We develop a Python library facilitating the development of distributed machine learning algorithms using embedded devices. As a demonstration of the library's functionality, we present implementations of geo-routing and classifier training.

The Averaging Problem

- Model: Segment data, take local action, exchange results with neighbors
- Decentralized computation (e.g., sensor networks)
  - Asynchronous and synchronous
  - Naturally extended to algorithms such as gradient descent and support vector machines

\[ \mathbf{x}_{t+1} = W^\alpha \mathbf{x}_t \]

The synchronous update rule, with the \( \mathbf{x} \) vector as each node's value at time \( t \) and \( \alpha \) as the number of iterations.

\[ W = \begin{bmatrix} 1 - \frac{d_1}{d_{\text{max}}+1} & \frac{1}{d_{\text{max}}+1} & \frac{1}{d_{\text{max}}+1} \\ \frac{1}{d_{\text{max}}+1} & 1 - \frac{d_1}{d_{\text{max}}+1} & \frac{1}{d_{\text{max}}+1} \\ \frac{1}{d_{\text{max}}+1} & \frac{1}{d_{\text{max}}+1} & 1 - \frac{d_1}{d_{\text{max}}+1} \end{bmatrix} \]

A sample stochastic matrix, with \( d_i \) as node \( i \)'s degree, and \( d_{\text{max}} \) as the maximum degree in the topology.

Geo-Routing

- Sensor network task
- Route message to given GPS coordinates over unknown topology
- Every node only knows own and neighbors' locations
- Greedily pass to closest neighbor

Future Work

- Testing with physical sensors
- Simulation of packet dropping, dying nodes
- Bandwidth/Power consumption tests for field deployments

Message Sending and Callbacks

```python
m = Messager()
neighbor = '5'
message = { 'num': 42 }
m.sendMessage(neighbor, message)
def callback(message, name):
    print('Got message from %s: %s' % (name, message))
m.registerCallback(callback)
```

Classifier Training

- Gradient Descent, SVM
- Use synchronous averaging to get feature vector
- Compute individual node errors and final classification

Acknowledgements

We would like to thank resin.io for generously donating the Raspberry Pis, and Professor Sarwate for his guidance on this project!