

## Contents

1. Appendix A ... Limitations
2. Appendix B ... Dataset preparation
3. Appendix C ... Algorithmic details
4. Appendix D ... Experimental details and discussions
5. Appendix E ... Additional experiments

## A Limitations

Our work considers only a fraction of contrastive models, we believe CLIC should work similarly for other models like SigLIP [57], which we were unable to test due to compute and time constraints. Moreover, we were only able to test LLaVA [31], whereas other large VLMs like VILA [28] and CogVLM [53] might also benefit from CLIC, which we leave for future work.

## B Data preparation

In this section, we describe the data preparation process in detail. Appendix B.1 showcases randomly selected images and captions from the PixelProse [50] dataset and from the Laion [46] dataset, along with captions generated for these images by CogVLM [53]. This part also presents further details on these datasets. In Appendix B.2, we explain the hard-negative generation procedure. In Appendix B.3 we show examples of the limitations of using a single image for generating negatives as done in [56]. Appendix B.4 shows ablations regarding other ways of combining images for CLIC.

### B.1 Training datasets

In this section, we present our data generation process. To show that CLIC performs well across diverse datasets, we train models from three different sources. We took a subset of 1M images from Laion with captions that were generated by CogVLM [53], using the query presented in Figure 3. We also used a recently proposed high-quality caption dataset in PixelProse [50], which contains 850k available samples from the RedCaps subset. In order to compare faithfully to our baselines in Table 1 who use CC3M, we also use a 1.2M subset of CC12M that comes from the same data stream as CC3M to train a ViT-B/32 model. Since our method requires detailed captions, we use the recaptioned version of CC12M from PixelProse. These two datasets come from a different stream of internet data than Laion. A selection of random image-caption pairs from both kinds of datasets is provided in Figure 4. As can be seen from the examples, the first sentence in each caption usually provides a general description of the image, while the other part of the caption often captures more specific details. To clean the data, we removed the starting strings like “This picture depicts/shows/demonstrates:”. Then, since the generated captions tend to be longer than those in the standard datasets like Laion and DataComp (the average caption length in our recaptioned dataset is 78 words), we split the captions into individual sentences as done in [33, 59]. This results in an average sentence length of 16 words for captions generated by CogVLM [53]. We retain only images whose captions contain more than one sentence, which is more than 99% of the 1M set for Laion. This approach favors captions that provide both general context and specific details about the images. For a fair comparison with NegCLIP, we also train CLIC with MS-COCO, the details of which are in Section 4.

**Query:** Can you please describe this image in a long and detailed paragraph? Please keep your descriptions factual and terse but complete. DO NOT add any unnecessary speculation about the things that are not part of the image. The description should be purely factual, with no subjective speculation.

Figure 3: **Query used for re-captioning for the CogVLM model [53].**

Table 7: **List of Universal POS Tags with their Meaning and Examples.** Examples are taken from ChatGPT. For further information, refer to <https://universaldependencies.org/u/pos/>.

POS Tag	Meaning	Example Words
<b>ADJ</b>	Adjective	happy, blue, large
<b>ADP</b>	Adposition (preposition)	in, on, under, with
<b>ADV</b>	Adverb	quickly, very, well
<b>AUX</b>	Auxiliary verb	is, was, will, do
<b>CCONJ</b>	Coordinating conjunction	and, but, or
<b>DET</b>	Determiner	the, a, an, this
<b>INTJ</b>	Interjection	wow, oh, hey
<b>NOUN</b>	Noun (common & proper)	dog, city, John
<b>NUM</b>	Numeral	one, two, 100
<b>PART</b>	Particle	not, 's (as in John's)
<b>PRON</b>	Pronoun	he, she, it, they
<b>PROPN</b>	Proper noun	London, Microsoft
<b>PUNCT</b>	Punctuation	., !, ?
<b>SCONJ</b>	Subordinating conjunction	because, although, if
<b>SYM</b>	Symbol	%, &
<b>VERB</b>	Verb	run, eat, write
<b>X</b>	Other (foreign words, typos)	jdhfk, asdfg

## B.2 Hard negative creation

Our hard negative creation process utilizes the spaCy package to identify the part-of-speech (POS) of words in a sentence. We consider only the coarse-grained POS level, which is universal and not language-specific. The complete list of universal POS tags is provided in Table 7. For each sentence, we classify all words, group them by their POS, and determine eligible categories for swapping. A valid swap nomination requires at least two words from the same POS category, excluding the following: "AUX", "CCONJ", "DET", "INTJ", "PART", "PUNCT", "SCONJ", "SYM" and "X". Once a valid category is identified, we randomly select it and swap two randomly chosen words within that category. We deliberately did not select specific tags to avoid overfitting to current compositionally benchmarks. If no category falls into these cases, we swap two words at random.

## B.3 In-scene vs across-scene swaps

Creating negatives by swapping words between sentences from different scenes offers several advantages over swapping within a single scene:

1. Swapping words within a single sentence can lead to negatives due to the changes in word order. For example, the sentence from NegCLIP: "The horse is eating the grass" can become "The grass is eating the horse". However, swapping words across sentences from different images greatly increases the diversity of negative examples. Taking the example from the paper (Section 1), concatenating the "The horse is eating the grass" image with an image of a dog chasing a cat may lead to the new negative: "The horse is chasing the cat. The dog is eating the grass". Since each image can be paired with each of the other images, this scales the diversity of negatives quadratically and helps the generalization of the method.
2. Creating negatives from a single scene, like in NegCLIP, may result in a modified sentence that still accurately describes the image. This might happen as in many sentences, changing the order of the words does not change the semantic meaning of the sentence. This is different from changing words across scenes, where the concatenated sentence will reflect the concatenated image only if we by chance swap words that have the same meaning (*we remind the reader that we do not allow swaps of the same word, i.e., 'man' can not be swapped with 'man'*). Here are a few examples from the CC12M subset from PixelProse:
  - "In the top left corner, a person wearing a green jacket" – switching *left* and *top* does not change the meaning.

- “there is a tuna salad with celery, onion, and mayonnaise” – changing *onion* and *celery* does not change the meaning.
- “Bugs Bunny, Taz, Lola Bunny, and Daffy Duck are standing on either side” – changing *Lola* and *Bugs*.
- “The Jeep has a black bumper, black wheels” – changing *bumper* and *wheels*.
- “A dining room and living room” – changing *dining* and *living*.
- “A black and gray backpack” – changing *gray* and *black*.

#### B.4 Non-random image concatenation

Combining images at random offers several advantages, such as computational efficiency and reduced risk of overfitting to specific benchmarks. However, to examine whether generating more plausible or grammatically correct negatives leads to better results (at the expense of higher computational resources), we construct negatives in a non-random manner using two strategies:

1. Combining images that share a common noun.
2. Swapping words with different semantic meanings and selecting images accordingly.

##### Common nouns

We begin by extracting all the nouns from the first sentence of each image using spaCy. For each image, we then sample up to five other images (without repetition) that share a common noun and use this to concatenate the images. We then continue as usual by swapping random words (excluding the common noun). Although we witnessed a higher  $S_{neg}$  loss (Eq. (5)), indicating that the hard negatives are indeed harder, the results on the downstream tasks remained largely unchanged (see Table 8, row SAME NOUN).

##### LLM swaps

To test the impact of non-meaningful or ungrammatical negatives, we made the following modification to our method to construct negatives that are grammatically correct by leveraging LLM based swaps:

1. We extracted the fine-grained Part-of-Speech (POS) tags for each word in the first sentence using spaCy’s most computationally expensive model, which took around 5 hours. In contrast, the version used for the random image-caption pairs employed a more computationally efficient model and took only a few minutes.
2. For each word, we queried Gemini 2.5-flash [5] to generate up to five replacement candidates from the same POS category, such that replacing the word changes the meaning of the caption, while keeping the sentence coherent. **Note:** these queries to Gemini via API calls took over 24 hours of compute time to find word replacements for the 1M Laion subset. In comparison, fine-tuning a CLIP model with CLIC only takes around 40 minutes.
3. During training, for each caption, we randomly selected a POS category and then randomly chose a word from that category. The second image and word replacement were selected based on the candidates from step 2.

This significantly more expensive variant of CLIC (LLM SWAPS in Table 8) results in harder negatives (as evidenced by a higher loss during training) and leads to improvements in Replace (ITT) for SugarCrep++ but degrades Swap (ITT) and TOT. While at first surprising, we hypothesize that the lower diversity of negatives compared to our original computationally cheap version of CLIC, with random replacement of words from the same POS category, could be a reason for this.

To check the quality of the negatives, we hand-labelled 75 negative samples from both LLM SWAPS and the random negatives used to train **CLIC-LAION**. This process yielded 69% meaningful, grammatically correct negatives with distinct semantic meanings from LLM SWAPS, compared to 28% from the **CLIC-LAION**. We include below 5 coherent and grammatically correct hard negatives examples, and 5 nonsensical or grammatically incorrect ones. While there are some errors, the overall quality is sufficient to prevent the model from relying solely on shortcuts, as many examples remain meaningful and grammatically correct, and the errors are relatively subtle.

Correct hard negative examples:

(postcard  $\leftrightarrow$  telegram)

N: a collage of various telegram marketing materials. a vintage postcard.

(stripes  $\leftrightarrow$  swirls)

N: a black satin jacket with white swirls on the collar and cuffs. a graphic design element that features a large red heart at the center surrounded by intricate golden stripes and patterns

(short  $\leftrightarrow$  thick)

N: a portrait of a woman with thick textured hair that has a mix of brown and blonde colors. an indoor setting possibly a café or a bakery with a man pouring a short brown liquid which appears to be chocolate onto a white countertop.

(plain  $\leftrightarrow$  fancy)

N: a woman standing against a fancy background. a scrapbooking paper pack titled ‘sunny memories’ by plain pants designs.

(corners  $\leftrightarrow$  hearts)

N: a pair of red envelopes with intricate gold embroidery on the hearts. two wooden objects that are shaped like corners.

Nonsensical or grammatically incorrect samples:

(whistle  $\leftrightarrow$  shout)

N: a red kettle with a shout attached to its spout. a man in a dramatic pose seemingly in the middle of a punch or a whistle

(made  $\leftrightarrow$  destroyed)

N: a hand-destroyed card with a floral design. a scene of destruction with a building that appears to have been damaged or made.

(printed  $\leftrightarrow$  removed)

N: a black bag with white lines and the word ‘thule’ removed on it. one of the nuts is shown with its cap printed revealing the coiled thread inside.

(setting  $\leftrightarrow$  breaking)

N: this image captures a moment in an outdoor park breaking where two individuals are engaged in a playful activity. a bear’s face that appears to be setting through a shattered glass surface.

(appears  $\leftrightarrow$  disappears)

N: it disappears to be a motorized treadmill with a digital display on the top. the headline reads ‘german billionaire appears on matterhorn’ and it is dated april 11 2018 at 11:35 am cdt.

Table 8: **Comparing with non-random image concatenation.** All methods use LAION data captioned by CogVLM and the ViT-B/32 architecture. Although training with non-random image concatenation leads to higher  $\mathcal{L}_{S-Neg}$  loss (Eq. (5)), the performance of the model remains the same.

Method	Downstream Evaluations						SugarCrepe++				SugarCrepe		
	Classification		Text Ret.		Image Ret.		Replace		Swap		Add	Rep.	Swap
	IMNET	ZS-10	COCO F30k		COCO F30k		ITT	TOT	ITT	TOT	ITT	ITT	ITT
CLIP [45]	63.3	61.4	74.1	95.1	54.6	83.5	69.5	60.5	45.7	25.9	72.9	80.0	62.7
CLIC-LAION	61.7	61.0	75.9	95.0	60.0	86.7	75.6	60.1	61.1	27.9	84.5	84.0	73.7
SAME NOUN	61.7	60.7	75.9	95.0	60.0	86.6	75.0	60.1	61.8	28.5	84.3	83.3	73.0
LLM SWAPS	61.8	61.1	76.1	94.7	59.8	86.7	76.7	59.9	60.7	24.0	83.3	83.9	72.1

## C Algorithmic details

---

**Algorithm 1** Training Procedure with Concatenated Images and Hard Negatives

---

```

1: function GENERATEPOSNEG(image pair  $(x_i, x_{i+m})$ , caption pair  $(y_i, y_{i+m})$ )
2:    $u_i = \text{RandomConcat}(x_i, x_{i+m})$ 
3:   Extract the sentences from the captions  $y_i^1, y_i^2, \dots, y_{i+m}^1, y_{i+m}^2, \dots$ 
4:   % Positive Generation
5:   Create  $t_i^{p1} = \text{Concat}(y_i^1, y_{i+m}^1)$  % Concatenate first sentences
6:   Create  $t_i^{p2} = \text{Concat}(y_{i+m}^1, y_i^1)$  % Shuffle order
7:   Create  $t_i^{p3} = \text{RandomConcat}(y_i^{k1}, y_{i+m}^{k2})$  for random  $k_1, k_2$  % Random order of random
   sentence pair
8:   Create  $t_i^{p4} = \text{RandomConcat}(y_i^{k3}, y_{i+m}^{k4})$  for random  $k_3, k_4$  % Random order of random
   sentence pair
9:   % Hard Negative Generation
10:  Extract linguistic categories for words in  $y_i^1$  and  $y_{i+m}^1$  using spaCy
11:  if common category exist then
12:    Select a random common category CAT (e.g. NOUN, ADJ, ...)
13:    Randomly select CAT words  $w_i, w_{i+m}$  from  $y_i^1, y_{i+m}^1$  respectively
14:  else
15:    Randomly select words  $w_i, w_{i+m}$  from  $y_i^1, y_{i+m}^1$  respectively
16:  end if
17:  Swap  $w_i$  and  $w_{i+m}$  in  $t_i$  to create  $t_i^n = \text{Swap}(t_i, w_i, w_{i+m})$ 
18:  return  $u_i, t_i^{p1}, t_i^{p2}, t_i^{p3}, t_i^{p4}, t_i^n$ 
19: end function

```

**Require:** Training images  $I$ , captions  $T$ , hyperparameters  $\lambda_{\text{Cont}}, \lambda_{\text{S-Neg}}, \lambda_{\text{Uni}}$

```

20: for each iteration do
21:   Sample a batch of images and corresponding captions  $\{x_i, y_i\}_{i=1}^m$ 
22:   if iteration % 2 = 0 then
23:     % Concatenated training images
24:     for each image caption pair  $x_i, y_i$  in the batch do
25:        $u_i, t_i^{p1}, t_i^{p2}, t_i^{p3}, t_i^{p4}, t_i^n = \text{GeneratePosNeg}((x_i, x_{i+m}), (y_i, y_{i+m}))$ 
26:     end for
27:     % Compute loss (Contrastive, Hard Negative, Uni Modal)
28:      $\mathcal{L} = \lambda_{\text{Cont}}\mathcal{L}_{\text{Cont}} + \lambda_{\text{S-Neg}}\mathcal{L}_{\text{S-Neg}} + \lambda_{\text{Uni}}\mathcal{L}_{\text{Uni}}$ 
29:   else
30:     % Standard CLIP training
31:     for each image caption pair  $x_i, y_i$  in the batch do
32:       Extract first sentence of caption  $y_i^1$ 
33:     end for
34:     Compute standard CLIP contrastive loss
35:   end if
36:   Update model parameters
37: end for

```

---

In this section, we present a pseudo-code of our algorithm, as discussed in Section 3 in the main paper. The pseudo-code can be found in Algorithm 1.

## D Experimental details and discussions

In this section, we give a detailed description of various training and baseline design choices.

### D.1 Further training details and discussions

As stated in the main paper, we train with alternating steps of CLIP loss (Eq. (1)) and the proposed loss (Eq. (7)). This decision is made to help the fine-tuned model retain its performance on downstream tasks, evident from the ablations in Table 4.

**Computational resources.** All the work was carried out on A100 40G GPUs. The training runs are across 4 GPUs. Running CLIC on smaller architectures like ViT-B/32 and ViT-B/16 took less than 30 minutes for the Laion dataset. Datasets such as RedCaps took longer due to high-resolution images. The larger ViT-L/14 models took around 1 hour per dataset. The evaluation time is negligible, in the order of 20 minutes per model for all datasets on a single GPU.

**Training parameters and details.** In all of the experiments, the loss parameters in Eq. (7) are set to  $\lambda_{\text{Cont}} = 1/2$ ,  $\lambda_{\text{S-Neg}} = 1/2$  and  $\lambda_{\text{Uni}} = 1$ . We train for one epoch on our 1M Laion subset and on our 850k PixelProse dataset, while for MS-COCO, we trained for five epochs; ablation on the number of epochs can be found in Table 19. All experiments are conducted at an image resolution of  $224 \times 224$ . Our effective batch size is  $200 \times 4$ , and we use a cosine scheduler, where the warm-up phase is 20% of the training time. The learning rate (LR) starts at  $1e-7$ , peaks at  $1e-6$ , and arrives at  $1e-8$  at the end. We used the standard AdamW [34] optimizer with beta parameters (0.9, 0.98) and  $\epsilon$  set to  $1e-8$  with a weight decay of 0.1.

**Training with COCO.** Although for most of the experiments, we use our 1M Laion subset or the PixelProse dataset, we also train with MS-COCO to be comparable to the original NegCLIP checkpoint from [56]. Note that due to training on MS-COCO all retrieval numbers on MS-COCO for NegCLIP from [56], and CLIC-COCO are no longer zero-shot. The same is also true to CoN-CLIP, who use COCO images as distractor in their method.

**Our NegCLIP baselines.** To have a baseline for all models considered in this work, we train our own version of NegCLIP (denoted everywhere with  $^\dagger$ ). It is used only when there are no available checkpoints (Table 3, Table 24 and Table 25), and in our ablation study (Table 20). This version of NegCLIP differs from the NegCLIP [56] as follows:

1. We do not use hard negative images that are added to the batch.
2. We use the same categories as in our method, while the NegClip paper employs six tags: two from the coarse-grained POS level ("ADV" and "ADJ"), two from the fine-grained level ("NN" for singular nouns and "NNS" for plural nouns), and two phrase-level categories (verb phrases and noun phrases).
3. The vision encoder is frozen during training.

We use their negative loss calculation (see Eq. (2)), where the hard negatives are part of the standard clip loss.

## D.2 Zero-shot datasets

The 10 zero-shot classification datasets we use are a subset from the CLIP\_benchmark<sup>2</sup>. Specifically, we use 1k images of each of the following datasets: Country-211 [44], Caltech-101 [15], OxfordPets [40], DTD [4], FGV Aircafts [37], StanfordCars [22], Cifar-10,100 [23], Food-101 [2].

## D.3 Dataset from other methods

To improve compositionality, many methods use re-labelled/captioned versions of standard datasets. The details of these recaptions can be found in Table 9. TripletCLIP uses recaptioned data from LaCLIP and DAC uses BLIP2 [25] to create better captions. Methods like CoN-CLIP and SVLC-R+L use the original captions and PaLM-2 and BERT [7] to generate hard-negatives for their base datasets.

## D.4 Other methods do not generalize

From the results on SugarCrepe++ and SugarCrepe, one sees that other compositionality enhancing methods like SVLC-R, DAC, etc., perform relatively well on SugarCrepe but fail to generalize to SugarCrepe++. We believe this is due to the way these methods operate. Specifically, most of these methods create hard-negatives aimed at certain attributes such as color, position, or size (for example, modifying P1: "a woman plays a black guitar" to N: "a woman plays a green guitar"). This aligns closely with what benchmarks like ARO, and SugarCrepe evaluate (with SugarCrepe generating more fluent and sensical negatives).

<sup>2</sup>[https://github.com/LAION-AI/CLIP\\_benchmark](https://github.com/LAION-AI/CLIP_benchmark)

Table 9: **Recaptioning/negative generation for different methods.** We show the different datasets and the respective re-captioning model used by compositionally enhancing methods.

Method	Source	Base Dataset	Re-caption Method
NegCLIP	[56]	MS-COCO	-
DAC-LLM	[9]	CC3M	BLIP2 [25]
DAC-SAM	[9]	CC3M	BLIP2 [25]
SVLC-R	[10]	CC3M	-
SVLC-R+L	[10]	CC3M	BERT [7]
CoN-CLIP	[49]	CC3M	PaLM-2 [1]
TripletCLIP	[41]	CC3M, CC12M	LaCLIP [12]
CLIC-COCO	Ours	MS-COCO	-
CLIC-LAION	Ours	Laion	CogVLM [53]
CLIC-CC12M	Ours	CC12M	PixelProse [50]
CLIC-RedCaps	Ours	RedCaps	PixelProse [50]

While such approaches can indeed improve performance on lexical understanding, they fail to capture word replacements that are not in the dictionary and fall outside of a predefined structure. More importantly, we believe these changes do not foster a deeper, more nuanced understanding of compositionality within the model. True compositional reasoning requires distinguishing between examples that differ not only lexically but also syntactically and semantically, for instance, recognizing that N2: “a woman plays a plastic guitar” is negative, whereas P2: “a guitar is being played by a woman” is positive. These cases involve more complex syntactic and semantic variations and move beyond lexical differences.

For SugarCrepe++, in addition to the evaluation in SugarCrepe, an additional lexically different text is introduced, and since these methods do not account for general compositional/structural variations in their hard-negative, they fall short on SugarCrepe++. On the other hand, the random word swapping in CLIC makes our method focus not on the specific attributes but the general composition of different words in the text. This makes our method generalize better and work equally well for both SugarCrepe++ and SugarCrepe.

## D.5 Detailing the LLaVA experiment

As LLaVA uses only the vision encoder from CLIP models trained at a higher image resolution ( $336 \times 336$  pixels), we performed an additional fine-tuning of ViT-L/14 with CLIC, this time also unfreezing the vision encoder. We employed the PixelProse dataset and reduced the effective batch size to  $40 \times 8$ , while keeping all other settings identical to those in Appendix D.1.

Subsequently, we fine-tuned LLaVA-1.5 using Vicuna-7b as the LLM. Initially, only the multi-modal projector was fine-tuned, followed by fine-tuning of both the projector and the LLM. This procedure follows the standard setup from the official LLaVA codebase<sup>3</sup>. To ensure a fair comparison, we conducted these experiments using both the original CLIP and our CLIC vision encoders. Hence, the two resulting models in Table 6 are both fine-tuned by us, making the subsequent comparison fair.

For the evaluation reported in Table 6, we adopt the VQAScore from Lin et al. [30] for both SugarCrepe++ and WinoGround. For standard VLM benchmarks, we select a subset of tasks from the original LLaVA codebase. Specifically, we assess question answering using accuracy on GQA [19] and TextVQA [47], chain-of-thought reasoning with ScienceQA-Images (SQA-I) [35], and perception via MME [54]. For all evaluations we use the default setup from the original LLaVA codebase.

## E Additional experiments

In this section, we present more detailed results of the experiments in Section 4 in the main paper and provide additional experiments.

<sup>3</sup><https://github.com/haotian-liu/LLaVA/tree/main>



**Extension of results from the main paper** As we only show averaged ITT and TOT numbers across different sets of SugarCrepe++ and SugarCrepe in the main paper, the full versions for ViT-B/32 models can be found in Table 17 and Table 18 resp. Similarly, for ViT-B/16 (omitted in the main part) and ViT-L/14, detailed results are shown in Table 24 and Table 25, respectively. Detailed compositionality results for the newer CLIP versions in CLIPA, EVA02-CLIP and CLIPS with their respective better versions achieved via CLIC can be found in Table 26. In Table 23, we give dataset wise accuracy of all the large models for the ZS-10 setting. WinoGround results for large model can be found in Table 27. The pre-trained checkpoints used for fine-tuning with CLIC are listed in Table 22.

**Improvements across architectures** In Table 3 in the main part of the paper, we show how CLIC improves compositionality and retrieval for differently pre-trained models. We visualize these improvements in Figure 5. Fine-tuning with CLIC yields improvements on all models for ITT on SugarCrepe++ and SugarCrepe. The improvements in case of CLIP are as high as 7% and as much as 9% for CLIPS.

**Ablating training steps** In Table 19, we show how the proposed CLIC changes with more training steps. Here, we use MS-COCO for training which means 5 epochs corresponds to 2.5 Laion epochs of our dataset. From the table, we see increasing gains in SugarCrepe++ numbers when increasing the number of training step. However, this comes with a marginal degradation in zero-shot classification, which goes down from 63.2% to 62.7% on IMAGENET as we move from 2 to 5 epochs. The same trend holds for our 1M set of Laion, where training for more than 1 epochs led to decay on IMAGENET and ZS-10 numbers with marginal gains on SugarCrepe++. Hence, for our runs in the main part of the paper with Laion (CLIC-LAION) we only train for 1 epoch.

**Ablating freezing of model components** In Table 20, we show how the proposed CLIC (frozen vision encoder) yields, on average, the best compositionality without sacrificing downstream performance. From the table, it is clear that freezing just the text encoder leads to the least amount of improvement on compositional benchmarks. Fine-tuning the whole model works better (in terms of retrieval) but the gains on SugarCrepe++ are smaller than freezing the vision encoder and it also degrades zero-shot classification performance. From these results, we infer that the most improving configuration is freezing the vision encoder, which is the final setting for CLIC. This also highlights that substantial gains can be made by improving the text encoder alone.

**Ablating number of additional positives** In Table 21, we show the impact of adding additional positives for each concatenated image ( $p_3$  and  $p_4$  from Section 3.1). Notably, increasing the number of positives has opposing effects on the TOT and ITT scores. In addition, after two positives, the effects on ITT in SugarCrepe++ become minimal, so we chose to add only two positives.

**Error bars** To show the resilience to randomness of CLIC, we conduct three runs for two versions of CLIC presented in Table 1 and Table 2 (CLIC-LAION and CLIC-RedCaps). We report the mean and standard deviation in Table 10 and Table 11, respectively. The standard deviation is small, particularly when considering that the results in the paper are rounded to one decimal place.

Table 10: **Small standard deviation across runs.** Results are reported as mean  $\pm$  std over three independent runs on compositionality benchmarks with the ViT-B/32 architecture.

Method	SugarCrepe++ [11]				WinoGround [52]			SugarCrepe [18]		
	Replace		Swap		Text	Image	Group	Add	Replace	Swap
	ITT	TOT	ITT	TOT	Score	Score	Score	ITT	ITT	ITT
CLIP	69.5	60.5	45.7	25.9	31.2	11.0	8.7	72.9	80.0	62.7
CLIC-LAION	75.34 $\pm$ 0.18	60.10 $\pm$ 0.11	61.55 $\pm$ 0.38	27.86 $\pm$ 0.08	31.75 $\pm$ 0.20	11.83 $\pm$ 0.31	9.33 $\pm$ 0.31	84.56 $\pm$ 0.09	83.84 $\pm$ 0.14	73.79 $\pm$ 0.12
CLIC-RedCaps	76.18 $\pm$ 0.16	57.82 $\pm$ 0.17	61.60 $\pm$ 0.07	23.34 $\pm$ 0.19	32.00 $\pm$ 0.61	11.67 $\pm$ 0.12	10.08 $\pm$ 0.24	86.27 $\pm$ 0.15	84.80 $\pm$ 0.04	72.54 $\pm$ 0.04

**Long captions retrieval** To evaluate the influence of exploiting superficial artifacts such as potential abrupt topic changes, generated by our concatenation scheme, we compared the retrieval performance on longer captions (up to 200 tokens) between the original model, other compositionality-enhancing methods, and our fine-tuned models (CLIC) on the ViT-B/32 architecture. This setup is adapted



Table 11: **Small standard deviation across runs.** Results are reported as mean  $\pm$  std over three independent runs on downstream tasks with the ViT-B/32 architecture.

Method	Classification		$I \rightarrow T$		$T \rightarrow I$	
	IMNET	ZS10	COCO	F30K	COCO	F30K
CLIP	63.3	61.4	74.1	95.1	54.6	83.5
CLIC-LAION	61.72 $\pm$ 0.02	60.89 $\pm$ 0.06	75.92 $\pm$ 0.17	95.10 $\pm$ 0.08	60.32 $\pm$ 0.64	86.70 $\pm$ 0.03
CLIC-RedCaps	62.41 $\pm$ 0.09	60.22 $\pm$ 0.03	76.00 $\pm$ 0.09	95.67 $\pm$ 0.05	59.42 $\pm$ 0.04	86.27 $\pm$ 0.02

from long-retrieval as previously done in [58]. We truncate the caption to the nearest punctuation (within context-length) as CLIP has a context length of 77. It is important to note that these longer captions describe a single scene but consist of multiple sentences. This design choice was deliberate: it helps us determine if our CLIC models have inadvertently overfit to abrupt topic changes or even to concatenation with punctuation. As can be seen in Table 12, only CLIC and TripletCLIP outperform the base model for both image and text retrieval, whereas DAC-LLM and SVLC-R+L are worse.

Table 12: **CLIC does not overfit to abrupt topic changes.** Evaluation of long-caption retrieval task for the ViT-B/32 architecture.

Method	$I \rightarrow T$	$T \rightarrow I$
CLIP	83.2	79.5
NegCLIP	82.6	80.7
DAC-LLM	67.6	72.5
SVLC-R+L	75.0	74.8
TripletCLIP	84.1	<b>81.5</b>
CLIC-COCO	<b>85.9</b>	<u>81.4</u>
CLIC-LAION	83.9	79.6
CLIC-RedCaps	<u>84.7</u>	79.9

**More retrieval results** In the main part of the paper, all results on MS-COCO and Flickr30k retrieval are reported with Recall@5. In Table 28, we additionally report Recall@1 for all ViT-B/32 models from Table 1. The overall ordering of methods is similar to Recall@5. Similarly, in Table 27, we additionally report Recall@1 for all ViT-L/14 models from Table 3.

**Visual examples and failure case analysis** To illustrate the strengths and limitations of the method and the compositionality task in general, we selected four of the SugarCrepe++ categories: Replace attribute, Replace relation, Swap attribute, and Swap object. For each, we examined the first instance in which the pretrained model’s prediction changed from incorrect to correct with CLIC-RedCaps, and vice versa. The results show that CLIC has lower absolute cosine similarity values compared to the pretrained model.

Table 13: Visual examples illustrating cases where the pretrained model’s prediction changed: from incorrect to correct with our CLIC-RedCaps version, and from correct to incorrect in the **replace attribute** class of SugarCrepes++. ✓ marks correct predictions and ✗ marks incorrect predictions.



$P_1$ : A person holding up a chocolate doughnut with a face drawn on it.  
 $P_2$ : A chocolate doughnut with a face drawn on it is being held up by a person.  
 $N$ : A person holding up a vanilla doughnut with a face drawn on it.

$P_1$ : A tan toilet and sink combination in a small room.  
 $P_2$ : A small room with a tan toilet and sink combination positioned in close proximity to one another.  
 $N$ : A white toilet and sink combination in a small room.

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✗)	0.323	0.306	0.309
CLIC (✓)	0.280	0.285	0.263

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✓)	0.325	0.341	0.319
CLIC (✗)	0.260	0.256	0.262

Table 14: Visual examples illustrating cases where the pretrained model’s prediction changed: from incorrect to correct with our CLIC-RedCaps version, and from correct to incorrect in the **replace relation** class of SugarCrepes++. ✓ marks correct predictions and ✗ marks incorrect predictions.



$P_1$ : A pizza covered in lots of greens on top of a table.  
 $P_2$ : A table has a pizza on top of it covered in lots of greens.  
 $N$ : A pizza covered in lots of greens next to a table.

$P_1$ : A zebra is standing in an open field.  
 $P_2$ : On an open field, stands a zebra.  
 $N$ : A zebra is running across an open field.

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✗)	0.300	0.280	0.293
CLIC (✓)	0.276	0.275	0.272

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✓)	0.311	0.312	0.298
CLIC (✗)	0.283	0.254	0.270

### Laion re-captioned images



An elderly man wearing glasses and a suit standing behind a lectern with a golden eagle design. He appears to be in a formal setting possibly delivering a speech or presentation. The background is dimly lit and there are some decorative elements such as flowers visible behind him.

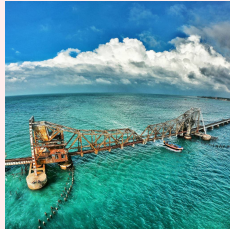


A dimly lit cave or underground setting. The cave has various elements such as stalactites rocks and a chalkboard with some drawings on it. The atmosphere seems mysterious and adventurous suggesting that the characters might be on a quest or exploration.



A bar chart that represents the average sales price of homes in Centerville Georgia over a 12-month period from January 2010 to December 2010. The chart shows two sets of data: the average sales price of homes and the average sales price to list price ratio.

### PixelProse images



In the foreground, there is a long bridge with a train on it. The bridge is made of metal and has a brown color. The train is red and white. Behind the bridge, there is a large body of water. The water is a blue-green color. There are some clouds in the sky. The image displays the Pamban Bridge, which is a still active train bridge located in Rameshwaram, India.



A pink and white frangipani flower. The flower is made up of five petals, with the pink petals on the outside and the white petals on the inside. The flower is surrounded by green leaves.



A small Bengal cat with brown fur and black stripes. The cat is lying on a cat tree, with its front paws hanging off the edge of the platform. The cat is looking at the camera with a curious expression.

Figure 4: **Random images from Laion and PixelProse, after re-captioning with CogVLM [53] and re-captions from Singla et al. [50].** For each caption, the first sentence and another two randomly sampled sentences are presented. The first sentence often describes the entire image, while the additional sentences highlight specific details. As can be seen, the captions generated by CogVLM [53] exhibit high quality similar to the PixelProse dataset. Note these models still hallucinate (e.g. train on the bridge) sometimes and not all captions are 100% correct.

Table 15: Visual examples illustrating cases where the pretrained model’s prediction changed: from incorrect to correct with our CLIC-RedCaps version, and from correct to incorrect in the **swap attribute** class of SugarCrepes++. ✓ marks correct predictions and ✗ marks incorrect predictions.



$P_1$ : One apple and several oranges sit in a bowl.

$P_2$ : several oranges and an apple are positioned in a bowl.

$N$ : Several apples and one orange sit in a bowl.

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✗)	0.306	0.315	0.309
CLIC (✓)	0.276	0.298	0.270

$P_1$ : Two giraffe and a zebra are standing in a field.

$P_2$ : In a field, two giraffes and a zebra are standing.

$N$ : A giraffe and two zebras are standing in a field.

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✓)	0.316	0.316	0.315
CLIC (✗)	0.305	0.290	0.295

Table 16: Visual examples illustrating cases where the pretrained model’s prediction changed: from incorrect to correct with our CLIC-RedCaps version, and from correct to incorrect in the **swap object** class of SugarCrepes++. ✓ marks correct predictions and ✗ marks incorrect predictions.



$P_1$ : Girls wash a motorcycle while men look on.

$P_2$ : The motorcycle is being washed by girls while men observe.

$N$ : Men wash a motorcycle while girls look on.

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✗)	0.365	0.356	0.358
CLIC (✓)	0.364	0.371	0.362

$P_1$ : A painting of a vase with a sunflower on a table.

$P_2$ : A vase containing a sunflower is positioned on a table in a painting.

$N$ : A painting of a sunflower with a vase on a table.

Algorithm	$\cos(P_1)$	$\cos(P_2)$	$\cos(N)$
CLIP (✓)	0.378	0.381	0.378
CLIC (✗)	0.317	0.321	0.322

Table 17: **Extended results for SugarCrepe++ for ViT-B/32 architecture.** This table is an extension of Table 1.

Method	Replace-obj		Replace-att		Replace-rel		Swap-obj		Swap-att		AVG	
	ITT	TOT	ITT	TOT	ITT	TOT	ITT	TOT	ITT	TOT	ITT	TOT
CLIP	86.8	83.7	65.6	59.3	56.2	38.6	46.1	19.2	45.2	32.7	60.0	46.7
NegCLIP	89.6	<u>94.5</u>	69.5	<u>76.4</u>	52.3	51.4	54.7	<u>33.1</u>	58.8	<u>56.7</u>	64.8	<u>62.5</u>
DAC-LLM	65.7	76.8	47.7	59.5	47.6	42.3	31.4	11.4	32.9	24.8	45.1	43.0
DAC-SAM	64.3	75.8	43.9	56.1	48.6	48.6	27.7	11.4	33.6	25.4	43.6	43.5
SVLC-R	82.9	89.5	61.5	67.4	47.6	51.5	49.4	20.4	53.3	36.3	58.9	53.0
SVLC-R+L	80.9	91.6	57.1	67.0	47.3	51.3	42.8	18.4	48.9	34.4	55.4	52.5
CoN-CLIP	88.1	91.5	66.6	69.4	52.3	51.6	44.1	22.4	51.9	42.6	60.6	55.5
TripletCLIP	86.8	92.3	71.7	74.0	<u>61.9</u>	<u>51.7</u>	38.8	24.1	47.9	42.3	61.5	56.9
CLIC-COCO	<b>90.2</b>	<b>97.3</b>	72.3	<b>81.6</b>	58.0	<b>59.3</b>	48.6	<b>33.5</b>	57.2	<b>60.8</b>	65.3	<b>66.5</b>
CLIC-LAION	90.0	84.5	<b>75.8</b>	57.9	61.0	38.0	<b>61.2</b>	22.4	<u>61.0</u>	33.4	<u>69.7</u>	47.2
CLIC-RedCaps	<u>90.1</u>	84.3	<u>75.1</u>	52.4	<b>62.9</b>	36.3	<u>60.8</u>	17.9	<b>62.4</b>	28.2	<b>70.3</b>	34.6

Table 18: **Extended results for SugarCrepe for ViT-B/32 architecture.** This table is an extension of Table 1.

Methods	IMAGENET	Add		Replace			Swap	
		Obj.	Att.	Obj.	Att.	Rel.	Obj.	Att.
CLIP	<b>63.3</b>	77.2	68.6	90.9	80.1	69.1	61.2	64.3
NegCLIP	60.9	<u>88.8</u>	84.7	93.8	<u>87.7</u>	73.9	<b>75.5</b>	<b>76.4</b>
DAC-LLM	51.1	<b>89.6</b>	<b>97.7</b>	<b>94.4</b>	<b>89.3</b>	<b>84.4</b>	<u>75.1</u>	74.2
DAC-SAM	52.3	87.5	<u>95.5</u>	91.2	85.9	<u>83.9</u>	71.8	<u>75.4</u>
SVLC-R	58.8	79.4	91.2	91.3	81.2	64.2	68.6	69.1
SVLC-R+L	59.7	75.8	81.1	88.1	76.8	62.6	64.1	66.7
CoN-CLIP	<u>63.2</u>	87.9	78.0	91.8	81.0	66.3	63.7	67.0
TripletCLIP	54.7	87.3	85.8	<b>94.4</b>	86.7	82.8	66.5	72.7
CLIC-COCO	62.7	<u>88.8</u>	82.4	93.6	85.9	71.6	65.7	72.8
CLIC-LAION	61.7	81.6	87.4	93.4	86.3	72.4	73.1	74.2
CLIC-RedCaps	62.3	84.2	88.7	93.3	86.9	74.0	71.8	73.3

Table 19: **Witnessing the effect of training steps on CLIC.** Although with more training steps, SugarCrepe++ numbers improve, it comes at the marginal degradation in downstream classification tasks. Hence, we do not train for more steps. \* are not zero-shot for MS-COCO evaluations.

Method	Fine-tuning Epochs	Downstream Evaluations						SugarCrepe++				SugarCrepe		
		Classification		Text Ret.		Image Ret.		Replace		Swap		Add		Rep. Swap
		IMNET	ZS-10	COCO	F30k	COCO	F30k	ITT	TOT	ITT	TOT	ITT	ITT	ITT
CLIP [45]	-	63.3	61.4	74.1	95.1	54.6	83.5	69.5	60.5	45.7	25.9	72.9	80.0	62.7
CLIC-COCO*	2	63.2	60.9	81.7	96.6	66.3	89.8	73.0	78.3	53.0	45.4	85.6	83.3	69.9
CLIC-COCO*	3	62.9	60.9	82.3	97.1	67.5	90.3	73.5	78.7	52.6	45.8	85.2	83.5	70.1
CLIC-COCO*	5	62.7	60.7	82.9	97.1	68.2	90.2	73.5	79.4	52.9	47.2	85.6	83.7	69.3

Table 20: **Effect of freezing/unfreezing different model components on CLIC for ViT-B/32.** All these runs are done with the CLIC-LAION version, and overall CLIC with frozen vision encoder works the best.

Method	Frozen	Downstream Evaluations						SugarCrepe++				SugarCrepe		
		Classification		Text Ret.		Image Ret.		Repl.		Swap		Add	Repl.	Swap
		IMNET	ZS-10	COCO	F30k	COCO	F30k	ITT	TOT	ITT	TOT			
CLIP [45]		63.3	61.4	74.1	95.1	54.6	83.5	69.5	60.5	45.7	25.9	72.9	80.0	62.7
CLIC	None	60.5	60.8	77.7	94.7	62.1	87.4	72.2	60.0	60.0	25.1	84.5	83.6	71.9
CLIC	Text	60.2	60.3	75.8	93.4	61.1	86.5	72.5	60.5	49.8	25.9	76.8	80.1	64.4
CLIC	Vision	61.7	61.0	75.9	95.0	60.0	86.7	75.6	60.1	61.1	27.9	84.5	84.0	73.7

Table 21: **Effect of adding additional positives on CLIC for ViT-B/32.** All experiments use the CLIC-LAION version, with the negative caption ( $n$ ). The column “Positives” reports addition of more positives ( $p_2, p_3, p_4, p_5$ ) from Section 3.1. Beyond four positives, changes are marginal, so we chose this configuration.

Method	Positives	Downstream Evaluations						SugarCrepe++				SugarCrepe		
		Classification		Text Ret.		Image Ret.		Replace		Swap		Add	Replace	Swap
		IMNET	ZS-10	COCO	F30k	COCO	F30k	ITT	TOT	ITT	TOT			
CLIP [45]		63.3	61.4	74.1	95.1	54.6	83.5	69.5	60.5	45.7	25.9	72.9	80.0	62.7
CLIC	$p_1$	62.0	61.3	75.5	94.4	60.0	86.4	69.0	67.1	51.8	35.8	85.9	83.1	70.9
CLIC	$+p_2$	61.8	61.2	75.5	94.6	59.9	86.5	69.7	68.6	53.3	35.3	85.6	82.8	72.3
CLIC	$+p_3$	61.7	60.8	75.8	94.9	59.8	86.6	73.8	61.7	58.7	30.2	84.5	83.8	72.6
CLIC	$+p_4$	61.7	61.0	75.9	95.0	60.0	86.7	75.6	60.1	61.1	27.9	84.5	84.0	73.7
CLIC	$+p_5$	61.7	60.7	76.2	95.1	59.8	86.7	75.6	59.3	61.5	27.0	84.1	83.6	73.0

Table 22: **Pre-training/Evaluation checkpoint locations.** Key values for the pre-trained models, taken from different sources.

Model	Source	Key
ViT-B/32	OpenClip	openai
ViT-B/16	OpenClip	openai
ViT-L/14	OpenClip	openai
NegCLIP	GitHub	mertyg/vision-language-models-are-bows
DAC-SAM	GitHub	SivanDoveh/DAC
SVLC-R+L	GitHub	SivanDoveh/TSVLC
CoN-CLIP	GitHub	jaisidhsingh/CoN-CLIP
TripletCLIP	GitHub	tripletclip/TripletCLIP
EVA-CLIP	OpenClip	eva02_large_patch14_clip_224
CLIPA	HF-Hub	UCSC-VLAA / hf-hub:UCSC-VLAA/ViT-L-14-CLIPA-datacomp1B
CLIPS	HF-Hub	UCSC-VLAA / ViT-L-14-CLIPS-224-Recap-DataComp-1B



Table 23: **Zeroshot image classification.** We report the zero-shot image classification performance for different models.

Model	Source	Food101	Cifar10	Cifar100	Cars	FGVS	DTD	Pets	Caltech101	Flowers	Country211	Mean
CLIP	[45]	93.6	96.4	76.3	76.5	30.4	54.6	94.9	86.4	80.6	33.1	72.3
CoN-CLIP	[49]	93.9	96.0	79.4	76.3	30.4	56.8	93.6	86.8	78.1	33.0	72.4
CLIC-LAION	ours	93.3	96.4	78.3	75.0	34.1	56.5	89.6	86.6	75.1	33.6	71.8
CLIC-RedCaps	ours	92.5	96.3	78.0	76.0	30.8	56.6	91.7	87.3	78.6	32.9	72.1
CLIPA	[26]	94.9	98.7	88.7	92.7	42.2	69.5	95.0	88.5	81.8	31.3	78.3
CLIC-LAION	ours	93.9	98.8	88.4	92.9	43.1	67.8	95.7	88.4	80.7	30.8	78.0
CLIC-RedCaps	ours	94.5	98.4	88.0	93.3	41.9	69.5	95.9	89.3	80.6	31.5	78.3
EVA-CLIP	[51]	94.1	99.7	90.6	89.7	37.4	61.8	94.7	88.6	77.0	32.7	76.6
CLIC-LAION	ours	94.1	99.7	90.6	89.7	37.4	61.8	94.7	88.6	77.0	32.7	76.6
CLIC-RedCaps	ours	92.9	99.4	89.9	88.8	33.4	62.6	92.3	89.1	77.4	32.5	75.8
CLIPS	[33]	93.2	98.2	86.9	91.6	39.4	66.2	94.9	86.9	78.2	29.4	76.5
CLIC-LAION	ours	91.8	98.1	87.1	90.6	37.6	64.2	93.8	86.8	78.1	27.2	75.5
CLIC-RedCaps	ours	92.7	98.0	86.8	91.2	37.7	67.1	94.5	87.8	80.8	30.0	76.7

Table 24: **Compositionality and downstream results for ViT-B/16.** We present a detailed set of results for the pre-trained CLIP, our version of NegCLIP, CLIP fine-tuned by CoN-CLIP, CLIC-LAION and CLIC-RedCaps (using pre-trained CLIP). For SugarCrepe++, we report only the ITT scores in detail. We note here that models with \* are not zero-shot for MS-COCO evaluations.

Model	Source	ZS-Class.		Text-Ret.		Image-Ret.		
		IMNET	Avg.	COCO	F30k	COCO	F30k	
CLIP	[45]	<u>68.3</u>	<u>65.5</u>	76.2	96.4	57.6	85.6	
NegCLIP <sup>†</sup>	Ours	64.5	64.0	74.2	94.7	<b>65.8</b>	<b>89.7</b>	
CoN-CLIP*	[49]	<b>68.9</b>	<b>69.1</b>	73.9	93.5	60.1	88.1	
CLIC-LAION	Ours	66.5	62.3	<u>77.6</u>	<u>96.9</u>	<u>63.6</u>	<u>88.9</u>	
CLIC-RedCaps	Ours	67.3	64.4	<b>79.0</b>	<b>97.0</b>	62.0	88.0	
Compositionality: SugarCrepe++								
Model	Source	Rep-obj	Rep-Att	Rep-Rel	Swap-obj	Swap-Att	Avg-ITT	Avg-TOT
CLIP	[45]	89.6	67.5	53.1	39.6	48.3	59.6	45.7
NegCLIP <sup>†</sup>	Ours	88.7	66.0	46.9	47.3	53.7	60.5	<u>51.9</u>
CoN-CLIP	[49]	88.1	66.6	52.3	44.1	51.9	60.5	<b>55.5</b>
CLIC-LAION	Ours	<b>91.6</b>	<u>75.9</u>	<u>62.5</u>	<b>55.9</b>	<b>62.0</b>	<u>69.6</u>	45.7
CLIC-RedCaps	Ours	<u>91.5</u>	<b>76.0</b>	<b>64.5</b>	<b>55.9</b>	<u>61.1</u>	<b>69.8</b>	38.6
Compositionality: SugarCrepe								
		Add-obj	Add-Att	Rep-Obj	Rep-Att	Rep-Rel	Swap-Obj	Swap-Att
CLIP	[45]	78.4	66.8	93.5	81.0	66.6	60.0	65.0
NegCLIP <sup>†</sup>	Ours	80.0	87.3	93.9	82.2	68.1	<u>69.0</u>	70.1
CoN-CLIP	[49]	<u>87.3</u>	79.6	93.6	81.0	53.3	59.2	65.2
CLIC-LAION	Ours	<b>88.4</b>	<b>91.0</b>	<b>95.6</b>	<b>86.5</b>	<b>75.5</b>	<b>71.0</b>	<b>73.9</b>
CLIC-RedCaps	Ours	87.2	<u>88.6</u>	<u>94.9</u>	<u>85.7</u>	<u>73.3</u>	<u>69.0</u>	<u>71.3</u>

Table 25: **Comparison of Sugarcrepe and Sugarcrepe++ along with Downstream Retrieval/Classification Performance for ViT-L/14.** We present in this table a detailed set of results for the pre-trained CLIP, our version of NegCLIP<sup>†</sup>, CLIP fine-tuned by CoN-CLIP, CLIC-LAION and CLIC-RedCaps (using pre-trained CLIP). For SugarCrepe++, we report only the ITT scores in detail.

Compositionality: SugarCrepe++								
Model	Source	Rep-obj	Rep-Att	Rep-Rel	Swap-obj	Swap-Att	Avg-ITT	Avg-TOT
CLIP	[45]	90.6	67.5	54.0	43.6	45.6	60.3	50.3
NegCLIP <sup>†</sup>	Ours	90.5	68.9	52.3	46.5	54.5	62.5	<u>52.4</u>
CoN-CLIP	[49]	92.2	68.6	55.6	44.1	47.9	61.7	<b>54.9</b>
CLIC-LAION	Ours	<b>94.6</b>	<b>79.6</b>	<b>62.8</b>	<b>59.6</b>	<b>60.0</b>	<b>71.3</b>	46.1
CLIC-RedCaps	Ours	<u>93.5</u>	<u>75.0</u>	<u>59.5</u>	<u>54.7</u>	<u>55.7</u>	<u>67.7</u>	41.3

Compositionality: SugarCrepe								
		Add-obj	Add-Att	Rep-Obj	Rep-Att	Rep-Rel	Swap-Obj	Swap-Att
CLIP	[45]	78.3	71.5	94.1	79.2	65.1	60.4	62.3
NegCLIP <sup>†</sup>	Ours	81.2	<u>86.0</u>	94.3	<u>83.1</u>	<u>71.1</u>	<u>69.4</u>	<b>70.6</b>
CoN-CLIP	[49]	<b>90.2</b>	77.6	95.2	81.7	67.0	65.3	63.1
CLIC-LAION	Ours	<u>86.1</u>	<b>89.3</b>	<b>96.5</b>	<b>86.4</b>	<b>73.8</b>	<b>71.0</b>	<u>71.6</u>
CLIC-RedCaps	Ours	85.8	82.8	<u>95.4</u>	82.9	70.0	64.5	67.7

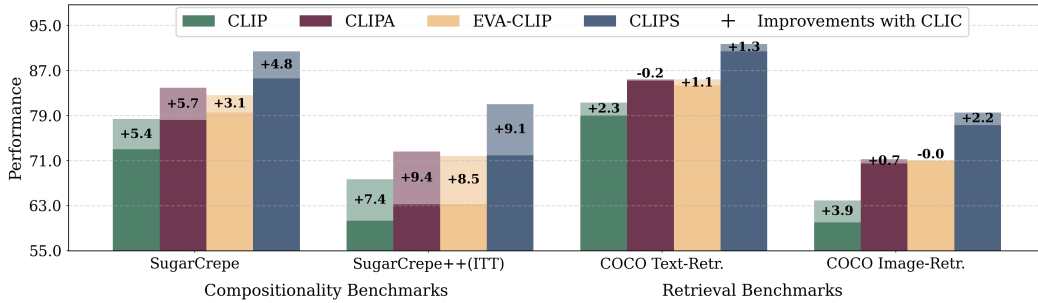


Figure 5: **CLIC improves differently pre-trained CLIP models.** We show for varied CLIP models, using CLIC (PixelProse) gives consistent improvements on both compositionality and downstream evaluation benchmarks. Specifically, to the best of our knowledge, CLIPS +CLIC-RedCaps yields SOTA numbers SugarCrepe++ for CLIP-like models.

Table 26: **Comparison of SugarCrepe and SugarCrepe++ performance for newer CLIP style models.** We present in this table a detailed set of results for the pre-trained CLIPA, EVA-CLIP, CLIPS, and their respective versions fine-tuned with CLIC. For SugarCrepe++, we report only the ITT scores in detail.

Compositionality: SugarCrepe++								
Model	Source	Rep-obj	Rep-Att	Rep-Rel	Swap-obj	Swap-Att	Avg-ITT	Avg-TOT
CLIPA	[26]	94.5	73.6	56.3	41.2	50.4	63.2	57.2
CLIC-LAION	Ours	93.9	79.1	62.1	52.2	57.9	69.1	52.1
CLIC-RedCaps	Ours	95.0	80.6	64.8	60.4	62.2	72.6	46.8
EVA02-CLIP	[51]	93.2	73.5	59.3	43.7	46.7	63.3	52.7
CLIC-LAION	Ours	93.8	79.7	61.8	55.9	61.1	70.4	45.2
CLIC-RedCaps	Ours	95.2	80.7	63.8	59.2	60.0	71.8	41.9
CLIPS	[33]	<u>95.5</u>	77.8	64.9	51.4	69.8	71.9	<b>65.1</b>
CLIC-LAION	Ours	<u>95.5</u>	<u>84.8</u>	<u>72.6</u>	<u>66.1</u>	<u>80.0</u>	<u>79.8</u>	62.3
CLIC-RedCaps	Ours	<b>95.9</b>	<b>85.1</b>	<b>73.7</b>	<b>68.6</b>	<b>81.7</b>	<b>81.0</b>	<u>62.9</u>
Compositionality: SugarCrepe								
		Add-obj	Add-Att	Rep-Obj	Rep-Att	Rep-Rel	Swap-Obj	Swap-Att
CLIPA	[26]	88.8	82.6	96.8	83.0	68.8	62.4	65.1
CLIC-LAION	Ours	93.5	92.2	97.1	89.3	78.3	72.2	70.9
CLIC-RedCaps	Ours	94.5	92.3	97.0	86.9	73.8	69.8	73.3
EVA02-CLIP	[51]	92.3	82.8	95.9	84.9	71.6	65.3	64.0
CLIC-LAION	Ours	85.5	91.3	96.4	86.5	72.7	70.6	70.5
CLIC-RedCaps	Ours	90.7	90.6	96.7	86.7	73.2	70.2	70.3
CLIPS	[33]	92.5	86.3	97.4	88.6	78.1	74.3	85.0
CLIC-LAION	Ours	<b>96.6</b>	<b>96.8</b>	<b>98.7</b>	<b>93.0</b>	<b>85.5</b>	<b>82.0</b>	<b>89.3</b>
CLIC-RedCaps	Ours	<u>95.1</u>	<u>95.5</u>	<u>97.9</u>	<u>92.2</u>	<u>83.2</u>	<u>81.2</u>	<u>87.8</u>

Table 27: **Retrieval@1 and WinoGround for ViT-L/14 models.** In addition to retrieval results with Recall@5 in Table 3, we report retrieval scores at Recall@1 and WinoGround results. \* are not zero-shot for MS-COCO. The improvements made by CLIC here are consistent with Recall@5.

Method	Retrieval				WinoGround		
	Text		Image		Text	Image	Group
	COCO	F30k	COCO	F30k			
CLIP [45]	56.0	85.1	35.2	64.7	28.7	11.0	8.5
CoN-CLIP* [49]	55.8	82.5	37.8	69.3	29.5	11.0	8.0
CLIC-LAION	56.8	85.4	40.2	70.4	30.2	10.7	8.2
CLIC-RedCaps	58.3	86.9	39.0	68.8	32.2	12.2	9.0
CLIPA	64.4	90.4	46.2	73.8	32.7	8.0	7.0
CLIC-LAION	63.0	86.7	46.8	73.6	28.7	8.7	6.7
CLIC-RedCaps	63.2	89.1	46.6	73.9	32.0	11.2	7.2
EVA-CLIP	63.7	90.5	46.8	77.3	32.7	12.5	10.2
CLIC-LAION	63.6	90.5	46.5	77.4	31.7	12.7	8.7
CLIC-RedCaps	63.2	89.8	46.4	76.7	36.0	11.7	8.2
CLIPS	<u>73.6</u>	<u>95.7</u>	54.2	82.6	36.2	16.0	12.5
CLIC-LAION	71.0	95.3	<b>56.6</b>	<b>85.7</b>	37.5	16.0	13.2
CLIC-RedCaps	<b>74.6</b>	<b>96.3</b>	<b>56.6</b>	<u>84.3</u>	<b>41.7</b>	<b>17.5</b>	<b>15.2</b>

Table 28: **Retrieval@1 for ViT-B/32 models.** In addition to retrieval results with Recall@5 in Table 2, we report retrieval scores at Recall@1. \* are not zero-shot for MS-COCO.

Method	Retrieval			
	Text		Image	
	COCO	F30k	COCO	F30k
CLIP [45]	49.5	78.5	30.1	59.1
NegCLIP* [56]	<b>60.8</b>	<b>85.1</b>	<b>45.1</b>	<b>70.9</b>
DAC-LLM [9]	33.0	64.9	36.7	53.2
DAC-SAM [9]	33.0	61.5	33.2	58.4
SVLC-R [10]	45.0	73.5	35.4	64.5
SVLC-R+L [10]	42.8	70.4	35.6	63.7
CoN-CLIP* [49]	48.5	75.7	30.3	59.7
TripletCLIP [41]	46.9	73.7	32.3	60.2
CLIC-COCO*	<u>60.0</u>	<u>84.7</u>	<u>41.1</u>	<u>68.3</u>
CLIC-LAION	52.2	79.9	35.2	62.6
CLIC-RedCaps	52.5	79.4	33.8	62.8

Prompt: Can you describe this image in detail?

Input Image



LLaVA-1.5-7b (CLIP)

The image features a small dog with a curly coat, **standing** on a sandy beach. The dog appears to be enjoying itself as it runs around on the sand. There are two people in the scene, **one standing** closer to the left side of the dog and **the other person** further back on the right side. The beach setting is complemented by the **presence of a backpack**, likely belonging to one of the people, placed on the sand near the center of the scene.

LLaVA-1.5-7b (CLIC)

The image features a small brown and white dog **running** on a sandy beach. The dog appears to be enjoying itself as it runs through the sand, kicking up some dust. There **is another person in the background**, possibly observing the dog or engaging in their own activities. In the scene, there are also **two handbags**, one near the person in the background and another closer to the dog.

Prompt: Can you describe this image in detail?

Input Image



LLaVA-1.5-7b (CLIP)

In the image, there are two people in a gym setting. A woman is standing on a machine, **holding a pair of dumbbells**, and appears to be working out. **A man is standing nearby, also holding a pair of dumbbells**. They are both focused on their workout routines. The gym is equipped with various pieces of equipment, including a bench, **a treadmill**, and **a car**. The bench is located in the background, while the treadmill is positioned towards the right side of the scene. The car is situated on the far right side of the image, **possibly indicating that the gym is located in a parking area**.

LLaVA-1.5-7b (CLIC)

The image features a woman in a pink shirt and black pants working out in a gym. She is **holding a pair of dumbbells and is in the process of lifting them above her head**. The woman appears to be focused on her workout, demonstrating her strength and determination. **In the background, there is another person who seems to be observing the woman's workout**. A car is visible in the far right corner of the image, **possibly parked outside the gym**. Additionally, there is a bench located near the center of the scene.

Prompt: Can you describe this image in detail?

Input Image



LLaVA-1.5-7b (CLIP)

The image features **a man and a woman sitting on a rooftop, both performing handstands**. The man is on the left side of the scene, while the woman is on the right side. They are **both balancing on their hands**, showcasing their impressive strength and flexibility. In the background, **there are two cars visible**, one on the left side and the other on the right side of the rooftop. The cars are parked at a distance, providing a sense of depth to the scene.

LLaVA-1.5-7b (CLIC)

The image features a woman and a man performing a handstand on a rooftop. **The woman is sitting on the rooftop, while the man is standing on his hands, doing a handstand**. They are both enjoying their time outdoors, showcasing their impressive balance and strength. In the background, **there are several cars parked**, adding to the urban setting of the scene. The cars are scattered around the rooftop, with some closer to the foreground and others further away.

Prompt: Can you describe this image in detail?

Input Image



LLaVA-1.5-7b (CLIP)

The image features a small dog wearing a red and white checkered shirt, standing next to a cat. They are both positioned in a grassy yard, with the dog on the left side and the cat on the right side of the scene. In the background, there are two potted plants, one located on the left side and the other on the right side of the yard. **A bench can be seen in the middle of the yard, providing a place for people to sit and enjoy the outdoor space**.

LLaVA-1.5-7b (CLIC)

The image features a small brown and white dog standing next to a large orange and white cat in a yard. Both animals appear to be looking at each other, possibly engaging in a playful interaction. The dog is positioned on the left side of the scene, while the cat is on the right. In the background, there are two potted plants, one located near the left side of the scene and the other on the right side. **A bench can also be seen in the yard, situated towards the right side of the scene**.

Figure 6: **Comparing response quality of different LLaVA-1.5-7b versions on random WinoGround validation images.** We compare the open-ended prompt (Can you describe this image in detail?) based image captioning of LLaVA with CLIP’s vision encoder to that of CLIC. Overall, we believe the responses from CLIC enabled LLaVA to be marginally better than the standard LLaVA, but both models show aspects of hallucinations. Incorrect parts are **highlighted** and the respective corrections (if available) are also **highlighted**.