RHUL MikeNet Neural Network Simulator

This file contains an overview of the files enclosed within RHUL_HSModels and the objectives achieved during the implementation of the RHUL MikeNet Neural Network Simulator.

Objective 0: Software needed for simulations & set up

The neural network simulation environment Mikenet produced by Mike Harm which can be used to implement the reading models reported in both Harm & Seidenberg, 2004 and Harm & Seidenberg, 1999 is now installed on an RHUL machine within the Rastle Lab. Instructions for its use can be found in the README files within README_MikenetInstall.txt and folders RHUL_HS04 and RHUL_HS99.

- Mikenet-v8.0.zip contains all necessary files required to install the Mikenet neural network simulation environment

Objective 1: Implementation of Harm & Seidenberg, 2004 (HS04)

HS04_Model4Replication (see RHUL_HSModels/RHUL_HS04/HS04_Model4Replication on RHUL Rastle Lab machine or within RHUL_HSModels.zip) is an implementation of Harm & Seidenberg, 2004 installed on the RHUL Rastle Lab machine. For details of the model and instructions for use refer to README_HS04_M4R.txt.

- RHUL_HS04 contains a copy of HS04_Model4Replication files. To run the model on any system on which MikeNet is already installed, unzip the folder and follow README_HS04_M4R.txt file instructions to compile and run the necessary code.

Objective 2: Train HS04 model using basic parameters and monosyllabic training set

HS04_Model4Replication/Training provides the files needed to train instantiations of the HS04 model. A trained version (weight matrix) is stored in HS04_Model4Replication/Testing as the file weights_after_4000000_trials.zip. Four trained instantiations of the model can also be found in the folder Simulations/HS04_M4R_TrainedNetworks on the RHUL machine (these were used to generate the results reported in this document) or HS04_M4R_TrainedNetworks.zip. These networks were trained on a monosyllabic training set using 'basic' parameters, i.e. parameter set derived from pilot simulations used to find parameterization capable of replicating HS04 training task performance (see README_HS04_M4R.txt & REAME_training.txt for further details). The corpus and pattern generation files used to produce the training materials can be found in HS04_Model4Replication/Patterns (see README_Patterns.txt for more detail).

Objective 3: Evaluate HS04 model

See subheadings below for details of how differing properties of the HS04 model were evaluated

Objective 3.a.: Evaluate HS04 training task performance

Scripts that allow production of measures used in HS04 to analyse model performance can be found in HS04_Model4Replication/Testing/Analyse_Training_Tasks (see README_testing.txt & README_Analyse_TrainingTasks.txt for further details and instructions for use). An overview of the development of training task performance for the model across 6 million training trials can be found in HS04_TrainingTaskPerformance.pdf. The charts display performance of 4 instantiations of the model (M - P in HS04_M4R_TrainedNetworks) each initiated with a different random seed. Networks were trained on 3 million pre-literate training trials followed by 3 million intermixed literacy and pre-literacy training trials (see README_HS04_M4R.txt & README_training.txt for further details). Model performance first reached or exceeded that reported in HS04 at 4 million trials, therefore the weight matrices for these networks were used for further model evaluation.

Objective 3.b.: Test model's generalization behaviour in terms of responses and settling times (Non-word reading: HS04 simulation 4)

To evaluate the model's ability to generalise beyond the training set the model was tested on its ability to read non-words as in HS04 simulation 4 (pg 26). As in HS04 the model was tested on its ability to read 86 nonwords from Glushko (1979) 43 of which were from consistent neighbourhoods and 43 from inconsistent neighbourhoods [Non-words from McCann & Besner, 1987 were not tested as the materials could not be found]. Our model read regular nonwords with accuracy 89.5% (σ = 0.045) compared to an accuracy reported in HS04 of 94%, while our model read irregular nonwords with accuracy 55.2% (σ = 0.048) compared to an accuracy reported in HS04 of 78%. Training networks on more reading trials (additional 5 million pre-literate and literacy training trials) did not significantly improve performance. Further, non-word reading performance did not improve significantly when the same model was trained on the same corpus but with frequency measures ranging from 1 – 0.1 (rather than 1 – 0.2).

The following files were used in this evaluation and allow the user to test networks on similar tasks (see README_testing.txt for further details):

- HS04_Test.c
- Analyse_HS04.R

Files within HS04_Model4Replication/Testing/RT provide the user with the ability to generate model reaction times/settling times. The accompanying RT_README.txt file provides an explanation for how RTs are derived and instructions for use.

Objective 3.c.: Quantify activation associated with a specific unit or layer

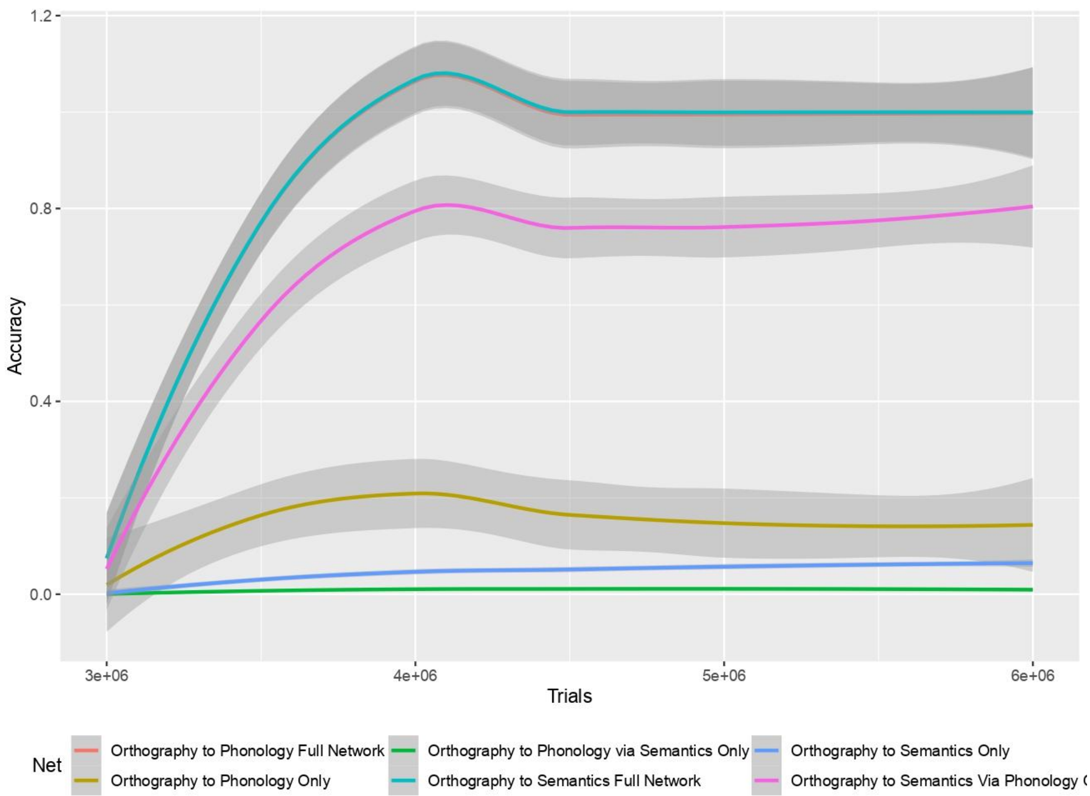
HS04_Test.c allows the user to output the activation of a specific unit or layer within the network on a range of tasks (see README_testing.txt for further details).

Objective 3.d.: Provide tools to generate division of labour measures (HS04 Division of labour simulations 6 & 7)

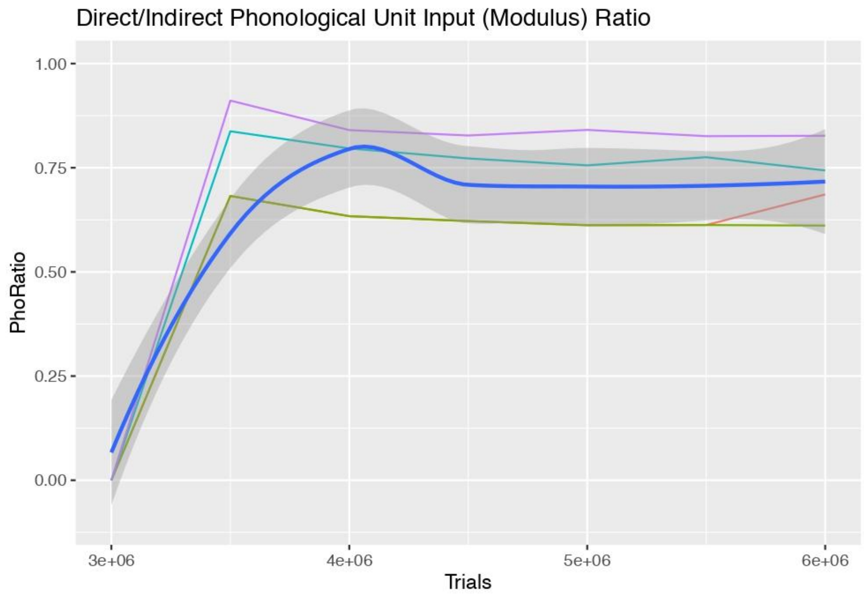
Within the folder HS04_Model4Replication/Testing/Division_of_Labour are two c scripts that allow the user to generate measures to examine the division of labour within the HS04 model. HS04_Test_NetLesion.c allows the user to run simulations within the triangle model with specific pathways lesioned/removed. While HS04_Test_Activation.c allows the user to output the mean quantity of activation entering a unit within a given layer of the network from a given path, thus allowing the production of measures as reported in HS04, Simulation 6, figure 12 and Smith, Monaghan & Huettig, 2021 (Division of labour subsection). For further details of how to use these scripts and what they do see README_DivisionOfLabour.txt.

Below we report the results of various division of labour analyses conducted using HS04_Test_NetLesion.c and HS04_Test_Activation.c and applied to the implementation of HS04 stored within HS04_Model4Replication.

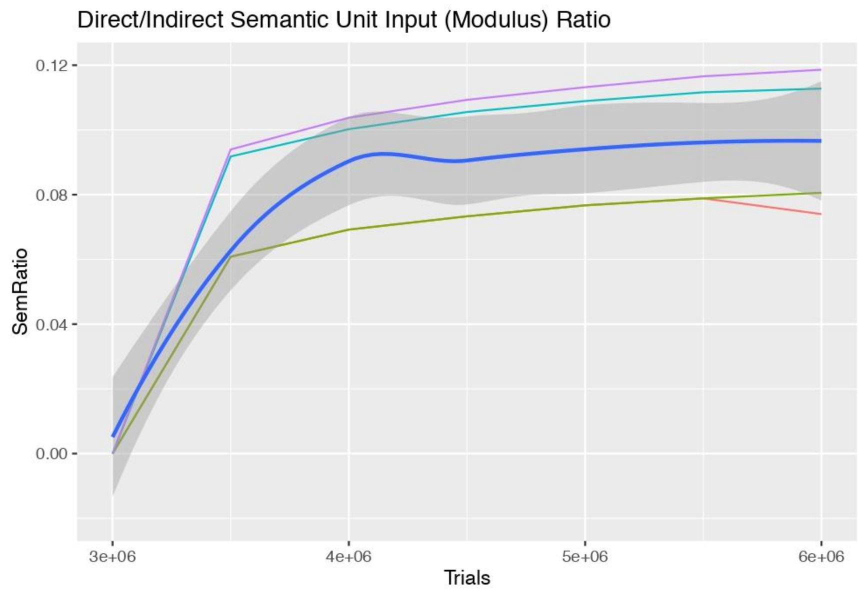
First, we first attempt to replicate HS04, Simulation 7, Figure 14 by reporting the model's ability to read words with paths lesioned. The following figure charts the accuracy (averaged over 4 instantiations of the model) in reading words while different paths in the network are lesioned at different stages across training. The chart shows performance across literacy training i.e. 3 million - 6 million training trials. The legend indicates the components that remain present in the lesioned networks (full network indicates a network without any paths lesioned). This figure can also be found in the Division_of_Labour/Charts folder titled Lesion_Accuracy_3mto6m.pdf.



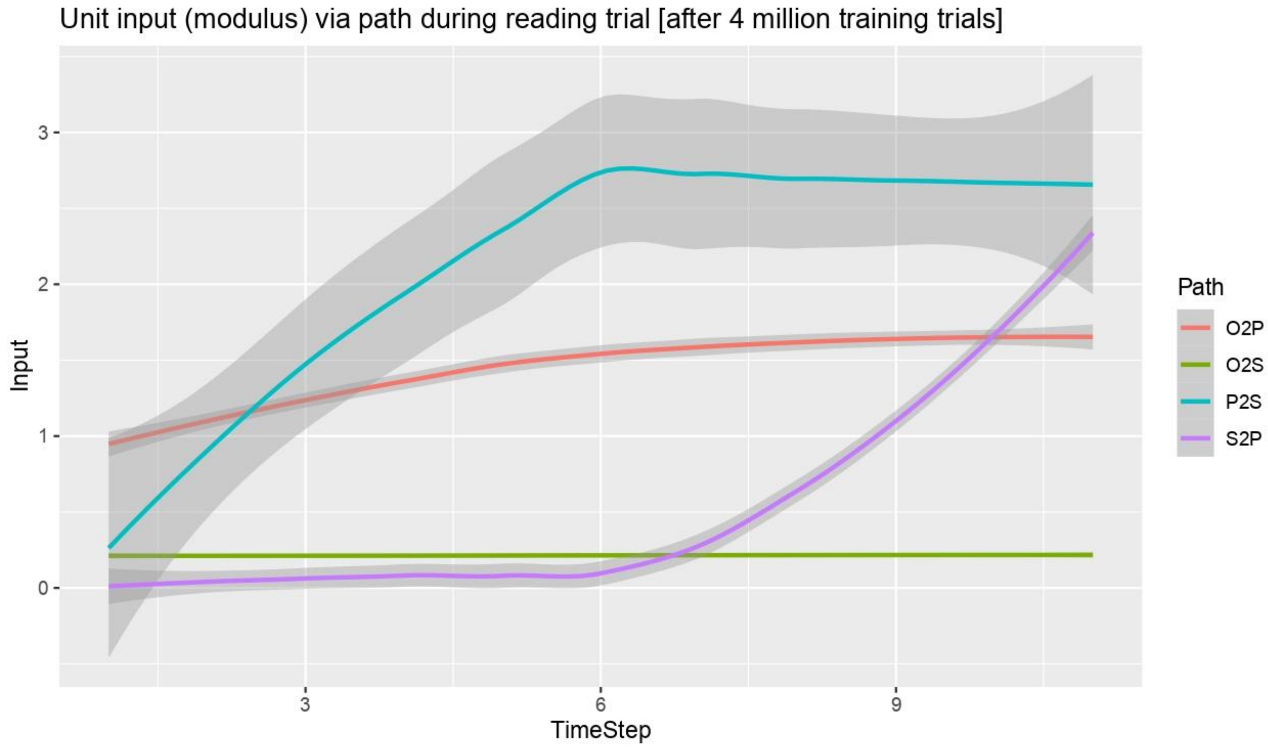
The following figure uses a method reported in Smith, Monaghan & Huettig, 2021 (Figures 16 & 19) to examine the change in the ratio of activation entering each output layer via either direct or indirect paths over the course of training.



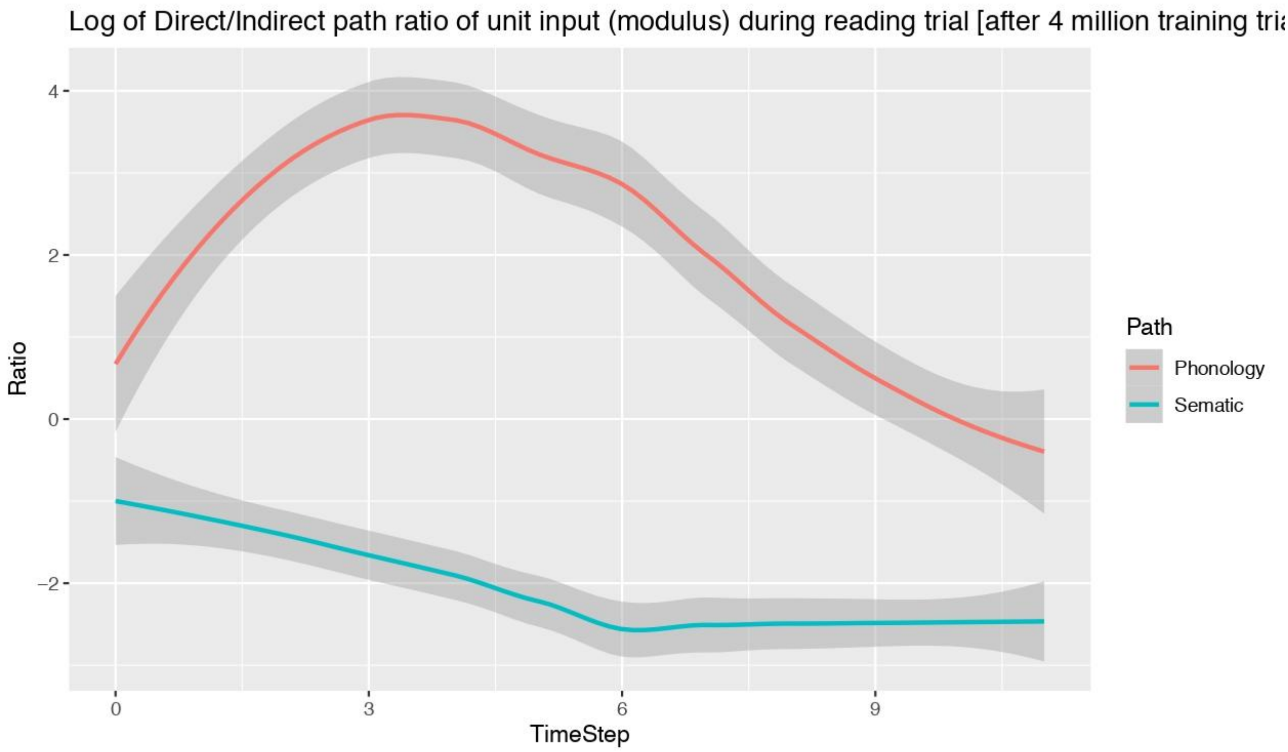
The PhoRatio is the mean activation entering a unit in the phonological layer directly (incl. via hidden layer) from orthography divided by the mean activation entering a unit in the phonological layer indirectly from orthography via the semantic layer (see HS04_ActivationRatio_Training_Pho.pdf).



Conversely, the SemRatio is the mean activation entering a unit in the semantic layer directly (incl. via hidden layer) from orthography divided by the mean activation entering a unit in the semantic layer indirectly from orthography via the phonological layer (see HS04_ActivationRatio_Training_Sem.pdf).



In an attempted replication of figure 12 from simulation 6 in HS04, this figure shows, as a test trial unfolds (time steps 0 - 11), the average modulus input to a unit in either the phonological or semantic layer via either the direct (orthography to semantics [O2S], orthography to phonology [O2P]) or indirect path (phonology to semantics [P2S], semantics to phonology [S2P]) for networks trained on 4 million training trials (averaged over 4 instantiations, tested on all words in training corpus, see HS04_UnitInputs_Trial_4m.pdf).



The above figure shows, as a test trial unfolds (time steps 0 - 11), the log ratio of average modulus input entering the phonological or semantic layer via either the direct path (orthography to semantics [O2S], orthography to phonology [O2P]) or indirect path (phonology to semantics [P2S], semantics to phonology [S2P]) for networks trained on 4 million training trials (averaged over 4 instantiations, tested on all words in training corpus). The ratio is structured direct / indirect and is a measure used to examine changes in distribution of labour within the triangle model in Smith, Monaghan & Huettig, 2021 (see HS04_LogInputRatio_Trial_4m.pdf).

Objective 3.e.: Analysis of representational similarity (analysis of hidden unit states)

Within the folder HS04_Model4Replication/Testing/RepresentationalSimilarity is an R script that allows the user to compare hidden layer states for comparisons of representational similarity. The user must first test a model using HS04_Test.c to generate a file reporting the activity of one of the HS04 network's hidden layers. The R script Analyse_RepSim_HS04.R can then be used to compare the hidden layer activity on each trial performed during the test. For further information refer to README_RepresentationalSimilarity.

(Optional) Objective 3.f.: Varying model training parameters (e.g. pre-literacy training)

See Objective 3.a.

(Optional) Objective 3.g.: Adding noise to input

Within the folder HS04_Model4Replication/Testing/Noise is the awk script NoisyPhoInput_PatGen.awk which can be used to generate noisy patterns from any corpus file that follows the same structure as 6k_AllReps_8Slot_NxF. The script allows for the application of different forms of noise to suit a range of research questions. The resulting pattern files can then be used to train and/or test the model in noisy environments. See README_HS04_Noise for further details.

(Optional) Objective 3.h.: Adding noise to network connections

Mikenet also offers a number of built in methods that allow the user to introduce noise to the input pattern, to the target and/or within the network, as activation passes between specified layers. Details of these methods and how to use them can be found in the file README_HS04_Noise.

(Additional) Objective 4: Implementation of Harm & Seidenberg (1999)

Enclosed within the package RHUL_HSModels is the folder RHUL_HS99 which contains a version of the Harm & Seidenberg, 1999 model. A simplified reading model that learns to map from orthography to phonology, but does not contain any semantic components. For further details and instructions for use see README_HS99.