Rock Paper Scissors Game on Jetson Nano

Asst. PhD(s) Andrei Radu
Lect. PhD. Georgian Nicolae

Multimedia Technologies in Biometrics and Information Security Applications
National University of Science and Technology ''Politehnica'' of Bucharest

October 2024

Content

1. Introduction

Rock Paper Scissors

Why Gestures ?

- The recognized hand gestures $\rightsquigarrow$ **game moves** for human player.
- Gesture recognition enhances human-computer interaction in various applications, especially gaming.

Rock Paper Scissors

Why Gestures ?

- The recognized hand gestures $\rightsquigarrow$ **game moves** for human player.
- Gesture recognition enhances human-computer interaction in various applications, especially gaming.

Figure 1: Rules of Rock Paper Scissors Lizard Spock

2. Dataset and Model

Used Dataset

HaGRID [KMK22]

- HaGRID (HAnd Gesture Recognition Image Dataset) contains **553,991** FullHD RGB images divided into **18 classes** of gestures.
- Used for hand detection, keypoints recognition and **hand gesture recognition**.

Figure 2: HaGRID Logo, inspired by the Harry Potter character: Rubeus Hagrid

Gestures and Moves

Figure 3: Classes of HaGRID. The term *inv* stands for *inverted*.

Gestures and Moves

Figure 3: Classes of HaGRID. The term *inv* stands for *inverted*.

Gesture to Move

- Rock: fist
- Paper: stop & stop inverted
- Scissors: two up, two up inverted, peace & peace inverted

Hand Recognition Model

Available Models

The authors proposed multiple architectures, for different tasks:
Hand Detection:

- RetinaNet (w/ ResNet-50)
- SSDLite (w/ MobileNet v3)
- Yolo (V7 tiny)

Gesture recognition (from full frame):

- ResNet (and ResNeXt)
- ViT
- MobilNet

Models Comparison

What are we looking for?

- A fast model, which will be able to run on Jetson Nano
- For computing inference time $\rightsquigarrow$ Intel(R) Xeon(R) Platinum 8168
- **What time do we have between the captured frames?**

Model	Size (MB)	Parameters (M)	Inf. time (ms)	F1
ResNet-18	89.6	11.2	49.25	97.5
ResNet-152	466.5	58.3	292.6	95.5
ResNeXt-50	184.6	23.2	135.6	**98.3**
ResNeXt-101	696.4	87	397.2	97.5
MobileNetV3 small	**12.5**	**1.6**	**10.6**	86.4
MobileNetV3 large	34	4.3	33.4	91.9
VitB16	686.6	85.9	325.5	91.1

Table 1: Model comparison for the full-frame detectors

MobilNets

MobilNet v1

MobilNet introduces the concept of **Depthwise convolution**:

- Filters are applied *independently* to each input channel, reducing the computation.
- Instead of convolving all channels together, each channel has its own 2D convolution filter.

$$\hat{G}_{k,l,m} = \sum_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m}$$

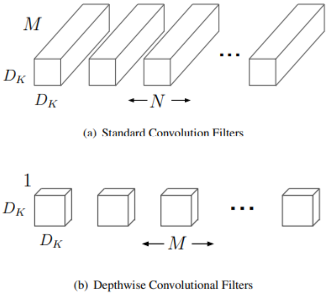

Figure 4: Convolution and Depthwise Convolution filters [How+17]

MobilNets

MobilNet v2

The second version introduces the **Inverted Residual block**

- Flips the traditional structure: **expands** the input dimensions first, applies depthwise convolutions, and then **projects back** to a smaller dimension.
- The input is first expanded to a higher-dimensional feature space, followed by pointwise and depthwise convolutions.

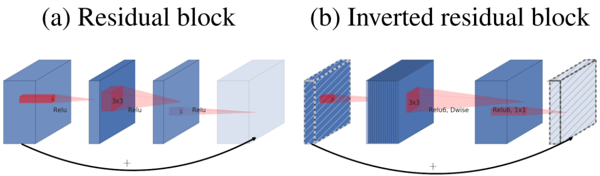

Figure 5: Residual block and Inverted Residual block [San+19]

MobilNets

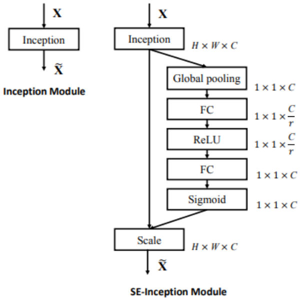

Figure 6: Squeeze and Excitation added to an Inception block [Hu+19]

MobilNet v3

Adds the Squeeze-and-Excitation (SE):
Squeeze Operation: Global Average Pooling is applied to each channel to generate a channel descriptor (a single value per channel). This "squeezes" the spatial dimensions into a single representative value.
Excitation Operation: The channel descriptor is passed through fully connected layers and a non-linear activation to generate channel-wise weights. These weights are multiplied back with the original feature map, thus "exciting" the most important channels.

ONNX Model Format

What is ONNX?

- ONNX (*Open Neural Network Exchange*) [BLZ+19] is an open-source format that facilitates interoperability between different ML frameworks.
- Supports models trained in frameworks like PyTorch, TensorFlow and more.

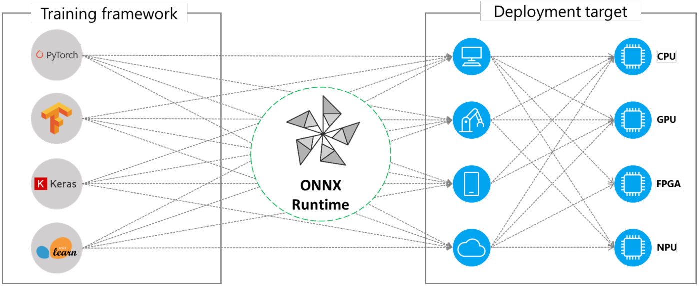

Figure 7: ONNX Operability [BLZ+19]

ONNX Model Conversion

Why Use ONNX?

- Allows easily converting models from one framework to another.
- Enables **optimized execution** across different hardware platforms: CPUs, GPUs, and specialized accelerators.
- ONNX has a large ecosystem with support from platforms like AWS, **NVIDIA** and Intel.

Exporting Models

Exporting models to ONNX is supported by multiple frameworks:

- PyTorch is natively supported by providing a simple function call: `torch.onnx.export()`.
- TensorFlow models can be converted using `tf2onnx` or `onnx-tf`.
- Tools like `onnx-simplifier` can be used to reduce model complexity.
- We can use `onnx-runtime` to run the models without having a framework installed.

3. Application

Application Flow

Restrictions

- Model accepts an input of 224px $\times$ 244px.
- We should use as much available data as possible.
- **What optimizations can we do, to make it run faster?**
- **How about the correctness of the app?**

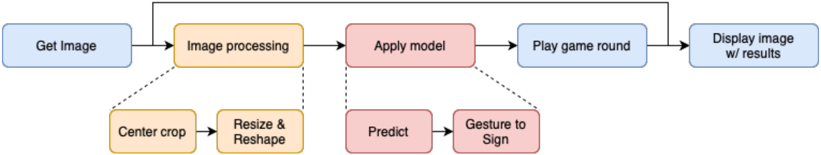

Figure 8: Application flow

Practical Part

Let's play!

- Go to the following repository: **github.com/andrei-radu/iasi-rps**
- Follow the installation instructions.
- Run the app!

Practical Part

Let's play!

- Go to the following repository: **github.com/andrei-radu/iasi-rps**
- Follow the installation instructions.
- Run the app!

Try it out

Complete the following table with the required information:

Prediction	Mean time for 1s	Speed-up
Each frame		-
3 times per s		
1 time per 1		

Practical Part II

Multiprocessing?

Instead of predicting based on several frames passed, dedicate a thread in which the computation is done.

- Use Python's `multiprocessing` library.
- Global variables and data structures might be useful.

Practical Part II

Multiprocessing?

Instead of predicting based on several frames passed, dedicate a thread in which the computation is done.

- Use Python's `multiprocessing` library.
- Global variables and data structures might be useful.

Rock Paper Scissors Lizard Spock?

- Choose two (or more) other signs from the recognised ones and associate them with new moves: **lizard** and **Spock**.
- Implement the new game logic, by modifying the scripts.
- The rules for this game are presented on the `Introduction` slide.

Practical Part II

Better strategy?

Try to implement new strategies for the computer. These can be based on the psychology of most players [Far15], which tend to:

- uses the same move if they won the last round.
- change the move if they lost.

Practical Part II

Better strategy?

Try to implement new strategies for the computer. These can be based on the psychology of most players [Far15], which tend to:

- uses the same move if they won the last round.
- change the move if they lost.

Two Player Game?

Change the image processing module such that:

- Create two images from the source, using the left-most and right-most parts.
- Predict the sign from each image.
- Modify the game to allow two players.

References I

[BLZ+19] Junjie Bai, Fang Lu, Ke Zhang, et al. *ONNX: Open Neural Network Exchange*. https://github.com/onnx/onnx. 2019.

[Far15] Neil Farber. *The Surprising Psychology of Rock-Paper-Scissors*. 2015. URL: https://www.psychologytoday.com/us/blog/the-blame-game/201504/the-surprising-psychology-rock-paper-scissors (visited on 10/21/2024).

[How+17] Andrew G. Howard et al. *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*. 2017. arXiv: 1704.04861 [cs.CV]. URL: https://arxiv.org/abs/1704.04861.

[Hu+19] Jie Hu et al. *Squeeze-and-Excitation Networks*. 2019. arXiv: 1709.01507 [cs.CV]. URL: https://arxiv.org/abs/1709.01507.

[KMK22] Alexander Kapitanov, Andrey Makhlyarchuk, and Karina Kvanchiani. "HaGRID - HAnd Gesture Recognition Image Dataset". In: *arXiv preprint arXiv:2206.08219* (2022).

[San+19] Mark Sandler et al. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. 2019. arXiv: 1801.04381 [cs.CV]. URL: https://arxiv.org/abs/1801.04381.