UI Description: Predicting Natural/Human Language summaries of mobile user interfaces by finetuning a pre-trained state-of-the-art image captioning model.

MBAJWA MICHAEL (STUDENT ID: 0210152807)

ABSTRACT 2

INTRODUCTION 2

RELATED WORK 3

SCREEN2WORDS: AUTOMATIC MOBILE UI SUMMARIZATION WITH MULTIMODAL LEARNING 3
UNBLIND YOUR APPS: PREDICTING NATURAL-LANGUAGE LABELS FOR MOBILE GUI COMPONENTS BY DEEP LEARNING 4
WIDGET CAPTIONING: GENERATING NATURAL LANGUAGE DESCRIPTION FOR MOBILE USER INTERFACE ELEMENTS 4
BLIP: BOOTSTRAPPING LANGUAGE-IMAGE PRE-TRAINING FOR UNIFIED VISION-LANGUAGE UNDERSTANDING AND GENERATION 4
MODELLING MOBILE INTERFACE TAPPABILITY USING CROWDSOURCING AND DEEP LEARNING 5

MATERIALS AND METHODS 5

DATA 5
Train, validation and test datasets *6*
METHOD 7
BLIP: Bootstrapping Language-Image Pre-training *7*
Finetuning the BLIP model *8*

RESULTS AND DISCUSSION 10

RESULTS 10
Evaluation Metrics *10*
DISCUSSION 16

CONCLUSION 17

REFERENCES 18

Abstract

Despite the global adoption of mobile devices, there exist no programs that can generate helpful summaries of the user interfaces of mobile applications. Clear summaries of mobile user interfaces that clearly define a user interface's major components and usefulness currently have a wide range of uses-cases. This project aims to train an already pre-trained state-of-the-art image captioning model to generate human language descriptions of mobile user interfaces. We use the Enrico dataset, a curated version of Rico. The training dataset amounts to 1297 image-caption pairs, the testing to 189 image-caption pairs and the validation to 166 image-caption pairs. The performance of the finetuned model on this testing data will be measured using commonly used metrics for evaluating image descriptions.

Introduction

Mobile devices have become an indispensable part of daily life in recent years. The mobile user interface (UI) screens on these devices have many graphical components that users can manipulate to perform various tasks.

A concise summary of these mobile UI screens is helpful in several language-based scenarios. Screen summaries, for example, can improve natural language search for UI screens across many databases, potentially expanding the scope of partnerships in the field of mobile app development. UI screen summaries can be used to automatically tag UI elements (identifying individual elements such as buttons or text fields). Another essential application of UI screen summaries is automatically announcing the created screen summaries to screen readers.

Despite the importance of concise human language summaries of mobile UI screens, summaries of entire phone screens are nonexistent on mobile devices. The current work on describing entire UI screens provides brief descriptions that do not adequately describe the full functionality of the mobile screens [1], and our goal is to build upon the limitations of previous work. To this end, this project focuses on UI screen description, a task that involves predicting Natural/Human Language summaries of mobile user interfaces using by finetuning a pre-trained image captioning model.

We hypothesize that we can generate high-quality natural language descriptions of UIs' by finetuning an existing pre-trained state-of-the-art image captioning model using a small dataset.

This project seeks to make the following valuable contributions;

- We propose the automatic generation of descriptions for UI, a task that predicts clear summaries of mobile UI. These generated summaries would clearly define their significant components and usefulness. We believe this work would be applicable in a wide range of language-based application scenarios.
- We develop a deep learning model by adapting an existing pre-trained image captioning model [2] for our target task, UI description generation. We believe this work will serve as a benchmark for future research in the following;
  - using existing deep learning models to solve a wide variety of image-text problems.
  - Connecting user interfaces and human language.

Related work

This project builds upon several areas of existing work, including; deep learning for understanding user interfaces of mobile applications, transfer learning and datasets of mobile interfaces.

Screen2Words: Automatic Mobile UI Summarization with Multimodal Learning

Screen2Words [1] uses image, text, and structural information from user interfaces to train a deep learning model that automatically summarizes mobile user interface screens in natural language. A significant contribution of this work is; combining the Transformer encoder-decoder model [3] with ResNet [4]. The work uses multiple data modalities, including text, image, and structures of a UI, instead of just unimodal data, leading to superior summarization accuracy.

This project would address various limitations of this work. The major limitation of this work is that the trained model often generates generic and incorrect summaries. Our work aims to solve this problem by finetuning an existing state-of-the-art model using a dataset of informative summaries of mobile user interfaces.

Unblind Your Apps: Predicting Natural-Language Labels for Mobile GUI Components by Deep Learning

Unblind Yout Apps [5] automatically predicts the labels of user interface components to enhance the accessibility of mobile applications. The invention of LabelDroid, a deep-learning model based on CNN and transformer encoder-decoder, is the work's main contribution. The researchers trained LabelDroid with images from widely-used mobile applications, and it can automatically predict the labels of image-based buttons on user interfaces. Another contribution of this work is open-sourcing a dataset of human-generated descriptions of user interface components.

Widget Captioning: Generating Natural Language Description for Mobile User Interface Elements

Widget captioning [6] proposes an approach for automatically generating natural language captions for elements in mobile user interfaces. This study makes two significant contributions: it creates and makes a dataset for widget captioning research and proposes a method for deep learning model configuration for captioning. This strategy, like [1], uses multimodal input data to improve the performance of deep learning models.

[5] and [6] have limitations in that they focus on individual user interface components and do not provide summaries that can represent the complete mobile user interface screen. This is one of the limitations that our project aims to overcome.

BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation

BLIP [2] is a cutting-edge Vision-Language Pre-training (VLP) framework that excels in various vision-language understanding tasks. The significant contribution of this work is the

BLIP framework, a Multimodal mixture of Encoder-Decoder (MED), which can be seamlessly applicable for transfer learning.
One drawback of this study is that BLIP is not trained on photos of mobile user interfaces with captions created by humans. It is trained on noisy image-text pairs gathered from the internet. If the model is directly used to generate summaries of mobile user interface displays, its performance will be poor. We will build on this work by finetuning the BLIP model using our user interface captioned dataset.

Modelling Mobile Interface Tappability Using Crowdsourcing and Deep Learning

Tapshoe [7] presents an approach for large-scale modelling of the tappability of mobile interfaces. This paper contributes a deep neural network model that learns human perceived tappability of interface elements from various interface parameters, such as the spatial, semantic, and visual aspects of an interface element's screen.
While this research has made considerable progress in utilizing deep learning to interpret mobile screens, one fundamental limitation is that it does not provide summaries for mobile user interfaces. Our research builds on this work by finetuning a model that can predict natural language summaries of entire user interfaces.

Materials and methods

DATA

We used the Enrico dataset [8], a curated version of Rico [9]. Enrico comprises 1460 mobile user interfaces categorized according to a design taxonomy of 20 UI layout categories, e.g. news, login, settings, tutorial, etc. Each user interface comprises a screenshot with associated metadata such as annotations of element types, visual and structural data, and interactive design properties.

A crowdsourcing study was conducted to collect human descriptions of the Enrico UIs. 146 participants (64 female, 80 male, 2 other) aged 18--76 (M=30 years) were recruited via Prolific[1]

[1] https://www.prolific.co

a crowsourcing platform with a large pool of workers, mostly from US and UK. All participants had a worker approval rate of 95% and could complete the study only once. Participants were proficient English speakers, had a normal or corrected-to-normal vision, and had not been diagnosed as having any reading disorder. The study took 15 minutes on average, and participants were paid 3.28 US dollars, corresponding to an hourly wage of $11.31.

The study was made available via a web-based application. Participants had to caption the screenshots in the test partition of the Enrico dataset, which were randomly sampled. Each participant described 15 UI screenshots on average.

For this work, we briefly analyzed the caption lengths generated by the human workers. For the entire 2121 captions, we had an average length of 83 characters. The minimum number of characters amongst all descriptions was three, and the maximum number of characters was 353. From the brief analysis, we noticed that the human captions were very diverse, and in some cases, they were either too brief or not informative. To remove unwanted captions, we manually went through the 2121 captions using Microsoft Excel, and we deleted the captions/summaries that fell into one or more of these categories;

- The summary is a description of how the participant feels about the screen.
- The summary only contains a description of the UI element's colour, shape and name.
- The summary contains only the name of the app.
- The summary contains less than five words.

After manually cleaning noisy data, we arrived at 1652 human-generated captions for 269 unique screenshots. The maximum number of captions generated per image is 15, while the minimum is 2. We had an average of 7 captions per image on our dataset.

Train, validation and test datasets

A training, validation and test dataset split was done to properly assess the finetuned network's generalization capability. The 269 images with descriptions from the enrico dataset [8] were split into 212 images for training, 30 for testing and 27 for validation, following a random distribution. When separating the dataset according to this split, the training dataset amounted to 1297 image-caption pairs, the testing 189 and the validation 166.

Method

BLIP: Bootstrapping Language-Image Pre-training

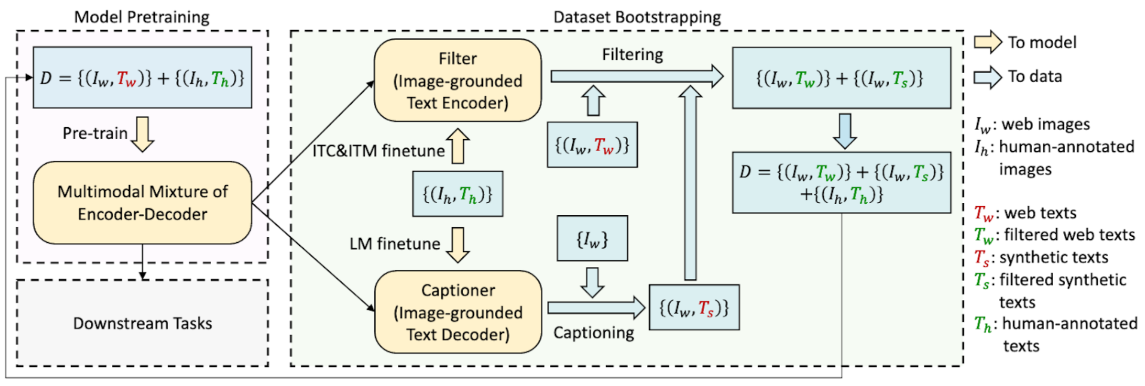
The deep learning model to be finetuned in this project is the Bootstrapping Language-Image Pre-training (BLIP) model [2], a model architecture that achieves state-of-the-art performance on both understanding and generation tasks. The BLIP framework is a Multimodal mixture of Encoder-Decoder (MED). We selected the BLIP architecture for the following reasons;

- It is a flexible model architecture capable of adapting to a broader range of downstream tasks, unlike other existing methods.
- It integrates an innovative data bootstrapping method, i.e. Captioning and Filtering (CapFilt), that makes learning from noisy web data more efficient.
- End-to-end finetuning is very simple as all requirements are readily available[2].
- It is proven to achieve state-of-the-art performance on image-text retrieval and image captioning.

BLIP [2] utilizes a multi-task model, a multimodal mixture of encoder-decoder. This gives the model the flexibility to perform downstream tasks such as image-text retrieval, image captioning, visual question answering and more. A Visual Transform (ViT) is used as the image encoder.

During pre-training, the BLIP model optimized three objectives concurrently. These objectives are Image-Text Contrastive Loss (ITC), Image-Text Matching Loss (ITM) and Language Modeling Loss (LM). Figure 1 shows the architecture of the BLIP framework.

[2] https://github.com/salesforce/BLIP



Model Pretraining
Dataset Bootstrapping
$D = \{(I_w, T_w)\} + \{(I_h, T_h)\}$
Pre-train
Multimodal Mixture of Encoder-Decoder
Downstream Tasks
Filter (Image-grounded Text Encoder)
ITC&ITM finetune
$\{(I_h, T_h)\}$
LM finetune
Captioner (Image-grounded Text Decoder)
Filtering
$\{(I_w, T_w)\}$
$\{I_w\}$
Captioning
$\{(I_w, T_s)\}$
$\{(I_w, T_w)\} + \{(I_w, T_s)\}$
$D = \{(I_w, T_w)\} + \{(I_w, T_s)\} + \{(I_h, T_h)\}$
To model
To data
I_w: web images
I_h: human-annotated images
T_w: web texts
T_w: filtered web texts
T_s: synthetic texts
T_s: filtered synthetic texts
T_h: human-annotated texts

Figure 1: Learning framework of BLIP. We introduce a captioner to produce synthetic captions for web images and a filter to remove noisy image-text pairs. The captioner and filter are initialized from the same pre-trained model and finetuned individually on a small-scale human-annotated dataset. The bootstrapped dataset is used to pre-train a new model. [3]

For this project, we finetuned the four different pretrained checkpoints of BLIP. These checkpoints are;

- BLIP with ViT-B (14M): BLIP pretrained with 14 million images and a Base Visual Transformer encoder.
- BLIP with ViT-B (129M): BLIP pretrained with 129 million images and a Base Visual Transformer encoder.
- BLIP with ViT-B (129M) and CapFilt-L: BLIP pretrained with 129 million images and a Base Visual Transformer encoder. This model also utilized a large CapFilt for data bootstrapping.
- BLIP with ViT-L (129M): BLIP was pretrained with 129 million images and a Large Visual Transformer encoder.

Finetuning the BLIP model

Finetuning a deep learning network is based on the concept of transfer learning. Transfer learning is a machine learning technique in which knowledge gained during training in a broadly-domain problem is applied to another related task or specific domain [10]. Finetuned learning experiments require a bit of learning, but they are often much faster than learning from scratch [11].

The pre-trained BLIP models were finetuned with the Enrico [8] dataset to generate descriptive captions of mobile user interfaces automatically. The Enrico dataset with just 1297 image captions is insufficient to train deep networks from scratch hence the use of the pre-trained weights from the BLIP model. For the UI captioning downstream task, we *finetune a specific path of the pre-trained model to achieve our objective*.

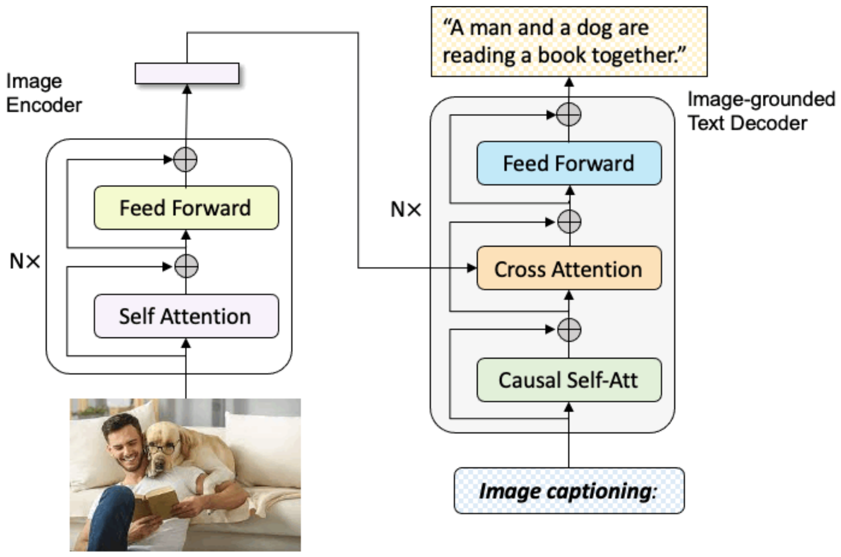


Figure 2 shows the BLIP model's finetuned path [2].

The developers of BLIP have made the code and models available on Github to facilitate finetuning. We cloned the git repository and adapted the available code for finetuning the BLIP model with the coco dataset [12] to suit the Enrico dataset. We first had to prepare our custom training, test and validation dataset following the format of the coco [12] captioning file. The BLIP model is loaded with pretrained weights and is finetuned with Language Modelling loss. The image-grounded text decoder, which tries to produce textual descriptions given an image, is activated by LM loss.

Our models are implemented in PyTorch [13] and fine-tuned on 2 NVIDIA Tesla V100 SXM2 16GB GPUs on the UL HPC platform [14]. We used a batch size of 16, and we finetuned with 25 and 100 epochs. We use AdamW [15] optimizer with a weight decay of 0.05. We take random image crops of image resolution 384 × 384 during finetuning, leading to faster inference time. Additionally, the model was finetuned using an initial learning rate of 1e-5. Finetuned model returns image captions of a maximum length of 45 words (we observed that maximum length above this threshold leads to word repetition and subsequently poor performance) and a minimum length of 10 words. Unlike the pretrained model, we did not include a prompt at the beginning of each caption.

We report the performance of our finetuned model based on metrics commonly used in machine translation and image captioning tasks: Bleu [16], Cider [17], Rouge-L [18], Meteor [19] and Spice [20].

Results and Discussion

Results

We evaluate our model's overall performance in this section using the metrics Bleu [16], Cider [17], Rouge-L [18], Meteor [19] and Spice [20] that are frequently used for machine translation and image captioning jobs.

Evaluation Metrics

Bleu [16] measures how similar machine-generated translations and human-created reference translations (i.e., ground truth) are. It is the product of n-gram precision and brevity penalty. We measure the bleu value by setting n as 1, 2, 3, 4, represented as Bleu_1, Bleu_2, Bleu_3, and Bleu_4.

METEOR [19] (Metric for Evaluation of Translation with Explicit ORdering) is another metric employed for machine translation evaluation. It is suggested to address various Bleu [16] drawbacks, such as the disregard for synonyms and recall ratio. Before calculating a weighted F-score with an alignment fragmentation penalty, the algorithm searches for exact, stem, synonym, and paraphrase matches between n-grams to align sentences.

ROUGE [18] (Recall-Oriented Understudy for Gisting Evaluation). We utilize ROUGE-L, which determines the similarity between the predicted sentence and the reference based on the longest common subsequence (short for LCS).

CIDEr [17] (Consensus-Based Image Description Evaluation) applies term frequency-inverse document frequency (tf-idf) weights to n-grams in the generated and reference sentences, which are then compared by summing their cosine similarity across n-grams.

SPICE [20] (Semantic Propositional Image CaptionEvaluation) compares semantic propositional content. It more accurately captures human judgments over model-generated captions than the above-mentioned automatic metrics.

All of these metrics have real-valued outputs within range (0,1). The higher the metric score, the more similar the machine-generated content description is to the ground truth. The score of these measures is 1 if the predicted results exactly match the ground truth. These metrics are computed using coco-caption code [12]. The test dataset, which the model had not encountered during training, was used to calculate all of the scores.

Table 1. Finetuned model performance based on common metrics

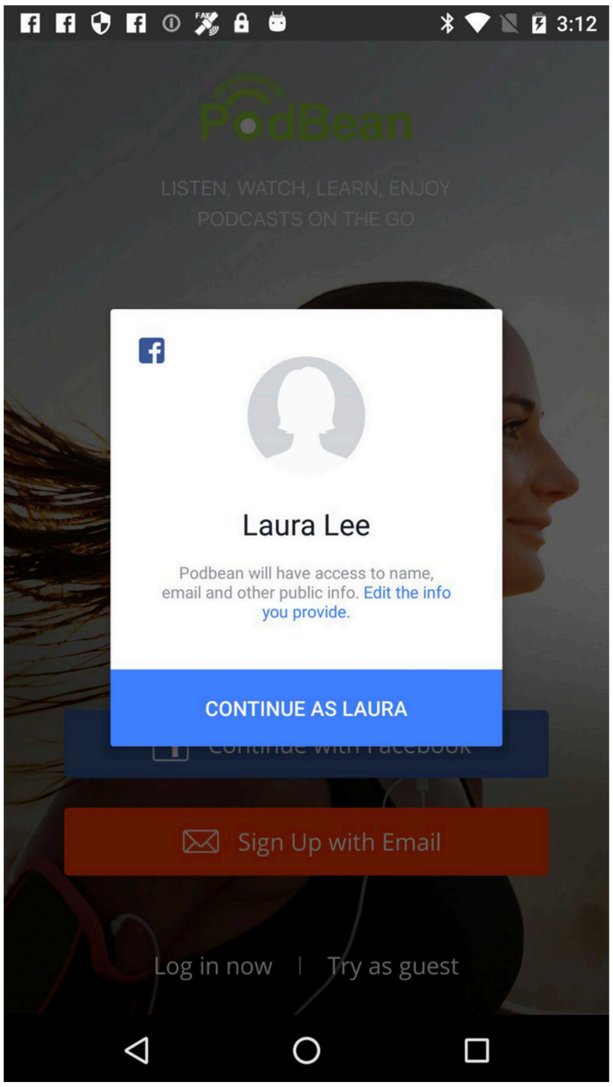
The scores are real-valued with a range (of 0,1). A higher score means better model performance.

Finetuned Model	**Bleu_1**	**Bleu_2**	**Bleu_3**	**Bleu_4**	**METEOR**	**ROUGE_L**	**CIDEr**	**SPICE**
BLIP with ViT-B (14M) **25 Epochs**	0.596	0.357	0.2305	0.1449	0.155	0.348	0.113	0.116
BLIP with ViT-B (129M) **25 Epochs**	0.640	0.4332	0.2979	0.1946	0.1658	0.3834	0.3267	0.1246
BLIP with ViT-B (129M) and CapFilt-L **25 Epochs**	0.7268	0.4791	0.3324	0.2267	0.1778	0.3863	0.3408	0.1219
BLIP with ViT-B (129M) and CapFilt-L **100 Epochs**	0.6593	0.4575	0.3130	0.2096	0.1659	0.3872	0.3144	0.1106
BLIP with ViT-L (129M) **25 Epochs**	0.7131	0.4653	0.3169	0.2073	0.1674	0.3799	0.3318	0.1109
BLIP with ViT-L (129M) **100 Epochs**	0.7278	0.5075	0.3615	0.2272	0.1701	0.4104	0.436	0.1068

The performance of our finetuned models based on the metrics as mentioned earlier are presented in table 1. We can observe that the finetuned models *BLIP with ViT-B (129M) and CapFilt-L (trained with 25 epochs)* and *BLIP with ViT-L (129M) (trained with 100 epochs)* outperform the other finetuned models across all measured metrics. The finetuned *BLIP with ViT-B (129M) and CapFilt-L* seems to perform better on 25 epochs compared to 100 epochs, while the reverse is the case for the finetuned *BLIP with ViT-L (129M).*

Qualitative performance compared to baseline BLIP models.

We believe a better approach to understanding the performance of our finetuned model is comparing the summaries generated by these finetuned models to those generated by the baseline model i.e the pretrained model that are not finetuned. To do this, we sampled random test images and generated captions for them using both the finetuned and the baseline models. The comparisons of the resulting outputs are displayed below:



Podbean will have access to name, email and other public info. Edit the info you provide.

BLIP with ViT-B (14M)
Baseline model: a cell phone with a picture of a woman on the screen.
Finetuned model: *a log in page for an app.*

BLIP with ViT-B (129M)
Baseline model: a woman's profile on her phone.
Finetuned model: a login screen.

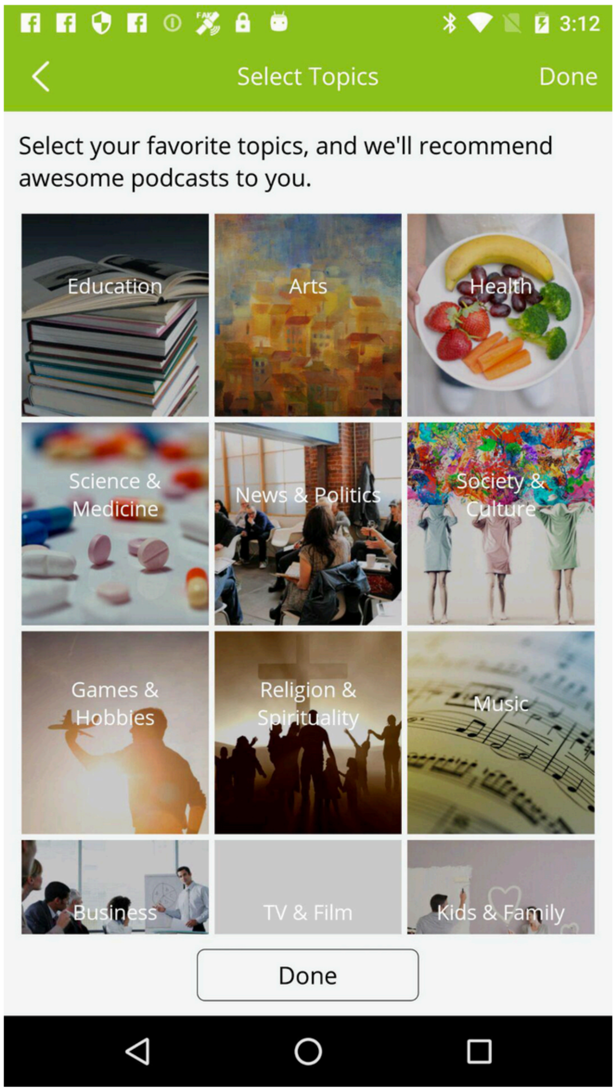
BLIP with ViT-B (129M) and CapFilt-L - 25 epochs
Baseline model: a woman's profile on a phone.
Finetuned model: a login page for an app.

BLIP with ViT-B (129M) and CapFilt-L - 100 epochs
Baseline model: a woman's profile on a phone.
Finetuned model: a facebook login page.

BLIP with ViT-L (129M) - 25 epochs
Baseline model: a woman's profile on the app.
Finetuned model: a login page for an app.

BLIP with ViT-L (129M) - 100 epochs
Baseline model: a woman's profile on the app.
Finetuned model: a login page for a dating app.

Figure 3: An image from the Enrico dataset with captions generated by finetuned and baseline models.


3:12

Select Topics

Done

Select your favorite topics, and we'll recommend awesome podcasts to you.

BLIP with ViT-B (14M)
Baseline model: a cell phone with a screenshots of people sitting around a table.
Finetuned model: a gallery app with images and text at the top.

BLIP with ViT-B (129M)
Baseline model: a group of people on a cell.
Finetuned model: a gallery screen with various pictures.

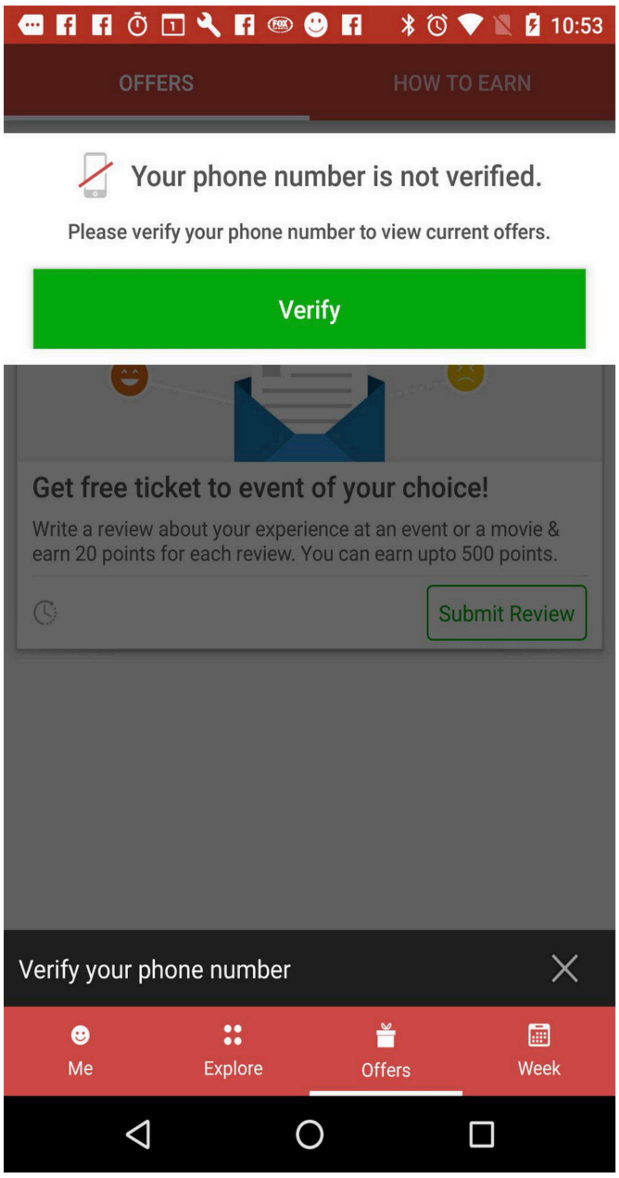
BLIP with ViT-B (129M) and CapFilt-L - 25 epochs
Baseline model:a group of people in a room.
Finetuned model: a gallery app with images and text at the top.

BLIP with ViT-B (129M) and CapFilt-L - 100 epochs
Baseline model:a group of people in a room.
Finetuned model: a gallery, with options to select your favorite pictures.

BLIP with ViT-L (129M) - 25 epochs
Baseline model:a group of people on a cell phone.
Finetuned model:a gallery app.

BLIP with ViT-L (129M) - 100 epochs
Baseline model: a group of people on a cell phone.
Finetuned model: a screenshot of an app.

Figure 4: An image from the Enrico dataset with captions generated by finetuned and baseline models.


10:53

OFFERS

HOW TO EARN

Your phone number is not verified.

Please verify your phone number to view current offers.

Verify

Get free ticket to event of your choice!

Write a review about your experience at an event or a movie & earn 20 points for each review. You can earn upto 500 points.

Submit Review

Verify your phone number

BLIP with ViT-B (14M)
Baseline model: a cell screen with a text message on it.
Finetuned model*: a social media app with text at the top and an arrow at the middle to.*

BLIP with ViT-B (129M)
Baseline model: a phone with a text message.
Finetuned model: a phone number verification app.

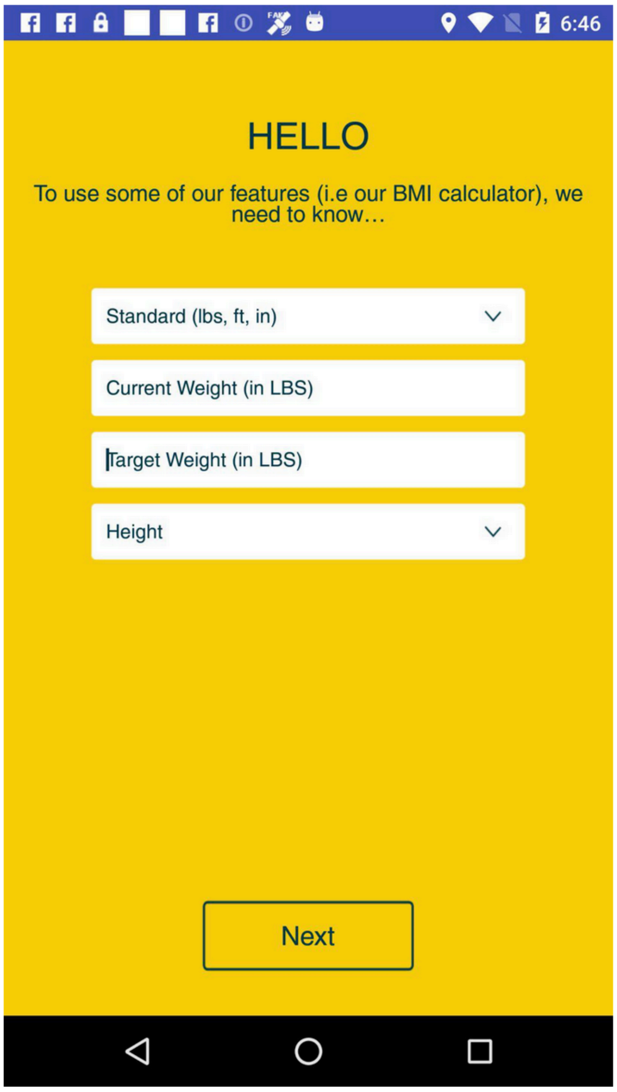
BLIP with ViT-B (129M) and CapFilt-L - 25 epochs
Baseline model: a phone with the text 'your phone is not verified'.
Finetuned model: an app that allows you to verify your phone number.

BLIP with ViT-B (129M) and CapFilt-L - 100 epochs
Baseline model: a phone with the text 'your phone is not verified'.
Finetuned model: a phone number on the screen.

BLIP with ViT-L (129M) - 25 epochs
Baseline model: a phone showing the phone number and how to use it.
Finetuned model: a phone number verification app.

BLIP with ViT-L (129M) - 100 epochs
Baseline model: a phone showing the phone number and how to use it.
Finetuned model: an app that allows you to verify your phone number.

Figure 5: An image from the Enrico dataset with captions generated by finetuned and baseline models.


HELLO

To use some of our features (i.e our BMI calculator), we need to know…

BLIP with ViT-B (14M)
Baseline model: a phone with the text ` ` i love you" on it.
Finetuned model*: a log in page of an app.*

BLIP with ViT-B (129M)
Baseline model: a cell phone with the help button highlighted.
Finetuned model: an app that allows you to set your weight in a range of weights.

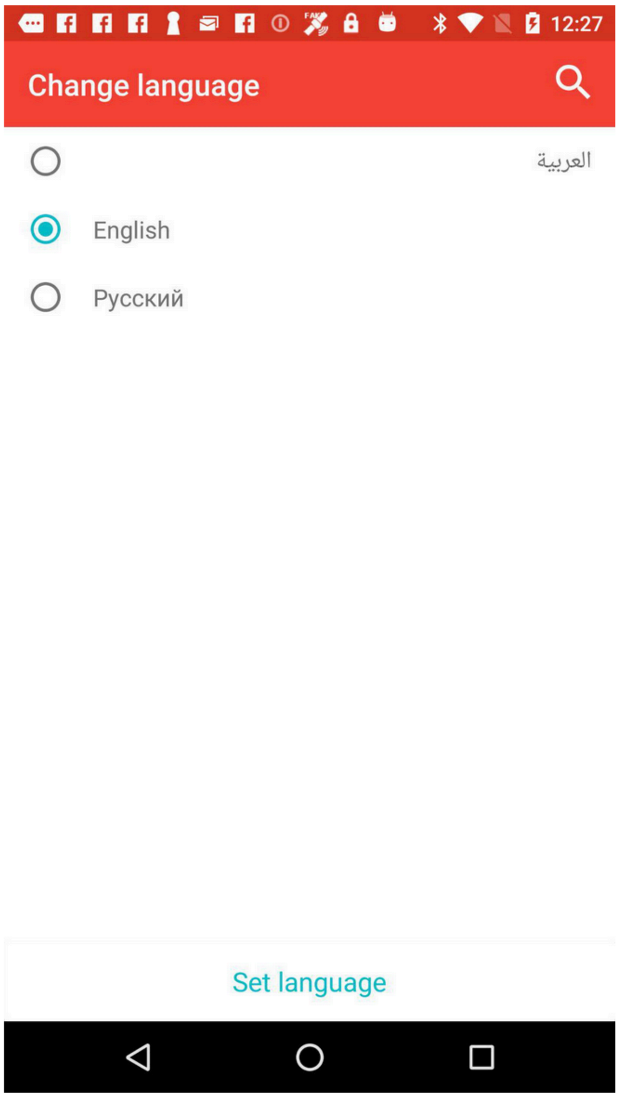
BLIP with ViT-B (129M) and CapFilt-L - 25 epochs
Baseline model: a cell with the text hello on it.
Finetuned model: an app that allows you to sign up or sign up.

BLIP with ViT-B (129M) and CapFilt-L - 100 epochs
Baseline model: a cell with the text hello on it.
Finetuned model: a login page of an app.

BLIP with ViT-L (129M) - 25 epochs
Baseline model: a cell phone with the text hello on it.
Finetuned model: an app that allows you to log in to your health calculator.

BLIP with ViT-L (129M) - 100 epochs
Baseline model: a cell phone with the text hello on it.
Finetuned model: a screenshot of a health app.

Figure 6: An image from the Enrico dataset with captions generated by finetuned and baseline models.



BLIP with ViT-B (14M)
Baseline model: a phone screen showing the settings settings.
Finetuned model*: a location app.*

BLIP with ViT-B (129M)
Baseline model: a phone with the language highlighted.
Finetuned model: a language learning app.

BLIP with ViT-B (129M) and CapFilt-L - 25 epochs
Baseline model: a phone with the language in arabic.
Finetuned model: an app that allows you to sign up or sign up.

BLIP with ViT-B (129M) and CapFilt-L - 100 epochs
Baseline model: a phone with the language in arabic.
Finetuned model: a language selection screen.

BLIP with ViT-L (129M) - 25 epochs
Baseline model: the language app on a smartphone.
Finetuned model: a language learning app.

BLIP with ViT-L (129M) - 100 epochs
Baseline model: the language app on a smartphone.
Finetuned model: a language app.

Figure 7: An image from the Enrico dataset with captions generated by finetuned and baseline models.

Figures 3-7 show example UI screen summarizations results by our different finetuned models and the baseline pretrained models. We can see from the results that all the finetuned models can generate coherent, understandable summaries, although not consistently accurate. Compared to the baseline models, we can see that the finetuned models most often generate more accurate summaries of the UI screenshots.

Discussion

There are still several limitations that future work could address. We found that no single finetuned model successfully predicted UI summaries for all input photos. The classification of user interfaces into UI layout categories—such as news, login, settings, etc.—and subsequent analysis of the effectiveness of each finetuned model within each category—will be the next route for future work. This will enable us to identify patterns, i.e., we can see if some models are more effective when predicting summaries for images belonging to a well-defined group.

While we studied the performance of the finetuned model using metrics like Bleu, METEOR, SPICE, etc., these do not provide a comprehensive picture of how well the finetuned model compares to the baseline models. In later studies, we believe using human employees to evaluate and score the quality of the summaries generated manually might give a better insight into how the finetuned models perform.

Due to time and resource constraints, we could not tune the various hyperparameters of our model other than the number of epochs. In future work, one could explore tuning other parameters such as the weight decay, learning rate, size of image cropping used, and maximum length of words generated.

Conclusion

We have presented a method for predicting Natural/Human Language summaries of mobile user interfaces (UI) by finetuning a pre-trained state-of-the-art image captioning model. We provided examples of situations where our work would be advantageous, such as improving UI screen search through natural language. We used the Enrico dataset, a curated version of Rico. We finetuned and evaluated 4 pretrained versions of the BLIP model architecture based on a cleaned version of the Enrico dataset. Comparing the results of the finetuned models with the baseline pretrained models show that our finetuned model better predicts summaries for user interfaces.
In our future work, we intend to learn more about how each finetuned model performs across various UI categories. We also hope to properly access the performance of the finetuned models by asking humans to score the predictions.

References

[1] G. L. X. Z. Z. C. T. G. Y. L. Bryan Wang, "Screen2Words: Automatic Mobile UI Summarization with Multimodal Learning," arXiv:2108.03353 [cs.HC], 2021.

[2] D. L. C. X. S. H. Junnan Li, "BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation," arXiv:2201.12086, cs.CV, 2021.

[3] N. S. N. P. J. U. L. J. A. N. G. L. K. I. P. Ashish Vaswani, "Attention Is All You Need," arXiv:1706.03762 [cs.CL], 2017.

[4] X. Z. S. R. J. S. Kaiming He, "Deep Residual Learning for Image Recognition," arXiv:1512.03385 [cs.CV], 2015.

[5] C. C. Z. X. X. X. L. Z. G. L. J. W. Jieshan Chen, "Unblind Your Apps: Predicting Natural-Language Labels for Mobile GUI Components by Deep Learning," arXiv:2003.00380 [cs.HC], 2020.

[6] G. L. L. H. J. Z. H. L. Z. G. Yang Li, "Widget Captioning: Generating Natural Language Description for Mobile User Interface Elements," arXiv:2010.04295 [cs.LG], 2020.

[7] Y. L. Amanda Swearngin, "Modeling Mobile Interface Tappability Using Crowdsourcing and Deep Learning," arXiv:1902.11247 [cs.HC], 2019.

[8] A. H. A. O. Luis A. Leiva, "Enrico: A High-quality Dataset for Topic Modeling of Mobile UI Designs," in *Proceedings of ACM Conf. on Human-computer interaction with mobile devices and services (MobileHCI)*, 2020.

[9] B. a. H. Z. a. F. C. a. H. J. a. A. D. a. L. Y. a. N. J. a. K. R. Deka, "Rico: A Mobile App Dataset for Building Data-Driven Design Applications," *Proceedings of Annual Symp. on User Interface Software and Technology (UIST),* pp. 845-854, 2017.

[10] Z. Q. K. D. D. X. Y. Z. H. Z. S. M. I. H. X. F. I. a. Q. H. Fuzhen Zhuang, "A Comprehensive Survey on Transfer Learning," arXiv:1911.02685 [cs.LG], 2020.

[11] D. H. M. S. S.P. Mohanty, "Using deep learning for image-based plant disease detection," *Front. Plant Sci.,* vol. 7, pp. 1-7, September 2016.

[12] T. M. M. B. S. J. H. J. P. P. R. D. D.´. P. a. Z. C. L. Lin, "Microsoft COCO: common objects in context," *n Fleet, D. J., Pajdla, T., Schiele, B., and Tuytelaars, T. (eds.), ECCV,* vol. 8693, p. 740–755, 2014.

[13] A. G. S. M. F. L. A. B. J. C. G. K. T. L. Z. G. N. A. L. e. a. Paszke, "Pytorch: An imperative style, high-performance deep learning library," *NeurIPS, 32,* p. 8026–8037, 2019.

[14] S. V. a. P. B. a. H. C. a. F. Georgatos, "VBCG_HPCS14: Management of an Academic HPC Cluster, The UL Experience," no. Proc. of the 2014 Intl. Conf. on High Performance Computing \& Simulation (HPCS 2014), pp. 959-967, July 2014.

[15] I. a. H. F. Loshchilov, "Decoupled weight decay regularization," arXiv preprint arXiv:1711.05101, 2017.

[16] S. R. T. W. a. W.-J. Z. Kishore Papineni, "BLEU: a method for automatic evaluation of machine translation," in *In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL ’02*, USA, 2002.

[17] C. Z. a. D. P. Ramakrishna Vedantam, "Cider: Consensus-based image description evaluation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[18] C.-Y. L. a. F. J. Och, "Orange: A method for evaluating automatic evaluation metrics for machine translation," in *20th International Conference on Computational Linguistics, COLING ’04*, Stroudsburg, PA, USA, 2004.

[19] M. D. a. A. Lavie, "Meteor universal: Language specific translation evaluation for any target language," in *Ninth Workshop on Statistical Machine Translation*, Stroudsburg, PA, USA, 2014.

[20] B. F. M. J. a. S. G. Peter Anderson, "SPICE: semantic propositional image caption evaluation," CoRR, abs/1607.08822, 2016.