

# VinVL: Making Visual Representations Matter in Vision-Language Models

Pengchuan Zhang<sup>♡†</sup>    Xiujun Li<sup>♡♣†</sup>    Xiaowei Hu<sup>♡</sup>    Jianwei Yang<sup>♡</sup>    Lei Zhang<sup>♡</sup>  
 Lijuan Wang<sup>♡</sup>    Yejin Choi<sup>♣</sup>    Jianfeng Gao<sup>♡</sup>

January 5, 2021

## Abstract

This paper presents a detailed study of improving visual representations for vision language (VL) tasks and develops an improved object detection model to provide object-centric representations of images. Compared to the most widely used *bottom-up and top-down* model [2], the new model is bigger, better-designed for VL tasks, and pre-trained on much larger training corpora that combine multiple public annotated object detection datasets. Therefore, it can generate representations of a richer collection of visual objects and concepts. While previous VL research focuses mainly on improving the vision-language fusion model and leaves the object detection model improvement untouched, we show that visual features matter significantly in VL models. In our experiments we feed the visual features generated by the new object detection model into a Transformer-based VL fusion model OSCAR [21], and utilize an improved approach OSCAR+ to pre-train the VL model and fine-tune it on a wide range of downstream VL tasks. Our results show that the new visual features significantly improve the performance across all VL tasks, creating new state-of-the-art results on seven public benchmarks. We will release the new object detection model to public.

## 1 Introduction

Vision language pre-training (VLP) has proved effective for a wide range of vision-language (VL) tasks [26, 36, 4, 34, 20, 19, 45, 21]. VLP typically consists of two stages: (1) an object detection model is pre-trained to encode an image and the visual objects in the image to feature vectors, and (2) a cross-modal fusion model is pre-trained to blend text and visual features. While existing VLP research focuses mainly on improving the cross-modal fusion model, this paper focuses on improving the object-centric visual representations and presents a comprehensive empirical study to demonstrate that visual features matter in VL models.

Among the aforementioned work, a widely-used object detection (OD) model [2] is trained on the Visual Genome dataset [16]. The OD model provides an object-centric representation of images, and has been used in many VL models as a black box. In this work, we pre-train a large-scale object-attribute detection model based on the ResNeXt-152 C4 architecture (short as X152-C4). Compared to the OD model of [2], the new model is better-designed for VL tasks, and is bigger and trained on much larger amounts of data, combining multiple public object detection datasets, including COCO [25], OpenImages (OI) [17], Objects365 [31]

---

<sup>♡</sup>Microsoft Corporation

<sup>♣</sup>University of Washington

† indicates equal contributions.

Visual feature	VQA		GQA		Image Captioning				NoCaps		Image Retrieval			Text Retrieval			NLVR2	
	test-dev	test-std	test-dev	test-std	B@4	M	C	S	C	S	R@1	R@5	R@10	R@1	R@5	R@10	dev	test-P
Anderson <i>et al.</i> [2]	73.16	73.44	61.58	61.62	40.5	29.7	137.6	22.8	86.58	12.38	54.0	80.8	88.5	70.0	91.1	95.5	78.07	78.36
Ours	<b>75.95</b>	<b>76.12</b>	<b>65.05</b>	<b>64.65</b>	<b>40.9</b>	<b>30.9</b>	<b>140.6</b>	<b>25.1</b>	<b>92.46</b>	<b>13.07</b>	<b>58.1</b>	<b>83.2</b>	<b>90.1</b>	<b>74.6</b>	<b>92.6</b>	<b>96.3</b>	<b>82.05</b>	<b>83.08</b>
Δ	<b>2.79</b> ↑	<b>2.68</b> ↑	<b>3.47</b> ↑	<b>3.03</b> ↑	<b>0.4</b> ↑	<b>1.2</b> ↑	<b>3.0</b> ↑	<b>2.3</b> ↑	<b>5.9</b> ↑	<b>0.7</b> ↑	<b>4.1</b> ↑	<b>2.4</b> ↑	<b>1.6</b> ↑	<b>4.6</b> ↑	<b>1.5</b> ↑	<b>0.8</b> ↑	<b>3.98</b> ↑	<b>4.71</b> ↑

Table 1: Uniform improvements on seven VL tasks by replacing visual features from Anderson *et al.* [2] with ours. The NoCaps baseline is from VIVO [9], and our results are obtained by directly replacing the visual features. The baselines for rest tasks are from OSCAR [21], and our results are obtained by replacing the visual features and performing OSCAR+ pre-training. All models are BERT-Base size. As analyzed in Section 5.2, the new visual features contributes 95% of the improvement.

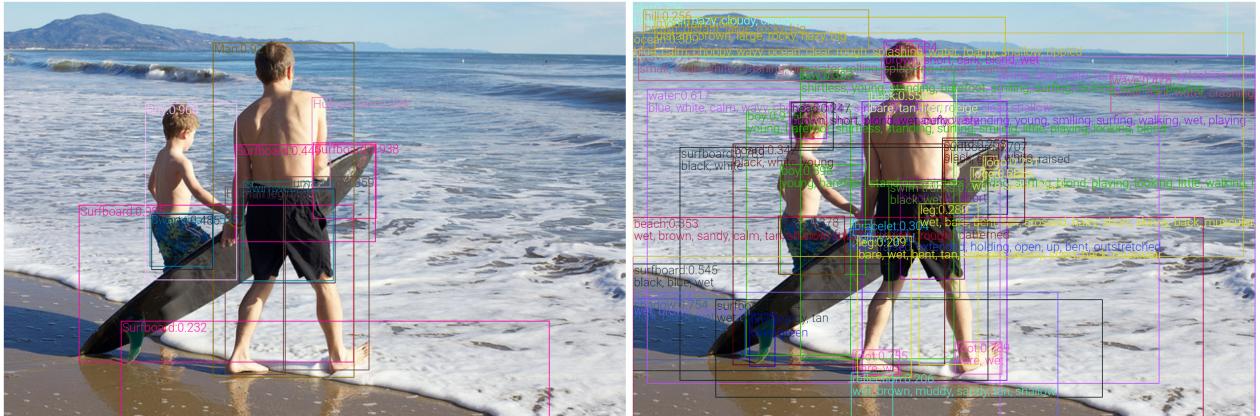


Figure 1: Predictions from an X152-FPN model trained on OpenImages (Left) and our X152-C4 model trained on four public object detection datasets (Right). Our model contains much richer semantics, such as richer visual concepts and attribute information, and the detected bounding boxes cover nearly all semantically meaningful regions. Compared with those from the common object classes in typical OD models (Left), the rich and diverse region features from our model (Right) are crucial for vision-language tasks. For concepts detected by both models, e.g., “boy”, attributes from our model offer richer information, e.g., “young barefoot shirtless standing surfing smiling little playing looking blond boy”. There are object concepts that are detected by our model but not by the OpenImages model, including fin, wave, foot, shadow, sky, hair, mountain, water, (bare, tan, light, beige) back, (blue, colorful, floral, multi colored, patterned) trunk, sand, beach, ocean, (yellow, gold) bracelet, logo, hill, head, (black, wet) swim trunks, black, wet swim trunks. Compared to the R101-C4 model of [2], our model produces more accurate object-attribute detection results and better visual features for VL applications; see Appendix A for the full pictures and predictions from [2].

and Visual Genome (VG) [16]. As a result, our OD model achieves much better results on a wide range of VL tasks, as shown in Table 1. Compared to other typical OD models, such as X152-FPN trained on OpenImages, our new model can encode a more diverse collection of visual objects and concepts (e.g., producing visual representations for 1848 object categories and 524 attribute categories), as illustrated by an example in Figure 1.

To validate the effectiveness of the new OD model, we pre-train a Transformer-based cross-modal fusion model OSCAR+ [21] on a public dataset consisting of 8.85 million text-image pairs, where the visual representations of these images are produced by the new OD model and are fixed during OSCAR+ pre-training.

We then fine-tune the pre-trained OSCAR+ for a wide range of downstream tasks, including VL understanding tasks such as VQA [8], GQA [13], NLVR2 [35], and COCO text-image retrieval [25], and VL generation tasks such as COCO image captioning [25] and NoCaps [1]. Our results show that the object-centric representations produced by the new OD model significantly improve the performance across all the VL tasks, often by a large margin over strong baselines using the classical OD model [2], creating new state of the arts on all these tasks, including GQA on which none of the published pre-trained models has surpassed the deliberately designed neural state machine (NSM) [12]. We will release the new OD model to the research community.

The main contributions of this work can be summarized as follows: (i) We present a comprehensive empirical study to demonstrate that visual features matter in VL models. (ii) We have developed a new object detection model that can produce better visual features of images than the classical OD model [2] and substantially uplifts the state-of-the-art results on all major VL tasks across multiple public benchmarks. (iii) We provide a detailed ablation study of our pre-trained object detection model to investigate the relative contribution to the performance improvement due to different design choices regarding diversity of object categories, visual attribute training, training data scale, model size, and model architecture.

## 2 Improving Vision (**V**) in Vision Language (**VL**)

Deep learning-based VL models typically consist of two modules: an image understanding module **Vision** and a cross-modal understanding module **VL**:

$$(\mathbf{q}, \mathbf{v}) = \mathbf{Vision}(Img), \quad y = \mathbf{VL}(\mathbf{w}, \mathbf{q}, \mathbf{v}), \quad (1)$$

where  $Img$  and  $\mathbf{w}$  are the inputs of the vision and language modalities, respectively. The output of the **Vision** module consists of  $\mathbf{q}$  and  $\mathbf{v}$ .  $\mathbf{q}$  is the semantic representation of the image, such as tags or detected objects, and  $\mathbf{v}$  the distributional representation of the image in a high-dimensional latent space represented using e.g., the box or region<sup>1</sup> features produced by a VG-pre-trained Faster-RCNN model [2]. Most **VL** models use only the visual features  $\mathbf{v}$ , while the recently proposed OSCAR [21] model shows that  $\mathbf{q}$  can serve as anchors for learning better vision-language joint representations and thus can improve the performance on various VL tasks.  $\mathbf{w}$  and  $y$  of the **VL** module of Equation (1) vary among different VL tasks. In VQA,  $\mathbf{w}$  is a question and  $y$  is an answer to be predicted. In text-image retrieval,  $\mathbf{w}$  is a sentence and  $y$  is the matching score of a sentence-image pair. In image captioning,  $\mathbf{w}$  is not given and  $y$  is a caption to be generated.

Inspired by the great success of pre-trained language models to various natural language processing tasks, vision-language pre-training (VLP) has achieved remarkable success in improving the performance of the cross-modal understanding module **VL** by (1) unifying vision and language modeling **VL** with Transformer and (2) pre-training the unified **VL** with large-scale text-image corpora. However, most recent works on VLP treat the image understanding module **Vision** as a black box and leave the visual feature improvement untouched since the development of the classical OD model [2] three years ago, despite that there has been much research progress on improving object detection by 1) developing much more diverse, richer, and larger training datasets (e.g. OpenImages and Objects 365), 2) gaining new insights in object detection algorithms such as feature pyramid network [23], one-stage dense prediction [24], and anchor-free detectors [37], and 3) leveraging more powerful GPUs for training bigger models.

In this work, we focus on improving **Vision** for better visual representations. We developed a new OD model by enriching the visual object and attribute categories, enlarging the model size and training on a

---

<sup>1</sup>We use the terms region and box interchangeably.

much larger OD dataset, and thus advanced the state of the arts on a wide range of VL tasks. We detail how the new OD model is developed in the rest of this section and then describe the use of OSCAR+ for VL pre-training in Section 3.

## 2.1 Object Detection Pre-training

To improve the OD model for VL tasks, we utilize four public object detection datasets. As most datasets do not have attribute annotations, we adopt a *pre-training and fine-tuning* strategy to build our OD model. We first pre-train an OD model on a large-scale corpus consisting of four public datasets, and then fine-tune the model with an additional attribute branch on Visual Genome, making it capable of detecting both objects and attributes.

**Data.** Table 2 summarizes the statistics of the four public datasets used in our object detection pre-training, including COCO, OpenImagesV5 (OI), Objects365V1, and Visual Genome (VG). These datasets have complementary characters, and are extremely unbalanced in terms of data size, object vocabulary, and the number of annotations in each class. For example, the VG dataset has a rich and diverse set of annotations for both objects and their attributes with an open vocabulary. But its annotations are noisy and suffer from the missing-annotation problem. The COCO dataset, on the other hand, is very well annotated. But the coverage of visual objects and attributes is much lower than that in VG although we use both its 80 object classes and 91 stuff classes to include as diverse visual concepts as possible. We take the following steps to build a unified corpus by combining the four datasets.

1. First of all, to enhance visual concepts of tail classes, we perform class-aware sampling for OpenImages and Objects365 to get at least 2000 instances per class, resulting in 2.2M and 0.8M images, respectively.
2. To balance the contribution of each dataset, we merge the four datasets with 8 copies of COCO ( $8 \times 0.11M$ ), 8 copies of VG ( $8 \times 0.1M$ ), 2 copies of class-aware sampled Objects365 ( $2 \times 0.8M$ ) and one copy of the class-aware sampled OpenImages (2.2M).
3. To unify their object vocabularies, we use the VG vocabulary and its object aliases as the base vocabulary, merge a class from the other three datasets into a VG class if their class names or aliases match, and add a new class if no match is found.
4. Finally, we keep all VG classes that contain at least 30 instances, resulting in 1594 VG classes and 254 classes from the other three datasets that cannot be mapped to the VG vocabulary, resulting in a merged object detection dataset that contains 1848 classes.

Source	VG	COCO w/ stuff	Objects365	OpenImagesV5	Total
Image classes	97k 1594	111k 171	609k 365	1.67M 500	2.49M 1848
Sampling	$\times 8$	$\times 8$	CA-2k, $\times 2$	CA-2k	5.43M

Table 2: Statistics of the Vision pre-training datasets. In sampling,  $\times k$  means  $k$  copies in one epoch and “CA-2k” means class-aware sampling with at least 2000 instances per class.

**Model Architecture (FPN vs C4).** Although [23] shows that the FPN model outperforms the C4 model for object detection, recent studies [14] demonstrate that FPN does not provide more effective region features for VL tasks than C4, which is also confirmed by our experimental results <sup>2</sup>. We thus conduct a set of carefully designed experiments, as to be detailed in Appendix E, and find two main reasons for this. The first is that all layers in the C4 model used for region feature extraction are pre-trained using the ImageNet dataset while the multi-layer-perceptron (MLP) head of the FPN model are not. It turns out that the VG dataset is still too small to train a good enough visual features for VL tasks and using ImageNet-pre-trained weights is beneficial. The second is due to the different network architectures (CNN vs. MLP). The convolutional head used in C4 has a better inductive bias for encoding visual information than the MLP head of FPN. Therefore, in this study we use C4 architecture for VLP.

**Model Pre-Training.** Following the common practice in object detection training, we freeze the first convolution layer, the first residual block, and all the batch-norm layers. We also use several data augmentation methods, including horizontal flipping and multi-scale training. To train a detection model with the X152-C4 architecture, we initialize the model backbone from an ImageNet-5K checkpoint [40] and train for 1.8M iterations with a batch size of 16 images.

## 2.2 Injecting attribute information into the model

Following [2], we add an attribute branch to the pre-trained OD model, and then fine-tune the OD model on VG to inject attribute information (524 classes). Since the object representations are pre-trained in the object detection pre-training stage, we can focus the VG fine-tuning on learning attributes by picking a much larger attribute loss weight 1.25, compared to 0.5 used in [2, 14]. Thus, our fine-tuned model significantly outperforms previous models [2, 14] in detecting objects and attributes on VG.

## 2.3 Efficient region feature extractor for VL tasks

With a richer set of visual objects and attributes, the classical class-aware non-maximal suppression (NMS) post-processing takes a significantly larger amount of time to remove overlapped bounding boxes, making the feature extraction process extremely slow. To improve the efficiency, we replace the class-aware NMS with the class-agnostic NMS that only conducts the NMS operation once<sup>3</sup>. We also replace the time-consuming conv layers with dilation=2 used in [2] with conv layers without dilation. These two replacements make the region feature extraction process much faster than that in [2] without any accuracy drop on VL downstream tasks. We report the end-to-end inference time of VL models with different vision models on a Titan-X GPU and a CPU with a single thread in Table 20 in Appendix F.

In summary, the pre-trained OD model serves as the image understanding module, as in Equation (1), to produce vision presentations ( $q, v$ ) for downstream VL tasks. Here,  $q$  is the set of detected object names (in text) and  $v$  is the set of region features. Each region feature is denoted as  $(\hat{v}, z)$ , where  $\hat{v}$  is a  $P$ -dimensional representation from the input of the last linear classification layer of the detection head (*i.e.*,  $P = 2048$ ) and  $z$  is a  $R$ -dimensional position encoding of the region (*i.e.*,  $R = 6$ )<sup>4</sup>.

---

<sup>2</sup>We find in our experiments that using the same training process, the X152-C4 model even produces better object detection result than the X152-FPN model. See Appendix E for details.

<sup>3</sup>Counting the NMS in the RPN module, there are in total 2 NMS operations in our efficient region feature extractor.

<sup>4</sup>It includes coordinates of the bounding boxes, and height & width.

### 3 OSCAR+ Pre-training

The success of VLP lies in the use of a unifying model architecture for a wide range of VL tasks and the large-scale pre-training of the unified model using objectives that correlate with the performance metrics of these downstream VL tasks. In this study we pre-train an improved version of OSCAR [21], known as OSCAR+ models, to learn the joint image-text representations using image tags as anchors for image-text alignment.

#### 3.1 Pre-training corpus

We build our pre-training corpus based on three types of existing vision and VL datasets: (1) image captioning datasets with human-annotated captions as  $w$  and machine-generated<sup>5</sup> image tags as  $q$ , including COCO [25], Conceptual Captions (CC) [32], SBU captions [28] and flicker30k [42]; (2) visual QA datasets with questions as  $w$  and human-annotated answers as  $q$ , including GQA [13], VQA [8] and VG-QAs; (3) image tagging datasets with machine-generated<sup>6</sup> captions as  $w$  and human-annotated tags as  $q$ , including a subset of OpenImages (1.67M images). In total, the corpus contains 5.65 million unique images, 8.85 million text-tag-image triples. The detailed statistics are presented in Table 16 in the Appendix. The size of the pre-training corpus could have been significantly increased by combining large-scale image tagging datasets, such as the full set of OpenImages (9M images) and YFCC (92M images). We leave it to future work to leverage much larger corpora for model pre-training.

Loss $w'/q'$	$(w, q/q', v)$ All $q$ 's (OSCAR)	$(w/w', q, v)$ $q$ 's from QA	3-way contrastive All $w$ 's	All (OSCAR+) $q$ 's from QA
VQA (vqa-dev)	<b>69.8</b> $\pm$ 0.08	<b>70.1</b> $\pm$ 0.08	69.5 $\pm$ 0.05	<b>69.8</b> $\pm$ 0.06
COCO-IR	73.9 $\pm$ 0.2	<b>75.0</b> $\pm$ 0.2	<b>75.0</b> $\pm$ 0.7	<b>78.3</b> $\pm$ 0.3

Table 3: Effects of different pre-training contrastive losses on downstream tasks (R50-C4 as **Vision** module and 4-layer Transformer as **VL** module in (1)). COCO-IR metric is Image-to-Text retrieval R@1 at COCO 1K test set. **Blue** indicates the best result for a task and **Black** indicates the runner-up.

#### 3.2 Pre-training Objectives

There are two terms in the OSCAR+ pre-training loss as in Equation (2).

$$\mathcal{L}_{\text{Pre-training}} = \mathcal{L}_{\text{MTL}} + \mathcal{L}_{\text{CL3}}. \quad (2)$$

$\mathcal{L}_{\text{MTL}}$  is the Masked Token Loss defined on the text modality ( $w$  and  $q$ ), following closely [21]. (See Appendix B.2 for details.)  $\mathcal{L}_{\text{CL3}}$  is a novel *3-way Contrastive Loss*. Different from the binary contrastive loss used in OSCAR [21], the proposed *3-way Contrastive Loss* to effectively optimize the training objectives used for VQA [41] and text-image matching [6]<sup>7</sup>. As shown in Equation 3,  $\mathcal{L}_{\text{CL3}}$  takes into account two types of training samples  $x$ : the {caption, image-tags, image-features} triplets of the image captioning and image tagging data, and the {question, answer, image-features} triplets of the VQA data.

<sup>5</sup>We use the same model to extract visual features.

<sup>6</sup>We use the captioning model released by OSCAR [21].

<sup>7</sup>[6] uses a deep-learning-based text-image matching model to select the best caption candidate for a given image.

$$\mathbf{x} \triangleq (\underbrace{\mathbf{w}}_{\text{caption}}, \underbrace{\mathbf{q}, \mathbf{v}}_{\text{tags\&image}}) \quad \text{or} \quad (\underbrace{\mathbf{w}, \mathbf{q}}_{\text{Q\&A}}, \underbrace{\mathbf{v}}_{\text{image}}) \quad (3)$$

To compute contrastive losses, negative examples need to be constructed. We construct two types of negative (unmatched) triplets for the two types of training samples, respectively. One is the polluted “captions”  $(\mathbf{w}', \mathbf{q}, \mathbf{v})$  and the other the polluted “answers”  $(\mathbf{w}, \mathbf{q}', \mathbf{v})$ . To classify whether a caption-tags-image triplet contains a polluted caption is a text-image matching task. To classify whether a question-answer-image triplet contains a polluted answer is an answer selection task for VQA. Since the encoding of [CLS] can be viewed as a representation of the triplet  $(\mathbf{w}, \mathbf{q}, \mathbf{v})$ , we apply a fully-connected (FC) layer on top of it as a 3-way classifier  $f(\cdot)$  to predict whether the triplet is matched ( $c = 0$ ), contains a polluted  $\mathbf{w}$  ( $c = 1$ ), or contains a polluted  $\mathbf{q}$  ( $c = 2$ ). The 3-way contrastive loss is defined as

$$\mathcal{L}_{\text{CL3}} = -\mathbb{E}_{(\mathbf{w}, \mathbf{q}, \mathbf{v}; c) \sim \tilde{\mathcal{D}}} \log p(c | f(\mathbf{w}, \mathbf{q}, \mathbf{v})), \quad (4)$$

where the dataset  $(\mathbf{w}, \mathbf{q}, \mathbf{v}; c) \in \tilde{\mathcal{D}}$  contains 50% matched triples, 25%  $\mathbf{w}$ -polluted triples, and 25%  $\mathbf{q}$ -polluted triples. For efficient implementation, the polluted  $\mathbf{w}'$  is uniformly sampled from all  $\mathbf{w}$ 's (captions and questions) and  $\mathbf{q}'$  is uniformly sampled from all  $\mathbf{q}$ 's (tags and answers) in the corpus. As demonstrated in Table 3, when only the answer-polluted triplets are used, i.e.,  $(\mathbf{w}, \mathbf{q}', \mathbf{v})$  with  $\mathbf{q}'$  sampled from  $\mathbf{q}$ 's from QA corpus, the contrastive loss simulates closely the objective for the VQA task but not the text-image retrieval task. As a result, the pre-trained model can be effectively adapted to VQA, but not so to text-image retrieval. By contrast, the proposed 3-way contrastive loss transfers well to both tasks.

### 3.3 Pre-trained models

We pre-train two model variants, denoted as OSCAR+<sub>B</sub> and OSCAR+<sub>L</sub>, which are initialized with parameters  $\theta_{\text{BERT}}$  of BERT base ( $L = 12, H = 768, A = 12$ ) and large ( $L = 24, H = 1024, A = 16$ ), respectively, where  $L$  is the number of layers,  $H$  the hidden size, and  $A$  the number of self-attention heads. To ensure that the image region features have the same input embedding size as BERT, we transform the position-augmented region features using a linear projection via matrix  $\mathbf{W}$ . The trainable parameters are  $\theta = \{\theta_{\text{BERT}}, \mathbf{W}\}$ . OSCAR+<sub>B</sub> is trained for at least 1M steps, with learning rate  $1e^{-4}$  and batch size 1024. OSCAR+<sub>L</sub> is trained for at least 1M steps, with learning rate  $3e^{-5}$  and batch size 1024. The sequence length of language tokens  $[\mathbf{w}, \mathbf{q}]$  and region features  $\mathbf{v}$  are 35 and 50, respectively.

## 4 Adapting to VL Tasks

We adapt the pre-trained models to seven downstream VL tasks, including five understanding tasks and two generation tasks. Each task poses different challenges for adaptation. This section briefly introduces the tasks and our fine-tuning strategy. We refer the readers to Appendix C for details.

**VQA & GQA** These two are the most widely used understanding task for evaluating VL models in the research community. The tasks require the model to answer natural language questions based on an image. In this study, we perform experiments on the widely-used VQA v2.0 dataset [8] and GQA dataset [13]. Following the setting of [2], for each question, the model picks an answer from a shared answer set (i.e., 3,129 candidates for VQA, 1,852 candidates for GQA). When adapting a VLP model to the VQA task, we

construct the input by concatenating a given question, object tags and object region features, and then feed the [CLS] output from OSCAR+ to a task-specific linear classifier with a softmax layer for answer prediction.

**Image Captioning & NoCaps** The captioning task is to generate a natural language caption for an image. This is the most widely used VL generation task in the research community – the Image Captioning Leaderboard<sup>8</sup> hosts more than 260 models as of December 10, 2020. To enable caption generation, we fine-tune OSCAR+ using the seq2seq objective. Each training sample is converted to a triplet consisting of a caption, a set of image region features, and a set of object tags. We randomly mask out 15% of the caption tokens, and use the encoding of the remaining context (the triplet) to predict the masked tokens. Similar to VLP [21, 45], the self-attention mask is constrained such that a caption token can only attend to the tokens before its position to simulate a uni-directional generation process. All caption tokens have full attentions to image regions and object tags but not the other way around. During inference, we first encode the image regions, object tags, and a special token [CLS] as input. Then the model starts to generate a caption by feeding in a [MASK] token and sampling a token from a vocabulary based on the token probability output. Next, the [MASK] token in the previous input sequence is replaced with the sampled token and a new [MASK] is appended for the next word prediction. The generation process terminates when the model outputs the [STOP] token or the generated sentence exceeds a pre-defined max length. We perform image captioning experiments on the COCO image captioning dataset [25]. Novel Object Captioning at Scale [1] extends the image captioning task to test a model’s capability of describing novel objects from the Open Images dataset [17] which are unseen in the training corpus. Following the restriction guideline of NoCaps, we use the predicted Visual Genome and Open Images labels to form the input tag sequences, and directly train OSCAR+ on COCO without the initialization from pre-training. VIVO [9] proposed a VLP technique by only using image tagging data, and achieved SOTA results on NoCaps by fine-tuning on COCO captions. We reproduced VIVO with only one change, i.e., replacing its original vision model with our new vision model, and improved the VIVO performance significantly (short as VinVL+VIVO), as reported in Table 9.

**Image(-to-Text) Retrieval & Text(-to-Image) Retrieval** Both tasks require the model to calculate a similarity score between an image and a sentence. Thus, the task is widely used to directly measure the quality of the cross-modal VL representation. Following [21], we formulate the task as a binary classification problem, where given a matched image-text pair, we randomly select a different image or a different sentence to form an unmatched pair. The representation of [CLS] is used as the input to a classifier to predict a score indicating how likely the given pair is matched. In testing, the predicted score is used to rank a given image-text pairs of a query. Following [19], we report the top- $K$  retrieval results on both the 1K and 5K COCO test sets.

**NLVR2** The dataset is developed for joint reasoning about natural language and images [35]. The task is to determine whether a text description is true about a pair of images. For fine-tuning, we first construct two input sequences, each containing the concatenation of the given text description and one of the images, and then two [CLS] outputs from OSCAR+ are concatenated to form the input to a binary classifier for prediction.

---

<sup>8</sup>Image Captioning Leaderboard: <https://competitions.codalab.org/competitions/3221>

## 5 Experiments & Analysis

### 5.1 Main Results

To account for model parameter efficiency, we group the SoTA models in three categories: (i) SoTA<sub>S</sub> indicates the best performance achieved by small models prior to the Transformer-based VLP models. (ii) SoTA<sub>B</sub> indicates the best performance produced by VLP models of a similar size to BERT base. (iii) SoTA<sub>L</sub> indicates the best performance yielded by VLP models that have a similar size to BERT large.

Table 4 gives an overview of the results of OSCAR+ (short for OSCAR+ with our new OD model in this subsection) on seven VL tasks, compared to previous SoTAs<sup>9</sup>. OSCAR+ outperforms previous SoTA models on all tasks<sup>10</sup>, often by a significantly large margin. The result demonstrates the effectiveness of the region features produced by the new OD model.

Task	VQA		GQA		Image Captioning				NoCaps		Image Retrieval			Text Retrieval			NLVR2	
	test-dev	test-std	test-dev	test-std	B@4	M	C	S	C	S	R@1	R@5	R@10	R@1	R@5	R@10	dev	test-P
SoTA <sub>S</sub>	70.55	70.92	—	<b>63.17</b>	38.9	29.2	129.8	22.4	61.5	9.2	39.2	68.0	81.3	56.6	84.5	92.0	54.10	54.80
SoTA <sub>B</sub>	73.59	73.67	61.58	61.62	40.5	29.7	137.6	22.8	86.58	12.38	54.0	80.8	88.5	70.0	91.1	95.5	78.39	79.30
SoTA <sub>L</sub>	74.75	74.93	—	—	<b>41.7</b>	30.6	140.0	24.5	—	—	57.5	82.8	89.8	73.5	92.3	96.0	79.76	81.47
OSCAR+B	<b>75.95</b>	<b>76.12</b>	<b>65.05</b>	<b>64.65</b>	40.9	<b>30.9</b>	<b>140.6</b>	<b>25.1</b>	<b>92.46</b>	<b>13.07</b>	<b>58.1</b>	<b>83.2</b>	<b>90.1</b>	<b>74.6</b>	<b>92.6</b>	<b>96.3</b>	<b>82.05</b>	<b>83.08</b>
OSCAR+L	<b>76.52</b>	<b>76.60</b>	—	—	41.0	<b>31.1</b>	<b>140.9</b>	<b>25.2</b>	—	—	<b>58.8</b>	<b>83.5</b>	<b>90.3</b>	<b>75.4</b>	<b>92.9</b>	<b>96.2</b>	<b>82.67</b>	<b>83.98</b>
Δ	1.77↑	1.67↑	3.47↑	1.48↑	0.7↓	0.5↑	0.9↑	0.7↑	5.9↑	0.7↑	1.3↑	0.7↑	0.5↑	1.9↑	0.6↑	0.3↑	2.91↑	2.51↑

Table 4: An overall comparison with SoTAs on seven tasks. Δ indicates the improvement over SoTA. SoTA with subscript S, B, L indicates performance achieved by small models, and models with the model size similar to BERT base and large, respectively. SoTAs: VQA is from ERNIE-VIL [43], GQA is from NSM [12], NoCaps is from VIVO [9], NLVR2 is from VILLA [7], the rest tasks are from OSCAR [21].

Method	ViLBERT Base	VL-BERT Base	VisualBERT Base	LXMERT Base	12-in-1 Base	UNITER Base	OSCAR Large	VILLA Base	ERNIE-VIL Base	InterBERT Ensemble*	OSCAR+ Base	OSCAR+ Large	
Test-dev	70.63	70.50	70.80	72.42	73.15	72.27	73.24	73.16	73.61	73.59	73.69	72.62	74.75
Test-std	70.92	70.83	71.00	72.54	—	72.46	73.40	73.44	73.82	73.67	74.87	72.85	74.93

Table 5: Evaluation results on VQA. \* denotes the No.1 ensemble model of InterBERT Large on the VQA leaderboard.

Method	LXMERT	MMN [3]	12-in-1	OSCAR <sub>B</sub>	NSM [12]	OSCAR <sub>B</sub>
Test-dev	60.00	—	—	<b>61.58</b>	—	<b>65.05</b>
Test-std	60.33	60.83	60.65	61.62	<b>63.17</b>	<b>64.65</b>

Table 6: Evaluation results on GQA.

In Tables 5 to 11, we report the detailed results for each downstream task, respectively. (i) The **VQA** results are shown in Table 5, where our single OSCAR+<sub>B</sub> model outperforms the best ensemble model (InterBERT large [22]) on the VQA leaderboard as of Dec. 12, 2020<sup>11</sup>. (ii) The **GQA** results are shown in Table 6, where OSCAR+ is the first VLP model that outperforms the neural state machine (NSM) [12] which contains some sophisticated reasoning components deliberately designed for the task. (iii) The **Image Captioning** results on the public “Karpathy” 5k test split are shown in Table 7. Table 8 shows on a

<sup>9</sup> All the (single-model) SoTAs are from the published results. For all the tables in this paper, **Blue** indicates the best result for a task, and gray background indicates results produced by OSCAR+.

<sup>10</sup>The only exception is B@4 on image captioning.

<sup>11</sup>VQA leaderboard: <https://eval.ai/web/challenges/challenge-page/514/leaderboard/1386>

Method	cross-entropy optimization				CIDEr optimization			
	B@4	M	C	S	B@4	M	C	S
BUTD [2]	36.2	27.0	113.5	20.3	36.3	27.7	120.1	21.4
VLP [45]	36.5	28.4	117.7	21.3	39.5	29.3	129.3	23.2
AoANet [10]	37.2	28.4	119.8	21.3	38.9	29.2	129.8	22.4
OSCAR <sub>B</sub> [21]	36.5	30.3	123.7	23.1	40.5	29.7	137.6	22.8
OSCAR <sub>L</sub> [21]	37.4	<b>30.7</b>	127.8	<b>23.5</b>	<b>41.7</b>	30.6	140.0	24.5
OSCAR+ <sub>B</sub>	<b>38.5</b>	30.3	<b>129.3</b>	<b>23.6</b>	40.9	<b>30.9</b>	<b>140.6</b>	<b>25.1</b>
OSCAR+ <sub>L</sub>	<b>38.5</b>	<b>30.4</b>	<b>130.8</b>	23.4	<b>41.0</b>	<b>31.1</b>	<b>140.9</b>	<b>25.2</b>

Table 7: Image captioning evaluation results (single model) on COCO “Karpathy” test split. (Note: B@4: BLEU@4, M: METEOR, C: CIDEr, S: SPICE.)

Method	BLEU@1		BLEU@2		BLEU@3		BLEU@4		METEOR		ROUGE-L		CIDEr-D	
	c5	c40	c5	c40										
BUTD [2]	80.2	95.2	64.1	88.8	49.1	79.4	36.9	68.5	27.6	36.7	57.1	72.4	117.9	120.5
AoANet [10]	81.0	95.0	65.8	89.6	51.4	81.3	39.4	71.2	29.1	38.5	58.9	74.5	126.9	129.6
X-Transformer [29]	81.9	95.7	66.9	90.5	52.4	82.5	40.3	72.4	29.6	39.2	59.5	75.0	131.1	133.5
OSCAR+	<b>81.9</b>	<b>96.9</b>	<b>66.9</b>	<b>92.4</b>	<b>52.6</b>	<b>84.7</b>	<b>40.4</b>	<b>74.9</b>	<b>30.6</b>	<b>40.8</b>	<b>60.4</b>	<b>76.8</b>	<b>134.7</b>	<b>138.7</b>

Table 8: Leaderboard of the state-of-the-art image captioning models on the COCO online testing.

Method	in-domain		near-domain		out-of-domain		overall		in-domain		near-domain		out-of-domain		overall	
	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE	CIDEr	SPICE
Validation Set															Test Set	
UpDown <sup>+</sup>	79.3	12.4	73.8	11.4	71.7	9.9	74.3	11.2	76.0	11.8	74.2	11.5	66.7	9.7	73.1	11.2
OSCAR <sub>B</sub> *	83.4	12.0	81.6	12.0	77.6	10.6	81.1	11.7	81.3	11.9	79.6	11.9	73.6	10.6	78.8	11.7
OSCAR <sub>I</sub> *	85.4	11.9	84.0	11.7	80.3	10.0	83.4	11.4	84.8	12.1	82.1	11.5	73.8	9.7	80.9	11.3
Human [1]	84.4	<b>14.3</b>	85.0	<b>14.3</b>	<b>95.7</b>	<b>14.0</b>	87.1	<b>14.2</b>	80.6	<b>15.0</b>	84.6	<b>14.7</b>	<b>91.6</b>	<b>14.2</b>	85.3	<b>14.6</b>
VIVO*	92.2	12.9	87.8	12.6	<b>87.5</b>	11.5	88.3	12.4	89.0	12.9	87.8	12.6	<b>80.1</b>	11.1	<b>86.6</b>	12.4
VinVL*	<b>96.8</b>	13.5	<b>90.7</b>	13.1	87.4	11.6	<b>90.9</b>	12.8	<b>93.8</b>	13.3	<b>89.0</b>	12.8	66.1	10.9	85.5	12.5
VinVL+VIVO	<b>103.7</b>	<b>13.7</b>	<b>95.6</b>	<b>13.4</b>	83.8	<b>11.9</b>	<b>94.3</b>	<b>13.1</b>	<b>98.0</b>	<b>13.6</b>	<b>95.2</b>	<b>13.4</b>	78.0	<b>11.5</b>	<b>92.5</b>	<b>13.1</b>

Table 9: NoCaps evaluation results. All the models are trained on COCO without additional image-caption pairs following the restriction of NoCaps. (UpDown<sup>+</sup> is UpDown+ELMo+CBS, the models with \* is +SCST+CBS, VinVL+VIVO is with SCST only.)

Method ↓	BERT	1K Test Set						5K Test Set					
		Text Retrieval			Image Retrieval			Text Retrieval			Image Retrieval		
		R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
Unicoder-VL [19]	B	84.3	97.3	99.3	69.7	93.5	97.2	62.3	87.1	92.8	46.7	76.0	85.3
UNITER [4]	B	—	—	—	—	—	—	63.3	87.0	93.1	48.4	76.7	85.9
OSCAR	L	—	—	—	—	—	—	66.6	89.4	94.3	51.7	78.4	86.9
OSCAR	B	88.4	99.1	99.8	75.7	95.2	98.3	70.0	91.1	95.5	54.0	80.8	88.5
OSCAR	L	<b>89.8</b>	<b>98.8</b>	<b>99.7</b>	<b>78.2</b>	<b>95.8</b>	<b>98.3</b>	73.5	92.2	96.0	57.5	82.8	89.8
OSCAR+	B	<b>89.8</b>	<b>98.8</b>	<b>99.7</b>	<b>78.2</b>	95.6	98.0	<b>74.6</b>	<b>92.6</b>	<b>96.3</b>	<b>58.1</b>	<b>83.2</b>	<b>90.1</b>
OSCAR+	L	<b>90.8</b>	<b>99.0</b>	<b>99.8</b>	<b>78.8</b>	<b>96.1</b>	<b>98.5</b>	<b>75.4</b>	<b>92.9</b>	<b>96.2</b>	<b>58.8</b>	<b>83.5</b>	<b>90.3</b>

Table 10: Text and Image retrieval evaluation on the COCO 1K and 5K test sets. (B for Base, L for Large)

Method	MAC	VisualBERT base	LXMERT base	12-in-1 base	UNITER		OSCAR		VILLA		OSCAR+			
				base	base	large	base	large	base	large	base	large	base	large
Dev	50.8	67.40	74.90	—	77.14	78.40	78.07	79.12	78.39	79.76	<b>82.05</b>	<b>82.67</b>		
Test-P	51.4	67.00	74.50	78.87	77.87	79.50	78.36	80.37	79.47	81.47	<b>83.08</b>	<b>83.98</b>		

Table 11: Evaluation results on NLVR2.

vision \ vl	no VLP	OSCAR <sub>B</sub> [21]	OSCAR+ <sub>B</sub> (ours)
R101-C4 [2]	68.52 ±0.11	72.38	72.46±0.05
VinVL (ours)	71.34 ±0.17	–	74.90±0.05

Table 12: Effects of vision (V) and vision-language (VL) pre-training on VQA.

concise version of the COCO image captioning online leaderboard<sup>12</sup>. The online testing setting reports the results on 40K images, with 5 reference captions (c5) and 40 reference captions (c40) per image. At the time of submitting this paper, our single model achieves No.1 on the entire leaderboard, outperforming all 263 models, including many ensemble (and anonymous) models. (iv) The Novel Object Captioning (**NoCaps**) results are shown in Table 9. Without any VLP, i.e. by directly training a BERT-based captioning model on COCO, the model with our new visual features (denoted as VinVL) already surpasses the human performance in CIDEr<sup>13</sup>. By adding VIVO [9] pre-training, our VinVL improves the original VIVO result by 6 CIDEr points and creates a new SoTA. (v) Overall, on all these tasks (VQA in Table 5, Image Captioning in Table 7, NoCaps in Table 9, Image-Text Retrieval in Table 10, NLVR2 in Table 11), we show that OSCAR+<sub>B</sub> can match or outperform previous SoTA large models, and OSCAR+<sub>L</sub> substantially uplifts the SoTA.

## 5.2 Ablation Analysis

We select the VQA task for the ablation study because its evaluation metric is well-defined and the task has been used as a testbed for all VLP models. To assist our analysis, we create a local validation set, vqa-dev, out of the standard validation set to select the best model during training for evaluation. vqa-dev contains randomly sampled 2K images and their corresponding questions, amounting to 10.4K image-QA pairs in total. Except for Table 4 and 5, all our VQA results are reported on this vqa-dev set. Unless otherwise specified, the reported STD is half of the difference of two runs of the VQA training with different random seeds.

In VQA, the VL model  $y = \mathbf{VL}(\mathbf{w}, \mathbf{q}, \mathbf{v})$  has  $\mathbf{w}$  as the question and  $y$  as the answer. We focus on studying the effect of visual features  $\mathbf{v}$  produced by different Vision models  $\mathbf{Vision}(Img)$  to better understand their relative contribution in the VQA performance. To eliminate the impact of using different tags  $\mathbf{q}$ , we use the same tags in the VQA models of OSCAR [21]. All the ablation experiments are conducted using models of the BERT-base size.

**How much do the V and VL matter to the SoTA?** Table 12 shows the VQA results with different vision models, i.e., R101-C4 model from [2] and our X152-C4 model pre-trained with 4 datasets (VinVL), and with different VLP methods, i.e., no VLP, OSCAR [21] and our OSCAR+. Taking the OSCAR<sub>B</sub> model with R101-C4 features as the baseline, the OSCAR+<sub>B</sub> model with our X152-C4 features improves the absolute accuracy from 72.38 to 74.90, in which the OSCAR+ pre-training contributes 5% of the gain (i.e., 72.38 → 72.46) and the vision pre-training (improved visual features) 95% (i.e., 72.46 → 74.90). This demonstrates that vision representations matter significantly in VLP and downstream tasks.

Taking the “no VLP” model with R101-C4 features as the baseline, Table 12 shows that the gains of VinVL ( $71.34 - 68.52 = 2.82$ ) and VLP ( $72.46 - 68.52 = 3.94$ ) are additive ( $74.90 - 68.52 \approx 2.82 + 3.94$ ).

<sup>12</sup>Image Captioning Leaderboard: <https://competitions.codalab.org/competitions/3221#results>

<sup>13</sup>NoCaps leaderboard: <https://eval.ai/web/challenges/challenge-page/355/leaderboard/1011>

model data	R50-FPN	R50-C4	R101-C4 [2]	X152-C4
VG	67.35 $\pm$ 0.26	67.86 $\pm$ 0.31	68.52 $\pm$ 0.11	69.10 $\pm$ 0.06
4Sets $\rightarrow$ VG	68.3 $\pm$ 0.11	68.39 $\pm$ 0.16	–	71.34 $\pm$ 0.17

Table 13: Ablation of model size and data size on training vision models.

Model Pre-training dataset	R50-FPN		R50-C4		X152-C4	
	ImageNet	4Sets	ImageNet	4Sets	ImageNet5k	4Sets
COCO <i>mAP</i>	40.2 [40]	44.78*	38.4 [40]	42.4	42.17	50.51
VG obj <i>mAP</i> <sup>50</sup>	9.6	11.3	9.6	12.1	11.2	13.8
attr <i>mAP</i> with gt boxes	5.4	5.5	6.3	6.1	6.6	7.1

\* Since our four pre-training datasets contain Objects365, it is not surprising that we obtain better results than 42.3 *mAP*<sup>50</sup> in [31], which is obtained by pre-training on Objects365.

Table 14: Effect of vision pre-training on object detection tasks.

This is intuitive because vision pre-training and VLP improve the Vision model **Vision**(*Img*) and VL model **VL**(*w*, *q*, *v*) separately. This also indicates that our pre-trained vision model can be utilized in any VL models by directly replacing their vision models, such as R101-C4 [2], with ours.

**How much do data and model sizes matter to the new vision model?** The improvement of VQA from R101-C4 [2] to VinVL (ours) in Table 12 is a compound effect of increasing model size (from R101-C4 to X152-C4) and data size (from VG to our merged four OD datasets). Table 13 shows the ablation of the two factors without VLP. Although VG’s large object and attribute vocabulary allows to learn rich semantic concepts, VG does *not* contain large amounts of annotations for effective training of deep models. Vision models trained using the merged four OD datasets perform much better than VG-only-trained models, and the improvement is larger with the increase of the model size.<sup>14</sup>

**How much does OD model architecture matter?** The choice of model architecture affects the VQA performance. Table 13 shows that R50-FPN under-performs R50-C5 when they are trained only on VG; but the performance gap diminishes when both are trained on the merged dataset (4Sets). A detailed comparison between FPN and C4 architectures is presented in Appendix E.

**How much does OD pre-training matter for object detection tasks?** Table 14 presents the object detection results on COCO and the object-attribute detection results on VG (1594 object classes, 524 attribute classes). The results show that OD pre-training benefits the object detection tasks. Note that the mAP on VG is much lower than that on typical OD datasets (such as COCO) due to two reasons: (1) VG contains a large number of object classes with limited and extremely unbalanced annotations, (2) there are many missing annotations in the VG evaluation data.<sup>15</sup> Although the mAP numbers are low, the detection result using X152-C4 is reasonably good; see Appendix A for more visualizations. We also see that FPN models

<sup>14</sup>The R101-C4 model in Table 13 is exactly the VG-pre-trained model from [2]. We do not train this model on our merged OD dataset because this model architecture is old-fashioned and is slow to train.

<sup>15</sup>As a reference, the R101-C4 model from [2] on VG with 1600 objects and 400 attributes has mAP of 8.7/7.8 evaluated in our code, whereas it was reported as 10.2/7.8 due to differences in OD evaluation pipeline.

Dataset name #obj & #attr	ImageNet 1000 & 0	VG-obj 317 & 0	VG w/o attr 1594 & 0	VG 1594 & 524	4Sets→VG 1848 & 524
R50-C4 + BERT <sub>B</sub>	66.13±0.04	64.25±0.16	66.51±0.11	67.86±0.31	68.39±0.16

Table 15: Effect of object-attribute vocabulary. We use all grid features (maximal 273) for the ImageNet classification model (first column), and maximal 50 region features for OD models (other columns).

perform consistently worse in attribute detection than C4 models, neither do FPN models show any advantage in object detection on VG. This contributes to the inferior performance of FPN, compared to C4, on downstream VL tasks, as discussed in Section 2.1.

**How much does the diversity of visual concepts, i.e., object and attribute vocabularies, matter?** We directly train vision models on different datasets, including (1) standard ImageNet with 1K classes (ImageNet), (2) Visual Genome with 317 object classes (VG-obj) that are shared with COCO 80 classes and OpenImagesV5 500 classes, (3) VG with all 1594 object classes (VG w/o attr), (4) VG with 1594 object classes and 524 attribute classes (VG), and (5) the merged OD dataset (4Sets) for pre-training and VG for fine-tuning. For all the OD models (the last four columns in Table 15), we initialize the OD training with an ImageNet-pre-trained classification model, and use maximal 50 *region* features per image as input to the VL fusion module. For the ImageNet pre-trained classification model (the second column in Table 15), we use all the *grid* features (maximal 273) for each image<sup>16</sup>. The results show that

- In general, vocabularies with richer objects lead to better VQA results: VG-obj < ImageNet < VG w/o attr. The VG-obj vocabulary contains 79 of 80 COCO classes (only missing potted plant) and 313 of 500 OpenImagesV5 classes, and is a good approximation of common object classes of typical OD tasks. However, our results show that this vocabulary is not rich enough for VL tasks because it misses many important visual concepts (e.g., sky, water, mountain, etc.) which are crucial for VL tasks, as also illustrated by the comparison of detected regions in Figure 1.<sup>17</sup>.
- Attribute information is crucial to VL tasks: models trained with attributes (VG and 4Sets→VG) are significantly better than those without attributes.
- Even for the small vision model R50-C4, vision pre-training improves visual features for VQA, i.e., 4Sets→VG is the best performer.

## 6 Conclusion

In this paper we have presented a new recipe to pre-train an OD model for VL tasks. Compared to the most widely used *bottom-up and top-down* model [2], the new model is bigger, better-designed for VL tasks, and pre-trained on much larger text-image corpora, and thus can generate visual features for a richer collection of visual objects and concepts that are crucial for VL tasks. We validate the new model via a comprehensive empirical study where we feed the visual features to a VL fusion model which is pre-trained on a large-scale paired text-image corpus and then fine-tuned on seven VL tasks. Our results show that the new OD model can substantially uplift the SoTA results on all seven VL tasks across multiple public benchmarks. Our

<sup>16</sup>Our use of grid feature follows PixelBert [1]. See Appendix F for details.

<sup>17</sup>Using the same training procedure on VG, we trained an R50-C4 model on the OpenImagesV5 dataset (500 classes). Using the region features produced by this model, the VQA performance is 63.55±0.14. The result is slightly worse than that of VG-obj because both VG and VQA images are from the COCO dataset but OpenImages images are not.

ablation study shows that the improvement is mainly attributed to our design choices regarding diversity of object categories, visual attribute training, training data scale, model size, and model architecture.

## Acknowledgement

We thank Xi Yin for her contributions to this project while she was in Microsoft. We thank Xiyang Dai for his conjecture that C4 arch is better than FPN because C4 arch makes better use of ImageNet initialization weights.

## References

- [1] Harsh Agrawal, Karan Desai, Yufei Wang, Xinlei Chen, Rishabh Jain, Mark Johnson, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. nocaps: novel object captioning at scale. In *ICCV*, 2019. [3](#), [8](#), [10](#), [23](#)
- [2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, 2018. [1](#), [2](#), [3](#), [5](#), [7](#), [10](#), [11](#), [12](#), [13](#), [17](#), [18](#), [20](#), [21](#), [22](#), [23](#), [26](#), [28](#), [30](#)
- [3] Wenhui Chen, Zhe Gan, Linjie Li, Yu Cheng, William Wang, and Jingjing Liu. Meta module network for compositional visual reasoning. *arXiv preprint arXiv:1910.03230*, 2019. [9](#), [22](#)
- [4] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*, 2019. [1](#), [10](#), [24](#)
- [5] Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. Vse++: Improved visual-semantic embeddings. *arXiv preprint arXiv:1707.05612*, 2(7):8, 2017. [24](#)
- [6] Hao Fang, Saurabh Gupta, Forrest Iandola, Rupesh K Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C Platt, et al. From captions to visual concepts and back. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1473–1482, 2015. [6](#)
- [7] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. *arXiv preprint arXiv:2006.06195*, 2020. [9](#)
- [8] 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. [3](#), [6](#), [7](#), [22](#)
- [9] Xiaowei Hu, Xi Yin, Kevin Lin, Lijuan Wang, Lei Zhang, Jianfeng Gao, and Zicheng Liu. Vivo: Surpassing human performance in novel object captioning with visual vocabulary pre-training. *arXiv preprint arXiv:2009.13682*, 2020. [2](#), [8](#), [9](#), [10](#), [11](#)
- [10] Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. Attention on attention for image captioning. In *ICCV*, 2019. [10](#), [23](#)
- [11] Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. Pixel-bert: Aligning image pixels with text by deep multi-modal transformers. *arXiv preprint arXiv:2004.00849*, 2020. [13](#)
- [12] Drew Hudson and Christopher D Manning. Learning by abstraction: The neural state machine. In *NeurIPS*, 2019. [3](#), [9](#)
- [13] Drew A Hudson and Christopher D Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. *arXiv preprint arXiv:1902.09506*, 2019. [3](#), [6](#), [7](#), [22](#)
- [14] Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. In defense of grid features for visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10267–10276, 2020. [5](#), [26](#)
- [15] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015. [23](#), [24](#)
- [16] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowd-sourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73, 2017. [1](#), [2](#)

- [17] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018. 1, 8, 23
- [18] Kuang-Huei Lee, Xi Chen, Gang Hua, Houdong Hu, and Xiaodong He. Stacked cross attention for image-text matching. In *ECCV*, 2018. 24
- [19] Gen Li, Nan Duan, Yuejian Fang, Dixin Jiang, and Ming Zhou. Unicoder-VL: A universal encoder for vision and language by cross-modal pre-training. *arXiv preprint arXiv:1908.06066*, 2019. 1, 8, 10, 23, 24
- [20] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019. 1
- [21] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer, 2020. 1, 2, 3, 6, 8, 9, 10, 11, 19, 20, 22, 23, 27
- [22] Junyang Lin, An Yang, Yichang Zhang, Jie Liu, Jingren Zhou, and Hongxia Yang. Interbert: Vision-and-language interaction for multi-modal pretraining. *arXiv preprint arXiv:2003.13198*, 2020. 9
- [23] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 3, 5
- [24] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 3
- [25] 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, 3, 6, 8, 22, 23
- [26] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. VilBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019. 1
- [27] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-Task vision and language representation learning. *arXiv preprint arXiv:1912.02315*, 2019. 24
- [28] Vicente Ordonez, Girish Kulkarni, and Tamara L Berg. Im2text: Describing images using 1 million captioned photographs. In *NeurIPS*, 2011. 6
- [29] Yingwei Pan, Ting Yao, Yehao Li, and Tao Mei. X-linear attention networks for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10971–10980, 2020. 10
- [30] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *CVPR*, 2017. 23
- [31] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 8430–8439, 2019. 1, 12
- [32] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Annual Meeting of the Association for Computational Linguistics*, 2018. 6, 23
- [33] Botian Shi, Lei Ji, Pan Lu, Zhendong Niu, and Nan Duan. Knowledge aware semantic concept expansion for image-text matching. In *IJCAI*, 2019. 24
- [34] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. VL-BERT: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019. 1
- [35] Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. A corpus for reasoning about natural language grounded in photographs. *arXiv preprint arXiv:1811.00491*, 2018. 3, 8, 24
- [36] Hao Tan and Mohit Bansal. LXMERT: Learning cross-modality encoder representations from transformers. *EMNLP*, 2019. 1
- [37] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: A simple and strong anchor-free object detector. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020. 3

- [38] Yaxiong Wang, Hao Yang, Xueming Qian, Lin Ma, Jing Lu, Biao Li, and Xin Fan. Position focused attention network for image-text matching. *arXiv preprint arXiv:1907.09748*, 2019. [24](#)
- [39] Zihao Wang, Xihui Liu, Hongsheng Li, Lu Sheng, Junjie Yan, Xiaogang Wang, and Jing Shao. CAMP: Cross-Modal adaptive message passing for text-image retrieval. In *ICCV*, 2019. [24](#)
- [40] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. <https://github.com/facebookresearch/detectron2>, 2019. [5](#), [12](#), [29](#)
- [41] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 21–29, 2016. [6](#)
- [42] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014. [6](#)
- [43] Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. *arXiv preprint arXiv:2006.16934*, 2020. [9](#)
- [44] Zhedong Zheng, Liang Zheng, Michael Garrett, Yi Yang, and Yi-Dong Shen. Dual-path convolutional image-text embedding with instance loss. *arXiv preprint arXiv:1711.05535*, 2017. [24](#)
- [45] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and VQA. *AAAI*, 2020. [1](#), [8](#), [10](#), [22](#), [23](#)

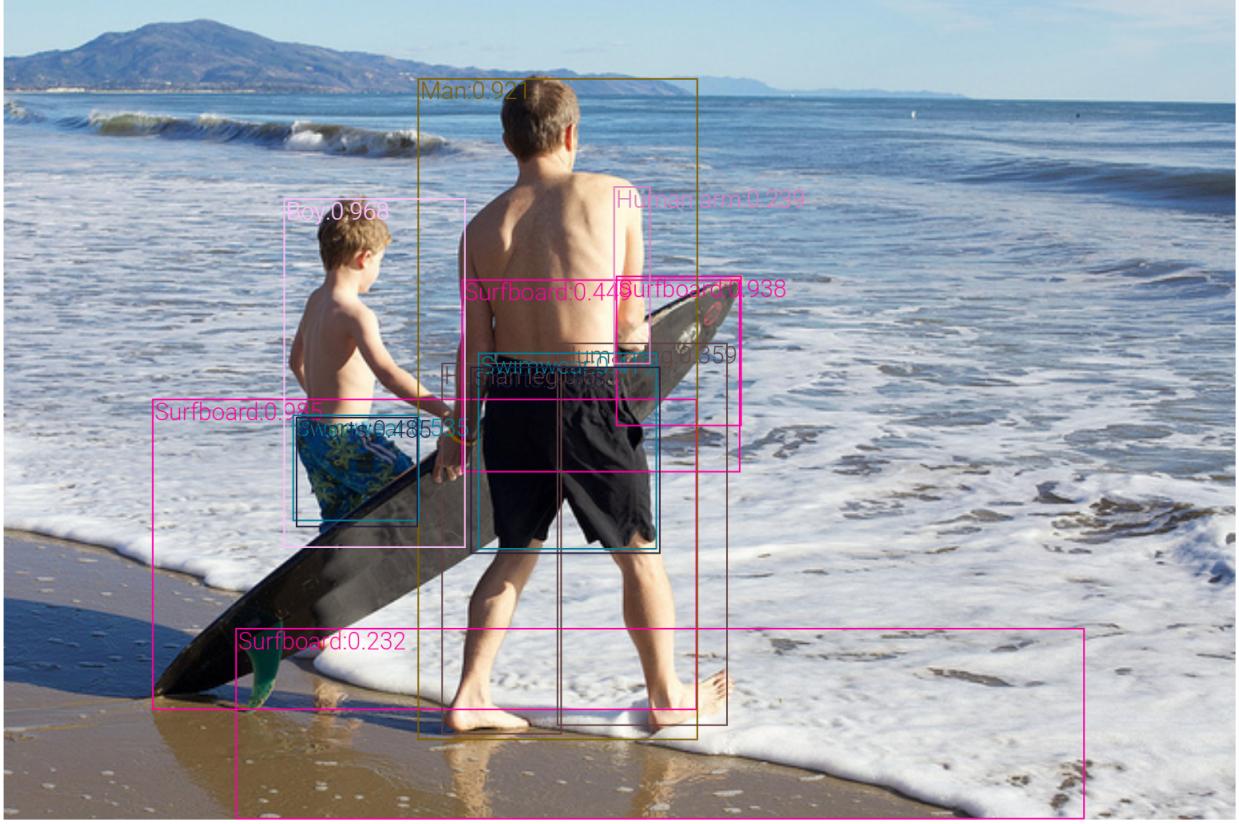


Figure 2: Predictions from X152-FPN trained on OpenImages. Test image: COCO\_test2015\_000000028839

## A Qualitative study of three pre-trained vision models

We apply three (pre-trained) object detection models on the image in Figure 1 and list their detection results for a more detailed comparison.

Detections from X152-FPN trained on Open Images V5. See Figure 2:

Surfboard; Surfboard; Surfboard; Surfboard; Man; Human leg; Human leg; Swimwear; Swimwear; Shorts; Shorts; Boy; Human arm.

Detections from R101-C4 trained on VG by Anderson *et al.* [2]. There are obviously wrong detections, marked in red. See Figure 3 (top):

black shorts; young, shirtless, standing, barefoot, surfing, little, playing boy; shirtless, standing, barefoot, walking, wet, surfing, young man; tan, bare, shirtless back; blue, clear, cloudy, hazy, light blue sky; young, shirtless, standing, surfing, barefoot, little boy; brown, short, wet, blond hair; brown, short, wet, blond hair; small, crashing wave; white, wet surfboard; white, crashing, big, rolling wave; wet, tan surfboard; green, blue fin; blue, calm, choppy, wavy, ocean, splashing, foamy, water, rough, sandy, wet ocean; wet, calm, sandy, splashing, wavy water; white, wet surfboard; bare, wet foot; blue, colorful, multi colored, floral shorts; calm, choppy, water, rough,

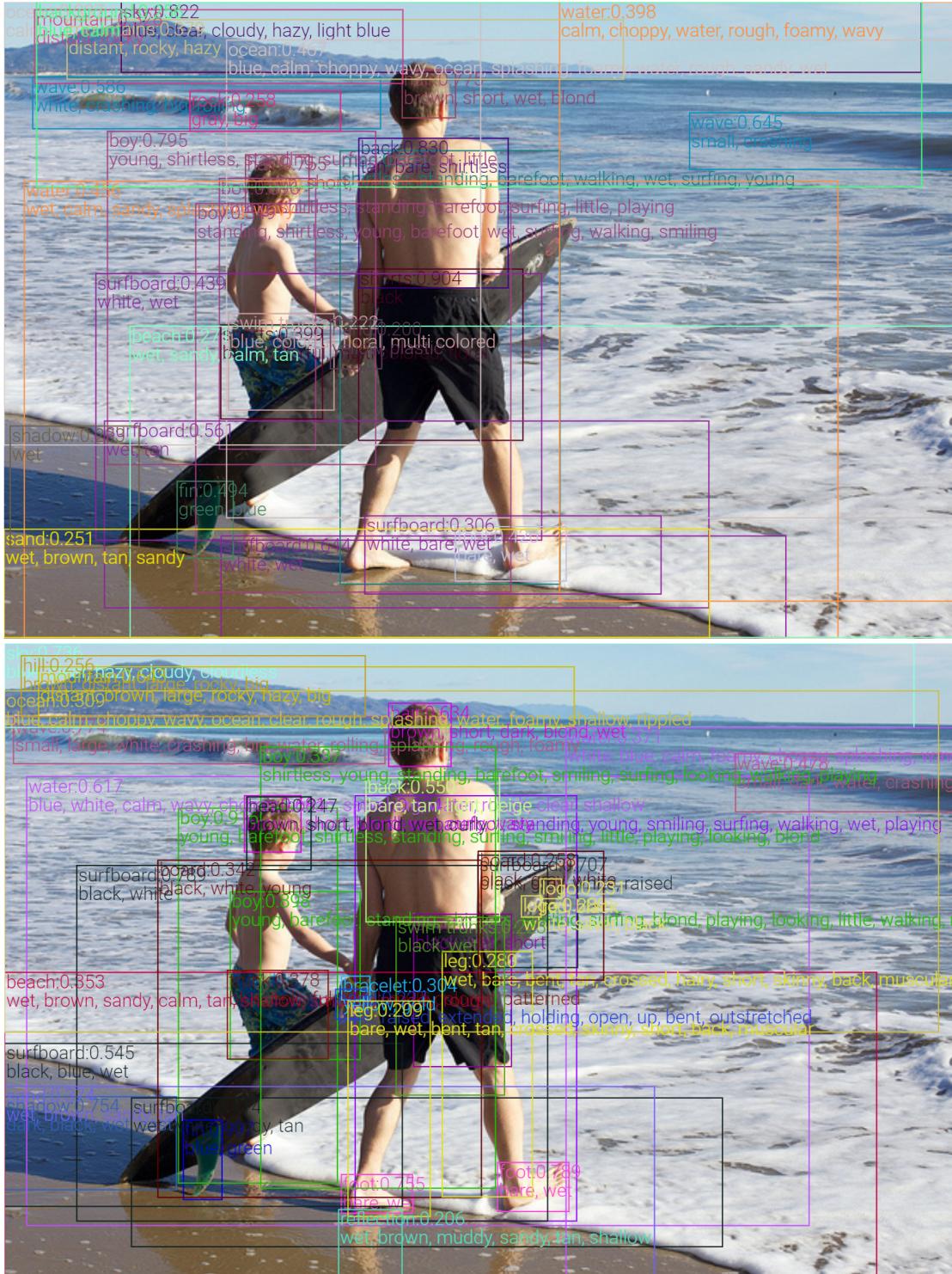


Figure 3: Predictions from R101-C4 trained on VG from [2] (top), X152-C4 pre-trained on 4 OD datasets and finetuned on VG (bottom). Test image: COCO\_test2015\_000000028839

foamy, wavy water; distant, rocky, hazy mountains; standing, shirtless, young, barefoot, wet, surfing, walking, smiling boy; calm ocean; distant, rocky mountain; white, bare, wet surfboard; wet, sandy, calm, tan beach; **gray, big rock; blue, calm background**; wet, brown, tan, sandy sand; wet shadow; blue, colorful, floral, multi colored swim trunks; **yellow, plastic hand.**

Detections from our pre-trained X152-C4 model pre-trained on four datasets and fine-tuned on VG. There are some repetitive detections, but no obvious wrong detections. See Figure 3 (bottom):

blue, green fin; young, barefoot, shirtless, standing, surfing, smiling, little, playing, looking, blond boy; young, barefoot, standing, shirtless, smiling, surfing, blond, playing, looking, little, walking, riding boy; shirtless, barefoot, standing, young, smiling, surfing, walking, wet, playing man; bare, wet foot; black, white surfboard; small, large, white, crashing, big, water, rolling, splashing, rough, foamy wave; bare, wet foot; dark, black, wet, cast shadow; blue, clear, hazy, cloudy, cloudless sky; black, gray, white, raised surfboard; black, wet, short short; brown, short, blond, wet, curly, wavy hair; distant, brown, large, rocky, hazy, big mountain; brown, short, dark, blond, wet hair; blue, white, calm, wavy, choppy, ocean, splashing, water, rough, clear, shallow water; bare, tan, light, beige back; black, blue, wet surfboard; small, dark, water, crashing, rolling, splashing, big wave; wet, white, sandy, tan surfboard; blue, colorful, floral, multi colored, patterned trunk; wet, brown, sandy, tan sand; white, blue, calm, foamy, choppy, splashing, wavy, ocean, rough, water, clear, shallow water; wet, brown, sandy, calm, tan, shallow, smooth, muddy, rough beach; black, white, young board; shirtless, young, standing, barefoot, smiling, surfing, looking, walking, playing boy; blue, calm, choppy, wavy, ocean, clear, rough, splashing, water, foamy, shallow, rippled ocean; yellow, gold bracelet; white, silver, black logo; wet, bare, bent, tan, crossed, hairy, short, skinny, back, muscular, extended, outstretched leg; black, gray, white board; brown, distant, large, rocky, big hill; brown, short, blond, wet, curly head; red, black logo; bare, raised, extended, holding, open, up, bent, outstretched hand; black, wet swim trunks; bare, wet, bent, tan, crossed, skinny, short, back, muscular leg; wet, brown, muddy, sandy, tan, shallow reflection.

## B OSCAR+ pre-training

### B.1 Pre-training Corpus

Table 16 shows the statistics of image and text of the pre-training corpora. In our ablation study, we have corpora of three different sizes: ‘Small’, ‘Medium’, ‘Large’. Different from OSCAR [21], we make use of image tagging datasets OpenImages, by generating captions using OSCAR’s image captioning model to form triplets of “(generated caption, image tags, image features)” for the OSCAR+ pre-training. Thanks to this self-training technique, our pre-training corpus can be scaled to a much larger scale by making use of large-scale image tagging datasets, e.g., OpenImages (9M) and YFCC (92M).

Small	0.22M Images, 2.5M QAs, 0.7M captions							
Medium	1.89M Images, 2.5M QAs, 0.7M captions, 1.67M pseudo-captions							
Large	5.65M Images, 2.5M QAs, 4.68M captions, 1.67M pseudo-captions							
Source	VQA (train)	GQA (bal-train)	VG-QA (train)	COCO (train)	Flickr30k (train)	OpenImages (od train)	CC (train)	SBU (all)
Image/Text	83k/545k	79k/1026k	87k/931k	112k/559k	29k/145k	1.67M/1.67M	3.1M/3.1M	875k/875k
$w, q, v$	Question, Answer, ImageFeatures			(Generated) Caption, (Generated) ImageTags, ImageFeatures				

Table 16: Statistics of the pre-training corpus.

## B.2 OSCAR+ pre-training objectives

**Masked Token Loss: A Loss Mimics Image Captioning.** The word tokens of image captions (questions)  $w$  and word tokens of object tags (answers)  $q$  share the same linguistic semantic space, and the Masked Token Loss (MTL) is applied on tokens of both  $w$  and  $q$ . We define the *discrete token sequence* as  $\mathbf{h} \triangleq [\mathbf{w}, \mathbf{q}]$ , and apply the Masked Token Loss (MTL) for pre-training. At each iteration, we randomly mask each input token in  $\mathbf{h}$  with probability 15%, and replace the masked one  $h_i$  with a special token [MASK]. The goal of training is to predict these masked tokens based on their surrounding tokens  $h_{\setminus i}$  and all image features  $v$  by minimizing the negative log-likelihood:

$$\mathcal{L}_{\text{MTL}} = -\mathbb{E}_{(\mathbf{v}, \mathbf{h}) \sim \mathcal{D}} \log p(h_i | \mathbf{h}_{\setminus i}, \mathbf{v}) \quad (5)$$

This is the same MTL as in OSCAR [21] and similar to the masked language model used by BERT. The masked word or tag needs to be recovered from its surrounding context, with additional image information to help ground the learned word embeddings in the vision context.

**3-way Contrastive Loss: A Loss Mimics Text-Image Retrieval and Visual Question Answering Simultaneously.** We present our 3-way contrastive loss in Section 3.2 in the main paper.

## B.3 Ablation of the two new techniques

**Effect of self-training: Leveraging Image Tagging data.** In Figure 4, we show the effect of self-training by making use of tagging data in OSCAR+, by fine-tuning OSCAR+ pre-training checkpoints on VQA. Compared with “OSCAR+, Small; VinVL” (green), “OSCAR+, Medium; VinVL” (yellow) adds the 1.7M OpenImages Tagging data into pre-training and its performance gets improved significantly, demonstrating the effect of self-training by making use of tagging data. As baselines, we also provide performance of OSCAR and OSCAR+ with image features from [2], which clearly demonstrates that the new image features pre-trained by VinVL matter significantly in the VL pre-training and VL downstream tasks.

**Effect of the new 3-way contrastive loss.** As illustrated in Table 3, with the new 3-way contrastive loss, the VQA performance keeps the same compared with the OSCAR pre-training, while the Text-Image Retrieval performance improves significantly compared with the OSCAR pre-training.

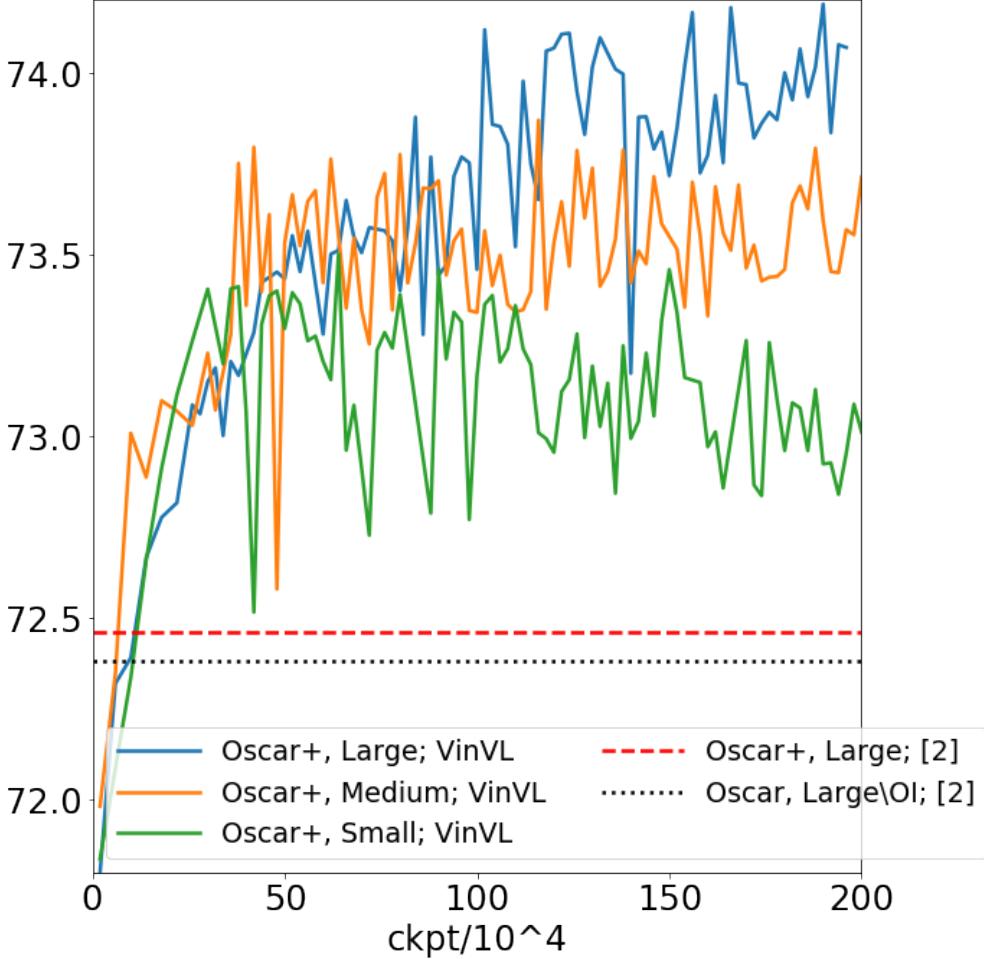


Figure 4: Effect of OSCAR+ pre-training corpus size and effect of self-training by making use of tagging data in OSCAR+. Each curve, with legend “VLP, Corpus; VisionFeature”, denotes a VLP experiment where the VLP method is either OSCAR or OSCAR+, the VLP pre-training Corpus is Small/Medium/Large (defined in Table 16), and VisionFeature is either our new vision features (VinVL for short) or those from [2] ([2] for short). X-axis denotes the pre-training iterations of OSCAR+ checkpoints. Y-axis is the vqa-dev accuracy of a VQA model initialized from the corresponding pre-training checkpoint and fine-tuned with a fixed scheme. Compared with “OSCAR+, Small; VinVL” (green), “OSCAR+, Medium; VinVL” (yellow) adds the 1.7M OpenImages Tagging data into the pre-training and its performance gets improved significantly, demonstrating the effect of self-training by making use of tagging data. The “OSCAR+, Large; VinVL” (blue) further scales up the pre-training corpus by adding Google Conceptual Captions and SBU datasets with generated tags and its performance gets further improved, demonstrating the effect of OSCAR+ pre-training corpus size. As baselines, we also provide performance of OSCAR and OSCAR+ with image features from [2], which clearly demonstrates that our new image features (VinVL) matter significantly in the VL pre-training and VL downstream tasks.

**Overall improvement from OSCAR to OSCAR+.** We point out that the improvement from OSCAR to OSCAR+ with image features from [2] is minor, because (1) we only add 1.7M OpenImages’ tagging data

to enlarge the pre-training corpus, which is a small portion compared with OSCAR’s original pre-training corpus (i.e., Large\OI, 3.98M images and 7.18M image-caption pairs), and (2) the new 3-way contrastive loss has more significant improvements in Text-Image Retrieval tasks instead of the VQA task, as illustrated in Table 3. We would expect much more significant improvements when we scale up the OSCAR+’s pre-training corpus to a much larger scale by adding large scale image tagging datasets, e.g., OpenImages (9M) and YFCC (92M).

## C Downstream Tasks Fine-tuning

We follow the downstream task fine-tuning recipes in OSCAR [21].

### C.1 VQA

Given an image and a question, the task is to select the correct answer from a multi-choice list, it requires the model to answer natural language questions based on an image. Here we conduct experiments on the widely-used VQA v2.0 dataset [8], which is built on the MSCOCO [25] images. Following [2], for each question, the model picks the corresponding answer from a shared set of 3,129 candidates.

When fine-tuning on the VQA task, the input sequence contains the concatenation of a given question, object tags and object region features, and then the [CLS] output from OSCAR+ is fed to a task-specific linear classifier for answer prediction. Similarly as the literature [2], we treat VQA as a multi-label classification problem – assigning a soft target score to each answer based on its relevancy to the human answer responses, and then we fine-tune the model by minimizing the cross-entropy loss computed using the predicted scores and the soft target scores. During inference, we simply use Softmax for answer prediction.

For VQA training, we random sample a set of 2k images from the MS COCO validation set as our validation set, the rest of images in the training and validation are used in the VQA fine-tuning. For the OSCAR+B model, we fine-tune for 25 epochs with a learning rate of  $5e^{-5}$  and a batch size of 128. For the OSCAR+L model, we fine-tune for 25 epochs with a learning rate of  $3e^{-5}$  and a batch size of 96.

### C.2 GQA

Similarly as VQA, GQA tests the reasoning capability of the model to answer a question. We conduct experiments on the public GQA dataset [13]. For each question, the model chooses an answer from a shared set of 1,852 candidates. Our fine-tuning procedure is following Oscar [21, 3], which first fine-tunes the model on unbalanced “all-split” for 5 epochs with a learning rate of  $5e^{-5}$  and a batch size of 128, and then fine-tuned on the “balanced-split” for 2 epochs.

### C.3 Image Captioning

An image captioning model generates a natural language description for a given image. To enable sentence generation, we fine-tune OSCAR+ using the seq2seq objective. The input samples are processed to triples consisting of image region features, captions, and object tags, in the same way as that during the pre-training. We randomly mask out 15% of the caption tokens and use the corresponding output representations to perform classification to predict the token ids. Similar to previous works [21, 45], the self-attention mask is constrained such that a caption token can only attend to the tokens before its position to simulate a unidirectional generation process. Note that all caption tokens will have full attentions to image regions and object tags but not the other way around.

During inference, we first encode the image regions, object tags, and a special token [CLS] as input. Then the model starts the generation by feeding in a [MASK] token and selecting a token from the vocabulary based on the likelihood output. Next, the [MASK] token in the previous input sequence is replaced with the selected token and a new [MASK] is appended for the next word prediction. The generation process terminates when the model outputs the [SEP] token. We use beam search (*i.e.*, beam size = 5) [2] in our experiments and report our results on the COCO image captioning dataset.

Though the training objective (*i.e.*, seq2seq) for image captioning is different from that used in pre-training (*i.e.*, bidirectional attention-based masked token loss), we directly fine-tune OSCAR+ for image captioning on COCO without additional pre-training on Conceptual Captions [32]. This is to validate the generalization ability of the OSCAR+ models for generation tasks. We use the same Karpathy split [15]. For the OSCAR+B model, we fine-tune with cross-entropy loss for 30 epochs with a batch size of 256 and an initial learning rate of  $1e^{-5}$  and then with CIDEr optimization [30] for 10 epochs with a batch size of 128 and initial learning rate of  $2e^{-6}$ . We compare with several existing methods, including BUTD [2], VLP [45], AoANet [10], OSCAR [21].

#### C.4 NoCaps

Novel Object Captioning [1] extends the image captioning task, is to test models’ capability of describing novel objects from the Open Images dataset [17] which are not seen in the training corpus. Following the restriction guideline of NoCaps, we train OSCAR+ on COCO without the initialization from pre-training, so no additional image-text pairs are used for training except COCO.

Since NoCaps images are collected from Open Images, we train an object detector using the Open Images training set and apply it to generate the tags. We conduct experiments from BERT model directly without pre-training as required by the task guidelines. For the OSCAR+B model, we train 30 epochs with a batch size of 256 and learning rate  $1e^{-4}$ ; further we perform CIDEr optimization with learning rate  $5e^{-6}$  and batch size 112 for 10 epochs. During inference, we use constrained beam search for decoding. We compare OSCAR+ with OSCAR [21] on this task.

#### C.5 Image-Text Retrieval

There are two sub-tasks: *image retrieval* and *text retrieval*, depending on which modality is used as the retrieved target. Both tasks calculate a similarity score between an image and a sentence, which heavily relies on the cross-modal representations.

Following Oscar [21], we formulate the retrieval as a binary classification problem, where given an aligned image-text pair, we randomly select a different image or a different sentence to form an unaligned pair. The final representation of [CLS] is used as the input to the classifier to predict whether the given pair is aligned or not. In the testing stage, the probability score is used to rank the given image-text pairs of a query.

Following [19], we report the top- $K$  retrieval results on both the 1K and 5K COCO test sets. We adopt the widely used Karpathy split [15] on the COCO caption dataset [25] to conduct our experiments. Specifically, the dataset consists of 113,287 images for training, 5,000 images for validation, and 5,000 images for testing. Each image is associated with 5 human-generated captions. For the OSCAR+B model, we fine-tune with a batch size of 256 for 40 epochs. The initial learning rate is set to  $2e^{-5}$  and linearly decreases. For the OSCAR+L model, we fine-tune with a batch size of 128 for 40 epochs. The initial learning rate is set to  $1e^{-5}$  and linearly decreases. We use the validation set for parameter tuning. We compare with

Mean -	64.02	64.49	64.57	65.48	66.49	66.75	
4Sets->VG	64.59	65.05	64.59	66.22	67.26	68.39	65.93
VG	64.91	65.95	65.24	66.68	68.17	68.13	66.40
VGw/oAttr	65.04	65.64	64.95	66.61	67.28	67.46	66.09
Grid-273	65.11	63.44	66.17	66.09	66.85	66.84	65.63
VG-obj	63.55	64.62	64.30	65.17	66.44	66.76	65.00
OI	63.41	64.14	63.31	64.15	65.38	65.38	64.21
Grid-50	62.55	63.54	64.02	63.92	64.72	64.76	63.78
							Mean

(O:500)      VG-obj (O:317)      ImageNet (O:100)      VG w/o Attr (O:1594)      VG (O:1594, A:524)      4Sets->VG (O:1594, A:524)      Mean

Figure 5: Overall comparison of vocabulary effect on VQA. X-axis: how the R50-C4 model is trained; Y-axis: how the feature is extracted (grid or region features, different kinds of boxes to extract region features). All region features have maximal 50 regions. The top row “Mean” is the average over all rows, showing the overall quality of different vision models. The far-right column “Mean” is the average over all columns, showing the overall quality of different feature extraction methods.

several existing methods, including DVSA [15], VSE++ [5], DPC [44], CAMP [39], SCAN [18], SCG [33], PFAN [38], Unicoder-VL [19], 12-in-1 [27], UNITER [4].

## C.6 NLVR2

Given a pair of images and a natural language, the goal of NLVR2 [35] is to determine whether the natural language statement is true about the image pair. For NLVR2 fine-tuning, we first construct two input sequences, each containing the concatenation of the given sentence (the natural language description) and one image, and then two [CLS] outputs from OSCAR+ are concatenated as the joint input for a binary classifier, implemented by an MLP.

For the OSCAR+B model, we fine-tune for 20 epochs with learning rate  $\{2e^{-5}, 3e^{-5}, 5e^{-5}\}$  and a batch size of 72. For the OSCAR+L model, we fine-tune for 20 epochs with learning rate of  $\{2e^{-5}, 3e^{-5}\}$  and a batch size of 48.

## D More on the Effect of Object-Attribute Vocabulary Size: disentangling the effects of region proposals and model weights

In Section 5.2, we demonstrate that the more diverse the visual concepts (object and attribute vocabularies) are, the better the visual region features for VL tasks. The better performance may come from the more diverse proposed regions where the region features are extracted (see the comparison in Figure 1, “region” for short), or from the better model weights that can produce better high-dimensional region representation even for the same region (“model” for short). In this section, we disentangle effects of region proposals and model weights, by performing synthetic experiments in which we use region proposals from one vision model and model weights from another vision model. Our results show that both the region proposals and

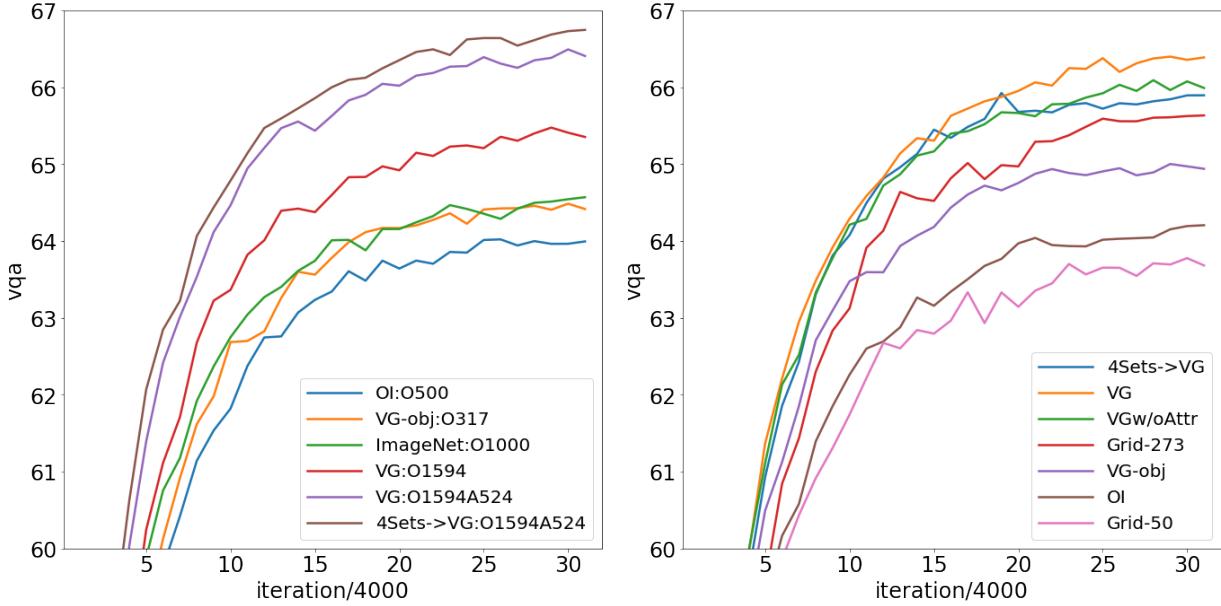


Figure 6: Left: comparison of object vocab and attribute vocab, average over all types of bounding boxes. Right: comparison of feature extraction methods, average over all types of pre-trained vision models. X-axis is the number of iterations when we take the checkpoint for evaluation. Y-axis is the VQA accuracy on our vqa-dev.

model weights matter for VL tasks.

### D.1 Disentangling the effects of region proposals and model weights on R50-C4

As in Section 5.2, We train vision models  $v = \text{Vision}(Img)$  on different datasets, i.e., OpenImages with 500 object classes (OI:O500), standard ImageNet with 1K classes (ImageNet:O1000), Visual Genome with 317 object classes (VG-obj), Visual Genome with 1594 object classes (VG:O1594), VG with 1594 object classes and 524 attribute classes (VG:O1594A524), pretrain on the merged 4 datasets and finetune on VG:O1594A524 (4Sets→VG:O1594A524). For each model, we also try different ways to extract features: (1) *region* features from different models’ proposed regions (same notations with models) where each image has maximal 50 region features, and (2) *grid* features where we use all grid features (Grid-273) or randomly sampled 50 grid features (Grid-50) for each image. We present the results of these model-region cross-combination experiments in Figure 5. We also present the mean accuracy over all box types to obtain a robust ranking of different checkpoints and the mean accuracy over all checkpoints to obtain a robust ranking of different box types. We have the following observations:

- The richer the object vocabulary is, the better for VQA: OI:500 ≈ VG-obj:O317 < ImageNet:O1000 < VG:O1594.
- Attribute information is crucial to VL tasks: all features trained with attributes (Columns with VG:O1594A524) are significantly better than those without attributes.
- Even for small vision backbone R50, vision pre-training makes vision features better: Column “4Sets→VG:O1594A524” are better than all other columns. Notice that the vision pre-training improves both the region features and the grid features.

- It is crucial to extract features from semantically diverse regions: regions from OI and VG-obj are significantly worse than all other regions, and is even worse than grid features.
- Grid features perform worse than region features with regions proposed by VG models. By comparing Row “Grid-273” with rows with VG regions, it seems hopeful to close this gap while paying more hardware memory and computational cost in cross-modal models VL. It is three times slower to **train** the “Grid-273” models than training models with region features.

In Figure 6, instead of just showing one final number, we provide the *mean evaluation curves along training trajectories* to demonstrate the ranking, as an even more robust evidence. These results further confirm the conclusions we draw in Section 5.2.

## D.2 Disentangling the effects of region proposals and model weights on the SoTA model

In Table 17, we alternate the combination of region proposals and model weights, and evaluate them on VQA. As we can see, the improvement of using boxes from the R101-C4 model [2] to extract features from our X152-C4 model is much bigger than that of using boxes from our X152-C4 model to extract features from the R101-C4 model [2], indicating pre-trained model weights are more important than regions. Inspired by this analysis, we propose the class-agnostic NMS for region selection in the box head of the OD model, which does not sacrifice any VQA performance but greatly improves the model’s inference speed. This analysis also suggests that large-scale OD pre-training should improve performance for grid-feature based VL models, as supported by more results in Appendix F.

region \ model	R101-C4 [2]	VinVL
R101-C4 [2]	68.52 ± 0.11	70.25 ± 0.05
VinVL	69.05 ± 0.06	71.34 ± 0.17

Table 17: Ablation of region and model on VQA.

## E More on FPN and Comparison of C4 and FPN

### E.1 Two reasons why FPN performs worse than C4 on VL tasks.

Our experimental results confirm the conclusion of [14] that the FPN model does not provide better region features for VL tasks than the C4 model (Columns “R50C4” vs. “R50FPN” in Table 18). Our analysis reveals two reasons. First of all, all layers involved in feature extraction in the C4 model have been pre-trained using ImageNet while the MLP head of FPN does not. It turns out that the VG dataset is still small to train a good visual features for VL tasks and using ImageNet-pre-trained weights is beneficial. This can be verified by two experiments: (1) When the R50-C4 model is trained on VG with its box head randomly initialized (VG-trained - R50C4 w/ box head randomly initialized), the C4 model’s performance is the same as FPN; and (2) C4 and FPN achieve the same performance after vision pre-training on 4 datasets (68.3 vs. 68.2). The second reason is due the network architecture (CNN vs. MLP) of the box head in the OD model. The convolutional head in C4 has a better inductive bias in encoding visual information than the MLP head in FPN. This can be verified by the fact that when vision features from randomly initialized models are used (Row “Initial” in Table 18), R50-C4 performs much better than R50-FPN, indicating that the initial C4 features encode much more useful visual information than the initial FPN features. The “random” C4 features nearly match the feature from ImageNet pre-trained model (Row “Initial” Column “R50C4”), while

	no image feature $w$	R50-C4 w/ box head randomly initialized	R50-FPN	R50-C4	4Sets→R50-FPN	4Sets→R50-C4
VG-trained Initial	–	67.6 ±0.13	67.6±0.30	68.0±0.16	68.3±0.11	68.2±0.05
	55.5±0.50	61.8 ±0.47	57.6±0.16	64.8±0.44	66.1±0.23	66.8±0.21

Table 18: C4 vs FPN architecture on VQA. Boxes used to extract features  $v$  and tags  $q$  used in VL model are the same with those used in OSCAR [21]. Row “Initial” means using the initialization model without VG training for feature extraction.

“random” FPN features are close to the performance without visual features as input (Row “Initial” Column “no image feature  $w$ ”).

## E.2 Effect of pooling methods in FPN on VQA performance.

Different from C4 models that extract region features from a single scale (the end of C4 block), FPN models extract region features from multiple scales adaptively based on the area of the region. Therefore, there is some in-homogeneity in FPN’s region features since they may come from different scales. In Figure 7, we show that this is not the cause of FPN’s worse performance than C4 on the VQA task. More specifically, we experiment with 4 pooling methods for FPN architecture. (1) adapt: the original FPN’s pooling method that extract features adaptively from different scales; (2) max: extract features from all scales and then do a max-pool; (3) avg: extract features from all scales and then do an average-pool; (4) concat: extract features from all scales and then concatenate them together. We also train multiple FPN models on VG with these pooling methods, with or without pre-training on the Objects365 dataset. We experiment on all possible combinations (in total  $8 \times 4$ ) of 8 vision models and 4 pooling methods on the VQA task. When there is a parameter dimension mis-match, e.g., non-concat FPN models but use concat pooling methods in VQA and vice versa, we specify those parameter randomly with PyTorch’s default initialization method. The results in Figure 7 shows that (1) there is no obvious difference in different pooling methods, with the default “adapt” and the “concat” methods perform slightly better than “max” and “avg”; (2) (without surprise) the performance is significantly worse when there is a parameter dimension mismatch between vision models and VL task feature extraction methods, i.e., non-concat FPN models but use concat pooling methods in VQA and vice versa. These results show that the pooling method (no matter in vision model training or in VL task feature extraction) is not the root cause of FPN’s worse performance than C4 on the VQA task.

## E.3 Large-scale object-detection pre-training of C4 and FPN models

In this paper, we have trained R50-C4, R50-FPN, R152-C4 and R152-FPN models on the merged object detection datasets described in Table 2. In Figure 8, we report the  $mAP^{50}$  of checkpoints from these 4 experiments on 4 validation sets: COCO with stuff (top left), Objects365 (top right), OpenImages (bottom left) and Visual Genome (1594 object classes, bottom right). For R50 models, the R50-FPN model is slightly better than C4 on COCO and Objects365 but slightly worse than C4 on Visual Genome. For R152 models, the R152-FPN model is consistently worse than the R152-C4 model on all 4 different datasets. Therefore, we finally use the R152-C4 model for downstream vision-language tasks.

	max	avg	adapt	concat		max	avg	adapt	concat
o365->vg-max	65.96	67.00	65.20	62.68		67.12	66.27	65.71	63.63
o365->vg-avg	66.58	66.96	66.88	61.03		66.55	66.62	66.15	63.62
o365->vg-adapt	67.10	66.63	67.43	63.96		66.57	67.01	67.60	63.52
o365->vg-concat	61.14	62.17	59.92	66.90		60.98	62.70	60.25	66.87
vg-max	66.64	66.76	64.55	64.22		66.94	65.76	64.64	62.89
vg-avg	66.14	66.88	67.06	62.60		67.00	67.06	66.74	63.77
vg-adapt	66.10	66.74	66.81	63.71		66.76	66.38	66.29	63.77
vg-concat	61.03	64.34	60.64	67.19		61.71	62.64	59.91	67.18

Figure 7: Pooling methods in FPN feature extraction are not the root cause of FPN’s worse performance than C4. X-axis: the pooling method when extracting features for VL tasks; Y-axis: the pooling method (vision model) when pre-training the visual feature extraction model. All experiments are using regions from the Bottom-up Top-down model [2]. Each combination is experimented twice with two random seeds, i.e. seed=42 on the left and seed=88 on the right. The results from two random seeds are consistent.

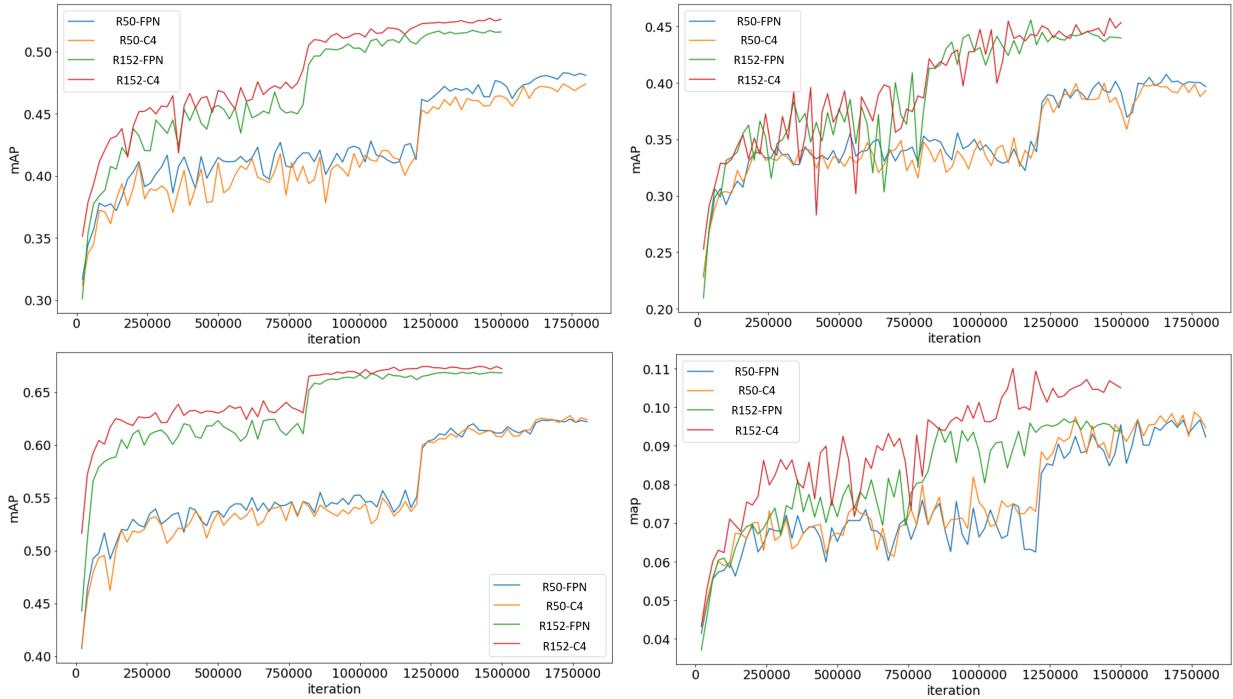


Figure 8: Checkpoints’  $mAP^{50}$  on 4 validation sets: COCO with stuff (top left), Objects365 (top right), OpenImages (bottom left) and Visual Genome (1594 object classes, bottom right). For R50 models, the R50-FPN model is slightly better than C4 on COCO and Objects365 but slightly worse than C4 on Visual Genome. For R152 models, the R152-FPN model is consistently worse than the R152-C4 model on all 4 different datasets.

	ImageNet-5k [40]	4Sets	VG with Attr	4Sets→VG
grid feature (273)	68.3±0.29	65.2±2.47	67.5±0.20	69.4*
region feature (50)	67.7±0.16	68.5±0.13	69.8±0.23	70.6±0.13

\* The other run failed and thus there is no std for this experiment.

Table 19: Ablation study of X152 models on VQA. Vision models in the last three columns are trained with initialization from the ImageNet-5k checkpoint in the first column. All the region features are extracted with boxes proposed by our best X152-C4 model (pre-trained on 4Sets and fine-tuned on VG). By comparing the first column and the last column, we see that our proposed vision pre-training (first on 4 sets and then on VG with attributes) improves performance for both the grid-feature based model and the region-feature based model. Since the X152 backbone is much larger than the R50 backbone in Figure 5, the larger model can make better use of the large pre-training datasets and thus have more significant improvements.

## F Grid feature

In Table 19, we train grid-feature based and region-feature based X152 models for VQA, with the vision models pre-trained on different vision datasets, i.e., “ImageNet-5k” from [40], our 4-dataset merged OD dataset 2 (4Sets), our VG dataset with 1594 object classes and 524 attribute classes (VG with Attr), and first 4Sets and then VG (4Sets→VG). Vision models in the last three cases are trained with initialization from the same ImageNet-5k checkpoint from [40]. All the region features are extracted with boxes proposed by our best X152-C4 model (pre-trained on 4Sets and fine-tuned on VG). By comparing “ImageNet-5k” and “4Sets→VG”, we see that our proposed vision pre-training improves performance for both the grid-feature based model and the region-feature based model. Since the X152 backbone is much larger than the R50 backbone in Figure 5, the larger model makes better use of the large pre-training datasets and thus has more significant improvements. It is interesting to see that for grid-feature based models, the “ImageNet-5k” model performs better than the “4Sets” model and the “VG with Attr”, while it is not the case for region-feature based models. This may indicate that how the vision model is trained (grid-feature wise or region-feature wise) may have big impact on the downstream VL tasks.

## G End-to-end inference efficiency

We report the end-to-end inference time of different VQA models on a Titan-X GPU and a Xeon E5 CPU in Table 20. For CPU evaluation, we force that the inference use only one CPU thread. The input image size is  $800 \times 1333$ , and we run the inference with batch size 1 (one image-question pair per batch). We can see that (1) vision models dominate the inference time, especially for large models; (2) models based on grid-feature are faster than those based on region feature; (3) with our proposed fast inference trick, region-feature models are greatly sped up and their inference time can be brought to within 3 times of that of grid-feature models on GPU. We find that on CPU with a single thread, our class-agnostic trick does not lead to time saving, because nearly all inference time is taken by the backbone and C4 head and the time from NMS operations is nearly ignorable on CPU.

Model	R50-C4		R101-C4 [2]		X152-C4	
	Vision	VL	Vision	VL	Vision	VL
Grid-50	0.059 $\pm$ 0.018	0.029 $\pm$ 0.002	0.083 $\pm$ 0.025	0.030 $\pm$ 0.003	0.355 $\pm$ 0.022	0.031 $\pm$ 0.003
Grid-273	0.056 $\pm$ 0.005	0.027 $\pm$ 0.002	0.082 $\pm$ 0.022	0.034 $\pm$ 0.001	0.344 $\pm$ 0.036	0.037 $\pm$ 0.004
Object	0.373 $\pm$ 0.040	0.031 $\pm$ 0.005	0.663 $\pm$ 0.042	0.034 $\pm$ 0.003	0.687 $\pm$ 0.064	0.036 $\pm$ 0.005
Object-eff	0.165 $\pm$ 0.029	0.029 $\pm$ 0.002	0.442 $\pm$ 0.119	0.036 $\pm$ 0.003	0.475 $\pm$ 0.049	0.037 $\pm$ 0.005
Grid-50 (cpu)	1.943 $\pm$ 0.244	0.480 $\pm$ 0.042	4.050 $\pm$ 0.398	0.469 $\pm$ 0.046	17.765 $\pm$ 1.693	0.501 $\pm$ 0.047
Grid-273 (cpu)	2.032 $\pm$ 0.230	1.368 $\pm$ 0.056	4.052 $\pm$ 0.372	1.283 $\pm$ 0.067	17.664 $\pm$ 1.713	1.326 $\pm$ 0.053
Object (cpu)	11.808 $\pm$ 1.322	0.500 $\pm$ 0.045	31.863 $\pm$ 7.932	0.585 $\pm$ 0.044	29.641 $\pm$ 3.097	0.565 $\pm$ 0.044
Object-eff (cpu)	11.729 $\pm$ 1.280	0.510 $\pm$ 0.044	31.791 $\pm$ 8.027	0.587 $\pm$ 0.043	29.687 $\pm$ 3.011	0.574 $\pm$ 0.036

Table 20: Time cost of end-to-end inference on VQA. All cross-modal models are BERT-Base. On the SOTA number obtained with X152-C4 region features, the performance *keeps the same* when changing to the efficient way to extract the feature while the efficiency greatly improves on GPU. The efficient version does not lead to time saving on CPU, because nearly all inference time is taken by the backbone and C4 head and the time from NMS operations is nearly ignorable on CPU.