What does CLIP know about a red circle?
Visual prompt engineering for VLMs

Aleksandar Shtedritski  Christian Rupprecht  Andrea Vedaldi
Visual Geometry Group, University of Oxford
{suny, chrisr, vedaldi}@robots.ox.ac.uk

Abstract

Large-scale Vision-Language Models, such as CLIP, learn powerful image-text representations that have found numerous applications, from zero-shot classification to text-to-image generation. Despite that, their capabilities for solving novel discriminative tasks via prompting fall behind those of large language models, such as GPT-3. Here we explore the idea of visual prompt engineering for solving computer vision tasks beyond classification by editing in image space instead of text. In particular, we discover an emergent ability of CLIP, where, by simply drawing a red circle around an object, we can direct the model’s attention to that region, while also maintaining global information. We show the power of this simple approach by achieving state-of-the-art in zero-shot referring expressions comprehension and strong performance in keypoint localization tasks. Finally, we draw attention to some potential ethical concerns of large language-vision models.

1. Introduction

Large Language Models (LLMs) such as GPT-2/3 [7, 39] and ChatGPT [1] have demonstrated surprising emerging behaviours. For example, these models can perform language translation without being explicitly trained for it, in a zero-shot manner. This can be partially explained by the fact that occurrences of the desired behaviours, such as translating between two languages, naturally occur in their enormous training corpus, which is, essentially, the Internet.

Interesting emergent behaviours have been observed in large Vision-Language Models (VLMs) like CLIP [38] too. For example, CLIP can be used for zero-shot classification by checking the compatibility of a given image with prompts such as “an image of a X”, where X is one of a set of class hypotheses to be tested.

Emergent behaviours are elicited by supplying suitably crafted inputs to the VLMs, often called prompts. As in the example above, researchers have mostly focused on engineering textual prompts, manipulating the textual input of the model. This approach is inspired by LLMs, where manipulating the textual modality is the only available option. However, VLMs are inherently multimodal and offer the possibility of manipulating both modalities, textual and visual. While the textual modality is the natural choice for expressing semantics, the visual modality can be better for expressing geometric properties such as location.

In this paper, we thus explore visual prompt engineering. We do so with two goals. The first goal is to contribute one more practical tool for extracting useful information from VLMs in a zero-shot manner. We demonstrate this by obtaining state-of-the-art zero-shot results in referring expressions comprehension by engineering visual prompts. The second goal is to characterise interesting and

Figure 1: Visual Prompt Engineering. We draw multiple annotations over an image and have CLIP choose the correct one given a caption. Here we show predictions for the given expressions. Top: Examples from RefCOCOg on referring expressions detection. Bottom: Example from SPair71k on keypoint localization.
unexpected properties of the VLMs and their training data, including identifying some behaviours that can raise ethical concerns.

Perhaps the most surprising of our findings is the effectiveness of a particular type of visual prompting: **drawing a plain red circle** on top of the image (Fig. 1). We show that this simple intervention steers the VLM to analyse/talk about the image region contained in the circle. This behaviour can then be used for tasks such as naming a specific object or object part or detecting particular image regions based on a description. The latter, for instance, is achieved by marking each object proposal with a red circle and using the VLM to find the best match with respect to the provided referring expression, achieving strong results on multiple benchmarks in the unsupervised regime. Furthermore, we show that prompting with a circle also works for finer-grained localization, marking specific object parts or keypoints instead of just whole objects.

We further contrast marking an image with the alternative of cropping it, which, from sliding window classifiers to region neural networks, is the canonical approach to steer the focus of an image-level predictor to a particular image region. We show that, for VLMs at least, marking is significantly more effective than cropping, possibly because it does not lose contextual information like the latter.

Apart from the practical applications, our findings reveal unexpected and intriguing properties of VLMs. We show empirically, that marking with a red circle is optimal among a selection of possible markers (variants of the circle, boxes, arrows, etc.). Presumably, the VLMs understand red circles out of the box because these appear sufficiently frequently in the training corpus, i.e., the Internet. While we do not have access to the full training data of CLIP, we corroborate this intuition by seeking examples of such images in YFCC15M, a dataset of CC-BY images.

Our analysis shows that red circles are indeed present even in a (comparatively small) dataset of images like YFCC15M, but they are rare. It is a testament to the extraordinary capacity of VLMs that such a behaviour can be learned from such rare events, without an explicit focus on doing so. We test models of different sizes/capacities and show that only the larger models exhibit this behaviour reliably, which corresponds to our intuition.

Finally, we note that the ability of VLMs to learn even from rare events such “red circles” can acquire both desirable and undesirable behaviours. Red circles, in particular, can have a negative connotation in the training data as they are often used by news outlets to mark missing people or criminals and, evidently, the model learns from such examples. As a result, we show that drawing a red circle in an image increases the probability that the model would characterise a person as a criminal or as a missing person.

To summarise, we make the following main contributions: (1) We propose marking as a new form of visual prompt engineering that is effective in extracting useful emergent behaviours in VLMs like CLIP; (2) We use the latter to achieve state-of-the-art zero-shot referring expressions comprehension using a VLM; (3) We provide an analysis of why marking is effective for these models, and link that to the training data and large model capacity; (4) We show that visual prompt engineering can also elicit unwanted behaviours, such as triggering problematic biases in the VLMs, revealing potential ethical issues.

## 2. Related work

**Emergent Behaviour from Large Scale Pretraining** has mainly been observed in Large Language Models (LLMs). Most notably, GPT-2 [39], GPT-3 [7], and ChatGPT [1] have been shown to be capable of tasks such as zero-shot translation, question answering, arithmetic, as well as planning actions for embodied agents [18]. Fine-tuning LLMs can also lead to models that can generate code from docstrings [8] or solve math problems [13, 23]. Only a few emergent zero-shot behaviours have been reported for VLMs like CLIP, mainly for classification [38] and OCR [33]. Generative VLMs like FLAMINGO [3] and BLIP [24] excel in captioning and visual question-answering tasks, but also have no way of solving pixel-level computer vision tasks.

**Prompting VLMs** is most commonly performed by prepending a set of learnable tokens to the text input [16, 21, 65, 66], vision input [19, 46, 64], or both text and vision inputs [41, 60], in order to easily steer a frozen CLIP model to solve a desired task. [4] learn augmentations in pixel space, such as padding around the image, or changing a patch of the image, which are optimized with gradient descent on a downstream task. [5] cast image inpainting as a visual prompting task, using a generative model trained on figures from academic papers. Coloring regions of an image has been used for the VCR task [61], where a model is fine-tuned on annotated images [62]. Colorful Prompt Tuning (CPT) [57] color regions of an image and use a captioning model to predict which object in an image an expression refers to by predicting its color. Similarly to CPT, we augment the input image in pixel space and perform zero-shot inference. However, we annotate the image in a human-like manner and show that our method is more powerful and more flexible than CPT.

**Referring Expression Comprehension** (REC) aims to localize a target object in an image that corresponds to a textual description. Most approaches to REC start with object proposals, for example, generated with Faster-RCNN [40], and learn to score them [17, 30, 31, 48, 54]. REC is sometimes considered together with referring expression generation — the task of generating a description of a given region. [31] use a comprehension model to guide a generator,
whereas [9] jointly train a detector with a caption generator. Some works model the scene as a graph [30, 48, 54] or use language parsers and grammar-based methods [12, 29], leading to a more interpretable result. More recently, transformer architectures have been used [14, 22, 25, 50, 55]. [14, 22, 25, 55] perform text-modulated object detection, where a transformer decoder takes the referring expression as an input and predicts a bounding box. [50] train with a text-to-pixel contrastive loss, which allows for a text-driven segmentation or detection at test time.

**Unsupervised Referring Expression Comprehension** is a less explored area, only made possible with the introduction of large pre-trained models such as CLIP [38]. ReCLIP [44] crops object proposals and ranks them using CLIP before an ad-hoc postprocessing step to take into account relations such as left/right, smaller/bigger, etc. CPT [57] colors object proposal boxes and use a pre-trained captioning model [63] to auto-regressively predict which colored proposal corresponds to the query description. Pseudo-Q [20] generates descriptions for multiple objects in an image, which is used to train a REC network. However, this model is not fully unsupervised as the pseudo descriptions it uses are generated using a captioning model trained on COCO.

**Visual Reasoning Using Large Pretrained Models** has been an area of significant interest in the last few years. In addition to referring expression detection [44], CLIP [38] has been used for semantic segmentation [27, 37]. [37] use CLIP to assign text labels to object parts after doing part co-segmentation in the latent space of a GAN. [27] utilize CLIP for open-vocabulary segmentation by using a general-purpose mask proposal network and CLIP as a classifier. CLIP has also been used for unsupervised object proposal generation [42] and open-set detection [15]. Semantic segmentation also emerges from image only [34, 49] or image-text [53] self-supervision.

**Bias of VLMs** is an increasingly popular area of research, as downstream applications come with the risk of perpetuating biases and stereotypes existing in the training data. However, methods for assessing the bias of a VLM are still not well established. [2] measure the misclassification rate of CLIP of faces of people of different races with non-human and criminal categories, whereas [6, 11, 47] measure fairness in retrieval results. Here, we show a different kind of bias, where the addition of a red circle over a person can trigger a negative connotation.

### 3. Method

Our goal is to develop visual prompting in **Vision-Language Models** (VLMs). VLMs solve prediction tasks that involve jointly processing text and images. For example, models such as CLIP are trained to match text and image samples. The input to such a VLM is an image $i \in \mathbb{R}^{H \times W}$ and text $t \in \Sigma^*$, where $\Sigma$ is an alphabet. The output is a score $s(i, t)$ that expresses the degree of compatibility between the supplied image and text.

#### 3.1. Prompt engineering

One of the most striking capabilities of VLMs is their ability to solve a variety of classification tasks with little to no further training at all, in a zero-shot manner. This is done by reducing the task of interest to that of evaluating the VLM on suitably-engineered image and text pairs.

For example, given an image-caption pair $(i, t)$, consider the problem of localizing a named object keypoint in the image. We can cast this as a question-answer problem, where the question $q \in Q$ is the name of the object keypoint (e.g., “right ear”, “front left leg”, ...) and the answer $a \in A$ is one of a discrete set of image locations.

Because the VLM computes a compatibility score $s(q, t)$ between an image $i$ and the text $t$, it cannot be used to map the question $q$ to the answer $a$ directly. However, via prompt engineering, we can use the VLM to construct a compatibility score $s(q, a|i, t)$ between question and answer, conditioned on the input image-text pair $(i, t)$. This score is in general given by the expression

$$s(q, a|i, t) = s(i_{qa}, t_{qa}), \tag{1}$$

where $i_{qa}$ and $t_{qa}$ are versions of the input image and text, obtained by transforming the latter to reflect the question-answer pair $(q, a)$.

The specific way Eq. (1) should be applied to a problem depends on the specific nature of the latter. For example, in the problem of localizing the named keypoints, it is natural to encode the name of the keypoint via the textual modality and its 2D location via the visual modality. For instance, in order to answer the question $q = “right ear”$ for a given input image $i$ with caption $t = “dog”$, we can engineer the textual prompt $t_{qa} = t_q = “an image of the right ear of a dog”$ to encode a description of the named entity. Likewise, we can engineer the visual prompt $i_{qa} = i_a$ in such a way as to ‘select’ the location $a$ in the image, using one of the methods discussed in Section 3.2. With this, we can answer the question by finding $\hat{a}(q|i, t) = \arg\max_{a \in A} s(q, a|i, t)$, that maximizes the score $s(q, a|i, t) = s(i_{qa}, t_{qa})$, which specializes Eq. (1).

In the following sections, we provide further details and apply these ideas to a few concrete tasks.

#### 3.2. Visual prompting via marking

The usual way of encoding location information in a visual prompt is to crop the image around the desired location, meaning that $i_a$ is the image cropped around $a$. This idea has been used extensively with VLMs, including to interpret referring expressions, where maximizing a score of the
form \( s(i_a, t_q) \) seeks for the image crop that best matches the referring expression \( t_q \).

In this paper, we explore an alternative approach for visual prompting that uses the concept of marking the desired region in the image. Marking quite literally means overlaying to the image \( i_a \) a circle, a box, or an arrow, which visually indicates the desired location \( a \).

While the idea of marking may sound strange, it is interesting for two reasons. First, differently from cropping, a marked image \( i_a \) preserves almost all the information contained in the input image \( i \), including contextual information that crops lack. Second, we show that marking works well with VLMs, outperforming cropping-based prompt engineering in some prediction tasks.

While the simplest marking consisting of a red circle is particularly effective, in Section 4 we explore several different ways of generating markings. We refer the reader to that section for further details and examples.

### 3.3. Tasks

We study the idea of mark-based prompt engineering by considering several zero-shot prediction tasks, from simple tasks such as matching keypoints to their names to more complex ones such as referring expression comprehension.

**Naming Keypoints.** The first and simplest task that we consider is matching the name of the keypoints of an object to their 2D locations in an image. The input is an image \( i \), a set of keypoint names \( Q \), and a set of corresponding keypoint locations \( A \subset \{0,\ldots,H-1\} \times \{0,\ldots,W-1\} \). The number of names and locations is the same \( (m = |Q| = |A|) \) and the goal is to match the two. We express the latter as predicting the square permutation matrix \( \Pi \in S_m \) that associates each name \( q \) to its corresponding location \( a \) (i.e., \( \Pi_{qa} = 1 \)).

In order to predict \( \Pi \), we use Eq. (1) to define the cost of associating name \( q \) to location \( a \) as \( C_{qa} = s(i_a, t_q) \) where \( i_a \) is obtained either via cropping or marking and \( t_q \) is just the name of the keypoints prefixed by the string “an image of”. For this problem, the role of questions and answers is symmetric and we decode the cost matrix \( C \) into a permutation matrix \( \Pi \) via optimal transport:

\[
\hat{\Pi}(i, Q, A) = \argmax_{\Pi \in S_m} \sum_{q \in Q, a \in A} \Pi_{qa} \exp\left(-\tau C_{qa}\right), \tag{2}
\]

where \( \tau > 0 \) is a temperature parameter. This optimization problem is solved efficiently via the Sinkhorn-Knopp algorithm [43], which renormalizes matrix \( C \).

**Keypoint Localization.** The second task is a more useful and difficult variant of the first. The goal is still to localize a named keypoint \( q \) in an image, but this time the locations \( A \) are a subset of a \( m \times m \) regular grid. These are further restricted to a salient image region extracted by using the unsupervised saliency method of [49] to avoid testing irrelevant locations in the background. The difference compared to naming keypoints is that this version of the problem does not assume prior knowledge of the possible locations of the keypoints. Given the name \( q \) of a keypoint, its location \( a \) is then obtained as \( \hat{a}(i, q) = \argmax_{a \in A} s(i_a, t_q) \) where \( i_a \) and \( t_q \) are as defined previously.
Table 1: Naming keypoints results on CUB and SPair71k. On SPair71k, we show results on all animal classes — bird, cat, dog, horse, sheep, cow. We show the percentage of correctly matched keypoints and names, given a list of them. We compare to randomly guessing the correct correspondence and cropping around the region of interest, rather than drawing an annotation. We compare all methods with and without normalization with the Sinkhorn-Knopp (SK) algorithm.

### Referring Expression Comprehension

Comprehending a referring expression means detecting an object in an image that corresponds to a textual specification that explicitly refers to it (e.g., “fourth dog from the right”). Similarly to prior work [20, 44, 57], given an image $i$, we approach this problem by extracting first a set of object proposals using the method from [58] and interpret those as the set of possible answers $A$. The set of questions $Q$ is instead a collection of referring expressions extracted from a given benchmark dataset. For each referring expression, the best matching proposal is then given by

$$\hat{a}(i, q) = \operatorname{arg\,max}_{a \in A} \left[ s(i_a, t_q) - \frac{1}{|Q|} \sum_{q \in Q} s(i_a, t_q) \right].$$  

The engineered prompts $i_a$ and $t_q$ are defined as in Section 3.3. In this case, we found it useful to subtract from the score the average with respect to all possible referring expressions $Q$. This weighs down hypotheses $a$ such as faces that are visually very salient and tend to respond very strongly to all questions $q$.

### 4. Experiments

We study the properties of visual marking in VLMs by considering first the three tasks of Section 3.3: naming keypoints, localizing keypoints, and referring expression comprehension.

#### 4.1. Naming Keypoints

Naming keypoints is a comparatively simple problem that has no direct application; however, it is simpler and faster to evaluate than the other tasks, so we use it to ablate various aspects of our method.

**Data and implementation details.** For this task, we consider the CUB-200-2011 (CUB) [51] and SPair71k [35] datasets. The first contains named keypoint annotations for each image, whereas the second only annotates matching keypoints in pairs of images, but does not name them. We thus augment the latter, manually naming each keypoint instance in each animal image. We further crop the images from SPair71k with the provided bounding boxes. For the VLM, we use the ViT-L/14@336px backbone. Please see the sup. matt. for details.

**Results.** Recall that, in this task, the output of the predictor is a permutation matrix $\Pi$ associating each keypoint location to a corresponding name. We report (i) the ratio of keypoint names that are mapped to the correct location and (ii) the ratio of keypoint locations that are mapped to the correct names. To the best of our knowledge, there are no prior works that associate keypoints with their names. We thus compare the result of this new task to (a) a random choice and (b) a baseline where $i_a$ is obtained by cropping.

As seen in Table 1, prompting via visual marking (red circles) significantly outperforms the baselines, achieving almost twice the accuracy. Using the Sinkhorn-Knopp (SK) algorithm to normalize the matching score further boosts results, mainly improving results for points that are ambiguous and close to each other, e.g., mouth and nose.

**What is the best visual marker?** We compare the use of (i) different shapes for highlighting a location: circle,
Table 2: Ablation of annotation types for naming keypoint. We evaluate on CUB and present results across a variety of colors and sizes (for marker shape) and sizes (for circle color). For full results refer to the sup. matt.

<table>
<thead>
<tr>
<th>Marker shape</th>
<th>Mean</th>
<th>Best</th>
</tr>
</thead>
<tbody>
<tr>
<td>Circle</td>
<td>33.5 ± 4.5</td>
<td>46.5</td>
</tr>
<tr>
<td>Arrow</td>
<td>28.3 ± 3.1</td>
<td>36.3</td>
</tr>
<tr>
<td>Square</td>
<td>24.1 ± 3.6</td>
<td>36.3</td>
</tr>
<tr>
<td>Cross</td>
<td>21.5 ± 6.3</td>
<td>34.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Circle color</th>
<th>Mean</th>
<th>Best</th>
</tr>
</thead>
<tbody>
<tr>
<td>Red</td>
<td>36.4 ± 5.1</td>
<td>46.5</td>
</tr>
<tr>
<td>Green</td>
<td>34.3 ± 4.2</td>
<td>43.3</td>
</tr>
<tr>
<td>Purple</td>
<td>34.0 ± 3.7</td>
<td>41.9</td>
</tr>
<tr>
<td>Blue</td>
<td>32.7 ± 3.9</td>
<td>41.1</td>
</tr>
<tr>
<td>Yellow</td>
<td>32.4 ± 4.0</td>
<td>40.8</td>
</tr>
</tbody>
</table>

Figure 4: Example Annotations for Keypoints. We show how the annotations we use look for different shapes and colors. We also experiment with multiple thicknesses and sizes. The red circle annotation on the left is the one we use throughout all evaluations unless stated otherwise.

rectangle, cross, arrow, (ii) different sizes, and (iii) different colors of the annotations, and show some examples in Fig. 4. We compare different shapes and colors in Table 2 and find that red circles perform best. Red is the best color despite the fact that it is a commonly occurring color in images, unlike colors like purple which can be found less often in nature and can thus be more distinctive, but lead to worse performance. We attribute this to the fact that this emergent capability of CLIP exists due to human-centric manipulations of its training data, and humans are likely to annotate using red circles, as shown next.

Are there visual markers in the training data? To explore the hypothesis that CLIP can zero-shot classify annotations on images because of similar examples seen during training, we find images in YFCC15M that contain markers (YFCC15M is a subset of the CLIP training data). To this end, we train a binary classifier using an ensemble of a ViT-B/16 and RN50x16 CLIP vision encoders to classify images in YFCC15M that contain annotations. We then use this to filter a 6M subset of YFCC15M and take the top 10k images with the highest score. Finally, we manually examine the 10k images and find 70 images that have annotations drawn on top of them. We show 3 such images in Fig. 5. Hence, the training data contains examples of markers, but they are very rare (~0.001%), suggesting that such behavior can only be learned from very large datasets by high-capacity models. This is further explored next.

How do different VLMs differ? We compare a number of CLIP models in Fig. 6. In general, we observe that the performance of keypoint matching improves with (i) the size of the pretrained dataset and (ii) the size of the vision encoder. The former holds true for CLIP models trained on WIT-400M vs YFCC-15M (which is a subset of WIT-400M). However, using the LAION-2B dataset for pretraining leads to worse results. We suspect this result comes from differences in filtering when creating WIT-400M and LAION-2B, where in the latter, examples of annotations might have been discarded due to a stronger focus on aesthetic images for generative models. Similarly, we see big gains in performance as we increase the size of the vision encoder, and the gains do not seem to converge with the biggest available models. We emphasize on the dramatic increase in performance of the WIT-400M pretrained CLIP — the biggest models improve on the performance of...
Given a set of ground-truth points $\mathbb{P}$ is widely used when evaluating semantic correspondences. As a metric, where only the largest GPT models perform well [7]. We draw similarities between this task and tasks in the domain of NLP, such as zero-shot or one-shot arithmetic, where only the largest GPT models perform well [7]. We argue that in a similar fashion, the vision encoder needs sufficient capacity and data in order to show this emergent behavior.

### 4.2. Localizing Keypoints

For this experiment, we use the same data and network architecture as for the previous one, but report the percentage of correct keypoints (PCK) as a metric, as the latter is widely used when evaluating semantic correspondences. Given a set of ground-truth points $\mathbb{P} = \{p_m\}_{m=1}^M$ and predictions $\hat{\mathbb{P}} = \{\hat{p}_m\}_{m=1}^M$, PCK is given by:

$$\text{PCK}(\mathbb{P}, \hat{\mathbb{P}}) = \frac{1}{M} \sum_{m=1}^M \mathbb{1}[\|p_m - \hat{p}_m\| \leq \delta].$$

Table 5: Comparison with state-of-the-art on REC. We report top-1 accuracy ($\%$). ‡ is the crop-based baseline of [44] and § is their method that adds relational resolving. "ZS" refers to zero-shot approaches. Drawing red circles outperforms other zero-shot approaches on most benchmarks, including ReCLIP‡ that post-processes results using manually designed relational rules. On RefCOCO+ and RefCOCOg, a red circle also outperforms Pseudo-Q and DTWREG that are not zero-shot and use weak supervision.

Here, $\delta$ is a distance threshold given by $\delta = \alpha \max(H, W)$, where $0 < \alpha < 1$ is a ratio and $(H, W)$ is the bounding box size. For all datasets we use $\alpha = 0.1$. Keypoint localization also utilizes an unsupervised saliency mask to ignore background locations.

Similarly to the naming task, we compare keypoint localization to random guessing and the crop-based baseline. As shown in Table 4, using red circles significantly outperforms both; as expected, results are further improved by using saliency to further filter keypoint locations. We show qualitative results in Fig. 3.

#### 4.3. Referring Expression Comprehension

**Datasets and implementation details.** Referring expression comprehension is commonly evaluated on the RefCOCO [59], RefCOCO+ [59], and RefCOCOg [32] datasets, all of which consist of images from the MS-COCO dataset [28] together with expressions that refer to a unique object in the image, which are also annotated with a bounding box. RefCOCO+ only contains appearance-based expressions, whereas RefCOCO and RefCOCOg contain relation-based expressions (e.g., containing the words left/closer/bigger). The test sets of RefCOCO and RefCOCO+ are split in two, where "testA" and "testB" contain only people and non-people, respectively. We evaluate using the percentage of correct predictions, where a box is correctly predicted if its intersection-over-union with the ground-truth box is over 0.5.

For the referring expressions task, we use an ensemble RN50x16 and ViT-L/14@336 CLIP backbones. Following prior work [44, 57], we score the bounding box proposals of MAAttNet [58].

**Results.** Using a red circle, we achieve state-of-the-art on most referring expressions comprehension baselines in the zero-shot setting, as shown in Table 5. Interestingly, this even outperforms ReCLIP [44], which is based on scor-
Table 6: Classification rate into criminal categories. We report ZS classification results in criminal categories on synthetic faces from FaceSynthetics and persons from COCO, with and without red circles. In brackets is the absolute difference from the crop without a red circle.

<table>
<thead>
<tr>
<th>Model</th>
<th>Red Circle</th>
<th>FaceSynthetics</th>
<th>COCO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Positive</td>
<td>Neutral</td>
</tr>
<tr>
<td>ViT-L/14@336px</td>
<td>✓</td>
<td>0.5%</td>
<td>35.6%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>0.0%</td>
<td>19.1%</td>
</tr>
<tr>
<td>ViT-L/14</td>
<td>✓</td>
<td>1.3%</td>
<td>40.8%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>0.0%</td>
<td>32.5%</td>
</tr>
<tr>
<td>ViT-B/16</td>
<td>✓</td>
<td>0.7%</td>
<td>49.1%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>0.0%</td>
<td>37.1%</td>
</tr>
<tr>
<td>ViT-B/32</td>
<td>✓</td>
<td>0.0%</td>
<td>85.2%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>0.0%</td>
<td>70.5%</td>
</tr>
<tr>
<td>RN50x16</td>
<td>✓</td>
<td>0.7%</td>
<td>46.4%</td>
</tr>
<tr>
<td></td>
<td>✓</td>
<td>0.1%</td>
<td>52.2%</td>
</tr>
</tbody>
</table>

4.4. Model biases and ethics

While drawing circles on images can extract useful behaviors from a VLM for a wide variety of legitimate image analysis tasks, it can also extract unwanted ones and must not be used for the analysis of sensitive data.

To demonstrate this fact, in Fig. 8 we take a random image from COCO that contains a male-looking and a female-looking individual, also adding a circle over each of them. We score the following sentences to the images: This is an image of a \{woman, man, missing person, suspected murderer\}. The apparent gender resolution is correct, but the circled images tend to be scored higher as missing or murderers. Blur added for privacy.

We further quantify these biases following [2], using synthetic faces from FaceSynthetics [52] and person crops from COCO [28]. For FaceSynthetics, we take 1000 random synthetic faces, and for COCO, we crop all bounding boxes for the class person from the validation set that have an area of at least 10% of the total area of the image, which comes down to 1352 crops. Following [2], we measure zero-shot classification rates into criminal categories. We introduce a “positive” category (honest man/woman/person), “neutral” category (man/woman/person) and a “criminal” category (criminal/thief/suspicious person). Finally, we zero-shot classify the original images and the images with circles. In Table 6 we present classification rates into criminal categories. We see that for all ViT encoders, the rate at which people are classified as criminals is significantly higher. This is problematic as such existing biases can lead to harmful consequences. Note that there are various limitations in this analysis, including the usage of binary gender.
5. Conclusions

We have shown that visual prompting via marking can extract useful behavior from VLMs such as CLIP in a zero-shot manner, achieving state-of-the-art zero-shot referring expression comprehension performance, and significantly outperforming traditional techniques like image cropping. Our analysis suggests that this behavior emerges because relevant samples of marking exist in the training data of the VLMs, but these samples are very rare. As a consequence, the behavior can only be learned by very large models trained on very large datasets. The analysis also shows that VLMs acquire undesirable behaviors too, where the mere addition of a red circle to an image increases the model’s belief that the image has a negative connotation.

Dataset Ethics. We use the RefCOCO, RefCOCO+, MSCOCO, FaceSynthetics, YFCC15M, CUB, SPair71k in a manner compatible with their terms. Some of these images may contain personal data (faces). In Sections 4.1 to 4.3 there is no extraction of biometric data. In Section 4.4 we use MS-COCO to demonstrate that such a method cannot reliably extract information about people due to the bias in the pre-trained CLIP model (there is no identification). The FaceSynthetics, used for the same purpose, is a dataset of synthetic faces, so it does not raise privacy concerns. For further details on ethics, data protection, and copyright please see https://www.robots.ox.ac.uk/~vedaldi/research/union/ethics.html

Acknowledgements. We thank Luke Melas-Kyrizai, Tim Franzmeyer, Rhydian Windsor and Bruno Korbar for proofreading. A. Shtedritski is supported by EPSRC EP/S024050/1. A. Vedaldi and C. Rupprecht are supported by ERC-CoG UNION 101001212. C. Rupprecht is also partially supported by VisualAI EP/T028572/1.

References


[19] Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Vi-


[45] Mingjie Sun, Jimin Xiao, Eng Gee Lim, Si Liu, and John Y Goulanmas. Discriminative triad matching and reconstruction for weakly referring expression grounding. IEEE
transactions on pattern analysis and machine intelligence, 43(11):4189–4195, 2021. 7


