DARE: AI-based Diver Action Recognition System using Multi-Channel CNNs for AUV Supervision

This work has been published in:


The copyright of this presentation is held by the authors and the LINKS lab.
Human Robot Interaction requires the ability to dynamically reprogram the robot’s mission parameters and human control input is limited to visual command.

- Traditional commands require waterproof joystick, keyboard or tablet.
- Diver gestures/ pose commands are more convenient and faster.

**Objective:** Develop a deep learning model on diver’s action images to perform action recognition faster and accurately

- Diver images collected in both open sea and swimming pool should be included.
- Minimize the classifying time for each action.
- High accuracy is required when classifying all the action types.
Literature Review

Existing Image Classification for Diver Gesture

- Images captured in ideal swimming pool and terrestrial environment
- RGB camera
- Convolution Neural network

- Images were captured in ideal swimming pool environment
- Monocular camera
- Motion trajectories

- Images were captured in ideal swimming pool and open sea environment
- Monocular RGB camera
- Fast Recurrent Convolution Neural Network

- Simulated target body velocity using acoustic wireless networks
- Arm motions classification
- Convolution Neural network

Research Gap: None or Limited amount of images captured in real open sea environments. Images captured using monocular RGB camera contain bland spot (information lost). Underwater diver motion classification involved not only arm but also whole body.

References:
Cognitive autonomous diving buddy (CADDY) gesture which include open sea and swimming pool scenario with 16 different gestures and 3 poses to recognize.

Underwater diver postures are focus on whole body including arm positions.

- Data collected location: **open seas** of Biograd na Moru, Croatia, an **indoor pool** in the Brodarski Institute, Croatia, and an **outdoor pool** in Genova, Italy.
- 16 different gestures + 1 true negative
- Up, down, backwards, carry, boat, 1-4, take a photo etc
- Stereo pairs of gestures \((9239+7190) \times 2 = 32,858\)
  
  \(9239 \rightarrow\) gestures, \(7190 \rightarrow\) true negatives

Underwater diver gesture sample images in various environments

---

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Start</td>
<td></td>
<td></td>
<td>Up</td>
<td></td>
<td></td>
<td>End</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Here</td>
<td></td>
<td></td>
<td>Take a photo</td>
<td></td>
<td></td>
<td>Four</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Carry</td>
<td></td>
<td></td>
<td>Tessellation</td>
<td></td>
<td></td>
<td>Two</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Down</td>
<td></td>
<td></td>
<td>One</td>
<td></td>
<td></td>
<td>Backward</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Three</td>
<td></td>
<td></td>
<td>Five</td>
<td></td>
<td></td>
<td>Number delimiter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Boat</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

CADDY dataset diver hand gesture
• Data collected location: **open seas** of Biograd na Moru, Croatia, an **indoor pool** in the Brodarski Institute, Croatia, and an **outdoor pool** in Genova, Italy.

• 3 different poses

• (1) turn horizontally (2) turning vertically (3) swim freely.

Stereo pairs of gestures \((3934+2722+6052)\times 2=25,416\)

---

Challenges of Underwater Diver Action Recognition

Critical Challenges

- Diver uncertainty
- Environment complexity
- Sensing uncertainties
- Fusion of stereo camera images
- Computational efficiency and reliability

Challenges of diver uncertainty and environment complexity
DARE Architecture

AUV
Stereo camera
Synchronized left and right images

Level number
Gesture code

Tree Topology Classifier
Training architecture

CNN based Robotic Application
Main Idea: Use transfer learning pre-trained Convolutional neural network (CNN) to extract useful features for training and classifying the diver’s gestures and poses.

<table>
<thead>
<tr>
<th>Model</th>
<th>Training Data</th>
<th>Computation</th>
<th>Training Time</th>
<th>Model Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traditional CNN</td>
<td>1000s to millions of label images</td>
<td>Compute intensive</td>
<td>Days to weeks for real problems</td>
<td>High (can overfit to small dataset)</td>
</tr>
<tr>
<td>Transfer learning pre-trained CNN</td>
<td>100s to 1000s of label images</td>
<td>Moderate computation</td>
<td>Minutes to hours</td>
<td>Good, depends on model structure</td>
</tr>
</tbody>
</table>

Benefits:

- Filters in convolution layer in CNN produce feature maps which contain important information. And Pooling layer will preserve the useful information but reduce the image size.
- Transfer learning prevents overfitting from training a network from scratch.
- Using different transfer learning nets provides a scope to observe the differences between the network structure and result.
## Pre-Trained Network Architectures

<table>
<thead>
<tr>
<th></th>
<th>AlexNet</th>
<th>VggNet</th>
<th>ResNet</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Depth</strong></td>
<td>8</td>
<td>16</td>
<td>18</td>
</tr>
<tr>
<td><strong>Input Layer size</strong></td>
<td>227x227x3</td>
<td>224x224x3</td>
<td>224x224x3</td>
</tr>
<tr>
<td><strong>Filter Size</strong></td>
<td>11x11, 5x5, 3x3</td>
<td>3x3</td>
<td>7x7, 3x3, 1x1</td>
</tr>
<tr>
<td><strong>Number of Conv layer</strong></td>
<td>5</td>
<td>13</td>
<td>17</td>
</tr>
<tr>
<td><strong>Number of Fully-connected layer</strong></td>
<td>3</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td><strong>Number of hyper-parameters</strong></td>
<td>61.1 million</td>
<td>138.4 million</td>
<td>11.2 million</td>
</tr>
</tbody>
</table>

### AlexNet architecture

- **Input**: 227 x 227 x 3
- **Conv1**: 55 x 55 x 96 (11 x 11)
- **Conv2**: 27 x 27 x 256 (5 x 5)
- **Conv3**: 13 x 13 x 256 (3 x 3)
- **Conv4**: 13 x 13 x 384 (3 x 3)
- **Conv5**: 13 x 13 x 384 (3 x 3)
- **FCL6**: 4096
- **FCL7**: 4096
- **FCL8**: # Class

### References


Feature Visualization Using AlexNet

Features process:
- Stage 1 contains 96 features in total with selected 25 random feature maps shown in figure.
- Convolutional layer used filters to extract information across the images such as edge and line.
- ReLu is an activation function which introduce non-linearity to the network.
- Max Pooling layer down-samples the images.
- Stage 5 output feature maps contains difference among these three different gestures.
Fusion of Stereo Images

Bi-Channel Feature Extraction

Objective

• Preserve correlation between left and right stereo images
• Maintain crucial diver AUV distance information
• Prevent overfitting

Multi-Channel Feature Extraction
DARE Architecture

Training architecture
Flattening

Left pool5 feature maps

Right pool5 feature maps

Left flatten size: 6x6xm

Right flatten size: 6x6xm
DARE Architecture

Training architecture

CNN based Robotic Application
Grouping Standard

- Similar gesture/hand shapes and orientation
- Similar glove features
- Similar arm position and body orientation
- Class looks distinct with other class should not be grouped

Hierarchy Tree-structured Neural Network Classifier

L0

L1

L2

GNet20
- One hand (back face)
- Two hands closed (back face)
- Two hands open (back face)
- True negatives (no, improper or wrong gesture)

GNet10
- RootNet
  - Gestures
  - Poses

GNet21
- One hand (front face)
- Three fingers or two fingers and a thumb

PNet11
- Turning horizontally
- Turning vertically
- Free swim

GNet30
- Thumb pointing up or no finger
- No finger

GNet31
- Finger pointing down or front
- Point downward
- Point forward

GNet32
- Finger pointing up
- Two fingers
- One finger

GNet33
- Thumb pointing down
- Four fingers

GNet34
- Four fingers

GNet35
- Five fingers

11/23/20 CNN based Robotic Application
**Hierarchy Tree-structured Neural Network Classifier**

**Classifier Training Process**
- Input: Multi FM with corresponding class,
- Output: Diver action class
- Train 11 small classifiers independently with same network architecture
- Parameters fine-tuning for each classifier

**Classifier Testing Process**
- Test feature extracted using multi-channel network first goes into RootNet
- Perform prediction on next level based on result from previous level
- Stop prediction when diver action class is the output
Analysis Setup

**Prepressing procedure:**
- 5-Fold cross validation with each fold 20% for testing and 80% for training
- Resize Image using down-sample ratio

**Parameters:**
- Solver: Stochastic gradient descent with momentum (SGDM)
- Initial learning rate: 0.001
- minibatch size: 64 (default), 500/100 (RootNet, GNet10 and PNet11)

**Machine:**
- MATLAB Deep Learning Toolbox
- Windows 10 computer with an Intel Core i7 processor and 32GB RAM
Performance Measures

### Confusion Matrix

<table>
<thead>
<tr>
<th>Target Class</th>
<th>C0</th>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0</td>
<td>C00</td>
<td>C01</td>
<td>C02</td>
</tr>
<tr>
<td>C1</td>
<td>C10</td>
<td>C11</td>
<td>C12</td>
</tr>
<tr>
<td>C2</td>
<td>C20</td>
<td>C21</td>
<td>C22</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Prediction Class</th>
<th>C0</th>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>C0</td>
<td>C0</td>
<td>C1</td>
<td>C2</td>
</tr>
<tr>
<td>C1</td>
<td>C1</td>
<td>C2</td>
<td>C0</td>
</tr>
<tr>
<td>C2</td>
<td>C2</td>
<td>C0</td>
<td>C1</td>
</tr>
</tbody>
</table>

### Correct Classification Rate:

- **True Positive Rate (TPR)**
  \[
  TPR_i = \frac{TP_i}{TP_i + FN_i}
  \]

- **True Negative Rate (TNR)**
  \[
  TNR_i = \frac{TN_i}{TN_i + FP_i}
  \]

**Balanced Individual Class Accuracy**

\[
Balanced Ind. Class Accuracy_i = \frac{TPR_i + TNR_i}{2}
\]

**Overall CCR**

\[
Overall CCR = \frac{\sum_i C_{ii}}{\sum_i \sum_j C_{ij}}
\]

### F1 Score:

- **Precision**
  \[
  Precision_i = \frac{TP_i}{TP_i + FP_i}
  \]

- **Recall**
  \[
  Recall_i = TPR_i = \frac{TP_i}{TP_i + FN_i}
  \]

\[
F1 Score_i = 2 \times \frac{Precision_i \times Recall_i}{Precision_i + Recall_i}
\]

**Overall F1 Score**

\[
Overall F1 Score = \frac{1}{N} \sum_{i=0}^{N} F1 Score_i
\]

Where \( N \) = total class number
Results

Individual Class Performance

Balanced accuracies of ResNet-based networks

Balanced accuracies of AlexNet-based networks

Balanced accuracies of VggNet-based networks

F1 scores of ResNet-based networks

F1 scores of AlexNet-based networks

F1 scores of VggNet-based networks
## Results

### Overall Performance

<table>
<thead>
<tr>
<th>Metrics</th>
<th>ResNet</th>
<th>AlexNet</th>
<th>VggNet</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Regular</td>
<td>MC</td>
<td>DARE</td>
</tr>
<tr>
<td>CCR (%)</td>
<td>86.03</td>
<td>88.80</td>
<td>89.47</td>
</tr>
<tr>
<td>F1 score</td>
<td>0.728</td>
<td>0.782</td>
<td>0.799</td>
</tr>
<tr>
<td>Training Time (hrs)</td>
<td>1.46</td>
<td>1.72</td>
<td>3.65</td>
</tr>
<tr>
<td>Testing Time (ms)</td>
<td>34.59</td>
<td>69.24</td>
<td>72.65</td>
</tr>
</tbody>
</table>
Conclusion

– Human robot interaction application using DARE achieve high CCR equal to 95.87% in relatively short amount of time
– DARE boost every individual class accuracy above 92%
– Suitable for real-time application

Future Work

– Convolutional neural network grid search to find the best combination of hyper-parameters
– Image pre-processing techniques can be embedded to improve the algorithm performance
– Automate tree construction for human-robot interaction