $P_\theta\left(x_t \mid x_{<t}\right)$. Then the selected token is concatenated with the input for the next step of generation.

A.3 Model Training with MLE

An encoder-decoder generation model is usually trained with Maximum Likelihood Estimation (MLE). Given a finite set of training data $\mathcal{D}$ consisting of a target and source data per sample, and a language model with parameters θ, the optimized object is defined as follows:

$$\mathcal{L}\left(\theta, \mathcal{D}\right) = -\sum_{i=1}^{|\mathcal{D}|} \sum_{t=1}^{|\mathbf{x}^{(i)}|} \log p_\theta \left(x_t^{(i)} \mid x_{<t}^{(i)}, S^{(i)}\right), \tag{10}$$

where $|\mathcal{D}|$, $|\mathbf{x}^{(i)}|$ represents the number of samples in dataset and the length of a target sequence respectively. $S^{(i)}$ represents the corresponding source data encoded by encoder of RMT. $x_t^{(i)}$ is the next-token of $x_{<t}^{(i)}$, and $x_{1:t}^{(i)}$ is a partial sequence truncated from target training sample. The parameters θ is updated by maximizing the log likelihood (i.e., probability of decoder's next-token prediction).

B Experimental Details

In our paper, all the experiments are conducted on a single NVIDIA A6000 GPU with 48GB of memory. We utilize the PyTorch deep learning framework as our implementation tool and employ the HuggingFace Transformers package to load a pre-trained GPT model. To implement the Residual Memory Transformer (RMT), we utilize the Transformer package in PyTorch-1.14.0, and the number of multi-head attention is set to 8. The post-normalization is applied to achieve a better performance of fine-tuning.

In the implementation of RMT for Llama2 and GPT-6-J, we replaced the Causal-Self-Attention and Causal-CLM-Attention sublayer with LlamaFlashAttention to add the position information, and Cross-Control-Attention are same to that of GPT2. As for the encoder of RMT, we also add the Rotary Position Embedding to the bidirectional self-attention.

B.1 Pre-training

Data Details. The pre-training stage is consistently applied across all experimental settings in our study. For this stage, we collect a dataset of 4 million sentences sampled from the Wikipedia corpus to serve as the training data. To optimize computational resources, we filtered out sentences that exceeded a length of 200 words. These samples are then noised using the following methods:

(1) Token Masking: We randomly mask tokens with a special token. The proportion of masked tokens in a sample is 20%.

(2) Token Deletion: We randomly delete tokens from the text. The proportion of deleted tokens in a sample is 20%.

(3) Span Replacing: Text spans of various lengths are randomly replaced with special tokens. Each span is replaced with a single special token. The proportion of replaced span tokens in a sample is 20%.

(4) Text Rotation: A token is randomly selected, and the text is rotated around it. This means the text began with the selected token and ended with its previous token.

In the total dataset, the ratios of *Token Masking*, *Token Deletion*, *Span Replacing*, and *Text Rotation* are 50%, 16.7%, 16.7%, and 16.7%, respectively. Since RMT shares GPT's tokenizer, which does not reserve a special token, we select the word "xxx" with no special meaning as the special token. we did not specifically tune the hyperparameters related to the ratios of different denoised tasks during

pretraining. Instead, we followed BART's [12] approach and selected the most important tasks, such as Text Infilling (included by Token Masking in RMT), to constitute the largest proportion of the task. The remaining tasks were divided equally among the other three tasks. In our experiments, we observed that the specific choice of pretraining tasks did not significantly impact the results in CTG. For instance, when we used Token Masking as the only pretraining task, the control performance on CommonGen reached 92.2%. By adding the other three tasks, the performance improved slightly to 93.9%.

Training Details. During the pre-training process, we set the following parameters for model training: the batch size of 64, learning rate of 5e-5, random seed of 42, and pre-training epoch of 1. We employ the AdamW as the optimizer; the pre-training of the RMT is conducted on a single NVIDIA A6000 GPU. Since the backpropagation process does not need to propagate into the Casual Language Model (CLM), the training of the model was efficient. As a result, the entire pre-training stage with GPT2-large is completed in approximately 30 hours.

B.2 Word Inclusion

Baselines. We closely follow the methodology of previous work, specifically the Non-Residual Prompt (NRP) (Carlsson et al., 2022). To establish a reliable reference point for the CommonGen dataset, we rely on the generation results of the baselines provided in the NRP paper. And for the NeuroLogic baseline, in order to keep the comparability, we use the same decoding algorithm setting with RMT, the beam search size is 4 and max generation length is set to 32. For the C2Gen dataset, we utilize the checkpoint provided by NRP, to generate the responding texts. However, the result shows NRP could not work well. As a result, we give up using NRP as a baseline in the C2Gen dataset.

Regarding the ChatGPT baseline, we employ prompts specifically designed for generating results on the CommonGen dataset. The specific prompt is as follows:

Continue to write a sentence, and include [$word_1$, $word_2$, ..., $word_n$], the sentence length should be within 15 words.

When it comes to the C2Gen dataset, the prompt instruction of ChatGPT is:

Continue to write a sentence, and include [$word_1$, $word_2$, ..., $word_n$], the sentence length should be within 15 words. Context: [context text].

Name	Value
num_beams	4
top_p	0.7
repetition_penalty	1.25
no_repeat_ngram_size	3

Table 7: Hyper-parameters for text generation in the inference stage.

Details of RMT. In our approach, we randomly select training data from the Wikipedia corpus and fuse the training data of CommonGen. Specifically, for the setting involving without-context word inclusion and sentence length control, we gather a total of 158,343 samples. To ensure manageable sentence lengths, we impose a limit of 32 tokens per sentence. To choose the appropriate checkpoint during training, we utilize the CommonGen validation dataset and selected the checkpoint based on coverage metrics. For the with-context setting, we collect 132,170 samples, each with a sentence length constraint of 128 tokens, and constructed a validation dataset to choose the checkpoint.

Every sample is composed of two parts: context and target. We use spacy library[5] to extract the keywords in the target text, of which are tagged with `VERB`, `NOUN`. The number of keywords in each sentence is restricted to the range from 3 to 5. The context length, target length, and keywords are used as control instructions to steer the generation process of the CLM. All above training data will be released and open source, once we have cleaned up our project.

During the fine-tuning stage, the learning rate of the AdamW optimizer is set to 1e-4, and the max epoch and batch size are 6 and 64, respectively. During the inference stage, For the control task without context (i.e., CommonGen dataset), we use the specific word "The" as context; the core parameters of the decoding algorithm are shown in Table 7.

B.3 Sentiment Control

Baselines. In this part, we build upon the baselines established by DisCup (Zhang and Song, 2022) and DEXPERT (Liu et al., 2021a), adopting their settings, checkpoints, and generated results. The results of baselines, including PPLM (Dathathri et al.,

[5] https://spacy.io/

2020), CTRL (Keskar et al., 2019), GEDI (Krause et al., 2021), and DEXPERT, are calculated from the generated texts provided DEXPERT (Liu et al., 2021a). As for DisCup, we use the checkpoint from the authors, the depth of re-ranked candidate token is set to top-k=90 to generate continuations using the prompts in test datasets.

Details of RMT. In our own approach, during the training stage, we incorporate the discriminator-involved loss introduced by DisCup and utilize the attribute classifier provided by their open-source code. During the training, we optimize RMT using two different strategies, one is to use top-k=90 algorithm to select re-rank candidate tokens and the other setting uses top-p=0.95 to select candidate tokens. The top-k setting increases the depth of token sampling, thus the diversity and control ability of the model will increase, yet with the burden of decreasing text fluency. We choose a low temperature value of 0.005 and a learning rate of 2e-4. The top-p keeps the re-ranked tokens within the distribution of base-CLM, which could potentially achieve higher fluency but lower text quality and control ability. The instructions for sentiment control are "*The sentiment: positive*" and "*The sentiment: negative*", respectively. During the inference stage, we employ the same generation parameters as those used in the word inclusion experiments described in Table 7.

C CTG Approaches Comparison

C.1 CTG Comparison

We select some notable works in controllable text generation (CTG) and compare their characteristics. The results are shown in Table 8. We compare them according to three key features: ***(1) Fine-grained Control*** suggests whether the method can potentially be applied to control various elements, such as emotion, topic, toxicity avoidance, word constraints, and structural control (e.g., personal profile, syntax tree). It indicates the method's ability to handle different control aspects effectively. ***(2) Efficient Generation*** evaluates whether the time-cost of text generation using the method is comparable to that of the original language model (LM). ***(3) Flexibility*** mainly considers two aspects. The first is the non-intrusive fashion, i.e, the method can operate without disturbing the language model's original textual stream, mitigating the harms of original LMs. The second requires that control-effect is distance-independent and enables long-distance control throughout the generation process. Flexibility is a very important trait in the LLM era. Specifically, the non-intrusive fashion of RMT decouples the control module from the base-CLM and the gradient does not need backpropagation into CLM with large-scale parameters, bringing significant training efficiency. Secondly, we no longer need to fine-tune the base CLM for specific control tasks, avoiding any adverse impact on its generalization ability (i.e., keeping the CLM from becoming task-specific model). Moreover, this design enables us to dynamically adjust the fine-grained control requirements during the text generation process at any time. For instance, we can enhance a well-trained CLM, such as ChatGPT, with external grounded documents, making it possible to determine whether external control information should be intervened through an automatic selection model (i.e., switching b/w controlled and free-style generation).

The above analysis suggests that controlling the generation process of LMs while balancing control flexibility, control granularity, and generating efficiency remains an open challenge. RMT shows potential in addressing these issues, making it a compelling solution for controllable text generation.

C.2 RMT v.s NRP

The most similar to RMT is Non-Residual Prompt (NRP) approach(Carlsson et al., 2022), which also requires pretraining and fine-tuning. Therefore, we make a thorough discussion of differences of RMT w.r.t NRP.

Principle reasonability. The principle of RMT appears intuitively more reasonable. RMT utilizes cross-attention to uniformly apply control conditions to each generated token, alleviating the "distance bias" problem that arises with increasing distance from the control prompts. Intuitively, NRP still uses distance-aware causal attention to encode control conditions, and seems inherently difficult to fully overcome the "distance bias" issue. In contrast, cross-attention in RMT enables the final representation of the current generation step to look up the control condition position-independently, which is more intuitively appealing.

Training Paradigm. The pretraining stage of RMT is general and task-independent, relying solely on self-supervised training (e.g., utilizing Wikipedia data). Various CTG tasks can be fine-tuned using task-specific corpora without the need

CTG Approach	Fine-grained Control	Efficient Generation	Flexibility
PPLM (Dathathri et al., 2020)	✘	✘	✔
CTRL (Keskar et al., 2019)	✔	✔	✘
NeuroLogit (Lu et al., 2022b)	✘	✘	✔
GEDI (Krause et al., 2021)	✘	✔	✔
DEXPERT (Liu et al., 2021a)	✘	✔	✔
FUDGE (Yang and Klein, 2021)	✘	✘	✔
COLD (Qin et al., 2022)	✔	✘	✔
Diffsion-LM (Li et al., 2022)	✔	✘	✔
Mix&Match (Mireshghallah et al., 2022)	✘	✘	✔
CoCon (Chan et al., 2021)	✔	✔	✘
DisCup (Zhang and Song, 2022)	✘	✔	✘
NRP (Carlsson et al., 2022)	✔	✘	✔
RMT(ours)	✔	✔	✔

Table 8: An overview of the key features in CTG approaches.

for additional data reconstruction. However, pretraining and finetuning stage of RMT is coupled and interrelated, limiting the applicability to various CTG tasks.

Training efficiency. RMT holds a clear advantage over NRP in computational source cost. RMT efficiently reuses the output of the base CLM, requiring fewer pretraining training data and expediting the pretraining process. Additionally, RMT is lightweight and is built on the top layer of the base-CLM. Hence, the backpropagation process does not need to propagate into the CLM, which will save a huge training time and GPU memory. Conversely, NRP, despite being initialized from the base CLM, necessitates training a separate base-CLM and involves multi-stage pretraining, leading to higher computational costs. As discussed in Appendix B.1, RMT's pre-training was conducted on a single NVIDIA A6000 GPU, completed in approximately 30 hours. In contrast, according to the report in the paper on NRP, they incurred approximately 400 GPU hours (using DGX-100 machines) for their training process.

Parameter efficiency. RMT is more parameter-efficient compared to NRP. NRP requires training additional base-CLM, whereas RMT only necessitates approximately 1/6 of the base-CLM's parameters.

D Human Evaluation

We conduct the human evaluation for the word inclusion and sentiment control tasks. To ensure the reliability of the evaluation, we invite three highly educated experts as evaluators, and each evaluator has expertise in English. For the evaluation process, we establish a unified criterion using a scale of 10 for each aspect. The scale ranges from 1 to 3 for poor performance, 4 to 7 for satisfactory performance, and 8 to 10 for excellent performance. This criterion allows evaluators to provide consistent and comparable scores for each aspect.

Regarding the CommonGen dataset and C2Gen, we randomly select 25 samples from test dataset for each CTG approach, resulting in a total of 25*7 =175 samples for CommonGen and 50 samples for C2Gen. For the sentiment control experiments, we create 10 positive-steering prompts composed of neutral and positive prompts, as well as 10 negative-steering prompts composed of neutral and negative prompts. Each prompt is evaluated using three CTG approaches, resulting in a total of 60 samples for the final human evaluation. To obtain the final results, we calculate the average score for each metric for every CTG method under comparison, based on the evaluations provided by the experts.

E Qualitative Examples

In this section, we list some examples generated by RMT and other different CTG approaches. Specifically, Table 9 shows the cases on the CommonGen dataset, and Table 10 presents the cases in the C2Gen dataset. As for attribute control, we show the examples on the adversarial steering setting (i.e., the polarity of the prompt's sentiment is opposite from the target sentiment) as shown in table 11.

Target Words: swim, pool, reach, end	
POINT (Zhang et al., 2020b)	water at the south end of the main swimming pool allows one to reach to the top where they can then swim to the main pool.
NeuroLogic (Lu et al., 2022b)	The end of the world is coming, and it will be swimming in blood pools of blood, and the blood will be so thick that it will reach the sky
NRP (Carlsson et al., 2022)	The swim pool at the end of the pool can be used to reach the other side of the building.
ChatGPT	I can't wait to swim in the pool and reach the end.
RMT (ours)	The swimming pool has reached its maximum size and is at the end of its life-spans.
Target words: look, heart, writer, leave, goer	
POINT (Zhang et al., 2020b)	i goer in my heart, but you will not leave me alone, do you look like a writer?.
KG-BART (Liu et al., 2021b)	a group of people looking at each other and leaving their hearts in the shape of a heart with a writer
NeuroLogic (Lu et al., 2022b)	The first thing I did was look at the goers and the heart rate, and I left the room and went to the writers room.
NRP (Carlsson et al., 2022)	The writer-producer would leave the show to look for a new job as a goer.
ChatGPT	The writer's heart sank as they watched the goer leave without a second look.
RMT (ours)	The writer leaves a heart-shaped letter for the goers to look at on the red carpet.

Table 9: The examples generated by different CTG approaches on the CommonGen dataset (the without-context setting).

	Target Words & Context
1	**Target words:** enjoy#friend#knit **Context:** Friendship is a relationship of mutual affection between people. It is a stronger form of interpersonal bond than an association, and has been studied in various academic fields. Such as communication, sociology, social psychology, anthropology, and philosophy.
2	**Target Words:** background#crowd#pose **Context:** I love photography. One of my favorite things to do is to take pictures of people modeling throughout the city. The beach is one of my favorite places because of the scenery and the atmosphere there.
3	**Target Words:** cigarette#sstreet#walk **Context:** James mulled over the fight with his girlfriend. The city noises were all blocked out by him. He dropped his lighter and then picked it up again.
	ChatGPT
1	My friend and I enjoy spending time together, often knitting and chatting.
2	The background of the beach makes for a stunning setting for a crowd pose.
3	As he lit his cigarette, he decided to take a walk down the street.
	RMT
1	Friendship can be enjoyed by both men and women and it is best knit between two close friends.
2	It is so much fun to pose in the background with a crowd of people.
3	As he walked up to the street, his cigarettes were gone from his mouth.

Table 10: The examples generated by different CTG approaches on the C2Gen dataset (the with-context setting).

Negative Prompt: *The real problem, though, is that we don* (**Positive Steering**)	
PPLM (Dathathri et al., 2020)	*The real problem, though, is that we don*'t have a sensible way of using them to share meta-information. The investigation, no evidenced-
CTRL (Keskar et al., 2019)	*The real problem, though, is that we don*'t know how long the product has been in use. It should have been tested for freshness and potency befor
GEDI (Krause et al., 2021)	*The real problem, though, is that we don*'t teach children all of these tools. And so what can parents do to help them acquire all this
DEXPERT (Liu et al., 2021a)	*The real problem, though, is that we don*'t know the full extent of what social causes contribute to obesity and other related health conditions, says study
DisCup (Zhang and Song, 2022)	*The real problem, though, is that we don*'t know what to do with our children but that's OK too because the world has to grow but
RMT (Ours)	*The real problem, though, is that we don*'t question whether 'good' or 'bad' will be better but keep putting our faith into not
Positive Prompt: *A President unafraid to stand* (**Negative Steering**)	
PPLM (Dathathri et al., 2020)	*A President unafraid to stand* out: No matter what the circumstances, Trump will never be the first president to fight one
CTRL (Keskar et al., 2019)	*A President unafraid to stand* up for what he believes in. This one could be added to my library. Rating: 2.0 I
GEDI (Krause et al., 2021)	*A President unafraid to stand* by NRA shameless hypocrites tweeted something totally me aningless that called a black woman a 'rapist'. Neither hurtful
DEXPERT (Liu et al., 2021a)	*A President unafraid to stand* on a bomb, is he?That was bad enough to threaten the security of every armed
DisCup (Zhang and Song, 2022)	*A President unafraid to stand* alone? I'm not a good person or good for a President to be? And even if I am
RMT (ours)	*A President unafraid to stand* up against corruption? Unfortunately, there appears little hope right now of getting any meaningful changes. Since Congress

Table 11: The generation examples of typical CTG approaches in the sentiment control task. We deliberately select two challenging prompts, and ask the above CTG approaches to conduct the adversarial steering of sentiment polarity.