An Efficient Post-hoc Framework for Reducing Task Discrepancy of Text Encoders for Composed Image Retrieval

Supplementary Material

A. Additional Implementation Details

A.1. CIR Datasets

FashionIQ [28] is a dataset that contains fashion-related images from three main categories: Shirts, Dresses, and Toptee. It has a total of 30,134 triplets, which were created from 77,684 images. As the ground truth labels are not publicly available, we utilize the results from the validation set for our analysis and comparison. CIRR [15] encompasses a wider range of domains and contains images with more complex descriptions compared to FashionIQ. It contains 36,554 triplets extracted from 21,552 images, which are sourced from the well-known NLVR2 natural language inference dataset [23]. As pointed out in previous works [3, 9, 20], CIRR and FashionIQ suffer from a significant number of false negatives, which can potentially lead to inaccurate retrieval evaluations [3, 20]. To address this issue, CIRCO [3], based on COCO images [14], is recently introduced by providing multiple positive images for each query. This approach enables a more reliable and robust mAP metric [5, 16], which is essential for accurate evaluation of retrieval performance.

We additionally provide results on two more benchmark datasets, GeneCIS [26] and COCO Object Composition introduced by [20], in Suppl. B.1. GeneCIS [26] is constructed based on COCO images and the Visual Attributes in the Wild dataset [17]. GeneCIS introduces four task variations: (1) focus on an attribute, (2) change an attribute (3) focus on an object, and (4) change an object. These tasks explore different aspects of image retrieval and manipulation. For the COCO Object Composition task, 5000 images from the COCO validation dataset are utilized to evaluate object composition. Our objective is to retrieve an image that contains an object specified by a query image, along with scenes or objects described using text. The composed query is constructed by combining "a photo of [$], [$obj1$], [$obj2$] ... and [$objn$]" where [$obji$] are text descriptions.

A.2. Details of text triplets

Here, we describe the details of the LLM-based and rule-based text triplet generation process. As shown in Figs. 1 and 2, which showcases examples of both LLM-based and rule-based triplets, both approaches produce natural and coherent text triplets. Note that none of the datasets used for generating text triplets overlap with the data used in the target CIR benchmarks, with the exception of the CASE dataset [12]. The source of the CASE dataset is VQA2.0 [7], which is constructed from the COCO dataset [14], potentially leading to overlap in cases involving COCO object composition [20].

[Detailed explanation on LLM-based triplets] As described in Sec. 3, besides Compodiff [8], we conduct experiments using various publicly available text triplets: IP2P [4], COVR [27], and CASE [12]. Although the primary objective of these approaches is to generate CIR triplets (I_r, T_c, I_t), they also produce text triplets. Below, we provide detailed descriptions of how text triplets are constructed in each approach (Note again that their final product is CIR triplets). There are two main ways to generate text triplets using LLMs: 1) generating both conditional text T_c and target caption T_t given reference caption T_r using fine-tuned LLM for this task, such as IP2P, Compodiff; and 2) generating only conditional text T_t given pairs (T_r, T_t) from pre-existing captions by identifying with visually or text semantically similar such as CoVR [27], and CASE [12]. In addition to these existing datasets, we implement an efficient in-context learning-based generation process. Examples and summaries of each dataset can be found in Tab. S1 and Tab. S2.

IP2P employs GPT-3 for text triplets generation and fine-tunes it with a human-curated small set of 700 text triplets. Namely, given reference captions T_r sampled from LAION-Aesthetics V2 6.5+ dataset [21], the corresponding conditional texts T_c and corresponding target captions T_t are manually written by humans. After fine-tuning on this small set of text triplets, the model generates 454k text triplets: reference captions T_r from the LAION-Aesthetics V2 6.5+ dataset [21] are provided as input to the fine-tuned LLM, whose output predicts the corresponding conditioning text T_c and the target caption T_t. Note that the LAION-Aesthetics dataset is not related to the original source datasets (FashionIQ, NLVR2, and MS-COCO) used in existing CIR benchmarks (FashionIQ, CIRR, and CIRCO), ensuring no overlap with the CIR benchmarks.

Compodiff enhances the scalability of the IP2P text triplet generation process by modifying the choice of LLM and expanding the fine-tuning dataset. As described in [8, Section 4], the OPT-6.7B model is utilized and fine-tuned with LoRA on the above 454k text triplets of IP2P [4]. Then, similar to the IP2P approach, given reference captions from the LAION dataset [21], fine-tuned LLM generates the corresponding conditioning texts and target captions.

COVR starts by identifying similar caption pairs from the WebVid2M dataset [1], which contains 2.5 million video-caption pairs. These pairs serve as the reference captions (T_r) and target captions (T_t). Then, given these pairs (T_r, T_t), LLM generates conditional captions that describe the differences between the paired captions. The LLaMA-7B model [25] is utilized for this purpose and is fine-tuned on an expanded version of the above 700 manually annotated triplets used in IP2P (adding 15 annotations for more diverse cases).

CASE uses VQA2.0 dataset [7], which consists of (image, question, answer) triplets. Given (I, Q, A) triplets, complementary triplets (I_c, Q, A_c) are manually selected based on visually similar image I_c with three rules: 1) the premise assumed in question Q holds for both I and I_c, 2) Q is logical for I_c, and 3) the answer A_c for I_c differs from A. Then, conditional text T_c is generated by GPT-3, describing differences between image pairs (I, I_c) without fine-tuning, leveraging in-context learning with a few examples. Since the VQA2.0 dataset is derived from the COCO dataset, COCO captions that match VQA2.0 images are used to form text triplets.

As seen in Tab. S1, compared to other approaches, the quality of the text triplets (T_r, T_c, and T_t) from CASE is not always satisfactory, which results in minimal performance gain as shown in Tab. 6. Namely, unlike other CIR datasets that first create high-quality text triplets before generating CIR triplets, CASE generates the conditioning text T_c using the reference image I_r and target image I_t. The provided reference text T_r and target text T_t are taken directly from the captions of reference image I_r and target image I_t in the VQA2.0 dataset. Therefore, due to the poor descriptiveness of these captions and their lack of consideration for the conditioning text T_c, it often fails to capture the differences between T_r and T_t adequately (while T_c might effectively explain the visual differences between I_r and I_t).

Efficient in-context learning refers to our efficient implementation which uses a recent and powerful LLM, LLaMA3-8B [6]. This approach performs in-context learning using reference captions T_r from the CC3M dataset [22], guided by a custom-designed prompt with a few examples of textual modifications (*e.g.*, replace, change, remove, ...). Specifically, given a reference caption T_r, the prompt instructs the model to generate a target caption T_t, which is a complete sentence that slightly differs from the corresponding reference caption. Then, the prompt guides the model to generate conditioning text that explains the differences between T_r and T_t, based on the above pre-defined textual modifications. Compared to Compodiff, which takes 3.8 hours to generate 1 million text triplets, this version requires only 1.5 hours. In Tab. 6, we verify that this more efficient version achieves competitive performance compared to the other fine-tuned LLM approaches.

Table S1. **Examples of text triplet datasets.**

Text triplets	**Reference text T_r**	**Conditioning text T_c**	**Target text T_t**
Rule-based	"another wall at my home"	"bedroom is added in place of wall"	"another bedroom at my home"
IP2P [4]	"watercolor of your pet!"	"make it a huge grizzly bear"	"watercolor of a huge grizzly bear!"
Compodiff [8]	"Chinese landscape watercolor painting"	"make the landscape a cityscape"	"chinese cityscape watercolor painting"
Efficient in-context learning	"young business woman on a bench"	"add a laptop"	"young business woman on a bench with a laptop"
CoVR [27]	"Two little boys are running"	"Have them dance"	"Two little boys are dancing"
CASE [12]	"A scone with an orange slice on a plate"	"This food is not acidic"	"a close up of a muffin on a plate on a table"

Table S2. **Summaries of text triplet dataset.**

Dataset	**Use LLM**	**Model**	**Fine-tuning strategy**	**# of text triplets**
Rule-based	✗	✗	✗	1.3M
IP2P [4]	✓	GPT-3	Fine-tuned on 700 human-written text triplets	450K
Compodiff [8]	✓	OPT-6.7B	Fine-tuned on 450k IP2P text triplets	2.5M
Efficient in-context learning	✓	LLaMA3-8B	In-context learning	1M
CoVR [27]	✓	LLaMA-7B	Fine-tuned on 700 human-written text triplets	700K
CASE [12]	✓	GPT-3	In-context learning	350K

[Detailed explanation on rule-based triplets] To construct rule-based triplets, we mainly follow the process described in [8, Section 4.1]. Firstly, given reference captions, important keywords like nouns are extracted with a part-of-speech (POS) tagger via the Spacy library. Then, the keyword is filtered by frequency filtering with hard thresholding to focus only on frequently occurring keywords. Specifically, we only use keywords that appear more than 100 times. After applying keyword frequency filtering, the remaining keyword list is used to create caption triplets (T_r, T_c, T_t). To generate text triplets, a keyword from the given T_r is selected, and alternative keywords are extracted based on text similarity scores ranging from 0.5 to 0.7, using the SBERT all-MiniLM-L6-v2 [18]. The target caption T_t is then constructed by substituting the original keyword with a similar alternative. The conditioning text T_c is generated using randomly selected pre-defined templates, as detailed in Tab. S3. Here, most of the templates are similar to that of Compodiff [8]. We use captions from the CC3M dataset [22] as reference captions T_r. Note that CC3M is not related to the existing CIR benchmarks, which again ensures no overlap with the CIR benchmarks.

Since the quality of the generated triplets with the above procedure may not be optimal, we employ an additional filtering process. Compodiff [8] employs an additional filtering process that uses cosine similarities between generated images and texts, calculated by CLIP encoders. However, as we do not have images for captions, we filter the inappropriate texts using only textual information inspired by LinCIR [9]. Namely, we calculate the similarity between the CLIP text embedding of T_t and the CLIP text embedding of "a photo of [\$]" where [\$] is obtained by T_t projected by ϕ from LinCIR (ViT-L/14). Following LinCIR noise (Unif$(0, 1) \times \mathcal{N}(0, 1)$) is injected before passing through ϕ. After calculating the above similarity, texts whose similarities are less than the threshold (0.75) are removed. The same process is also applied to the reference caption T_r and the intersection of filtering processes for T_t and T_r is used for the final dataset whose size becomes 1.3M. As described in Suppl. B.6, we verify that this filtering process is effective. However, this does not imply that the effectiveness of rule-based text triplets is solely dependent on the use of a projection module in the filtering process; even without filtering, the enhancement from RTD remains valid.

```
{
  "source_caption": "what do you do with automobile model for $60 k",
  "target_caption": "what do you do with model for $60 k",
  "relative_caption": "without automobile"
},
{
  "source_caption": "a collage of my latest artwork includes oil pastel and acrylic paintings",
  "target_caption": "a collage of my latest artwork includes water pastel and acrylic paintings",
  "relative_caption": "alter oil to match water"
},
{
  "source_caption": "baseball player hits a home run against sports team",
  "target_caption": "baseball customer hits a home run against sports team",
  "relative_caption": "player is removed and customer takes its place"
},
{
  "source_caption": "another wall at my home",
  "target_caption": "another bedroom at my home",
  "relative_caption": "bedroom is added in place of wall"
},
{
  "source_caption": "tennis player faces a tough schedule if she is to advance",
  "target_caption": "tennis player faces a tough routine if she is to advance",
  "relative_caption": "change schedule to routine"
},
```

Figure 1. Example of rule-based triplet datasets

```
{
    "source_caption": "Christopher Nolan got advice from Steven Spielberg before making",
    "target_caption": "Steven Spielberg got advice from Walter Mitty before making",
    "relative_caption": "get advice from Walter Mitty"
},

{
    "source_caption": "by Koh Chip Whye - Black & White Buildings & Architecture",
    "target_caption": "by Koh Chip Whye - Colorful Buildings & Architecture",
    "relative_caption": "make the buildings more colorful"
},
{
    "source_caption": "The Most Hyperrealistic Images Of Beautiful Bathing Women With Their Heads Underwater",
    "target_caption": "The Most Hyperrealistic Images Of Beautiful Bathing Women With Their Heads Underwater and Octopus Arms",
    "relative_caption": "make the women have octopus arms"
},
{
    "source_caption": "Mountains above clouds - p312m1472749 by Mikael Svensson",
    "target_caption": "Mountains on Mars - p312m1472749 by Mikael Svensson",
    "relative_caption": "Put the mountains on Mars"
},
{
    "source_caption": "Le bouquiniste Paris -60x60",
    "target_caption": "The New York City book store -60x60",
    "relative_caption": "Instead of Paris, make it New York."
},
```

Figure 2. Example of LLM-based triplet datasets

Table S3. The full 50 keyword converting templates

”replace `${source}` with `${target}`”	”substitute `${target}` for `${source}`”
”apply `${target}`”	”`${source}` is removed and `${target}` takes its place”
”convert `${source}` to `${target}`”	”modify `${source}` to become `${target}`”
”replace `${source}` with `${target}`”	”customize `${source}` to become `${target}`”
”update `${source}` to `${target}`”	”change `${source}` to match `${target}`”
”substitute `${target}` for `${source}`”	”`${target}` is introduced after `${source}` is removed”
”alter `${source}` to match `${target}`”	”`${target}` is added in place of `${source}`”
”upgrade `${source}` to `${target}`”	”`${target}` is introduced as the new option after”
”amend `${source}` to fit `${target}`”	”`${source}` is removed and `${target}` is added”
”opt for `${target}`”	”`${source}` is removed and `${target}` is introduced”
”`${source}` is removed”	”`${target}` is added as a replacement for `${source}`”
”add `${target}`”	”`${target}` is the new option available”
”if it is `${target}`”	”`${target}` is added after `${source}` is removed”
”`${target}` is the updated option”	”`${target}` is introduced after `${source}` is retired”
”`${target}` is the updated choice”	”tweak `${source}` to become `${target}`”
”`${source}` is replaced with `${target}`”	”has no `${source}`”
”change `${source}` to `${target}`”	”alter `${source}` to `${target}`”
”swap `${source}` for `${target}`”	”redesign `${source}` as `${target}`”
”turn `${source}` into `${target}`”	”adapt `${source}` to fit `${target}`”
”choose `${target}` instead of `${source}`”	”`${target}` is the new choice”
”`${target}` is the new selection”	”exchange `${source}` with `${target}`”
”transform `${source}` into `${target}`”	”show no `${source}`”
”no `${source}`”	”remove `${source}`”
”delete `${source}`”	”not a `${source}`”
”with no `${source}`”	”without `${source}`”

A.3. Details on integration with FTI4CIR [13] and Context-I2W [24].

FTI4CIR [13] enhances the fine-grained capability of the projection module ϕ by separately handling subjects and attributes using distinct projection modules, ϕ_s and ϕ_a. To achieve this, they leverage BLIP-generated captions that explicitly separate subject and attribute information in the text domain, formatted as "[primary subject(s)] + [detailed description]." Unlike the subject module ϕ_s, which focuses on global subject information, ϕ_a processes localized attribute details. FTI4CIR extracts attribute features by adaptively aggregating patch features via an additional transformer. However, this approach requires an extra forward pass of the transformer with full sequences of visual embeddings for ϕ_a, rather than a single pooled embedding, making it incompatible with our refined concatenation scheme (RC). Therefore, we do not use ϕ_a during the training of RTD (of course, it is used in inference). Instead, we focus on training ϕ_s while incorporating attribute information in the text domain. Specifically, we generate text triplets using BLIP-generated captions to separately capture subject and attribute information, through "Efficient In-Context Learning" (see Suppl. A.2). Then, we provide subject information to ϕ_s while directly using attribute information in text format. For example, given text triplets: T_r : "a room with a chair and a table", T_c : "replace the chair with a sofa", T_t : "a room with a sofa and a table". our refined concatenation scheme (RC) ensures that only textual subject information is input to ϕ_s. This extraction is feasible because BLIP-generated captions (from FTI4CIR) already provide subject and attribute separation.

Context-I2W [24] introduces a context-dependent projection module that selects relevant visual information. To achieve this, Context-I2W employs a more complex projection mechanism that utilizes context captions such as "A [REPLACE] passes the ball with his teammate during a training session", which removes the first subject term. Then, these context captions and full visual sequence embeddings from the visual encoder are fused in the Context-I2W projection module. Since it requires full sequences of visual embeddings rather than a single pooled embedding, it can be incompatible with our refined concatenation scheme (RC). In such cases, our method can still be applied simply by replacing the text encoder. Namely, instead of training RTD with the projection module from Context-I2W, we simply replace the text encoder with the one updated by RTD using the projection module ϕ from Pic2Word [20]. Namely, during inference, we use the projection module from Context-I2W alongside the RTD-updated text encoder (which was trained with the projection module from Pic2Word). The strong performance of this variant suggests that RTD does not need to be trained with each specific projection module, again highlighting its strong generalizability.

B. Additional experimental results

B.1. Results on GeneCIS [26] and COCO object composition

We observe that integrating our approach with projection-based CIR methods results in consistent yet marginal performance improvements on GeneCIS, as shown in Tab. S4. The relatively smaller performance difference compared to other datasets can be attributed to the types of conditioning text used in GeneCIS. Namely, GeneCIS only uses the fixed four text instructions "change attribute", "focus attribute", "change object", and "focus object", which is different from the usual text instruction we expected (*e.g.*, "change the dog to a cat").

In the experiment on COCO object composition, we observe a significant performance improvement, similar to the results obtained on other datasets in Tab. S5. This finding reaffirms that our approach, when combined with ZS-CIR methods, consistently achieves strong performance, demonstrating its generalizability.

B.2. Results on different backbones (ViT-B/32, ViT-L/14)

Full results of Fig. 3 can be found in Tab. S6 and Tab. S7. Tab. S6 shows the evaluation results on the FashionIQ dataset. In the table, we observe that the incorporation of our approach with projection-based CIR methods significantly improves the performance across all three existing projection-based CIR methods (SEARLE, Pic2Word, and LinCIR) and all backbones (ViT-B/32 and ViT-L/14). For example, regardless of the choice of projection-based CIR methods and backbones, the minimum performance gain for average R@10 and R@50 scores is greater than 2 and 3.5 points, respectively. Tab. S7 shows a similar trend on the CIRR and CIRCO datasets. Notably, in some metrics on the CIRR and CIRCO datasets, the performance improvements achieved through our method (ViT-B/32) even exceed those obtained by employing a larger backbone (ViT-L/14), which demonstrates the effect of our method. Specifically, in the CIRR R@1 score, SEARLE + RTD (26.29) and LinCIR + RTD (24.82) using ViT-B/32 surpasses the original results of SEARLE (24.89) and LinCIR (23.76) using ViT-L/14.

B.3. Results on larger backbone (ViT-G/14)

As reported in Tab. 2, we evaluate the performance of RTD using the significantly larger backbone (ViT-G/14 [10]). As described in Sec. 4.3, we use the projection module ϕ from LinCIR [9]. Since the pre-trained projection module ϕ for LinCIR

Table S4. GeneCIS results

		Average		
		R@1	R@2	R@3
ViT-B	Pic2Word	11.13	21.08	31.05
	+RTD	12.03 (+**0.90**)	21.61 (+**0.53**)	31.09 (+**0.04**)
	SEARLE	12.19	22.56	32.03
	+RTD	12.82 (+**0.63**)	22.97 (+**0.41**)	32.44 (+**0.41**)
	LinCIR	12.23	21.29	30.80
	+RTD	12.83 (+**0.60**)	22.83 (+**1.54**)	32.22 (+**1.42**)
ViT-L	Pic2Word	11.18	21.45	30.55
	+RTD	11.92 (+**0.74**)	22.32 (+**0.87**)	31.33 (+**0.78**)
	SEARLE	12.30	22.08	31.29
	+RTD	12.40 (+**0.10**)	22.82 (+**0.74**)	32.37 (+**1.08**)
	LinCIR	12.45	22.66	32.06
	+RTD	13.18 (+**0.73**)	23.12 (+**0.46**)	32.77 (+**0.71**)

Table S5. COCO object composition results

		COCO		
		R@1	R@5	R@10
ViT-B	Pic2Word	6.88	13.6	17.52
	+RTD	7.62 (+**0.74**)	20.23 (+**6.63**)	28.69 (+**11.17**)
	SEARLE	9.52	21.45	29.38
	+RTD	11.01 (+**1.49**)	24.34 (+**2.89**)	32.84 (+**3.46**)
	LinCIR	7.15	18.38	27.3
	+RTD	9.59 (+**2.44**)	21.66 (+**3.28**)	30.66 (+**3.36**)
ViT-L	Pic2Word	10.26	23.67	32.53
	+RTD	10.26 (+**0.00**)	24.66 (+**0.99**)	33.56 (+**1.03**)
	SEARLE	12.07	26.13	35.17
	+RTD	14.38 (+**2.31**)	29.74 (+**3.61**)	38.09 (+**2.92**)
	LinCIR	11.37	24.53	33.85
	+RTD	14.6 (+**3.23**)	29.84 (+**5.31**)	38.87 (+**5.02**)

[9] (ViT-G/14) is not publicly available, we reproduce it and integrate RTD with it. We emphasize that similar to our previous results, RTD again achieves remarkable gains across all datasets. Here, we set the learning rate as 10^{-6}.

B.4. More efficient variants

Tab. S10 presents the results of the more efficient implementations of our approach in terms of the number of updated parameters. Specifically, instead of updating the entire set of parameters of the text encoder, we explore updating only a few layers of the network when applying RTD, Our findings indicate that updating only the fully connected layers (denoted as "Whole FCs") nearly matches the performance of the full model while using less than half the number of learnable parameters (40.72 vs. 40.53 average scores). Additionally, we verify that updating only three fully connected layers, whose parameter size matches the projection module ϕ and constitutes 11.5% of the full model, is also sufficiently effective. We test various three-layer updating strategies: "First 3 FCs": the first three layers (closest to the input), "middle 3 FCs": the middle three layers, "Last 3 FCs": the last three layers, and "Interleave 3 FCs": an interleaved selection of three layers (first, middle, and last layers). Among these, we verify that the "Interleave 3 FCs" shows the best result, maintaining competitive performance

Table S6. **FashionIQ validation results.** The results of RTD combined with Pic2Word [20], SEARLE [3], and LinCIR [9] across different CLIP backbones (ViT-B/32 and ViT-L/14) are shown.

		Shirt		Dress		Toptee		Average	
		R@10	R@50	R@10	R@50	R@10	R@50	R@10	R@50
ViT-B/32	Pic2Word	13.40	28.46	8.48	20.77	13.31	29.68	11.73	26.30
	+RTD	23.06 (+9.66)	40.48 (+12.02)	20.33 (+11.85)	41.75 (+20.98)	24.12 (+10.81)	46.35 (+16.67)	22.5 (+10.77)	42.86 (+16.56)
	SEARLE	24.78	41.85	17.90	36.99	25.24	46.71	22.64	41.85
	+RTD	26.69 (+1.91)	44.31 (+2.46)	20.72 (+2.82)	43.13 (+6.14)	26.67 (+1.43)	48.75 (+2.04)	24.7 (+2.06)	45.4 (+3.55)
	LinCIR	18.55	34.64	15.67	33.86	20.19	40.08	18.14	36.20
	+RTD	23.65 (+5.10)	42.74 (+8.10)	19.98 (+4.31)	41.75 (+7.89)	24.73 (+4.54)	46.56 (+6.48)	22.79 (+4.65)	43.68 (+7.48)
ViT-L/14	Pic2Word	26.59	42.93	21.32	43.53	28.10	48.19	25.34	44.88
	+RTD	27.97 (+1.38)	46.96 (+4.03)	23.50 (+2.18)	46.65 (+3.12)	31.31 (+3.21)	53.09 (+4.90)	27.59 (+2.25)	48.90 (+4.02)
	SEARLE	26.94	45.34	19.58	40.80	28.45	49.77	24.99	45.30
	+RTD	32.63 (+5.69)	50.39 (+5.05)	23.2 (+3.62)	47.25 (+6.45)	32.18 (+3.73)	54.56 (+4.79)	29.34 (+4.35)	50.73 (+5.43)
	LinCIR	30.42	47.99	21.86	44.77	29.98	50.38	27.42	47.71
	+RTD	32.83 (+2.41)	50.44 (+2.45)	24.49 (+2.63)	48.24 (+3.47)	33.4 (+3.42)	54.56 (+4.18)	30.24 (+2.82)	51.08 (+3.37)

Table S7. **CIRR and CIRCO test results.** Details are the same as Tab. S6.

		CIRR			CIRCO			
		R@1	R@5	R@10	mAP@5	mAP@10	mAP@25	mAP@50
ViT-B/32	Pic2Word	13.64	37.45	52.22	2.85	3.24	3.89	4.31
	+RTD	23.59 (+9.95)	51.76 (+14.31)	65.16 (+12.94)	6.39 (+3.54)	6.66 (+3.42)	7.64 (+3.75)	8.16 (+3.85)
	SEARLE	23.71	53.3	66.84	8.90	9.42	10.64	11.34
	+RTD	26.29 (+2.58)	56.41 (+3.11)	69.74 (+2.90)	11.26 (+2.36)	12.11 (+2.69)	13.63 (+2.99)	14.37 (+3.03)
	LinCIR	18.87	45.66	58.43	6.25	6.74	7.62	8.10
	+RTD	24.82 (+5.95)	53.47 (+7.81)	66.87 (+8.44)	8.94 (+2.69)	9.35 (+2.61)	10.57 (+2.95)	11.21 (+3.11)
ViT-L/14	Pic2Word	24.22	51.49	64.05	8.27	9.10	10.09	10.75
	+RTD	27.86 (+3.64)	56.24 (+4.75)	68.48 (+4.43)	9.13 (+0.86)	9.63 (+0.53)	10.68 (+0.59)	11.27 (+0.52)
	SEARLE	24.89	52.31	65.69	11.62	12.72	14.33	15.13
	+RTD	26.63 (+1.74)	56.17 (+3.86)	68.96 (+3.27)	16.53 (+4.91)	17.89 (+5.17)	19.77 (+5.44)	20.68 (+5.55)
	LinCIR	23.76	52.89	66.46	13.00	14.11	15.81	16.68
	+RTD	26.63 (+2.87)	56.17 (+3.28)	68.96 (+2.50)	17.11 (+4.11)	18.11 (+4.00)	20.06 (+4.25)	21.01 (+4.33)

Table S8. FashionIQ validation results on larger OpenCLIP ViT-G/14 backbone [10].

Method	Shirt		Dress		Toptee		Average	
	R@10	R@50	R@10	R@50	R@10	R@50	R@10	R@50
LinCIR (reported in [9])	46.76	65.11	38.08	60.88	50.48	71.09	45.11	65.69
LinCIR (reproduced)	46.61	64.72	38.18	60.54	49.26	70.83	44.68	65.36
+RTD	47.20 (+0.59)	66.24 (+1.52)	39.86 (+1.68)	63.01 (+2.47)	51.56 (+2.30)	72.51 (+1.68)	46.21 (+1.54)	67.26 (+1.90)

Table S9. CIRR and CIRCO test results on larger OpenCLIP ViT-G/14 backbone [10].

ViT-G	CIRR			CIRCO			
	R@1	R@5	R@10	mAP@5	mAP@10	mAP@25	mAP@50
LinCIR (reported in [9])	35.25	64.72	76.05	19.81	21.01	23.03	24.18
LinCIR (reproduced)	34.94	64.51	76.12	20.63	21.93	24.12	25.20
+RTD	36.31 (+1.37)	67.47 (+2.96)	78.31 (+2.19)	21.08 (+0.45)	22.29 (+0.36)	24.46 (+0.34)	25.44 (+0.24)

with the full model (40.72 vs. 39.88 average scores). We believe these findings suggest a promising direction for enhancing the training efficiency of our approach by selectively updating only specific layers of the text encoder.

Table S10. **More efficient variants.** "Learnable params (%)" denotes the percentage of learnable parameters relative to the entire set of parameters in the text encoder. Other details are the same as Tab. 6.

Training variants	Learnable params (%)	CIRR R@5	CIRR R@10	CIRCO mAP@10	CIRCO mAP@25	FashionIQ R@10	FashionIQ R@50	Avg
Baseline(LinCIR)	0%	54.29	67.76	12.67	14.45	27.42	47.71	37.38
+RTD (Full model)	100%	57.90	71.13	16.10	17.84	30.24	51.08	40.72
+RTD (Whole FCs)	45.8%	57.76	71.35	15.03	16.90	30.31	51.81	40.53
+RTD (Front 3 FCs)	11.5%	55.65	69.83	13.95	15.81	28.69	49.92	38.98
+RTD (Middle 3 FCs)	11.5%	56.69	70.03	14.66	16.58	28.55	49.84	39.39
+RTD (Last 3 FCs)	11.5%	56.84	69.74	14.81	16.70	29.16	50.43	39.61
+RTD (Interleave 3 FCs)	11.5%	57.21	70.65	15.20	17.13	28.91	50.17	39.88

B.5. Effectiveness of RTD across dataset scales

We conduct experiments with various scales of training text triplets. For the small-scale text triplets, we sub-sampled text triplets from LLM-based text triplets. Thus, the last row in the Tab. S11 denotes the original result (LLM-based RTD result). We also measure the effectiveness of RTD using large-scale text triplets (up to 5M) by combining publicly available text triplets (IP2P, CoVR) with ours (LLM-based, rule-based). Here, validation splits of all three benchmark datasets are utilized.

As shown in Tabs. S11 and S12, we confirm that small-scale text triplets are sufficient to achieve the effectiveness of RTD. We believe the main reason for this is that, to reduce task discrepancy, only the relationship between the concatenated caption (reference caption + conditioning caption, T_{r+c}) and the target caption T_t needs to be learned. We believe this learning task requires much less data compared to learning representations from scratch. Moreover, since the text encoder is already pre-trained, the model does not need significant changes to learn this simple but crucial learning task for CIR.

Table S11. **Results across different scales of LLM-based text triplets**. In each row, text triplets are sub-sampled from 2.5M original LLM-based text triplets provided by Compodiff [8]. Other details are the same as Tab. 6.

# of triplets	CIRR R@5	CIRCO mAP@10	FashionIQ R@10	Avg
1K	56.64	15.66	29.89	34.06
50K	57.40	15.95	30.77	34.71
100K	57.16	16.03	30.57	34.59
2.5M	57.90	16.10	30.24	34.75

Table S12. Results of larger-sized text triplets. Other details are the same as Tab. 6.

IP2P	CoVR	Compodiff	Template-based	CIRR R@5	CIRCO mAP@10	FashionIQ R@10	Avg	# of triplets
✓				58.65	15.94	29.62	34.74	450k
	✓			59.82	15.35	29.58	34.92	700k
		✓		57.90	16.10	30.24	34.75	2.5M
			✓	56.71	15.01	30.37	34.03	1.3M
✓	✓			59.32	16.10	30.81	35.41	1.25M
✓	✓	✓		59.08	16.15	30.97	35.40	3.75M
✓	✓	✓	✓	58.65	16.54	31.22	35.47	5.05M

B.6. Ablations on filtering process

In rule-based text triplet generation, we highlight that the filtering process using the projection module from LinCIR is *marginally effective*. As demonstrated in the Tab. S13, even without the filtering procedure, the enhancement of RTD from

LinCIR remains considerable. This result demonstrates that the effectiveness of our rule-based text triplets is not solely dependent on the use of the projection module from LinCIR in the filtering process.

Table S13. Ablations on filtering process. Other details are the same as Tab. 6.

Type	LinCIR-based filtering	CIRR R@5	CIRCO mAP@10	FashionIQ R@10	Avg
LinCIR	-	54.29	12.67	27.42	31.46
+RTD (rule-based)	✗	55.49	14.75	30.24	33.49
+RTD (rule-based)	✓	56.71	15.01	30.37	34.03

B.7. Usage of InfoNCE-based symmetric loss

Although only one text encoder is updated (with the encoder used for target representations kept frozen), we adopt a symmetric InfoNCE loss to leverage bidirectional alignment and enrich the pool of negative samples. We observe that this symmetric formulation outperforms asymmetric variants (which also have fewer negatives), improving CIRR val R@1 by over 2 points.

B.8. Text triplets with multiple keyword changes

Unlike previous experiments that change a single keyword, we conduct experiments on text triplets with multiple keyword changes in a single reference caption, using both template-based and LLM-based strategies. The template-based strategy can be *easily* extended by (1) extracting multiple keywords per caption (when available), (2) applying the predefined templates from Tab. S3 to each keyword, and (3) concatenating the resulting phrases to form the conditioning text (and using the keyword-altered caption as the target text). For the LLM-based strategy, we can extend the in-context approach described in Sec. 3.1 by simply providing different prompts to handle multiple keyword changes. As shown in Tab. S14, the effect of the extension appears marginal, likely because current CIR benchmarks are not designed to clearly distinguish between single and multiple changes. We believe that the ease of constructing goal-specific text triplets, as demonstrated above, combined with the simplicity of RTD, makes this approach promising for future enhancements and more complex scenarios.

B.9. Zero-shot accuracy on ImageNet-1k

We measure zero-shot accuracy on ImageNet-1k [19] before and after RTD training using the ViT-L/14 backbone, following the default CLIP evaluation protocol. As RTD is tailored for CIR, it is expected that fine-tuning the CLIP text encoder with RTD (74.48) results in a slight performance drop (-0.63) compared to keeping it frozen (73.85). The degradation is minor and not unique to RTD, as similar trends may appear in other CIR methods with synthetic CIR triplets, where preserving alignment with the pre-trained backbone is not explicitly considered. Compared to other variants, RTD adds minimal storage overhead for the CIR task, requiring only the adapted text encoder to be saved. As also discussed in Suppl. B.4, it can be further reduced by updating and saving only a small subset of parameters. We leave the challenge of preventing the forgetting of pre-trained knowledge while improving CIR performance to future work.

Table S14. Additional results on multiple keywords (Here, two keywords). Other details are the same as Tab. 6.

	Source	Multi keywords	CIRR (R)		CIRCO (mAP)		FIQ (R)		Average
			@5	@10	@10	@25	@10	@50	
LinCIR	-	-	54.3	67.8	12.7	14.5	27.4	47.7	37.4
+RTD	Rule-based	✗	56.7	70.3	**15.0**	**17.0**	30.4	**51.9**	**40.2** (+ 2.9)
		✓	**57.0**	**71.0**	14.8	16.8	**30.7**	50.9	**40.2** (+ 2.9)
	LLM	✗	**59.3**	**71.8**	**15.8**	**17.5**	29.7	51.4	40.9 (+ 3.5)
		✓	58.0	71.7	15.5	**17.5**	**31.3**	**52.4**	**41.1** (+ 3.7)

B.10. Ablations on noise injection

We conduct an ablation study of the different noise types employed for the "refined concatenation scheme" shown in Fig. 2. We compare three different noise types, uniform distribution, Gaussian distribution, and LinCIR-ish noise ($\text{Unif}(0,1) \times \mathcal{N}(0,1)$).

Table S15. Noise type variation on CIRR/CIRCO dataset

				CIRR			CIRCO			
		Noise type	Scale	R@1	R@5	R@10	mAP@5	mAP@10	mAP@25	mAP@50
ViT-B/32	Pic2Word	-	-	13.64	37.45	52.22	2.85	3.24	3.89	4.31
	+RTD	Unif(-1,1)	1	23.23	50.55	64.28	4.29	4.57	5.19	5.57
		$\mathcal{N}(0,1)$	1	21.18	47.78	61.47	4.09	4.26	4.83	5.17
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	23.52	51.13	64.53	5.13	5.46	6.17	6.62
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	23.01	51.18	64.84	4.29	4.57	5.19	5.57
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	23.59	51.76	65.16	6.39	6.66	7.64	8.16
	SEARLE	-	-	23.71	53.3	66.84	8.9	9.42	10.64	11.34
	+RTD	Unif(-1,1)	1	26.07	55.98	69.18	10.87	11.55	12.97	13.65
		$\mathcal{N}(0,1)$	1	26.41	56.68	69.47	10.91	11.53	12.88	13.6
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	26.02	55.47	68.15	10.43	11.07	12.37	13.07
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	26.29	56.41	69.74	11.26	12.11	13.63	14.37
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	26.43	56.58	69.76	11.42	12.04	13.38	14.1
	LinCIR	-	-	18.87	45.66	58.43	6.25	6.74	7.62	8.1
	+RTD	Unif(-1,1)	1	24.39	52.77	66.39	6.81	7.27	8.28	8.84
		$\mathcal{N}(0,1)$	1	24.63	53.52	66.63	7.6	7.97	8.92	9.49
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	24.58	53.3	66.65	9.6	10.11	11.47	12.15
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	24.82	53.47	66.87	8.94	9.35	10.57	11.21
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	25.4	54.58	67.69	8.17	8.53	9.72	10.35
ViT-L/14	Pic2Word	-	-	24.22	51.49	64.05	8.27	9.1	10.09	10.75
	+RTD	Unif(-1,1)	1	28.24	55.95	68.77	8.14	8.81	9.83	10.37
		$\mathcal{N}(0,1)$	1	27.06	53.95	66.43	7.08	7.66	8.57	9.07
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	28.24	57.35	68.65	10.04	10.63	11.71	12.31
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	27.86	56.24	68.48	9.13	9.63	10.68	11.27
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	27.71	55.68	68.02	8.14	8.78	9.84	10.35
	SEARLE	-	-	24.89	52.31	65.69	11.62	12.72	14.33	15.13
	+RTD	Unif(-1,1)	1	26.96	56.99	69.52	15.82	16.78	18.54	19.39
		$\mathcal{N}(0,1)$	1	27.66	57.54	69.57	15.24	15.93	17.65	18.44
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	26.31	55.88	69.4	16.05	17.26	19.12	20.01
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	27.04	56.82	69.95	16.53	17.89	19.77	20.68
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	27.93	57.76	70.19	17.35	18.66	20.52	23.44
	LinCIR	-	-	23.76	52.89	66.46	13	14.11	15.81	16.68
	+RTD	Unif(-1,1)	1	26.58	56.31	68.94	17.23	18.2	20.11	21.03
		$\mathcal{N}(0,1)$	1	26.75	55.64	68.48	16.45	17.57	19.37	20.3
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	26.7	56.22	69.08	17.24	18.27	20.24	21.19
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	26.63	56.17	68.96	17.11	18.11	20.06	21.01
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	26.99	56.1	69.01	17.33	18.3	20.21	21.13

We also examine the scale of LinCIR-ish noise from 0.1, 0.5, and 1. We report the test scores for CIRR and CIRCO, as well as the FashionIQ validation scores for Pic2Word, SEARLE, and LinCIR in Tab. S15 and Tab. S16. In the tables, we observe that all noise distributions show decent performance and LinCIR-like noises show slightly better performances than uniform distribution and normal distribution. We also observe that the different scale choice for the LinCIR-like noise somewhat affects the overall performances. In the main experiments, we chose 0.5 for the noise scale, following the observed performance improvements.

C. Generating text triplets cost

Although generating text triplets is not our main contribution, for comprehensive understanding, we compare the generation time of LLM-based and rule-based approaches. Even when using LLMs, constructing text triplets is significantly more cost-effective than CIR triplets. Specifically, CIR triplets involve: 1) a subsequent, computationally intensive text-to-image

Table S16. Noise type variation on FashionIQ dataset

				Shirt		Dress		Toptee		Average	
		Noise type	Scale	R@10	R@50	R@10	R@50	R@10	R@50	R@10	R@50
ViT-B/32	Pic2Word	-	-	13.4	28.46	8.48	20.77	13.31	29.68	11.73	26.3
	+RTD	Unif(-1,1)	1	21.84	37.63	18.49	39.61	23.0	43.91	21.11	40.38
		$\mathcal{N}(0,1)$	1	20.36	37.54	16.16	38.18	21.67	42.48	19.4	39.4
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	22.23	39.35	19.98	41.7	23.81	45.23	22.01	42.09
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	24.53	43.82	20.33	41.55	26.01	48.75	23.62	44.7
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	23.06	40.48	20.33	41.75	24.12	46.35	22.5	42.86
	SEARLE	-	-	24.78	41.85	17.90	36.99	25.24	46.71	22.64	41.85
	+RTD	Unif(-1,1)	1	23.75	42.25	20.18	40.36	25.14	46.46	23.02	43.02
		$\mathcal{N}(0,1)$	1	24.14	42.25	20.23	40.16	24.17	46.35	22.85	42.92
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	25.12	44.85	20.92	41.40	26.57	47.63	24.20	44.62
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	26.69	44.31	20.72	43.13	26.67	48.75	24.70	45.40
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	25.07	44.01	20.43	41.00	26.11	47.12	23.87	44.04
	LinCIR	-	-	18.55	34.64	15.67	33.86	20.19	40.08	18.14	36.20
	+RTD	Unif(-1,1)	1	21.79	39.35	18.89	40.21	23.66	45.33	21.45	41.63
		$\mathcal{N}(0,1)$	1	22.37	38.67	19.53	40.11	23.71	44.37	21.87	41.05
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	23.95	44.11	19.83	41.99	26.62	47.58	23.47	44.56
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	23.65	42.74	19.98	41.75	24.73	46.56	22.79	43.68
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	22.82	41.12	19.78	41.70	25.09	47.07	22.56	43.29
ViT-L/14	Pic2Word	-	-	26.59	42.93	21.32	43.53	28.10	48.19	25.34	44.88
	+RTD	Unif(-1,1)	1	27.87	45.93	23.90	46.80	31.21	52.22	27.66	48.32
		$\mathcal{N}(0,1)$	1	26.94	44.95	23.45	45.56	30.34	51.45	26.91	47.32
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	28.26	47.64	24.05	47.20	31.21	53.70	27.84	49.51
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	27.97	46.96	23.50	46.65	31.31	53.09	27.59	48.90
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	28.41	46.91	24.10	46.21	31.11	52.27	27.87	48.46
	SEARLE	-	-	26.94	45.34	19.58	40.80	28.45	49.77	24.99	45.30
	+RTD	Unif(-1,1)	1	30.13	46.57	22.16	46.90	28.76	50.74	27.02	48.07
		$\mathcal{N}(0,1)$	1	26.99	43.23	21.17	44.82	27.54	49.06	25.23	45.70
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	32.63	50.39	23.20	47.25	32.18	54.56	29.34	50.73
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	31.80	49.31	23.20	47.30	31.41	54.00	28.80	50.20
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	30.03	47.06	22.41	47.05	30.39	52.42	27.61	48.84
	LinCIR	-	-	30.42	47.99	21.86	44.77	29.98	50.38	27.42	47.71
	+RTD	Unif(-1,1)	1	31.94	50.10	24.44	48.19	33.04	54.26	29.81	50.85
		$\mathcal{N}(0,1)$	1	31.70	49.41	23.90	48.19	33.23	53.54	29.27	50.38
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.1	32.92	50.64	24.49	48.74	33.50	55.02	30.31	51.47
		$\mathcal{N}(0,1)\times$ Unif(0,1)	0.5	32.83	50.44	24.49	48.24	33.40	54.56	30.24	51.08
		$\mathcal{N}(0,1)\times$ Unif(0,1)	1	32.43	50.54	24.64	48.79	33.25	54.77	30.11	51.36

generation phase [4, 8], or 2) the availability of image or video datasets along with an additional collection phase for semantically similar images or videos [12, 27]. In contrast, generating text triplets bypasses these resource-heavy steps. For example, using 8 A100 GPUs, generating 1M text triplets takes 0.1 hours with the rule-based approach and 3.8 hours with the LLM-based approach from Compodiff [8] (OPT-6.7B). As described in Suppl. A.2, a more efficient text triplet generation method using in-context learning with LLaMA3-8B reduces the generation time to 1.5 hours without the need for fine-tuning.

Therefore, while generating text triplets with LLMs incurs a higher cost compared to rule-based methods, it is still significantly faster (15 times) than generating CIR triplets (as used in CompoDiff), which utilize the text generation step as a preliminary phase for subsequent text-to-image generation. Thus, we believe that LLM-based generation remains viable, and the rule-based approach offers greater efficiency.

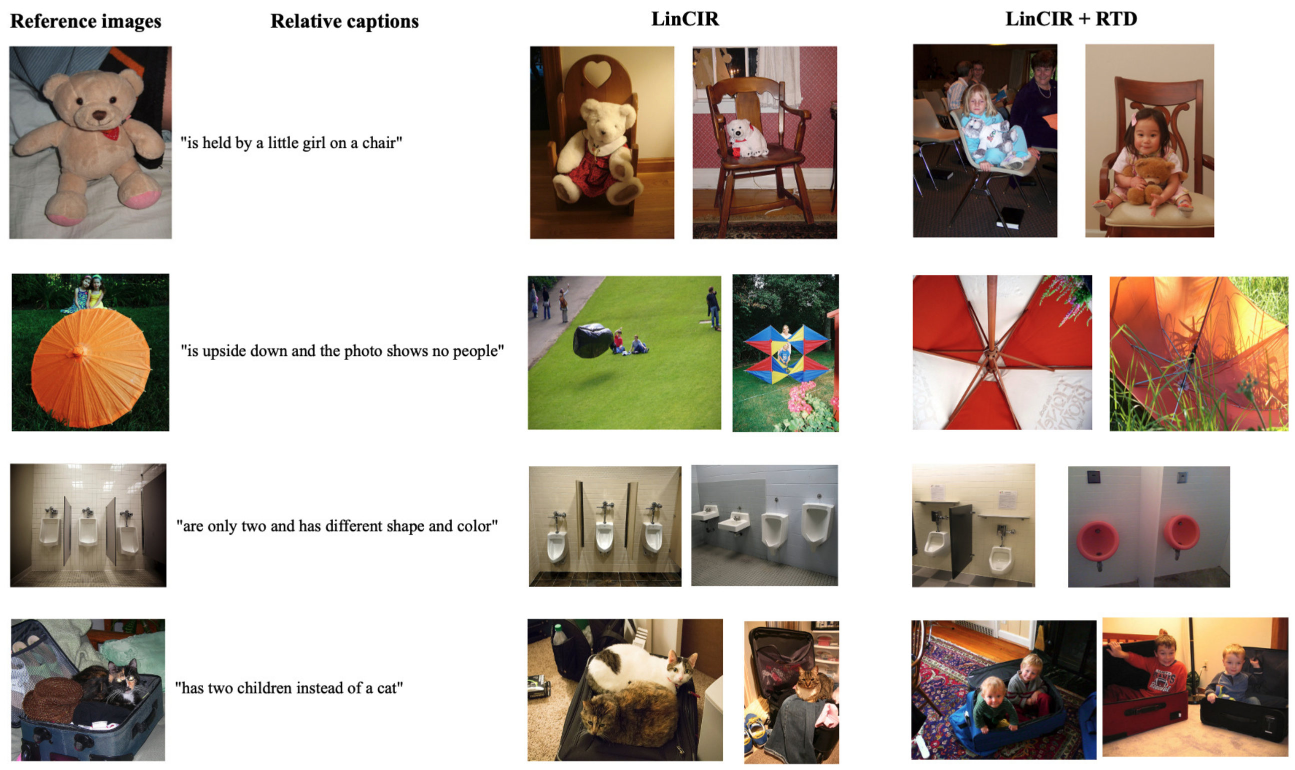
Reference images
Relative captions
LinCIR
LinCIR + RTD
"is held by a little girl on a chair"
"is upside down and the photo shows no people"
"are only two and has different shape and color"
"has two children instead of a cat"

Figure 3. Qualitative Results on CIRCO dataset

D. Qualitative example on CIRCO

We qualitatively illustrate the results of incorporating RTD into LinCIR on the CIRCO dataset in Fig. 3. The visual examples provide an intuitive demonstration of how the integration of RTD enhances the performance of LinCIR, effectively capturing the semantic meaning of the modification descriptions while preserving the relevant visual information from the reference image.

E. Discussion and limitation

First, we have mainly focused on evaluating the integrability of RTD with representative projection-based CIR methods [3, 9, 20]. However, we have not yet explored or tested the extensibility of RTD to other CIR approaches that achieve strong performance, such as those utilizing human-annotated CIR triplets (supervised) [2], synthetically generated CIR triplets [8, 12, 27], or training-free methods [11]. Given the core motivation behind RTD, its adaptability to those CIR approaches that directly train fusion modules or backbones using CIR triplets may be limited. However, considering the strong performance and practical advantages (such as efficient training and inference) offered by projection-based CIR methods compared to other variants, we believe that integrating RTD with them remains a valuable direction in the CIR domain. Second, RTD addresses only a specific aspect of task discrepancy in projection-based CIR methods by focusing solely on the text encoder, leaving the projectors and image encoder unchanged. However, considering the broad applicability of RTD to projection-based methods and the relatively high computational cost of forward passes through the image encoder, we believe this targeted focus offers the most practically viable direction. We leave further exploration of other components for future work. Third, as a post-hoc method, RTD naturally depends on the underlying backbones and projectors. While this can be seen as a limitation, it also underscores the versatility of RTD, as it can be easily applied to a wide range of existing methods. Moreover, the ease of constructing goal-specific text triplets, combined with the simplicity of RTD, makes it a promising direction for future improvements and more complex scenarios. Finally, as RTD uses text triplets, its performance depends on the quality of training text triplets. However, as shown in Tab. 6, it remains robust across diverse triplet types, including template-based ones. With the rise of LLMs, generating high-quality text triplets has become much easier (than image counterparts), further enhancing the practicality and scalability of RTD.

F. Societal Impacts

Although our paper demonstrates promising outcomes in the ZS-CIR task, further examination of the data and the model is essential prior to practical deployment. Since our method focuses mainly on optimization for accuracy, unwanted social implications can occur. For example, real-world images from databases and user-generated text may inadvertently cause harmful cases.

References

[1] Max Bain, Arsha Nagrani, Gül Varol, and Andrew Zisserman. Frozen in time: A joint video and image encoder for end-to-end retrieval. In *ICCV*, 2021. 2

[2] Alberto Baldrati, Marco Bertini, Tiberio Uricchio, and Alberto Del Bimbo. Conditioned and composed image retrieval combining and partially fine-tuning clip-based features. In *CVPR*, 2022. 12

[3] Alberto Baldrati, Lorenzo Agnolucci, Marco Bertini, and Alberto Del Bimbo. Zero-shot composed image retrieval with textual inversion. In *ICCV*, 2023. 1, 7, 12

[4] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *CVPR*, 2023. 1, 2, 11

[5] Sanghyuk Chun, Wonjae Kim, Song Park, Minsuk Chang, and Seong Joon Oh. Eccv caption: Correcting false negatives by collecting machine-and-human-verified image-caption associations for ms-coco. In *ECCV*, 2022. 1

[6] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 2

[7] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017. 1, 2

[8] Geonmo Gu, Sanghyuk Chun, HeeJae Jun, Yoohoon Kang, Wonjae Kim, and Sangdoo Yun. Compodiff: Versatile composed image retrieval with latent diffusion. *TMLR*, 2023. 1, 2, 3, 8, 11, 12

[9] Geonmo Gu, Sanghyuk Chun, Wonjae Kim, , Yoohoon Kang, and Sangdoo Yun. Language-only efficient training of zero-shot composed image retrieval. In *CVPR*, 2024. 1, 3, 5, 6, 7, 12

[10] Gabriel Ilharco, Mitchell Wortsman, Ross Wightman, Cade Gordon, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. Openclip, 2021. 5, 7

[11] Shyamgopal Karthik, Karsten Roth, Massimiliano Mancini, and Zeynep Akata. Vision-by-language for training-free compositional image retrieval. In *ICLR*, 2024. 12

[12] Matan Levy, Rami Ben-Ari, Nir Darshan, and Dani Lischinski. Data roaming and quality assessment for composed image retrieval. In *AAAI*, 2024. 1, 2, 11, 12

[13] Haoqiang Lin, Haokun Wen, Xuemeng Song, Meng Liu, Yupeng Hu, and Liqiang Nie. Fine-grained textual inversion network for zero-shot composed image retrieval. In *SIGIR*, 2024. 5

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 1

[15] Zheyuan Liu, Cristian Rodriguez-Opazo, Damien Teney, and Stephen Gould. Image retrieval on real-life images with pre-trained vision-and-language models. In *ICCV*, 2021. 1

[16] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *ECCV*, 2020. 1

[17] Khoi Pham, Kushal Kafle, Zhe Lin, Zhihong Ding, Scott Cohen, Quan Tran, and Abhinav Shrivastava. Learning to predict visual attributes in the wild. In *CVPR*, 2021. 1

[18] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*, 2019. 2

[19] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 9

[20] Kuniaki Saito, Kihyuk Sohn, Xiang Zhang, Chun-Liang Li, Chen-Yu Lee, Kate Saenko, and Tomas Pfister. Pic2word: Mapping pictures to words for zero-shot composed image retrieval. In *CVPR*, 2023. 1, 5, 7, 12

[21] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *NeurIPS Dataset and Benchmark*, 2022. 1

[22] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 2, 3

[23] Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. A corpus for reasoning about natural language grounded in photographs. In *ACL*, 2019. 1

[24] Yuanmin Tang, Jing Yu, Keke Gai, Jiamin Zhuang, Gang Xiong, Yue Hu, and Qi Wu. Context-i2w: Mapping images to context-dependent words for accurate zero-shot composed image retrieval. In *AAAI*, 2024. 5

[25] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 2

[26] Sagar Vaze, Nicolas Carion, and Ishan Misra. Genecis: A benchmark for general conditional image similarity. In *CVPR*, 2023. 1, 5

[27] Lucas Ventura, Antoine Yang, Cordelia Schmid, and Gül Varol. CoVR: Learning composed video retrieval from web video captions. In *AAAI*, 2024. 1, 2, 11, 12

[28] Hui Wu, Yupeng Gao, Xiaoxiao Guo, Ziad Al-Halah, Steven Rennie, Kristen Grauman, and Rogerio Feris. Fashion iq: A new dataset towards retrieving images by natural language feedback. In *CVPR*, 2021. 1