Modeling Human Body Motion Via Neural Networks

Zeyuan Liu
School of Computer Science & Engineering, Seoul National University
zeyuan-liu@bi.snu.ac.kr

Background

• WEKA: a collection of machine learning algorithms for data mining (Weka 3)
• ROS provides libraries and tools to help software developers create robot applications
• Motion Detection: a process of confirming a change in position of an object relative to its surrounding or the change in the surroundings relative to an object

Research Questions

• How does Multiplayer perception as a means of neural networks apply to motion detection?
• Can we achieve optimality by analyzing WEKA outputs, given certain inputs and variables?
• How reliable is the combination of ROS and WEKA for human body motion detection?

Method

• Step 1: collect data using ROS with three different motions. Upon collecting the data, we need to convert the data into a matrix with 4 columns to a matrix with 60 columns via Matlab commands. Lastly, we import it as the data in the .arff file.
• Step 2: based on the data, apply cross-validation to separate it into a training data set and a test data set. Cross-validation is a technique for one that wants to estimate the accuracy of a predictive model. (Michie, Spiegelhalter & Taylor, 1994)
• Step 3: process the data using WEKA, where the entire data set is divided between training data and test data depending on the users setting, and apply multi-layer perceptrons with a number of parameters to execute the data (Richard, 1997)

Data

• collected using Kinect through ROS
• 15 joint angles with 4 coordinate points
• 60 features associated with a single motion
• Three body motions collected and used for the project
• Each motion recorded at 100 different times

Results & Discussion

Graphical Visualizations of the data

• A general structure of the learning process of Multiplayer perceptrons

- Distribution of each joint with its associated four coordinates over 100 different intervals

WEKA output of nine experiments subject to different variable sets:

<table>
<thead>
<tr>
<th>Learning Rate</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.3</td>
<td>99.87</td>
</tr>
<tr>
<td>0.4</td>
<td>99.87</td>
</tr>
<tr>
<td>0.5</td>
<td>99.87</td>
</tr>
<tr>
<td>0.6</td>
<td>99.83</td>
</tr>
<tr>
<td>0.7</td>
<td>99.83</td>
</tr>
</tbody>
</table>

Results from varying learning rates while keeping all other settings fixed including momentum at 0.2:

<table>
<thead>
<tr>
<th>Momentum</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>99.87</td>
</tr>
<tr>
<td>0.3</td>
<td>99.87</td>
</tr>
<tr>
<td>0.4</td>
<td>99.87</td>
</tr>
<tr>
<td>0.5</td>
<td>99.83</td>
</tr>
</tbody>
</table>

Results from varying momentum while keeping all other fixed including learning rate at 0.8:

- In this project, we collect data through Kinect using Weka, and manage to collect 300 instances. Then, we separate the data into a training set and a test set by applying 10 folds cross-section validation. We try to classify the test set by exploiting knowledge from the training set. With Weka, we have achieved an accuracy close to 100% by classifying the test set with Multilayer Perceptron classifier.

References