BUS : Efficient and Effective Vision-language Pre-training with Bottom-Up Patch Summarization.

Chaoya Jiang[1], Haiyang Xu[2]*, Wei Ye[1]*, Qinghao Ye[2], Chenliang Li[2], Ming Yan[2], Bin Bi[2]
Shikun Zhang[1], Fei Huang[2], Songfang Huang[2]
[1]National Engineering Research Center for Software Engineering, Peking University
[2]DAMO Academy, Alibaba Group
shuofeng.xhy@alibaba-inc.com, wye@pku.edu.cn

Abstract

*Vision Transformer (ViT) based Vision-Language Pre-training (VLP) models have demonstrated impressive performance in various tasks. However, the lengthy visual token sequences fed into ViT can lead to training inefficiency and ineffectiveness. Existing efforts address the challenge by either bottom-level patch extraction in the ViT backbone or top-level patch abstraction outside, not balancing training efficiency and effectiveness well. Inspired by text summarization in natural language processing, we propose a **B**ottom-**U**p Patch **S**ummarization approach named BUS , coordinating bottom-level extraction and top-level abstraction to learn a concise summary of lengthy visual token sequences efficiently. Specifically, We incorporate a Text-Semantics-Aware Patch Selector (TSPS) into the ViT backbone to perform a coarse-grained visual token extraction and then attach a flexible Transformer-based Patch Abstraction Decoder (PAD) upon the backbone for top-level visual abstraction. This bottom-up collaboration enables our BUS to yield high training efficiency while maintaining or even improving effectiveness. We evaluate our approach on various visual-language understanding and generation tasks and show competitive downstream task performance while boosting the training efficiency by 50%. Additionally, our model achieves state-of-the-art performance on many downstream tasks by increasing input image resolution without increasing computational costs over baselines.*

1. Introduction

Large-scale pre-training of vision-language models has recently received tremendous success on a wide range of cross-modal tasks [45, 7, 14, 28, 54, 26, 49]. Such vision-language models learn cross-modal representations from a large number of image-text pairs by aligning the visual and linguistic modalities.

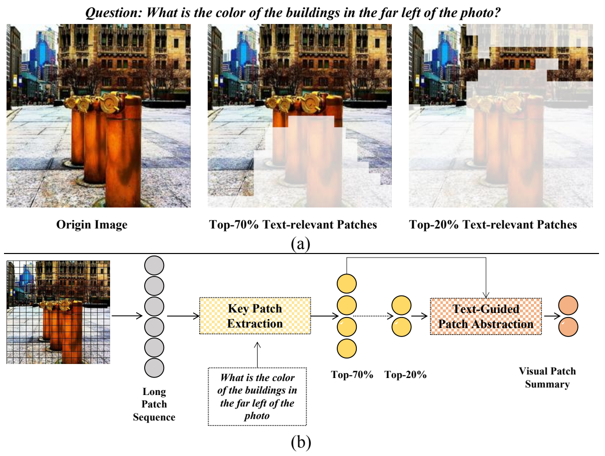

Question: What is the color of the buildings in the far left of the photo?
Origin Image
Top-70% Text-relevant Patches
Top-20% Text-relevant Patches
(a)
Key Patch Extraction
What is the color of the buildings in the far left of the photo
Long Patch Sequence
Top-70%
Top-20%
Text-Guided Patch Abstraction
Visual Patch Summary
(b)

Figure 1: Subfigure (a) shows the examples of selected text-relevant patch in the VQA scenario. Subfigure (b) shows the overview of our proposed bottom-up patch summarization.

Most recent works [14, 49, 26, 20, 11] adopt ViT as the visual encoder or cross-modal fusion encoder, due to its excellent ability to model the fine-grained long visual sequences from the image grids or patches.

Despite the impressive progress of ViT-based VLP models, they still face challenges of training inefficiency and ineffectiveness caused by lengthy visual token sequences. Firstly, long visual sequences will bring heavy self-attention calculation for visual representation modeling and cross-modal fusion, leading to time-consuming training. Secondly, long visual sequences contain many redundant patches irrelevant to the text semantics. For instance, as illustrated in Figure 1 (a), during the VQA task, when answering the question "What is the color of the buildings in the far left of the photo?", about 80% of the image patches may be irrelevant with the question. On the one hand, those

[*] Corresponding Author.

text-irrelevant patches (e.g., the yellow building in the image) will hinder the fine-grained alignment between the textual and visual modalities. On the other hand, they will lead to the overshadowing of brief linguistic signals (e.g., of short image captions) by complex visual ones during the cross-modal fusion, namely the "vanishing information" problem of textual information [23].

The limitations above underscore the importance of reducing visual token sequences. Recent related efforts can be categorized into two lines.

- **Top-level Abstraction**. The first line tackles the issue outside the ViT-based visual backbones from a top-level perspective [24, 3]. Specifically, these works use a fixed number of learnable latent query vectors to query the long visual sequences output, obtaining the final fixed-length visual sequence representations, in an abstractive way. An obvious bottleneck is that they can not optimize the costly and potentially unnecessary self-attention calculation in the visual backbones. Meanwhile, this visual representation abstraction process only considers the semantics of the visual modality, ignoring the textual guidance and consequently leading to representation deficiency.

- **Bottom-level Extraction**. The second line focuses on reducing patch tokens in the ViT-backbone from the bottom-level perspective, usually in an extractive manner [39, 29, 16] . The problem here lies in that overly reducing the visual sequence by extracting critical tokens in the backbone, while accelerating the attention calculation, may deconstruct images' structural information. Therefore, balancing efficiency and effectiveness remains a bottleneck.

To achieve a better trade-off between the efficiency and effectiveness of VLP, we propose integrating the merits of top-level abstraction and bottom-level extraction. Inspired by bottom-up text summarization[4, 17], which first select key phrases and then abstractively generate the final text summaries, we design a bottom-up summarization process for visual tokens. We first exploit coarse-grained key patch extraction in the ViT backbone, with regulation from text modality, to identify text-relevant tokens and remove potentially redundant ones, reducing the computational cost in the ViT backbone. Then fine-grained text-guided patch abstraction is performed upon the output sequence of the ViT backbone to obtain a further condensed visual representation sequence.

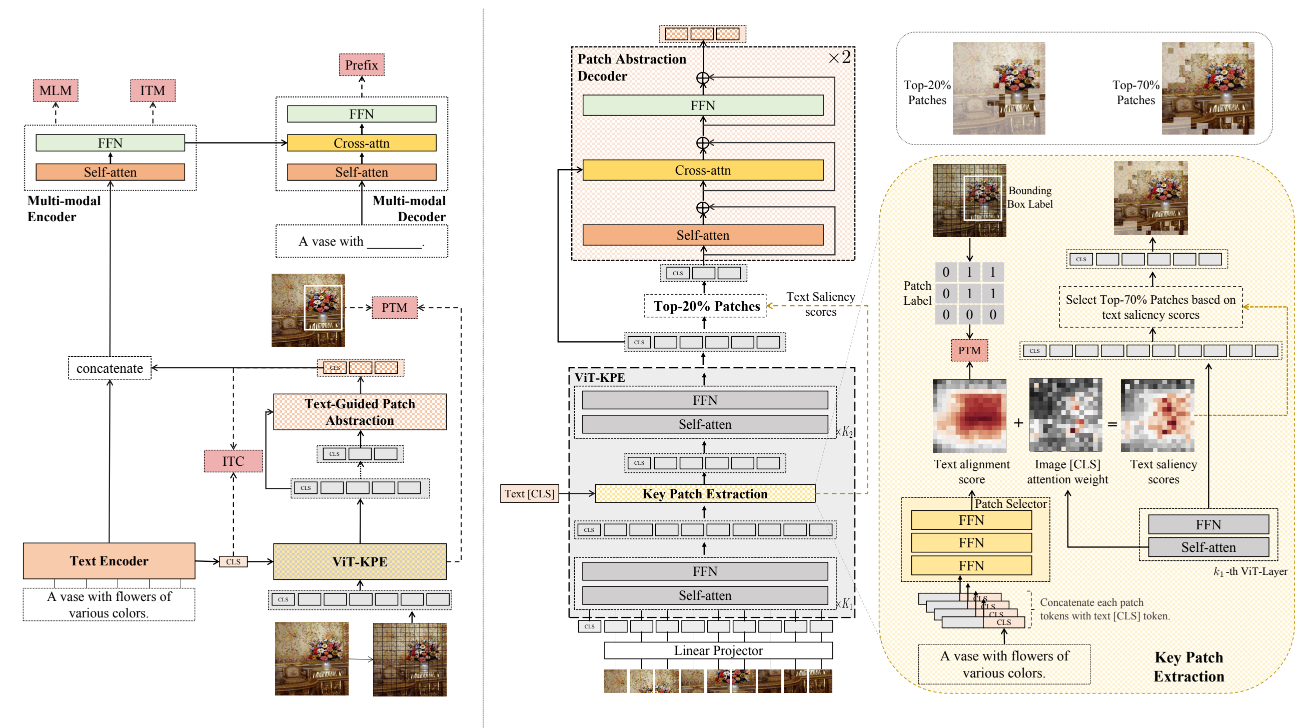
Specifically, we incorporate a Text Semantic-aware Patch Selector (TSPS) module into the ViT-based backbone for bottom-level extraction. We transform object/region annotations to patch-level annotations to train an effective extractor with a novel auxiliary pre-training task named Patch-Text Matching (PTM), which facilitates patch extraction and fine-grained patch-text alignment. Next, we introduce a lightweight Transformer-based Patch Abstraction Decoder (PAD) for top-level abstraction. It takes the top-K text-relevant patch tokens from the output sequence of ViT-backbone as the input and the overall visual sequence as the encoder hidden states to generate the final visual patch summary.

We evaluate BUS on various representative VL understanding and generation tasks, including visual question answering, cross-modal retrieval, and image captioning. We find that by reducing the length of the patch sequence to 20% of its original length, we can not only get competitive or better downstream task performance but also enjoy a significant increase in efficiency over previous similar VLP models. For instance, BUS reduces about 51% of the inference time (see Table 7) and even improves by about 0.3 on the VQA test-dev with the same experimental settings. Furthermore, by increasing the input image resolution, BUS achieves state-of-the-art downstream task performance (e.g., 78.28 on VQA test-dev) which benefits from processing more image tokens without increasing computational costs.

2. Related Work

2.1. Vision-Language Pre-training

Previous research on vision language pre-training has primarily fallen into two categories: Detector-based VLP models and CNN/ViT-based VLP models. Detector-based methods, such as [32, 27, 45, 28, 7, 54], use a two-step training pipeline that first extracts visual features using a pre-trained object detector and then aligns text and visual features using a cross-modal pre-training model. While some region-based methods, such as [47], use lightweight model architectures to reduce computation costs, they still face expensive computational and time-consuming object detection. Recently, Vision Transformer (ViT) based methods, such as [26, 20, 38, 48, 25, 23, 49, 20] have emerged as a promising alternative to detector-based approaches. ViT-based models eliminate the need for object detectors in feature extraction, enabling end-to-end vision language learning. However, they struggle with lengthy visual token sequences and lack fine-grained cross-modal alignment information. These long visual sequences also increase computation costs and introduce noise visual information for cross-modal fusion. To address these challenges, we propose a Bottom-Up Patch Summarization approach that coordinates bottom-level extraction and top-level abstraction to efficiently learn a concise summary of lengthy visual token sequences.



(a) Overview (b) Bottom-Up Patch Summarization

Figure 2: (a) Overview of our VLP model BUS . (b) The details of the Bottom-Up Patch Summarization.

2.2. Text Summarization

Existing text summarization methods can be roughly categorized into extractive summarization [34, 50, 59] and abstractive summarization [44, 41, 36, 57, 22, 46]. Extractive summarization extracts key sentences from the source document to form a summary. It can be formulated as a sentence classification problem [34, 33, 8]. Abstractive summarization paraphrases important content after understanding the original document and constructs an abstract with newly generated words and coherent expressions. Our proposed Bottom-Up patch Summarization is unpaired from the Text Summarization Task which contains a key patch extraction module and patch abstraction module.

3. Method

In this section, we first provide an overview of our BUS architecture. Then, we introduce the proposed Bottom-Up patch summarization which includes the Key Patch Extraction (KPE) incorporated in the Vision Transformer (ViT) backbone with a Text Semantic-aware Patch Selector (TSPS) and the Text-Guided Patch Abstraction (TPA) conducted on the output sequence of the ViT backbone. We leave the introduction of the pre-training task which includes the pre-training task Patch-Text Matching used to learn TSPS and the pre-training schedule in Appendix A.3.

3.1. Model Architecture

As depicted in Figure 2 (b), BUS comprises a ViT-based image encoder with a Text Semantic-aware Patch Selector (TSPS) for coarse-grained Key Patch Extraction (KPE), a text encoder, a lightly Patch Abstraction Decoder(PAD) for fine-grained Text-Guided Patch Abstraction (TPA), a multi-modal fusion encoder for performing cross-modal interaction, and a multi-modal decoder for text generation. (Note that the text encoder contains 10 Transformer layers and PAD contains only 2 Transformer layers, which are initialized with BERT$_{base}$ [10]).

Formally, let us consider an input image-text pair denoted as (I, T). For the input text, we feed it to the text encoder and obtain the text representation $T = \{t_{cls}, t_1, t_2, \cdots, t_m\}$, where t_{cls} is the embedding of the text [CLS] token used to summarize the global semantic information of the text. For the input image, we divide it into n non-overlapping patches $P = \{p_{cls}, p_1, p_2, \cdots, p_n\}$. Then, we feed the patch sequence to the visual encoder. In the ViT backbone, we apply KPE to select text-relevant image patches that can reduce the visual sequence length and improve the training and inference efficiency of the ViT backbone. Suppose the output patch sequence representation of the ViT-backbone can be denoted as $V = \{v_{cls}, v_1, v_2, \cdots, v_u\}$, where $u<n$. Then, we feed the output patch sequence to PAD and conduct TPA to obtain the

final visual summary of the long patch sequence, which can be denoted as $\hat{V} = \{\hat{v}_{cls}, \hat{v}_1, \hat{v}_2, \cdots, \hat{v}_s\}$, where $s \ll u$. The image and text representations are concatenated and fed into the cross-modal encoder. We obtain the cross-modal representations $\{c_{cls}, c_1, c_2, \ldots, c_l\}$, where $l = s + m$. The cross-modal representations can be used to fine-tune downstream multi-modal understanding tasks. Additionally, the output cross-modal representations $\{c_{cls}, c_1, c_2, \ldots, c_l\}$ of the multi-modal encoder are fed into a Transformer decoder for sequence-to-sequence learning

3.2. Bottom-Up Patch Summarization

The proposed Bottom-Up Patch Summarization consists of two steps: Key Patch Extraction (KPE) and Text-Guided Patch Abstraction (TPA). KPE is conducted within the visual backbone to select a coarse-grained subset of text-relevant patch tokens. Then, outside the backbone, Text-Guided Patch Abstraction is conducted to abstract the selected tokens and obtain a more prominent visual summary.

3.2.1 Key Patch Extraction

Figure 2 (b) shows how we perform Key Patch Extraction (KPE) in the ViT-based visual backbone based on the Text Semantic-aware Patch Selector (TSPS). Similar to the text extractive summarization methods[50, 59], we view the KPE as a patch classification task that aims to classify whether a patch is aligned with the text semantic and should be selected. Specially, suppose the TSPS is plugged between the k_{th} ($1 \leq k < N$) Transformer layer and $(k+1)_{th}$ Transformer layer and the output patch sequence features of k_{th} Transformer layer is $v^k = \{v^k_{cls}, v^k_1, \cdots, v^k_n\}$. The text [CLS] feature is output by the text encoder and represents global information of the input text T. We will first concatenate the text [CLS] feature with each image patch token as follows:

$$\dot{v}^k_i = concat(v^k_i, t_{cls})$$

where $v^k_i \in R^d, t_{cls} \in R^d, \dot{v}^k_i \in R^{2d}, i \in \{1, 2, \ldots, n\}$. Then the concatenated patch features $\dot{v}^k_i$ are fed to TSPS, which is a Multi-layer Perceptron (MLP) that contains three linear layers and is used to predict the alignment score between patches and the input text T. The first two linear layers will linearly project the concatenated patch features $\{\dot{v}^k_i\}$ to the hidden representations $\{h^k_i\}$ and then the hidden representations $\{h^k_i\}$ is fed to the last linear layer denoted as $\mathbf{F}_\theta$ which can be seen as a classifier to predict whether the patches are relevant to the input text. The output of the last linear layer has only one dimension and will be fed to a Sigmoid activation function. Formally, the alignment score a_i between the i_{th} image patch and input text T can be calculated as follow:

$$a_i = Sigmoid(\mathbf{F}_\theta(h^k_i)), i \in \{1, 2, \ldots, n\}$$

To learn an effective patch selector, we generate patch-level training labels by transforming the bounding box annotations (as described in Appendix A.2). However, this approach introduces bias, as illustrated in Figure 2 (b), where TSPS predicts all patches within the bounding box with a high alignment score, even if some of them are irrelevant. To address this issue, we incorporate the attention map of the image [CLS] token into the selection process. Previous work has shown that in ViTs, the image [CLS] token pays more attention to class-specific tokens than to tokens in non-object regions [6]. By weighting the alignment score a and the attention map p of the image [CLS] token to other tokens, we can highlight the patches corresponding to objects in the detection box and suppress other patches. This approach reduces the bias introduced by the bounding box labels and can be formulated as follows:

$$\dot{a}_i = \beta * \mathbf{F}_N(a_i) + (1 - \beta) * \mathbf{F}_N(p_i)$$

Here, p_i is the attention value of the image [CLS] token to the i-th patch token, calculated in the k_{th} Transformer layer. We define $\dot{a}_i$ as the text saliency score of the i-th patch in the sequence, β as a hyper-parameter, and $\mathbf{F}_N$ as a normalization function for p_i and a_i. Then, we select the top-u image patch tokens from the patch sequence $\{v^k_{cls}, v^k_1, \cdots, v^k_n\}$ based on their text saliency scores $\{\dot{a}_1, \cdots, \dot{a}_n\}$, where $u = n \times \alpha$ and α is the selection ratio that controls the proportion of selected patches to total patches. The selected top-u image patch tokens are kept and we reconstruct the k_{th} visual sequence as $v^k = \{v^k_{cls}, v^k_1, \cdots, v^k_u, v^k_u\}$. Then the reconstructed visual sequence is fed to the next $(k+1)_{th}$ Transformer layer.

3.2.2 Text-Guided Patch Abstraction

While the coarse-grained key patch extraction removes some redundant visual tokens, many text-irrelevant tokens remain in the ViT backbone. However, we need to be cautious not to remove too many visual tokens in the ViT backbone, as this can lead to the loss of important structural information and affect the distribution of hidden representations in the backbone. To address this issue, we propose the Text-guided Patch Abstraction (TPA) outside the ViT backbone, using a lightweight Patch Abstraction Decoder (PAD). As shown in Figure 2 (b), the PAD consists of two transformer modules, each of which includes a self-attention layer and a cross-attention layer. Similar to [3] and [24], the model structure of PAD employs a similar way, but with a key difference: we select top-s image patch tokens $\{\overline{v}_{cls}, \overline{v}_1, \cdots, \overline{v}_s\}$ from the output patch sequence $\{v_{cls}, v_1, \cdots, v_s\}$ of ViT backbone based on the text saliency scores $\{\dot{a}_1, \cdots, \dot{a}_n\}$, and take them as the input to guide the PAD to condense visual information while they used the static learnable embeddings without taking any prior of text information. Noted that $s = \gamma * u$ and γ

is the selection ratio for TPA. These top-s patch tokens are highly relevant to the text semantics and provide a strong prior to help the PAD learn a condensed visual summary and highlight text-relevant visual information. After we select the top-s image patch tokens $\{\overline{v}_{cls}, \overline{v}_1, \cdots, \overline{v}_s\}$, we will feed them to PAD. In each transformer module of the PAD, $\{\overline{v}_{cls}, \overline{v}_1, \cdots, \overline{v}_l\}$ first undergoes the self-attention layer, followed by a cross-attention with the output sequence $\{v_{cls}, v_1, \cdots, v_s\}$. Finally, the output of the PAD denoted as $\hat{V} = \{\hat{v}_{cls}, \hat{v}_1, \hat{v}_2, \cdots, \hat{v}_s\}$ serves as the final visual summary.

4. Experiments

4.1. Data & Setup

In our work, we follow the pre-training setup established in [26] and utilize the same pre-training dataset consisting of 4 million images with associated texts. The dataset comprises two in-domain datasets, MS COCO [30] and Visual Genome [21], as well as three out-domain datasets, Conceptual Captions [42], and SBU Captions [35]. Further details on the dataset can be found in Appendix A.1.

We pre-train the model for 30 epochs with a total batch size of 1024 on 8 NVIDIA A100 GPUs. We use a 10-layer Transformer for the text encoder, a 2-layer transformer for the patch abstraction decoder, a 3-layer for the cross-modal encoder network, and a 12-layer Transformer for the cross-modal decoder. Specifically, we initialize the text encoder using the first 10 layers of the BERT$base$ [10] model, initialize the patch abstraction decoder using the last 2 layers of BERT$base$, initialize the cross-modal encoder network using the last 3 layers of BERT$base$.

4.2. Main Result

We evaluate the effectiveness of our proposed model, BUS , on four well-established vision-language downstream tasks: Visual Question Answering (VQA), Cross-modal Retrieval, Image Captioning, and Visual Grounding (VG). The Key Patch Extraction (KPE) is performed after the 6th Transformer layer in the ViT encoder, where about 70% text-relevant patch tokens are selected after the coarse-grained extraction. For the Text-Guided Patch Abstraction (TPA), we select the top 20% of text-relevant patch tokens as the input for the Patch Abstraction Decoder (PAD). Our experiments in subsection 4.5 and subsection 4.5 demonstrate that this setting achieves the desired trade-off between downstream task performance and model inference speed. Please refer to Appendix A.1 for more information on the datasets and fine-tuning hyper-parameters, and to Appendix C for details on the comparison methods used in the experiments.

4.2.1 Visual Question Answering

The VQA task [1] requires the model to answer natural language questions given an image. Following the approach proposed in [26], we treat VQA as an answer-generation problem. We evaluate the performance of our proposed model, BUS , by submitting our results to the evaluation server [1] in Table 1. and report the test-dev and test-std scores in Table 1. Our results show that BUS achieves comparable performance with state-of-the-art models under the same image resolution (384 $\times$ 384) while being about 50% faster in terms of model inference time (as reported in subsection 4.6). Furthermore, when we increase the image resolution to 512 $\times$ 512, our model achieves state-of-the-art performance while maintaining a similar inference computation cost to other baselines. These results demonstrate the effectiveness and efficiency of BUS in VQA tasks.

4.2.2 Image Captioning

For the image captioning task, where there is no textual input, we set the hyper-parameter β to 0 and select patches based on the attention weight of the image [CLS] token to other image tokens. Following [28], we fine-tuned BUS with cross-entropy loss and then with CIDEr optimization for an extra 5 epochs. Our experiments, as shown in Table 1, demonstrate that BUS achieves comparable results with SOTA models when the image resolution is set to 384 $\times$ 384. Moreover, when we set the image resolution to 512 $\times$ 512, BUS performs even better on CIDEr evaluation, surpassing state-of-the-art models.

4.2.3 Image-Text Retrieval

We conduct experiments on MSCOCO [30] and Flickr30K [37] datasets for both image-to-text retrieval (TR) and text-to-image retrieval (IR), and jointly optimize the ITC loss and the ITM loss during fine-tuning. The results are reported in Table 2. Our model demonstrates comparable performance with other VLP baselines, as shown in the experimental results.

4.2.4 Visual Grounding

Table 3 demonstrates the performance of BUS in the visual grounding task. When the image resolution is set to 384 $\times$ 384, our model achieves comparable results with competitive baseline methods. Thanks to our bottom-up summarization mechanism, we are able to improve the image resolution and obtain better results without incurring additional computational costs compared to other methods. When we increase the image resolution to 512 $\times$ 512, our

[1] https://eval.ai/web/challenges/challenge-page/830/overview

Models	# Pre-train Data	VQA		COCO Caption								NoCaps	
				Cross-entropy Optimization				CIDEr Optimization					
		Test-std	Test-dev	B@4	M	C	S	B@4	M	C	S	C	S
E2E-VLP [51]	4M	73.25	73.67	36.2	-	117.3	-	-	-	-	-	-	-
OSCAR [28]	6.5M	73.16	73.44	-	-	-	-	41.7	30.6	140.0	24.5	83.4	11.4
VinVL [58]	5.65M	76.52	76.60	38.5	30.4	130.8	23.4	41.0	31.1	140.9	25.2	97.3	13.8
METER [11]	4M	77.68	77.64	-	-	-	-	-	-	-	-	-	-
BLIP [25]	14M	77.54	77.62	38.6	-	129.7	-	-	-	-	-	**105.1**	**14.4**
SimVLM [49]	1.8B	77.87	78.14	39.0	**32.9**	**134.8**	24.0	-	-	-	-	-	-
ALBEF [26]	4M	74.54	74.70	-	-	-	-	-	-	-	-	-	-
ALBEF [26]	14M	75.84	76.04	-	-	-	-	-	-	-	-	-	-
XVLM [56]	4M	78.07	78.09	39.8	-	133.1	-	41.3	-	140.8	-	-	-
mPLUG [23]	4M	77.55	77.73	39.3	30.1	132.4	23.34	41.2	30.8	140.2	25.2	98.3	12.9
BUS $_{384}$	4M	77.89	77.98	39.5	30.9	132.6	23.93	41.4	31.0	140.7	25.3	98.8	12.9
BUS $_{512}$	4M	**78.28**	**78.34**	**40.04**	31.3	133.6	**24.12**	**41.8**	**31.4**	**141.1**	**25.9**	99.1	13.2

Table 1: Evaluation Results on VQA, image captioning on COCO Karpathy test split [19] and NoCaps [2] validation set. B@4: BLEU@4, M: METEOR, C: CIDEr, S: SPICE. More details about comparison models in Appendix C, BUS $_{384}$ means we set image resolution to 384×384 during finetuning. Similar, BUS $_{512}$ means we set image resolution to 512×512.

Models	# Pre-train data	MSCOCO (5K test set)						Flickr30K (1K test set)					
		TR			IR			TR			IR		
		R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
ALIGN [15]	1.8B	77.0	93.5	96.9	59.9	83.3	89.8	95.3	99.8	100.0	84.9	97.4	98.6
OSCAR [28]	4M	70.0	91.1	95.5	54.0	80.8	88.5	-	-	-	-	-	-
E2E-VLP [51]	4M	-	-	-	-	-	-	86.2	97.5	98.92	73.6	92.4	96.0
UNITER [7]	4M	65.7	88.6	93.8	52.9	79.9	88.0	87.3	98.0	99.2	75.6	94.1	96.8
VLMo [48]	4M	78.2	94.4	97.4	60.6	84.4	91.0	95.3	99.9	100.0	84.5	97.3	98.6
ALBEF [26]	14M	77.6	94.3	97.2	60.7	84.3	90.5	95.9	99.8	100.0	85.6	97.5	**98.9**
XVLM [56]	4M	80.4	95.5	**98.2**	63.1	**85.7**	**91.6**	96.8	99.8	100.0	86.1	97.4	98.7
mPLUG [23]	4M	80.5	95.4	97.9	63.3	85.3	91.2	96.7	99.8	100.0	86.5	97.5	98.8
BUS	4M	**80.6**	**95.7**	98.0	**63.6**	85.5	91.5	**97.0**	99.8	100.0	**86.9**	**97.8**	98.8

Table 2: Evaluation results of image-text retrieval on Flickr30K [37] and COCO datasets [30].

Model	RefCOCO+		
	testA	testB	val
UNITER [7]	75.90	81.45	66.70
VL-BERT [43]	72.59	78.57	62.30
ViLBERT [32]	72.34	78.52	62.61
VILLA [12]	76.17	81.54	66.84
MDETR [18]	79.52	84.09	70.62
UNICORN [52]	80.30	85.05	**71.88**
XVLM [56]	80.17	86.36	71.00
mPLUG [23]	80.07	85.21	71.03
BUS $_{384}$	80.11	86.03	71.21
BUS $_{512}$	**80.36**	**86.61**	71.84

Table 3: Evaluation results of visual grounding on RefCOCO+. We use the accuracy of IOU 0.5 on visual grounding (a prediction is right if the IoU between the grounding-truth box and the predicted bounding box is larger than 0.5)

model outperforms all state-of-the-art methods, demonstrating the effectiveness and efficiency of BUS .

4.3. Efficiency of BUS

To investigate the efficiency of the BUS , we conduct the following experiments. We first compare the computational complexity of recent SOTA VLP models and report the Floating Point Operations Per second (FLOPs) which is a widely used evaluation metric for model computational complexity. In addition, we evaluate the computational speed of our model by comparing the throughput and latency of different models. We use a Xeon Platinum 8163 CPU and an NVIDIA V100 GPU to calculate the latency and throughput. As shown in Table 4 and Figure 3, our BUS not only has the lowest computational complexity (e.g., 19.03 of FLOPs) but also the fastest computational speed (e.g., 360.23 Throughput and 14ms Latency).

4.4. The Impact of Selection Ratio for Text-Guided Patch Abstraction

To investigate the impact of the selection ratio on Text-Guided Patch Abstraction (TPA), we trained BUS with varying selection ratios for the input token in the patch abstraction decoder of TPA. We kept the selection location of the Text Semantic-aware Patch Selector after the 6th transformer layer and the selection ratio for Key Patch Extraction at 70% unchanged. As shown in Figure 4, two conclusions

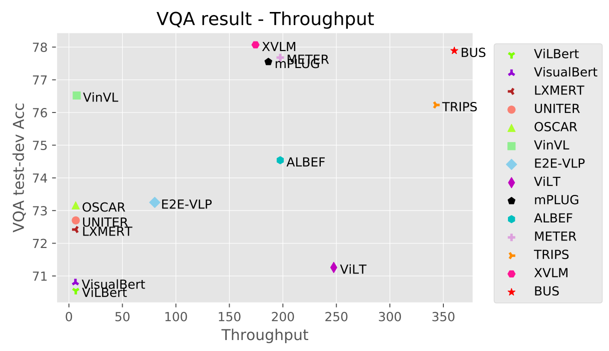


Figure 3: The visualization of the VQA test-dev result and Throughput of different VLP models.

Models	Latency (ms)	FLOPs (G)	Throughput
UNITER [7]	870ms	949.90	6.42
OSCAR [28]	860ms	956.40	6.35
VinVL [58]	640ms	1023.30	7.32
E2E-VLP [51]	70ms	144.30	80.23
ViLT [20]	19ms	55.40	247.53
ALBEF [26]	22ms	33.42	197.52
XVLM [56]	27ms	38.65	174.42
TRIPS [16]	11ms	20.89	343.05
mPLUG [23]	24ms	36.63	186.42
BUS	**10**ms	**19.03**	**360.23**

Table 4: The comparison of the efficiency of different models. FLOPs, throughput, and latency are reported here. Since FLOPs are proportional to input size, for a fair comparison, we use same the input size with [20], which is 197 for image patches length and 40 for text tokens length. We also keep the same setting when calculating throughput and latency.

Locs	SR-KPE	SR-TPA	VQA	FLOPs (G)	Throughput
4	40%	20%	76.22	16.22	443.48
4	70%	20%	77.19	18.53	381.65
4	90%	20%	77.38	20.04	338.12
6	40%	20%	76.87	17.24	410.05
6	70%	20%	77.89	19.03	360.23
6	90%	20%	77.92	21.77	316.11
8	40%	20%	76.97	19.09	353.13
8	70%	20%	77.94	21.65	312.06
8	90%	20%	78.01	23.24	242.88

Table 5: Results of pre-training and fine-tuning BUS with different selection locations and selection ratios. We report the text-dev score results of VQA, FLOPs, and Throughput. SR-KPE refers to the Selection Ratio for Key Patch Extraction, and SR-TPA refers to the Selection Ratio for Text-Guided Patch Abstraction.

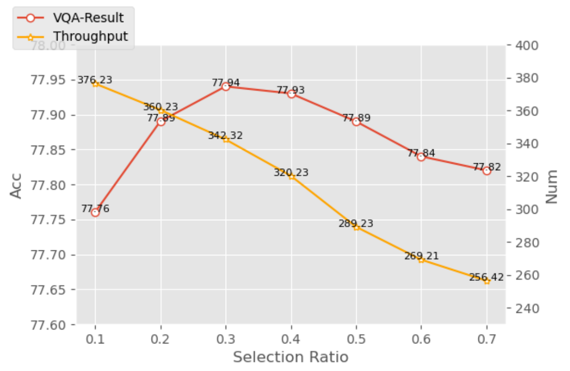


Figure 4: VQA performance and throughput of BUS on different selection ratio of Text-Guided Patch Abstraction.

can be drawn: First, as the selection ratio increases, VQA performance first improves and then decreases, indicating that the “vanishing information” problem can hinder the effectiveness of the VLP model and highlighting the importance of shortening the visual path sequence. Second, setting the selection ratio to 20% can achieve a good trade-off between effectiveness and efficiency.

4.5. The Impact of Selection Location and Selection Ratio for Key Patch Extraction

To validate the impact of the location of the Text Semantic-aware Patch Selector (TSPS) in the ViT backbone and selected ratio in the KPE on the efficiency and effectiveness of BUS , we trained BUS with different selection locations and selection ratios. Note that when calculating FLOPs and throughput, we set the input image size to 224×224 and the input text length to 40. As shown in Table 5, two conclusions can be drawn: (1), plugging the TSPS after shallower layers can reduce computational complexity but deteriorate accuracy. For example, the accuracy drops considerably with the remarkable increase in throughput when TSPS is placed after the 4th layer. A possible explanation is that patch embeddings in shallow layers cannot sufficiently represent visual semantics, making it difficult to learn fine-grained patch-text alignment, which leads to a drop in accuracy. (2), too many undetected image tokens fused in the TSPS module will considerably decrease downstream task performance. For example, if we place the TSPS after the 4th layer in ViT and set the keeping ratio to 40%, performance will decrease to 76.22 on the VQA task, compared to 77.89 of the model with a 70% selection ratio in the 6th layer.

4.6. Fine-tuning on Higher Resolution Images

To test the efficacy of our approach, we fine-tune BUS for the VQA task, taking images of varying resolutions as input. Table 6 shows our results. Our experiments demonstrate that increasing the input image resolution facilitates the model by allowing it to take more image tokens, leading

Loc	SR-KPE	SR-TPA	image size	VQA	FLOPs	Troughout
-	-	-	384 × 384	77.55	84.87	63.57
6	70%	20%	224 × 224	77.03	**19.03**	**360.23**
6	70%	20%	256 × 256	77.61	24.04	255.03
6	70%	20%	304 × 304	77.78	32.89	210.62
6	70%	20%	384 × 384	77.89	47.24	123.52
6	70%	20%	464 × 464	78.16	75.62	83.83
6	70%	20%	512 × 512	**78.28**	84.31	64.21

Table 6: Results of BUS finetuning on VQA task with different resolution images. The settings for calculating FLOPs and throughput are the same as Table 4 except for the image resolution. The first row in the table reports the result of the recent SOTA baseline mPLUG[23].

to improved performance. For instance, when fine-tuned with 512×512 images, BUS achieves a score of 78.28 on the VQA task, outperforming the baseline model fine-tuned with 384×384 images while maintaining a similar level of computational complexity.

4.7. Ablation Study

model	Loc	SR-KPE	SR-TPA	VQA	FLOPs(G)	Throughput
BUS	6	70%	20%	77.89	19.03	360.23
-w/o KPE	-	-	20%	77.80	23.15	246.56
-w/o TPA	6	70%	-	77.64	25.63	232.42
-w/o BUS	-	-	-	77.58	37.62	170.23
-w/o IA	6	70%	20%	77.67	18.89	362.40
-w/o TA	6	70%	20%	77.43	18.67	366.23

Table 7: The result of ablations. We fine-tune BUS on VQA and report test-dev results, FLOPs, and Throughput. The setting for calculating FLOPs and throughput is the same as Table 4.

We conducted ablation studies to investigate the effects of our proposed bottom-up patch summarization mechanism. Specifically, we examined the impact of removing the Key Patch Extraction (KPE) and Text-Guided Patch Abstraction (TPA) on both performance and efficiency. In Table 7, "w/o EPA" denotes the case where we remove KPE from the visual backbone but keep TPA outside the visual backbone, while "w/o TPA" is the opposite case where we remove TPA but keep KPE. "w/o BUS" indicates the removal of overall Bottom-Up Summarization from our VLP model. As shown in Table 7, we observed that removing KPE and only using TPA on the top level did not result in a significant improvement in performance, but it led to a decrease in model efficiency. This suggests that the additional computational cost inside the ViT backbone cannot be ignored, and it highlights the effectiveness and efficiency of KPE. On the other hand, when we removed top-level TPA but kept bottom-level KPE, we observed a decrease

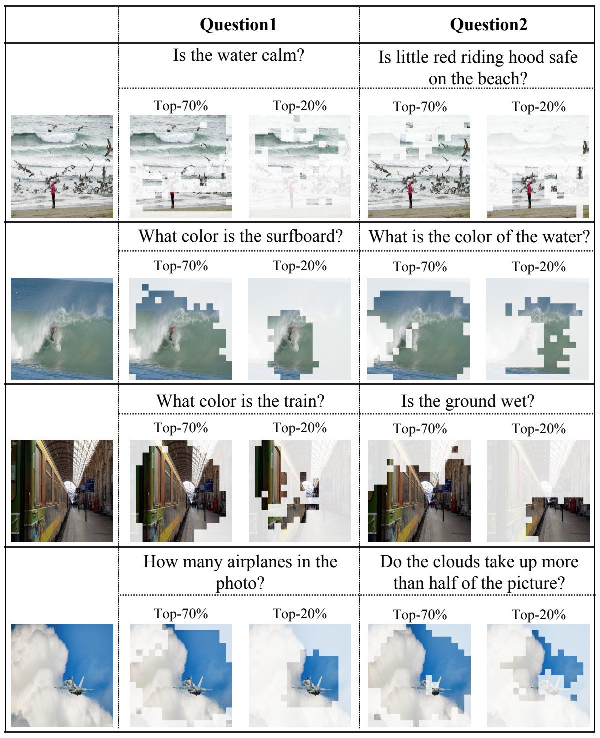

Question1
Question2
Is the water calm?
Is little red riding hood safe on the beach?
Top-70%
Top-20%
What color is the surfboard?
What is the color of the water?
What color is the train?
Is the ground wet?
How many airplanes in the photo?
Do the clouds take up more than half of the picture?

Figure 5: The visualization of the VQA case and the selected text-relevant image patches.

in VQA performance and throughput, which indicates the importance of TPA in achieving good performance. Furthermore, when we removed both KPE and TPA (i.e., "w/o BUS"), we observed a more significant decrease in both performance and speed. This finding underscores the effectiveness and efficiency of our bottom-up patch summarization mechanism, which achieved a better trade-off between performance and efficiency.

In addition, we also verified the effectiveness of the guidance of text semantics for patch summarization and the effectiveness of introducing attention maps of the visual [CLS] token in KPE. Specifically, "w/o TA" means that in KPE and TPA, we only use the attention maps of the visual [CLS] token to other tokens to select tokens. Table 7 shows a significant decrease in performance, which proves the effectiveness of the guidance of text semantics for patch summarization. In addition, "w/o IA" means that we only use the text alignment score predicted by TSPS to select tokens, without using visual information. We also found a slight decrease in performance, which confirms that introducing visual information (attention map of image [CLS] token) can address the bias introduced by bounding box labels mentioned in Subsection 3.2.1.

4.8. Case Study

The proposed bottom-up patch summarization will select the text-consistent image tokens both in the ViT backbone and outside the ViT backbone with the regulation of the text semantics. To further investigate the effectiveness of text guidance for the bottom-up patch summarization, we visualize the VQA case and the selected text-relevant image patches in Figure 5. It can be seen that based on different text questions, we can effectively select the text-relevant patches while reducing the other text-irrelevant patches. For example, in the first case, the first question is "Is the water calm?", we can effectively select the patches of the water in the image while ignoring the human and the beach. For the second question "Is little red riding hood safe on the beach", we can still focus on the corresponding image patches ("little red riding hood" and "beach"). We demonstrate more cases in Appendix D.

5. Conclusion

We have presented BUS with a novel Bottom-Up Patch Summarization mechanism that achieves an ideal trade-off between efficiency and effectiveness, resulting in a highly efficient and effective ViT-based VLP model. Our approach utilizes a Text-Semantics-Aware Patch Selector to perform bottom level Key Patch Extraction, followed by a Transformer-based Patch Abstraction Decoder for top-level visual abstraction. The combination of the two components enables our BUS to learn a concise summary of lengthy visual token sequences efficiently and effectively. The experiment shows our method improves efficiency due to the reduction of visual sequences while keeping or even improving the performance of downstream tasks.

References

[1] Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Devi Parikh, and Dhruv Batra. Vqa: Visual question answering. *International Journal of Computer Vision*, 123:4–31, 2015. 5, 13

[2] 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. *CoRR*, abs/1812.08658, 2018. 6, 14

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *ArXiv*, abs/2204.14198, 2022. 2, 4

[4] Valerie Anderson and Suzanne Hidi. Teaching students to summarize. *Educational leadership*, 46(4):26–28, 1988. 2

[5] Bin Bi, Chenliang Li, Chen Wu, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. Palm: Pre-training an autoencoding&autoregressive language model for context-conditioned generation. *arXiv preprint arXiv:2004.07159*, 2020. 12

[6] Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9630–9640, 2021. 4

[7] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020. 1, 2, 6, 7, 14

[8] Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. *ArXiv*, abs/1603.07252, 2016. 3

[9] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 12

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019. 3, 5, 12

[11] Zi-Yi Dou, Yichong Xu, Zhe Gan, Jianfeng Wang, Shuohang Wang, Lijuan Wang, Chenguang Zhu, Zicheng Liu, Michael Zeng, et al. An empirical study of training end-to-end vision-and-language transformers. *arXiv preprint arXiv:2111.02387*, 2021. 1, 6, 14

[12] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. In *NeurIPS*, 2020. 6, 15

[13] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017. 13

[14] Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. Pixel-bert: Aligning image pixels with text by deep multi-modal transformers. *ArXiv*, abs/2004.00849, 2020. 1

[15] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V Le, Yunhsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. *arXiv preprint arXiv:2102.05918*, 2021. 6, 14

[16] Chaoya Jiang, Haiyang Xu, Chenliang Li, Ming Yan, Wei Ye, Shikun Zhang, Bin Bi, and Songfang Huang. Trips: Efficient vision-and-language pre-training with text-relevant image patch selection. In *Conference on Empirical Methods in Natural Language Processing*, 2022. 2, 7, 15

[17] Hongyan Jing and Kathleen R McKeown. The decomposition of human-written summary sentences. In *Proceedings of the 22nd annual international ACM SIGIR conference on Re-*

search and development in information retrieval, pages 129–136, 1999. 2

[18] Aishwarya Kamath, Mannat Singh, Yann LeCun, Ishan Misra, Gabriel Synnaeve, and Nicolas Carion. Mdetr - modulated detection for end-to-end multi-modal understanding. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1760–1770, 2021. 6

[19] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3128–3137, 2015. 6, 14

[20] Wonjae Kim, Bokyung Son, and Ildoo Kim. Vilt: Vision-and-language transformer without convolution or region supervision. In *ICML*, 2021. 1, 2, 7, 15

[21] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123:32–73, 2016. 5, 12, 13

[22] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *ArXiv*, abs/1910.13461, 2020. 3

[23] Chenliang Li, Haiyang Xu, Junfeng Tian, Wei Wang, Ming Yan, Bin Bi, Jiabo Ye, Hehong Chen, Guohai Xu, Zheng Cao, Ji Zhang, Songfang Huang, Fei Huang, Jingren Zhou, and Luo Si. mplug: Effective and efficient vision-language learning by cross-modal skip-connections, 2022. 2, 6, 7, 8, 15

[24] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *ArXiv*, abs/2301.12597, 2023. 2, 4

[25] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. *arXiv preprint arXiv:2201.12086*, 2022. 2, 6, 14, 15

[26] Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq R. Joty, Caiming Xiong, and Steven C. H. Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *NeurIPS*, 2021. 1, 2, 5, 6, 7, 12, 13, 14

[27] 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*, abs/1908.03557, 2019. 2

[28] Xiujun Li, Xi Yin, Chunyuan Li, Xiaowei Hu, Pengchuan Zhang, 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 *ECCV*, 2020. 1, 2, 5, 6, 7, 13, 14

[29] Youwei Liang, Chongjian Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Evit: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2022. 2

[30] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5, 6, 12, 13, 14

[31] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 12

[32] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019. 2, 6, 15

[33] Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 3

[34] Shashi Narayan, Shay B Cohen, and Mirella Lapata. Ranking sentences for extractive summarization with reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1747–1759, 2018. 3

[35] Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. Im2text: Describing images using 1 million captioned photographs. In *NIPS*, 2011. 5

[36] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *International Conference on Learning Representations*, 2018. 3

[37] Bryan A. Plummer, Liwei Wang, Christopher M. Cervantes, Juan C. Caicedo, J. Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *International Journal of Computer Vision*, 123:74–93, 2015. 5, 6, 14

[38] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 2

[39] Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *NeurIPS*, 2021. 2

[40] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1179–1195, 2017. 14

[41] Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, 2017. 3

[42] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, 2018. 5

[43] 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*, abs/1908.08530, 2020. 6, 14

[44] Ayesha Ayub Syed, Ford Lumban Gaol, and Tokuro Matsuo. A survey of the state-of-the-art models in neural abstractive text summarization. *IEEE Access*, 9:13248–13265, 2021. 3

[45] Hao Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *ArXiv*, abs/1908.07490, 2019. 1, 2, 13, 15

[46] Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. Abstractive document summarization with a graph-based attentional neural model. In *ACL*, 2017. 3

[47] Jianfeng Wang, Xiaowei Hu, Pengchuan Zhang, Xiujun Li, Lijuan Wang, L. Zhang, Jianfeng Gao, and Zicheng Liu. Minivlm: A smaller and faster vision-language model. *ArXiv*, abs/2012.06946, 2020. 2

[48] Wenhui Wang, Hangbo Bao, Li Dong, and Furu Wei. Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. *ArXiv*, abs/2111.02358, 2021. 2, 6, 13, 14

[49] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision. *ArXiv*, abs/2108.10904, 2021. 1, 2, 6, 13, 14

[50] Wen Xiao and Giuseppe Carenini. Extractive summarization of long documents by combining global and local context. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3011–3021, 2019. 3, 4

[51] Haiyang Xu, Ming Yan, Chenliang Li, Bin Bi, Songfang Huang, Wenming Xiao, and Fei Huang. E2e-vlp: End-to-end vision-language pre-training enhanced by visual learning. *ArXiv*, abs/2106.01804, 2021. 6, 7, 14

[52] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. Crossing the format boundary of text and boxes: Towards unified vision-language modeling. *ArXiv*, abs/2111.12085, 2021. 6

[53] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Faisal Ahmed, Zicheng Liu, Yumao Lu, and Lijuan Wang. Crossing the format boundary of text and boxes: Towards unified vision-language modeling. *CoRR*, abs/2111.12085, 2021. 15

[54] 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. In *AAAI*, 2021. 1, 2

[55] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *European Conference on Computer Vision*, pages 69–85. Springer, 2016. 14

[56] Yan Zeng, Xinsong Zhang, and Hang Li. Multi-grained vision language pre-training: Aligning texts with visual concepts. *ArXiv*, abs/2111.08276, 2021. 6, 7, 12, 15

[57] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020. 3

[58] P. Zhang, X. Li, X. Hu, J. Yang, L. Zhang, L. Wang, Y. Choi, and J. Gao. Vinvl: Making visual representations matter in vision-language models. 2021. 6, 7, 14

[59] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuan-Jing Huang. Extractive summarization as text matching. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6197–6208, 2020. 3, 4