## Q1 Hyperplane Geometry [1 mark]

a) Give the equation for a 2-dimensional hyperplane in $R^3$ which is orthogonal to the vector $w=(6,3,2)^T$ and has distance d=8 from the origin.

the equation of the plane perpendicular to w at distance 8 from origin is given by

$$f(\vec{x}) = \left\{\vec{x} \in \mathbb{R}^3, \frac{w}{||w||} \cdot \vec{x} = 8\right\}$$

where $||w|| = \sqrt{6^2 + 3^2 + 2^2} = \sqrt{49} = 7$ and $\vec{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$

Thus equation becomes

f= $\left(\frac{1}{7}\right)\begin{pmatrix} 6 \\ 3 \\ 2 \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ - 8 = 0

7 f(x,y,z) = $6x + 3y + 2z - 56$

b) Calculate the distance of the point p = (10, 1, 5) from the above hyperplane.

Substituting p into the plane equation gives:
f(10,1,5) = 6(10) + 3(1) + 2(5) – 56 = 73 - 56 =17

**7 f(10,1,5) = 17**
**(6(10)+3(1)+2(5)) /7 -8 = 2.4**

## Q2 Variations of the Two-Spiral Task [3 marks]

Perform an experimental study on the following variations of the two-spiral task:

a) (ANN training): Start with the "original dataset" of Lang and Witbrock (1988) with 194 training points (see Figure below). How fast and how well can you solve this task using a feed-forward NN? (dataset will be supplied in blackboard) [10% of Q2 mark]

**Problem Space:**
Binary classification problem (pixel is either one class or the other)
Two interwoven spirals that cannot be linearly separated (cant use perceptrons).

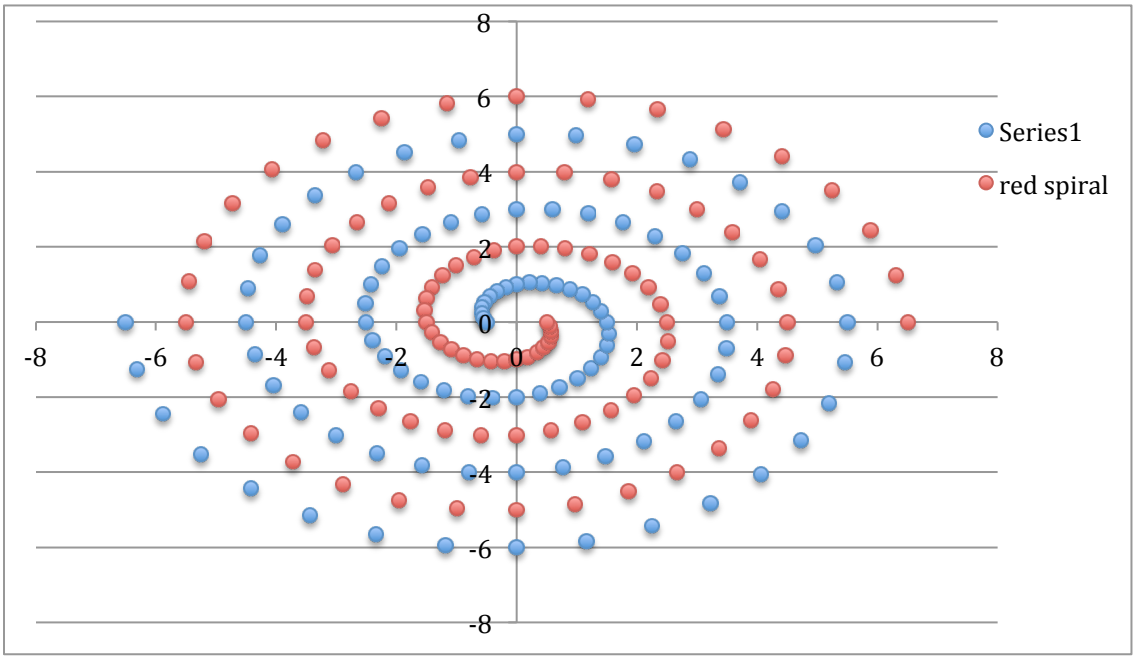



**Figure 1: Original Two-Spiral Training DataSet, ** (radius 6.5, density 1)**
**** Equations from 'Variations of the Two-Spiral Task', Chalup, Connection Science Vol 19, 2007**

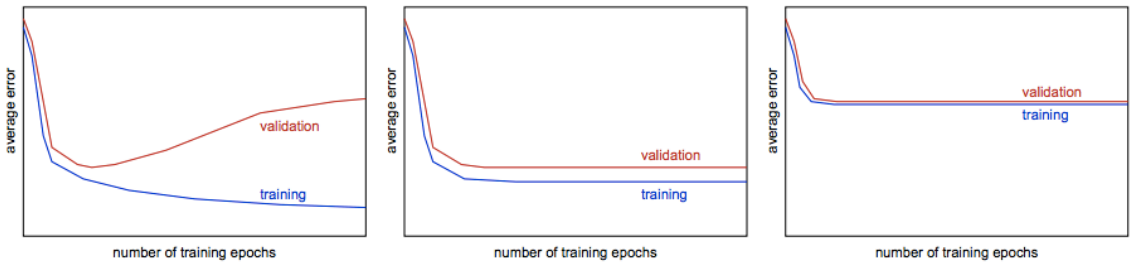



**Figure 2: Generalisation of Network.**
**Left (overfitted, divergence): training successful but doesn't generalize.**
**Middle (good, low avg error + convergence): training successful does generalize.**
**Right (underfitted, both high avg error): training bad and generalization bad.**

**Experiment Approach and Assumptions**:

PyBrain software package was used to test both neural networks and support vector machines.
A gradient descent back propagation algorithm was used for neural networks.
An RBF kernel machine used for support vector machine (SVM).

**Neural Networks**
Only networks of 2x(hidden layers)x1 have been used since this is a binary problem.
To cut down on the simulations learning decay rate was kept constant at 0.999
Weight decay also kept as default at 0.
All neural networks using sigmoid error function by default.
A maximum of 10,000 epochs for any test was set.

Using Baum-Haussler's tule of thumb, a target range of hidden units was decided to be between 60-200.

$$N_{hidden} \leq \frac{N_{train} \cdot E_{tolerance}}{N_{points} + N_{out}}$$

Where
$N_{train}$ = epochs
$E_{tolerance}$ = [12, 8, 4] for [1000, 5000, 10000] epochs respectively
$N_{points}$ = 194
$N_{out}$ = 2

$$N_{hidden} \leq \frac{1000 \cdot 12}{194 + 2} \qquad N_{hidden} \leq \frac{5000 \cdot 8}{194 + 2} \qquad N_{hidden} \leq \frac{10000 \cdot 4}{194 + 2}$$

$N \approx 60$ $N \approx 200$ $N \approx 200$

Although others have sometimes used less units they used more advanced techniques, for example, Lang and Witbrook used shortcut connections between layers, Yu Et al used adaptive activation functions, others used fuzzy method and data partitioning etc.

For each network, about five tests were conducted:
Epoch levels (1000x2, 5000x2, 10000x1) x
Two Param Tests for each epoch level
1 - default learning rate 0.25 and default momentum 0.9
2- learning rate 0.01 and momentum 0.3

Activation of these networks was then plotted and the success of the test was calculated by the average classification results of the two spirals plotted when comparing result of activated inputs to the expected outputs as well as visually noting the average quality of classification compared to optimal.

Classification Success:

- Looks nothing like spirals (0-25% classification)
  (this includes when everything has been classified one color).
- Some spiral like shapes are evident but not strong (25-50% classification).
- Spiral shapes are evident and pretty good but not excellent quality (50-80%).
- Two spirals clearly evident and barely any misclassification (+80%).

Speed of the categorisation was visually estimated based on the level of the error function after the test was finished, for example, if a 1000 epoch test has an error of 11% at completion then this is 12/11 = 1.1 of its goal error tolerance and so it performed very quickly, whereas a 1000 epoch test that was at 20% scores only 12/20 = 0.6
This is for a constant error rate. If the error goal is achieved before the end of the test but then the error flattens out then the score is multiplied by the proportion of the test that it was achieved by. For example if the 12% level for the 1000 epoch test were achieved after only 300 epochs it would score 3.0 despite ending on 12%.
If the error function has already flattened at a score significantly above the goal then a 0 was given for speed.

Although generalization was planned to be estimated for each test based on the convergence/divergence of training versus validation errors as in figure 2 above, time constraints have resulted in a broad estimation of generalization of a general network architecture based upon the performance of it's simulations.

A couple of contrast tests using an arc-tan activation function instead of the standard sigmoid function were also conducted and contrasted.

**SVM (support vector machine)**

The Radial Basis Function kernel (RBF) is the SVM method used here as a more efficient alternative to this non-linear binary classification task.
It is one of the most common kernel machines used for SVM classification.

The C parameter defines how high the cost is of misclassification versus making the decision plane more complex. A low C will attempt to make the decisions very simple but often creates classification errors in the process.
Gamma is a generalization parameter. Low gamma corresponds to higher generalization (perhaps more error in the original dataset) whilst high gamma corresponds to low generalization (perhaps more correct in the original dataset).

**Results:**

When concluding on the generalization of an architecture, the average of classification performance was looked at as well as the average speed.
For each case, only the non-zero tests (where classification was judged to be zero) are averaged.
The accuracy of the generalization results are not highly reliable as not enough tests have been but nevertheless it is a rough estimate.

| Hidden Layers | Method | Epochs | Params | | Class. | Speed |
|---|---|---|---|---|---|---|
| | | | LR | Mom | | |
| (60) | BackProp | 1000 | 0.01 | 0.3 | 0.62 | 0 |
| | | | 0.01 | 0.8 | 0.74 | 1.0 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 5000 | 0.01 | 0.3 | 0.77 | 0.8 |
| | | | 0.01 | 0.8 | 0.76 | 0.8 |
| | | | 0.25 | 0.9 | 0 | 0 |
| (100) | | 10000 | 0.05 | 0.5 | 0 | 0 |
| Generalization | Low | | | Avg. | 0.72 | 0.65 |
| | | | | S.D | 0.07 | 0.44 |

| Hidden Layers | Method | Epochs | Params | | Class | Speed |
|---|---|---|---|---|---|---|
| | | | LR | Mom | | |
| (30,30) | BackProp | 1000 | 0.01 | 0.3 | 0.64 | 5.0 |
| | | | 0.01 | 0.8 | 0.67 | 1.0 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 5000 | 0.01 | 0.3 | 0.82 | 1.6 |
| | | | 0.01 | 0.8 | 0.70 | 0.9 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 10000 | 0.05 | 0.5 | 0 | 0 |
| Generalization | Low | | | Avg. | 0.71 | 2.1 |
| | | | | S.D | 0.08 | 1.94 |

| Hidden Layers | Method | Epochs | Params | | Class | Speed |
|---|---|---|---|---|---|---|
| | | | LR | Mom | | |
| (20,20,20) | BackProp | 1000 | 0.01 | 0.3 | 0.63 | 3.3 |
| | | | 0.01 | 0.8 | 0.66 | 3.2 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 5000 | 0.01 | 0.8 | 0.91 | 2.7 |
| | | | 0.01 | 0.3 | 0.93 | 8.0 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 10000 | 0.05 | 0.5 | 0.99 | 8.0 |
| Generalization | Good | | | Avg. | 0.78 | 4.3 |
| | | | | S.D | 0.16 | 2.48 |

| Hidden Layers | Method | Epochs | Params | | Class | Speed |
|---|---|---|---|---|---|---|
| | | | LR | Mom | | |
| (10,40,10) | BackProp | 1000 | 0.01 | 0.3 | 0.75 | 1.2 |
| | | | 0.01 | 0.8 | 0.55 | 2.0 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 5000 | 0.01 | 0.3 | 0.95 | 3.2 |
| | | | 0.01 | 0.8 | 0.73 | 1.15 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 10000 | 0.05 | 0.5 | 0.88 | 0.1 |
| Generalization | Moderate | | | Avg. | 0.75 | 1.9 |
| | | | | S.D | 0.16 | 0.96 |

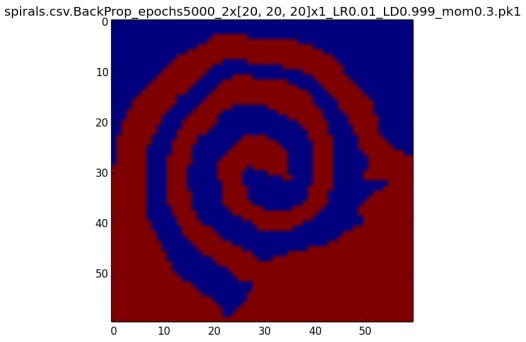
| Hidden Layers | Method | Epochs | Params | | Class | Speed |
|---|---|---|---|---|---|---|
| | | | LR | Mom | | |
| (10,20,20,10) | BackProp | 1000 | 0.01 | 0.3 | 0.55 | 2.5 |
| | | | 0.01 | 0.8 | 0.64 | 2.0 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 5000 | 0.01 | 0.3 | 0.90 | 1.7 |
| | | | 0.1 | 0.8 | 0.76 | 1.6 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 10000 | 0.05 | 0.5 | 0.74 | 0.1 |
| Generalize | Low | | | Avg. | 0.71 | 1.95 |
| | | | | S.D | 0.15 | 0.4 |

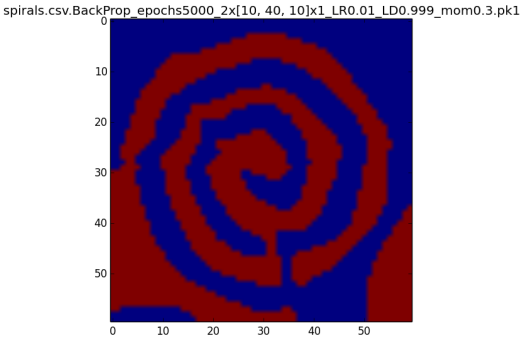
| Hidden Layers | Method | Epochs | Params | | Class | Speed |
|---|---|---|---|---|---|---|
| | | | LR | Mom | | |
| (12x12x 12x12x12) | BackProp | 1000 | 0.01 | 0.3 | 0.66 | 1.0 |
| | | | 0.01 | 0.8 | 0.75 | 0.8 |
| | | | 0.25 | 0.9 | 0 | 0 |
| | | 5000 | 0.01 | 0.3 | 0.95 | 2.0 |
| | | | 0.01 | 0.8 | 0.85 | 1.6 |
| | | | 0.25 | 0.9 | 0 | 0 |
| (10,10,10,10,10) | | 10000 | 0.05 | 0.5 | 0.73 | 0.1 |
| Generalization | Good | | | Avg. | 0.8 | 1.35 |
| | | | | S.D | 0.13 | 0.55 |

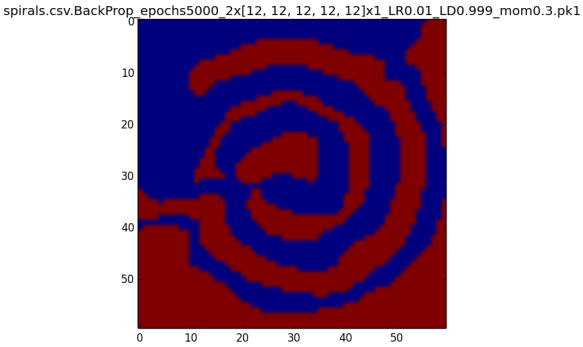
## Best Performers

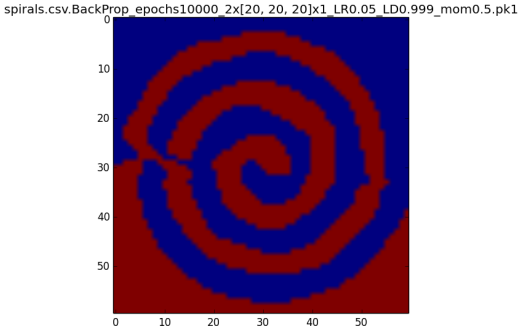
| Epochs | 60 | | 30,30 | | 20,20,20 | |
|---|---|---|---|---|---|---|
| 1000 | 0.62 | 0 | 0.64 | 5 | 0.63 | 3.3 |
| 1000 | 0.74 | 1 | 0.67 | 1 | 0.66 | 3.2 |
| 1000 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5000 | 0.77 | 0.8 | 0.82 | 1.6 | 0.91 | 2.7 |
| 5000 | 0.76 | 0.8 | 0.7 | 0.9 | 0.93 | 8 |
| 5000 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10000 | 0 | 0 | 0 | 0 | 0.99 | 8 |

| Epochs | 10,40,10 | | 10,20,20,10 | | 12,12,12,12,12 | |
|---|---|---|---|---|---|---|
| 1000 | 0.75 | 1.2 | 0.55 | 2.5 | 0.66 | 1 |
| 1000 | 0.55 | 2 | 0.64 | 2 | 0.75 | 0.8 |
| 1000 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5000 | 0.95 | 3.2 | 0.9 | 1.7 | 0.95 | 2 |
| 5000 | 0.73 | 1.15 | 0.76 | 1.6 | 0.85 | 1.6 |
| 5000 | 0 | 0 | 0 | 0 | 0 | 0 |
| 10000 | 0.88 | 0.1 | 0.74 | 0.1 | 0.73 | 0.1 |

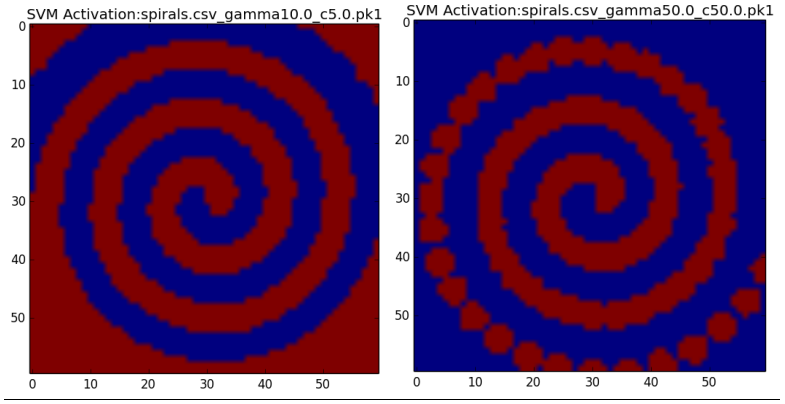


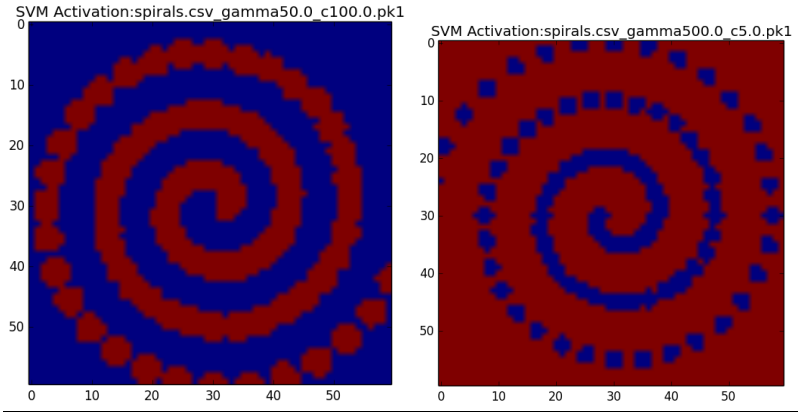



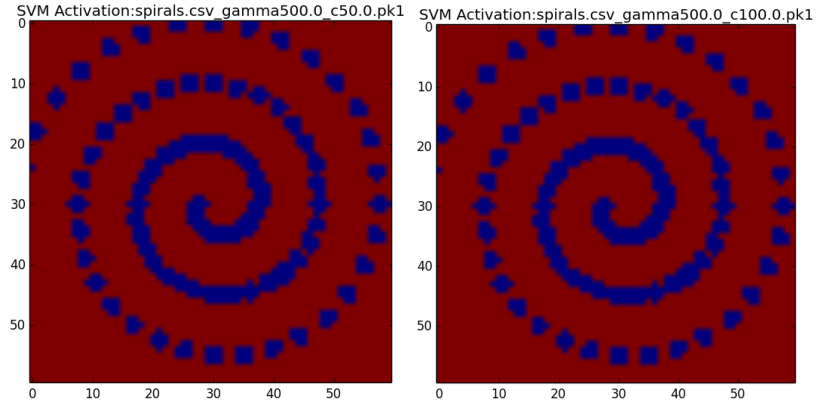



| SVM - RBF | | Classification | Speed | Generalization |
|---|---|---|---|---|
| c | γ | | | |
| 5 | 10 | 0.98 | 10 | High |
| 5 | 500 | 0.96 | 10 | Moderate/Low |
| 50 | 50 | 0.93 | 10 | Good |
| 50 | 500 | 0.95 | 10 | Moderate/Low |
| 100 | 50 | 0.96 | 10 | Good |
| 100 | 500 | 0.95 | 10 | Moderate/Low |



**Discussion**:

The ANN's were not very fast at solving this highly non-linear classification.
They required at least 5000 epochs to get good results.
However they did solve the problem with optimal parameters being:

- dense [20, 20, 20] network
- optimal learning rate of at least 0.01
- optimal momentum rate of at least 0.3

The SVM method was a lot faster in solving the two-spiral problem; in a matter of seconds the kernel machine produced results. The neural networks were slower and thus required more planning with parameter selection or mass iteration.
The (20,20,20) and (12,12,12,12,12) networks seemed to generalize the most, with either more hidden layers or more densely connected hidden layers suggesting better generalization.
Whilst it wasn't explored whether more hidden units would result in better performance, beyond an optimal number it is thought not to be the case.
When the learning rate was low enough, increasing the momentum above 0.5 seemed to improve classification but at a cost of speed.

**Follow up experiments:**

- More and less (than 60) hidden units
- Arctan, Gaussian, bi-radial (or other) activation function instead of standard sigmoid
- Weight decay rate and learning decay rate parameter adjustment
- Transform dataset into polar coordinates before training
- Evolino and RProp algorithms
- Variations on standard forward connectivity (eg. shortcuts)
- Batch training

b) (ANN vs. SVM): Generate your own variation of the 2-spiral task. Then solve the associated classification task using ANNs and discuss your approach and solution in comparison to a). [20% of Q2 mark]

The variation of the 2-spiral task that I looked at involved training on the same dataset that had noise and overlap introduced into the spirals.
To obtain this spiralipsoid galaxy like shape more data points were used (169 for each spiral). A single and then double noisy spiral is shown below:

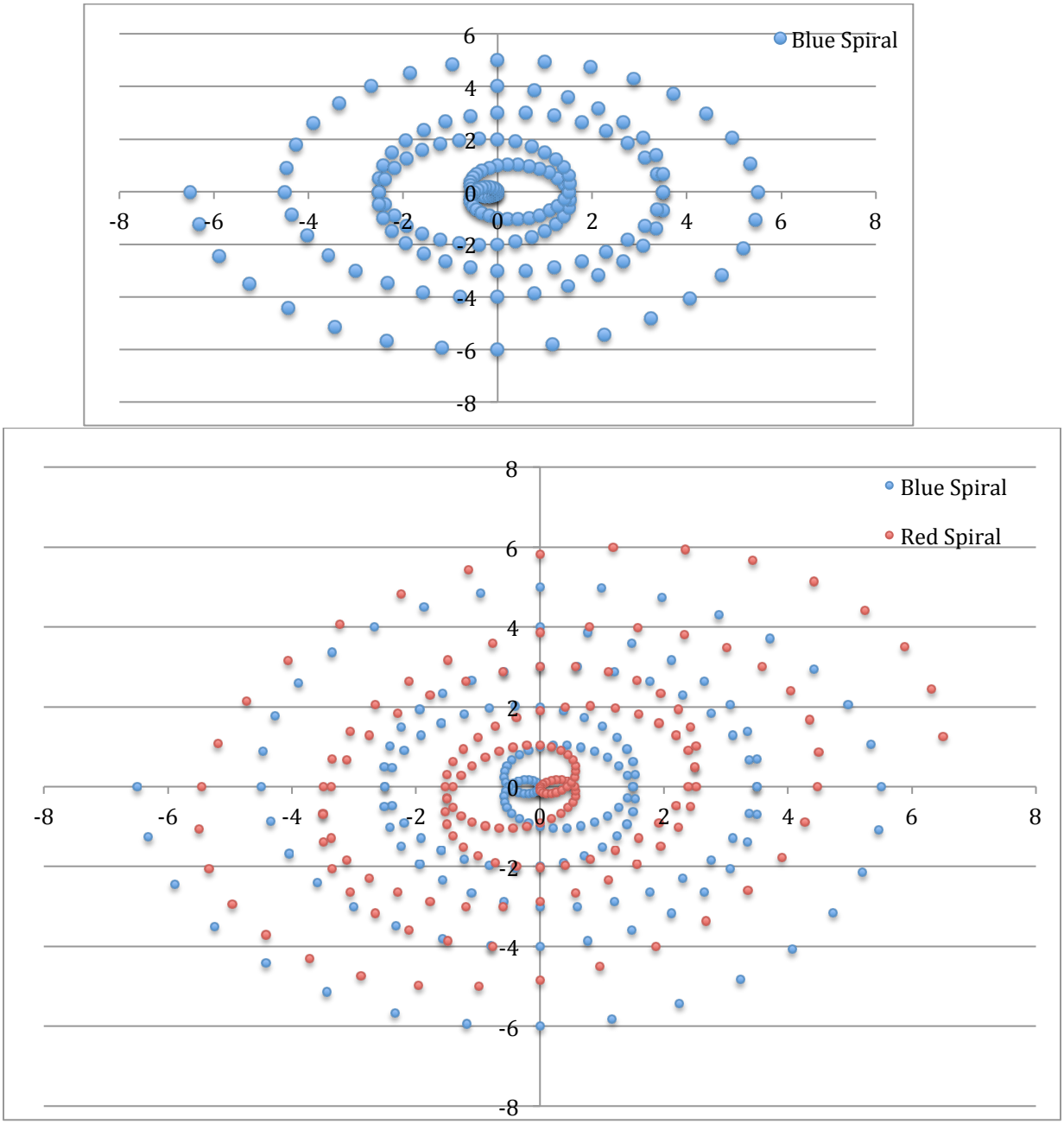



Figure 3: Noisy Two-Spiral Dataset

$$N_{hidden} \leq \frac{N_{train} \cdot E_{tolerance}}{N_{points} + N_{out}}$$

Where

$N_{train}$ = epochs

$E_{tolerance}$ = [12, 8, 4] for [1000, 5000, 10000] epochs respectively

$N_{points}$ = 338

$N_{out}$ = 2

$$N_{hidden} \leq \frac{1000 \cdot 12}{338 + 2}$$

$N \approx 35$

$$N_{hidden} \leq \frac{5000 \cdot 8}{338 + 2}$$

$N \approx 118$

$$N_{hidden} \leq \frac{10000 \cdot 4}{338 + 2}$$

$N \approx 118$

## Approach

### ANN

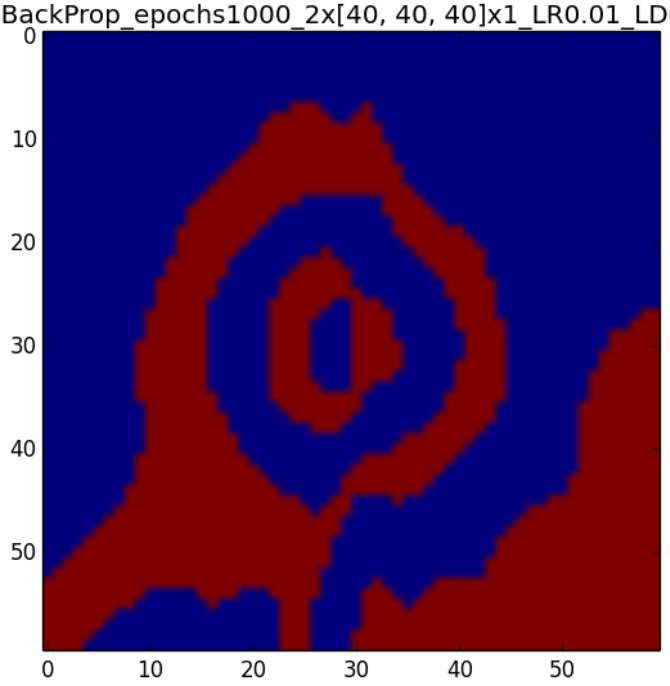
- A target number of maximum hidden units was chosen as 120 per above.
- Champion of two-spirals problem was considered early on.
- Error function was observed.

### SVM

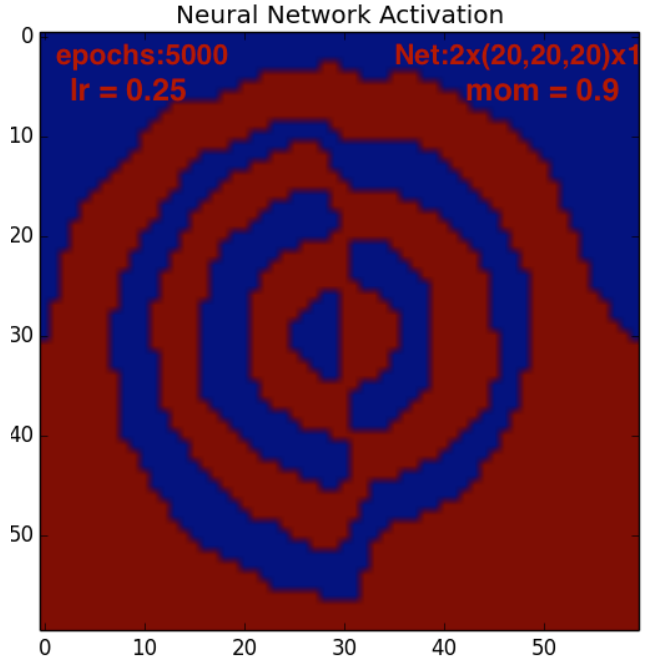
- A small number of tests were run with parameters c and gamma chosen to achieve maximum coverage.

# NoisySpiral ANN Results

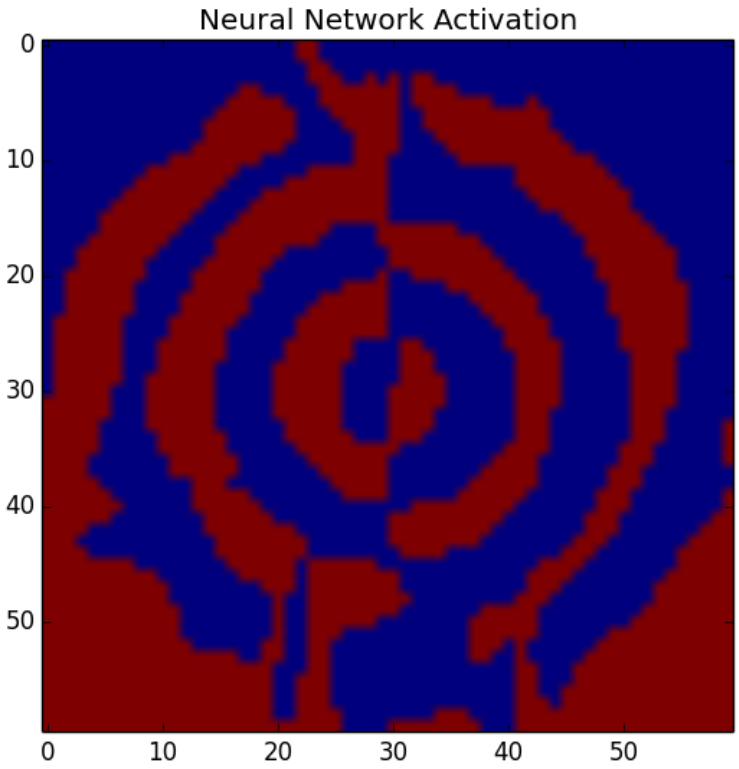
spirals.noisy.csv.BackProp_epochs1000_2x[40, 40, 40]x1_LR0.01_LD0.999_mom0.3.pk1

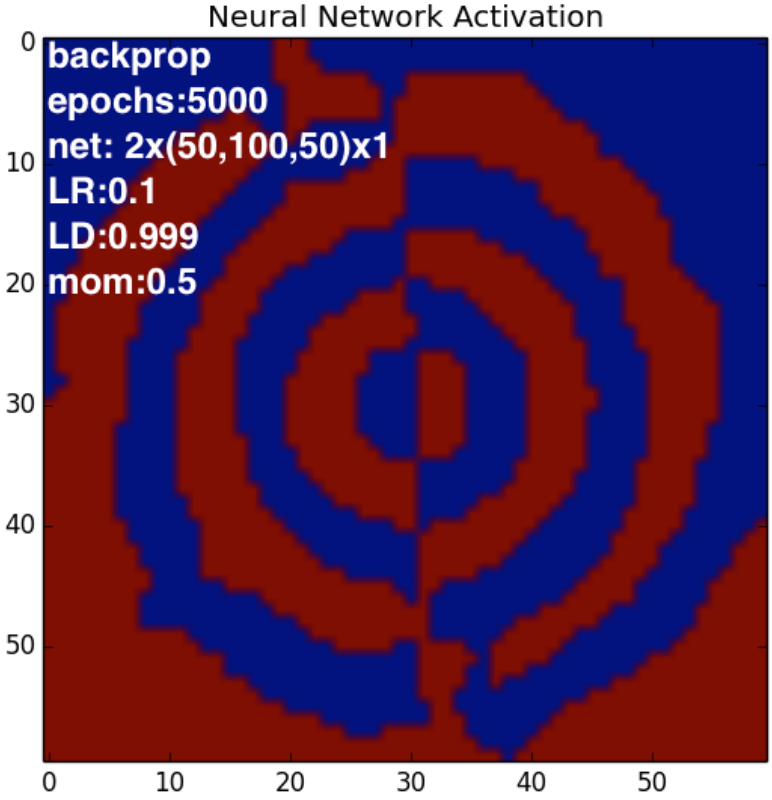


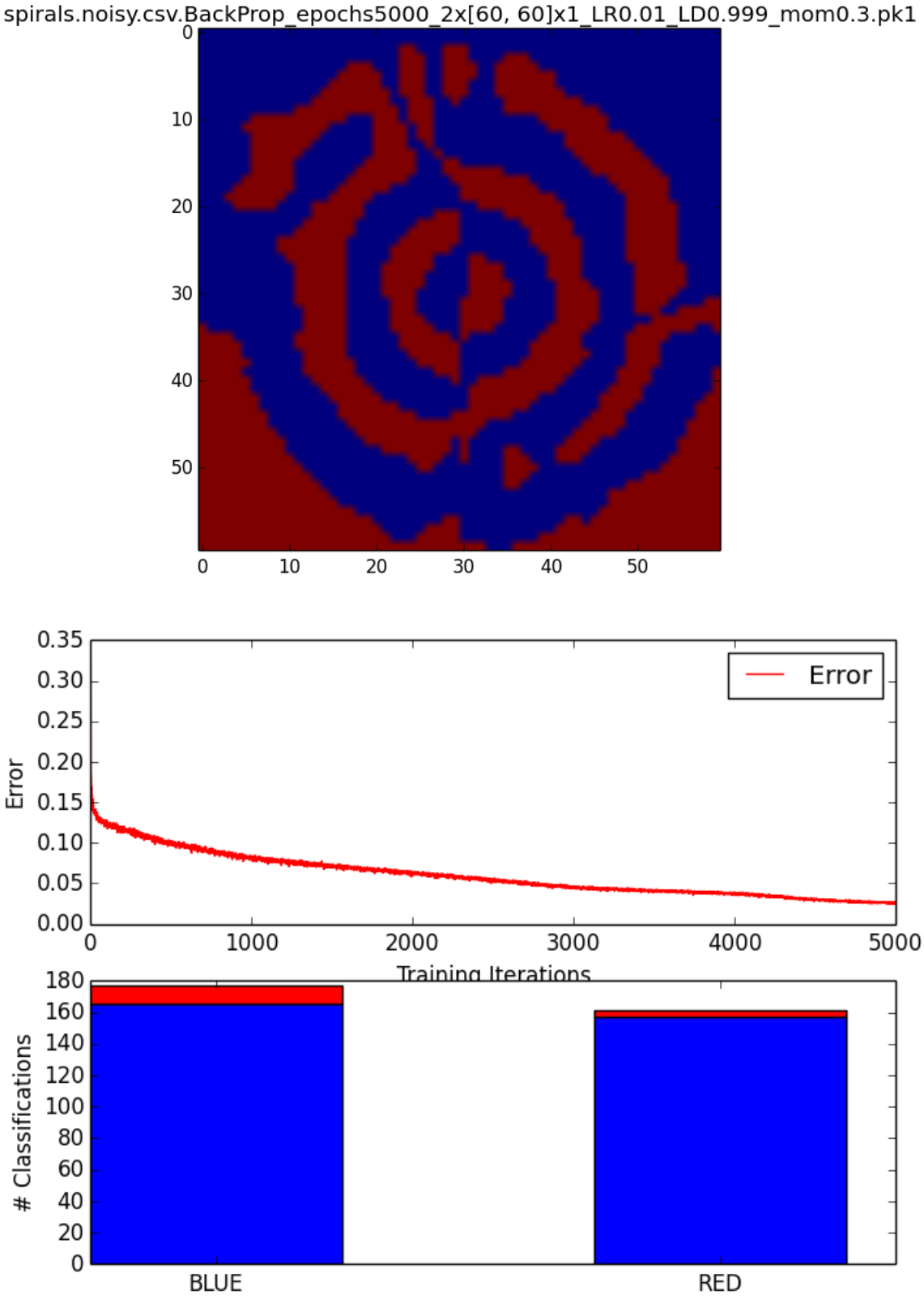

Neural Network Activation

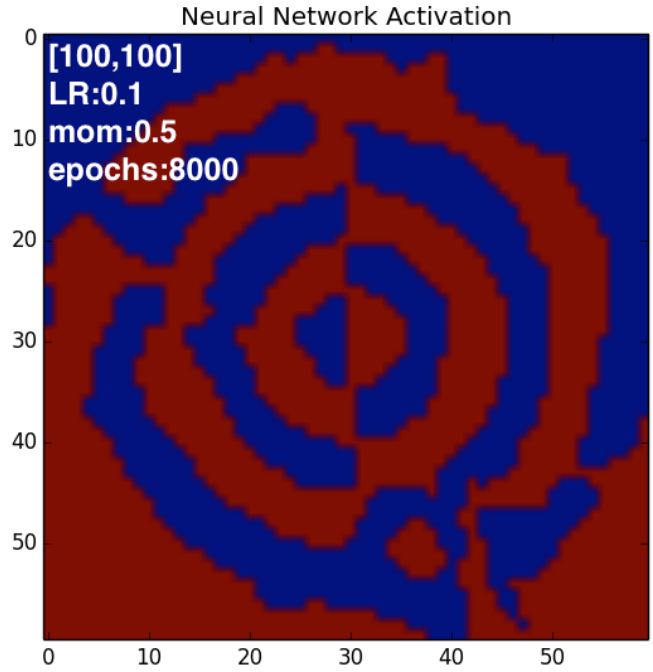


5spiral: [40,40,40] LR:0.01 mom:0.5 5000epochs

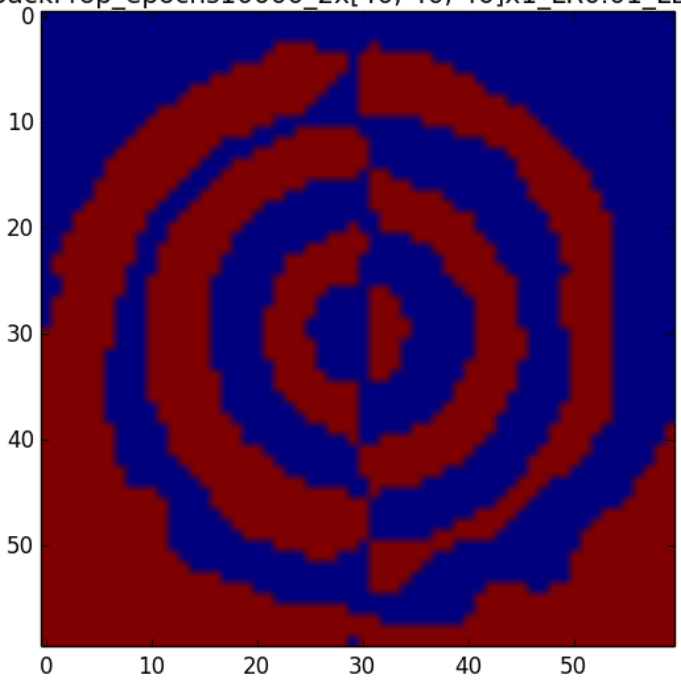


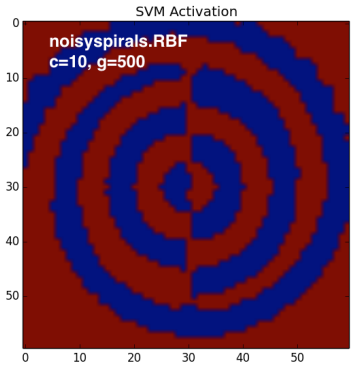

spirals.noisy.csv.BackProp_epochs5000_2x[60, 60]x1_LR0.01_LD0.999_mom0.3.pk1
0
10
20
30
40
50
0
10
20
30
40
50
Error
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
0
1000
2000
3000
4000
5000
Training Iterations
# Classifications
180
160
140
120
100
80
60
40
20
0
BLUE
RED

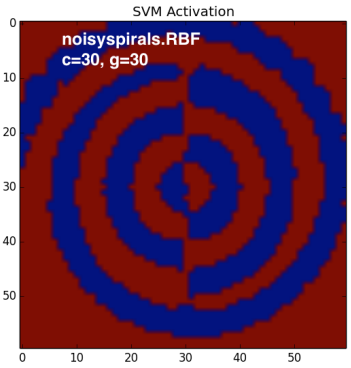


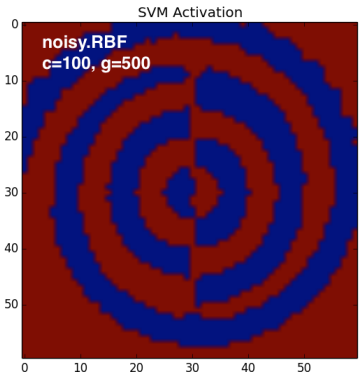

spirals.noisy.csv.BackProp_epochs10000_2x[40, 40, 40]x1_LR0.01_LD0.999_mom0.3.pk1

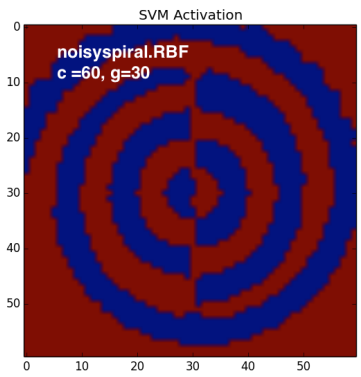


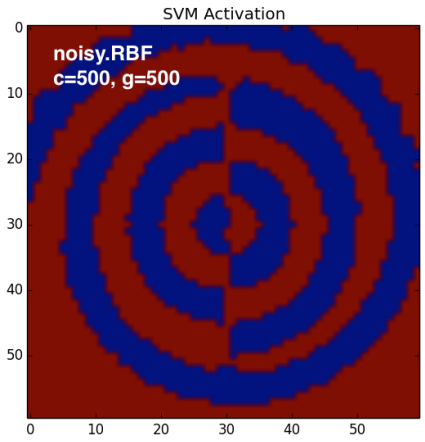
# NoisySpirals RBF Results

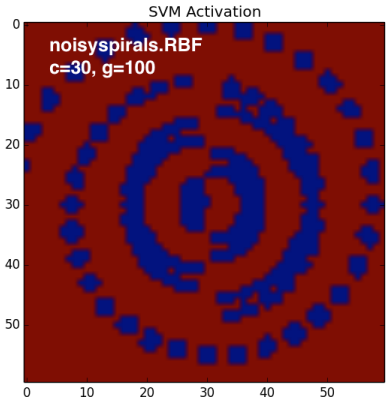


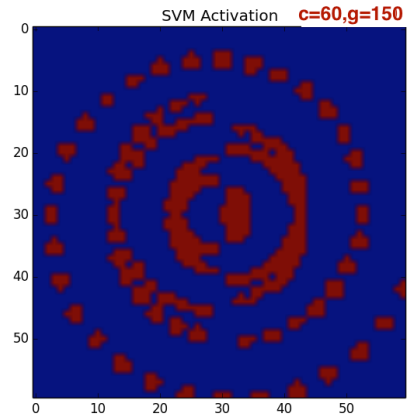



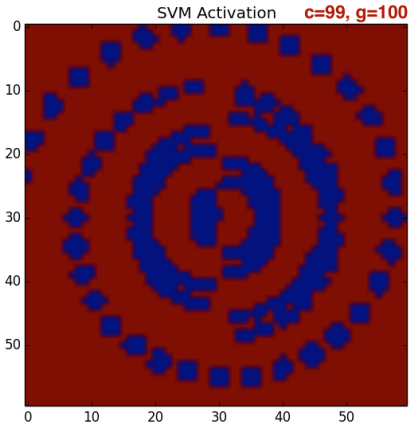



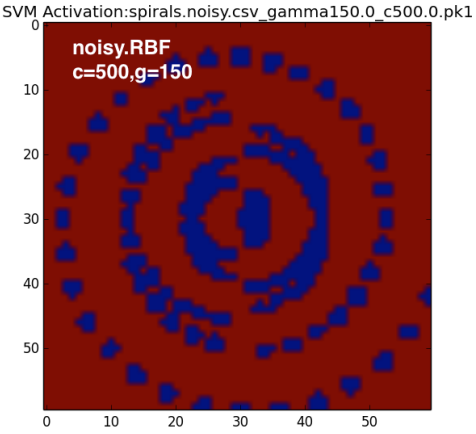



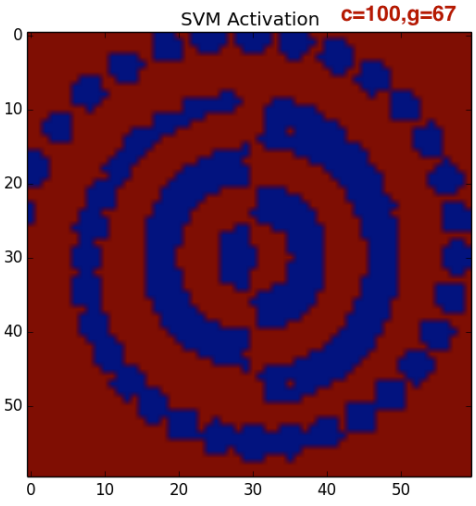




SVM Activation
noisyspirals.RBF
c=30, g=100


SVM Activation
c=60,g=150


SVM Activation
c=99, g=100


SVM Activation:spirals.noisy.csv_gamma150.0_c500.0.pk1
noisy.RBF
c=500,g=150


SVM Activation
c=100,g=67

- 
- 
- 

. c) (ANN vs. SVM): Compare ANNs and SVMs on solving the two classification tasks in (a) and (b). [30% of Q2 mark]

. say something about densely vs not densely connected layers

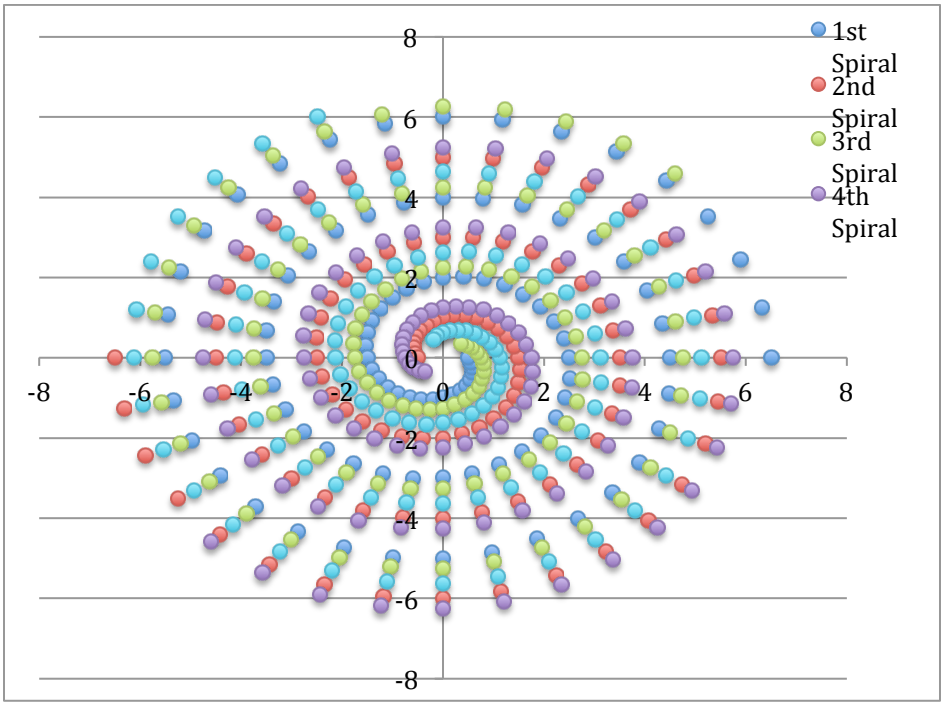
The ANNS in the original two-spirals problem were slow to solve the problem but did eventually solve it, with denser networks showing better generalization and speed, as well as in most cases accuracy except for a [10,40,40,10] network achieving 95% accuracy at 5000 epochs, greater than [20,20,20] (93%) which is denser. However, relatively speaking this network is still quite dense and in fact, the [20,20,20] network was the overall champion in achieving the most accurate and fast result (99%, 800%, 10,000epochs).

Comparatively, the SVM solved the problem very fast, with simple decisions (c=5), and produced the most accurate classification with a low generalization parameter suggesting high generalization (gamma=10).
Although it wasn't looked at in this study, even the quickest ANN solution would fail to compete for speed against the SVM on this problem.

In the interest of brevity, performance metrics were deferred for analysis of the Noisy Spiral problem in favor of visual results, but statistics would show (and indeed it was evident as well inspecting error functions) that the average classification performance versus speed performance was much slower for this task than the original 2spirals problem. That is, to get similar classification accuracy (with analogous networks), more time would be needed training with more highly optimized learning and momentum rates as well as other parameters.

Whilst the original two-spirals problem was a problem of non-linear separability, the noisy-spiral problem with overlap required a solution using higher dimensionality (SVM) to notice overlapping features. This was evident in the overlapping iris parts of the noisy spirals, which the ANN's failed to classify with precision but the SVM could do quite well with easy parameter selection. This was not merely a matter of the ANNs needing lower learning rates or momentum, since optimizing for the inner features would compromise the main outer spiral features common to the original two-spiral problem. The amount of neurons was already optimal, and error rates were leveling off, so there is no reason to believe ANN's could achieve this accuracy with more neurons or training.
The more densely connected networks again performed much better than sparsely connected ones.
The SVM solution showed a sweet zone of parameters distinguishing the detailed features whilst outside of this zone the main spiral features were classified.
This bipartite solution can be useful for certain problems.

. d) (5-spiral task): Generate a 5-spiral data set. Show and discuss how well your task can be solved with a suitable ANN or SVM. [40% of Q2 mark]




$$N_{hidden} \leq \frac{N_{train} \cdot E_{tolerance}}{N_{points} + N_{out}}$$

Where

$N_{train}$ = epochs

$E_{tolerance}$ = [12, 8, 4] for [1000, 5000, 10000] epochs respectively

$N_{points}$ = 485

$N_{out}$ = 2

$$N_{hidden} \leq \frac{1000 \cdot 12}{485 + 2}$$

$N \approx 25$

$$N_{hidden} \leq \frac{5000 \cdot 8}{485 + 2}$$

$N \approx 82$

$$N_{hidden} \leq \frac{10000 \cdot 4}{485 + 2}$$

$N \approx 82$

**Approach**

**ANN**

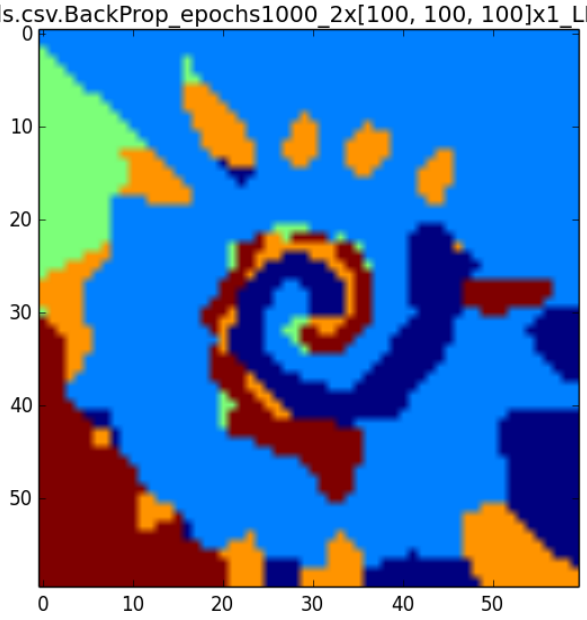
- A target number of maximum hidden units was chosen as 80 per above.
- Champion of two-spirals problem was considered early on.
- Error function was observed and used to make decisions.
- Activation results scaled to produce correct classifications.

**SVM**

- A small number of tests were run with parameters c and gamma chosen to achieve maximum coverage.
- Activation results scaled to produce correct classifications.

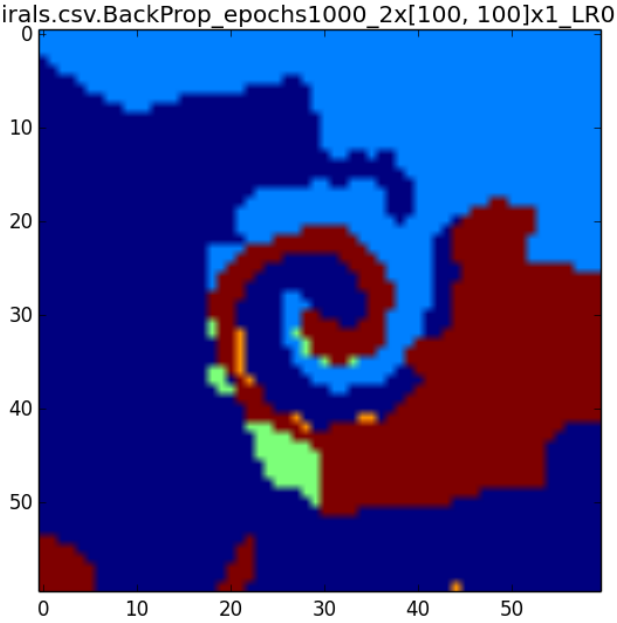
# 5Spiral ANN Results

test.results.5Spirals/5spirals.csv.BackProp_epochs1000_2x[100, 100, 100]x1_LR0.001_LD0.999_mom0.3.pk1



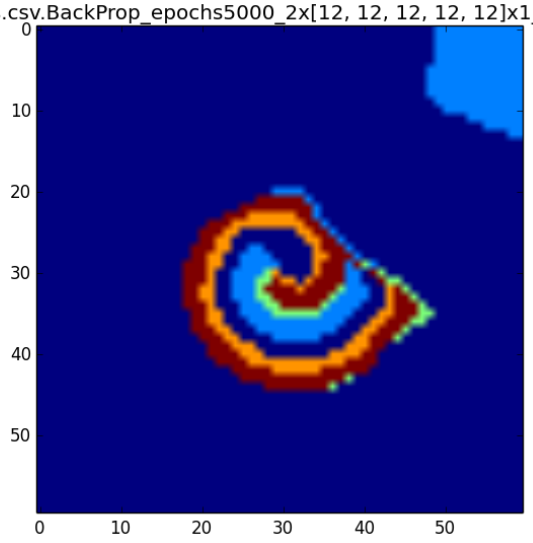

---

test.results.5Spirals/5spirals.csv.BackProp_epochs1000_2x[100, 100]x1_LR0.01_LD0.999_mom0.3.pk1



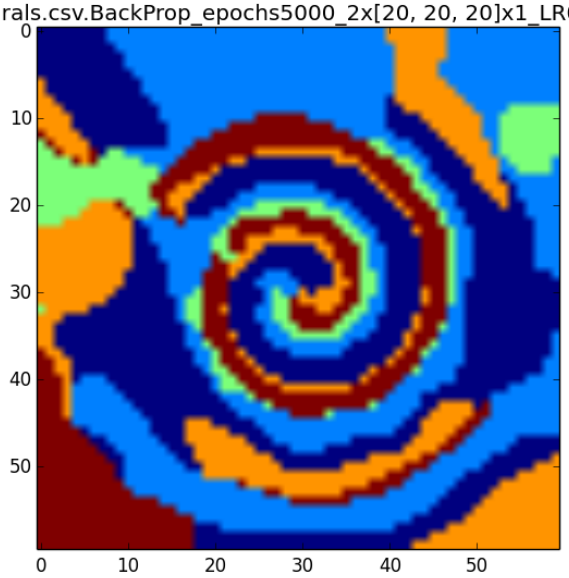

---

test.results.5Spirals/5spirals.csv.BackProp_epochs5000_2x[12, 12, 12, 12, 12]x1_LR0.01_LD0.999_mom0.3.pk1



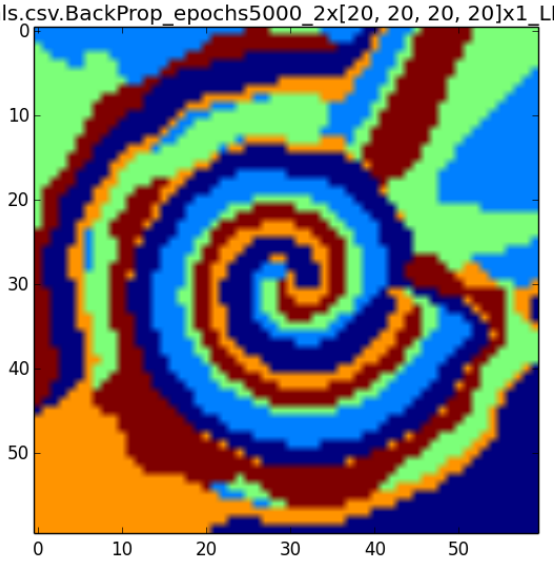

---

test.results.5Spirals/5spirals.csv.BackProp_epochs5000_2x[20, 20, 20]x1_LR0.01_LD0.999_mom0.3.pk1



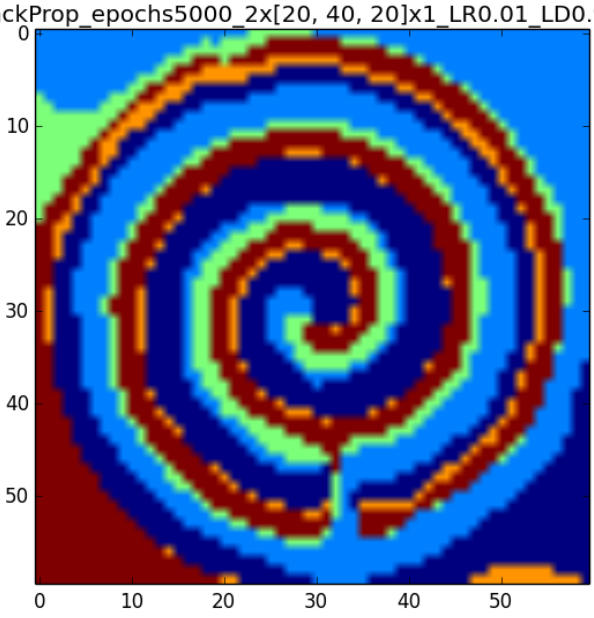

---

test.results.5Spirals/5spirals.csv.BackProp_epochs5000_2x[20, 20, 20, 20]x1_LR0.01_LD0.999_mom0.3.pk1



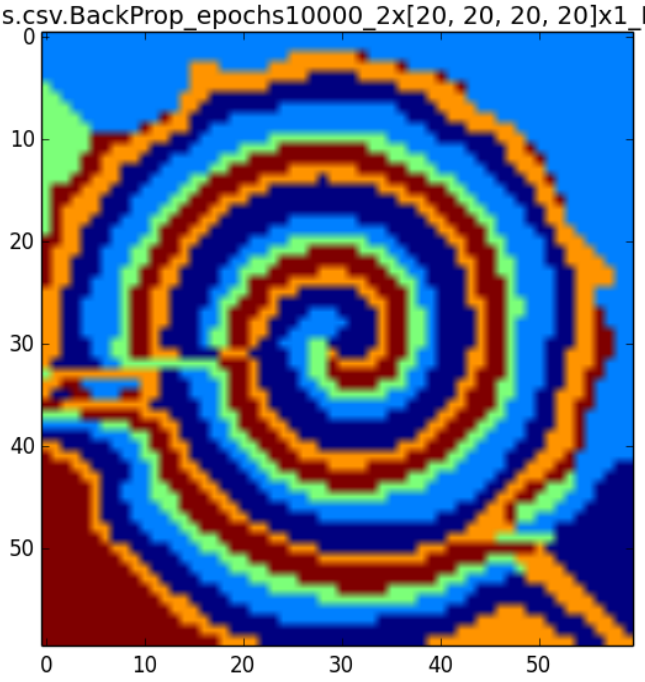

---

5spirals.csv.BackProp_epochs5000_2x[20, 40, 20]x1_LR0.01_LD0.999_mom0.3.pk1

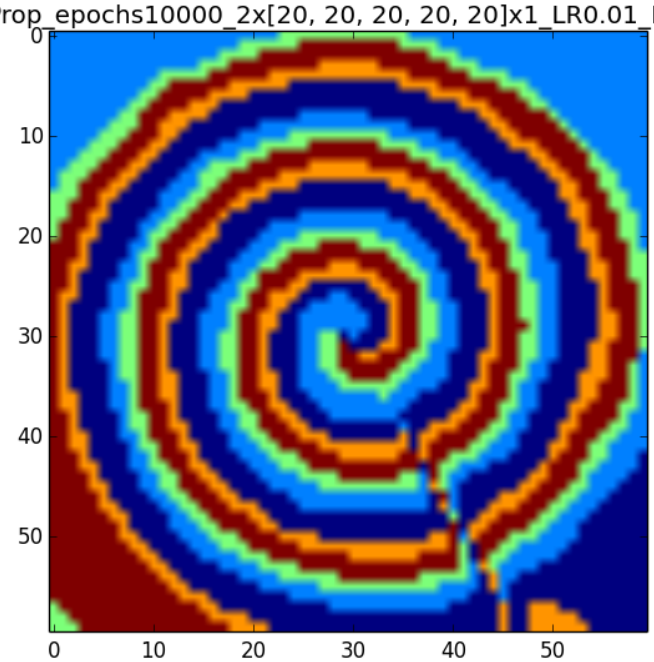


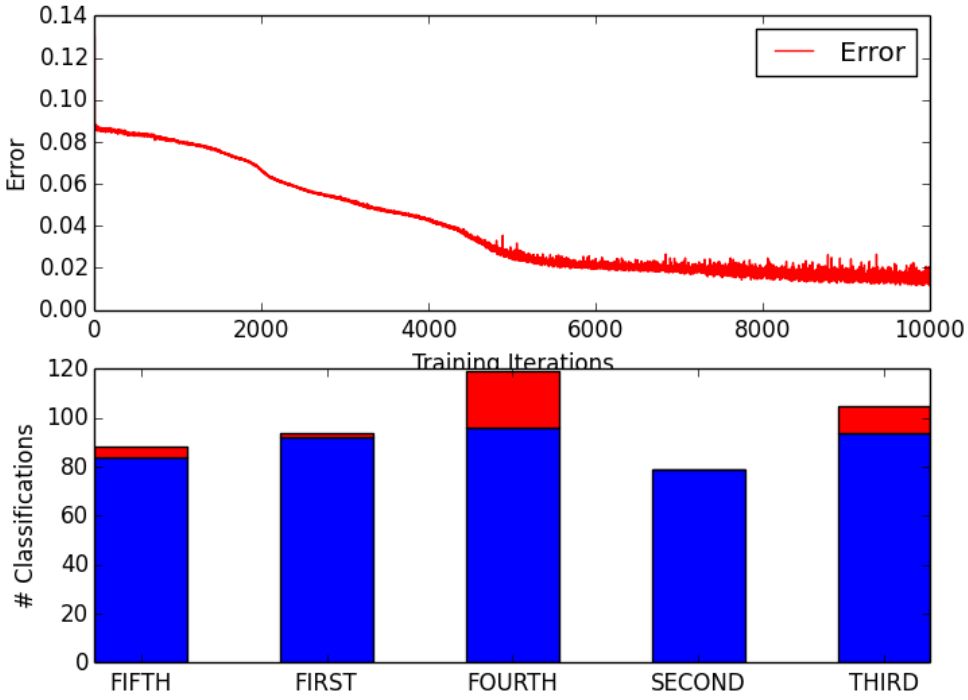

---

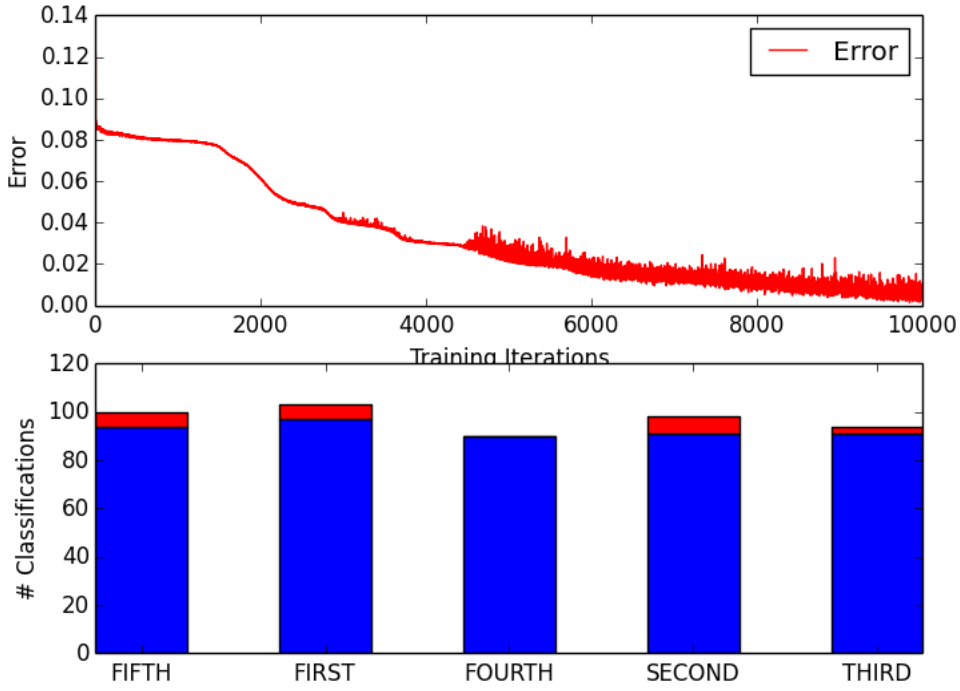
test.results.5Spirals/5spirals.csv.BackProp_epochs10000_2x[20, 20, 20, 20]x1_LR0.01_LD0.999_mom0.3.pk1




5spirals.csv.BackProp_epochs10000_2x[20, 20, 20, 20, 20]x1_LR0.01_LD0.999_mom0.3.pk1

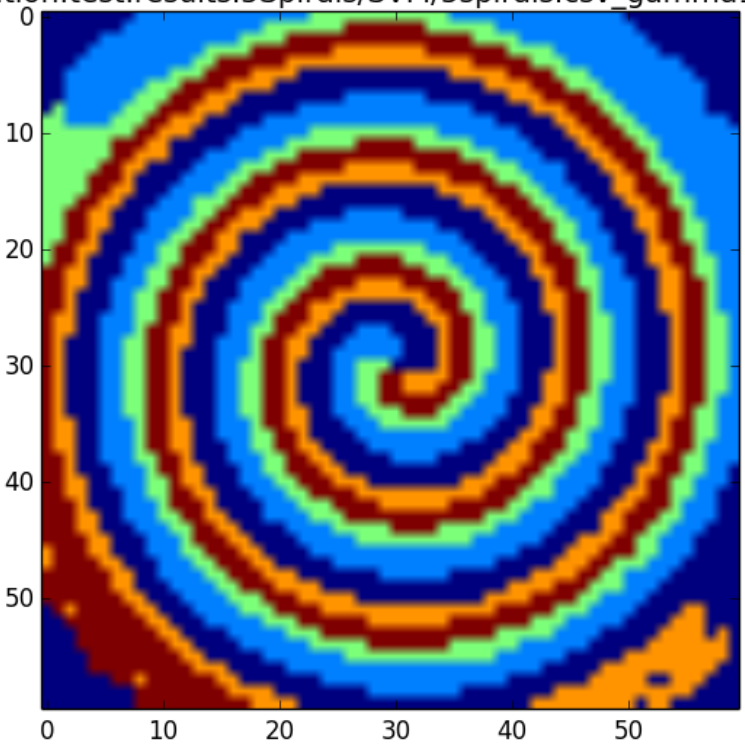



Error
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
2000
4000
6000
8000
10000
Training Iterations
# Classifications
120
100
80
60
40
20
0
FIFTH
FIRST
FOURTH
SECOND
THIRD

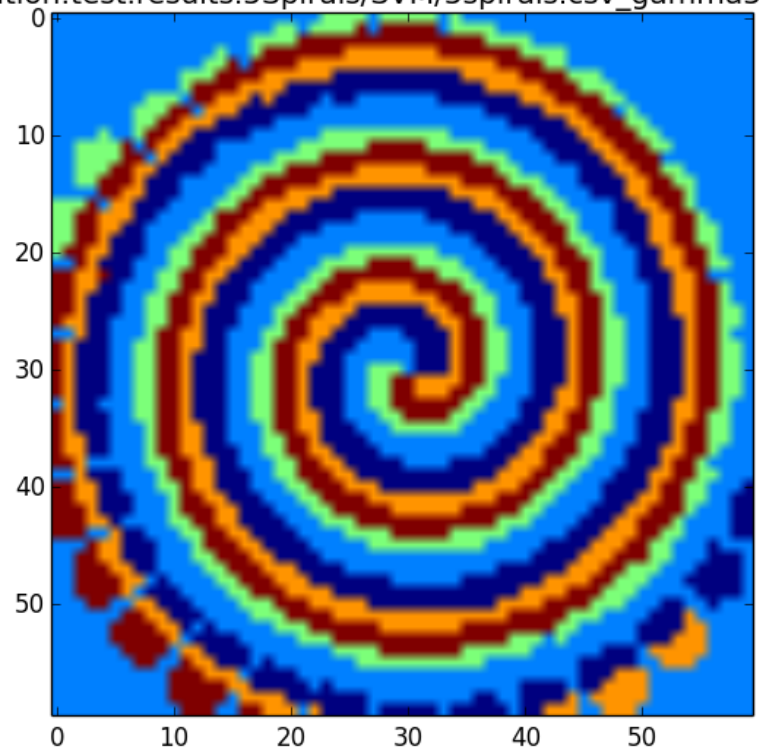

Error
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
0
2000
4000
6000
8000
10000
Training Iterations
# Classifications
120
100
80
60
40
20
0
FIFTH
FIRST
FOURTH
SECOND
THIRD

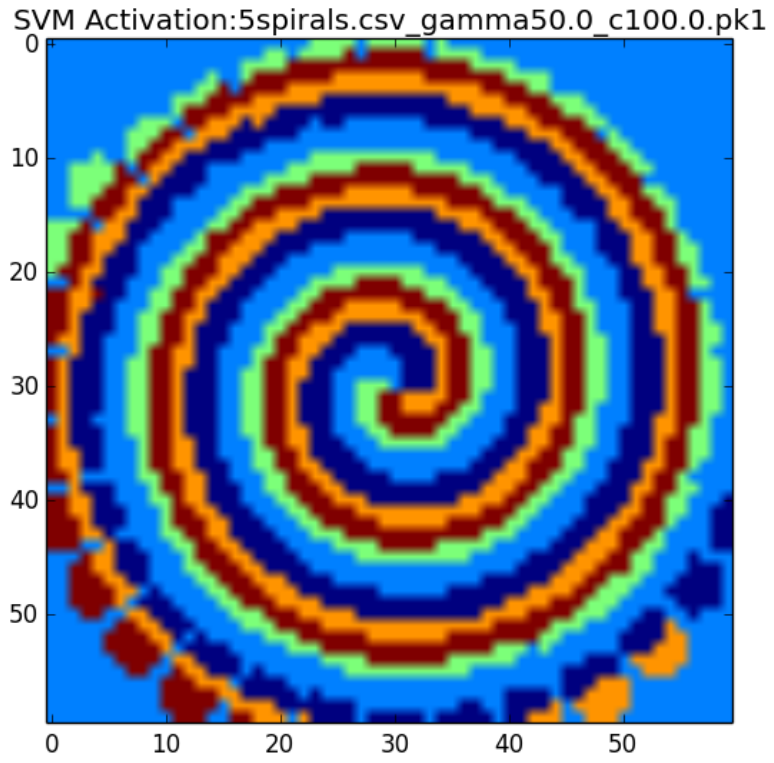
# 5Spiral SVM Results

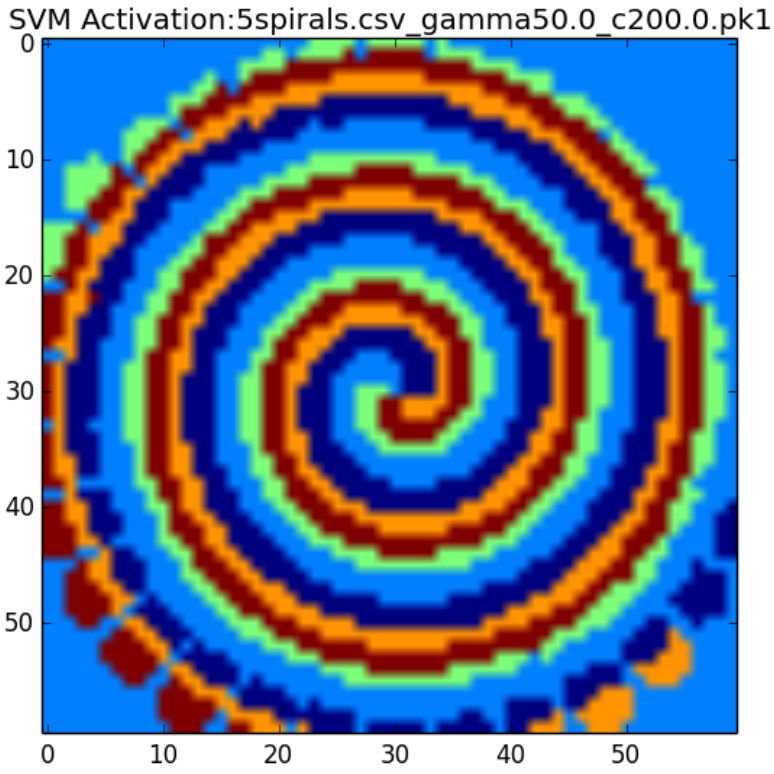
SVM Activation:test.results.5Spirals/SVM/5spirals.csv_gamma10.0_c5.0.pk1




SVM Activation:test.results.5Spirals/SVM/5spirals.csv_gamma50.0_c10.0.pk1




SVM Activation:5spirals.csv_gamma50.0_c100.0.pk1
0
10
20
30
40
50
0
10
20
30
40
50


SVM Activation:5spirals.csv_gamma50.0_c200.0.pk1
0
10
20
30
40
50
0
10
20
30
40
50

**Discussion of 5-Spirals**

Both the ANN and SVM methods were able to solve the 5spiral classification problem so long as the activation results were scaled into 5 classification regions.
The SVM was again faster but both methods showed that the methods to produce good classification for the 2-spiral problem generalized quite well for the 5-spiral problem.
The 5-spiral problem did benefit from a 5th layer of hidden units to get its best result.

# Q3 Autoencoder [3 marks]

Generate a dataset that uses sparse coding to encode the 26 letters of the alphabet:

A = (1,0,0,...,0) B = (0,1,0,...,0) ...Z = (0,0,0,..., 1)

Train a 26-H-26 multilayer perceptron (i.e. 26 input units, a hidden layer of H units and an output layer of 26 units) on the identity function that maps 0 to 0, 1 to 1, ..., 26 to 26.

Part I: Determine experimentally what is the minimal number of hidden units, H, required for training the network successfully (Hint: Check chapter 4 of the book (Mitchell, 1997)). Try di↵erent training algorithms.

$$N_{hidden} \leq \frac{N_{train} \cdot E_{tolerance}}{N_{points} + N_{out}}$$

$$N_{hidden} \leq \frac{50 \cdot 10}{8 + 8} \quad \textbf{= 31}$$

$$N \approx 25$$

Suggestion from [Machine Learning Chapter 4, Mitchell, 1997] was to use [8, 3, 8] for 90% accuracy or [8, 30, 8] for 92%.

**Approach**

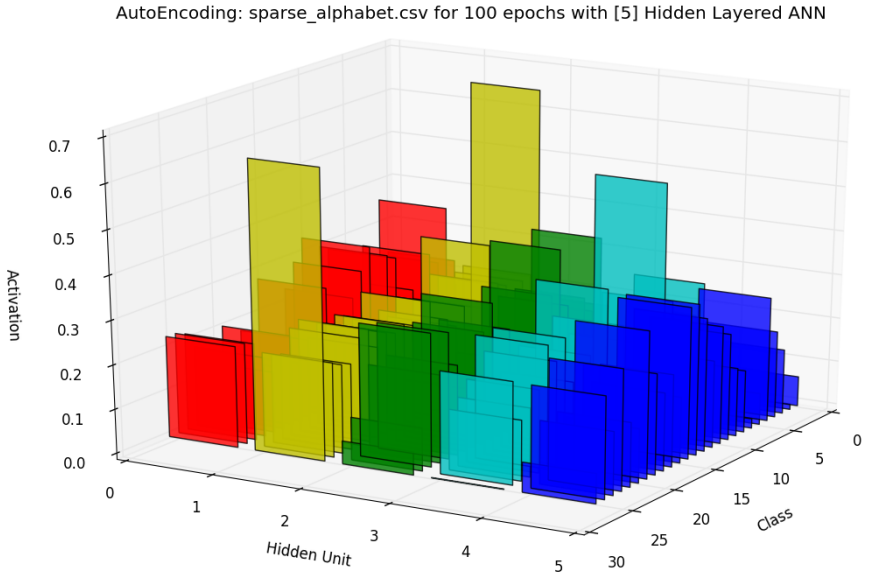
- Input and output layers chosen to be the same to reflect the number of classes required, that is, the number of characters in the English alphabet (26).
- Maximum hidden units selected to enable a possible encoding of the English language. (n = 5, since $2^5 = 32$)
- Look at other encodings that may have features of the English alphabet.

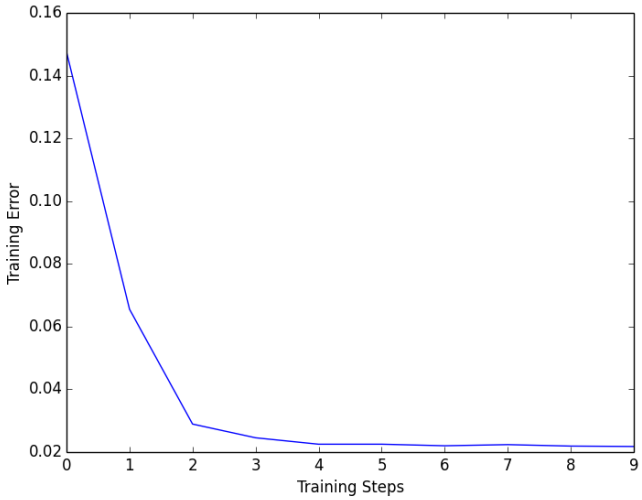
# Autoencoder Results

A number of trainings were conducted using an autoencoder implemented with BackProp (using a soft max layer) in PyBrain to investigate possible features that demonstrates unsupervised learning.

## Approach

- A 26 x [Hidden Units] x 26 network was trained using BackProp using a soft max layer which implemented an autoencoding.
- Hidden Layers with (3, 4, 5, 13, 52) were used.
- The activation results were inspected for possible feature encodings.



**Results**

Consider the activation results for the mapping of 26 input sparse encodings to 5 hidden units. We can see that the inputs more or less distinctly map into binary (5digits), but with some errors (2%).
It's likely that the last two letters encoding as the same binary number, here, is due to error.

**threshold=0.15?**

```
A(1):[ 0.19929564  0.16254736  0.26041631  0.07036624  0.30737445]    00001 1
B(2):[ 0.14413845  0.1800519   0.2052831   0.31118724  0.15933931]    00010 2
C(3):[ 0.37995648  0.24506104  0.12477057  0.06413216  0.18607976]    11000 24
D(4):[ 0.22137066  0.23853191  0.15231818  0.15989996  0.22787928]    01000 8
E(5):[ 0.011206e-02   7.47e-01   1.6729e-03   1.0000e-08  2.39e-01]   01001 9
F(6):[ 0.17646535  0.24648039  0.19701277  0.31456445  0.06547704]    01010 10
G(7):[ 0.44015443  0.01846106  0.0871924   0.38292027  0.07127184]    10110 22
H(8):[ 0.3759867   0.12410286  0.08516705  0.26750624  0.14723714]    10000 16
I(9):[ 0.31207191  0.21376767  0.10263843  0.22054297  0.15097902]    10010 18
J(10):[ 4.1836e-04   8.2966e-01   1.5402e-01  1.5225e-02 6.6406e-04]  01100 12
K(11):[ 0.62159205  0.07453808  0.00154656  0.08436249  0.21796081]   10011 19
L(12):[ 0.33358665  0.28160078  0.1247224   0.22479072  0.03529945]   11000 24
M(13):[ 0.00189624  0.39325648  0.17301529  0.41768805  0.01414393]   01110 13
N(14):[ 0.16014358  0.36919668  0.0663816   0.30729828  0.09697986]   11010 26
O(15):[ 0.05039107  0.55012683  0.14492008  0.07339083  0.18117119]   01111 15
P(16):[ 0.1667309   0.18275972  0.24911289  0.248421    0.15297549]   00100 4
Q(17):[ 0.12057101  0.37858374  0.1348192   0.16979631  0.19622973]   01001 9
R(18):[ 0.40793956  0.27327593  0.19840433  0.03135238  0.08902781]   11000 24
S(19):[ 3.7120e-05   1.2620e-04   3.4177e-02  9.5600e-01 9.6525e-03]  00111 7
T(20):[ 0.03603696  0.00176841  0.13052833  0.75697025  0.07469606]   00110 6
U(21):[ 5.6555e-01   2.2148e-01   4.1257e-02   1.2936e-04 1.7157e-01] 11001 25
V(22):[ 0.5598831   0.02456325  0.10040516  0.00780739  0.3073411 ]   10101 21
W(23):[ 0.31742092  0.04239528  0.19615824  0.20232625  0.24169931]   10111 23
X(24):[ 0.02742627  0.28452999  0.11198224  0.57299101  0.00307049]   01010 10
Y(25):[ 0.60521364  0.00235271  0.18799745  0.03108881  0.17334739]   10101 21
Z(26):[ 0.19929564  0.16254736  0.26041631  0.07036624  0.30737445]   10101 21
```

If we assume the codes labeled in decimal blue, have one erroneous bit, then we can see that there are 5 erroneous bits per (21x5 = 105) total bits, which is approximately the 2% error rate.

No feature encodings were immediately evident in the other tests.

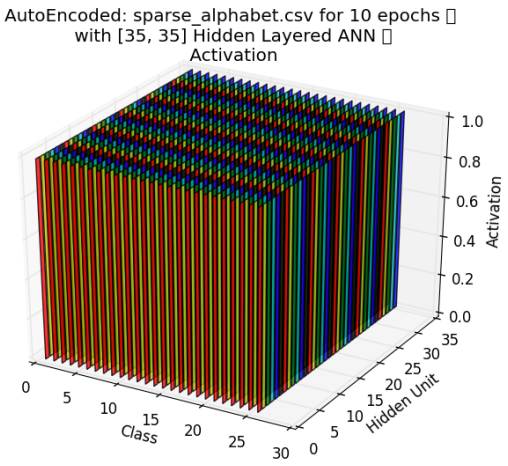
Autoencoding is an example of unsupervised learning to extract features that a superviser may not have thought of or have the knowledge or capacity to train for.

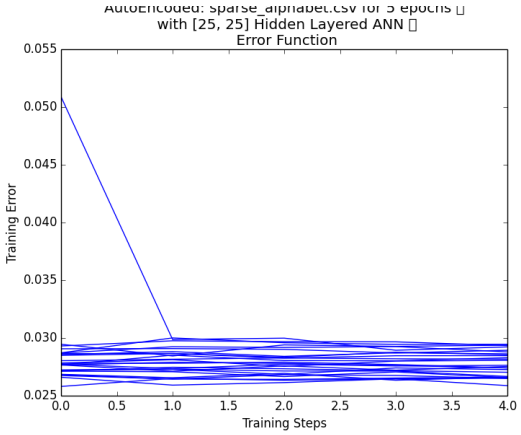
Part II: Conduct training experiments using 26-H1-H2-26 ANNs with two equally sized hidden layers. Determine experimentally what is the minimal number of hidden units in H1 and H2 required for training the network successfully. How does it compare to the above experiments with your 26-H-26 multilayer perceptron?

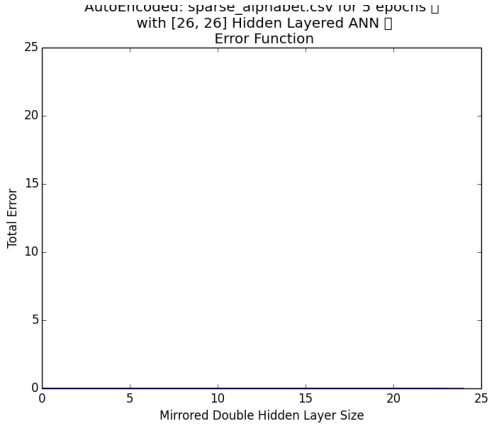
In your report describe what you did in the experiments and what was the outcome. Finally discuss what role the hidden layers plays in this experiment and what role this type of network could possibly play in real applications.

I first thought that the network would need at least as many neurons as in the one layer case, and so a successful training would be done with H1=H2 = 3 to again achieve at least 5 classes for encoding into 5bits. However results did not seem to reflect this.

The error function is always zero with these mirroring networks suggesting that the identity function has been achieved and the network has been trained successfully.



**Role of hidden layers in Autoencoders**

"The first layer of a stacked autoencoder tends to learn first-order features in the raw input (such as edges in an image). The second layer of a stacked autoencoder tends to learn second-order features corresponding to patterns in the appearance of first-order features (e.g., in terms of what edges tend to occur together--for example, to form contour or corner detectors). Higher layers of the stacked autoencoder tend to learn even higher-order features." [source: http://ufldl.stanford.edu/wiki/index.php/Stacked_Autoencoders] In our test because H1 = H2 , H2 doesn't extract anything more from H1.

**Applications**

- Compression (i.e. sparser representation) for image, video, audio, codes.
- Feature extraction without the need for a supervisor to define the feature (i.e. on unlabeled data).