MtNet: A Multi-Task Neural Network for Dynamic Malware Classification

Wenyi Huang – Pennsylvania State University
Jay Stokes – Microsoft Research

Microsoft

Motivation

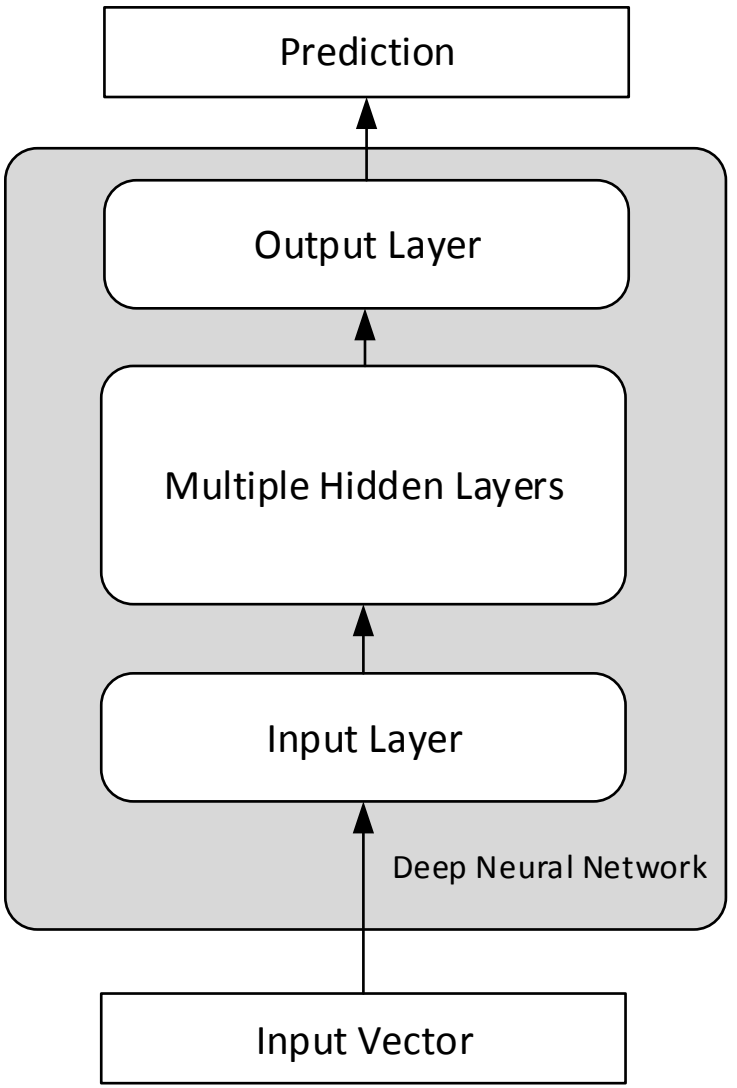
Deep learning has led to large improvements in speech and object recognition

Early work on deep learning for malware classification has not shown the same improvements

- DNN for dynamic analysis – (Dahl, 2013)
- ESN,RNN for dynamic analysis – (Pascanu, 2015)
- DNN for static analysis – (Saxe, 2015)

Can recent improvements help?

Deep Learning



Layer

$y = s(Wx + b)$, where s is activation function (sigmoid, tanh, ReLU)

Deep

Two or more hidden layers

Output layer

Softmax

Normalizes output probability

Objective function

Cross entropy

Investigate

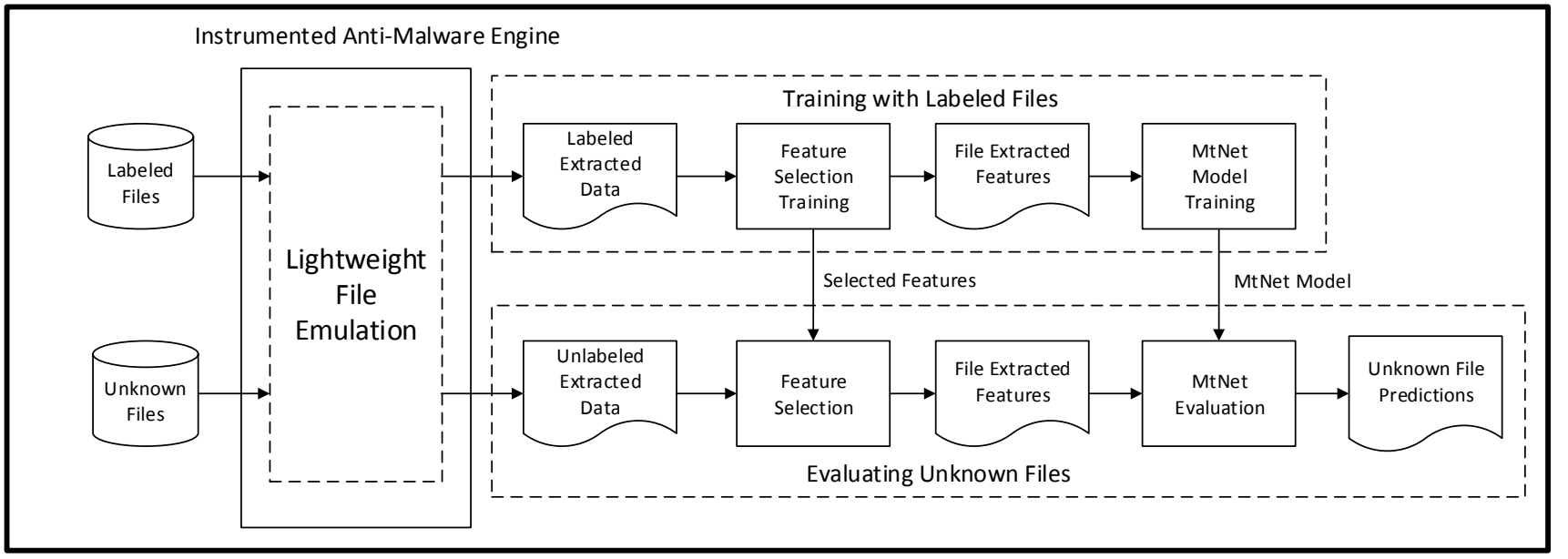
Can multi-task learning improve malware classification?

How do the various deep learning components affect the classification accuracy?

Can we improve detection rates at extremely low false positive rates?

MtNet System

MtNet High-Level Overview



Feature Selection - Mutual Information

Malware Classification

Given features extracted from an executable file

Determine if it is malicious
Classify it into a known family

Dataset

4,500,000 training
2,000,000 testing
2-classes (malware + benign)
100-classes (98 family + malware + benign)

Toolkit

Microsoft Computational Network Toolkit (CNTK)

MtNet Features

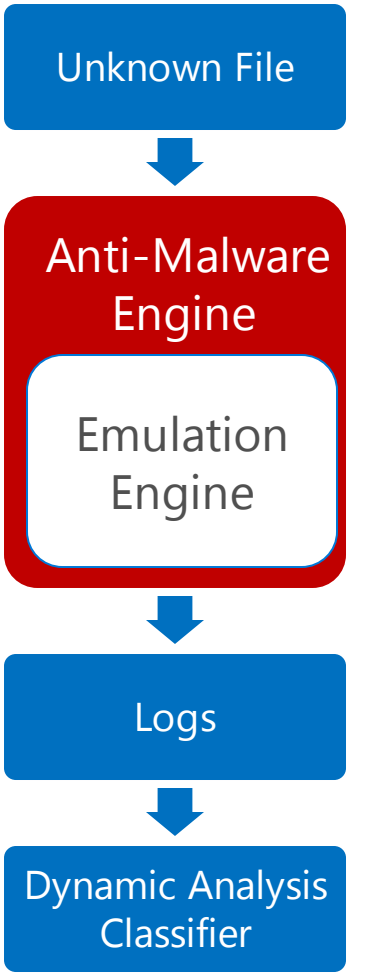
Unpacked File Strings

"stopkeylogger"

"removing bot..."

API Call + Parameter Value

KERNEL32.DLL!HeapCreate dwInitialSize=4096
NTDLL.DLL!RtlAllocateHeap Size=320

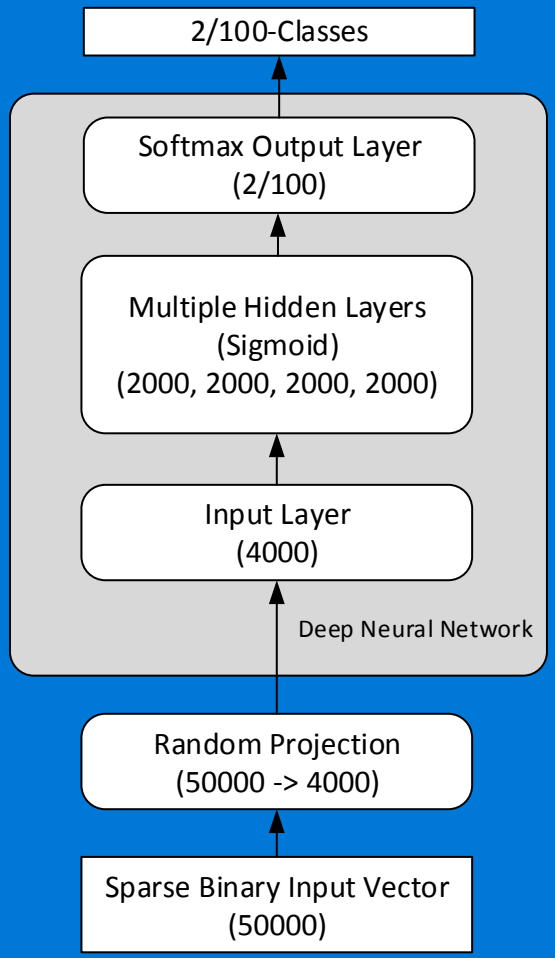


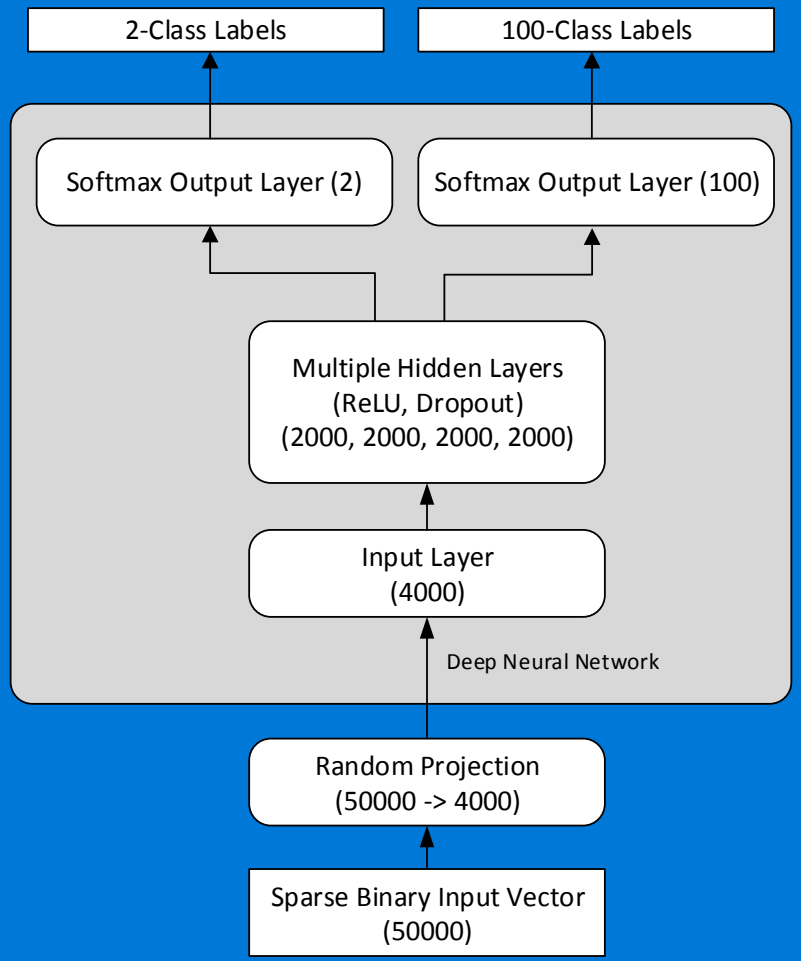
Trigrams of API Calls

KERNEL32.DLL!GetVersion, KERNEL32.DLL!HeapCreate, NTDLL.DLL!RtlAllocateHeap
KERNEL32.DLL!HeapCreate, NTDLL.DLL!RtlAllocateHeap, KERNEL32.DLL!InitializeCriticalSection

Multi-Task Learning and Other Algorithmic Improvements

Proposed Architecture





Reimplemented Baseline (Dahl, 2013)

MtNet

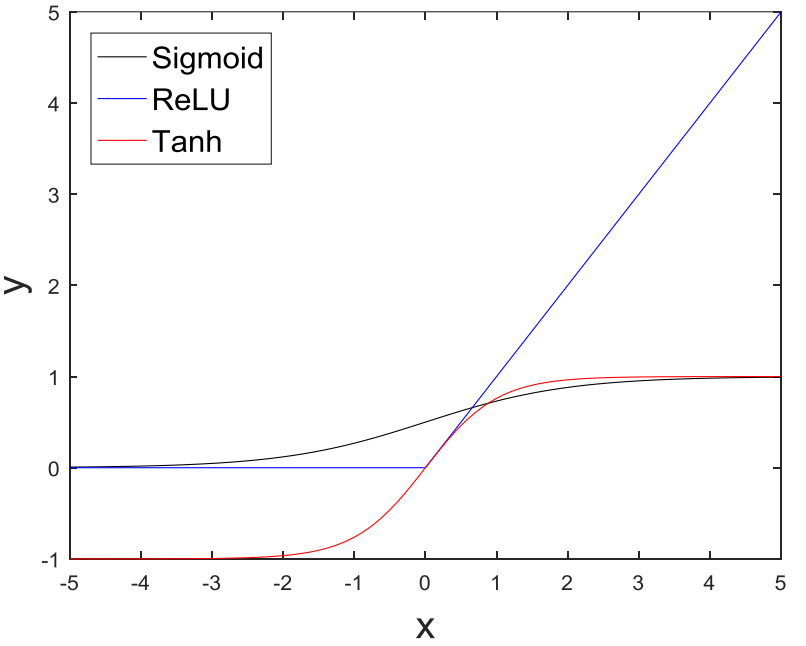
Activation Function

Common activation functions

Sigmoid: $f(x) = \frac{1}{1+e^{-x}}$

Tanh: $f(x) = \frac{1-e^{-2x}}{1+e^{-2x}}$

Rectified linear: $f(x)$=max(0,x)



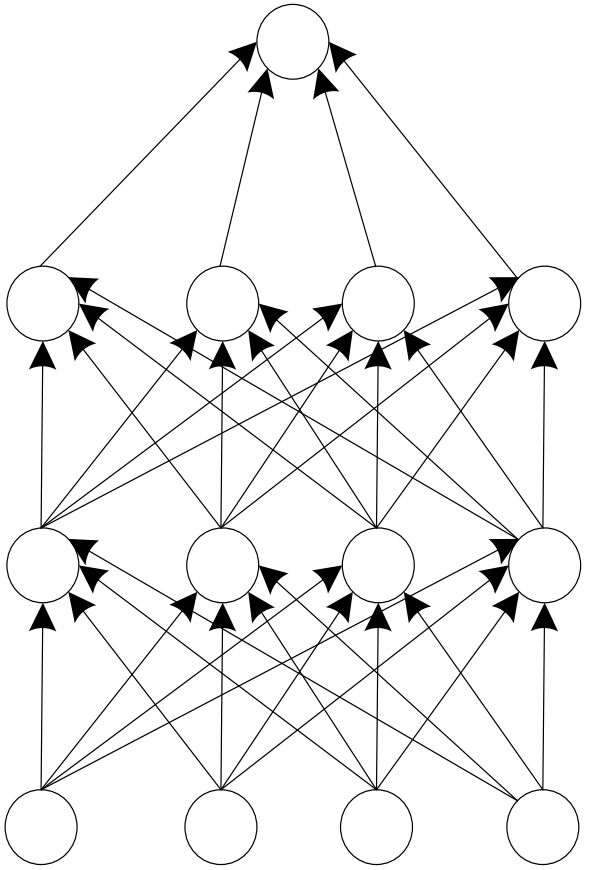
Rectified-Linear Units (ReLU) instead of sigmoid function

Can sometimes help you go deep

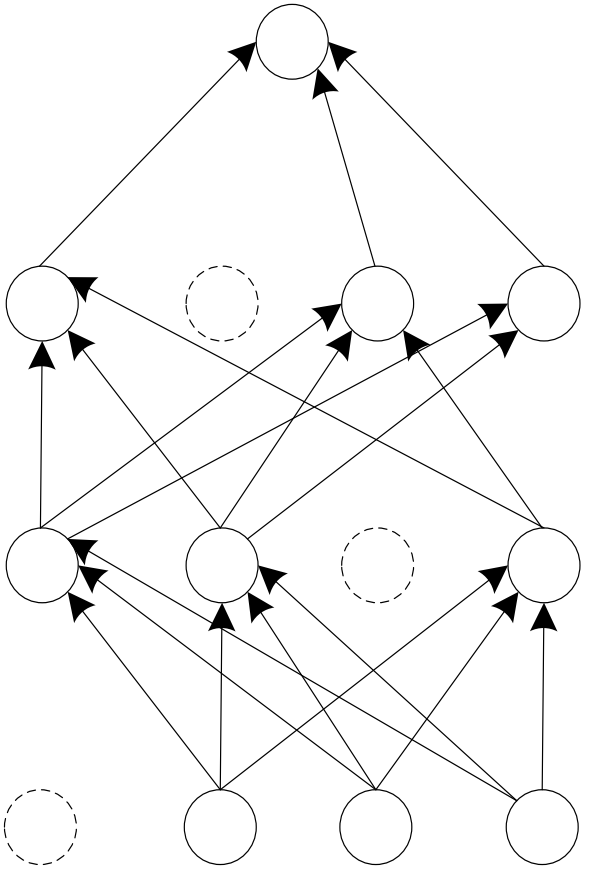
(Pascanu15,Saxe15)

Dropout

(Pascanu15,Saxe15)



Standard Training



Training with Dropout

Results

Binary Single-Task Results

	Dahl13 Baseline Model (Original Data)	Reimplemented Dahl13 Baseline Model (Our Data)		Our Single-Task Model (Our Data)	
Training Samples	2.4M	4.5M		4.5M	
Number Features	179K	50K		50K	
Setup	Sigmoid	Sigmoid		ReLU, Dropout	
Layers	Test Error (%)	Test Error (%)	Epoch	Test Error (%)	Epoch
1	0.49	0.5906	190	0.3711	64
2	0.50	0.4882	186	0.3702	82
3	0.51	0.4845	200	0.3686	77
4		0.4934	200	**0.3683**	81

Increasing improvement through 4 layers

ReLU activation function cuts training epochs by over half

Multi-Task Results

	2-Class Test Error (%)		100-Class Test Error (%)	
Layers	Multi-Task	Single-Task	Multi-Task	Single-Task
1	0.3657	0.3711	**2.935**	**2.935**
2	**0.3577**	0.3702	3.025	2.983
3	0.3618	0.3686	3.026	2.982
4	0.3655	0.3683	3.070	2.970

2-Classes: Improves results for all layers

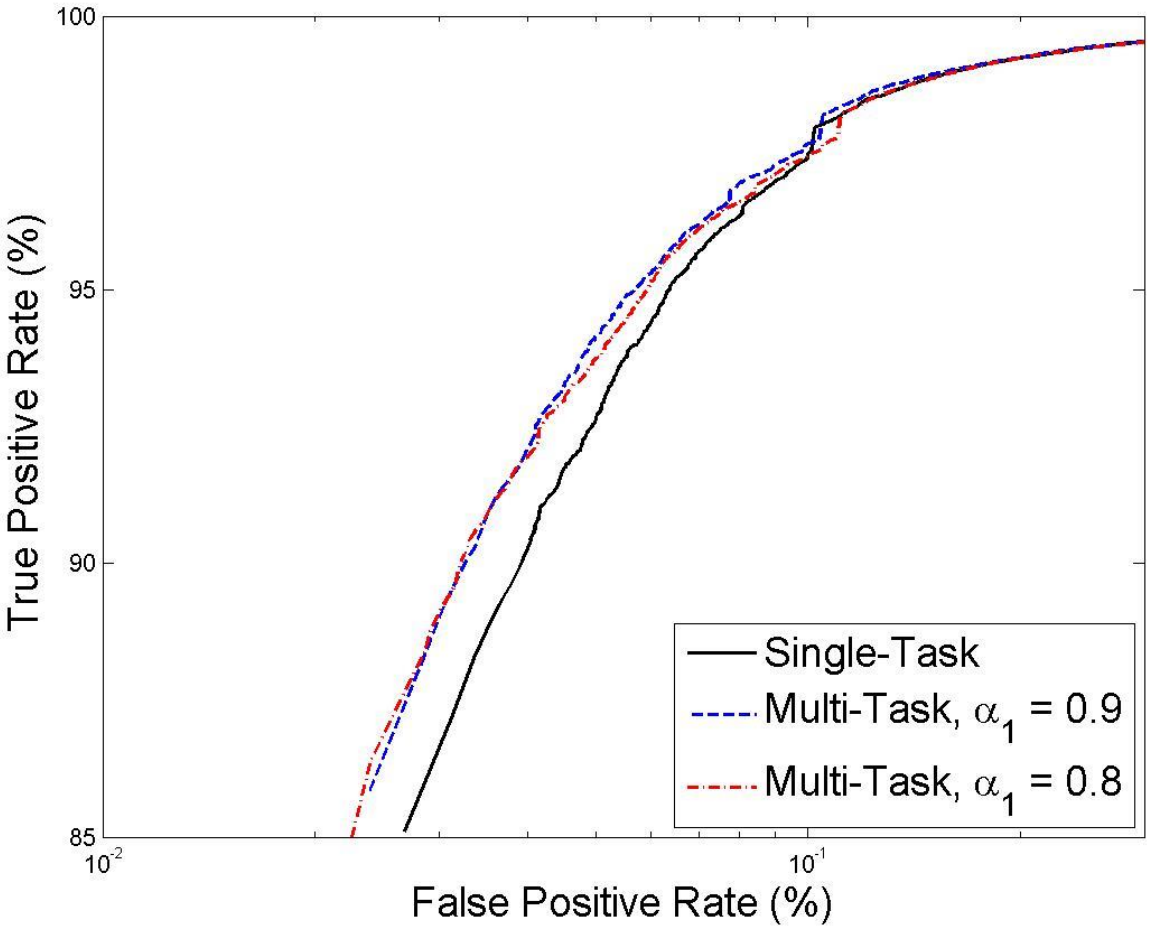
100-Classes: Same or slightly worse

Optimized for improved 2-Class performance

Multi-Task Vs Single-Task Binary Classification

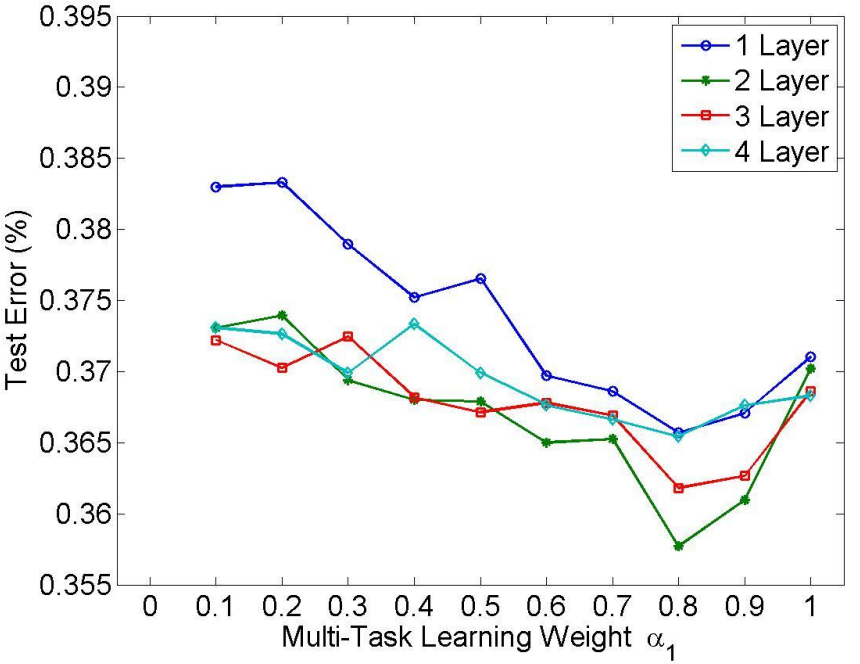
$$L_{MT}(\theta(\boldsymbol{x})) = \alpha_1 L_2(\theta(\boldsymbol{x})) + \alpha_2 L_{10.0}(\theta(\boldsymbol{x}))$$

$$\alpha_1 + \alpha_2 = 1$$

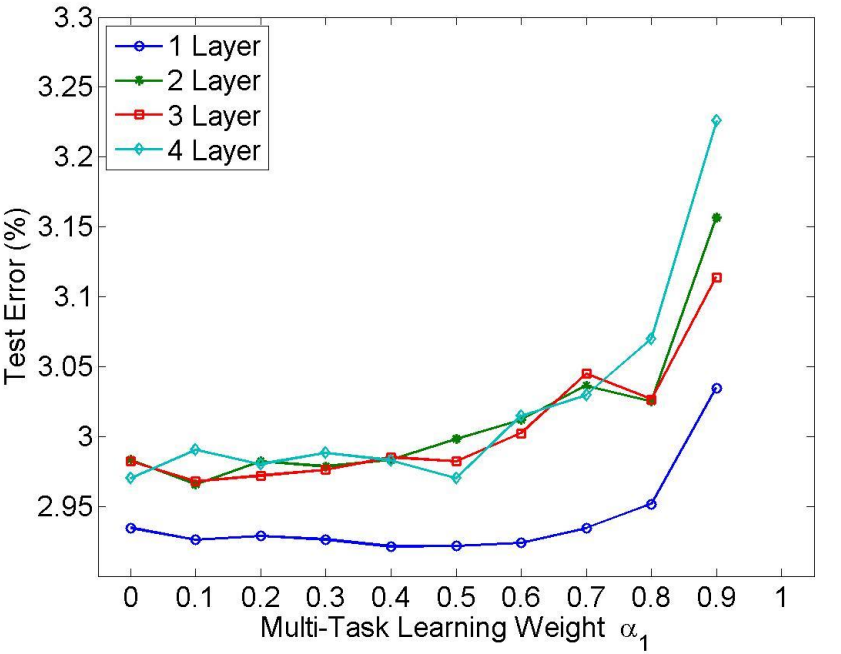


Multi-Task Weight Comparison

$$L_{MT}(\theta(\boldsymbol{x})) = \alpha_1 L_2(\theta(\boldsymbol{x})) + \alpha_2 L_{100}(\theta(\boldsymbol{x}))$$

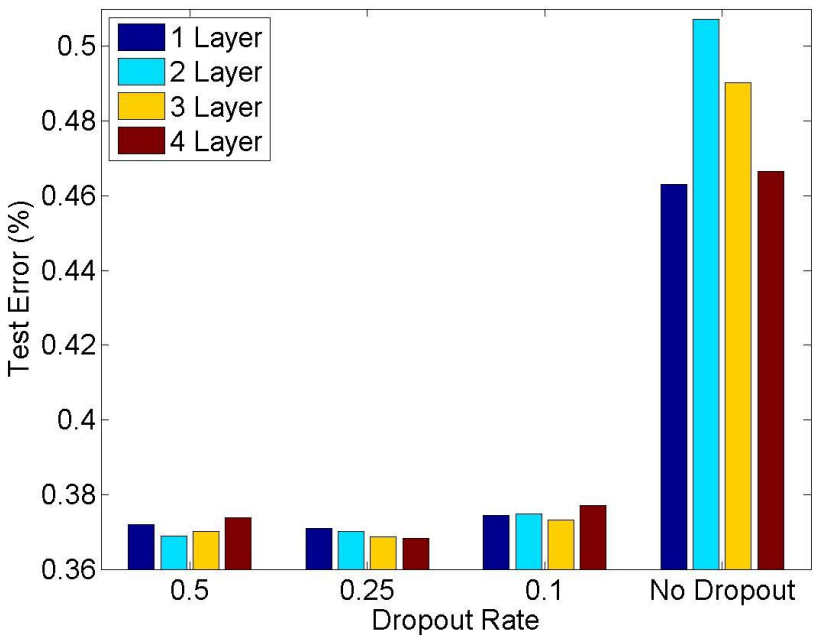


Binary Classification

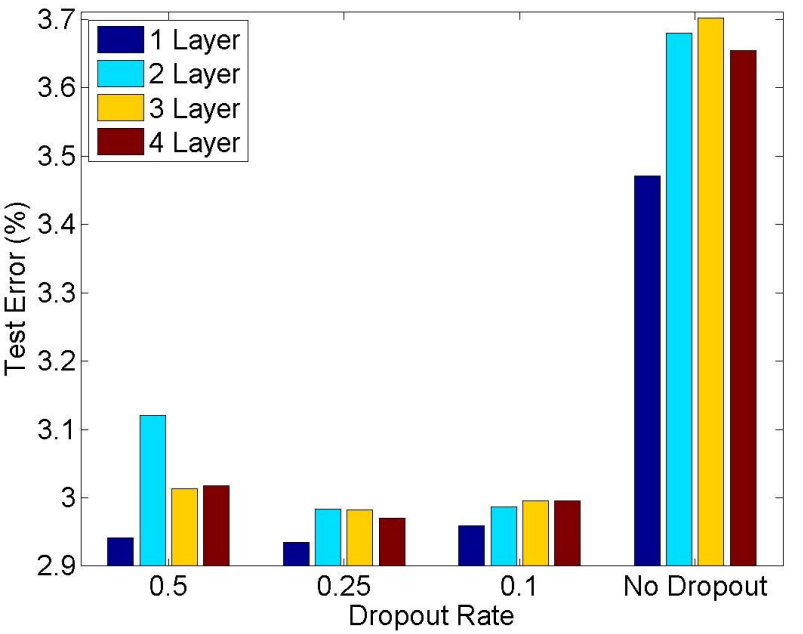


Family Classification

Dropout Rate



Binary Classification



Family Classification

Conclusions

MtNet performance improvement over baseline

2-Class: 26.2% , 100-Class: 19.2%

Dropout is the main factor in error rate improvement

Multi-task learning reduces error rate

Helps at low FP rates

Modest improvement from deep learning

Rectified linear activation functions reduce training epochs by over half for binary classification

Best family classification error rate 2.94%