GUI-focused Overviews of Mobile Development Videos

Mohammad Alahmadi[1,2], Abdulkarim Khormi[1,3], Sonia Haiduc[1]
[1]Florida State University, Tallahassee, FL, United States
{alahmadi, khormi, shaiduc}@cs.fsu.edu
[2]University of Jeddah, Jeddah, Saudi Arabia
[3]Jazan University, Jizan, Saudi Arabia

ABSTRACT

The need for mobile applications and mobile programming is increasing due to the continuous rise in the pervasiveness of mobile devices. Developers often refer to video programming tutorials to learn more about mobile programming topics. To find the right video to watch, developers typically skim over several videos, looking at their title, description, and video content in order to determine if they are relevant to their information needs. Unfortunately, the title and description do not always provide an accurate overview, and skimming over videos is time-consuming and can lead to missing important information. We propose a novel approach that locates and extracts the GUI screens showcased in a video tutorial, then selects and displays the most representative ones to provide a GUI-focused overview of the video. We believe this overview can be used by developers as an additional source of information for determining if a video contains the information they need. To evaluate our approach, we performed an empirical study on iOS and Android programming screencasts which investigates the accuracy of our automated GUI extraction. The results reveal that our approach can detect and extract GUI screens with an accuracy of 94%.

CCS CONCEPTS

• **Software and its engineering** → *Documentation*; • **Computer vision** → *Image recognition*;

KEYWORDS

Programming video tutorials, Mobile development, Software documentation, Deep learning, Video mining

ACM Reference Format:
Mohammad Alahmadi[1,2], Abdulkarim Khormi[1,3], Sonia Haiduc[1]. 2020. GUI-focused Overviews of Mobile Development Videos. In *42nd International Conference on Software Engineering Companion (ICSE '20 Companion), October 5–11, 2020, Seoul, Republic of Korea.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3377812.3390900

1 INTRODUCTION

While developers have greatly benefited from Q&A forums such as StackOverflow, video programming tutorials are increasing in popularity due to their ability to convey information in a faster, more efficient way, while also showing programs as they are both written and executed [4]. Moreover, video tutorials are often less time consuming to produce by tutors compared to writing text-based tutorials [3]. In consequence, millions of programming videos are currently available on hosting platforms such as YouTube.

To make the process of finding the right video easier and faster, each video is usually complemented with metadata (title and description) to summarize the video content. However, previous works found that the metadata of programming screencasts are not always descriptive [7]. On the other hand, skimming over several videos to find the most relevant one takes substantial time and effort and can lead to missing useful programming screencasts[8]. We want to address these limitations and believe that offering developers more types of information about a video could help them better and faster understand what a video is about. Given the sharp increase in the demand for mobile development, in this paper we focus specifically on mobile programming video tutorials.

When developing a mobile application, the Graphical User Interface (GUI) is one of the most important aspects, as it plays a crucial role in the success of the mobile app [5]. At the same time, designing and implementing an effective GUI is one of the most challenging tasks for mobile developers [5, 6] and, therefore, one for which they are likely to seek online help. In consequence, we believe GUI information is of interest and importance to developers and therefore aim to provide GUI-focused overviews of video tutorials. These could be used as an additional source of information by developers when trying to understand what a video is about or if it is relevant to their current information need.

In this paper, we present a novel approach to locate, extract, and filter GUI screens from mobile programming videos, resulting in GUI-focused overviews of each video. To evaluate our approach, we performed an empirical study on 1,000 iOS and Android mobile programming videos. The results revealed that our approach locates and extracts GUI screens with an accuracy of 94%.

2 METHODOLOGY

Our goal is to accurately locate GUI screens embedded in mobile programming video tutorials, such that they can be extracted and displayed in an overview of the videos. To accomplish this goal, our approach utilizes an object detector to locate GUI screens in video frames, based on an approach previously used to identify the location of the code editing window within programming screencasts [1]. To train the object detector network for this task, we first need to collect videos that display GUI screens, then manually annotate the video frames with the precise position of the GUIs (*i.e.*, the coordinates of the boxes that delimit them, aka bounding boxes) and then feed this information into the network. We used YouTube[1] to manually collect 1,000 iOS and Android video tutorials. During

[1]http://www.youtube.com

ICSE '20 Companion, October 5–11, 2020, Seoul, Republic of Korea

ACM ISBN 978-1-4503-7122-3/20/05.
https://doi.org/10.1145/3377812.3390900

the data collection process, we ensured that we had a diverse set of videos such that the presented GUI screens appear in various forms across the videos (*e.g.*, different GUI emulators, different orientations - landscape or portrait, different background colors, etc.). To obtain video frames, we extracted one frame per second from each video. Then, we filtered out duplicate frames, identified based on their distinctive features, determined using the Speeded Up Robust Features (SURF) algorithm [2] from the field of computer vision. The remaining frames were manually classified into the *GUI* or *nonGUI* categories, meaning that they either contained a mobile GUI screen or not. To train an object detector to detect the location of a mobile GUI in an image, we need to annotate GUI screens in each frame with *bounding boxes* that surround them. Given the fact that a total of 14,720 frames contained GUI screens, it would require a massive amount of human effort to manually annotate each of them with bounding box information. Thus, we randomly selected 2,000 GUI and 2,000 nonGUI frames from the total, making sure that we collect frames from all videos and that the same number of GUI and nonGUI frames are extracted from each video. Then, each frame containing GUI screens was manually annotated with a bounding box that surrounded the GUI screen region. The nonGUI frames were automatically annotated by setting the bounding box to the entire frame, as we needed the CNN network to learn the features of these entire frames and distinguish them from the GUI frames.

To locate GUI screens in video frames, we used the a specific type of CNN-based object detector, namely the Faster R-CNN [10], in tandem with the Inception-Resnet V2 feature extractor [11]. The object detector was trained end-to-end using back-propagation where the weights get adjusted using Stochastic Gradient Descent (SGD) with momentum optimizer [9]. To evaluate our approach, we performed 10-fold cross-validation. For each fold, we split our data into 80% training, 10% validation, and 10% test sets. We used the validation set to avoid overfitting, and the test set to evaluate the model's accuracy. To improve our model's performance, we augmented our data by randomly flipping and scaling the GUI screens. We trained our model for a total of 4,000 iterations, until there was no significant network improvement.

After obtaining the GUI screens' locations within video frames, our approach crops them and extracts them in a set. This extracted set of GUI screens can, however, contain duplicate GUIs. To remove the duplicate GUI screens and keep only the most representative ones, we used SURF to extract the features of each GUI screen and then compute the distance between the features of each GUI pair to find a match. When a match is found, only one of the frames is kept and its duplicates are removed.

3 RESULTS

Table 1 shows the results of the automatic binary classification into GUI-containing and non-GUI containing (nonGUI) frames. Overall, our model performed extremely well, achieving both an *F-Score* and an *Accuracy* of 98% for both the GUI and nonGUI categories, as well as a *Precision* and *Recall* of up to 99%. Table 2 presents the results of localizing the GUI screens within GUI frames with respect to the ground truth bounding box. The table shows the results at different IoU (intersection-over-union between the predicted location and the ground truth location) thresholds starting from 0.70 up to 0.90 with a step size of 0.10. Our model achieved an *Average Precision* of 92% at an IoU threshold of 0.90, and an *Accuracy* of 94% at the same threshold.

Table 1: The binary classification results

Category	Precision	Recall	F-Score	Accuracy
GUI	0.99	0.98	0.98	0.98
NonGUI	0.98	0.99	0.98	0.98

Table 2: The average precision and accuracy of localizing GUI screens using 10-fold cross-validation

IoU Threshold	Average Precision	Accuracy
0.70	0.98	0.98
0.80	0.97	0.97
0.90	0.92	0.94

4 CONCLUSION

In this paper, we proposed a novel approach to locate and extract GUI screens from mobile programming tutorials, with the purpose of offering developers a GUI-focused overview of a video. We fine-tuned a CNN on frames extracted from iOS and Android programming videos and conducted an empirical evaluation to assess our approach's accuracy. The evaluation showed that our approach was able to accurately classify and locate GUIs in video frames.

5 ACKNOWLEDGEMENTS

Sonia Haiduc was supported in part by the National Science Foundation grants CCF-1846142 and CCF-1644285.

REFERENCES

[1] Mohammad Alahmadi and et al. 2020. Code Localization in Programming Screencasts. *Empirical Software Engineering* (2020), 1–37.
[2] Herbert Bay and et al. 2006. Surf: Speeded up robust features. In *European conference on computer vision*. Springer, 404–417.
[3] Laura MacLeod and et al. 2015. Code, camera, action: How software developers document and share program knowledge using YouTube. In *Proceedings of the 23rd IEEE International Conference on Program Comprehension (ICPC'15)*. Florence, Italy, 104–114.
[4] Laura MacLeod and et al. 2017. Documenting and sharing software knowledge using screencasts. *Empirical Software Engineering* 22, 3 (June 2017), 1478–1507.
[5] Kevin Moran and et al. 2018. Automated reporting of GUI design violations for mobile apps. *arXiv preprint arXiv:1802.04732* (2018).
[6] Tuan Anh Nguyen and Christoph Csallner. 2015. Reverse engineering mobile application user interfaces with remaui (t). In *Automated Software Engineering (ASE)*. IEEE, 248–259.
[7] Esteban Parra and et al. 2018. Automatic tag recommendation for software development video tutorials. In *Proceedings of the 26th ICPC*. ACM, 222–232.
[8] Luca Ponzanelli and et al. 2016. Too long; didn't watch!: Extracting relevant fragments from software development video tutorials. 261–272. https://doi.org/10.1145/2884781.2884824
[9] Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural Networks* 12, 1 (1999), 145–151.
[10] Shaoqing Ren and et al. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *arXiv:1506.01497 [cs]* (June 2015).
[11] Christian Szegedy and et al. 2016. Inception-v4, Inception-ResNet and the impact of residual connections on learning. *arXiv:1602.07261 [cs]* (Feb. 2016).