SPOTLIGHT: MOBILE UI UNDERSTANDING USING VISION-LANGUAGE MODELS WITH A FOCUS

Anonymous authors
Paper under double-blind review

ABSTRACT

Mobile UI understanding is important for enabling various interaction tasks such as UI automation and accessibility. Previous mobile UI modeling often depends on the view hierarchy information of a screen, which directly provides the structural data of the UI, with the hope to bypass challenging tasks of visual modeling from screen pixels. However, view hierarchy is not always available, and is often corrupted with missing object descriptions or misaligned bounding box positions. As a result, although using view hierarchy offers some short-term gains, it may ultimately hinder the applicability and performance of the model. In this paper, we propose *Spotlight*, a vision-only approach for mobile UI understanding. Specifically, we enhance a vision-language model that only takes the screenshot of the UI and a region of interest on the screen—the *focus*—as the input. This general architecture is easily scalable and capable of performing a range of UI modeling tasks. Our experiments show that our model obtains SoTA results on several representative UI tasks and outperforms previous methods that use both screenshots and view hierarchies as input. Furthermore, we explore the multi-task learning and few-shot prompting capacity of the proposed models, demonstrating promising results in the multi-task learning direction.

1 INTRODUCTION

Computational understanding of mobile user interfaces (UI) is a crucial step for achieving intelligent UI behaviors such as UI automation, and addressing interaction scenarios such as accessibility. Recently, mobile UI understanding has attracted numerous research interests. Previous works have proposed various UI modeling tasks and datasets, including widget captioning (Li et al., 2020c), screen summarization (Wang et al., 2021), command grounding (Li et al., 2020a; Bai et al., 2021; Burns et al., 2022) and other tasks (Li et al., 2022; He et al., 2020) on the mobile screen. Many of these works focus on bridging natural language and graphical user interfaces, which have shown the potential for enabling language-based interaction.

A mobile UI screen can come with a view hierarchy—a structural representation of the screen—in addition to the screenshot image. Using view hierarchy as input allows a model to directly acquire detailed information of UI objects such as their types, text content and positions on the screen, bypassing challenging visual modeling tasks such as inferring object information from screenshots (Li et al., 2021; Zhang et al., 2021). Previous works have shown the benefit of using view hierarchy in UI modeling in several tasks. For example, models using view hierarchy have achieved better performance than the vision-only variants in UI captioning tasks (Li et al., 2020c; Wang et al., 2021).

However, recent work has revealed that mobile UI view hierarchies constantly have inaccurate information about the UI screen, e.g., missing object text (Ross et al., 2018; 2017) and misaligned bounding boxes (Li et al., 2022). Such inaccurate information in the input can seriously hinder models in realizing their full potential in UI modeling. Although recent work has proposed methods for repairing view hierarchies (Li et al., 2022), substantial effort is still needed to robustly denoise raw view hierarchies. On top of these, UI screen data does not always have view hierarchies available in them, such as mobile UI images crawled from the web. Fetching view hierarchy at runtime in a mobile environment also imposes additional system constraints for the applicability of models that rely on view hierarchies.

In this paper, we investigate the direction of visual-only input by using UI screenshots, without using view hierarchies as input, for UI modeling tasks. We observe that many UI modeling tasks essentially aim to learn a mapping between the UI objects and text. As a result, vision-language models, a class of models that encode visual (and language) modalities and decode text answers, become a natural choice for the model architecture. Although previous works show that vision-only models generally perform worse than the models using both visual and view hierarchy input (Li et al., 2020c; Wang et al., 2021), we believe that visual language models offer two unique opportunities: 1) the simple architecture enables a model easily scalable, and 2) many heterogeneous tasks can be universally represented by the two core modalities of vision and language. These advantages have been evidenced by the recent successes of the vision-language models (Chen et al., 2022; Alayrac et al., 2022; Yu et al., 2022; Wang et al., 2022).

In contrast to previous visual-language tasks in the general domain, which usually use an entire image as input, UI modeling tasks are often concerned with a specific object or area on the screen. This requires a vision-language model to be able to focus on the object or area of interest. Thus, we propose *Spotlight*[1], which enhances a vision-language model to generate text responses with respect to a focus object or region to support various UI modeling tasks (see Figure 1). In our experiments, we initialize Spotlight by leveraging pretrained large ViT (Dosovitskiy et al., 2021) and T5 (Raffel et al., 2019) checkpoints. We then pretrain Spotlight with an unlabeled dataset consisting of about 2.5 million mobile UI screens, which is followed by one of the three modeling strategies: single-task finetuning, multi-task finetuning or few-shot learning.

Our main contributions are three-fold. First, we propose a novel vision-language model architecture that is capable of finetuning, multi-task learning and few-shot learning for mobile UI tasks. The model can easily scale and generalize to other tasks without architectural changes. The model advances the art of pixel-only UI understanding and avoids the need to use view hierarchies as input that has many drawbacks in practice. Secondly, we develop a method for creating a large-scale pretraining dataset from automatically collected mobile screens. This pretraining dataset is crucial for the vision-language model to learn the prior knowledge of the unique domain of mobile screens. Finally, we conduct extensive experiments over the proposed model, including focus region representations and modeling strategies. Our experiments show that the proposed models obtain new SoTA performance in both single-task and multi-task finetuning for the three tasks, including widget captioning, screen summarization and command grounding. We also examine the feasibility of the proposed model for few-shot prompting.

2 Related Work

UI modeling problems have drawn widespread interest from researchers in both the ML and HCI fields (Li et al., 2020c; Wang et al., 2021; Li et al., 2020a; Burns et al., 2022; Bai et al., 2021; Zhang et al., 2021; Wu et al., 2021; He et al., 2020). With the overarching goal of enabling intelligent UIs and addressing mobile accessibility, previous works have proposed a rich set of mobile UI modeling tasks, including tasks, datasets and benchmarks. Widget captioning (Li et al., 2020c) aims to generate natural language description for UI objects on the screen, which can be used accessibility features such as the TalkBack[2] screen reader to improve user experience for vision-impaired users. Screen2Words expands UI captioning by proposing the task for summarizing the entire screen (Wang et al., 2021). Command grounding maps a natural language command to a UI object on the screen, via single-step or multi-step interactions (Li et al., 2020a; Burns et al., 2022; Bai et al., 2021). These previous works used both the screenshot and the view hierarchy of a screen as input in their models, and revealed that multimodal input improve the performance over vision-only variants (Li et al., 2020c; Wang et al., 2021). Prior works have also borrowed language modeling techniques such as BERT (Devlin et al., 2019) for representation learning to facilitate downstream tasks (Bai et al., 2021; He et al., 2020). Recently, Li et al. (2021) investigated the feasibility of multi-task modeling for achieving a range of UI tasks simultaneously. To combat the lack of accurate view hierarchy, prior works have investigated model-based approaches for repairing noisy view hierarchies (Li et al., 2022), or UI object detection over screenshots to derive information such as object attributes and structure information (Zhang et al., 2021; Wu et al., 2021). In this work, we investigate vision-

[1]The name draws the analogy that a spotlight illuminates a target region.

[2]`https://support.google.com/accessibility/android/answer/6283677?hl=en`

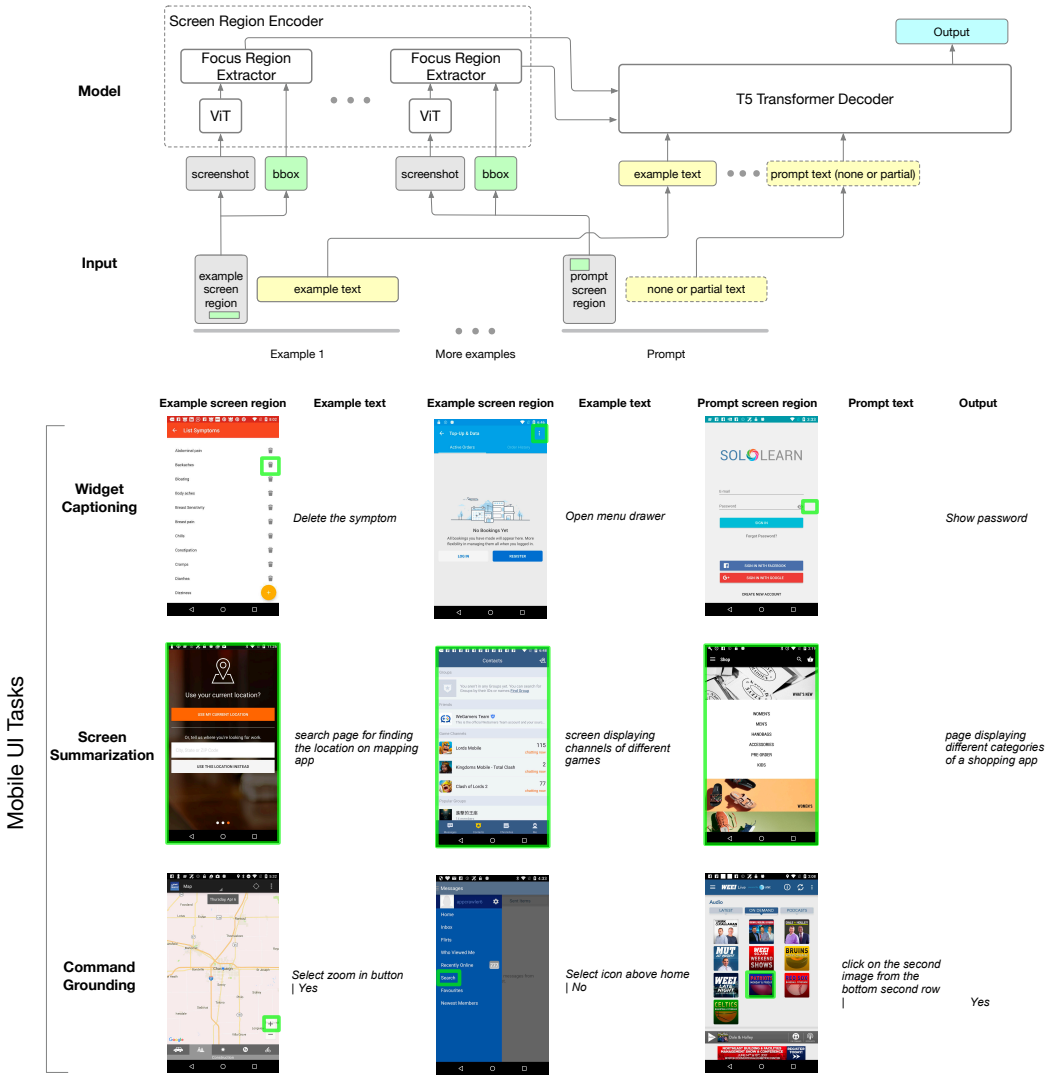


Figure 1: An illustration of the model architecture and UI task examples. ViT (Dosovitskiy et al., 2021) and T5 (Raffel et al., 2019) are initialized with checkpoints pretrained in the general domain of natural image and language.

input-only approaches for three representative UI modeling tasks that are established by prior works: widget captioning, screen summarization and command grounding.

Many of these mobile UI tasks are concerned with specific objects or areas on the UI screen. Previous works have employed different methods to allow a model to focus on an object or region of interest on the screen. Li et al. (2020c); Wang et al. (2021); Bai et al. (2021) used the screenshot pixels *cropped* for individual objects as input to the model. Li et al. (2021) used a focus map to bias the ResNet outputs to pay more attention to the pixels corresponding to an object. Schoop et al. (2022) added a fourth channel to an RGB screenshot image that indicates the target region with a binary mask—the 1s correspond to the area of the object. The cropping and bias-based approach might lose the context of the object, although providing a direct localization of the object. While adding a fourth channel also keeps the information of the entire image, it is incompatible with existing vision models checkpoints that are pretrained on 3-channel RGB images. In this work, we investigate two distinct methods for region representations.

Recently, rapid progress has been seen for the large language and vision models, such as GPT-3 (Brown et al., 2020), T5 (Raffel et al., 2019), PaLM (Chowdhery et al., 2022), CLIP (Radford

et al., 2021), DALL-E (Ramesh et al., 2022) and Imagen Saharia et al. (2022). These models exploited large unlabeled data crawled from Web and demonstrate impressive finetuning and few-shot learning capacities. Vision-language models that encode images and decode text have leveraged the large pretrained vision and language components and observed significant performance gain, e.g., Flamingo (Alayrac et al., 2022) and PaLI (Chen et al., 2022). Specifically, Flamingo showed that the few-shot prompting ability of the large language model can generalize to multimodal cases using multiple pairs of image and text as input. We adapt this model architecture to explore few-shot and multi-task learning for mobile UI tasks, along with finetuning.

3 DATA

To introduce the problem space, we elaborate on the UI modeling tasks that we address and discuss the datasets we use for our investigation.

3.1 PRETRAINING DATASET

We use a large-scale dataset that consists of 2.69 million mobile UI screenshots with paired view hierarchies. The dataset was collected internally by crawling various mobile apps. As mentioned in Introduction (Section 1), many of the UI tasks are learning a mapping between a UI object and a natural language description, we propose to pretrain the models to decode text-based attributes of individual objects in view hierarchies, with only the screenshot image as the input.

Similar to previous work (Li et al., 2020c), we only use leaf objects as they are more likely to represent a functionality of the UI. For each object, we use three text-based attributes from a view hierarchy node: `text`, `content description`, and `resource id`. These attributes are showed to be important features for the models using view hierarchy as inputs (Li et al., 2020c; Wang et al., 2021). Note that we derive the output labels from view hierarchies but not use view hierarchy information as input.

To compensate for the missing text in view hierarchies, we further run OCR on the screenshot to extract additional text that might be present on the screen. Recognized single-line text with 80% or higher confidence are used. To improve the data quality, we preprocess the dataset to remove URLs, Unicode, single-character and non-alphabetical strings. We also remove text that rarely occurs, i.e., fewer than 5 times, or consists of common or generic phrases (e.g., *"text view"*) that do not convey useful information about the object. In our pretraining, we randomly sample one out of the four text properties as the corresponding text label of a target object. Table 1 summarizes the statistics of the pretraining dataset.

Screen/View Hiearchy	Objects	Text	Content Desc	Resource id	OCR Text
2.5M	51.6M	12.1M	2.2M	13.9M	29.1M

Table 1: Statistics of the pretraining dataset.

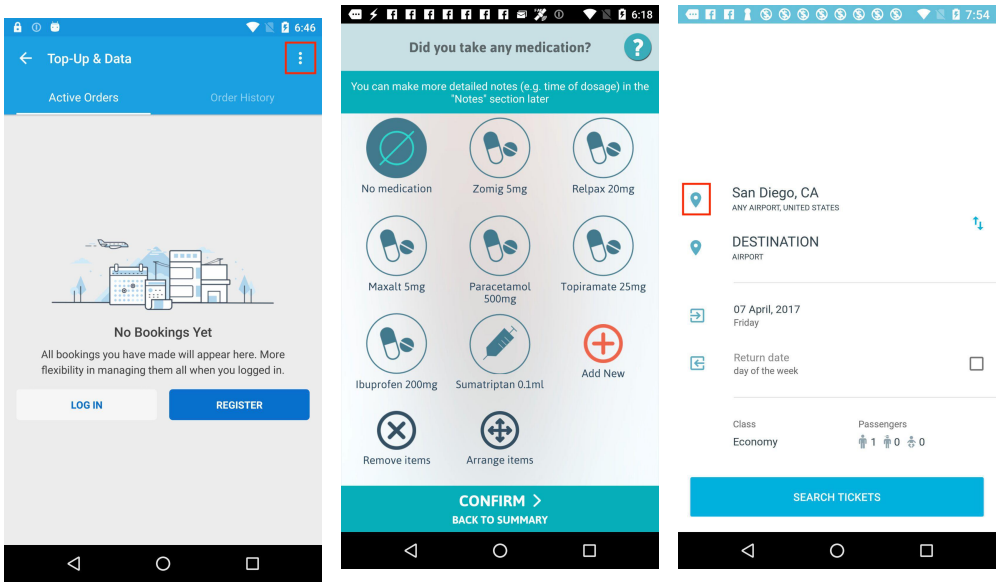
3.2 UI TASK DATASETS

From various UI modeling tasks, we select three representative tasks to evaluate our models (see Figure 1). They are listed below.

- Widget Captioning (Li et al., 2020c): given an object on the screen, predict a natural language description for its functionalities.
- Screen Summarization (Wang et al., 2021): given a mobile UI screen, predict a high-level summary for its contents and functionalities.
- Command Grounding (Li et al., 2021): given a natural language command, find the UI object on the screen that matches the intent of the command.

These three tasks require models to learn the mapping between an UI object and a natural language description or command in both directions, i.e., $object \rightarrow language$ and $language \rightarrow object$.

Furthermore, it is essential for the model to be able to both zoom in to focus on a small region and zoom out to grasp the full screen for screen summarization at the same time. The statistics of the three datasets can be found in the previous papers. We use the same dataset splits for comparison. Figure 2 shows an example of each task.



(a) Widget Captioning: *open drawer menu*

(b) Screen Summarization: *multiple options of medicine in health app*

(c) Command Grounding: *click on location icon left to san diego ca*

Figure 2: Examples of the three UI tasks. For Widget Captioning and Command Grounding, the target object is highlighted for illustration purposes.

4 MODELS

Our models are designed based on the general encoder-decoder architecture that has been widely used for visual language modeling (Chen et al., 2022; Alayrac et al., 2022; Wang et al., 2022). We employ ViT, a popular choice, for encoding images (screenshots). In contrast to previous visual language models, there is an important difference in Spotlight. UI modeling tasks are often concerned with a specific region on the screen. This unique aspect of UI modeling requires a model to be able to concentrate on a specific region. To this end, we investigate two methods for realizing the Focus Region Extractor illustrated in Figure 1 that computes region representation.

Region-of-Interest (ROI) Pooling: ROI aligned is introduced previously for obtaining the feature map of a specific region on an image (He et al., 2017). This offers a parameter-free approach for average or max pooling the feature map of an image over a specific region. The method takes care of low spatial resolutions in the latent feature space using interpolation. In our work, the bounding box of an object is used to directly *crop* the ViT outputs (image encodings) corresponding to a target object, and the cropped encodings are then averaged spatially to result in a fixed-length vector for the object. This method provides a direct localization of the object in the encoded visual signals. The method embodies a *soft* cropping method as the resulted representation carries the information beyond the target area, including the contextual information on the screen. This is in contrast to *hard* cropping methods that crop the raw pixels of the target area and lose the access to the screen context.

Region Summarizer: Previous works have used random vectors as input in Transformer models to query into visual modalities for specific tasks. For example, DeTR (Carion et al., 2020) used random queries for retrieving visual objects on an image. Flamingo (Alayrac et al., 2022) summarizes image encodings by using random vectors to resample the image encodings. Inspired by these methods, we

propose to use the bounding box of an object as attention queries to the ViT outputs. During training, the models will learn to summarize a target region indicated by a bounding box query to focus on the object area. We hypothesize that the region summarizer, as a Transformer model, has the capacity to learn how to represent a specific region while taking into account of the screen context. As we will discuss later, our experiments show that Region Summarizer offers superior performance over the ROI pooling. The pseudo code of the Region Summarizer is shown in Listing 1.

```
def region_summarizer(
    bbox,          # The bounding box coordinates in [batch, 4, 1].
    vit_outputs,   # The ViT outputs in [batch, vit_seq_len, hidden_dim].
    num_layers     # Number of cross-attention layers.
  ):
  # First, map bounding box coordinates to dense vectors.
  bbox = Dense(bbox)

  # Compute positional embeddings for the four coordinates.
  positions = [[1, 2, 3, 4]]
  pe = SinusoidalEmbed(positions)

  # Combine coordinates with the positional embeddings.
  bbox = bbox + pe
  bbox = Dense(bbox)

  # Perform cross-attention to extract region information.
  x = bbox
  kv = concat([vit_outputs, bbox], axis=1)
  for i in range(num_layers):
    x = x + CrossAttention_i(q=x, kv)
    x = x + Dense_i(x)
  return x
```

Listing 1: Region Summarizer Psuedo Code

We use an auto-regressive decoder to generate the text tokens. For the pretraining task, widget captioning and screen summary, it is straightforward that the decoder will attend to the regional screen representations and decode the captions or summaries auto-regressively. For command grounding, we treat it as a binary classification problem. Given a user command, we inspect each object on the screen by asking the model to predict either $True$ or $False$, by appending either $True$ or $False$ to the end of the command prompt. Note that in this case only the last token is to be predicted and the command acts as a prefix and is encoded via self-attention in the decoder (see Figure 1).

4.1 Pretraining

During pretraining, we train the model to decode the text of an object, and the text is randomly sampled from the four text elements of an object: text, content description, resource id and OCR text. We found that the object text from view hierarchy contains very frequent words or phrases, e.g., *submit* and *input password*. To counter the imbalance, we subsample the frequent phrases as in word2vec (Mikolov et al., 2013) with $t = 1e - 5$. As we want to explore few-shot prompting in addition to finetuning, we sample a random number of examples $n \in [1, 5]$ and pad the input sequence to 5 examples, similar to Flamingo (Alayrac et al., 2022). The ViT encoder encodes each screenshot independently, while the decoder decodes the full sequence of the text of all the pairs, concatenated by BOC (Beginning of the chunk) and EOC (End of the chunk) tokens. In this way, we hope the model can generalize to use sequences of multiple screenshot-object pairs and potentially learn the few-shot prompting capacity for UI tasks as the previous work has shown for tasks in the general domain.

4.2 Finetuning, Multi-task Learning and Few-shot Prompting

We investigate three modeling strategies in our work. For finetuning, we finetune the pretrained model (Section 4.1) for each of the three UI tasks separately. We use one screenshot-object pair in each example for both finetuning training and evaluation. For multi-task learning, we finetune the

pretrained model (Section 4.1) for the three tasks simultaneously. We sample $n = 2$ screenshot-object pairs for each example from each task during training. Therefore, during evaluation we can prompt the model with an screenshot-object pair from the training set of the target task, which the prompt acts as the task indicator. For few-shot prompting, we use the pretrained model directly for evaluation, i.e., no further change for the model parameters is needed. We experiment with $n \in [0, 4, 8, 16, 32]$ shots to see if the model performance can improve as more example prompts or context are given. Note that we only need to vary the number of screenshot-object pairs in each example without modifying the model architecture for the three use cases.

5 Experiments and Results

5.1 Model Variants

Similar to previous vision-language models (Chen et al., 2022; Alayrac et al., 2022), we use existing checkpoints of ViT and T5 models to initialize corresponding components in our models in all the experiments. We experiment with two different sized ViT models, B/16 and L/16, pretrained on 300M natural images (Dosovitskiy et al., 2021)[3]. ViT L/16 has a similar parameter size as the mT5 base model (Xue et al., 2020), for which we also reuse the pretrained checkpoint [4]. In addition, we explore the two approaches of extracting region representation of the focus area from the ViT outputs, namely ROI pooling and region summarizer (Section 4). Therefore, we have four model variants in terms of size and region representation, listed in Table 2. The models with region summarizer use four cross-attention layers to extract information from the ViT outputs, thus having slightly more parameters than the ones with ROI pooling that is parameter free. The hyperparameters of the models can be found in Appendix A.

T5	ViT	Focus	ViT Enc Layers	Dec Layers	# Parameters
base	B/16	ROI pooling	12	12	592M
base	L/16	ROI pooling	24	12	815M
base	B/16	Region summarizer	12	12	620M
base	L/16	Region summarizer	24	12	843M

Table 2: The model variants in our experiments.

5.2 Finetuning

We finetune the pretrained model for each of the downstream tasks. As shown in Table 3, we can see that our vision-only approaches obtain state-of-the-art performance for all the three tasks, outperforming previous baselines that use both screenshots and view hierarchies as input. Training our models for each of these downstream tasks from scratch, i.e., still initialized with ViT and T5 checkpoints but without our pretraining, did not lead to good performance. This shows that our pretraining dataset and approach is essential for warming up the model and familiarizing it with the UI tasks. For focus region representation, Region Summarizer performs generally better than ROI.

5.3 Multi-task Learning

For multi-task learning, we let a Spotlight model learn the three tasks simultaneously, which is more challenging than finetuning for each specific task. To avoid information leak, we need to ensure that the test dataset of each task does not overlap the joint training set for multi-task learning, i.e., no screens from the same mobile app should appear in both the training and test splits. To do so, we re-split the widget captioning and command grounding datasets according to the app-wise splits of screen summarization (Wang et al., 2021), which leads to a larger test set. We sample training batches from the widget captioning, screen summary and command grounding tasks with the weights of [2, 1, 17], which are tuned on the development set. During evaluation, we applied the same model onto the three tasks with a prefix screen-object pair sampled from each training set to

[3] https://github.com/google-research/vision_transformer

[4] https://github.com/google-research/multilingual-t5

	Model	Captioning	Summarization	Grounding
Baselines	Widget Caption (Li et al., 2020b)	97.0	-	-
	Screen2Words (Wang et al., 2021)	-	61.3	-
	VUT (Li et al., 2021)	99.3	65.6	82.1
ROI	B/16 from scratch	72.2	39.0	22.1
	L/16 from scratch	77.5	42.4	21.7
	B/16 with pretraining	103.1	68.9	86.5
	L/16 with pretraining	113.2	83.8	87.5
Region Summarizer	B/16 from scratch	50.2	43.5	39.0
	L/16 from scratch	49.7	55.2	53.6
	B/16 with pretraining	108.1	76.9	87.5
	L/16 with pretraining	**121.6**	**90.7**	**89.0**

Table 3: Finetuning Performance. CIDEr scores are reported for widget captioning and screen summarization while accuracy for command grounding.

prompt the model to perform different tasks—as a task indicator. As shown in Table 4, multi-task Spotlight models still outperform previous models, and perform on par with task-specific models (Table 3) on grounding tasks. Although multi-task performance on widget captioning and screen summarization is lower than finetunning performance, they are still significantly better than the published benchmark results.

	ViT	Widget Captioning	Screen Summarization	Grounding
ROI	B/16	100.1	65.4	85.9
	L/16	112.0	81.6	87.8
Region Summarizer	B/16	100.9	65.9	87.7
	L/16	114.6	81.1	89.2

Table 4: The performance of Spotlight models for multi-task learning.

5.4 Few-shot Learning

Few-shot learning is an important capability demonstrated by large vision and language models (Alayrac et al., 2022; Brown et al., 2020), which a large model can be adapted for a specific downstream task solely using examples or prompts. We investigate the few-shot learning for adapting our pretrained model for the widget captioning task. We report the results of using $n \in \{0, 4, 8, 16, 32\}$ shots in Table 5. When there is 0 shot, the pretrained model is directly tested on the widget captioning task, without examples. Although at most five screen-object pairs are used in each example during pretraining, our models can generalize to more shots during inference. We observed marginal improvement over the zero-shot condition when more shots of the task are given. We did not include results for screen summarization and command grounding as the models did not show meaningful few-shot performance. We conjecture that the screen summarization and command grounding are too far from the pretraining data distribution, as we use a much smaller and more specific dataset than the ones in general domain (Alayrac et al., 2022). Using a larger and more diverse pretraining dataset might improve the few-shot learning performance for UI tasks.

Focus	ViT	0	4	8	16	32
ROI	B/16	38.8	39.0	39.2	40.0	40.3
	L/16	49.3	49.2	49.3	49.3	49.5
Region Summarizer	B/16	37.1	36.7	36.9	37.5	37.4
	L/16	48.8	49.6	49.9	49.6	49.8

Table 5: The performance of few-shot prompting on the Widget Captioning task.

6 DISCUSSIONS

Spotlight models advance the vision-only models for UI tasks by using a large-scale pretraining dataset and a general architecture that can support different use cases. As shown in Section 5, pretrained models significantly outperformed the models trained from scratch. There are various ways of how to use the unlabeled dataset to pretrain UI models (He et al., 2020; Bai et al., 2021). In our early exploration, in addition to pretraining the model to decode object text from view hierarchies, we experiment with decoding a serialized view hierarchy directly. However, this approach led to poor pretraining performance, likely due to the issue that the serialized view hierarchy contains a lot of noise (Li et al., 2022), which hinder the model performance. Nevertheless, the structural information buried in the view hierarchy is valuable and how to use it for better pretraining remains an interesting direction for future work.

The proposed vision-only models outperformed previous models using both images and view hierarchies as input. As previous models depend on the text information in the view hierarchy as input features, missing text can significantly affect their performance. We compare the performance of Spotlight with the benchmark model of the widget captioning task (Li et al., 2020b) on screens with a varying degree of missing text (Table 6). For this analysis, we sort all the screens in the test dataset based on the percentage of UI elements on the screen that have missing text, e.g., 10% means the top 10% screens that have the highest missing text percentages, i.e., the least text availability, and 50% means the half of the test screens that have the most missing text. We can see that the Spotlight model performs stably across screens with missing text in their view hierarchies, because Spotlight does not use view hierarchy as input. However, the performance for the benchmark widget captioning model decreased significantly as the text information become less available in the view hierarchy, and view hierarchy text information serves as an important feature for the previous model.

Model	% of Test Screens with Least Text					
	10%	20%	30%	40%	50%	All
Widget Captioning (Li et al., 2020c)	84.6	81.3	86.9	86.0	88.0	97.0
Spolight L/16 Region Summarizer	118.0	107.0	117.5	113.1	117.2	121.6
Δ	33.4	25.7	30.6	25.1	29.2	24.7

Table 6: Widget Captioning CIRDEr scores on screens with varying availability of text in view hierarchy. Δ shows the performance improvement by Spotlight.

Compared to the recent large vision-language model efforts (Alayrac et al., 2022; Chen et al., 2022) that use models of size $O(10B)$ parameters, we have only investigated the use of a smaller versions of ViT and T5 components which have up to 843M parameters. Our experiments consistently show the trend that a larger model yields better performance, which promises that simply scaling up the model sizes would potentially lead to an even larger performance gain. Our model architecture is simple and can easily adopt larger ViT and T5 checkpoints. Our model architecture is generic, which takes a screenshot and a region of interest as well as text prompts, and outputs text responses. Our model can be easily applied to more UI tasks, e.g., icon classification (Deka et al., 2017), tappability prediction (Schoop et al., 2022) and multi-step grounding (Li et al., 2020a; Burns et al., 2022) as these tasks can be formulated as a text decoding task.

7 CONCLUSIONS

We presented Spotlight, a vision-only approach for mobile UI understanding, which alleviates the need to use view hierarchy. We design Spotlight based on a general vision-language model architecture and enable the model to focus on a region of interest on the screen. This architecture is easy to scale and can benefit from the success of existing large vision-language models pretrained for the general domain. Spotlight is able to fulfill a range of UI modeling tasks. We investigated various region representations and model sizes. We report our findings on a variety of modeling strategies and use cases, including finetuning, multi-tasking learning and few-shot prompting. Our experiments show that Spotlight obtains SoTA results on several representative UI tasks and outperforms previous works that use both screens and view hierarchies.

REFERENCES

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, Andrew 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, 2022. URL https://arxiv.org/abs/2204.14198.

Chongyang Bai, Xiaoxue Zang, Ying Xu, Srinivas Sunkara, Abhinav Rastogi, Jindong Chen, and Blaise Agüera y Arcas. Uibert: Learning generic multimodal representations for UI understanding. In Zhi-Hua Zhou (ed.), *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pp. 1705–1712. ijcai.org, 2021. doi: 10.24963/ijcai.2021/235. URL https://doi.org/10.24963/ijcai.2021/235.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Andrea Burns, Deniz Arsan, Sanjna Agrawal, Ranjitha Kumar, Kate Saenko, and Bryan A. Plummer. A dataset for interactive vision-language navigation with unknown command feasibility, 2022. URL https://arxiv.org/abs/2202.02312.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers, 2020. URL https://arxiv.org/abs/2005.12872.

Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Nan Ding, Keran Rong, Hassan Akbari, Gaurav Mishra, Linting Xue, Ashish Thapliyal, James Bradbury, Weicheng Kuo, Mojtaba Seyedhosseini, Chao Jia, Burcu Karagol Ayan, Carlos Riquelme, Andreas Steiner, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. Pali: A jointly-scaled multilingual language-image model, 2022. URL https://arxiv.org/abs/2209.06794.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022. URL https://arxiv.org/abs/2204.02311.

Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pp. 845–854, New York, NY, USA, 2017. Association

for Computing Machinery. ISBN 9781450349819. doi: 10.1145/3126594.3126651. URL https://doi.org/10.1145/3126594.3126651.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://www.aclweb.org/anthology/N19-1423.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 2980–2988. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.322. URL https://doi.org/10.1109/ICCV.2017.322.

Zecheng He, Srinivas Sunkara, Xiaoxue Zang, Ying Xu, Lijuan Liu, Nevan Wichers, Gabriel Schubiner, Ruby B. Lee, and Jindong Chen. Actionbert: Leveraging user actions for semantic understanding of user interfaces. *CoRR*, abs/2012.12350, 2020. URL https://arxiv.org/abs/2012.12350.

Gang Li, Gilles Baechler, Manuel Tragut, and Yang Li. Learning to denoise raw mobile ui layouts for improving datasets at scale, 2022. URL https://arxiv.org/abs/2201.04100.

Yang Li, Jiacong He, Xin Zhou, Yuan Zhang, and Jason Baldridge. Mapping natural language instructions to mobile UI action sequences. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8198–8210, Online, July 2020a. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.729. URL https://www.aclweb.org/anthology/2020.acl-main.729.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. Widget captioning: Generating natural language description for mobile user interface elements. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5495–5510, Online, November 2020b. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.443. URL https://www.aclweb.org/anthology/2020.emnlp-main.443.

Yang Li, Gang Li, Luheng He, Jingjie Zheng, Hong Li, and Zhiwei Guan. Widget captioning: Generating natural language description for mobile user interface elements, 2020c.

Yang Li, Gang Li, Xin Zhou, Mostafa Dehghani, and Alexey A. Gritsenko. VUT: versatile UI transformer for multi-modal multi-task user interface modeling. *CoRR*, abs/2112.05692, 2021. URL https://arxiv.org/abs/2112.05692.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality, 2013. URL https://arxiv.org/abs/1310.4546.

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, 2021. URL https://arxiv.org/abs/2103.00020.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683, 2019. URL http://arxiv.org/abs/1910.10683.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O. Wobbrock. Epidemiology as a framework for large-scale mobile application accessibility assessment. In *Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '17, pp. 2–11, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4926-0. doi: 10.1145/3132525.3132547. URL http://doi.acm.org/10.1145/3132525.3132547.

Anne Spencer Ross, Xiaoyi Zhang, James Fogarty, and Jacob O. Wobbrock. Examining image-based button labeling for accessibility in android apps through large-scale analysis. In *Proceedings of the 20th International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '18, pp. 119–130, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5650-3. doi: 10.1145/3234695.3236364. URL http://doi.acm.org/10.1145/3234695.3236364.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S. Sara Mahdavi, Rapha Gontijo Lopes, Tim Salimans, Jonathan Ho, David J Fleet, and Mohammad Norouzi. Photorealistic text-to-image diffusion models with deep language understanding, 2022. URL https://arxiv.org/abs/2205.11487.

Eldon Schoop, Xin Zhou, Gang Li, Zhourong Chen, Björn Hartmann, and Yang Li. Predicting and explaining mobile ui tappability with vision modeling and saliency analysis, 2022. URL https://arxiv.org/abs/2204.02448.

Bryan Wang, Gang Li, Xin Zhou, Zhourong Chen, Tovi Grossman, and Yang Li. Screen2words: Automatic mobile UI summarization with multimodal learning. *UIST'21*, 2021. URL https://arxiv.org/abs/2108.03353.

Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, and Furu Wei. Image as a foreign language: Beit pretraining for all vision and vision-language tasks, 2022. URL https://arxiv.org/abs/2208.10442.

Jason Wu, Xiaoyi Zhang, Jeff Nichols, and Jeffrey P. Bigham. Screen parsing: Towards reverse engineering of ui models from screenshots. 2021. URL https://arxiv.org/pdf/2109.08763.pdf.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. *CoRR*, abs/2010.11934, 2020. URL https://arxiv.org/abs/2010.11934.

Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models, 2022. URL https://arxiv.org/abs/2205.01917.

Xiaoyi Zhang, Lilian de Greef, Amanda Swearngin, Samuel White, Kyle Murray, Lisa Yu, Qi Shan, Jeffrey Nichols, Jason Wu, Chris Fleizach, Aaron Everitt, and Jeffrey P Bigham. Screen recognition: Creating accessibility metadata for mobile applications from pixels. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi: 10.1145/3411764.3445186. URL https://doi.org/10.1145/3411764.3445186.

A HYPERPARAMETERS

We use image resolution $[540, 360]$ for all our experiments. The mobile screenshots are resized with aspect ratio preserved and padded to $[540, 360]$ if necessary. The paddings in the pixels are ignored by the ViT encoder using masking. The object bounding boxes are normalized to $[0, 1]$. We use 0.1 dropout in all our models. For all the training experiments, we use batch size 128 and

linear warmup/rsqrt decay for learning rate. During inference decoding, beam size 5 is used. The maximum text decoding length is 30 for each image-object pair.

Pretraining: We pretrain the four Spotlight model variants to 100k steps with learning rate 6e-3 for models with ROI pooling and learning rate 9e-3 for models with region summarizer, with linear warmup to 10k steps. Each training example consists of $n \in [1, 5]$ image-object pairs sampled from the pretraining dataset. The model with ViT B/16 is trained using 64 Google Cloud TPU v3 cores for 28 hours. The model with ViT L/16 is trained using 128 Google Cloud TPU v3 cores for 43 hours.

From Scratch: We train the three UI tasks from scratch, while still initialize ViT and T5 with their pretrained checkpoints, using 20k steps. We use learning rate 1e-4 for the widget captioning and command grounding task and 1e-3 for screen summarization.

Finetuning: For all the finetuning tasks, we use learning rate 1e-4. We use 10k training steps for widget captioning and screen summarization and 20k steps for command grounding.

Multi-task Learning: We use learning rate 2e-4 and train the multi-task models to 50k steps. The sampling weights are $[2, 1, 17]$ for the widget captioning, screen summary and command grounding tasks. We use $n = 2$ image-object pairs in each example during training. During evaluation, an image-object pair from the training set of the target task is used to prompt the model to perform the task.