A. Pretraining details

We employed the AdamW optimizer [31] with a weight decay of 0.02 for training our models. The learning rate was warmed up to 1e-5 (ViT-B/16) and 1e-4 (BERT$_{base}$) in the first 1000 iterations, then decayed to 1e-6 according to a cosine schedule. The pre-training of BUS required approximately 60 hours and was performed on 8 A100-80G GPUs using a 4M pre-training dataset for about 20 epochs.

To improve the generalization of vision encoders during pre-training, we applied RandAugment [9] to random image crops of size 256 × 256. In the fine-tuning stage for VQA, image captioning, and visual grounding tasks, we increased the image resolution. For image-text contrastive learning, we set the queue size to 65,536 and the momentum coefficient to 0.995.

A.1. Pretraining data

	COCO	VG	SBU	CC3M
image	113K	100K	860K	3M
text	567K	769K	860K	3M

Table 8: Statistics of the pre-training datasets.

	image	Captions	Objects	Regions
COCO	0.11M	0.55M	0.45M	-
VG	0.10M	-	2.0M	3.7M

Table 9: Statistics of objects/regions annotations used in the pre-training.

Table 8 shows the statistics of the 4M images with texts used in the pre-training stage. Additionally, we use object/region annotations from the COCO [30] and VG [21] datasets, as shown in Table 9, and provide statistics of object and region annotations for each dataset. We follow the object/region annotations provided by [56], which filter out some samples due to: 1) invalid annotations (e.g., negative values for bounding boxes or boxes being outside of the images); 2) boxes being too small ($\leq$ 1%); 3) highly overlapped textual descriptions of regions ($\geq$ 75%), etc. After pre-processing, we keep 446,873 COCO objects (from 859,999), 2,043,927 VG objects (from 3,802,349), and 3,699,598 VG regions (from 5,402,953).

A.2. Pretraining Task

We pre-train our model with five standard objectives: Image-Text Contrastive learning (ITC), Image-Text Matching (ITM), and Masked Language Modeling (MLM),Prefix Language Modeling (PrefixLM), Patch-Text Matching (PTM). These pre-training tasks are optimized jointly. In this subsection, we will firstly introduce the last four pre-training task and then give the details of the Patch-Text Matching .

Image-text Contrastive (ITC) For BUS , We follow the [26] and apply ITC to align the image representation and text representation from the unimodal encoders. For the image, the image feature corresponding to the image [CLS] token is chosen as the image representation. For the text, the text token feature corresponding to the text [CLS] token is the text representation.

Image-Text Matching (ITM) The goal of image-text matching is to predict whether the input image and text are matched. We follow the design of [26] and select hard negative image-text pairs based on the contrastive text-image similarity. We take the text [CLS] embedding of the multimodal encoder's output as the joint representation, followed by a Multi-Layer Perceptron (MLP) layer for prediction.

Masked Language Modeling (MLM) The task setup is basically the same as in BERT [10], where we randomly mask 15% of tokens in text and the model is asked to predict these masked words with the cross-modal representations.

Prefix Language Modeling (PrefixLM). This task aims to generate the caption given an image and predict the text segment subsequent to the cross-modal context as [5]. It optimizes a cross entropy loss by maximizing the likelihood of text in an autoregressive manner.

A.2.1 Patch-Text Matching

The key component for the bottom-up patch summarization is the Text Semantic-aware Patch Selector (TSPS) which needs to predict the fine-grained alignment scores between the image patches and input text to select the text-relevant patches. However, such fine-grained patch-text alignment capabilities of traditional ViT-based models are weak as the lack of fine-grained patch-text labels. To address the above difficulties, we introduce a novel pre-training task named Patch Text Matching (PTM) which facilitates the patch detector training and drives our model to learn the fine-grained patch-text alignment.

In most object objection and visual grounding datasets, objects and regions are typically paired with a class label or text description. Therefore, for each (object/region) bounding box in an image, we can obtain a corresponding text description (For the object class label, we can transfer it to a text description using a text template such as "this is a [Class Label]"). We then transform the bounding box annotations into patch-level labels by assigning a label of 1 to an image patch if it overlaps with the bounding box and 0 otherwise. Different text descriptions and bounding boxes result in different patch labels, enabling us to generate fine-grained patch-text labels that serve as supervisory signals for pre-training our model.

During pre-training, we randomly sample a mini-batch of images from object detection/visual grounding datasets such as COCO [30] or VG [21]. For each image, we randomly select an object/region bounding box and translate the bounding box annotation to the image patch label sequence following the aforementioned transformation rule. We then feed the batch of text descriptions of the bounding boxes and the images to BUS. We expect the TSPS to predict all patches that overlap with the bounding box with the guidance of the bounding box text description.

Once TSPS has predicted the alignment scores between image patches and text, we calculate the binary cross-entropy loss between the alignment scores and patch labels using the following equation:

$$\mathbf{L}_{PTM} = \frac{1}{n}\sum_{i=1}^{n} Y_i log\left(a_i\right) + \left(1 - Y_i\right) log\left(1 - a_i\right) \quad (1)$$

Here, a_i is the alignment score between the i^{th} patch in the image and the input text, and Y_i is the patch label of the i^{th} patch. Besides, at the beginning of pre-training, as the PTM loss has not yet converged, thus the performance of the patch selector is not ideal, we select the image patches directly based on the attention weights of the image [CLS] token to other patch tokens by setting the hyper-parameter β to 0. As the PTM loss gradually converges, we will progressively set a large value to β.

After calculating the PTM loss $\mathbf{L}_{PTM}$, we then randomly sample a mini-batch of normal image-text pairs from the dataset of 4M images and calculate the Image-Text Contrastive (ITC) loss $\mathbf{L}_{ITC}$, Image-Text Matching (ITM) loss $\mathbf{L}_{ITM}$, Masked Language Modeling (MLM) loss $\mathbf{L}_{MLM}$ and Prefix Language Modeling (PrefixLM) loss $\mathbf{L}_{Prefix}$ based on other four pre-training objectives. We assign equal loss weights to each pre-training loss, and thus the full pre-training loss is:

$$\mathbf{L} = \mathbf{L}_{ITC} + \mathbf{L}_{ITM} + \mathbf{L}_{MLM} + \mathbf{L}_{Prefix} + \mathbf{L}_{PTM} \quad (2)$$

A.3. Pretraining Schedule

In this subsection, as shown in Algorithm 1, we give a algorithm of the pretraining schedule of our model BUS .

B. Downstream Task Details

We evaluate BUS on the four downstream vision-language tasks. The hyperparameters that we use for fine-tuning on the downstream tasks are listed in Table 10. Following [26], all tasks adopt RandAugment, AdamW optimizer with a weight decay of 0.05 and a cosine learning rate schedule. Next we introduce the dataset settings in detail.

```
Algorithm 1: Pre-training of BUS
Input: Large scale pretraining dataset $\mathcal{D}$,
       Object/Region Dataset $\mathcal{O}$, the number of
       pre-training epochs $T$, the pre-training
       learning rate $\alpha$, the batch size $B_D$ of dataset
       $\mathcal{D}$, the batch size $B_O$ of dataset $\mathcal{O}$.
Initialize the parameters $\theta$ of our model $M$ ;
for $t = 1$ to $T$ do
    Randomly sample a mini-batch of $B_O$ Images
     $\{\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_{B_O}\}$ from $\mathcal{D}$ ;
    for $i = 1$ to $B_O$ do
        Select a object or region $r_i$ from image $\hat{v}_i$ ;
        Translate the object class label $\hat{y}_i$ to text
         description $\hat{t}_i$;
        Translate the bounding box annotation of $r_i$
         to patch annotations $Y^i = \{y^i_1, y^i_2, \ldots, y^i_n\}$
         ;
    Run forward of $M$ on the mini-batch of
     image-text pairs
     $\{\{\hat{v}_1, \hat{t}_1\}, \{\hat{v}_2, \hat{t}_2\}, \ldots, \{\hat{v}_{B_O}, \hat{t}_{B_O}\}\}$ and
     $\{Y^1, Y^2, \ldots, Y^{B_O}\}$ to obtain the loss $\mathcal{L}_{PTM}$ ;
    Randomly sample a mini-batch of $B$ Image-Text
     Pairs $\{\{v_1, t_1\}, \{v_2, t_3\}, \ldots, \{v_{B_D}, t_{B_D}\}\}$
     from $\mathcal{D}$ ;
    Run forward of $M$ on the mini-batch of
     image-text pairs
     $\{\{v_1, t_1\}, \{v_2, t_3\}, \ldots, \{v_{B_D}, t_{B_D}\}\}$ to obtain
     the losses $\mathcal{L}_{ITC}, \mathcal{L}_{ITM}, \mathcal{L}_{MLM}, \mathcal{L}_{Prefix}$ ;
    Calculate the overall loss:
    $\mathbf{L} = \mathbf{L}_{ITC} + \mathbf{L}_{ITM} + \mathbf{L}_{MLM} + \mathbf{L}_{Prefix} + \mathbf{L}_{PTM}$;
    Backward the overall loss $\mathbf{L}$ and update the
     parameters of $M$ using gradient descent with
     learning rate $\alpha$ and the average loss $\mathbf{L}$ over the
     mini-batch:
    $\theta \leftarrow \theta - \alpha \frac{1}{B} \sum_{i=1}^{B} \nabla_\theta \mathcal{L}(\theta; s_i)$ ;
return $M$ with pre-trained parameters $\theta$ ;
```

VQA. The VQA task [1] requires the model to answer natural language questions given an image. Most methods [45, 48, 28, 49] deal with visual question answering tasks as multi-label classification on pre-defined answer sets. This strategy achieves strong performance, but it is not suitable for real-world open scenarios. We conduct experiment on the VQA2.0 dataset [13], which contains 83k/41k/81k images for training/validation/test. Following [26], we use both training and validation splits for training, and incorporate additional training data from Visual Genome [21]. Following [28], we concatenate the question with the object labels and OCR tokens extracted from image.

Task	LR (ViT-L/BERT$_{base}$)	batch size	epochs
VQA	2e-5/5e-6	1024	8
Captioning†	1e-5&8e-7	256	5
Retrieval	1e-5/2e-6	256	5
Visual Grounding	2e-5/2e-6	512	120

Table 10: Finetuning hyperparameters for downstream tasks. † denotes two stages fine-tuning.

Image Captioning. Image captioning requires generating a descriptive and fluent caption for a given image. We evaluate the performance of BUS on two popular datasets: COCO Caption [30] and NoCaps [2]. We fine-tune BUS on the training set of COCO Caption and test it on the same Karpathy split [28, 49] as well as the NoCaps validation set. To fine-tune BUS on COCO Caption, we follow the approach in [28] and first train the model with cross-entropy loss for 5 epochs with a learning rate of 1e-5 and a batch size of 256. We then further fine-tune the model with CIDEr optimization [40] for an additional 5 epochs with a smaller learning rate of 8e-7. We use the best checkpoint on COCO Caption to predict on the NoCaps validation set. During inference, we use beam search with a beam size of 10 and set the maximum generation length to 20.

Image-Text Retrieval. We conducted experiments on both image-to-text retrieval (TR) and text-to-image retrieval (IR) using the COCO [30] and Flickr30K [37] datasets and used the widely-used Karpathy split [19] for both. COCO contains 113k/5k/5k images for train/validation/test, while Flickr30K contains 29k/1k/1k images for train/validation/test. During fine-tuning, we jointly optimized the ITC loss and the ITM loss following the approach in [26, 25]. During inference, we first selected the top-k candidates by computing the dot-product similarity between the image and text encoder features (We set $k = 256$ for COCO and $k = 128$ for Flickr30K). For efficiency of coarse-grained ranking, we directly set β to 0 and selected the patch based on the attention weights of the image [CLS] token to other patch tokens. During the fine-grained reranking for the top-k candidates, we set β to 0.8 and reranked the candidates based on their ITM scores.

Visual Grounding. The task of visual grounding involves localizing the referred object in an image given a plain text query. Instead of directly regressing bounding boxes, our approach concatenates visual features with textual features, which are then fed into the multi-modal decoder to predict the object's coordinates. We evaluate our method on the referring expression grounding dataset: RefCOCO+[55]. The RefCOCO+ dataset contains 19K images and 141K queries.

C. Comparison Models

- **E2E-VLP** [51]: proposes the first end-to-end VLP method for both V+L understanding and generation, with a unified Transformer encoder-decoder architecture.
- **VinVL** [58]: pre-trains a large-scale object-attribute detection model with much larger amounts of supervised data on four public object detection datasets for extracting better region-based visual feature.
- **OSCAR** [28]: proposes to use object tags detected in images as anchor points to ease the learning of cross-modal alignments, where the input to the Transformer is a combination of image, text and object tags.
- **METER** [11]: systematically investigates how to design and pre-train a fully transformer-based VL model in an end-to-end manner.
- **VLMo** [48]: presents a unified vision-language pre-trained model that jointly learns a dual encoder and a fusion encoder with a modular Transformer network.
- **SimVLM** [49]: different from previous VLP methods that only use limited (4M-10M) image-text pairs for pre-training, it proposes a simple VLP model with a single prefix language modeling objective, which pre-trains on a extremely large aligned cross-modal data of about 1.8B noisy image-text pairs. This is also a latest state-of-the-art method on image captioning.
- **ALBEF** [26]: introduces a contrastive loss to align the image and text representations before fusing them through cross-modal attention, which enables more grounded vision and language representation learning.
- **UNITER** [7]: proposes an improved single-stream VLP method, by designing two new pre-training strategies: 1) it uses conditional masking on pre-training tasks instead of random masking strategy, 2) it designs a new word-region alignment pre-training task via the use of optimal transport to explicitly encourage fine-grained alignment between words and image regions.
- **ALIGN** [15]: leverages a noisy dataset of over one billion image alt-text pairs, obtained without expensive filtering or post-processing steps in the Conceptual Captions dataset.
- **VLBERT** [43]: is a pioneering work to pre-train a single-stream multi-modal Transformer, which jointly trains both the Transformer-based cross-modal fusion and Fast R-CNN image feature extractor in both pre-training and fine-tuning phases. It is widely used as a baseline method for VLP models.

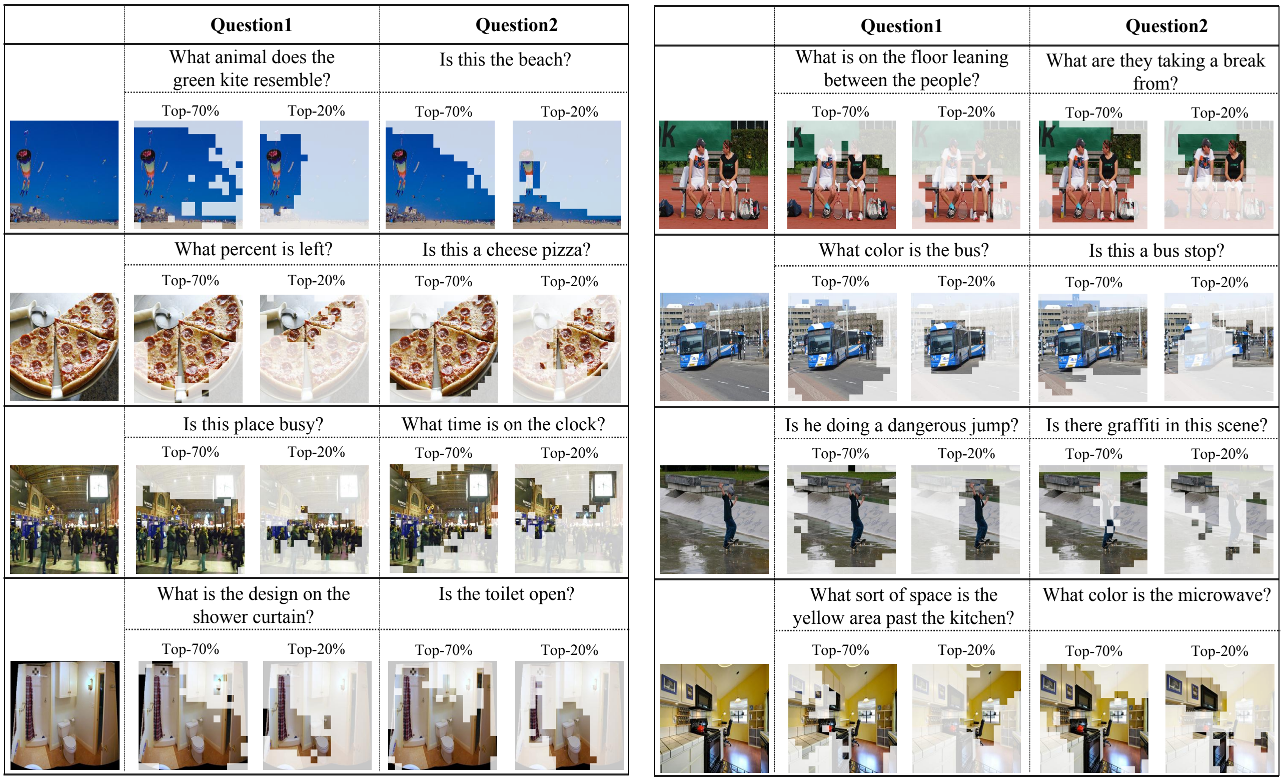
Question1
Question2
What animal does the green kite resemble?
Is this the beach?
What percent is left?
Is this a cheese pizza?
Is this place busy?
What time is on the clock?
What is the design on the shower curtain?
Is the toilet open?
What is on the floor leaning between the people?
What are they taking a break from?
What color is the bus?
Is this a bus stop?
Is he doing a dangerous jump?
Is there graffiti in this scene?
What sort of space is the yellow area past the kitchen?
What color is the microwave?
Top-70%
Top-20%

Figure 6: The visualization of the VQA cases and the selected text-relevant image patches.

VILT [20]: adopts linear projection and word embedding as the visual and textual encoders, and uses the visual transformer as the cross-modal encoder to align and fuse the features of both modalities in an end-to-end manner.

- **VILLA** [12]: is the first known effort on large-scale adversarial training for vision-and-language (V+L) representation learning.
- **XVLM** [56]: proposes to learn multi-grained alignments which locates visual concepts in the image given the associated texts, and in the meantime align the texts with the visual concepts.
- **BLIP** [25]: proposes a new VLP framework which transfers flexibly to both vision-language understanding and generation tasks. It effectively utilizes the noisy web data by bootstrapping the captions.
- **UNICORN** [53]: proposes a vision-language (VL) model that unifies text generation and bounding box prediction into a single architecture.
- **LXMERT** [45]: is the pioneering work to pre-train a two-stream multi-modal Transformer, which consists of an object relationship encoder, a language encoder and a cross-modality encoder. It is widely used as a baseline method for VLP models.
- **ViLBERT** [32]: proposes one of the first work that extend the BERT architecture to a multi-modal two-stream VLP model, which processes both visual and textual inputs in separate streams that interact through co-attentional transformer layers.
- **mPLUG** [23]: is a vision-language foundation model for both cross-modal understanding and generation and introduces an effective and efficient vision-language architecture with novel cross-modal skip-connections.
- **TRIPS** [16]: is a vision-and-language pre-training model which reduces the visual sequence progressively with a patch-selection layer in the visual backbone for efficient training and inference.

D. Case Study

In this subsection, we visualize more VQA cases and the selected text-relevant image patches in Figure 6. Note that these two examples are not cherry-picked. The phenomenon in these examples is commonly observed among other samples.