

TRAWL – A Traffic Route Adapted Weighted Learning Algorithm

Enda Fallon^{***}, Liam Murphy^{*}, John Murphy^{*}, Chi Ma^{**}

Performance Engineering Laboratory, University College Dublin, Ireland^{*}
Software Research Institute, Athlone Institute of Technology, Athlone, Ireland^{**}
E-mail efallon@ait.ie, Liam.Murphy@ucd.ie, j.murphy@ucd.ie, cma@research.ait.ie

Abstract. Media Independent Handover (MIH) is an emerging standard which supports the communication of network critical events to upper layer mobility protocols. One of the critical features of MIH is the event service. This service supports predictive network degradation events, which are triggered based on link layer metrics. For set route vehicles, the cyclical nature of movement enables a degree of network performance prediction. We propose to capture this performance predictability through a Traffic Route Adapted Weighted Learning algorithm (TRAWL). TRAWL is a feed forward neural network whose output layer is configurable for both homogeneous and heterogeneous networks. TRAWL uses an unsupervised back propagation learning mechanism which captures predictable network behaviour while also considering dynamic performance characteristics. We evaluate the performance of TRAWL using a commercial metropolitan heterogeneous network. We illustrate that TRAWL has significant performance improvements over existing MIH link triggering mechanisms.

Keywords: MIH, vehicular systems, handover, neural networks

1. Introduction

The IEEE 802.21 working group propose the MIH standard [1] to support the communication of network critical events to upper layer mobility protocols. While the MIH standard defines the interface for communication of link layer metrics to upper layer mobility protocols it does not provide specifics on the mechanisms which should be employed to trigger such events. Many existing algorithms [2][3][4] define static thresholds for performance metrics such as RSS. When these thresholds are exceeded events such as Link_Going_Down (LGD) and Link_Down (LD) are triggered. For set route vehicle systems such approaches are limited as they do not consider how the cyclical nature of movement enables historic performance metrics to influence predictive link triggering.

In this paper we focus on the optimisation of network handover for set route vehicles such as public transport busses and trains. Such vehicles typically operate in preconfigured routes which are repeated at routine intervals sometimes many times a day. We propose TRAWL, an unsupervised feed forward neural network, which captures repetitive network behaviour while also considering the dynamic performance characteristics of heterogeneous networks.

TRAWL consists of 2 major components; Route Identification and Management (RIM) and an Unsupervised Vehicle Learning Algorithm (UVLA). RIM is responsible for the configuration and removal of routes in the system. RIM also maintains the relationship between Access Points (APs) and vehicle routes. UVLA consists of a feed forward neural network which implements the decision logic for TRAWL. UVLA input consists of a selection of normalised performance metrics. The number of neurons in the output layer is dependent on the network configuration; 1 for homogeneous networks, 2 or more for heterogeneous networks. When the stimulation of a neuron exceeds a user defined activation threshold, the neuron “fires” producing a binary positive output. Positive binary output triggers path switchover to the path specified by the inputs. UVLA aims to maximise throughput per route cycle. In order to determine the learning rate we calculate the rate of change of a linear regression line through historic cycle throughput. A large rate of change results in a large alteration of synaptic weights and vice versa.

We evaluate our approach using performance metrics from a commercial heterogeneous network installation. We illustrate that our approach has significant performance improvement over existing MIH link event triggering algorithms.

2. Related Work

Many existing MIH implementations utilize a performance threshold P_{thres} to generate the MIH LGD event. In such scenarios the relationship between the time that P_{thres} (actual or projected) is exceeded, T_{deg} , and the time at which path handover is initiated, T_{h-init} , can be expressed as follows:

$$T_{h-init} = \alpha_{lgd}(T_{deg})$$

α_{lgd} is an anticipation factor which is applied to T_{deg} to adjust the aggressiveness with which the LGD event is triggered. Many implementations are based on pre-defined P_{thres} , mostly associated with RSS. If the current RSS crosses P_{thres} the LGD event is generated [2][3][4]. The NIST MIH implementation in NS2 [2] utilizes the power level of received packets $RXThresh(P_{thres})$ and $Pr_limit(\alpha_{lgd})$ to control link event triggering.

There are a significant number of recent studies in the area of network handover for vehicle based systems. These studies can be generally categorized in the area of Vehicle Ad-hoc NETworks (VANET) or infrastructure mode network access. Our investigation relates to the latter. In [6], SWiFT focuses on handover optimisation for vehicles travelling greater than 200kmph. SWiFT uses the speed of MN movement and RSS as the basis for link event triggering. It does not consider the condition of the link in the handover decision. [7] proposes a MIP handover mechanism for VANET which maintains the original Care of Address (CoA) configured at the original AP. [8] uses Proxy Mobile IPv6 (PMIP) and Host Identity Protocol (HIP) to reduce handover latency for urban vehicular systems. In [9] we propose multi-homing rather than MIP to pre-configure alternate paths prior to network handover. [10] proposes a collaborative approach in which APs use MN position prediction to limit the potential for retransmission. Such an approach has significant network infrastructure requirements. [11] proposes an architecture for network selection in vehicular systems based on network metrics, user requirements and application QoS. We propose that

the cyclical nature of public transport vehicles enables performance predictability which is not exploited by any of these approaches.

There are a number of ongoing studies which evaluate how artificial intelligence techniques can be used in the optimisation of network handover. [12] [13] propose a mutually connected neural network in order to optimise load balancing and QoS for the entire network. Our work focuses on the optimisation of throughput for client devices. [14] proposes a Hopfield neural network which considers multiple input parameters in the selection of networks. The study does not consider how the cyclical nature of routes used by public vehicles can be used in the optimisation of weights.

3. Technology Overview

3.1 Media Independent Handover

MIH is an emerging IEEE standard [1] which proposes a framework to support seamless handover between homogenous and heterogeneous networks. MIH provides an architecture for communicating network critical metrics to Upper Layer Mobility Protocols (ULMP). MIH supports 3G, HSDPA, Bluetooth, WiFi and WiMax. It consists of 3 services; an event service, information service and command service.

The event service is responsible for communicating dynamic network conditions to ULMP through events such as LGD and LD. These events can be used to optimize handover time. The information service is responsible for communicating information existing within a geographical area, such as BS locations, to ULMP. The command service enables ULMP to control the physical, data link and logical link layers through a set of handover commands.

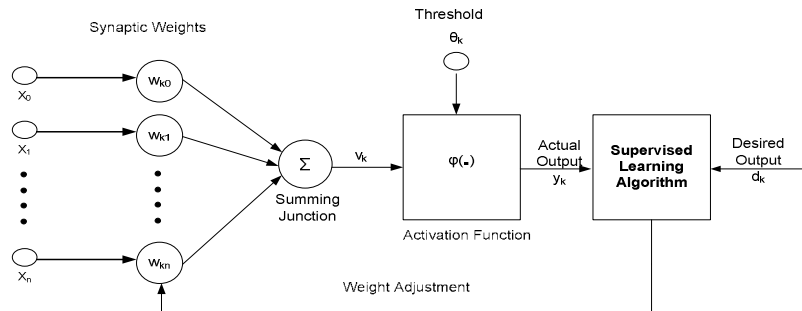
3.2 Artificial Neural Networks (ANN)

ANN are data processing models which are based on the operation of the brain. ANN consist of a large number of interconnected neurons working in parallel, typically used for pattern recognition and data classification. This paper proposes TRAWL, an ANN used to capture historic performance trends for cyclical route vehicle systems. These trends are weighted against dynamic performance metrics.

Fig 1 illustrates a supervised learning ANN. Values $x_0, x_1, x_2, \dots, x_n$ are provided as input to the neuron. The neuron has 2 modes of operation; training or trained. In trained mode, the neuron applies synaptic weights $w_{k0}, w_{k1}, \dots, w_{kn}$ which enhance or degrade the input values. These weighted values are summed and an activation function $\phi(\cdot)$ is applied. $\phi(\cdot)$ determines whether the neuron should “fire”, producing an output y_k which classifies the input pattern.

Training mode can be implemented through supervised or unsupervised learning. In supervised learning the ANN will have an offline training phase in which neural outputs are compared against a training set. Alterations are made to the synaptic weights to limit the error in classification between the output y_k and the training set d_k . When the ANN correctly classifies the input pattern, the ANN operates in trained mode. Unsupervised learning has no external training patterns. In this mode the ANN self organizes data presented to the network and detects recurrent properties.

Fig. 1 A Supervised Learning Neural Network



4. TRAWL – A Traffic Route Adapted Weighted Learning Algorithm

Our TRAWL algorithm consists of 2 components:

Route Identification and Management (RIM) – is responsible for the identification and management of vehicle routes. Using the geographical position of the vehicle RIM distinguishes existing, altered or new routes.

Unsupervised Vehicle Learning Algorithm (UVLA) – implements the path selection intelligence within TRAWL. UVLA is a feed forward neural network which operates with a single output neuron for homogeneous networks or 2 output neurons for heterogeneous networks. Back propagation and weight adjustment are implemented each time the vehicle completes a cycle of a route.

The pseudocode below outlines the structure of the TRAWL algorithm. TRAWL dynamically configures and maintains traffic routes using GPS coordinates. Having read the GPS coordinates, TRAWL determines if the current geographical position uniquely identifies a route. If the geographical position is not previously configured, a new route is created and training is initiated. If the current geographical position uniquely identifies a previously defined route, the TRAWL algorithm determines if ANN training is required for that route.

TRAWL operates in either training or trained mode. As an unsupervised algorithm TRAWL does not have an offline training phase. TRAWL uses initial end user synaptic weights to determine if handover is required. Following each route cycle the throughput is calculated and synaptic weights are adjusted. TRAWL is trained when (a) the training process does not update synaptic weights (b) synaptic weight updates have no effect on throughput. TRAWL ensures that synaptic weights remain relevant to changing network conditions by applying an `accuracyThreshold`. Throughput is measured for every route cycle and if the `accuracyThreshold` is exceeded training is reinitiated.

```

Struct Route{
    param trainingmode = false //training or trained
    param GPS_Cord startOfRoute
    param GPS_Cord[] existingRoute // co-ords for existing route
    param float[] weights // synaptic weights
    param float activationThreshold //if summation > then fire
    param float[] historicThroughput //previous cycle throughput
    param float learningRate // rate of altering of synaptic weights
    param float accuracyThreshold // if throughput < reinitiate training
}

Routine::TRAWL()
    GPS_Cord CurrentPosition = get GPS_Position()
    foreach(Route) // RIM route management
        if(CurrentPosition contained in Route.StartOfRoute)//cycle complete
            historicThroughput[] += throughputforcurrentcycle
            UVLAccheckAccuracy(historicThroughput)
            if(trainingmode==true)
                UVLAttraining()
            else
                UVLAcaculatehandover()
        else if(CurrentPosition contained in Route.ExistingRoute)
            UVLAcaculatehandover()
        else // coords will form a new route
            if(start of new route)
                create Route newRoute
                newRoute.StartofRoute = CurrentPosition
                newRoute.ExistingRoute[] += CurrentPosition
            if(trainingmode==false)
                trainingmode = true
            UVLAcaculatehandover()
            else if(trainingmode==true)
                UVLAcaculatehandover()

Routine::UVLAcaculatehandover()
    foreach(AP)
        param float[] normalisedmetric
        foreach(performancemetric)
            float metric = GetPerformanceMetric()
            normalisedmetric[] = NormaliseMetric(metric)

        param activationValue = (weights[0]*normalisedmetric[0])+.....
        if(activationValue>threshold)
            implementhandover(AP) // neuron fires

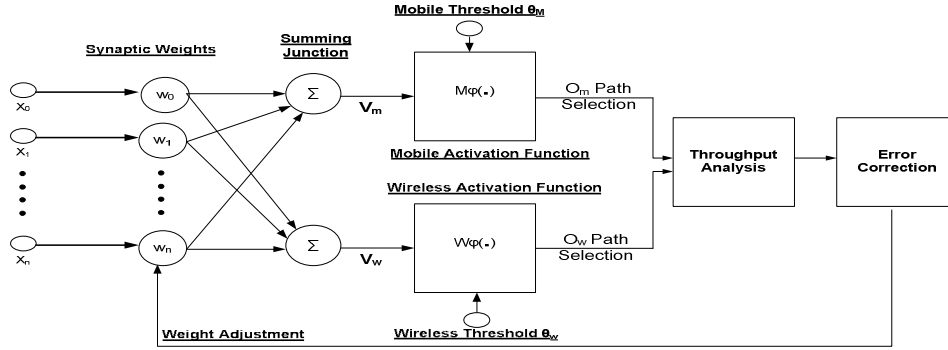
Routine:: UVLAttraining ()
    param slope = slopeofLinearRegression(HistoricThroughput)
    param errorCorrection = slope*learningRate
    foreach(weight)
        weight+=errorCorrection // alter weights

Routine:: UVLAccheckAccuracy()
    Param slope = slopeofLinearRegression(HistoricThroughput)
    if(abs(slope)>accuracyThreashold)
        trainingmode = true //reinititae training

```

The ULVA model consists of X_0, X_1, \dots, X_n neuron inputs corresponding to the selected performance metrics. The number of neurons in the output layer is dependent on the network configuration. Fig 2 illustrates a UVLA configuration with 2 output neurons for a heterogeneous network with wireless and mobile components.

Fig. 2 UVLA Neural Network with 2 Output Neurons



Each neuron is a linear threshold gate producing a binary output for path switchover. O_y is defined as follows:

$$V_y = \sum_{i=0}^N X_i W_i \quad (1)$$

$$O_y = \begin{cases} 1 & \text{if } V_y \geq \theta \\ 0 & \text{if } V_y < \theta \end{cases} \quad (2)$$

W_i are synaptic weights for each performance metric. V_y is the sum of weighted inputs. θ_y is a user configured activation threshold. If the stimulation of the neuron V_y meets the activation threshold θ_y , the neuron “fires” producing a binary positive output. A positive output indicates that path switchover should occur. The aim is to maximise throughput per route cycle, therefore we calculate the rate of change, c , of a linear regression line for historic throughput as follows:

$$c = \frac{\sum(x-x')(y-y')}{\sum(x-x')^2} \quad (3)$$

Using c we can determine the rate by which alterations to synaptic weights affect throughput. A positive c indicates that synaptic weight alterations have a beneficial effect on throughput and vice versa. A large c (positive or negative) indicates that ULVA requires numerous training cycles and vice versa.

In order to control the rate of learning we define a user configurable learning rate constant r . The selection of an appropriate learning rate is critical for the effective operation of the algorithm. Too low a learning rate makes the network learn very slowly. Too high a learning rate makes weights diverge, resulting in little learning. We define the error correction, ΔW , as the product of c and r .

$$\Delta W = c * r \quad (4)$$

5. An Experimental Analysis of a Commercial Heterogeneous Network Installation

In recent years heterogeneous networking has gained acceptance as the next logical step in wireless and mobile networking. The ITU have formalized this trend through the fourth generation wireless mobile networks (4G) set of standards. Many mobile operators are embracing heterogeneous networking. Initiatives such as the

TeliaSonera Homerun and British Telecom’s OpenZone have made heterogeneous networking a reality. In this section we analyze the performance characteristics of one such commercial deployment. Fig 3 illustrates the deployment of APs in Belfast city centre for a tourist bus route. Using NetStumbler [15] we record RSS for all APs as illustrated in fig 4.

Fig. 3 AP Deployment Belfast City Centre



Fig. 4 RSS for APs in Belfast City Centre

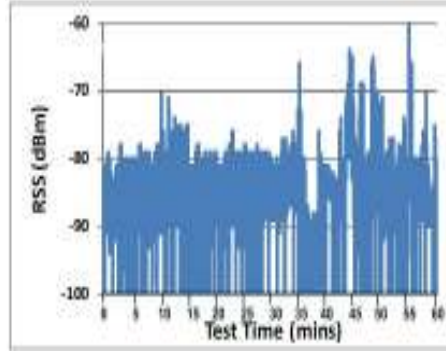
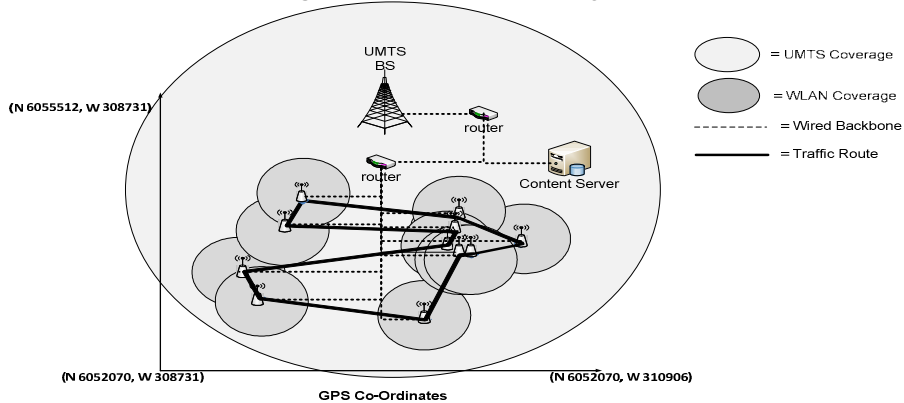


Figure 4 illustrates the RSS received by a MN located on the bus as it follows the route outlined in Fig 3 during a 57 minute circular trip. The average RSS over the duration of the test was -85.24dBm. Figs 3 and 4 illustrate a higher concentration of APs in the city centre. In the city centre the average RSS experienced by the MN was -83.23dBm while in the suburbs the average RSS was -90.2dBm. We use the results of this experimental study as input to the simulated model in Section 5.

5. Simulated Evaluation of the TRAWL Algorithm

In this section, we evaluate the performance of TRAWL for the network configuration described in Section 4 using NS2 with the MIH mobility package from NIST [2]. In order to integrate the geographical location of the route in NS2, we record the GPS coordinates for all junctions. Using these coordinates we simulate a circular route of 4.82km which is traversed in 57 minutes. Fig 5 illustrates the topology of the model.

Fig. 5 Simulated Network Configuration



Each AP has a transmit power of 0.281838W, transmit antenna gain of 1, receive antenna gain of 1 and an antenna height of 1.5M. This provides an outdoor signal range of approx 250M. The MIH parameters CStresh (link detection) and RXThresh (link utilisation) were set to -90dBm and -85dBm respectively. Simulation enhancements as described in [16] were included in the model. The UMTS core network was configured with a 622Mbit link capacity and a delay of 15ms. The WLAN back haul network was configured with a 100MBit capacity and a 1ms delay. The transport layer mobility protocol SCTP was used to implement network mobility. FTP data was then transmitted from the MN towards a back end content server.

We evaluate the performance of TRAWL for the configuration illustrated in fig 5 with/without congestion for homogeneous WLAN and heterogeneous WLAN/3G network configurations. In all cases we define the initial weights $w_1=.35$, $w_2=.3$, $w_3=.4$, $w_4=.25$ corresponding to the performance metrics loss rate, RTT, RSS and link bandwidth. These weights reflect our initial understanding of the relative importance of each performance metric. The value range .25 to .4 was chosen since values below this size did not result in the stimulation of neurons and hence no path selection occurred. Figs 6, 8, 10 and 11 illustrate the total throughput and synaptic weights for each of the learning cycles. Figs 7, 9, 11 and 13 illustrate the throughput, rate of learning and error correction for each of the learning cycles. Tables 2-5 are based on figs 6-13 and provide a more detailed view of the configuration of TRAWL parameters for each learning cycle.

Local maxima can have a negative effect on ANN performance as learning is centred on a local maximum value. Table 1 illustrates the TRAWL parameters for learning cycles 5-10 for the non congested heterogeneous network. After 9 cycles the slope of the throughput linear regression is 0 indicating that no error correction is required. The synaptic weight configuration $w_1=.575$, $w_2=.518$, $w_3=.590$, $w_4=.516$ has been optimised around a local maximal value. In order to avoid local maxima we introduce a positive or negative random weight adjustment in the range 0-0.2 every 5 learning cycles. In cycle 10 the random weight adjustments $w_1=-.112$, $w_2=-.181$, $w_3=-.069$, $w_4=-.145$ are applied avoiding the local maxima and resulting in the final trained throughput of 74.4Mbytes.

Table 1. Local Maxima in a non congested heterogeneous network configuration

| Cycle | w1 (Loss) | w2 (RTT) | w3 (RSS) | w4 (Bw) | Thres | Throughput | Slope | Error Correction | Learning Rate |
|-------|--------------|-------------|-------------|------------|-------|------------|--------|---------------------|------------------|
| 5 | 0.582 | 0.525 | 0.597 | 0.523 | 1 | 49.15 | 5.141 | 0.010 | 0.002 |
| 6 | 0.592 | 0.535 | 0.607 | 0.533 | 1 | 49.15 | -3.939 | -0.008 | 0.002 |
| 7 | 0.585 | 0.528 | 0.600 | 0.526 | 1 | 49.15 | -3.89 | -0.008 | 0.002 |
| 8 | 0.577 | 0.520 | 0.592 | 0.518 | 1 | 49.15 | -0.928 | -0.002 | 0.002 |
| 9 | 0.575 | 0.518 | 0.590 | 0.516 | 1 | 49.15 | 0 | 0.000 | 0.002 |
| 10 | 0.463 | 0.337 | 0.521 | 0.371 | 1 | 59.38 | 2.046 | 0.004 | 0.002 |

Fig. 6 Throughput and Synaptic Weights for a Non Congested Homogenous Configuration

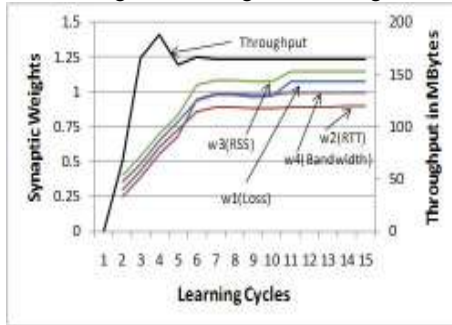


Fig. 7 Throughput and Error Correction for a Non Congested Homogenous Configuration

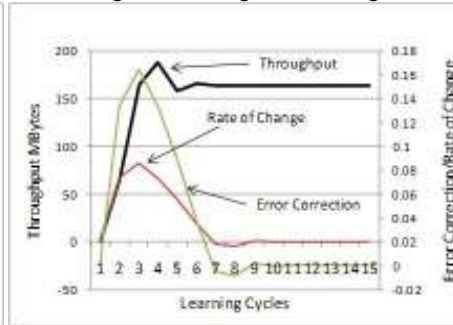


Fig. 8 Throughput and Synaptic Weights for a Non Congested Heterogeneous Configuration

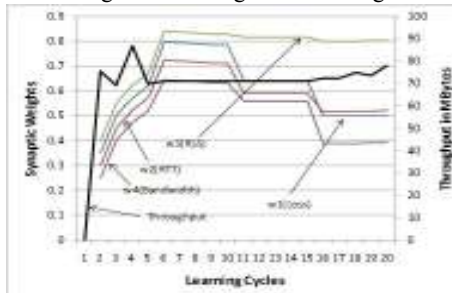


Fig. 9 Throughput and Error Correction for a Non Congested Heterogeneous Configuration

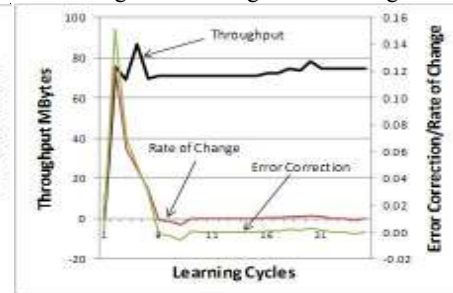


Fig. 10 Throughput and Synaptic Weights for a Congested Heterogeneous Configuration

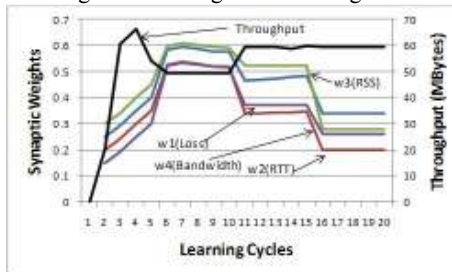


Fig. 11 Throughput and Error Correction for a Congested Heterogeneous Configuration

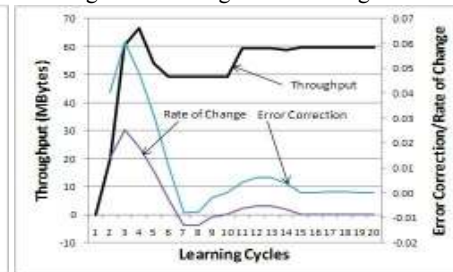


Fig. 12 Throughput and Synaptic Weights for a Congested Homogeneous Configuration

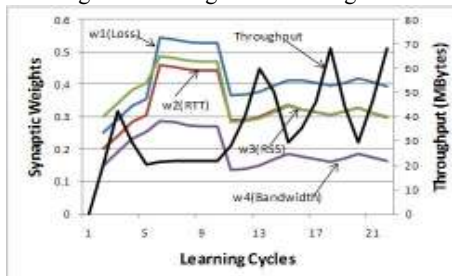


Fig. 13 Throughput and Error Correction for a Congested Homogeneous Configuration



Table 2. TRAWL Parameters for a Non Congested Homogenous Configuration

| Cycle | w1 (Loss) | w2 (RTT) | w3 (RSS) | w4 (Bw) | Thres | Throughput | Rate of Change | Error Correction | Learning Rate |
|-------|-----------|----------|----------|---------|-------|------------|----------------|------------------|---------------|
| 1 | 0.350 | 0.300 | 0.400 | 0.250 | 1 | 66.82 | 66.82 | 0.13364 | 0.002 |
| 2 | 0.484 | 0.434 | 0.534 | 0.384 | 1 | 164.9 | 82.45 | 0.1649 | 0.002 |
| 3 | 0.649 | 0.599 | 0.699 | 0.549 | 1 | 188.03 | 66.217 | 0.132434 | 0.002 |
| 4 | 0.781 | 0.731 | 0.831 | 0.681 | 1 | 158.49 | 43.819 | 0.087638 | 0.002 |
| 5 | 0.939 | 0.853 | 1.045 | 0.948 | 1 | 166.06 | 19.207 | 0.038414 | 0.002 |
| 6 | 0.977 | 0.891 | 1.083 | 0.986 | 1 | 164.16 | -2.345 | -0.00469 | 0.002 |
| 7 | 0.972 | 0.886 | 1.078 | 0.981 | 1 | 164.15 | -4.209 | -0.008418 | 0.002 |
| 8 | 0.964 | 0.878 | 1.070 | 0.973 | 1 | 164.16 | 0.943 | 0.001886 | 0.002 |
| 9 | 0.966 | 0.880 | 1.072 | 0.975 | 1 | 164.16 | -0.38 | -0.00076 | 0.002 |
| 10 | 1.071 | 0.889 | 1.144 | 0.996 | 1 | 164.18 | 0.005 | 0.000010 | 0.002 |
| 11 | 1.071 | 0.889 | 1.144 | 0.996 | 1 | 164.18 | 0.008 | 0.000016 | 0.002 |
| 12 | 1.071 | 0.891 | 1.144 | 0.996 | 1 | 164.18 | 0.006 | 0.000012 | 0.002 |
| 13 | 1.071 | 0.893 | 1.144 | 0.996 | 1 | 164.18 | 0.004 | 0.000008 | 0.002 |
| 14 | 1.071 | 0.895 | 1.144 | 0.996 | 1 | 164.18 | 0 | 0.000000 | 0.002 |

Table 3. TRAWL Parameters for a Non Congested Heterogeneous Configuration

| Cycle | w1 (Loss) | w2 (RTT) | w3 (RSS) | w4 (Bw) | Thres | Throughput | Rate of Change | Error Correction | Learning Rate |
|-------|-----------|----------|----------|---------|-------|------------|----------------|------------------|---------------|
| 1 | 0.350 | 0.300 | 0.400 | 0.250 | 1 | 75.44 | 75.44 | 0.150880 | 0.002 |
| 5 | 0.796 | 0.722 | 0.839 | 0.643 | 1 | 70.95 | -0.847 | -0.001694 | 0.002 |
| 10 | 0.641 | 0.591 | 0.817 | 0.560 | 1 | 70.89 | -0.012 | -0.000024 | 0.002 |
| 15 | 0.498 | 0.517 | 0.799 | 0.388 | 1 | 72.33 | 0.288 | 0.000576 | 0.002 |
| 20 | 0.568 | 0.530 | 0.991 | 0.528 | 1 | 74.4 | 0.742 | 0.001484 | 0.002 |
| 25 | 0.568 | 0.530 | 0.991 | 0.528 | 1 | 74.4 | 0 | 0.000000 | 0.002 |

Table 4. TRAWL Parameters for a Congested Heterogeneous Configuration

| Cycle | w1 (Loss) | w2 (RTT) | w3 (RSS) | w4 (Bw) | Thres | Throughput | Slope | Error Correction | Learning Rate |
|-------|-----------|----------|----------|---------|-------|------------|-------|------------------|---------------|
| 1 | 0.25 | 0.2 | 0.3 | 0.15 | 1 | 20.2 | 20.2 | 0.040 | 0.002 |
| 5 | 0.582 | 0.525 | 0.597 | 0.523 | 1 | 49.15 | 5.141 | 0.010 | 0.002 |
| 10 | 0.463 | 0.337 | 0.521 | 0.371 | 1 | 59.38 | 2.046 | 0.004 | 0.002 |
| 15 | 0.340 | 0.200 | 0.279 | 0.261 | 1 | 59.47 | 0.063 | 0.000 | 0.002 |
| 19 | 0.340 | 0.200 | 0.279 | 0.261 | 1 | 59.47 | 0.063 | 0.000 | 0.002 |

Table 5. TRAWL Parameters for a Congested Homogenous Configuration

| Cycle | w1 (Loss) | w2 (RTT) | w3 (RSS) | w4 (Bw) | Thres | Throughput | Slope | Error Correction | Learning Rate |
|-------|-----------|----------|----------|---------|-------|------------|---------|------------------|---------------|
| 1 | 0.25 | 0.2 | 0.3 | 0.15 | 1 | 20.2 | 20.2 | 0.040 | 0.002 |
| 5 | 0.542 | 0.457 | 0.485 | 0.286 | 1 | 21.52 | -1.904 | -0.004 | 0.002 |
| 10 | 0.365 | 0.288 | 0.284 | 0.137 | 1 | 28.11 | 1.309 | 0.003 | 0.002 |
| 15 | 0.411 | 0.321 | 0.322 | 0.177 | 1 | 35.118 | -4.1064 | -0.008 | 0.002 |
| 20 | 0.406 | 0.311 | 0.310 | 0.174 | 1 | 46.4 | -3.829 | -0.008 | 0.002 |
| 21 | 0.394 | 0.297 | 0.299 | 0.162 | 1 | 67.89 | 19.265 | 0.039 | 0.002 |

In Section 2 we described the mechanism used in [2] to implement the MIH LGD event. Table 6 compares the performance of this approach against a fully trained TRAWL implementation.

Table 6. Performance Comparison of NS2 MIH LGD and Fully Trained TRAWL LGD

| | NS2 MIH LGD Throughput (MB) | TRAWL LGD Throughput (MB) | Performance improvement |
|------------------------------------|--------------------------------|------------------------------|----------------------------|
| Non Congested/Homogenous | 41.65 | 164.18 | 394% |
| Non Congested/Heterogeneous | 71.96 | 74.4 | 4% |
| Congested/Homogenous | 42.4 | 59.47 | 40% |
| Congested/Heterogeneous | 44.98 | 67.89 | 51% |

Table 6 illustrates that when TRAWL is fully trained it has significantly better performance than [2] for MIH LGD event triggering, particularly for a non congested homogenous WLAN configuration. In order to explain the significant performance differential we consider fig 4 in more detail. The average RSS was -85.24dBm, ranging from -55 to -100 dBm. -85dBm is considered a point of significant performance degradation in WLAN. TRAWL, through back propagation, determines that for a non congested network, handover at -85dBm does not yield optimal throughput.

For an unsupervised ANN such as TRAWL, learning is performed online. It is critical therefore that the training of synaptic weights is performed with minimal delay. The optimisation of synaptic weights is a trade off between learning time and throughput. Fig 6, fig 7 and table 2 illustrate that while 14 cycles were required, within 2 cycles, the output of TRAWL is within 1% of its trained value. Table 7 compares the performance of a partially trained TRAWL, 2 learning cycles, against [2].

Table 7. Comparison of NS2 MIH LGD and Partially Trained TRAWL LGD (2 Cycles)

| | NS2 MIH LGD Throughput (MB) | TRAWL LGD Throughput (MB) | Performance improvement |
|------------------------------------|--------------------------------|------------------------------|----------------------------|
| Non Congested/Homogenous | 41.65 | 164.9 | 396% |
| Non Congested/Heterogeneous | 71.96 | 69.19 | -4% |
| Congested/Homogenous | 42.4 | 42.09 | <1% |
| Congested/Heterogeneous | 44.98 | 60.28 | 134% |

Table 7 illustrates that after only 2 learning cycles TRAWL has at least equivalent performance to [2]. In the non congested homogeneous and congested heterogeneous configurations TRAWL has significantly outperformed [2] after 2 learning cycles.

5. Conclusions and Future Work

In this paper we proposed TRAWL, a feed forward neural network, which uses the cyclical nature of public transport routes in order to optimise MIH link triggering. TRAWL uses an unsupervised back propagation learning mechanism which captures predictable network behaviour while also considering dynamic performance characteristics. We evaluate TRAWL using performance metrics from a commercial

heterogeneous network. Results presented illustrate up to a 400% performance improvement over traditional MIH link triggering approaches for this network installation. For an unsupervised ANN such as TRAWL the optimisation of synaptic weights is a trade off between learning time and throughput. While a number of cycles are required to fully train TRAWL, we illustrate that early in the training cycle throughput is close to its final trained value.

Future work will extend the TRAWL algorithm to consider metrics applicable to media streaming applications.

References

- [1] Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and metropolitan area networks – Part 21: Media Independent Handover Services", LAN/MAN Standards Committee of the IEEE Computer Society, 21 Jan. 2009.
- [2] NIST, "The Network Simulator NS-2 NIST Add-on – Neighbor Discovery", January 2007
- [3] V. Mhatre, K. Papagiannaki, "Using Smart Triggers for Improved User Performance in 802.11 Wireless 24 Networks", ACM Mobisys'06, pp.246-259, 2006.
- [4] S. Woon, N. Golmie, Y.A. Sekercioglu, "Effective Link Triggers to Improve Handover Performance", IEEE PIMRC06, pp.1-5, 2006
- [5] UC Berkeley, LBL, UCS/ISI, Xerox Parc (2005). NS-2 Documentation and Software, Version 2.29. <http://www.isi.edu/nsnam/ns>
- [6] K.R Kumar, P. Angolkar, D. Das, "SWiFT: A Novel Architecture for Seamless Wireless Internet for Fast Trains", Ramalingam, R.Vehicular Technology Conference, Singapore, 2008. VTC Spring 2008. IEEE, pp. 3011-3015, May 2008
- [7] O. Hayoung, K. Chong-kwon "A Robust handover under analysis of unexpected vehicle behaviors in Vehicular Ad-hoc Network" pp. 1-7, Vehicular Technology Conference (VTC 2010-Spring), 2010
- [8] S. Céspedes U., X. (Sherman) Shen. "An Efficient Hybrid HIP-PMIPv6 Scheme for Seamless Internet Access in Urban Vehicular Scenarios", in Proc. IEEE Global Telecommunications Conference (GLOBECOM) 2010. Miami, USA.
- [9] E. Fallon, Y. Qiao, L. Murphy, G. Muntean, "SOLTA: a service oriented link triggering algorithm for MIH implementations", Proceedings of the 6th International Wireless Communications and Mobile Computing Conference, Caen, France, pp. 1146-1150, 2010
- [10] A. Ryo, K. Hiroyuki, S. Ryoichi, T. Tatsuro "Control of transmission timing using information on predicted movement in opportunistic roadside-to-vehicle communication" Proceedings of Ubiquitous and Future Networks (ICUFN) Jeju Island, Korea (South), pp. 262-267, 2010
- [11] I. Chantaksinopas, P. Oothongsap, A. Prayote "Framework for network selection transparency on vehicular networks" Proceedings of International Conference on Electrical Engineering/Electronics Computer Telecommunications and Information Technology (ECTI-CON), pp. 593-597, 2010
- [12] M. Hasegawa, K. Ishizu, H. Murakami, H. Harada "Experimental evaluation of distributed radio resource optimization algorithm based on the neural networks for Cognitive Wireless Cloud" Proceedings of IEEE 21st International Symposium on Personal, Indoor and Mobile Radio Communications Workshops (PIMRC Workshops), pp. 32-37 2010
- [13] M. Hasegawa, T. Takeda, G. Miyamoto, H. Tran, H. Murakami, K. Ishizu, S. Filin, H. Harada "Design and Implementation of A Distributed Radio Resource Usage Optimization Algorithm for Heterogeneous Wireless Networks" Proceedings of Vehicular Technology Conference Fall, pp. 1-7 2009
- [14] V. Rakovic, L. Gavrilovska, "Novel RAT selection mechanism based on Hopfield neural networks" Proceedings of International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2010
- [15] Netstumbler Network Analysis Software, <http://www.netstumbler.com/>
- [16] C. Palazzi, B. Chin, P. Ray, G. Pau, M. Gerla, M. Rocchetti "High Mobility in a Realistic Wireless Environment: a Mobile IP Handoff Model for NS-2" 3rd International Conference on Testbeds and Research Infrastructure for the Development of Networks and Communities, 2007. TridentCom 2007.