YoGlad

A Real-time Yoga Pose Tutor System

Group 5

Chen Hao
Chiu Man Shan (Roy)
Kuch Swee Cheng (Ethan)
Zhao Lutong

CONTENT

01 Project Objective

02 Project Scope

03 Technical Approach

System Architecture
ML Techniques

04 Project Progress Update

Pixel-wise Model Training
Keypoint Based Model Training
Real-time Testing Demo Video

Project Objective

Background

Yoga is a widely popular means to reduce stress in modern lifestyle. In US, there are about 36 million Americans that practice yoga with around 100,000 registered yoga teacher, a ratio of 360:1. This has led many to turned to numerous self-help resources for learning yoga, such as books or videos from YouTube and TikTok.

Rising Injury Rate for Yoga

The rise in yoga's injury rate is believed to stem from unqualified teachers offering classes through online videos to eager beginner due to its low barrier of entry.

Pose Detection and Estimation

Keypoint Detection, Google MediaPipe Holistic together with OpenCV

Pose Classification

Image Preprocessing, Kernel Filtering, Pose Classification using CNN, Random Forest, SVM, Transfer Learning

Pose Correction

Pose Classification Accuracy, Keypoint Calculation and Compensation, Pose Correction Prompts

Real-Time Pose Detection

Real Time Classification and Pose Correction Prompts thorough web browser

https://financesonline.com/yoga-statistics/

Project Scope

In this Intelligent Systems project, we aim to achieve the means of a low-cost real-time yoga pose correction systems that could achieve a 90% classification accuracy while providing correction prompt offered through browser.

Data Sources

- Kaggle Image Dataset
- YouTube + CVAT labelling

Data Preparation

Hyperparameters and Layers configuration

- Deep Learning Model Tuning and Layers configuration.
- FC classification layers exploration work

Model Tuning

Pose Correction

- Validating pose classification accuracy and result
- Pose angle calculation and correction prompts

Validation

Modelling

Pose Classification

- Image Preprocessing and Training
- Classification using CNN+LSTM, Active Random Forest and SVM

Hybrid Machine Learning Model

Ensemble Systems

- Incorporating multiple weak classifier systems such as SVM, LR and RF into arbitrator
- Explore weighted decision into trained Keras Model

Deployment

Real-time Deployment

- Web browser with camera interface for user
- Pre-load trained model (OpenCV.js and Keras.js) into browser

Data Required/Processed for Model Training

Training data contains both the input and output values

1

Dataset for 5 yoga poses from Kaggle

- *5 classes of yoga poses, each composes about 300 images*
- *Photos mostly taken under perfect lighting, with distinguished background*

2

Capture real life training data from online

- *To make training data's distribution closer to testing or predicting data*
- *Augment data by customized dataset:*
  - *Extract frame from YouTube Yoga Instructors videos*
  - *Manually label the images with CVAT*

3

Extract body keypoint data from images

- *Utilize MediaPipe to extract the keypoint data from images of yoga practitioner performing different poses*

Technical Approach

System Architecture and ML Techniques

System Architecture

Dataset

kaggle

YouTube + CVAT

Training Scripts

Trained Model

Keras.js
to load model from the browser

OpenCV.js
to extract frames from video

Web browser with camera

ML Techniques

Pixel-wise Model Training (CNN)

Keypoint Based Model Training (RNN/LSTM)

Hybrid Machine Learning Model

Pose Estimation

Pose Correction Rules

Pose Corrective Instruction

Project Progress Update

Pixel-wise Model Training, Keypoint Based Model Training & Real-time Testing Demo

Pixel-wise Model Training

Convolutional neural networks (CNNs) based feature extraction methods and supervised learning algorithms

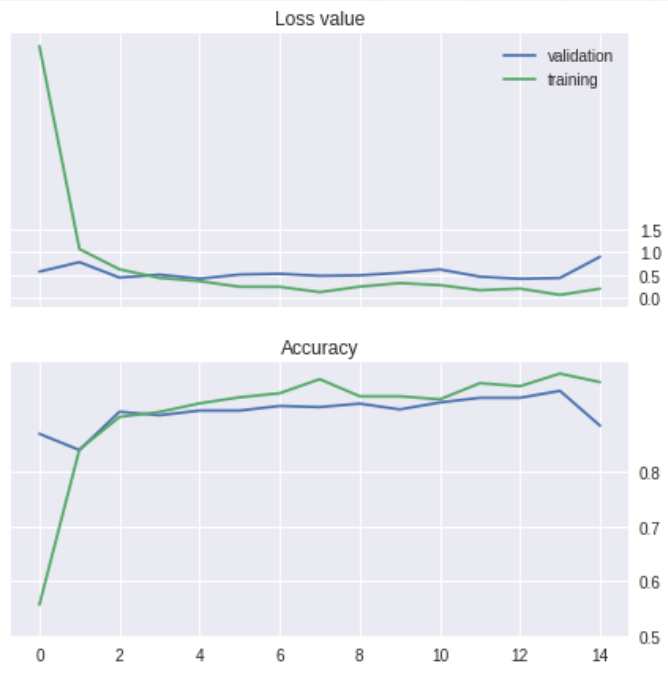
Model	Trained Iteration	Best Test Accuracy
VGG16	5	0.9617
VGG19	5	0.9723
InceptionV3	30	0.9489

Data Augmentation

- Resale
- Zoom 20%
- Horizontal Flip

Transfer Learning

- Pre-trained on ImageNet dataset
- Freeze all the convolutional layers and discard the last 3 FC layers
- Add and train the dense layer and use softmax as the activation function

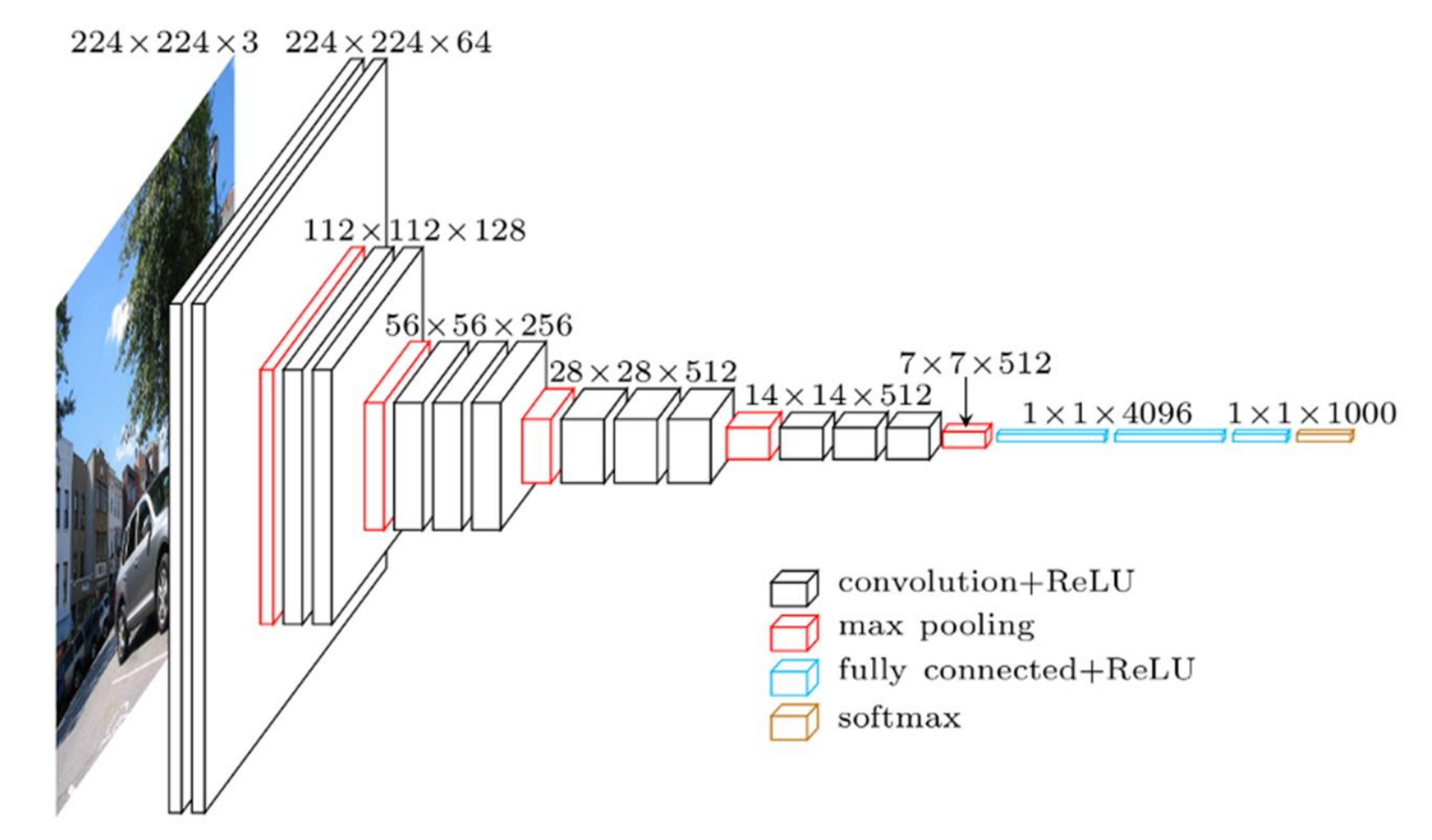


Pixel-wise Model Training

VGG 16

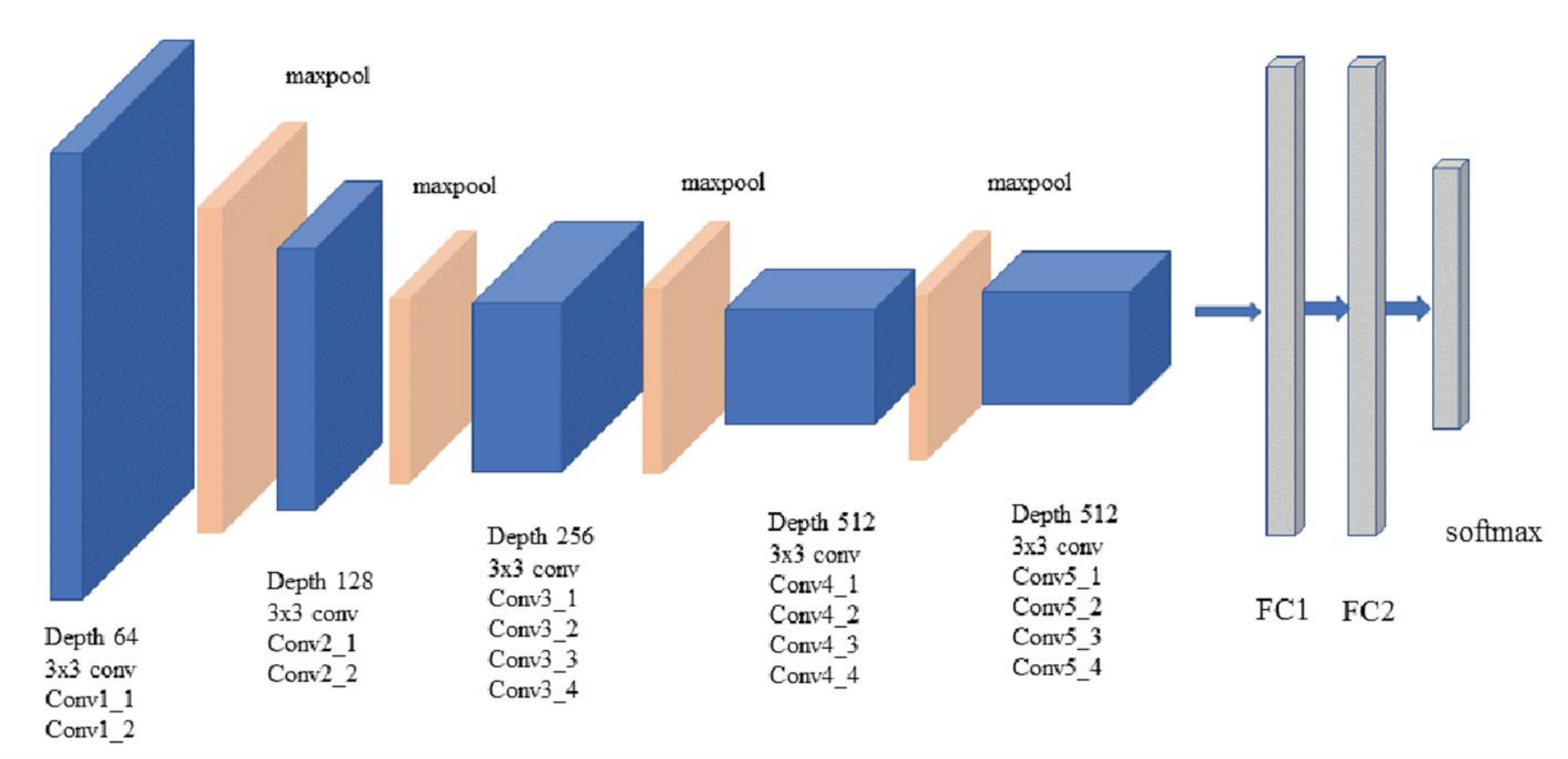
VGG 19

InceptionV3



Pixel-wise Model Training

- VGG 16
- VGG 19
- InceptionV3

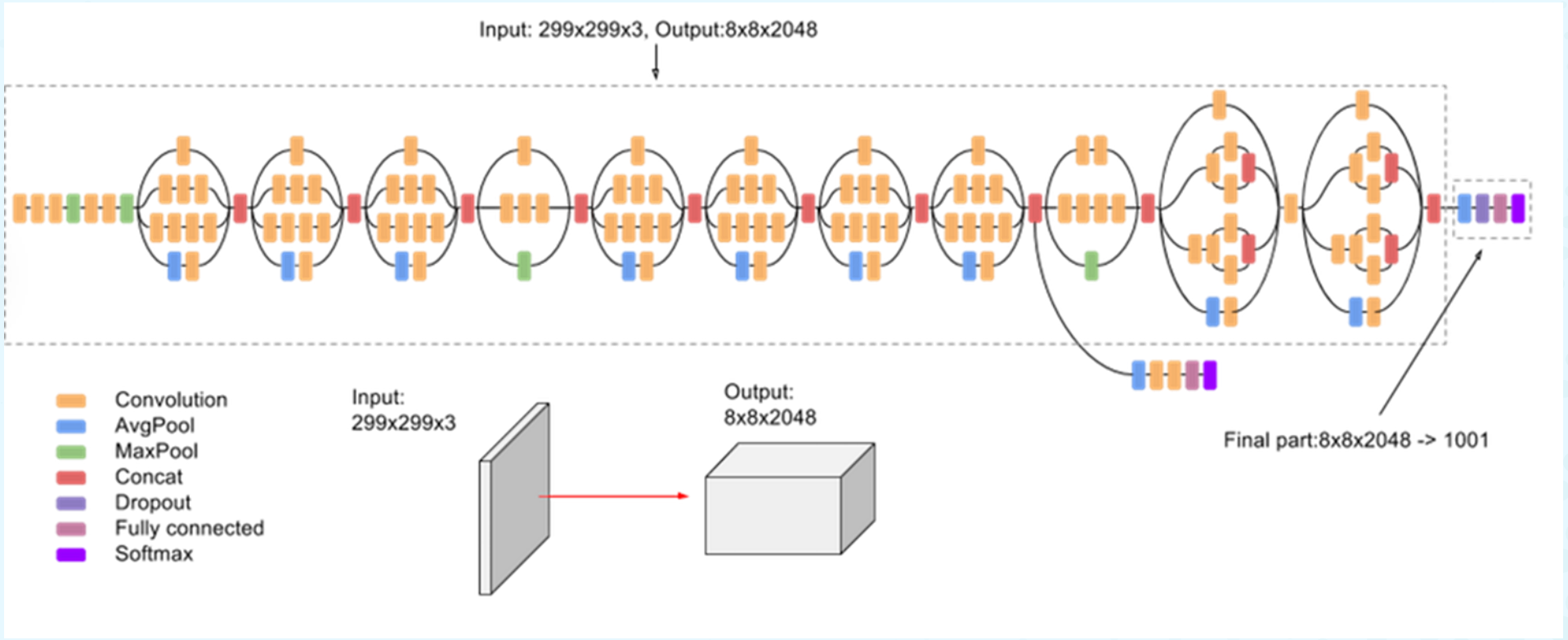


Pixel-wise Model Training

VGG 16

VGG 19

InceptionV3



Project Progress Update

Pixel-wise Model Training, Keypoint Based Model Training & Real-time Testing Demo

Keypoint Based Model Training

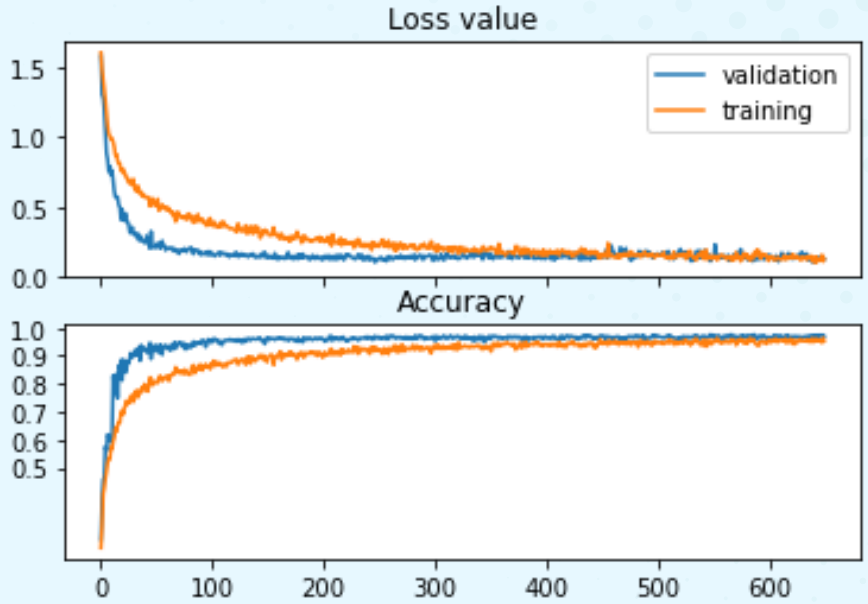
Keypoint Detection → **Preprocess Data and Create Labels**

Build and Train the Model

Layer (type)	Output Shape	Param #
lstm (LSTM)	(None, 1, 64)	82688
lstm_1 (LSTM)	(None, 1, 128)	98816
lstm_2 (LSTM)	(None, 1, 128)	131584
lstm_3 (LSTM)	(None, 64)	49408
dense (Dense)	(None, 64)	4160
dense_1 (Dense)	(None, 32)	2080
dense_2 (Dense)	(None, 5)	165

Total params: 368,901
Trainable params: 368,901
Non-trainable params: 0

Trained Iteration	Best Test Accuracy
650	0.9745



Project Progress Update

Pixel-wise Model Training, Keypoint Based Model Training & Real-time Testing Demo

Real-time Testing Demo Video

THANK YOU

Group 5