StarWars-Companion

Zhu Zengliang Liu Yanqing Tan Weipeng Wang Wenyao Yi Ruiyu

Macau University of Science and Technology, Macau

Abstract

We developed a camera-based application so that users can easily search for characters they are not familiar with, without even knowing their names and other information. Users simply aim the camera at the characters, and the application will immediately recognize who he is and show his background and images.

Background

Star Wars is a famous film/television series and was started in 1970. So far, the franchise has published 12 movies and 14 TV series, and it continues to attract more and more people. However, the entire series spans a long time period. Many non-hardcore fans sometimes have difficulty recognizing a previously introduced characters, as the character may have previously appeared in an episode more than a decade ago. In another case, newcomers usually don't start watching the series from the beginning because it is time-consuming. Then this means they won't know old characters that appeared in the previous episodes.

Related Work

To avoid developing a derivative product, we searched the github.com/app store for similar works.

1. *StarWars library*: On the Android platform, an application gives users pictures of the characters, heights, and so on. But it requires a name to search.
2. *StarWars-ML-classification*: On the PC platform, a classification program, can only provide the characters' name, and only 2 characters can be classified. This program also cannot work if multiple characters are in the same frame.
3. *The Star Wars Wiki (Wookieepedia)*: The website is full of detailed information about Star Wars.
4. APPs published by Disney and its coop-companies: These apps provide a variety of functions such as making sound effects and so on.

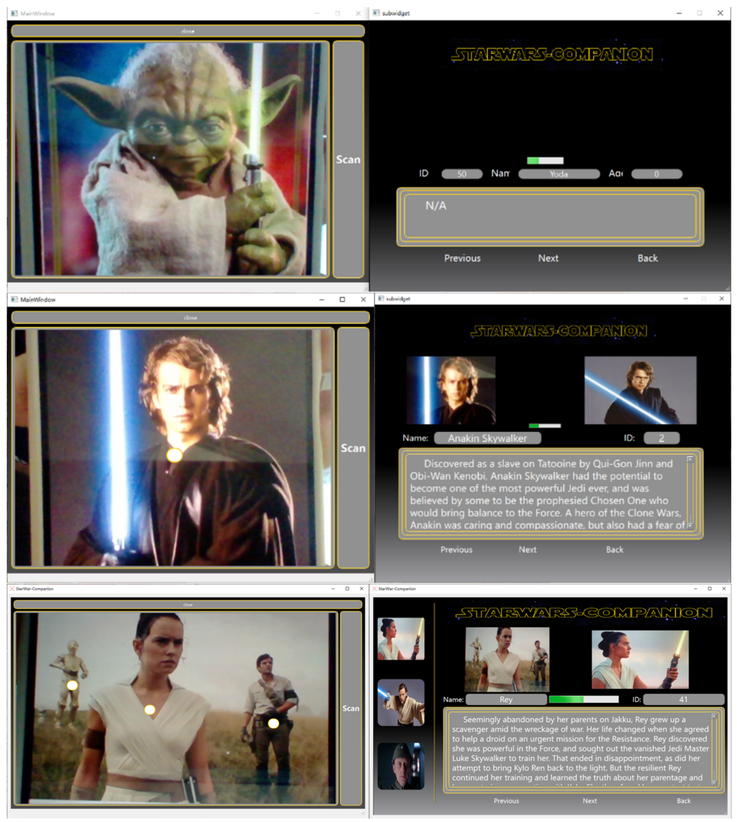
Software Architecture

Model-view-controller is the basic software design pattern we used to implement our project. To be more specific, we divided controller layer into detailed modules like camera module, database module and deep learning module. View and front-end controller are combined into GUI modules, which contains two pages, one for image capturing, another for showing information.

- **Camera module** is a sequence of functions defined to run the camera as we want it.
- **Deep learning module** provides the localization and classification functions, which will be explained in detail in *Deep Learning* section.
- **Database module** includes the database file[1] and the interfaces to query the database and return results.
- **GUI** plays a role as front-end controller and viewer. Next section will introduce GUI in detail.

User Interface

GUI have gone through 3 versions, keeping the main idea while improving information demonstration. The main idea is that the user clicks the scan button to take a photo , then the app shows the information of the character the user select. The figures on the right show three iterations of the information page. The first version only shows characters' names. The second version shows characters' images, background info., and the photo taken by users. The third version shows top-3 probable predictions for users to choose from.



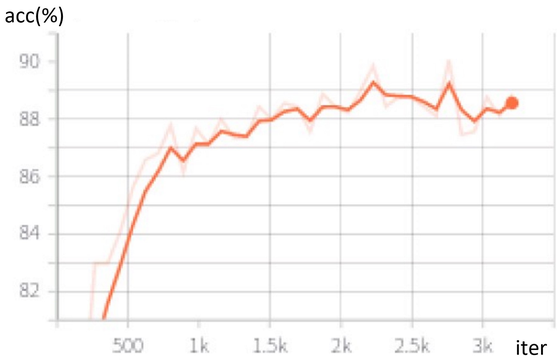
Testing

In the test section, many of the test tools were tried out, some of which did not work as well as the tutorial suggested. Finally, we decided to use Qtest for testing. Unit tests are done on each module to test whether the function of each function can be appropriately called.

Deep Learning

We used deep learning for two tasks: classification and localization.
Localization: We used a YOLOv5s model to predict a bounding box on each potential Star Wars characters in an image. YOLOv5 is a one-stage state-of-the-art detection model. In our project, we adopted an off-the-shelf one pre-trained on COCO dataset. Through experiment, we found that it generalizes well in our use case. For data preprocessing, we normalized RGB channels with the mean and standard deviation of COCO dataset.
Classification: We used a MobileNet v3 to classify the identities of Star Wars characters. The dataset used for our training was from Kaggle[2]. We modified the final feed forward network so that the model produces a 1x51 probability map, corresponding to the 51 classes of characters in the Kaggle dataset.

- Training:
  For training, we fine-tuned a MobileNet v3 model pretrained on ImageNet dataset. We frozen the backbone (CNNs) of the model, and only update the parameters of the feed forward network.
- Data preprocessing:
  We normalized the training & referencing data with the mean and std. of ImageNet dataset. For training, we used several data augmentations: flipping, random cropping, random scaling, random erasing, random rotating. We also spilt the dataset into training set, val. set, and test set. The ratio of these three is 6:2:2.
- Result:
  We trained the model for 40 epochs, and we picked the best-performing checkpoint on val. set as the final model. The model obtained 87% top-1 accuracy and 91% top-5 accuracy on the test set. The curve of accuracy during training is shown below.



val. set top-1 acc curve

Conclusion

Star Wars Companion's goal is to allow Star Wars fans to use their device to recognize major star-wars characters and provide the characters' information. We implemented our program from deep learning modules to a graphical user interface and used the database to store all the characters' information we collected. Besides, we also used unit tests to test our code. Looking back at all three sprints we have done, the things we've accomplished are extraordinarily. Yet, there are still plenty of parts to be improved, including the GUI appearance, widget layout, diversity of character information, and recognition accuracy.

References

[1] Star-wars. https://www.starwars.com/ Accessed April 27, 2022.

[2] MATHURIN ACHÉ. Star wars images. https://www.kaggle.com/mathurinache/star-wars-images/ Accessed February 23, 2022.