UMBC

AN HONORS UNIVERSITY IN MARYLAND

BCAP: An Artificial Neural Network Pruning Technique to Reduce Overfitting

By: **Kiante Brantley**
Graduate Student
M. S. Computer Science

Guided By: **Dr. Tim Oates**

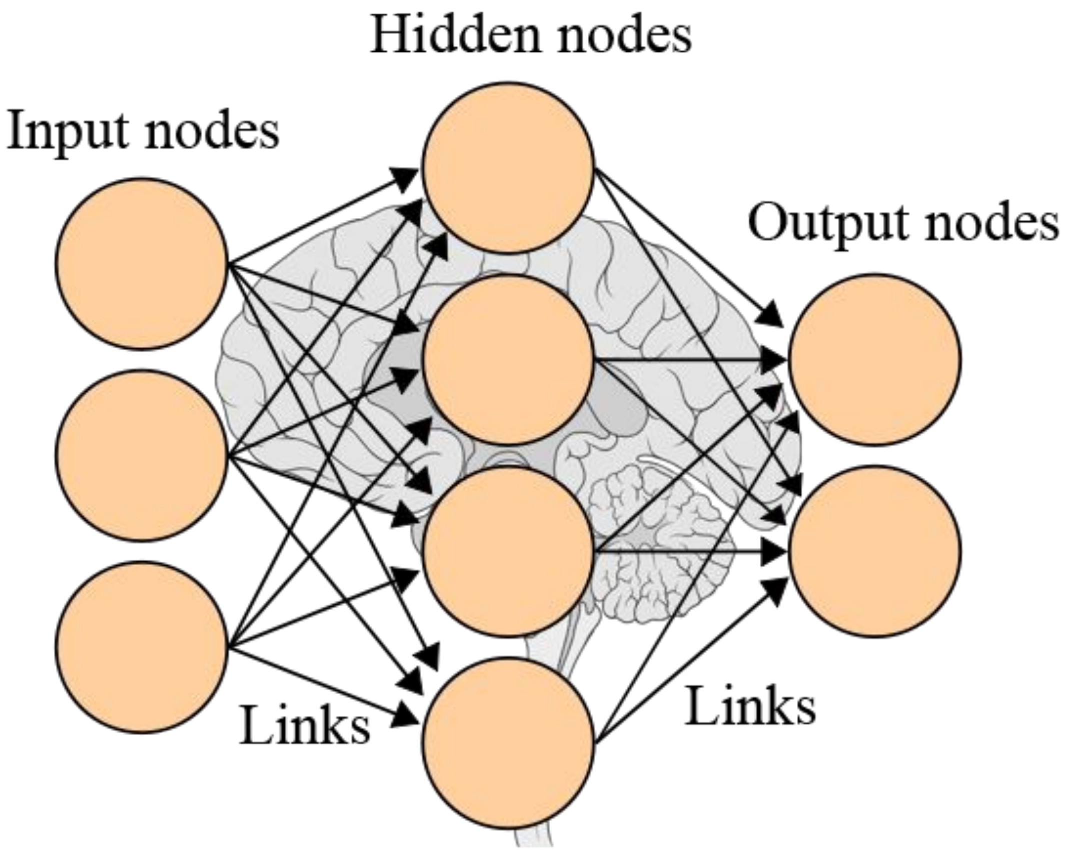
cognition | robotics | learning
CORAL

Index

- Motivating Problem
- Research Objective
- Methodology
- Experiments
- Analysis
- Conclusions
- Future Work

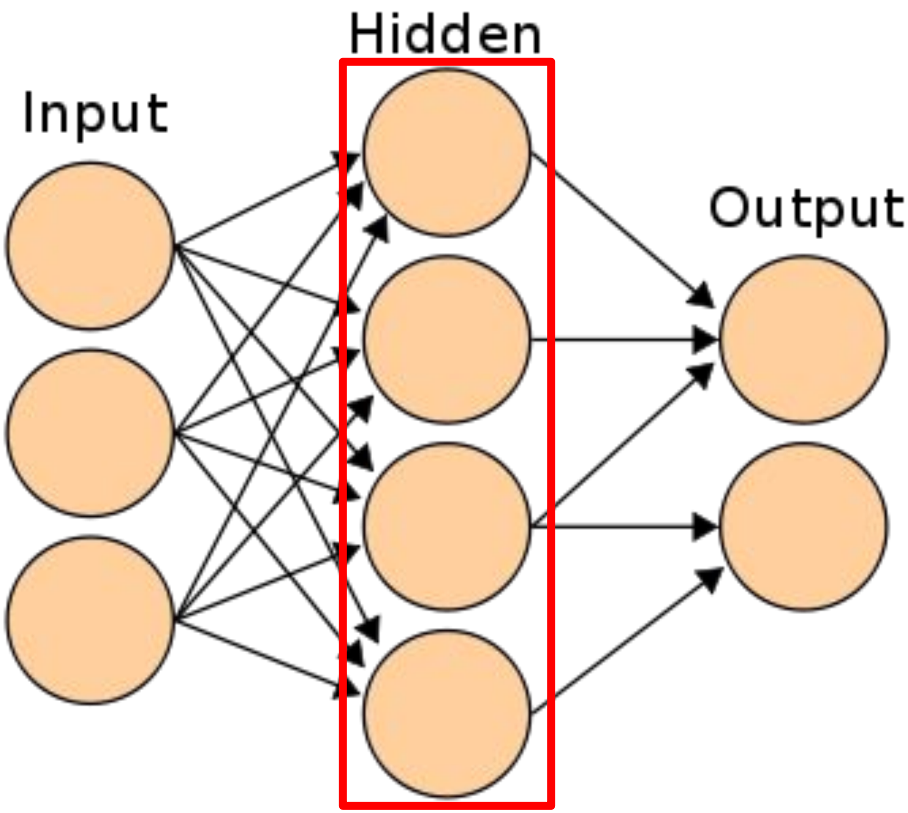
What is artificial neural network (ANN)?

Machine learning model inspired by biological neural networks



But what is artificial neural network (ANN)?

Functional mapping between input values and output values



What is the problem?

Larger networks:

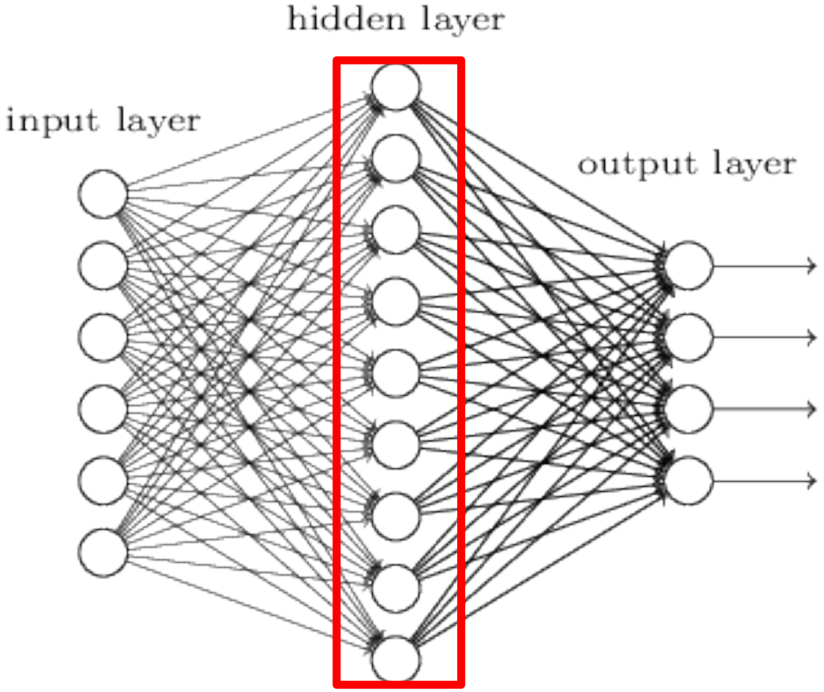
Pro's:

- Learn quickly
- Less sensitive to initial condition
- Less sensitive to local minima

Con's:

- Overfitting

[Reed, 1993]



What is the problem?

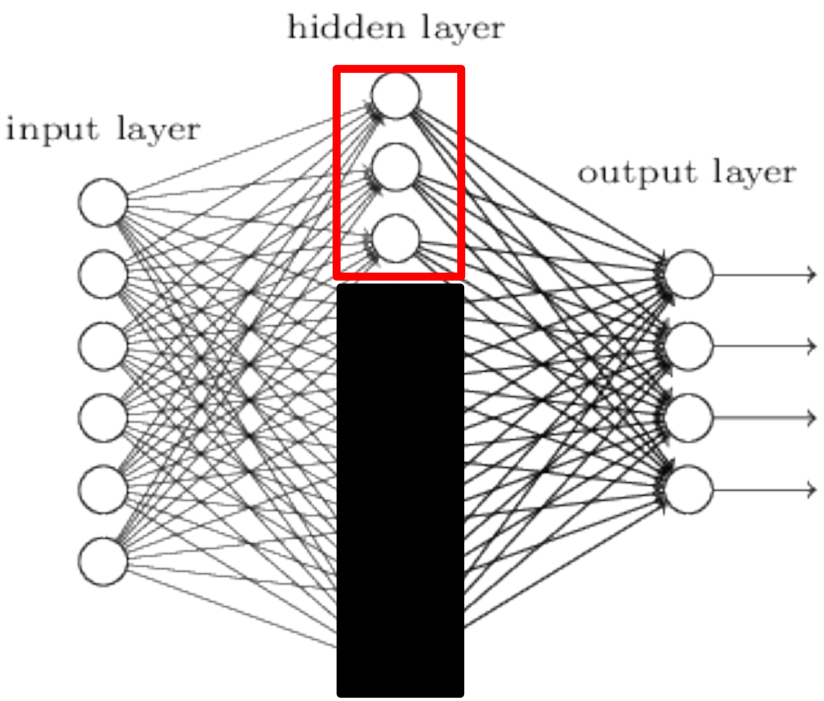
Smaller networks:

Pro’s:

- Generalize better
- Faster build
- Faster to compute
- Easier understand

Con’s:

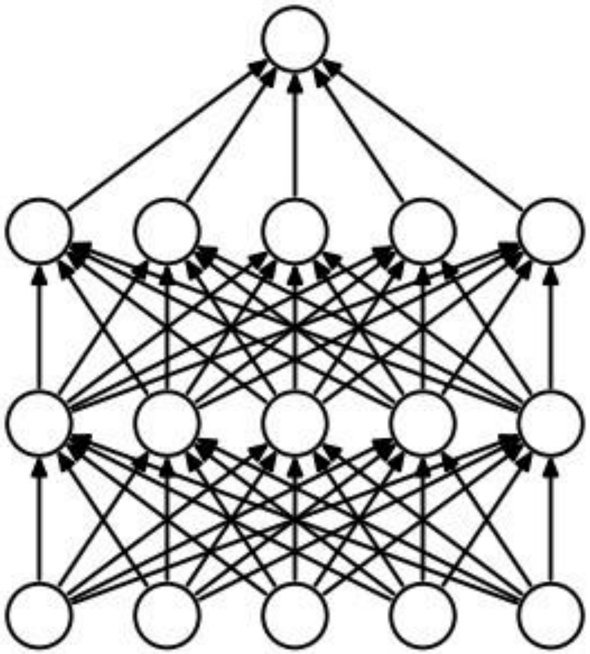
- Underfitting



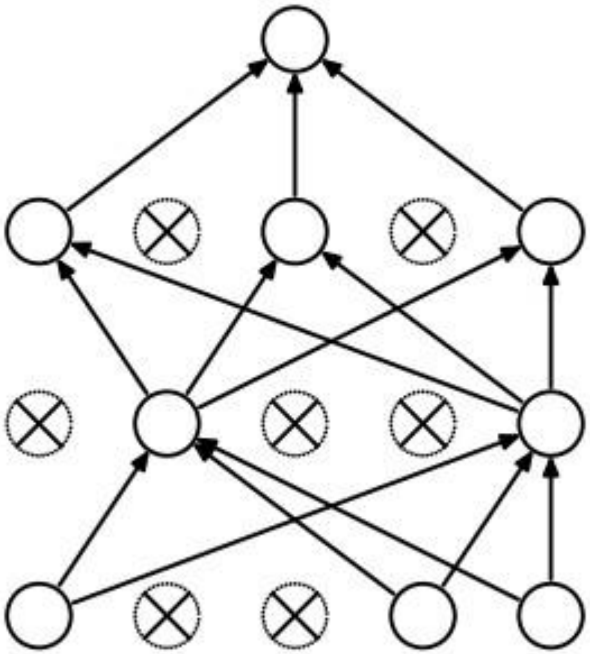
[Reed, 1993]

Current Approaches to solving overfitting:

Dropout [Hinton et al., 2014]



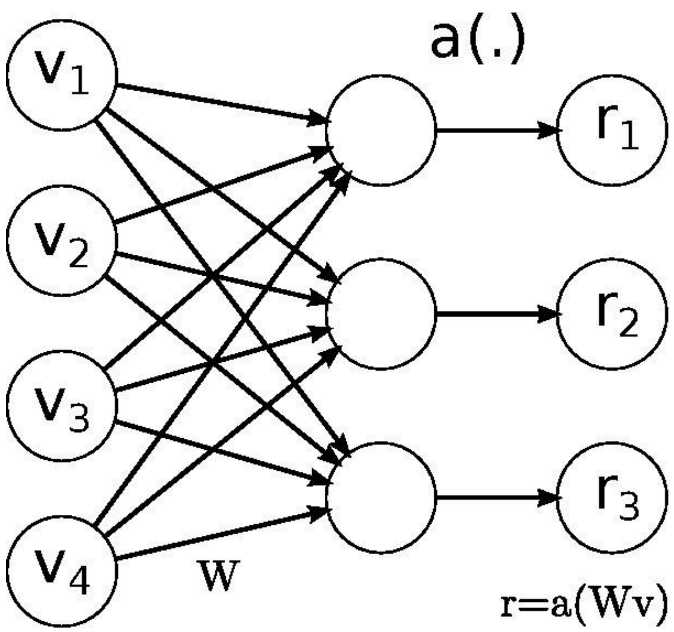
(a) Standard Neural Net



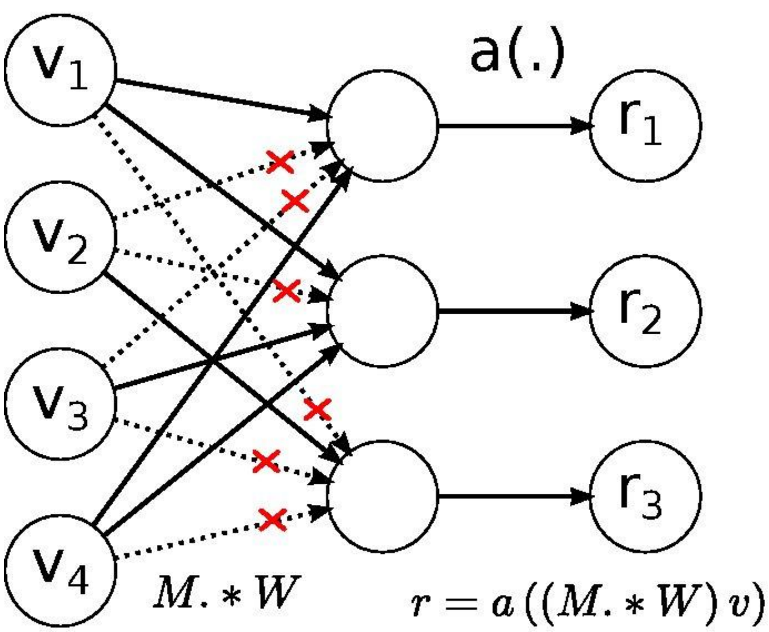
(b) After applying dropout.

Current Approaches to solving overfitting:

DropConnect [Wan et. al, 2013]



Regular Network:



DropConnect Network:

But ……

Can we harness the benefits of both small and large networks

Larger networks:

Pro's:
- Learn quickly
- Less sensitive to initial condition
- Less sensitive to local minima

Con's:
- Overfitting

Smaller networks:

Pro's:
- Generalize better
- Faster build
- Faster to compute
- Easier understand

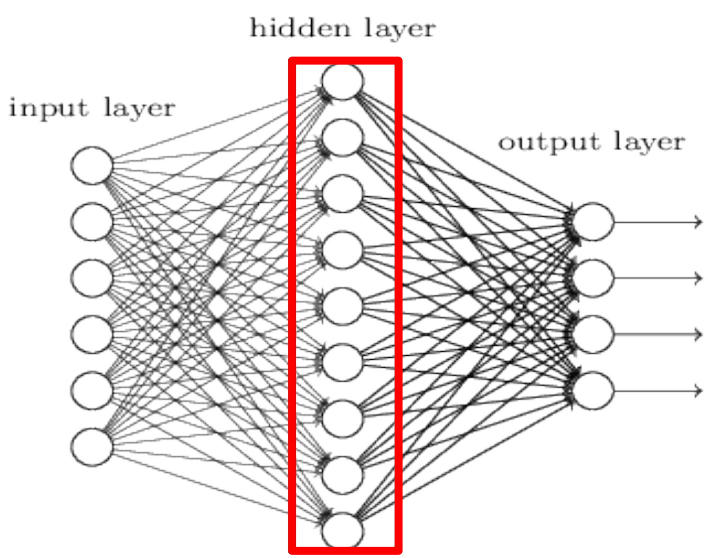
Con's:
- Underfitting

[Reed, 1993]

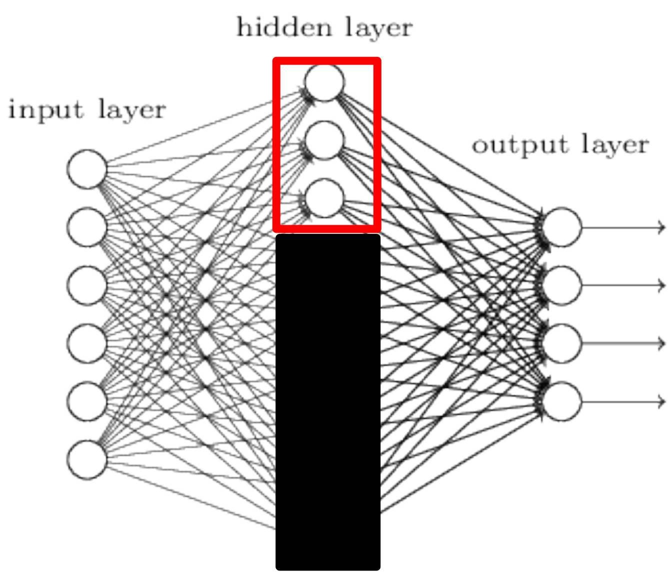
Lets try pruning ……

Can we train on a large network and remove hidden units to make the network smaller

Train:



Prune:



Pruning Neural Networks:

Neural Net Pruning - Why and How [Sietsma et. al, 1988]

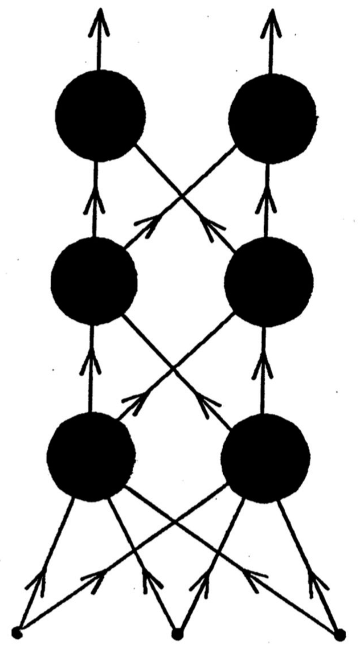
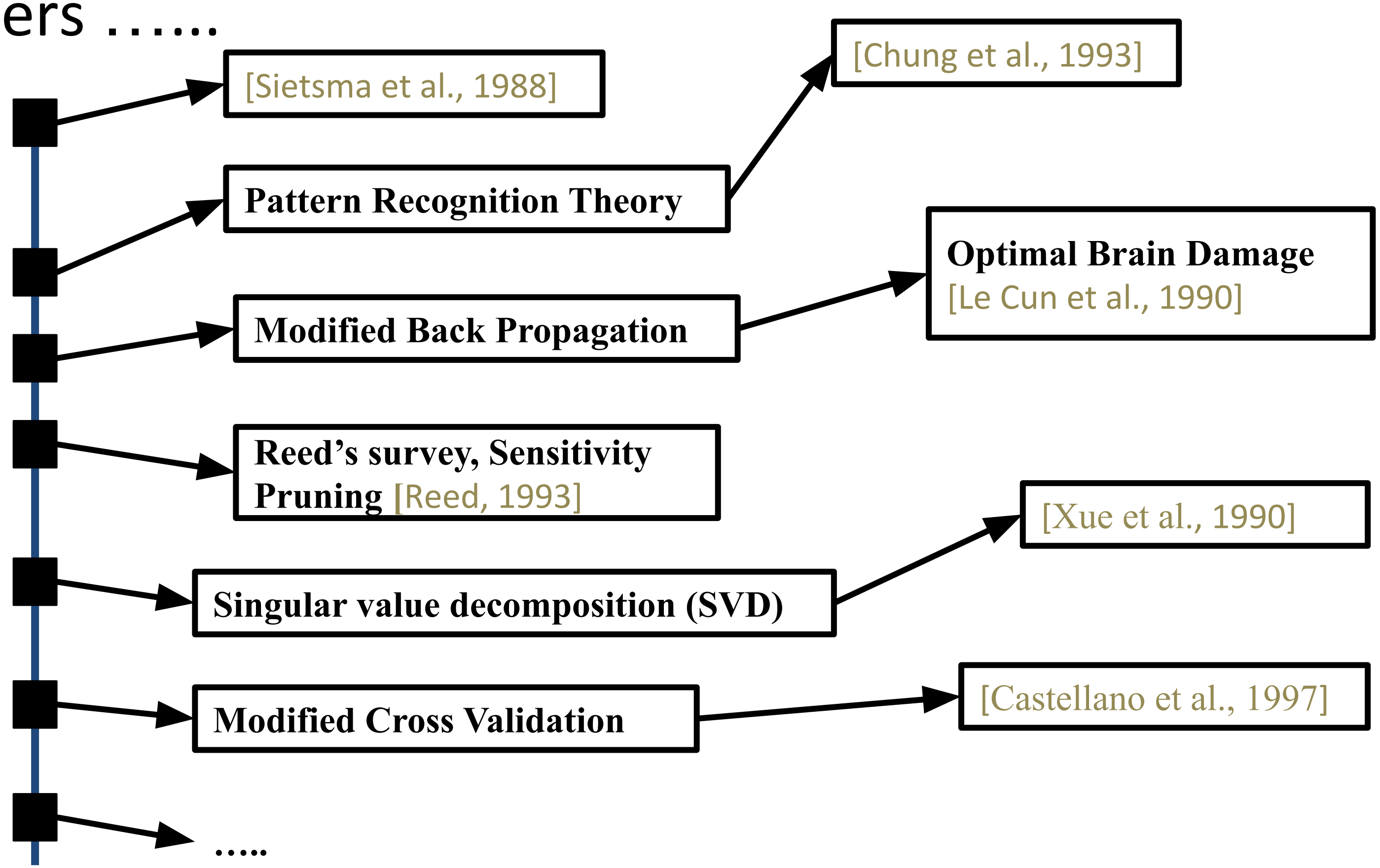


Figure 1.

Pattern	First Layer Outputs				
	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5
straight lines					
1	0.1	1	0	0	1
2	0.1	1	0	0	1
3	0.1	1	0	0	1
wavy lines					
4	0.1	0	0	0	1
5	0.2	1	1	1	0
6	0.2	1	1	0	0

Pruning Neural Networks:

Others ......



Research Objective

Develop a pruning technique that can be applied to a fully connected layer of a neural network to addresses two issues that neural networks face: overfitting and tuning hyperparameters

Research Objective

Our Contributions…

- A novel pruning technique that can be applied to fully connected hidden layer of a neural network to address overfitting and tuning the network configuration.

- Demonstration of BCAP's ability to prune data sets in two domains: Life science and Vision

BCAP: Model Description

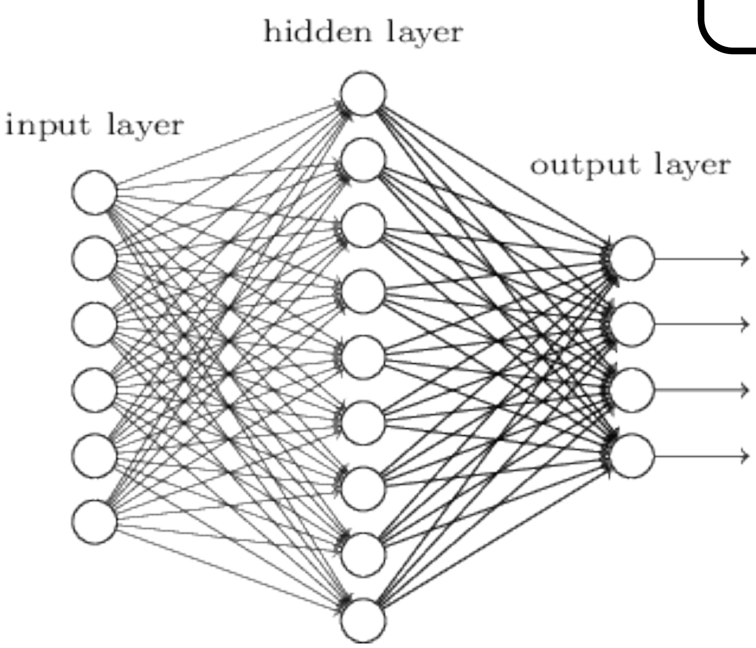
Finding Prunable hidden Units

Pruning Hidden units

Error Metric

Stopping Criteria

Groups and Cosine Curve Approximation



Feedforward Network:

Definitions:

n - number of records in the data,

l - index for fully connected hidden layer

h_l- number of hidden units in a layer l

x^l - output of layer l with dimension h by n

w^l - weights matrix associated with the layer l

b^l - bias matrix associated with the layer l

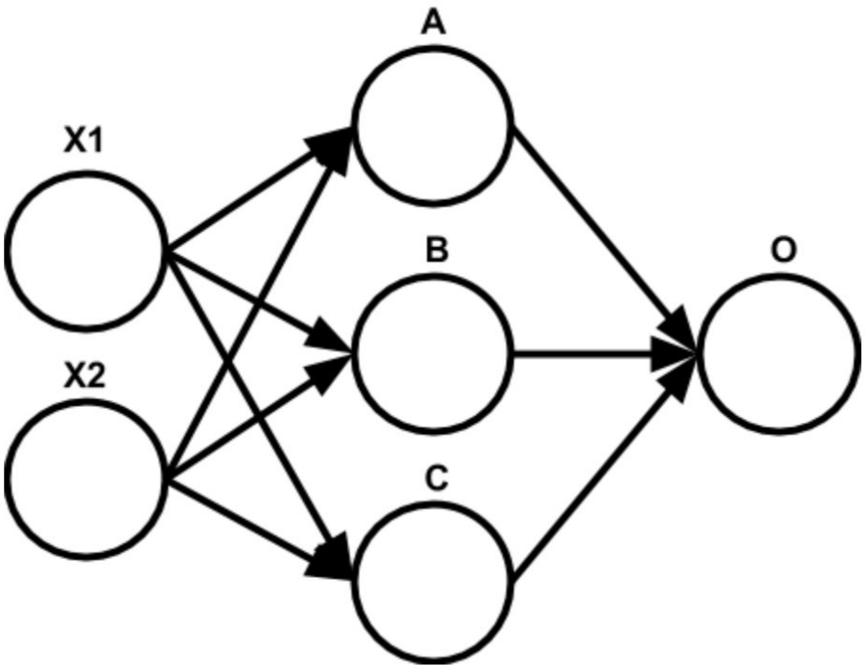
$f(x)$ - activation function

$z^{(l+1)}$ - input matrix to l+1

Formal Feedforward Operation:

$$z^{(l+1)} = w^{(l+1)} x^l + b^{(l+1)}$$

$$x^{(l+1)} = f(z^{(l+1)})$$



BCAP Modification on FeedForward Network:

BCAP can be described as mapping the h dimensions of the weight and bias vectors to h', where $h' <= h$

Original FeedForward Model:

$$z^{(l+1)} = w^{(l+1)} x^{l} + b^{(l+1)}$$

$$x^{(l+1)} = f(z^{(l+1)})$$

$$BCAP : w^{(l+1)} \rightarrow \Delta w^{(l+1)}_{h'_l \, x \, n}$$

$$b^{(l+1)}_{h_l \, x \, 1} \rightarrow \Delta b^{(l+1)}_{h'_l \, x \, 1}$$

BCAP FeedForward Model:

$$z^{(l+1)} = \Delta w^{(l+1)} x^{l} + \Delta b^{(l+1)}$$

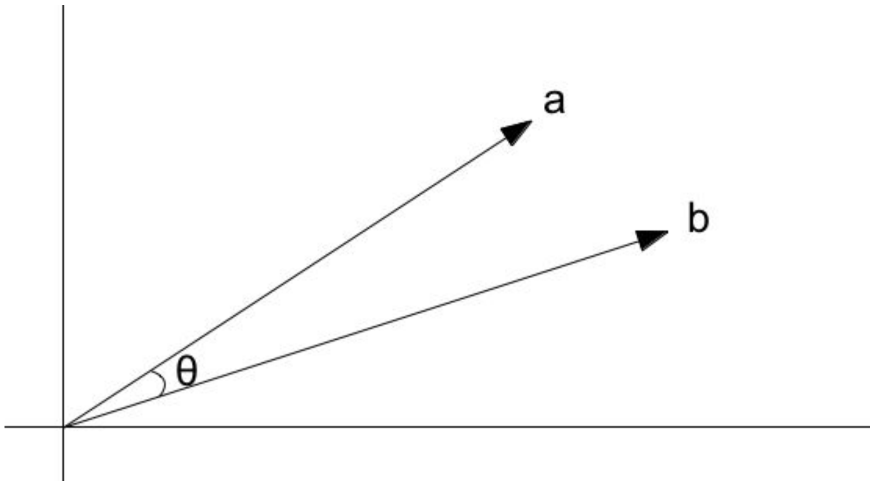
$$x^{(l+1)} = f(z^{(l+1)})$$

$$\Delta w^{(l+1)}, \Delta b^{(l+1)} = bcap(x^{l}, w^{(l+1)}, b^{(l+1)})$$

Finding Prunable hidden Units:

A node is prunable if it produces output **similar** to another node in the same layer and the cosine similarity is used to detect the similarity between hidden units.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$



Finding Prunable hidden Units:

$$Z = W x + b$$

Two Records

$$Z = \begin{pmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{31} \end{pmatrix} \begin{pmatrix} \boxed{\begin{matrix} 8 \\ 3 \end{matrix}} & \boxed{\begin{matrix} 9 \\ 4 \end{matrix}} \end{pmatrix} + \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

$$Z = \begin{pmatrix} (8W_{11} + 3W_{12}) + b_1 & (9W_{11} + 4W_{12}) + b_2 \\ (8W_{21} + 3W_{22}) + b_1 & (9W_{21} + 4W_{22}) + b_2 \\ (8W_{31} + 3W_{32}) + b_1 & (9W_{31} + 4W_{31}) + b_2 \end{pmatrix}$$

$$Z = \begin{array}{cc} \begin{pmatrix} \boxed{.5 \quad .7} \\ \boxed{.2 \quad .3} \\ \boxed{.5 \quad .7} \end{pmatrix} & \begin{matrix} A \\ B \\ C \end{matrix} \end{array}$$

Three hidden hidden units

BCAP: Model Description

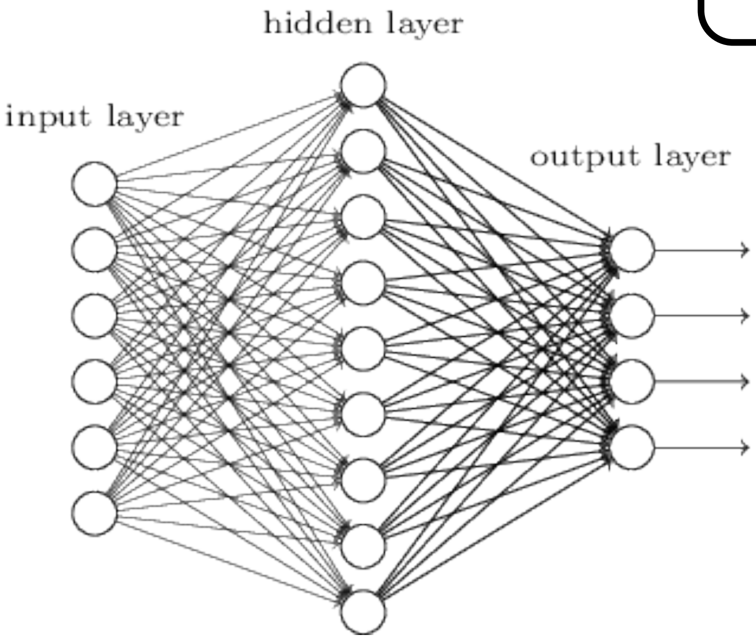
Finding Prunable hidden Units ✓

Pruning Hidden units

Error Metric

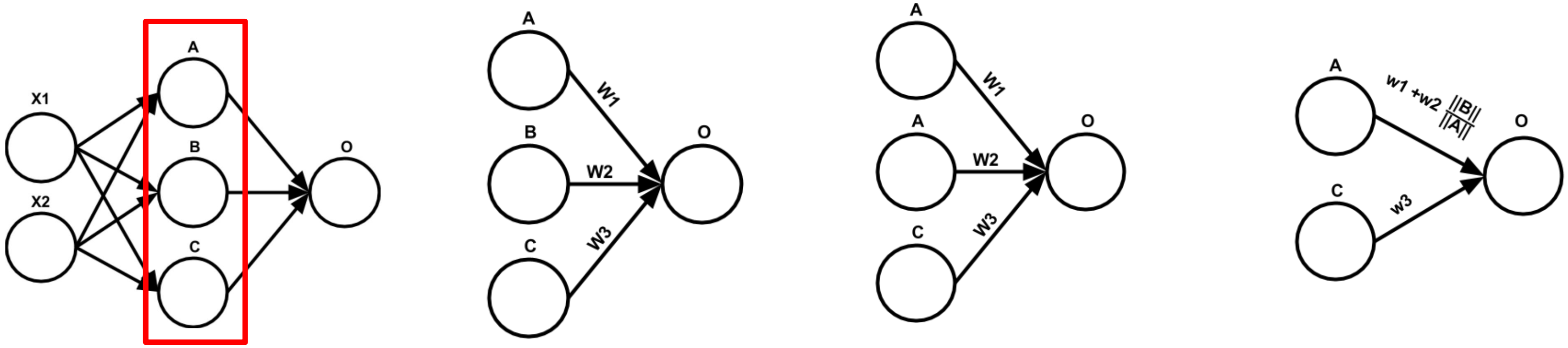
Stopping Criteria

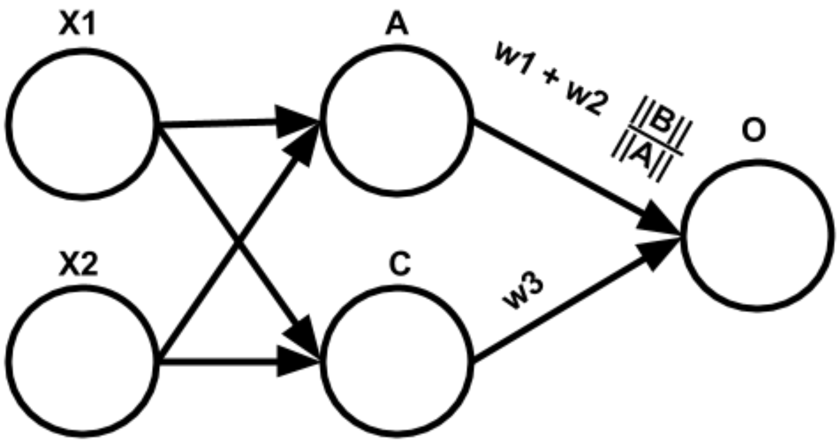
Groups and Cosine Curve Approximation



Pruning Hidden Units:

When dealing with a fully-connected hidden layer, every hidden unit receives the same input from the previous layer





Pruning Hidden Units:

We formally describe how to mathematically combine hidden units:

Given a hidden layer x^l, with two hidden units β_1 and β_2 that are similar based on the cosine similarity measure:

Using the definition of feed-forward:

$$z^{(l+1)} = w^{(l+1)} x^l + b^{(l+1)}$$

$$z^{(l+1)} = \sum_{i=1}^{n} w_i^{(l+1)} x_i^l + b_i^{(l+1)}$$

We expand the summation in order to set up the merge for pruning β_2:

$$z^{(l+1)} = (w_1^{(l+1)} h_1^l + b_1^{(l+1)}) + (w_2^{(l+1)} h_2^l + b_2^{(l+1)}) + (w_3^{(l+1)} h_3^l + b_3^{(l+1)}) + \ldots\ldots\ldots\ldots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)})$$

Using the expanded version of the feed-forward network and our assumption that β_1 and β_2 are similar, without loss of generality, let $\beta_1 = h_1^l$ and $\beta_2 = h_2^l$, so:

$$z^{(l+1)} = (w_1^{(l+1)} \beta_1 + b_1^{(l+1)}) + (w_2^{(l+1)} \beta_2 + b_2^{(l+1)}) + (w_3^{(l+1)} h_3^l + b_3^{(l+1)}) + \ldots\ldots\ldots\ldots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)})$$

Since, β_1 is similar to β_2 in direction, we scale β_1 magnitude to β_2 by $\beta_1 \frac{\|\beta_2\|}{\|\beta_1\|}$ and replacing β_2 by this scaled value:

$$\Delta z^{(l+1)} = (w_1^{(l+1)} \beta_1 + b_1^{(l+1)}) + (w_2^{(l+1)} \beta_1 \frac{\|\beta_2\|}{\|\beta_1\|} + b_2^{(l+1)}) + (w_3^{(l+1)} h_3^l + b_3^{(l+1)}) + \ldots\ldots\ldots\ldots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)})$$

$$\Delta z^{(l+1)} = [(w_1^{(l+1)} + w_2^{(l+1)} \frac{\|\beta_2\|}{\|\beta_1\|}) \beta_1 + (b_1^{(l+1)} + b_2^{(l+1)})] + (w_3^{(l+1)} b + b_3^{(l+1)}) + \ldots\ldots\ldots\ldots + (w_n^{(l+1)} h_n^{(l+1)} + b_n^{(l+1)})$$

BCAP: Model Description

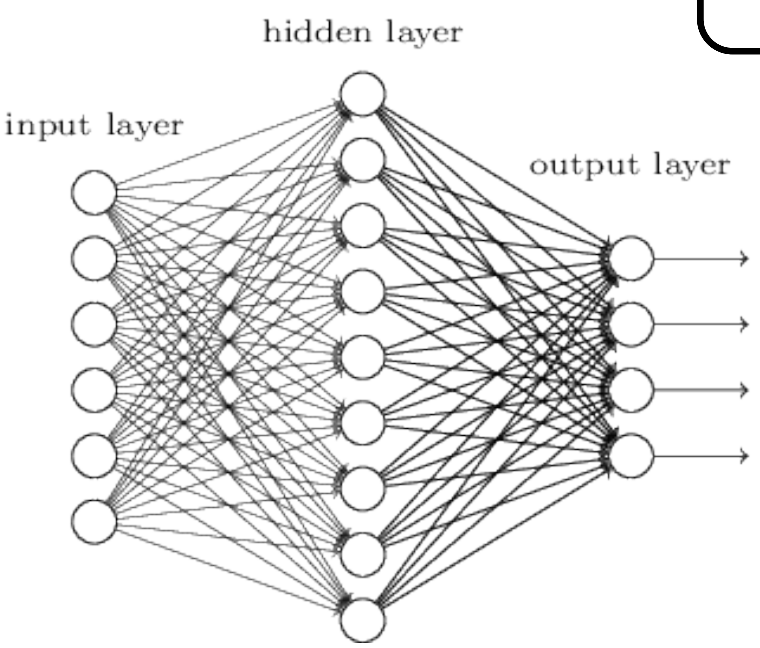
Finding Prunable hidden Units ✓

Pruning Hidden units ✓

Error Metric

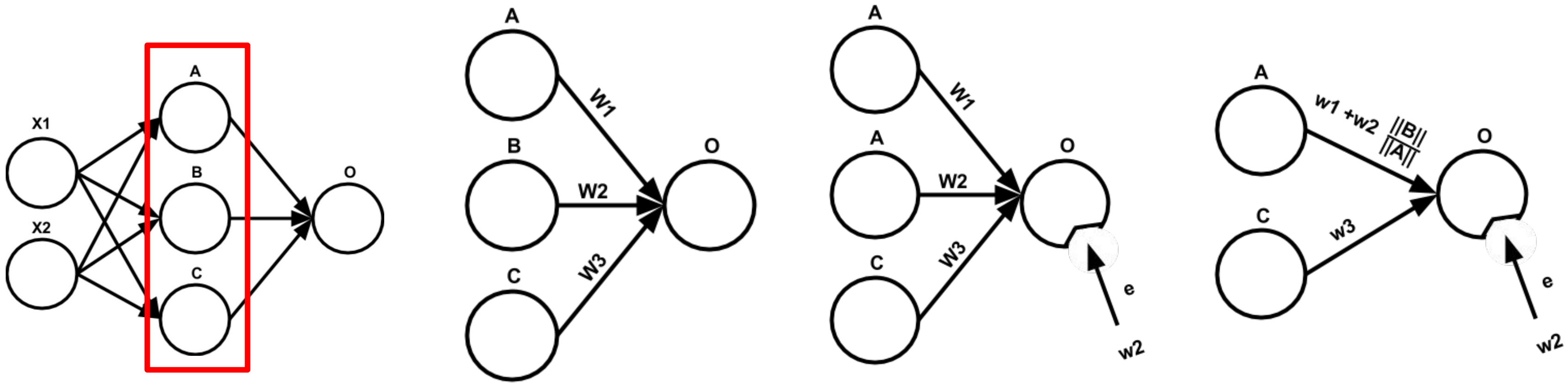
Stopping Criteria

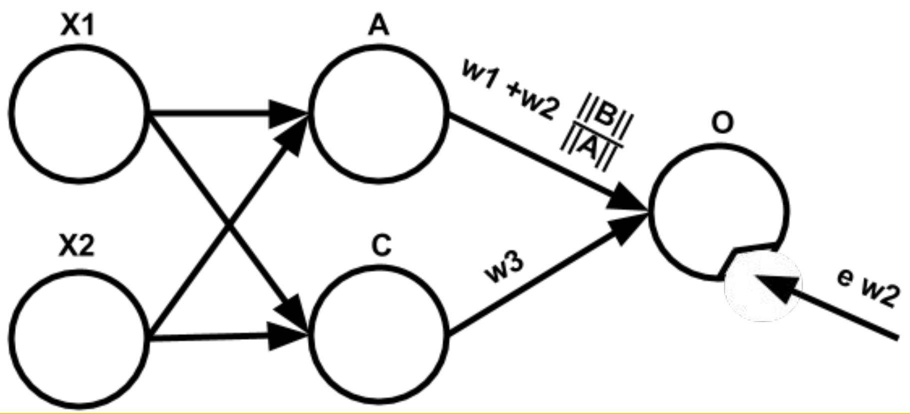
Groups and Cosine Curve Approximation



Error Metric:

In practice, if there are hidden units in a hidden layer *l* that are similar, the cosine similarity measurement produced is not exactly 1





Error Metric:

Formally we describe how to mathematically the error that is introduced:

If the same two hidden units β_1 and β_2 are similar but their cosine similarity is not exactly 1, that means that $\beta_1 \frac{\|\beta_2\|}{\|\beta_1\|} \approx \beta_2$ or $\beta_1 \frac{\|\beta_2\|}{\|\beta_1\|} = \beta_2 + e$, where e is some error difference between the two hidden units.

$$z^{(l+1)} = (w_1^{(l+1)} \beta_1 + b_1^{(l+1)}) + (w_2^{(l+1)} \beta_2 + b_2^{(l+1)}) + \dots\dots\dots\dots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)})$$

Since, $\beta_1 \frac{\|\beta_2\|}{\|\beta_1\|} - e$ is similar to β_2 :

$$\Delta z^{(l+1)} = (w_1^{(l+1)} \beta_1 + b_1^{(l+1)}) + (w_2^{(l+1)} (\beta_1 \frac{\|\beta_2\|}{\|\beta_1\|} - e) + b_2^{(l+1)}) + \dots\dots\dots\dots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)})$$

$$\Delta z^{(l+1)} = (w_1^{(l+1)} \beta_1 + b_1^{(l+1)}) + (w_2^{(l+1)} \beta_1 \frac{\|\beta_2\|}{\|\beta_1\|} + b_2^{(l+1)}) + \dots\dots\dots\dots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)}) - \boxed{(w_2^{(l+1)} e - b_2^{(l+1)})]}$$

$$\Delta z^{(l+1)} = [(w_1^{(l+1)} + w_2^{(l+1)} \frac{\|\beta_2\|}{\|\beta_1\|}) \beta_1 + (b_1^{(l+1)} + b_2^{(l+1)})] + \dots\dots\dots\dots + (w_n^{(l+1)} h_n^l + b_n^{(l+1)}) - \boxed{[(w_2^{(l+1)} e - b_2^{(l+1)})]}$$

Error Metric:

In general we would like to minimize the effect on the network's performance capability with respect on the error e discussed because we do not know if the effect is positive (i.e. increasing generalization) or negative (decreasing generalization).

Our error measurement is based on the Mean Square Error (MSE):

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Error Metric:

The MSE equation defined above is not sufficient enough for a error metric because it has depends on the size of hidden layer. Dividing by size of the hidden layer shares the error measure across all the nodes in the layer, rather than letting one node's change dominate the error

$$\text{MMSE} = m(\hat{y}, y) = \frac{E[((\hat{y}-y)w)^2]}{h^l}$$

$$= \frac{\frac{1}{n}\sum_{i=1}^{n}((\hat{y_i}-y_i)w)^2}{h^l}$$

BCAP: Model Description

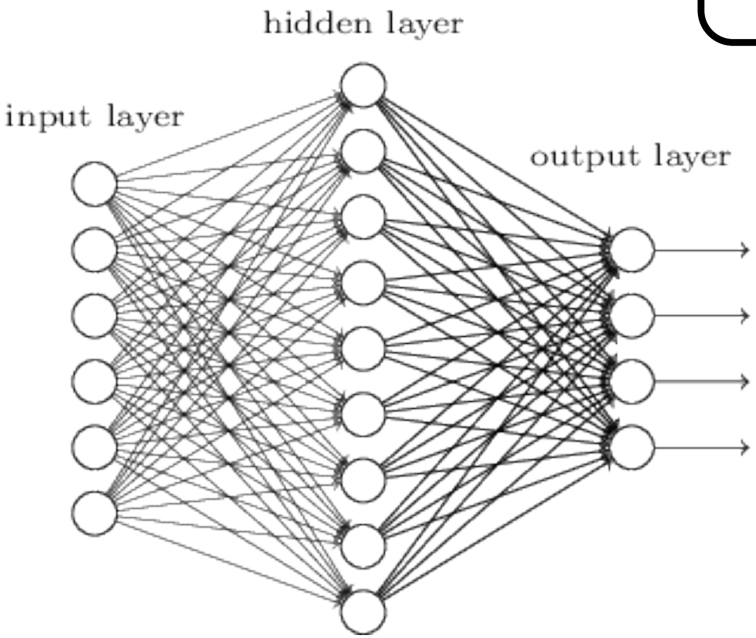
Finding Prunable hidden Units ✓

Pruning Hidden units ✓

Error Metric ✓

Stopping Criteria

Groups and Cosine Curve Approximation



Stopping Criteria:

We would like prune hidden units from the most similar to least similar, where the maximum amount of error introduced in the network from all the prunes is less than ε.

Hidden units pairs:	Cosine angle (smallest to largest):	MMSE per pair:	Total MMSE error:
(β_1, β_2)	$\theta_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)} \leq \varepsilon$
(β_2, β_3)	$\theta_{(\beta_2,\beta_3)}$	$e_{(\beta_2,\beta_3)}$	
(β_1, β_3)	$\theta_{(\beta_1,\beta_3)}$	$e_{(\beta_1,\beta_3)}$	

Stopping Criteria:

We would like prune hidden units from the most similar to least similar, where the maximum amount of error introduced in the network from all the prunes is less than ε.

Hidden units pairs:	Cosine angle (smallest to largest):	MMSE per pair:	Total MMSE error:
(β_1, β_2)	$\theta_{(\beta_1, \beta_2)}$	$e_{(\beta_1, \beta_2)}$	$e_{(\beta_1, \beta_2)} \leq \varepsilon$
(β_2, β_3)	$\theta_{(\beta_2, \beta_3)}$	$e_{(\beta_2, \beta_3)}$	$(e_{(\beta_1, \beta_2)} + e_{(\beta_2, \beta_3)}) \leq \varepsilon$
(β_1, β_3)	$\theta_{(\beta_1, \beta_3)}$	$e_{(\beta_1, \beta_3)}$	

Stopping Criteria:

We would like prune hidden units from the most similar to least similar, where the maximum amount of error introduced in the network from all the prunes is less than ε.

Hidden units pairs:	Cosine angle (smallest to largest):	MMSE per pair:	Total MMSE error:
(β_1,β_2)	$\theta_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)} \leq \varepsilon$
(β_2,β_3)	$\theta_{(\beta_2,\beta_3)}$	$e_{(\beta_2,\beta_3)}$	$(e_{(\beta_1,\beta_2)} + e_{(\beta_2,\beta_3)}) \leq \varepsilon$
(β_1,β_3)	$\theta_{(\beta_1,\beta_3)}$	$e_{(\beta_1,\beta_3)}$	$(e_{(\beta_1,\beta_2)} + e_{(\beta_2,\beta_3)} + e_{(\beta_1,\beta_3)}) \geq \varepsilon$

Stopping Criteria:

One way to find the optimal amount of prunes with respect the stopping parameter ε is by iterating over all the possible prunable hidden units from the most similar to least similar.

If hidden layer *l* has *n* hidden units, the number of 2-combinations that can be formed is roughly n^2:

$$\binom{N}{2} = \frac{N(N-1)}{2} \approx \frac{N^2}{2}$$

BCAP: Model Description

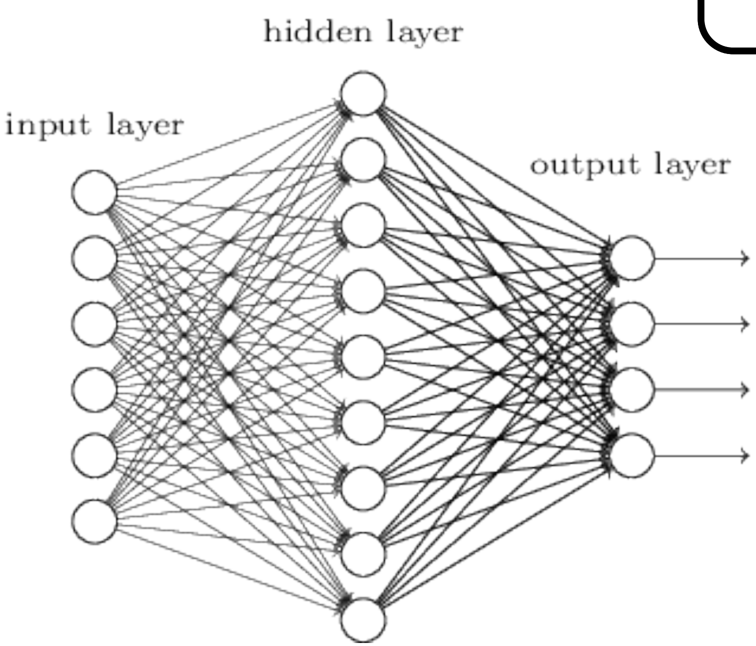
Finding Prunable hidden Units ✓

Pruning Hidden units ✓

Error Metric ✓

Stopping Criteria ✓

Groups and Cosine Curve Approximation



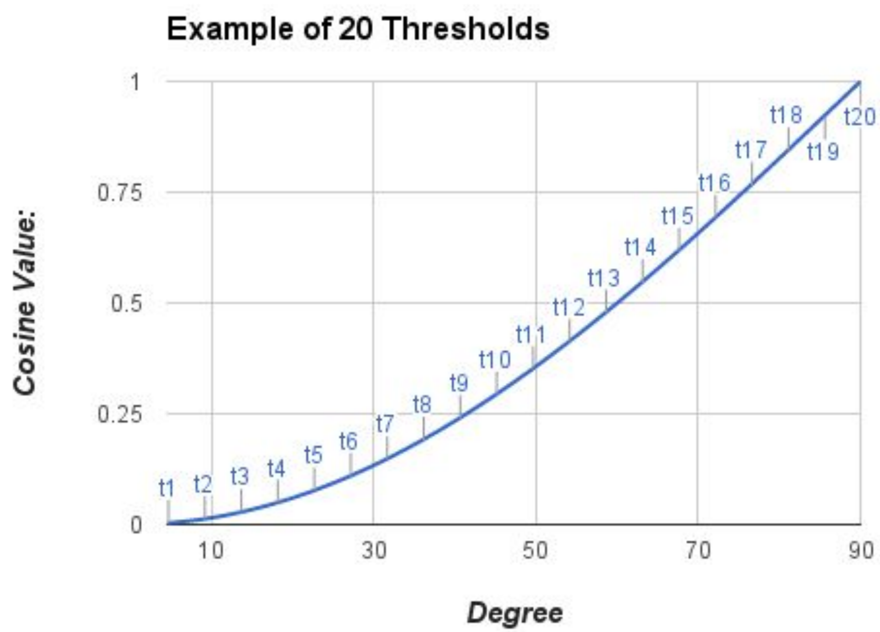
Groups and Cosine Curve Approximation:

This means that it will roughly take us quadratic time to explore all of angles produced from computing the similarity between hidden units. We would like to reduce this computation time by approximating the optimal angle associated with the last possible prunable hidden unit pair.

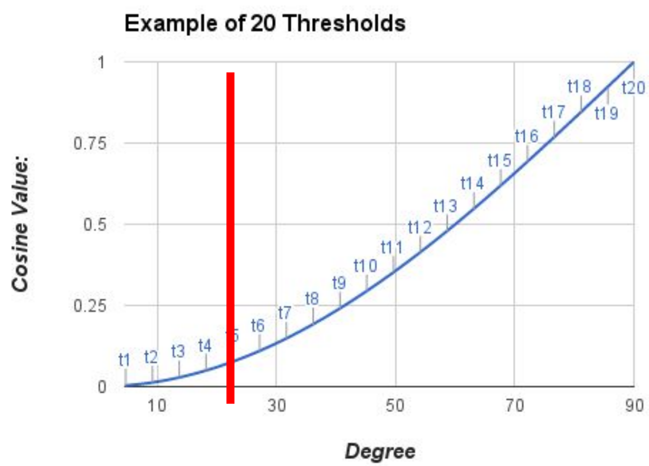
Hidden units pairs:	Cosine angle (smallest to largest):	MMSE per pair:	Total MMSE error:
(β_1,β_2)	$\theta_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)} \leq \varepsilon$
(β_2,β_3)	$\boxed{\theta_{(\beta_2,\beta_3)}}$	$e_{(\beta_2,\beta_3)}$	$(e_{(\beta_1,\beta_2)} + e_{(\beta_2,\beta_3)}) \leq \varepsilon$
(β_1,β_3)	$\theta_{(\beta_1,\beta_3)}$	$e_{(\beta_1,\beta_3)}$	$(e_{(\beta_1,\beta_2)} + e_{(\beta_2,\beta_3)} + e_{(\beta_1,\beta_3)}) \geq \varepsilon$

Groups and Cosine Curve Approximation:

The cosine similarity is a function that has a domain [0º,180º] and range [1,0]where the cosine of degree 0º is 1, cosine of degree 90º is 0, and the cosine of degree 180º is -1. We can divide the cosine similarity into i parts and use those values to find the optimal angle threshold denoted as t_i.



Groups and Cosine Curve Approximation:



the
.
al

Hidden units pairs:	Cosine angle (smallest to largest):	MMSE per pair:	Total MMSE error:
(β_1,β_2)	$\theta_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)}$	$e_{(\beta_1,\beta_2)} \le \varepsilon$
(β_2,β_3)	$\boxed{\theta_{(\beta_2,\beta_3)}}$	$e_{(\beta_2,\beta_3)}$	$(e_{(\beta_1,\beta_2)} + e_{(\beta_2,\beta_3)}) \le \varepsilon$
(β_1,β_3)	$\theta_{(\beta_1,\beta_3)}$	$e_{(\beta_1,\beta_3)}$	$(e_{(\beta_1,\beta_2)} + e_{(\beta_2,\beta_3)} + e_{(\beta_1,\beta_3)}) \ge \varepsilon$

Groups and Cosine Curve Approximation:

In addition to creating a series of t_i to find the optimal degree threshold, we need to combine similar hidden units into groups denoted as G_i. If we do not group hidden units, we would be performing the same amount of computation as the iterative version.

Similar Hidden unit pairs:	$(\beta_1,\beta_2),(\beta_2,\beta_3)$ $(\beta_3,\beta_4),(\beta_5,\beta_6)$	$(\beta_1,\beta_2),(\beta_2,\beta_3)$ $(\beta_3,\beta_4),(\beta_5,\beta_6)$
MMSE error:	$G_1 = e_{(\beta_1,\beta_2,\beta_3,\beta_4)}$ $G_2 = e_{(\beta_5,\beta_6)}$	$e_{(\beta_1,\beta_2)}, e_{(\beta_2,\beta_3)}$ $e_{(\beta_3,\beta_4)}, e_{(\beta_5,\beta_6)}$
	2 MMSE checks for 2 Groups	4 Individual MMSE checks

BCAP: Model Description

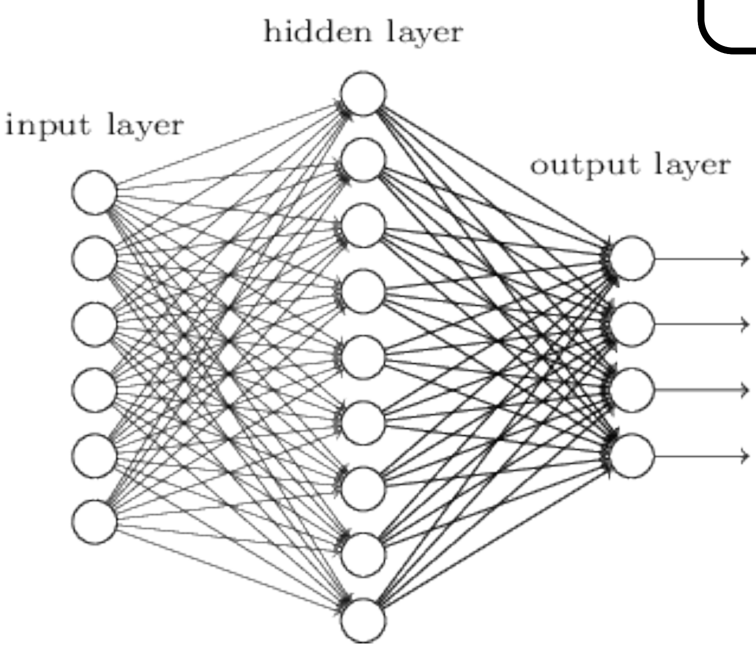
Finding Prunable hidden Units ✓

Pruning Hidden units ✓

Error Metric ✓

Stopping Criteria ✓

Groups and Cosine Curve Approximation ✓



BCAP Formal Algorithm:

Input: x^l - hidden units of layer l, $w^{(l+1)}$ - weights of layer l and $b^{(l+1)}$ - biases of layer l, t_i - threshold of similarity

Output: updated $\Delta w^{(l+1)}$ and $\Delta b^{(l+1)}$

Find Equivalent Hidden Units:

Let $M = SIM(x^{(l)^t}, x^{(l)})$

For each threshold t_i :

Let v_k and v_j be hidden units in x^l

Compare v_k and v_j using ~ *equivalence relation* $M_{k,j} \leq 1 - t_i$

Form Groups based on ~ *equivalence relation* formed from t_i

$G = \{G_1, G_2, \ldots\ldots\ldots, G_n\}$: Each G_i is $1 \; x \; h^l$ vector with 1's in index of *equivalence* units and 0's everywhere else.

Choose Pruning Groups:

For each G_i formed from t_i, compute the model MMSE $m(G_i)$ of the group. If $m(G_i) \leq \varepsilon$, accept the group. Denote the last accepted set of groups G_i as G_i', where:

$$G_i' = [G_1, G_2, \ldots\ldots\ldots, G_n]$$

Prune Hidden Units:

$$\Delta w^{(l+1)} = G_i' * w^{(l+1)}$$
$$\Delta b^{(l+1)} = G_i' * b^{(l+1)}$$

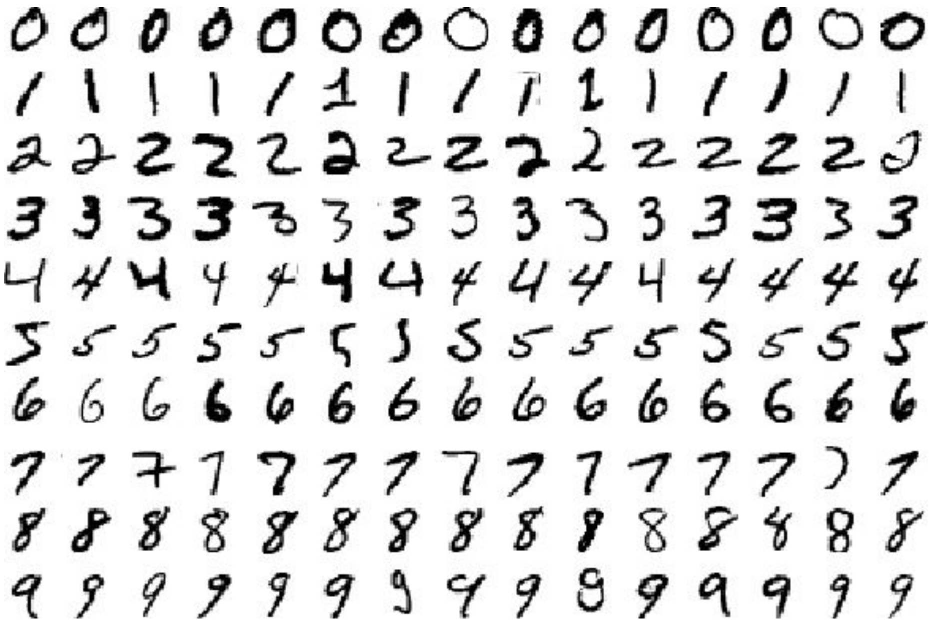
Experiments:

We evaluate the BCAP pruning technique on fully connected neural networks layers trained on various datasets in different domains for classifications tasks.

We intentionally use hidden layer sizes larger than necessary to take advantage of large networks ability to learn fast and then we either prune the network after or during training. If the training duration (i.e epochs) is too short, we avoid applying BCAP during training.

Experiments:

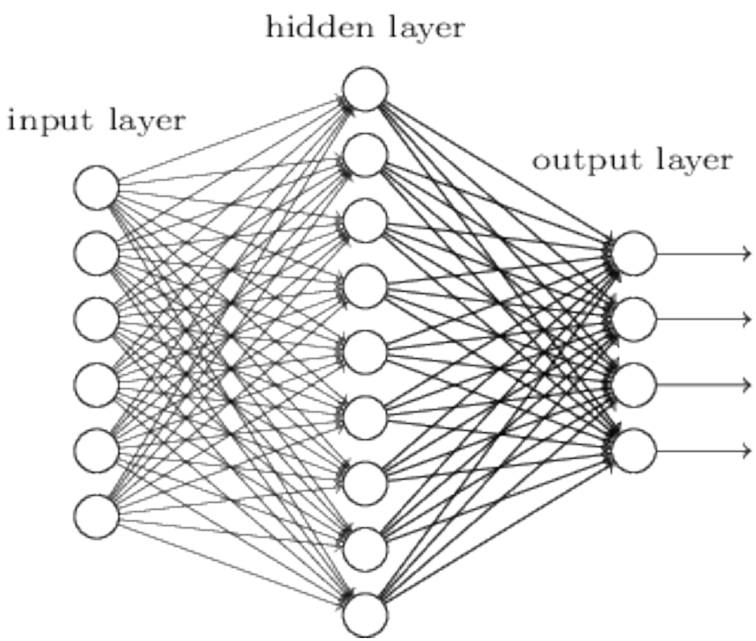
The MNIST data set contains 28 x 28 pixel black and white handwritten digit images. There are 10 handwritten digits in this data set which range from 0 to 9 (10-classes). The training and test set contains digits from each of the classes



Experiments: [2-layer MLP with 800 in each layer]

Method	Unit Type	Start Arch.	End Arch.	Error %
MLP (Simard et al., 2003)	Sigmoid	800-800 units	800-800 units	1.60%
MLP + DropOut (Hinton et al., 2014)	Sigmoid	800-800 units	800-800 units	1.35%
MLP + DropOut (Hinton et al., 2014)	ReLU	800-800 units	800-800 units	1.25%
MLP + DropOut + BCAP	ReLU	800-800 units	795-509 units	1.17%

Pruning Analysis



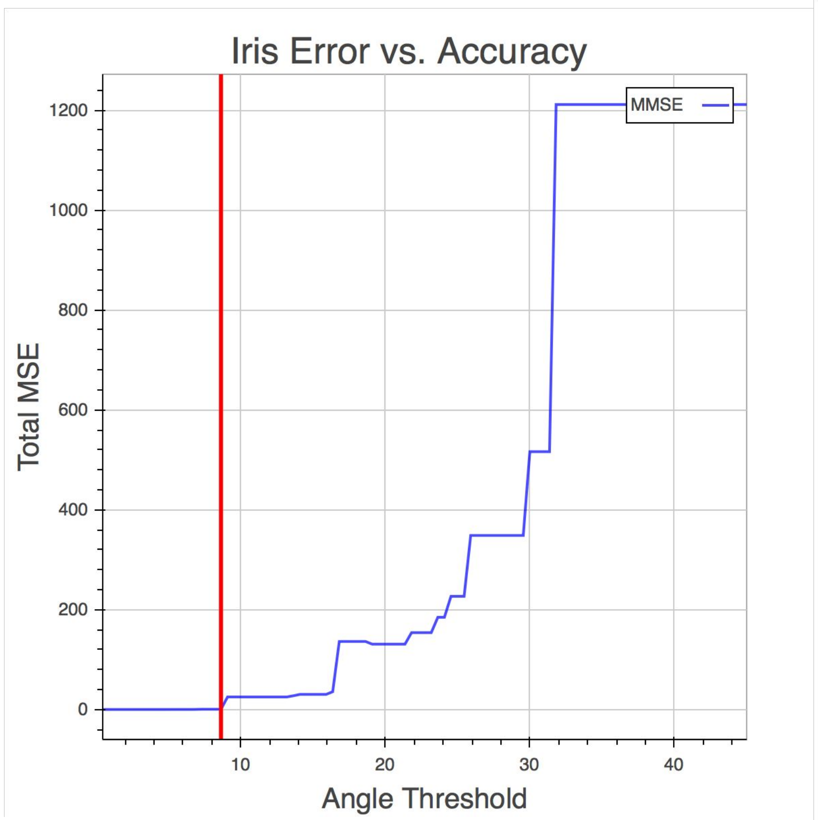
Tuning of BCAP Error ε

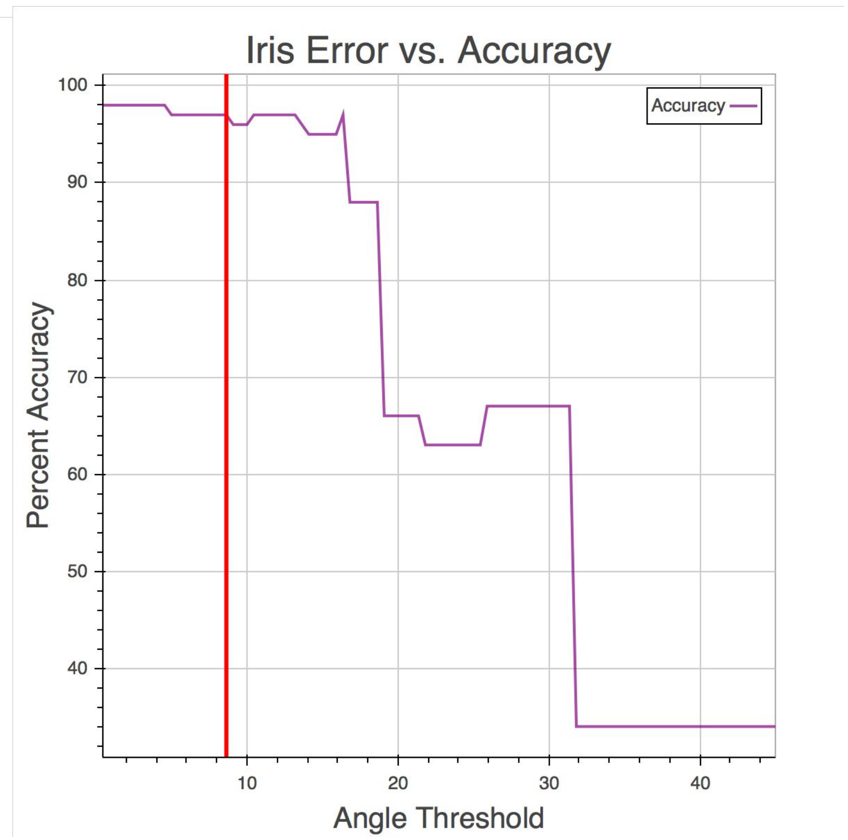
Tradeoff Between Accuracy and Network Size

Pruning Epoch for Activation functions

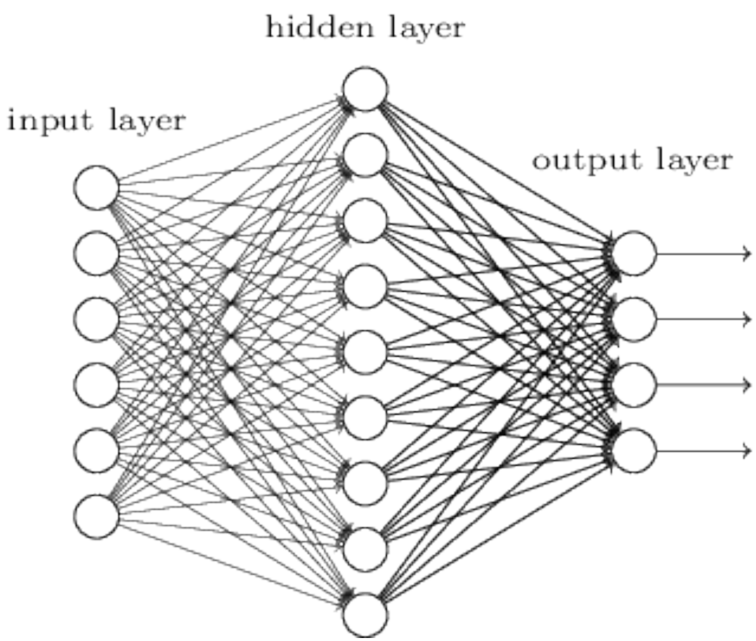
Tuning of BCAP Error ε:

This parameter decides on how much error will be introduced into the network after pruning.





Pruning Analysis



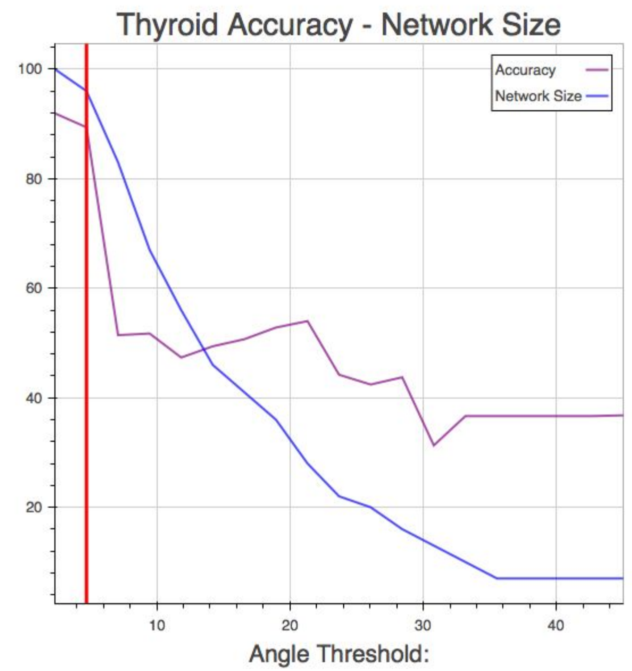
Tuning of BCAP Error ε ✓

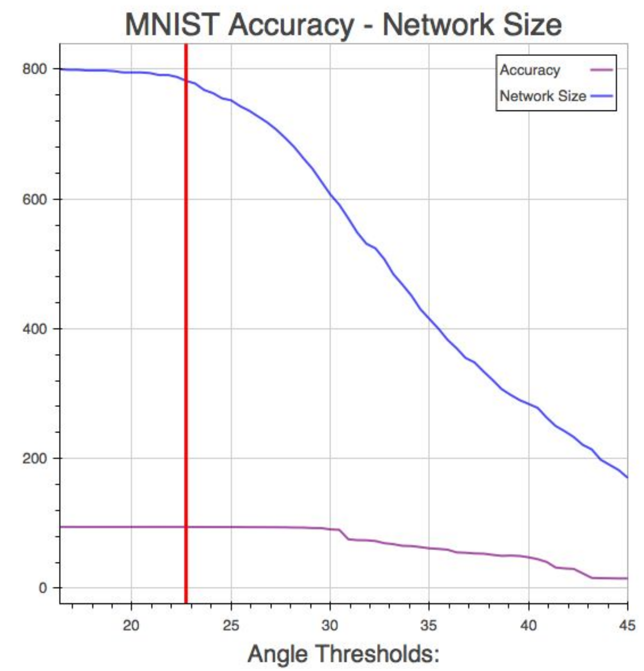
Tradeoff Between Accuracy and Network Size

Pruning Epoch for Activation functions

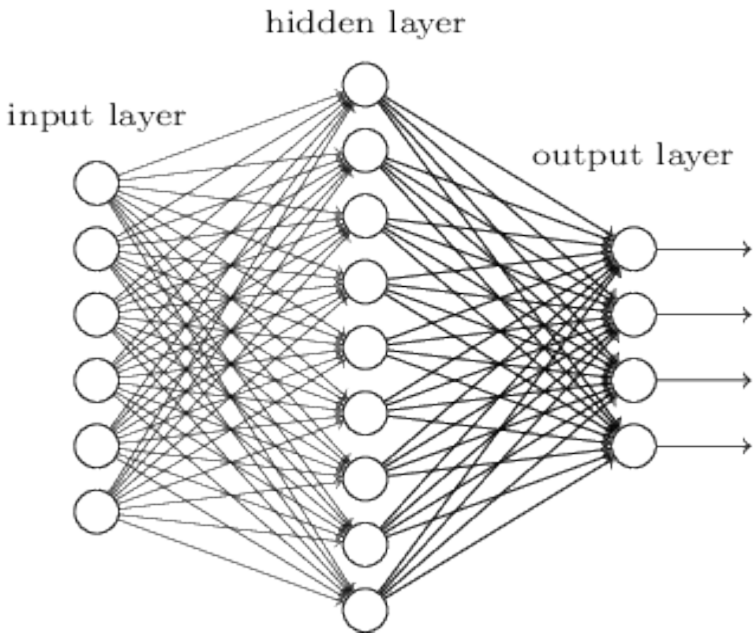
Tradeoff Between Accuracy and Network Size:

When pruning neural networks there is an inherent trade off between the neural network size and accuracy. We would like to minimize the network size and maximize the accuracy of the network with respect to the test data set





Pruning Analysis

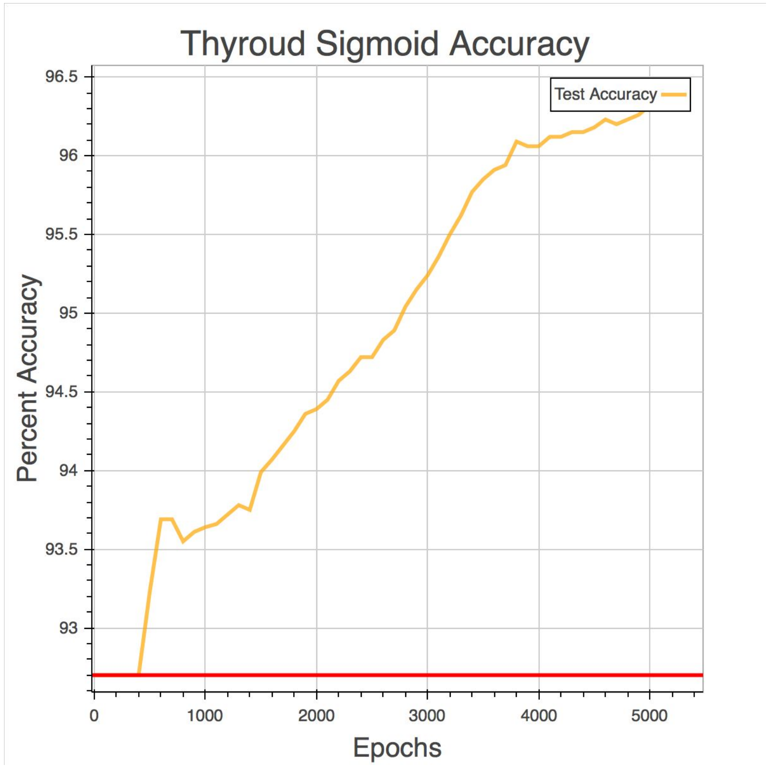


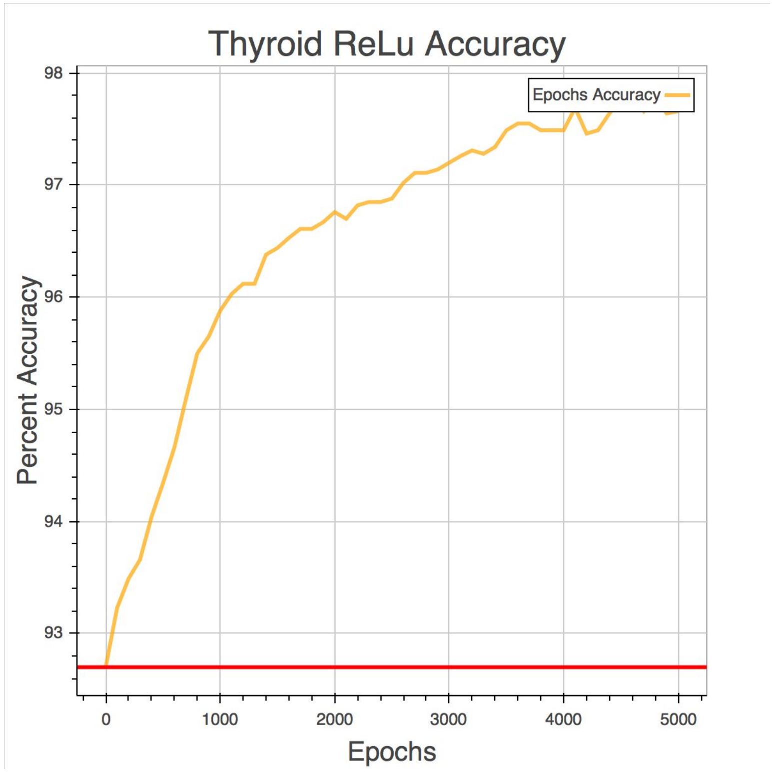
Tuning of BCAP Error ε ✓

Tradeoff Between Accuracy and Network Size ✓

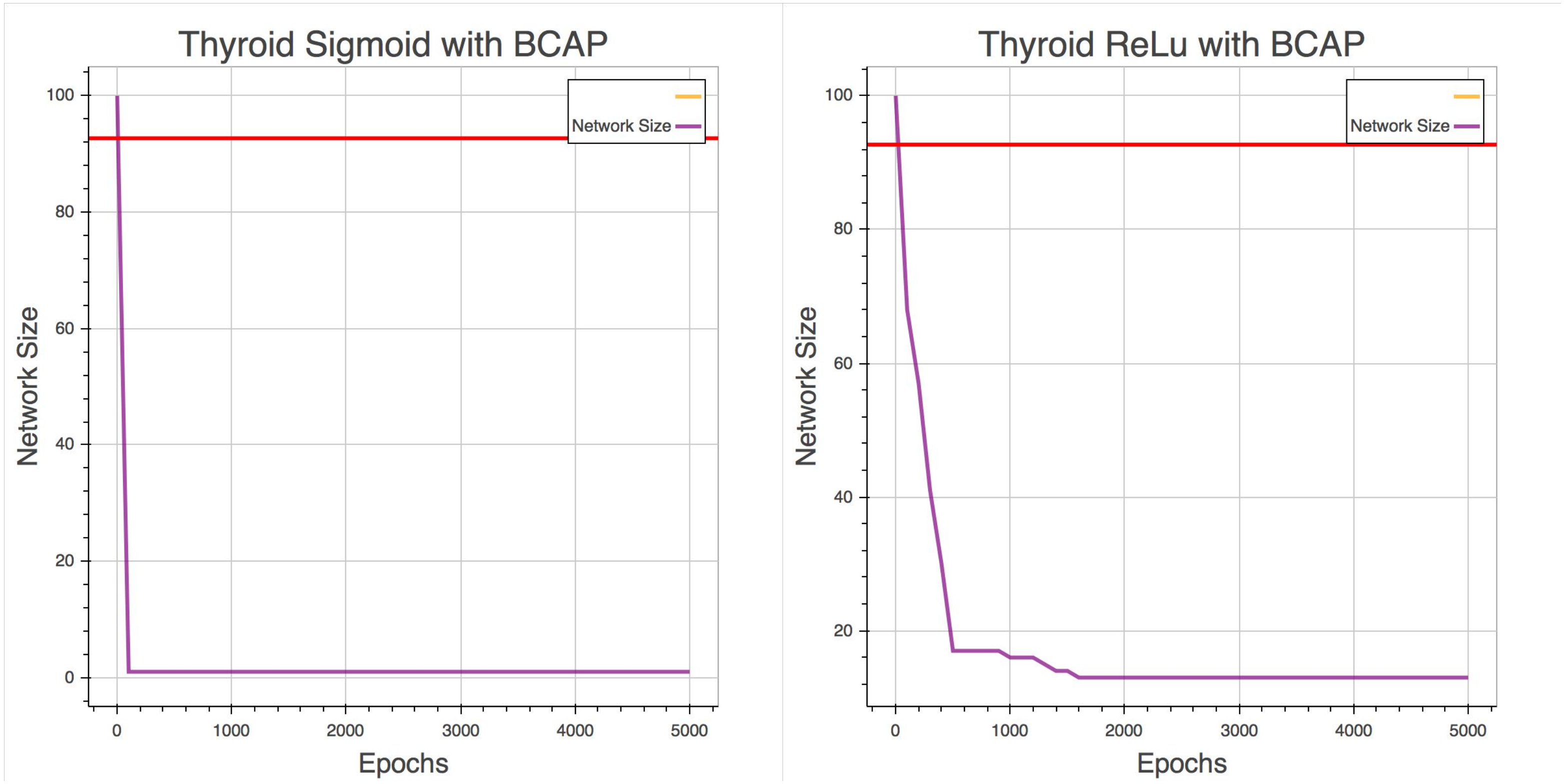
Pruning Epoch for Activation functions

Pruning Epoch for Activation functions:





Pruning Epoch for Activation functions:



Conclusions from our work

- Method for pruning a fully connected layer of a neural network
- Evidence that pruning neural network can be effective

Future work

- Explore other types of neural networks besides MLP
- Exploring neural networks with more than 2 hidden layers
- Evaluate pruning hidden units whose directions are opposite

Future work

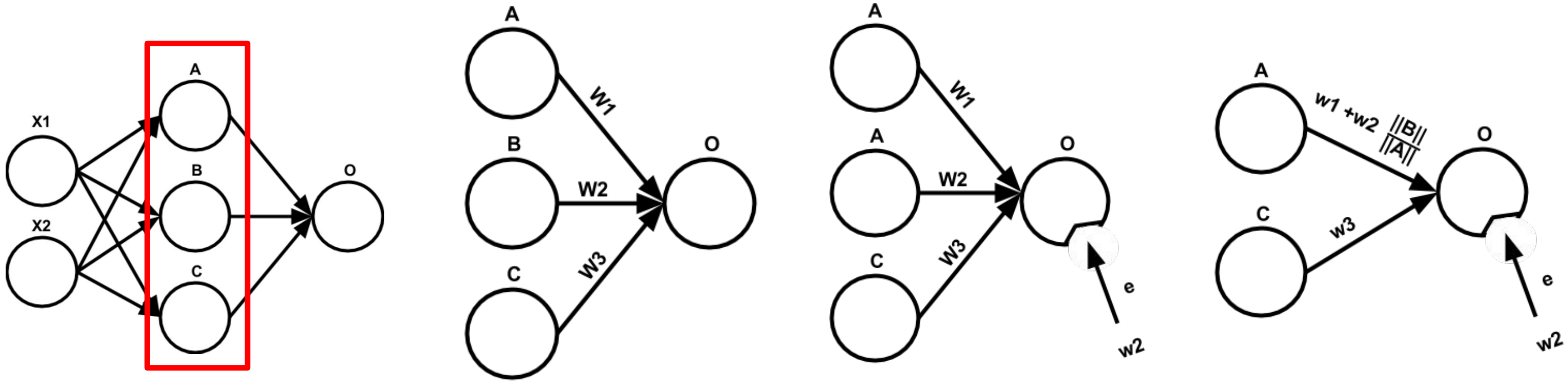
- Explore other types of neural networks besides MLP
-
-

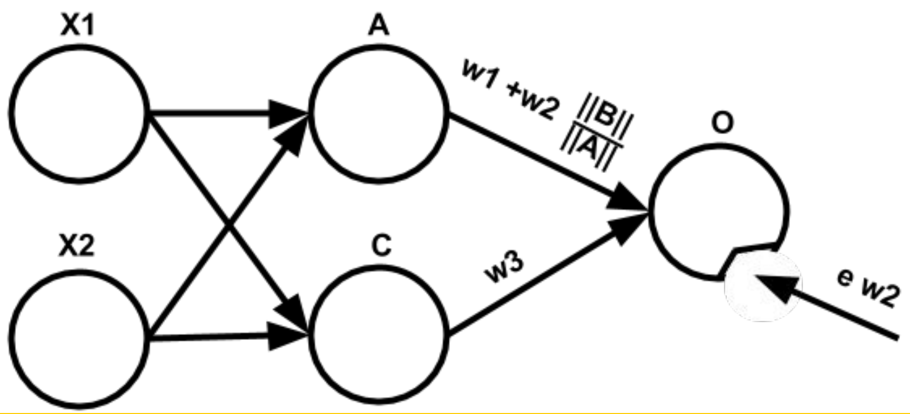
Our method is a promising technique to help reduce overfitting and increasing generalization in a neural network.

References

- Reed, Russell. "Pruning Algorithms - A Survey." IEEE Transactions On Neural Networks 4.5 (1993)
- Srivastava, Nitish, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." Journal of Machine Learning Research (2014)
- Wan, Li, Matthew Zeiler, Sixin Zhang, Yann LeCun, and Robert Fergus. "Regularization of Neural Networks Using DropConnect."
- Cun, Yann Le, John S. Denker, and Sara A. Sola. "Optimal Brain Damage." (1990)
- CHUNG, F.L. "A NODE PRUNING ALGORITHM FOR BACKPROPAGATION NETWORKS." International Journal of Neural Systems 03,.03 (1993)
- Schiffmann, W., M. Joost, and R. Werner. "Synthesis and Performance Analysis of Multilayer Neural Network Architectures." (1992):
- Xue, Qiuzhen, Yu Hen Hu, and Paul Milenkovic. "Analyses of the Hidden Units of the Multi-layer Perceptron and Its Application in Acoustic-to-articulatory Mapping." IEEE Transactions On Neural Networks 1.4 (1990)
- Castellano, G., A.m. Fanelli, and M. Pelillo. "An Iterative Pruning Algorithm for Feedforward Neural Networks." IEEE Trans. Neural Netw. IEEE Transactions on Neural Networks 8.3 (1997)

Thank you


X1
X2
A
B
C
O
A
B
C
O
W1
W2
W3
A
A
C
O
W1
W2
W3
e
w2
A
C
O
w1 +w2 ||B|| / ||A||
w3
e
w2


X1
X2
A
C
O
w1 +w2 ||B|| / ||A||
w3
e w2