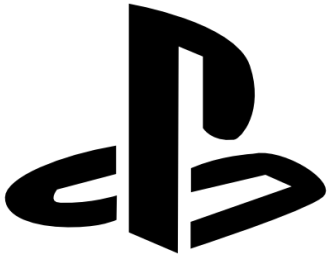
PS4 Game Recommender

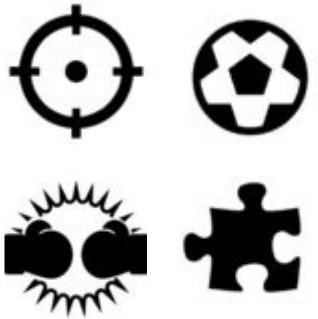
Team Spinosaurus:
Aayush Behl, Kolton Luu, Harshil Rajesh Patel, Nick Zhang

Introduction to our Application

Android App



PS4 Games



User-Specific Recommendations

Inputs and Outputs

3 User Input Images

Processing

Up to 5 Recommended PS4 Game Titles

Major Components

Datasets

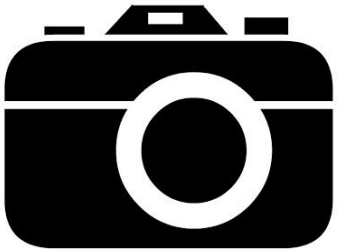
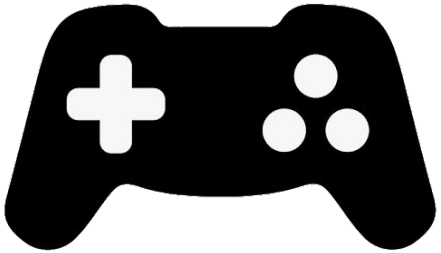
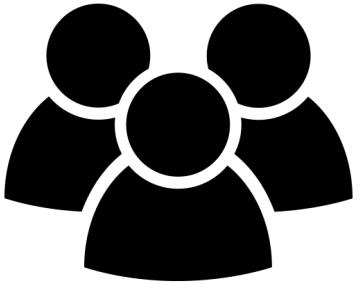


Image Dataset



Game Info Dataset



User Ratings Dataset

ML Components

YOLO Model:

Trained to extract the parts of the input game covers that correspond the the game title

Recognition / Classifier Model:

Trained by us to recognize the game title image from the YOLO model and classify it into the index in the games info dataset.

Recommender Model:

Built and trained by us to learn the attributes of each game. Takes the index from the classifier model as input and makes recommendations based on the similarities between games.

Non ML Components

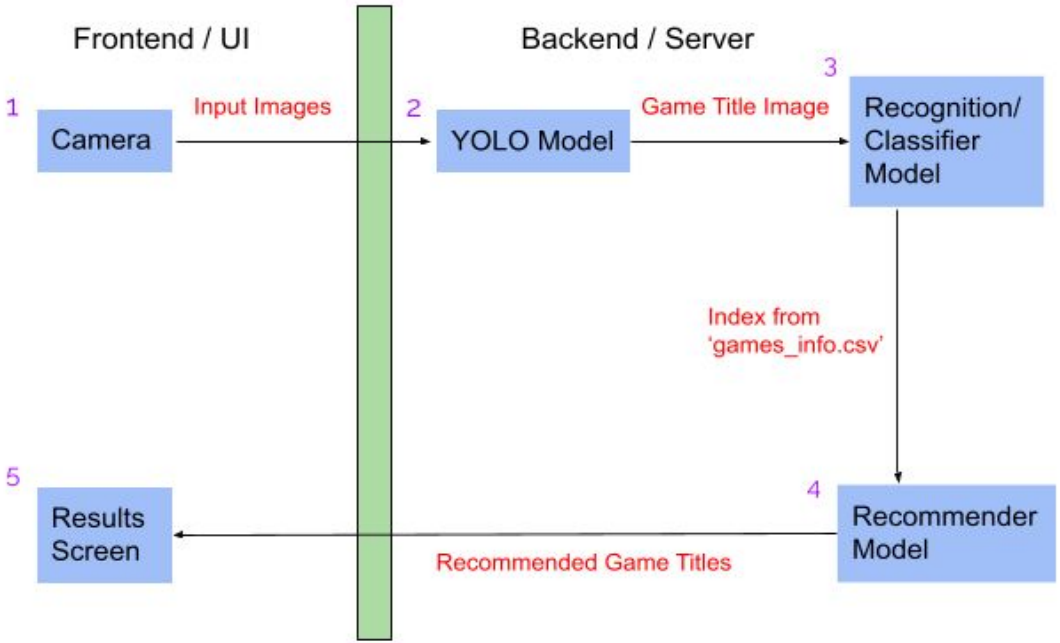
Frontend / App UI:

Kotlin based Android app that provides the interface for the user to communicate with the machine learning models on the backend server (eg, camera, buttons, input/output screens).

Backend / Server:

Implemented with Node.Js and stores the Machine Learning models as well as the scripts that run them. Responsible for using the three Machine Learning models to compute the final output, which is then communicated to the frontend.

The Flow of Data



Putting Everything Together

M1

DATASET CREATION

Web Scraping and formulating the .csv files for the game/user datasets

M2

MODEL TRAINING

Building and training the game recognition/classifier and recommender models

M3

BUILDING THE APP UI

Building the User Interface such as buttons, screens and imageViews

M4

APP DEPLOYMENT

Building a backend, establishing a connection to the frontend and deploying to Google Cloud

Technical and Design Decisions

1. YOLO model retrieves the fragment of the input game cover that corresponds to the game title before passing it into the recognition/classifier model for better accuracy.

2. Augmenting the user ratings dataset according to the metadata we collected (genres, developers, etc) to help the recommender model learn the attributes of the games better.

3. Holding all the stuff for heavy computation (i.e. datasets, ML models, scripts) on google cloud to minimize the size and processing time of the app.

LIVE DEMO!