

Development & Optimization of Machine Learning Algorithms & Models of Relevance to Large Databases & Homeland Security Counterterrorism Efforts at START

November 22, 2016, SMA Lecture Series

Principal Investigator: Prabhakar Misra

Research Associates: Raul Garcia-Sanchez and Daniel Casimir

Department of Physics & Astronomy, Howard University, Washington, DC

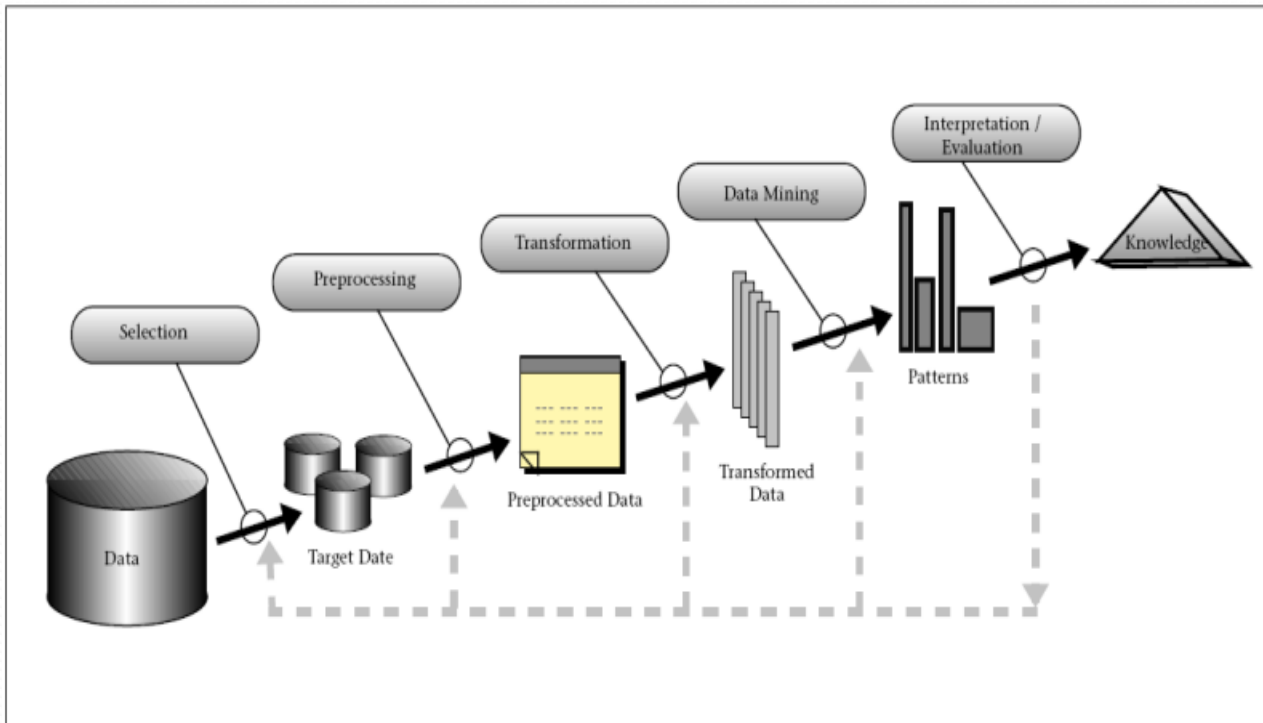
&

START, University of Maryland, College Park, MD

Overview

- Knowledge Discovery through Databases (KDD)
- Profiles of Incidents Involving CBRN by Non-state actors (POICN) database
- Patterns of CBRN Use Among Non-state Actors Analysis (Inside-out regression)
- Neural Networks using Global Terrorism Database (GTD)
- Logistic Perceptron, Hopfield Network & Heat Maps
- Summary & Conclusions
- Future Work
- Acknowledgments

Knowledge Discovery through Databases (KDD)



1. Goals/Objectives of KDD process
Converting the goals and requirements of the end-user into a “data-mining” goal.
2. Target Data set creation
Initial stage, data collection
Familiarization with the Data
**POICN Training/Orientation*
3. Pre-processing of data
Creation of the final data set from the raw data to be used in the model,
Noise removal, missing data entries, etc.
4. *Transformation
Developing a useful representation of the target data.
Reduction of the number of effective variables input into the predictive model.
Possible tools include neural networks, cluster analysis...
5. *Choice of Data Mining Method
Clustering, Summation, Dependency Modeling, Classification, Change and Deviation Detection.

Machine Learning Data Mining Methods

- Supervised, unsupervised, and ensemble learning algorithms.
- Input data matrix, (default, each row is used for one observation, and one column for a variable)
- The response data by default are in column vectors. (If the output Y, is for a regression model, Y must have numeric values. For classification, Y can be numeric, categorical, character, or logical)

Machine Learning Data Mining Tools

Supervised Learning

- **Linear Regression**
- Nonlinear Regression
- Generalized Linear Models
- Classification Trees
- Support Vector Machines
- Discriminant Analysis
- Naïve Bayes Classification
- Nearest Neighbors
- Model Building & Assessment

Unsupervised Learning

- Hierarchical Clustering
- K-Means Clustering
- Gaussian Mixture Models
- Hidden Markov Models
- Cluster Evaluation

<http://www.mathworks.com>

Profiles of Incidents involving CBRN by Non-state Actors (POICN) Database

- Relational database of terror events that focuses on the involvement of Chemical Biological Radiological & Nuclear (CBRN) weapons.
- Ranges from 1990 – 2011
- ~471 total events
- Hundreds of variables of sorted by various categories. e.g. **Event Location**: {Attack Country, Attack Region,...}, **Actor Information**: {Perp Group Name, Perpetrator Type,...}, etc.
- **Inclusion of variables to handle doubt, and uncertainty related to any of the included in events. For example **credibility**, and **objectivity** variables dealing with event source information.*

POICN Representative Sample

RussiaNIS	MiddleEast	SouthAsia	is.bio	is.chem	perp.lone	perp.relex	perp.cult	perp.ethnonat
0	0	0	0	1	0	1	0	0
0	0	0	0	0	0	1	0	0
0	0	0	0	1	0	0	1	0
0	0	0	1	1	0	1	0	0
0	0	0	1	1	0	0	0	0
0	1	0	1	1	0	1	0	0
0	0	0	1	1	0	1	0	0
0	0	0	1	1	0	1	0	0
0	0	1	0	1	0	1	0	0
0	0	1	0	1	0	1	0	0
0	0	0	1	1	0	1	0	0
0	0	1	0	1	0	0	0	0

Based on 2015 POICN data, matrix arranged in binary format (0,1)

CBRN Pattern Use by Non-State Actors

- Logistic regression, and unique “inside out” variation of logistic regression, using relations among variables in order to profile the input cases.
- Output/Dependent variable are two choices: “seeking a CBRN weapon” denoted as Type **A** event, and “Possession of a CBRN weapon” denoted as Type **B** event.
- Type A Events = {*protoplots, plots, attempted acquisition, possession of a non-weaponized agent*}
- Type B Events = {*possession of a weapon, threat with possession, attempted use of a weapon, use of a CBRN during an attack*}
- Subcategories in POICN (Codebook), Chapter. V. “Event Information”, Subsection C. “EVENT_TYPE” categorical variable. (e.g. protoplot = 0, plot = 1, possession of a weapon = 4,...)

“Patterns of CBRN Use by Non-State Actors: Analyzing the Evidence”, R. Breiger, P. Murray, and L. Pinson, Annual Conventional of the International Studies Association (ISA), 2013.

Logistic Regression

$$y = \beta_0 + \sum \beta_i X_i + \varepsilon_i$$

y = Response, output,
dependent variable, etc.

X_i = Predictors,
independent variables,
etc.

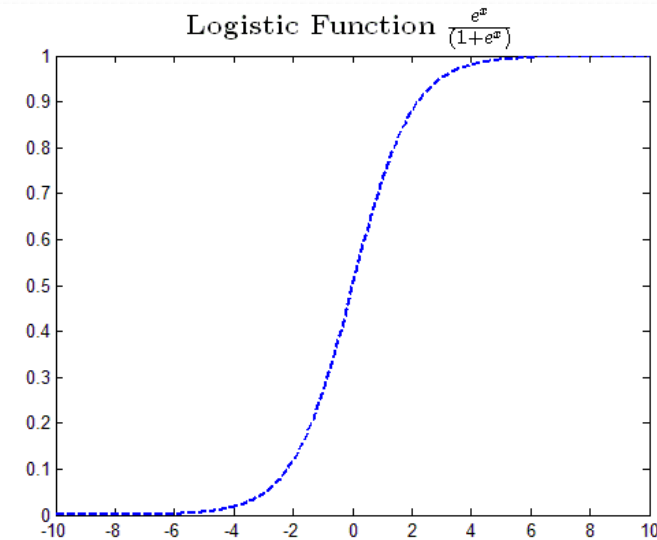
β_i = Fitted coefficients of
linear model

ε_i = Error, noise...

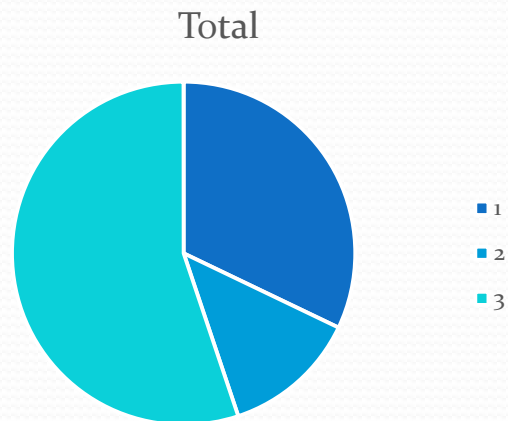
Special case of
linear regression.
Used for binary/nominal
discrete responses.
e.g. “success or failure”, or
“*Type B CBRN event or not*”.

Also, likewise for some of
the input predictors.

Y in this case is now the
“logit”, or log of the ratio of
the odds of success to the
odds of failure.



Target Data Subset



Row Labels	Count of CREDIBILITY
1	53
2	21
3	91
Grand Total	165

breakdown of
credibility levels of events from **1998-2011**

Data from *POICN 2.11 2014*

(**165** events were coded as 1, 2, or 3 in this case)

1 Event for 1998-2011 listed as -99

3 Events for 1998-2011 listed as -88

"Patterns of CBRN Use by Non-State Actors: Analyzing the Evidence"
R. Breiger, P. Murray, L. Pinson, Annual Convention of the International Studies Association, 2013

Logit Model Results (Cases from POICN 2.11 & 2.2)

Generalized Linear regression model:

$\text{logit}(\text{Var10}) \sim 1 + \text{Var1} + \text{Var2} + \text{Var3} + \text{Var4} + \text{Var5} + \text{Var6} + \text{Var7} + \text{Var8} + \text{Var9}$

Distribution = Binomial

Labels	Breiger et al's coefficient estimates	Our coefficient estimates
Intercept	-0.25	-4.0533
RussiaNIS	2.2	-0.084233
Middle East	1.35	0.30323
South Asia	3.06	0.26297
Bio	-0.24	0.33975
Chem	1.9579	0.86233
Lone Actor	-0.56	-0.52259
Religious Extremist	-2.58	0.25766
Cults	-2.49	-0.30323
Ethno nationalists	-2.06	-0.16353

“>> *GeneralizedLinearModel.fit()*” Matlab command

42 out of 165 observations were used for this sample display

POICN Summary

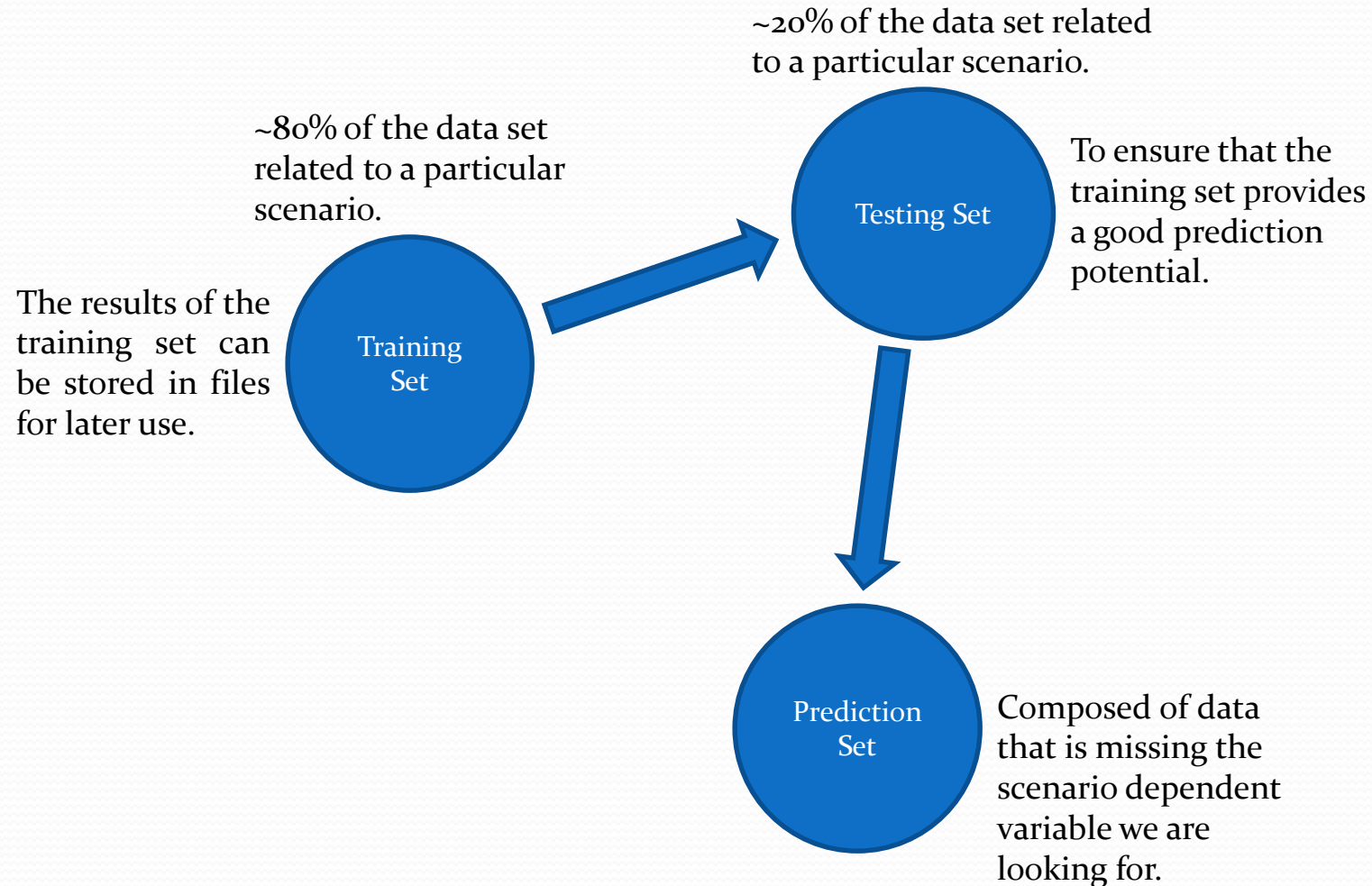
- Ran a binary logistic model with no interactions among predictors on current POICN data.
- Same 9 binary predictors and responses {Possession of CBRN = 1, Seeking CBRN = 0} for the 1998-2011 time frame that Breiger et al used.
- Reached the same conclusion as Breiger et al. regarding correlation with a Type B event for all but two of the variables [RussiaNIS, BIO] in our reproduction attempt.
- We continue efforts at reproducing Breiger et al's analysis, e.g. logit model with interactions, "inside out" method, data cleaning our POICN inputs, etc.
- Apply Neural Network techniques in this CBRN pattern of use analysis.

Global Terrorism Database (GTD)

Data Set Snapshot

iyear	imonth	iday	country	attacktypei	targetypei	targetsubtypei	weaptypei	weapsubtypei	ishostkid	gname
1973	1	7	233	1	14	69	5	5	o	Irish Republican Army (IRA)
1988	4	13	43	2	3	23	5	2	o	Manuel Rodriguez Patriotic Front (FPMR)
2012	11	19	97	3	14	76	6	11	o	Palestinians
1989	7	16	45	3	21	107	6	16	o	
2009	7	17	95	3	14	69	6	17	o	
1986	1	22	61	3	21	107	6	16	o	Farabundo Marti National Liberation Front (FMLN)
1980	6	26	200	2	2	20	5	2	o	Muslim Brotherhood
1994	4	30	177	2	14	75	5	2	o	Revolutionary United Front (RUF)
1992	5	8	137	1	1	9	5	2	o	Mozambique National Resistance Movement (MNR)
1987	6	24	145	3	21	107	6	16	o	Nicaraguan Resistance
1991	10	22	159	1	2	14	5	3	o	Shining Path (SL)
1982	8	29	233	1	4	33	6	17	o	Irish Republican Army (IRA)
1988	11	23	61	3	21	107	6	16	o	Farabundo Marti National Liberation Front (FMLN)
1997	7	14	186	2	3	22	5	5	o	Liberation Tigers of Tamil Eelam (LTTE)
1978	3	1	185	3	6	43	6	28	o	Basque Fatherland and Freedom (ETA)
1990	1	6	160	1	2	21	5	3	o	New People's Army (NPA)
1987	9	16	183	1	3	23	5	2	o	African National Congress (South Africa)
2004	12	30	95	3	2	15	6	15	o	
2012	12	31	95	3	15	86	6	15	o	Al-Qa'ida in Iraq
2009	3	7	4	3	1	12	6	17	o	Taliban
1988	3	7	121	3	1	11	6	16	o	
1991	10	11	159	2	4	29	5	2	o	Shining Path (SL)

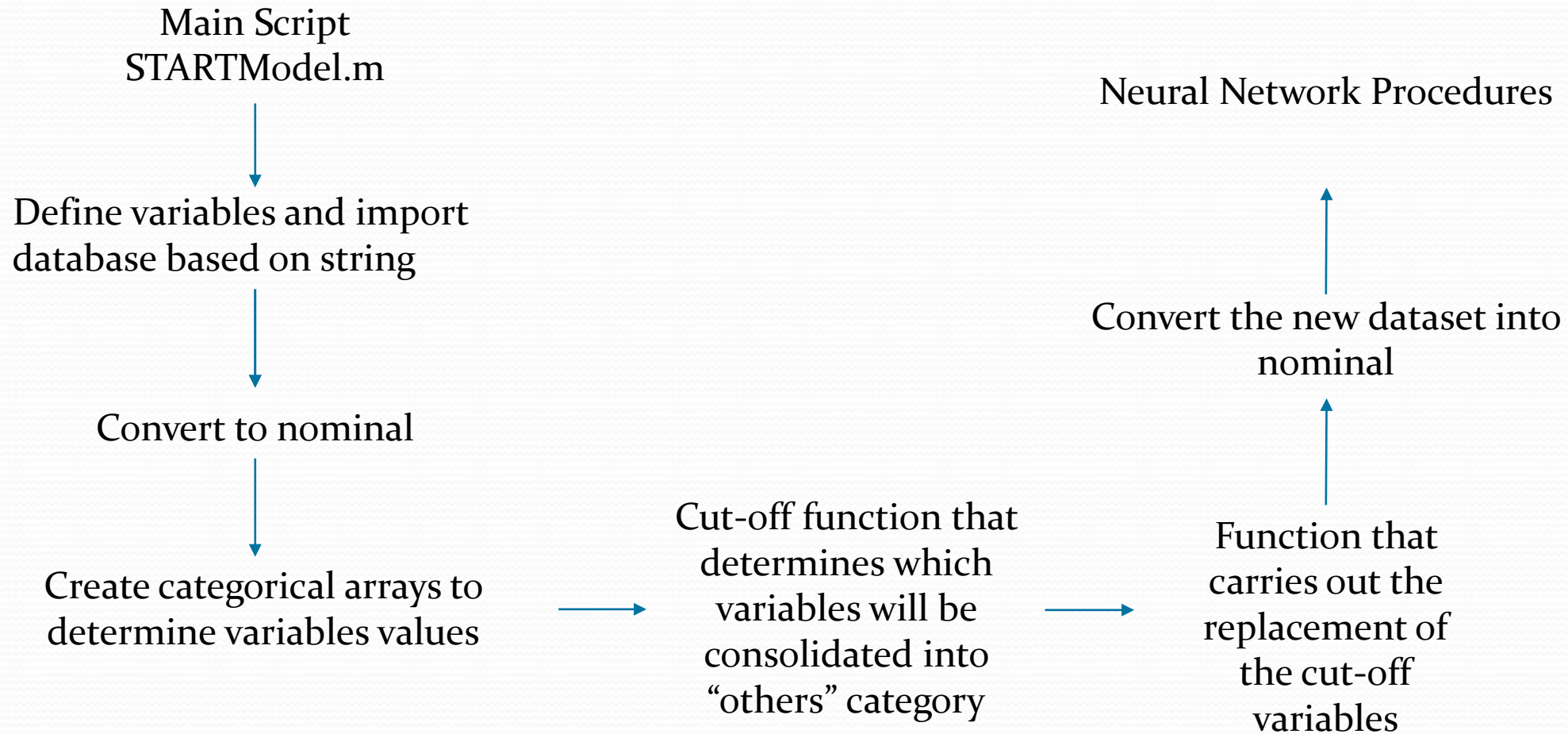
Neural Network Structure Data Sets



Neural Network Script

- Import Training/Test Set and Prediction Set from Excel Files.
- Prepare the Neural Network Data from the imported data.
- Partition the Training/Test for crossvalidation.
- Sections for each Neural Network or Machine Learning Model, encapsulated with an if(false), allows user to select desired model to use when running the script.
- Similar structure for all models:
 - Breakup the combined Training/Test Set into set partitions.
 - Use training set to generate a model.
 - Use network generated on test set, then compare the results with the actual correct value.
 - Once the above step is acceptable, use network with prediction set data.

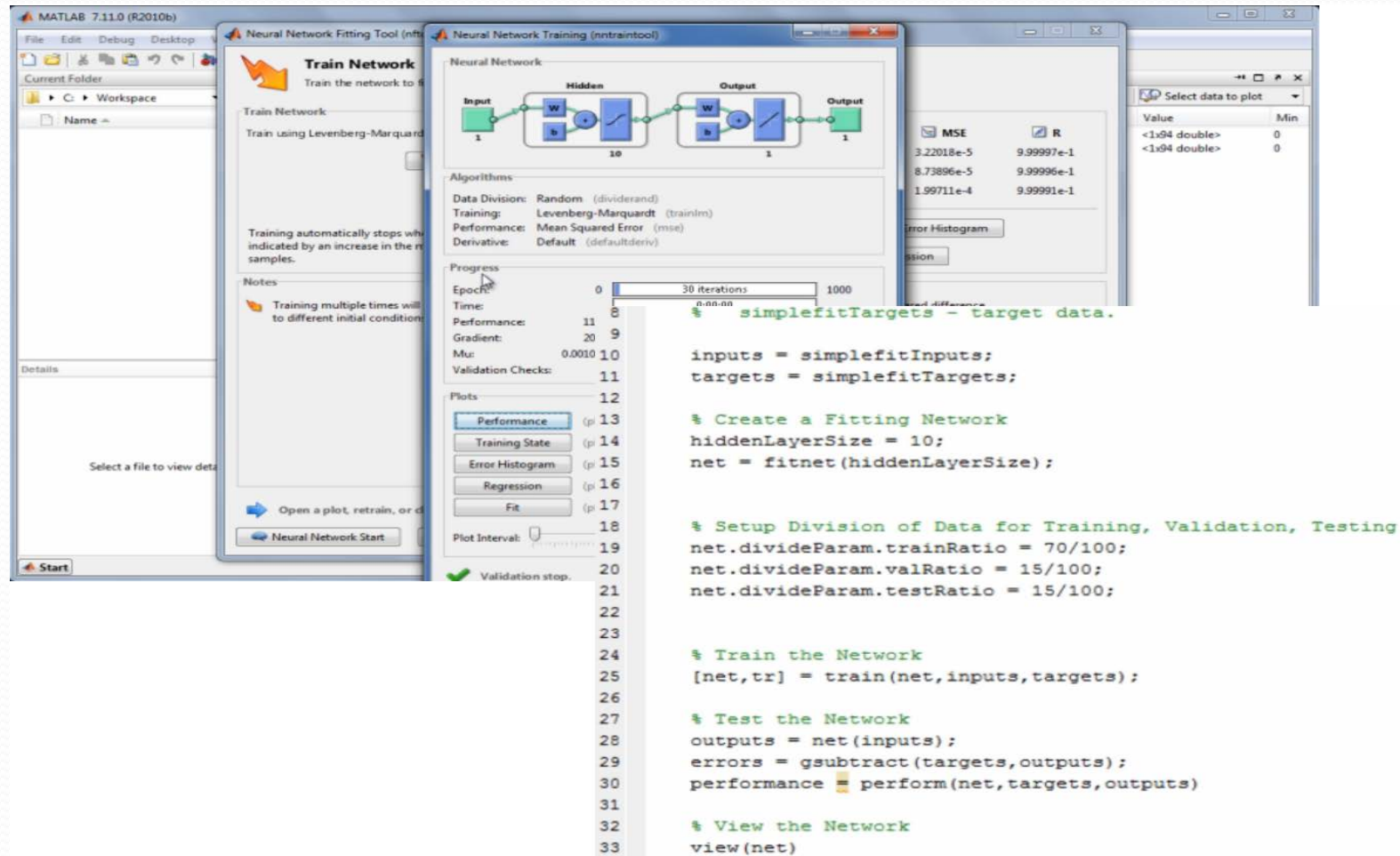
Script Structure



Neural Networks

Network Name	Network Type	Predictor Type	Comments
NN ₁	Fit	Numeric	First test for variable correlation.
NN ₂	Fit	Categorical	Better than NN ₁ .
NN ₃	Fit	Dummy Indicators	Faster than NN ₁ and NN ₂ while averaging the same fitting results as NN ₂ .
NN ₄ -Sub-1	Pattern Recognition	Dummy Indicators	First attempt at pattern recognition using dataset 1.
NN ₄ -Nosub-1	Pattern Recognition	Dummy Indicators	Removed subtype variables.
NN ₄ -Nosub-2	Pattern Recognition	Dummy Indicators	Removed subtype variables and ishostkid.
Consolidated-Nosub-1	Pattern Recognition	Dummy Indicators	All variables consolidated as seen in Table 1.
Consolidated-Nosub-2	Pattern Recognition	Dummy Indicators	Country and Date unconsolidated, other variables consolidated.
MoreVars-1	Pattern Recognition	Numeric	Consolidation attempts high misclassification error suggests more variable approach using dataset 2. Additionally, test to see whether numeric or categorical are better in Pattern Recognition.
MoreVars-2	Pattern Recognition	Dummy Indicators	Consolidation attempts high misclassification error suggests more variable approach using dataset 2. Additionally, test to see whether numeric or categorical.

Neural Network Toolbox



The image displays the MATLAB Neural Network Toolbox interface. The main window is titled "Train Network" and shows the "Train using Levenberg-Marquardt" option. The "Neural Network Training (nntraintool)" window is open, showing the network architecture (Input, Hidden, Output) and the training progress. The "Performance" plot shows the Mean Squared Error (MSE) and the R-squared value (R) over 1000 iterations. The "Error Histogram" plot shows the distribution of errors. The "Script Editor" window shows the following code:

```
1 % simplefitTargets - target data.
2
3 inputs = simplefitInputs;
4 targets = simplefitTargets;
5
6 % Create a Fitting Network
7 hiddenLayerSize = 10;
8 net = fitnet(hiddenLayerSize);
9
10 % Setup Division of Data for Training, Validation, Testing
11 net.divideParam.trainRatio = 70/100;
12 net.divideParam.valRatio = 15/100;
13 net.divideParam.testRatio = 15/100;
14
15 % Train the Network
16 [net,tr] = train(net,inputs,targets);
17
18 % Test the Network
19 outputs = net(inputs);
20 errors = gsubtract(targets,outputs);
21 performance = perform(net,targets,outputs)
22
23 % View the Network
24 view(net)
```

Misclassification Results

Network Name	Training Confusion	Test Confusion	Validation Confusion	Cross-validation Confusion
NN4-Sub-1	21.84%	35.76%	34.00%	34.94%
NN4-Nosub-1	18.25%	32.85%	33.64%	32.55%
NN4-Nosub-2	19.47%	33.65%	33.40%	33.22%
Consolidated-Nosub-1	51.46%	59.06%	58.76%	58.72%
Consolidated-Nosub-2	22.53%	34.39%	34.63%	33.70%
MoreVarsNum	30.08%	37.09%	36.29%	36.38%
MoreVarsText	17.56%	30.22%	30.54%	29.77%

Global Excel Database Import

- Previously the Matlab coding only accommodated for the GTD database.
- The code has been rewritten and now uses the START database format to import any database.
- In addition, it converts number cells in the database to character strings to allow for conversion of all values to nominal variables.
 - These improvements allow us to import the databases without the need to make changes to the database's excel file.
- Replaces empty cells with -99, which serves as unknown, for the sake of placing unknown as the first nominal code.
- Once the C++ implementation is completed fully, it will allow the user to select which database to search and what variables to eliminate.

Category sorting and searching

- Wrote Matlab code that returns the nominal code associated to the variable entry, as well as the number of instances it occurs within the database.
- Code can be modified to account for additional entries but works well under the generic implementation.
 - Under the C++ compiled implementation, this update can be used to generate lists of variables and search the database for particular values.
- Figure on the right shows the instances of group name occurring in the December 2014 GTD, sorted by number of occurrences (-99 being the dummy unknown for the purpose of setting it as the 0 code).

MATLAB Variable: getCatArray.gname			Page 1
Apr 28, 2015			1:10:36 PM
	1	2	3
1	0	-99	40749
2	2007	Shining Path (SL)	3924
3	731	Farabundo Marti National Liberation Front (FMLN)	3093
4	983	Irish Republican Army (IRA)	2218
5	2129	Taliban	2068
6	377	Basque Fatherland and Freedom (ETA)	1719
7	1835	Revolutionary Armed Forces of Colombia (FARC)	1588
8	1252	Liberation Tigers of Tamil Eelam (LTTE)	1364
9	571	Communist Party of India - Maoist (CPI-Maoist)	1217
10	1521	New People's Army (NPA)	1139
11	1204	Kurdistan Workers' Party (PKK)	1066
12	1482	National Liberation Army of Colombia (ELN)	994
13	1597	Palestinians	933
14	1529	Nicaraguan Democratic Force (FDN)	871
15	1303	Manuel Rodriguez Patriotic Front (FPMR)	717
16	2146	Tehrik-i-Taliban Pakistan (TTP)	641
17	2012	Sikh Extremists	574
18	608	Corsican National Liberation Front (FLNC)	545
19	441	Boko Haram	521
20	97	Al-Qa'ida in Iraq	495
21	2224	Tupac Amaru Revolutionary Movement (MRTA)	494
22	52	African National Congress (South Africa)	469
23	105	Al-Shabaab	418
24	1582	Other	418
25	1273	M-19 (Movement of April 19)	402

Modifying variable entries







- The nominal code conversion takes place in the entire database, which prevents code mismatch due to a possible value being in the test set and not in the training set or vice versa.
 - Would cause a shift in the codes and therefore lead to incorrect results.
 - This approach resulted in an increase in the value for country and weapon types, meaning that in at least these two variables the codes were mismatched in our previous approach.
- Converted dummy variable indicator into a smaller format that uses binary values as a basis.
 - The basis for this is to treat each binary string as an entry, only needing a significantly reduced number of entries (i.e. reducing 2405 gname entries to 12 values of 0/1)
 - This approach resulted in large misclassification errors and we're currently looking into the reason for this odd result.
- Table on the right shows the result of combinations between the new format (binary) and the previous format (dummy).

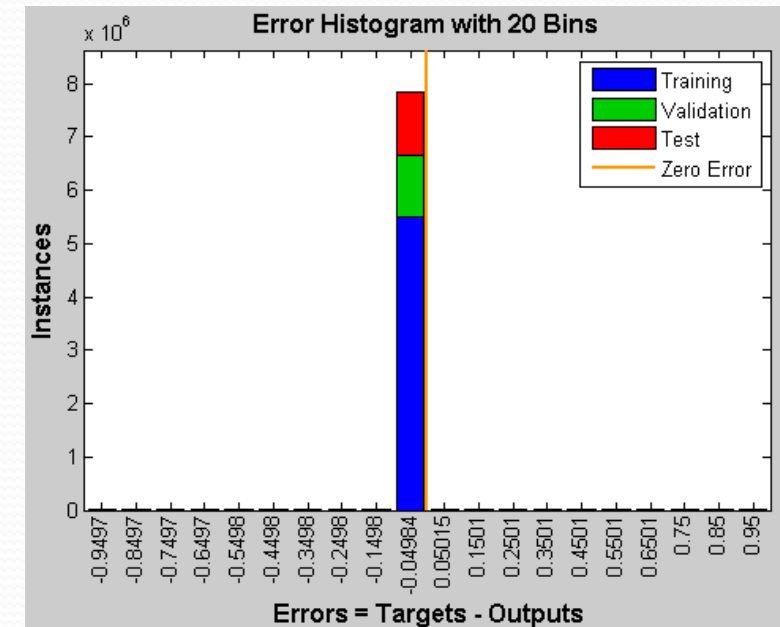
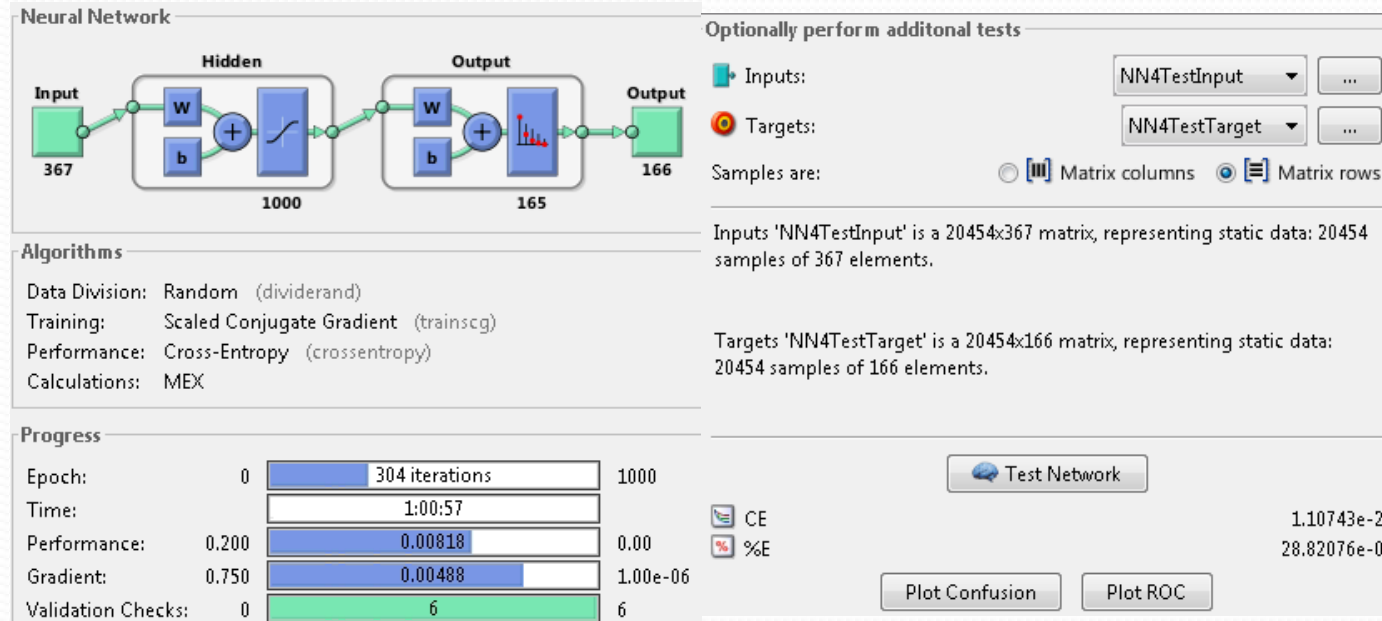
Input	Response	Result
Binary	Binary	Negative
Binary	Dummy	Positive
Dummy	Binary	Negative
Dummy	Dummy	Positive

Matlab Script Advances

- Set a string constant that determines which dataset will be imported at the beginning of the main script.
- Debugged import function to ensure that all sheets are imported properly.
 - Added code to deal with issues that arise in Matlab from variable names using /.
 - Consolidated the “unknown” variables (i.e. -66, -99, -9, unknown) into a single -99 value.
 - Fixed an issue that can arise from certain numeric cells appearing as NAN long after the database variables ended.
- Developed the functions:
 - Determine which variables fall below a certain threshold in order to group them together into a new “Other” category, or if one already exists, add them to it. This creates an array with said values.
 - Can also determine the variables that were not cut-off, if further information is necessary.
 - The function that takes the above cut-off array and creates a dataset based on the results.
 - Function runs correlation functions in every pair combination of variables in the data set.

Results as a consequence of the cut-off

Results			
	 Samples	 CE	 %E
 Training:	33410	4.21881e-0	21.34390e-0
 Validation:	7159	12.00268e-0	27.69939e-0
 Testing:	7159	11.99623e-0	27.40606e-0



GTD Results Summary

- Subtypes had little impact in the misclassification error percentage and removing them increased accuracy by 3%, but they helped in reducing the number of iterations and computing time.
- While removing some variables from the same dataset does not seem to impact misclassification significantly, the more variable approach seems to provide better results.
- Many of the initial features that were worked on during the first period have been fully automated and generalized in order to accommodate a broad array of data.
- The current consolidation implementation has improved misclassification percentages by 8%.

Recap of Neural Network Specifics

- Set a string constant that determines which dataset will be imported at the beginning of the main script.
- Debugged import function to ensure that all sheets are imported properly.
 - Added code to deal with issues that arise in Matlab from variable names using /.
 - Consolidated the “unknown” variables (i.e. -66, -99, -9, unknown) into a single -99 value.
 - Fixed an issue that can arise from certain numeric cells appearing as NAN long after the database variables ended.
- Developed the functions:
 - Determine which variables fall below a certain threshold in order to group them together into a new “Other” category or if one already exists, add them to it. This creates an array with said values.
 - Can also determine the variables that were not cut-off, if further information is necessary.
 - That takes the above cut-off array and creates a dataset based on the output results.

Matlab Script Advances

- *Pinpointed the issues that arise as a result of the binary implementation.*
 - *The cutoff function improves overall percentages for all variable types when the dataset is large.*
 - *For small data sets where the training time is low, dummy variables are good enough.*
- *Consolidated single-instance variable values into several common variables.*
 - *The cutoff function developed carries out this operation.*
- *Improved on the neural network approach currently being used through Matlab and developed code that helps manipulate the data in such a way that it might be easier for the neural network to train.*
 - *Multiple code fixes were implemented to address issues that arose from data import. We currently have no issues importing all START databases in their current incarnations.*
- *Expanded on the efforts for this period relating the extraction of meaningful data from the START datasets (which began with determining the amount of occurrences in each data).*
 - *The development of the correlator function and the byproduct arrays from the cutoff function provide additional information on the nature of the data.*

Matlab Script Advances Continued

- *Developed other cut-offs, in addition to the #entries cut-off.*
 - *New “Cut-off” - Modified the previously developed confusion analysis scripts to cause them to loop to attempt to find a result below a certain cut-off. For some pairs of data sets, it has been possible to get percentages as low as 1.5%.*
 - *This approach coupled with the correlator function and looping through all pairs of data is being used as part of the missing data population function.*
- *Development of clustering and regression functions utilizing the neural network toolbox to run - similar to the current process of pattern recognition. We have created an Amazon EC2 instance to allow for faster analysis of data, which can be used to test all of these processes for random sets of data.*

Matlab Script Enhancements

- *Developed functions similar to the pattern recognition script for use in clustering and regression and determined ways to analyze the data from the neural network.*
- *Identified errors that are caused by different runtime errors and arranged to fix them by encapsulating them in try-catch statements.*
 - *Ran the main script and determined what possible errors using the current dataset could occur. This approach is being expanded to include additional datasets from the various START databases for comprehensive testing.*

Improvements in Progress

- Implementing the Matlab code into C++/C# for a user interface.
- Address the missing data issues based on the results of loop variable correlation discussed in the previous slide.
- Continue running and testing this implementation using neural networks to find the optimal indices and hidden networks.
- *Slurm* script refinements for Deepthought2 job submission.
- Develop other cut-offs in addition to the #entries cut-off.
- Create a “for” loop that goes through each pair of variables and determines the level of correlation, potentially giving weights based on the results and feeding those weights to the neural network (as opposed to the current automated method).
- Use a time-oriented approach in order to catalog certain values (i.e. like groups that have no terrorist activity in decades) under “Inactive”.

POICN Revisited

- **Conversion to neural networks**

- Attempting to answer Research Question raised by Breiger, Murray & Pinson (2013): “How can we define a network among the cases of (CBRN events) such that that network yields the same predicted logits ($\log(p/1-p)$) as are produced by the standard logistic regression model?
- Neural Network Weights vs Regression Results

- **Perceptron Creation**

- Use the “nnet” neural network fitting command with a formula of the form:
 $y \sim x_1 + x_2 + \dots$ instead of with input and output matrix and vector, respectively.
- Monitor the results of using both TRUE and FALSE for the “entropy” flag which decides to perform the fitting via the maximum likelihood and least squares methods, respectively.

Logistic Perceptron Creation

Contains $I + 1$ input units, x_0, \dots, x_I of the input vector
 x_0 is a constant input of 1's

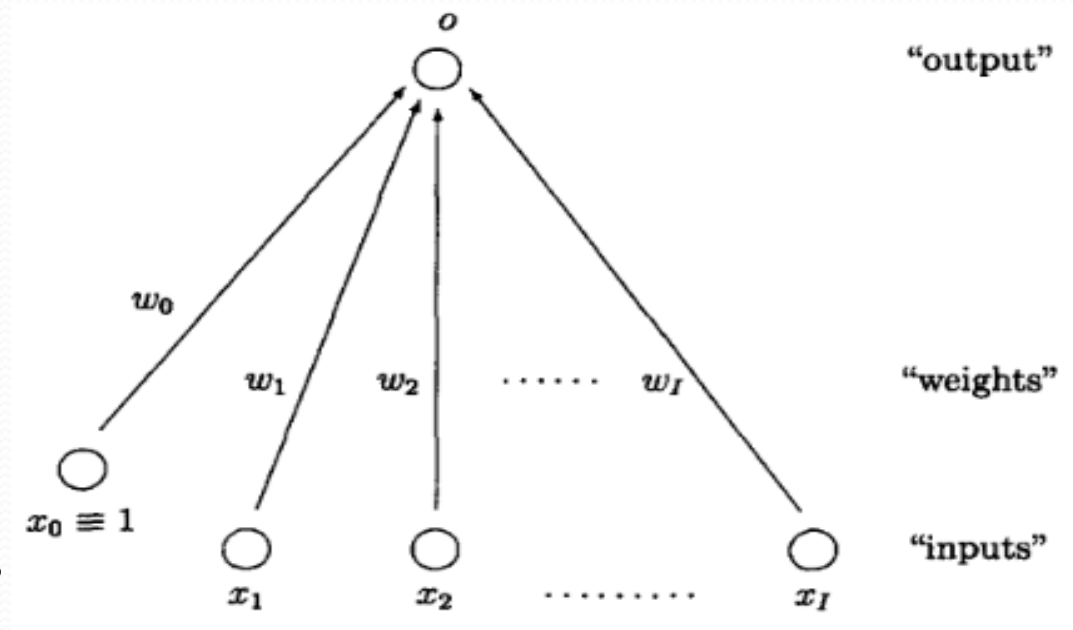
The output function o is again the “logit” function of the input x , and weights w . $o = f(x, w) = \Lambda(w_0 + \sum_{i=1}^I w_i x_i)$

Where $\Lambda(u) = \frac{1}{1 + e^{-u}}$

Learning accomplished by minimizing the “learning or energy” function, $E(w) = \sum_{n=1}^N \left(y^{(n)} - f(x^{(n)}, w) \right)^2 = 0$

Learning also accomplished using the alternative energy function known as the “Kullback-Leibler Distance.”

$$E^* = \sum_{n=1}^N \left[y^{(n)} \log \frac{y^{(n)}}{f((x^{(n)}, w))} + (1 - y^{(n)}) \log \frac{1 - y^{(n)}}{1 - f((x^{(n)}, w))} \right]$$

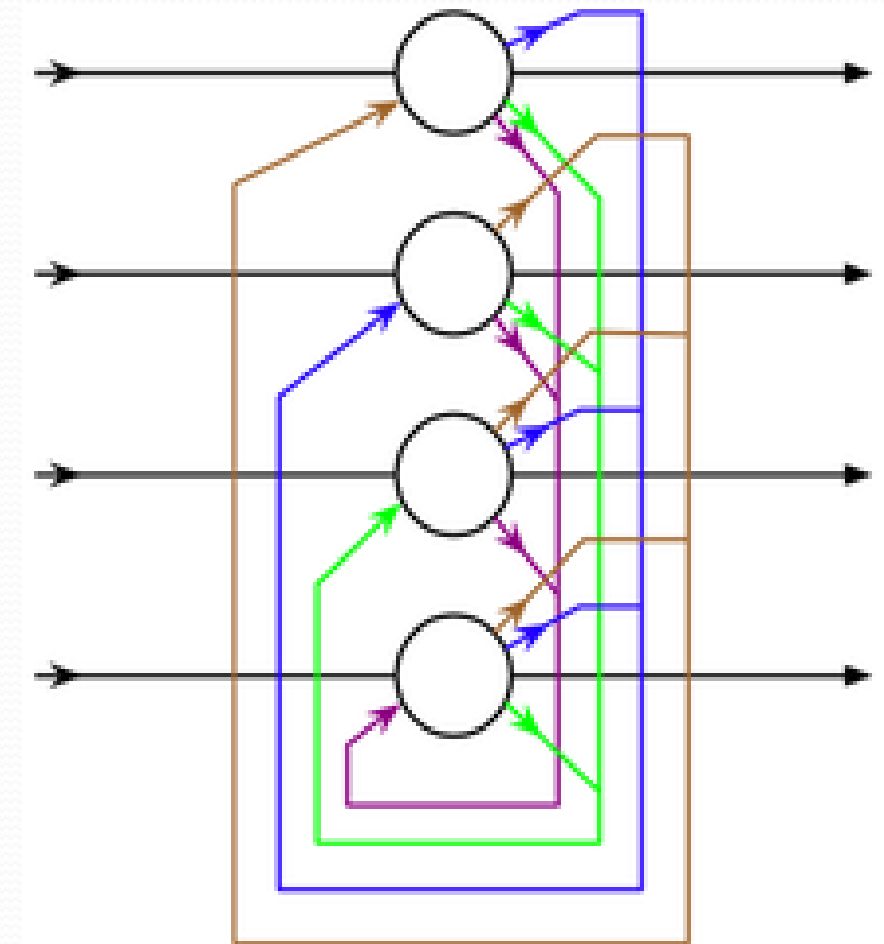


M. Schumacher et. al., Comp. Stat. & Data Analysis, (21), No. 6, 1996

Hopfield Network

Hopfield Network	2-D Ising Spin System
E, “Energy Functional”	H_N , “Hamiltonian”
W_{ij} , Matrix of weights of neuron connections	W_{ij} , Nearest neighbor interaction energy of spins
s_j , neuron in state j	s_j , spin in state j

$$H_N = E = \left(-\frac{1}{2}\right) \sum_{i \neq j} W_{ij} \sigma_i \sigma_j$$



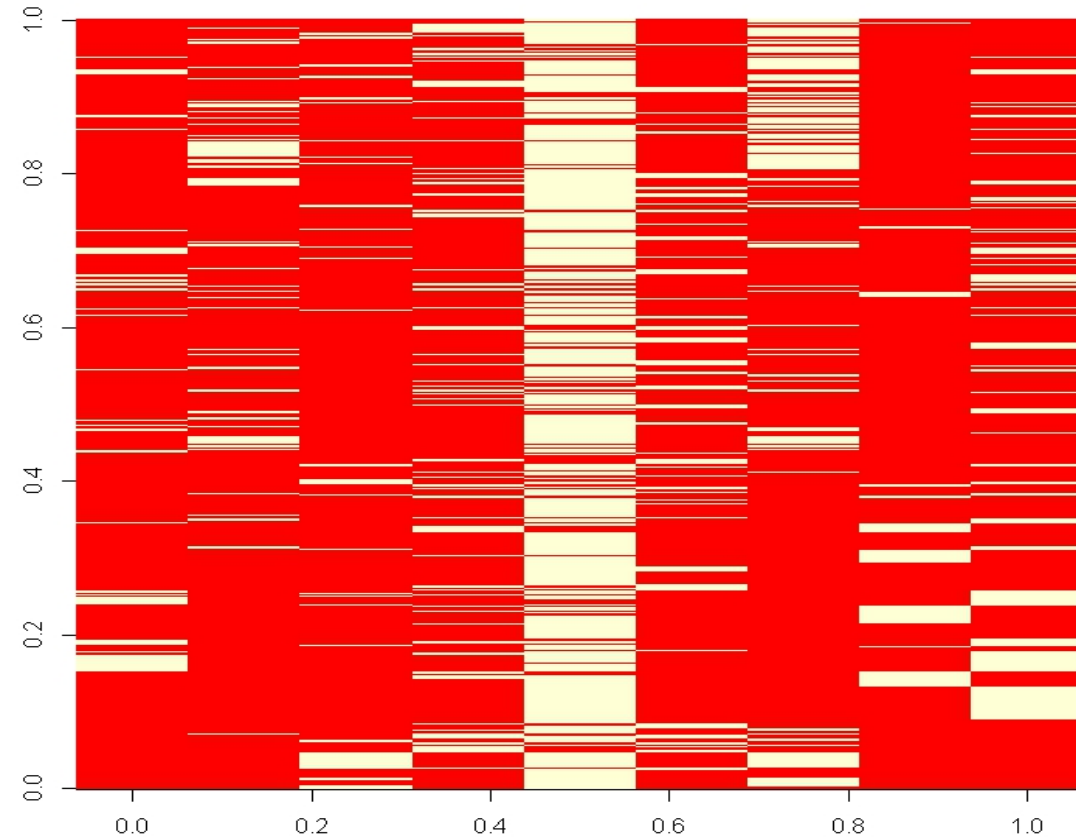
Initial Hopfield Network Exploration

Russia NIS	Middle East	South Asia	is.bi o	is.ch em	perp.l one	perp.r elex	perp. cult	perp.eth nonat
o	o	o	o	1	o	1	o	o
o	o	o	o	o	o	1	o	o
o	o	o	o	1	o	o	1	o
o	o	o	1	1	o	1	o	o
o	o	o	1	1	o	o	o	o
o	1	o	1	1	o	1	o	o
o	o	o	1	1	o	1	o	o
o	o	o	1	1	o	1	o	o
o	o	1	o	1	o	1	o	o
o	o	1	o	1	o	1	o	o
o	o	o	1	1	o	1	o	o
o	o	1	o	1	o	o	o	o

Heat Map of an “Adaptive Resonance Theory”,
Hopfield Network.
(Created with R’s RSNNS Add-on Package)

Pairwise snapshot of Type B (red) and Type A
(white) events among the cases.

Based on 2015, POICN data



Exploring the Statistical Thermodynamics of Neural Networks based on POICN Data

- Try to match the resulting Logistic Perceptron Weights & Logistic Regression Coefficients
- Compare the Hopfield Neural Network results with the logistic regression coefficients reported by Breiger & Pinson , 2013, ISA Report, Table 3, Standardized & Unstandardized

Explore the statistical thermodynamics aspects of neural networks based on POICN data, and ultimately examine the consequences of Breiger's novel reverse logistic regression concept in terms of the statistical thermodynamics of Hopfield neural networks.

Summary & Conclusions

- Improved pattern recognition, regression and clustering, for START databases (GTD and POICN) using neural networks
- Efficient dealing of missing or incomplete data with innovative Matlab script and cutoff functions
- Efficient data compression, missing data regression and clustering for START databases
- Logistic regression and neural networks comparison study
- Expansion to other databases (e.g. Profiles of Individual Radicalization in the U.S. - PIRUS) with modified machine learning code
- Developed Matlab scripts that can now import all START databases

Future Work

- Optimization of the Matlab code to speed up the execution phase of the script
- Use the Matlab Coder/Compiler toolbox to convert the script into a programming language (e.g. C/C#/C++, Visual Basic and/or Java) & then use the resulting code to develop a user-interface
- Study logistic regression using other START databases, besides POICN
- Use neural networks as a data compression tool for images and videos and subsequent pattern recognition
- Expansion of current START DHS efforts dealing with static database information to dynamic information, such as streaming data
- Implementation and fusion of social science techniques with physical & mathematical concepts to develop hybrid machine learning algorithms
- Generalization & combination of social events data from multiple sources (DHS, IARPA, DARPA, DOD, etc.) into one global system that catalogs and makes determinations based on user needs

Acknowledgements

- I would like to thank my research collaborators, Raul Garcia-Sanchez and Daniel Casimir, who worked on this DHS-START project during the period 2014-16.
- I would like to thank the Department of Homeland Security (DHS) for funding provided through the Minority Serving Institutions (MSI) program that enabled my team to work with mentors (Gary Ackerman, Markus Binder, John Sawyer, William Braniff and Eamon Diab) at *START* on this exciting research project as applied to the Global Terrorism Database (GTD), the Profiles of Incidents involving CBRN by Non-state actors (POICN), and the Profiles of Individual Radicalization in the United States (PIRUS) database efforts at START.