Taking Initiative

Posted on April 23, 2008September 21, 2016 | Artificial Intelligence, Neural Networks

Basic Neural Network Tutorial : C++ Implementation and Source Code

i
52 Votes

So I've now finished the first version of my second neural network tutorial covering the implementation and training of a neural network. I noticed mistakes and better ways of phrasing things in the first tutorial (https://takinginitiative.wordpress.com/2008/04/03/basic-neural-network-tutorial-theory/)(thanks for the comments guys) and rewrote large sections. This will probably occur with this tutorial in the coming week so please bear with me. I'm pretty overloaded with work and assignments so I haven't been able to dedicate as much time as I would have liked to this tutorial, even so I feel its rather complete and any gaps will be filled in by my source code.

Introduction & Implementation

Okay so how do we implement our neural network? I'm not going to cover every aspect in great detail since you can just look at my source code. I'm just going to go over the very basic architecture. So what critical data storage do we need for our neural network?

- Our neuron values
- Our weights
- Our weight changes
- Our error gradients

Now I've seen various implementations and wait for it… here comes an OO rant: I don't understand why people feel the need to encapsulate everything in classes. The most common implementation of a neural network I've come across is the one where each neuron is modeled as an object and the neural network stores all these objects. Now explain to me that advantages of such an approach? All you're doing is wasting memory and code space. There is no sane reason to do this apart from trying to be super OO.

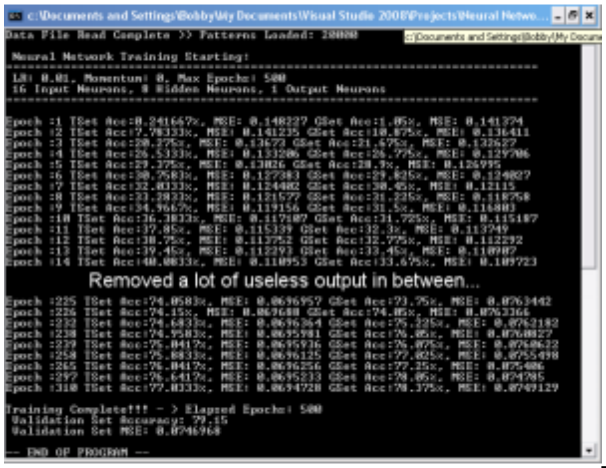
The other common approach is to model each layer as an object? Again what the hell are people smoking? I guess i can blame this on idiot professors that know the theory but can't code their way out of a paper bag. Like for example my father is a math's professor and is absolutely brilliant in regards to developing algorithms and just seeing elegant solutions to problems but he cannot write code at all. I've had to convert his original discrete pulse transform Matlab code into c++, and in the process of converting/optimizing it, I ended up developing a whole new approach and technique that wasnt possible to achieve directly from the maths – the so called fast discrete pulse transform.

I also tend to be a bit of a perfectionist and am firm believer in occam's razor (well a modified version) – "the simplest solution is usually the best solution". Okay enough ranting – the point is dont use classes when you dont need to! Useless encapsulation is a terrible terrible mistake, one that most college students make since its drilled into them during their years of study!

So below is how I structured my neural network and afaik it's as efficient as possible. If anyone can further optimize my implementation please do! I'd love to see your techniques (I love learning how to make code faster).

A neural network is a very simple thing and so must be implemented very simply. All the above values are just numbers so why can't I just use multi-dimensional arrays. The short answer is that I can and I do. Storing each of those sets of number in multi-dimensional arrays is the most efficient and simplest way of doing it with the added benefit that if you use the index variables i/j/k for input/hidden/output loops respectively, then your code will almost exactly resemble the formulas in the theory tutorial and in this one.

So now that the major architecture design decision has been mentioned, here's what the end result looks like:


c:\Documents and Settings\Bobby\My Documents\Visual Studio 2008\Projects\Neural Netwo...
c:\Documents and Settings\Bobby\My Docume
Removed a lot of useless output in between...

```
Data File Read Complete >> Patterns Loaded: 20000

 Neural Network Training Starting!
==========================================================================
 LR: 0.01, Momentum: 0, Max Epochs: 500
 16 Input Neurons, 8 Hidden Neurons, 1 Output Neurons
==========================================================================

Epoch :1 TSet Acc:0.241667%, MSE: 0.148227 GSet Acc:1.05%, MSE: 0.141374
Epoch :2 TSet Acc:7.78333%, MSE: 0.141235 GSet Acc:10.875%, MSE: 0.136411
Epoch :3 TSet Acc:20.275%, MSE: 0.13673 GSet Acc:21.675%, MSE: 0.132627
Epoch :4 TSet Acc:26.5333%, MSE: 0.133206 GSet Acc:26.775%, MSE: 0.129706
Epoch :5 TSet Acc:29.375%, MSE: 0.13026 GSet Acc:28.9%, MSE: 0.126995
Epoch :6 TSet Acc:30.7583%, MSE: 0.127383 GSet Acc:29.825%, MSE: 0.124027
Epoch :7 TSet Acc:32.0333%, MSE: 0.124402 GSet Acc:30.45%, MSE: 0.12115
Epoch :8 TSet Acc:33.2833%, MSE: 0.121577 GSet Acc:31.225%, MSE: 0.118758
Epoch :9 TSet Acc:34.9667%, MSE: 0.119156 GSet Acc:31.5%, MSE: 0.116803
Epoch :10 TSet Acc:36.3833%, MSE: 0.117107 GSet Acc:31.725%, MSE: 0.115107
Epoch :11 TSet Acc:37.85%, MSE: 0.115339 GSet Acc:32.3%, MSE: 0.113749
Epoch :12 TSet Acc:38.75%, MSE: 0.113752 GSet Acc:32.775%, MSE: 0.112292
Epoch :13 TSet Acc:39.45%, MSE: 0.112293 GSet Acc:33.45%, MSE: 0.110907
Epoch :14 TSet Acc:40.0833%, MSE: 0.110953 GSet Acc:33.675%, MSE: 0.109723

Epoch :225 TSet Acc:74.0583%, MSE: 0.0696957 GSet Acc:73.75%, MSE: 0.0763442
Epoch :226 TSet Acc:74.15%, MSE: 0.069688 GSet Acc:74.05%, MSE: 0.0763366
Epoch :232 TSet Acc:74.6833%, MSE: 0.0696364 GSet Acc:75.225%, MSE: 0.0762182
Epoch :238 TSet Acc:74.9583%, MSE: 0.0695981 GSet Acc:76.05%, MSE: 0.0760827
Epoch :239 TSet Acc:75.0417%, MSE: 0.0695936 GSet Acc:76.075%, MSE: 0.0760622
Epoch :258 TSet Acc:75.0833%, MSE: 0.0696125 GSet Acc:77.025%, MSE: 0.0755498
Epoch :265 TSet Acc:76.0417%, MSE: 0.0696256 GSet Acc:77.25%, MSE: 0.075406
Epoch :297 TSet Acc:76.6417%, MSE: 0.0695233 GSet Acc:78.05%, MSE: 0.074785
Epoch :310 TSet Acc:77.0333%, MSE: 0.0694728 GSet Acc:78.375%, MSE: 0.0749129

Training Complete!!! - > Elapsed Epochs: 500
 Validation Set Accuracy: 79.15
 Validation Set MSE: 0.0746968

-- END OF PROGRAM --
```

(https://takinginitiative.files.wordpress.com/2008/04/training.png)

Initialization of Neural Network

Okay so how do we initialize our neural network? Well again its super simple, we set all the values of the inputs, deltas and error gradients to 0. What about the weights?

The weights between the layers be have to randomly initialized. They are usually set to small values often between in the range of [-0.5, 0.5], but there are many other initialization strategies available for example, we can normally distribute the weights over the number of inputs but changing the range to: [-2.4/n, 2.4/n] where n is the number of input neurons.

Technically you can set the weights to a fixed value since the weight updates will correct the weights eventually but this will be terribly inefficient. The whole point of setting the initialization range is to reduce the number of epochs required for training. Using weights closer to the required ones will obviously need less changes than weights that differ greatly.

If for example you have an implementation where the data for training coming in is very similar, for example an image recognition system that takes in various silhouettes and their classification and trains using that data. After a few training sessions with different data you can observe the range in which the final weights are found and initialize the weights randomly within that range, this technique will further reduce the number of epochs required to train the network.

Training Data and Training Problem

So we've created our neural network but what good does that do without data to train it on? I've selected a letter classification problem as an example. I downloaded a letter recognition dataset from the UCI machine learning repository. (http://archive.ics.uci.edu/ml/datasets/Letter+Recognition (http://archive.ics.uci.edu/ml/datasets/Letter+Recognition)).

I'll be using this test data to explain the basics of training your neural network. The first thing is to format the dataset, in our case we had a list of 16 attributes and the letter those attribute represent. To make life simple the first problem I selected was to distinguish the letter 'a' from the rest. So I simply replaced the letter with a 1 for an "a" and a 0 for everything else, this effectively reduces the output to a single Boolean value.

Another problem i've included is a vowel regonition problem, where the output neuron is 1 for (a,e,i,o,u) and 0 for all other letters. This is a more complex problem and as such the accuracy achieved will be much lower.

So you can already see that we have 16 inputs (the attributes) and a single output, so most of our NN architecture is done, how many hidden neurons do we need? Well we don't know, we have to play around with different numbers of hidden neurons till we get a good result.

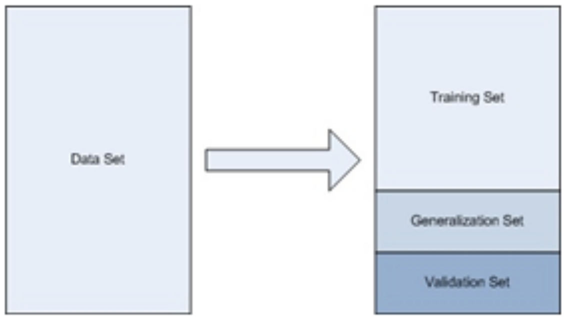
I've also noticed that there seems to be an accuracy ceiling for every NN's architecture. A good method to find out how many hidden neurons you need would be to: train the network several times (10+) and see what the average accuracy is at the end and then use this accuracy as a measure to compare different architectures.

Something that was asked in the comments section was what is the range for the input values, well the sigmoid functions is continuous over the range (-inf, inf) so any values will work but it a good idea to keep the values within the active region (region of greatest change / steepest gradient) of your activation function.

Okay moving along, so we've "formatted" our data the next step is to split it up so that we can train our neural network with it.

The Training Data Sets

So in our case we have this massive data set of 20000 patterns (entries), we can't just stick all of the data into our network since the network will learn that data and we have no way of checking how well the network will do with unseen data. This problem is referred to as over-fitting, basically the network starts remembering the input data and will fail to correctly handle unseen data. I don't want to go into too much detail here as this is something of an advanced research topic. Once you are familiar with neural networks you can read up on over-fitting on your own.



(https://takinginitiative.files.wordpress.com/2008/04/dataset1.png)

So we don't want the network to memorize the input data, so obviously we'll have to separate our training set. Classically we split the data set into three parts: the training data, the generalization data and the validation data. It is also often recommended to shuffle the initial dataset before splitting it to ensure that your data sets are different each time.

- The training data is what we used to train the network and update the weights with so it must be the largest chunk.
- Generalization data is data that we'll run through the NN at the end of each epoch to see how well the network manages to handle unseen data.
- The validation data will be run though the neural network once training has completed (i.e. The stopping conditions have been met), this gives us our final validation error.

The classic split of the dataset is 60%, 20%, 20% for the training, generalization and validation data sets respectively. Let's ignore the validation set for now; the training set is used to train the network, so if you can remember from the last tutorial for every piece of data (pattern) in the training set the following occurs:

- Feed pattern through NN
- Check errors and calculate error gradients
- Update weights according to error gradients (back propagation)

Once we have processed all the patterns in the training data set, the process begins again from the start. Each run through of all the patterns in the training set is called an epoch. This brings us to the question how do we decide how many epochs to run?

Stopping Conditions

There are several measures used to decide when to stop training. I'm going to list the various measures, what they mean and how to calculate them.

- **Maximum Epochs Reached** – The first measure is really easy, all it means is that the NN will stop once a set number of epochs have elapsed.
- **Training Set Accuracy** – This is the number of correctly classified patterns over the total number of patterns in the training set for an epoch. Obviously the higher the accuracy the better. But remember that this is the accuracy on previously seen patterns so you can use this alone to stop your training.
- **Generalization Set Accuracy** – This is the number of correctly classified patterns over the total number of patterns in the generalization set for an epoch. This gets calculated at the end of an epoch once all the weight changes have been completed. This represents the accuracy of the network in dealing with unseen patterns. Again you can't use this measure alone since this could have a much higher accuracy that the training set error.
- **Training Set Mean Squared Error (MSE)** – this is the average of the sum of the squared errors (desired – actual) for each pattern in the training set. This gives a more detailed measure of the current networks accuracy, the smaller the MSE the better the network performs.
- **Generalization Set Mean Squared Error (MSE)** – this is the average of the sum of the squared errors (desired – actual) for each pattern in the generalization set.

$$MSE = \frac{\sum_{i=0}^{n}(desired\ value\ i - actual\ value\ i)^2}{n}$$

$$where\ n = number\ of\ patterns\ in\ set$$

(https://takinginitiative.files.wordpress.com/2008/04/mse.png)

So now we have these measures so how do we stop the network, well what I use is either the training and generalization accuracies or MSEs in addition to a maximum epoch. So I'll stop once both my training and generalization set accuracies are above some value or the MSE's are below some value. This way you cover all your bases.

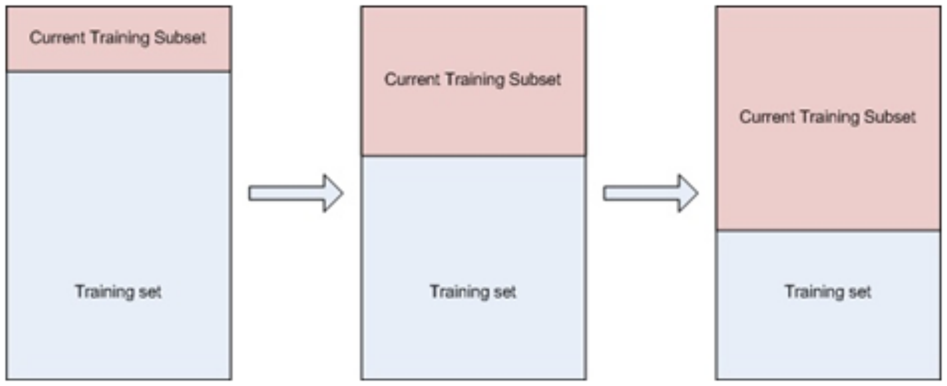
Just remember that while your accuracies might be the same over several epochs the MSE may have changed, the MSE is a measure with a much higher resolution and often is a better choice to use for stopping conditions.

Now we come to an important point that people often overlook, every time you train the network the data sets are different (if you shuffled the initial dataset as recommended earlier) and the weights are random, so obviously your results will differ each time you train the network. Usually the only difference is the number of epochs required to reach the NN's accuracy ceiling.

Advanced Data Partitioning Methods

There are several advanced concept I want to deal with here. Firstly when dealing with large datasets, pushing through a massive training dataset through the network may not be the best solution. Since the NN will have to process all the patterns in the training set over and over, and there could be tens of thousands of patterns, you can imagine how slow this will be.
So there are several techniques developed to partition the training set to provide better performance in regards to both time taken to train and accuracy. I'm only going to cover two basic ones that I've implemented into my data set reader found below.

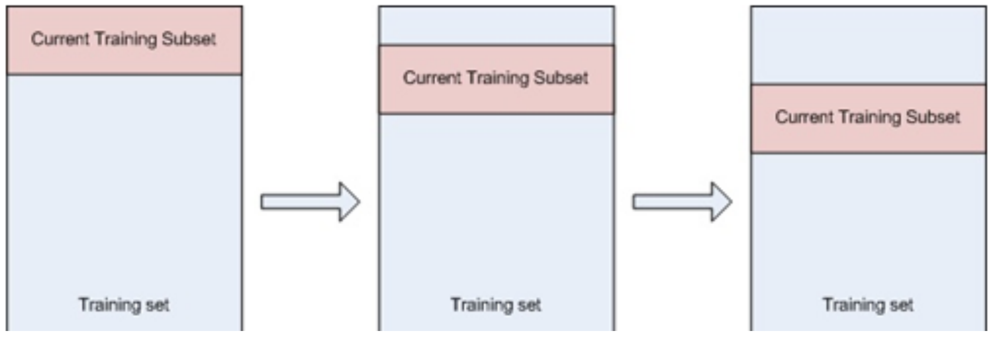
Growing Dataset:



(https://takinginitiative.files.wordpress.com/2008/04/growingdataset.png)

This approach involves training with a growing subset of the training set; this subset will increase with a fixed percentage each time training has completed (stopping conditions for the neural network have been met). This carries on until the training data set contains all the training patterns.

Windowed Data set:

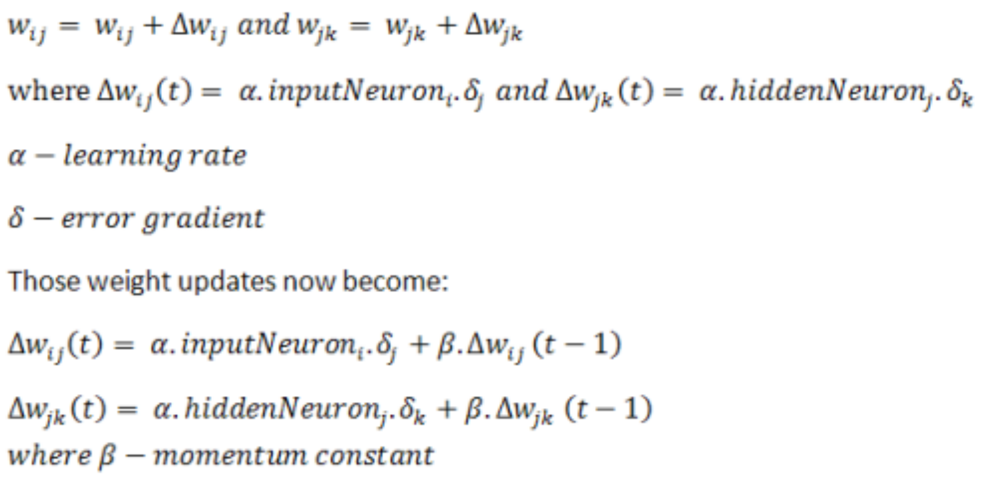


This approach creates a window of fixed size and moves it across the dataset. This move occurs once training has completed for that window. You'll specify the window size and the step size. This method stops once the window has covered the entire training data set.

Momentum

Momentum is a technique used to speed up the training of a BPN. Remember that the weight update is a move along the gradient of the activation function in the direction specified by the output error. Momentum is just that, it basically keeps you moving in the direction of the previous step. This also has the added bonus that when you change direction you don't immediately jump in the opposite direction but your initial step is a small one. Basically if you overshoot you've missed your ideal point and you don't wish to overshoot it again and so momentum helps to prevent that.

The only change to the implementation is in regards to the weight updates, remember from part one that the weights updates were as follows:



$$w_{ij} = w_{ij} + \Delta w_{ij} \text{ and } w_{jk} = w_{jk} + \Delta w_{jk}$$

$$\text{where } \Delta w_{ij}(t) = \alpha . inputNeuron_i . \delta_j \text{ and } \Delta w_{jk}(t) = \alpha . hiddenNeuron_j . \delta_k$$

$\alpha - learning\ rate$

$\delta - error\ gradient$

Those weight updates now become:

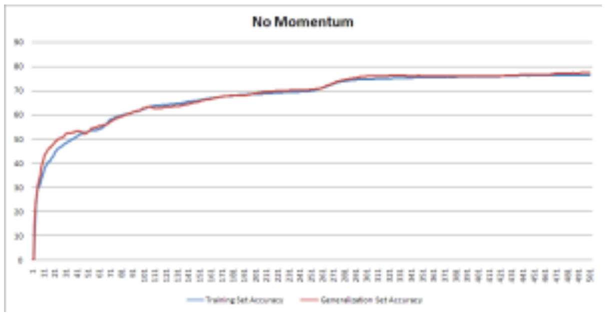
$$\Delta w_{ij}(t) = \alpha . inputNeuron_i . \delta_j + \beta . \Delta w_{ij}(t-1)$$

$$\Delta w_{jk}(t) = \alpha . hiddenNeuron_j . \delta_k + \beta . \Delta w_{jk}(t-1)$$

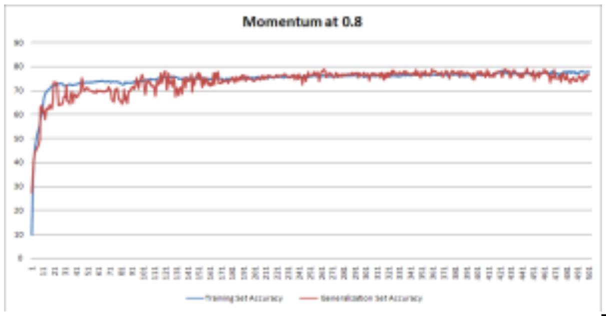
$where\ \beta - momentum\ constant$

(https://takinginitiative.files.wordpress.com/2008/04/momentum.png)

Momentum is usually set to a high value between 0 and 1. You'll notice that with momentum set to 0 the weight updates are identical to original ones. The effect of this momentum term is shown below for our training problem with the learning rate set to 0.01 and the momentum set to 0.8.



(https://takinginitiative.files.wordpress.com/2008/04/nomomentumgraph.png)



(https://takinginitiative.files.wordpress.com/2008/04/momentumgraph.png)

As you can see from the graphs the BPN will usually converge a lot faster with momentum added than without it and it also has the added benefit of allowing the back-propagation to avoid local minima's in the search space and to traverse areas where the error space doesn't change. This is evident from all the fluctuations seen in the graph.

The momentum formula's shown above are also known as **the generalized delta rule.**

Batch / Online learning

The last thing I want to go over is the difference between batch learning and stochastic or on-line learning.
Stochastic learning occurs when the neuron weights are updated after each individual piece of data is passed through the system. The FFNN therefore changes with every piece of data and is in a constant state of change during training. This is the way we've been doing it up to now.

Batch Learning on the other hand stores each neuron weight change when it occurs, and only at the end of each epoch does it update the weights with the net change over the training set. This means the neural network will only update once at the end of each epoch. Implementation wise this change is minor as all you need to do is just store the weight changes (the delta w values) and just update the weights at the end of the epoch.

The effects of this I'll leave up to you guys to discover for yourselves. I can't spoon feed you everything.

Source Code and Implementation

In the zip file below you'll find the complete visual studio 2k8 project for the following:

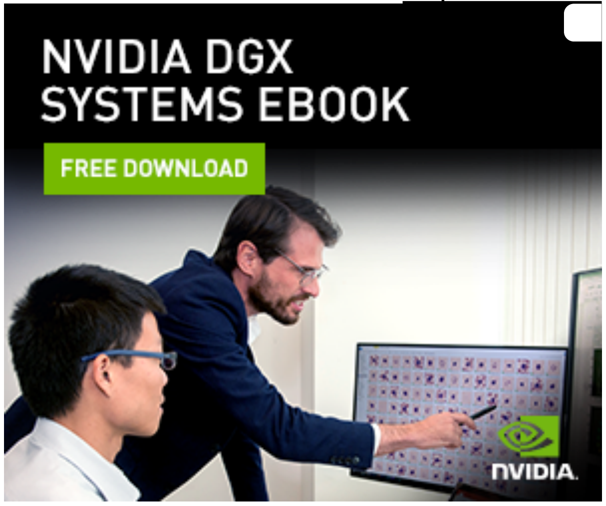
- My neural network class (has CSV logging, and has supports for momentum and batch/stochastic learning)
- My CSV data reader class (loads CSV files, has several data partitioning approaches built in)
- The test data files for the above problem

- A test implementation of the above training problem

My code is highly commented and very easy to read so any questions I haven't answered should hopefully be answered by the code. I hope this has helped you guys understand neural networks.

Neural Network Project and C++ Source code : GitHub (https://github.com/BobbyAnguelov/NeuralNetwork)
My First Neural Network Tutorial : Theory of a Neural Network (https://takinginitiative.wordpress.com/2008/04/03/basic-neural-network-tutorial-theory/)

Report this ad



Report this ad

Artificial Intelligence   **back-propagation**   **Neural Network**

203 thoughts on "Basic Neural Network Tutorial : C++ Implementation and Source Code"

Pingback: Basic Neural Network Tutorial - Theory « Bobby Anguelov's Blog
Pingback: Basic Neural Network Tutorial : C++ Implementation and Source Code « gmgPuBlog

Somehow i missed the point. Probably lost in translation 🙂 Anyway … nice blog to visit.

cheers, Vamper!!

Vamper , June 23, 2008 at 8:00 pm
Reply
One of the best reasons to model a neuron as an object is that you can extend the parent class from neuron to create "sub networks" that are specially trained, with one manager network on top. And, what are you coding on? An apple II+? Memory's like 10 bucks a gig, and your neuron class takes less than a K of extra space… WAY less. OOP's the way to go. Accept it.

NegaScout , June 27, 2008 at 6:03 pm
Reply
Hi NegaScout,

as long as you use NNs just for fun you can waste all the memory you want. When you start doing real research, you discover quite soon that calculations are very eager for memory and CPU time, and you are always short of time…

Riccardo , December 14, 2010 at 7:19 pm
Reply
I agree with the array approach but I am quickly discovering the difficulties in attempting to perform genetic operations on a network implemented as a multidimensional array. Not saying its not possible, just still trying to work out the logistics. Objects make this easy of course (i.e. you can have a .Child = function in the object etc)…But all of your arguments against using classes are all still valid AFAIK. That said, even the simplest evolutionary techniques (i.e. adding a random Gaussian distributed number on to a weight and keeping it if the fitness improves) converges far quicker than back-propagation on a simple MLP. (for me anyway…).

Nick Emblow , March 15, 2012 at 1:06 pm
I agree with this, I am working on Artificial Neural networks for research purpose where I have to generate a population of them in an evolutionary algorithm fashion and run over lots of generations. And As you said, you quickly realize the limitations of memory and CPU with his stuff, especially if you are not using a high-spec PC. Even when using GPU computing with help of NVIDIA Graphic Cards, the cost of transferring data for operations and back to your memory would be something to worry about.

Bottom line is… OOP is not just suited for it in my type of work, atleast not fullscale.

P.S Kudos for your tutorial, 🙂

Abdullah, Neural Net Researcher , January 4, 2013 at 5:59 pm
To create what you want I'd just need several full networks and can avoid any inheritance. Also just because you have memory and resources doesn't mean you should use them. I always try to aim for the smallest, neatest, most efficient approach to anything.

It just adds unnecessary complexity, apart from the case where you'd want each neuron in a single network to use a different activation function or input aggregate (since thats all a neuron contains), but till now i haven't seen any academic literature supporting such a network.

Explain to me the benefits again? separation of the training was a good idea since it allows multiple ways of the training the network and reduced the amount of code. Encapsulating neurons isn't.

Also I'd love to know where you live cause memory is around $60 a gig (for the good quality stuff) and i'm running a monster: e8400 @ 4.3ghz, 8800gtx @ 8800ultra speeds, 1.3tb of hard drive space and 4 gigs of ddr2-1200 ram…

***Bobby* , June 28, 2008 at 3:26 am**
Reply
I agree after looking into the information I figured you can avoid less cycle ticks by using factorials instead of a number N where N should be ! instead of a number as we see that the figure explains MSE as a means of squared error whereas the root cause would be the inverse of pathegoram when a^2+b^2c^2

***Joseph Hightower Simmons* , May 1, 2014 at 3:43 am**
Reply
I've been reading through NN tutorials for the past few weeks, this is by far the best. Many thanks Bobby.

***dan* , September 11, 2008 at 3:45 pm**
Reply
in defense of the author, for systems such as microcontrollers, or GPU's, you need to be really conservative with memory

***s* , September 14, 2008 at 8:02 am**
Reply
thanks a lot,it's good to me.Can you send me some data about "Basic Neural Network Tutorial : C++ Implementation and Source Code".

***chi* , September 27, 2008 at 9:35 am**
Reply
nice post bro..

***ump student* , October 8, 2008 at 6:52 am**
Reply
Занимаюсь дизайном и хочу попросить автора takinginitiative.wordpress.com отправить шаьлончик на мой мыил) Готов заплатить…

***spenly* , October 14, 2008 at 12:33 pm**
Reply
hey, my russian is terrible, from what i gather you want me to send you an email about something?

***Bobby* , October 14, 2008 at 12:40 pm**
Reply
hi, this tutorials are quite awesome, many thanks for them
but i do have a question, i read you were doing some image processing with neural networks, could you explain a bit how do you do that?
i know some basics of image processing etc, but i have no idea how to apply neural networks with image processing
for example, if you have some picture with some red rectangle, what is the input for neural network (whole picture, part of picture, filtered picture, etc) ?

thanks in advance
cheers!

ps: is it possible, that the c implementation is almost two times as fast as c++? (i might have bugs in my program, but i think stuff works just fine, oh and i compiled both the same way)

Lightenix , November 8, 2008 at 10:36 pm
Reply
i used it for the classification of images, basically it told me what the image represented.

Your input to a NN is entirely dependent on what you want to put into the network and what you want out, you need to define your inputs and outputs accordingly.

as for the implementation speed, what do you mean its two times faster? does it learn twice as fast or does it take half the clock cycles per epoch? Remember that the learning time varies every time you run it, so for one run you might get it to reach 90% in 50 epochs and for the next run it might take 400. you cannot make any sort of learning time expectation…

Bobby , November 8, 2008 at 11:57 pm
Reply
Will your program work for this?
Question: Implement an Artificial Neural Network (ANN) based classifier that takes as input a 10×10 array of binary values.
1) The input array should contain digits (0 to 9) written in different patterns.
2) Each digit should have at least 3 training examples in the training data so you
have to create a total of 10×3 = 30 training examples.
3) Your classifier should find a weight vector that can be used to classify patterns
of this type.
4) Number of input, output, and hidden layer size will be your choice and you should
provide justification for that. Try to be simple in your design.
An example input of digit 1:
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 1 0 0 0 0
0 0 0 0 1 1 0 0 0 0
0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0 0 0
0 0 0 0 1 1 1 0 0 0
0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0

Afnan , May 21, 2015 at 10:42 am
Reply
i used srand(1) for initializations, soo all weights would be the same at the start and both programs were run 150 epochs, also weights in both programs were initialized the same way, soo i don’t think there should be any differences each run
for timing i used this:
time_t secondsStart, secondsEnd;
secondsStart = time (NULL);
which gives accuracy of a second, anyway, programs returned 26s and 42s, soo greater accurancy shouldn’t be needed
i started measuring time at the start of main and ended it at the end. This way it is measuring whole

initialization, even saving weights (which i don't have).

anyway, bugs are not excluded =)

cheers

Lightenix , November 9, 2008 at 12:43 am
Reply
that timing method is inaccurate, if you want use my timing class for the timing and let me know what results you get.

wanna send me your code I'm curious to see what you've done? banguelov (@) cs.up.ac.za

Bobby , November 9, 2008 at 9:37 am
Reply
hi bobby, this really helps me as a complete beginner in NN, but why i can't download the zip files?

liyen , November 9, 2008 at 8:57 pm
Reply
Hey. Love the tutorial. Although I use a different programming language so would it be ok if I emailed you with specific questions? If so, could you email me.

So hard to find in depth NN tutorials. This is definetly the best. Thanks!

Louis , November 15, 2008 at 9:20 am
Reply
hey bro, thanks for the kind words, I'm crazy busy at the moment but you can mail me any questions you want but I'll only be able to reply mid next week.

Bobby , November 15, 2008 at 9:31 am
Reply
Thanks mate. Can i have your email address pwease? I couldn't find it on the site 😛

The current question I have at the moment regarding Bias and threshold. Should there be a bias neuron in each hidden layer. Say I have 2 hidden layers, does that mean I need 2 bias neurons?

Louis , November 16, 2008 at 2:17 pm
Reply
Hey, Excellent tutorial, you explained the concept very well!

I have an issue with character recognition using Matlab NN toolbox, how do I pass an image to a Neural Network? I've done the feature extraction using image processing but i dont know how to pass the image to the train function. you can reach me at cyber_kobe@hotmail.com
thanx in advance,

jago , November 17, 2008 at 7:50 pm
Reply
Great work, man. I have one question: after computing the weights and all the NN parameters, how do you apply this latter to a new data without training?

Thanks.

***melfatao* , November 20, 2008 at 10:05 pm**
Reply
I'm take an Introduction to Nueral Networks module and intend to try and implement one myself, this is a lot less high brow than the daunting notes made on the mathematics of it all.

Thanks for summing it up in English for us mere mortals 🙂

***Adrian* , December 10, 2008 at 12:31 am**
Reply
Hey….
how can i run this program in visual studio 6(visual C++) ?
thanks in advance…

***Jewel* , December 21, 2008 at 6:23 am**
Reply
just create a new empty c++ project and import the files, then compile. should work perfectly 😛

***Bobby* , December 23, 2008 at 4:10 pm**
Reply
Hi..Bobby, firstly, thanks you for your explanations.
I am a new student who learn ANN.
I ask about what parameters should I fill in the project ?
would you mind send me the project you mean..here is my email: ismail2610@yahoo.com
Thanks you in advance Bobby.

***ismaildatuak* , June 4, 2015 at 4:20 am**
Reply
G'day Bobby,

Thanks for the C++ code and the tutorial. Still trying to get my head around NN's, and your tutorial was most helpful.

I want to try using a NN to predict horse racing results…

What I was thinking of doing was to compare two horses at a time – have x number of inputs and 1 ouput ie horseA beat horseB (0/1). What I'm trying to work out is how to organise the input. I have a comprehensive database going back several years, so I have plenty of data to play with. I can extract whatever I like using SQL and Python to massage the query result sets and create the csv file. No problem there. What I would like to know is how to lay out the input data – two horses per line, each line containing back history and stats for both horses… But how do I tell the NN which data belongs to which horse? Or am I barking up the wrong tree?

Thanks again and kind regards

***Peter* , February 17, 2009 at 4:11 pm**
Reply
Greetings,

For being a perfectionist, you surely do have a lot of spelling and/or typos in this post. Hehe … sorry to bust your balls about that one, but I would like to say that this code is great and is exactly what I was looking for. You have a talent of expressing complex situations in a simple manner. Usually when

I read some source code I find on the internet, I find myself running around their code wondering why the hell it's so complicated. With yours I opened it, compiled it even after switching the .sln file to MS Visual Studio 2005 on the first try, and ran it in the debugger very smoothly. The code is a little slow, but I have a feeling some of the time is spent printing out all the cout's but I'll about that one soon.

Anyways, great work and I'll leave some more feedback when I get more into the code and seeing how extendible it is.

,Brian T. Hannan

Skype: brianthannan
email: bnubbz@hotmail.com or bnubbz@gmail.com

Brian T. Hannan , February 20, 2009 at 11:13 pm
Reply
haha, yeh I don't bother proof reading the tutorials, I neither have the time nor the patience for it.

The code would probably be slow exactly for the reason you mentioned, it's meant to be example code more so than a fully working implementation.

As for the simplicity, most things are really simple but are just poorly explained, or the people explaining them over complicate things to make themselves seem smarter.

anyhows, glad the tut helped you!

Bobby , February 21, 2009 at 12:19 am
Reply
Bobby,

one thing I still can't grasp when it comes to neural nets is the expected output. What is this? Is there a node per possible identification? for example, to recognise between an 'x' and an 'o' do you need 3 outputs. One at 100% confidence of an 'x', one for an 'o' and a third for garbage?

Thanks for a great tutorial, its been a lot of help

Scott

Scott Higgins , February 24, 2009 at 10:05 pm
Reply
yeh scott, that's exactly right, the output is basically a big selector with a node for each type. In some cases you might use both states for your output.

like there was a case where i had 6 classes that i needed to distinguish from, so i used 3 outputs and then used the binary representations of 1-6 on them for the output:

ie: no class (000), class 3(011), class 5(101) etc…

Bobby , February 26, 2009 at 9:36 am
Reply
fantastic, I have managed to get your nerual net to work perfectly with my mouse gesture recognition program. It is very good code and easy to follow, thank you.

I have, however, found a bug with your code. I am going to try and fix it, and if I do I will let you know.

Basically, if I extend the number of input nodes to 32 and hidden nodes to 20, then train the network and save the weights. Running the program again, loading in the weights, appears to have a memory leak? I can feed forward once maybe twice and then it has a moan about the heap.

I will let you know how I get on,

Thank you again,

Scott

***Scott Higgins** , February 27, 2009 at 1:40 pm*
Reply
it's very possible, the saving and loading of weights was an afterthought and i did it in a rush, its possible I've done something silly, I'll see if i can find the time to track it down.

***Bobby** , February 28, 2009 at 8:59 am*
Reply
I like the code and layout, and am intrigued by the simplicity and speed, but it seems that the code is suited for only one purpose: classification. And given that neural networks are not the best for classification, I was hoping this could be more general (real valued outputs, being able to run exactly one iteration of backprop) Also, you only allow one hidden layer? I think maybe 2 hidden layers max is good, but only one isn't using the full strength of neural networks (it will have proven limits, and isn't necessarily the best for every situation). Would adding these features significantly degrade the performance? I would change the code myself, but I am afraid to, and if you do it, it will be more self consistent, and other users will be able to benefit. Thanks.

***Trevor Standley** , March 2, 2009 at 9:10 pm*
Reply
why do you say neural networks are not the best at classification? A neural net is by definition a classifier, and as such is what it's most commonly used for. You give it inputs and it give you an output based on the inputs, it classifies the inputs into a set output.

Adding an extra hidden layer is easy, but once again do you have any evidence that adding this extra layer improves the accuracy of the net?

Adding an extra layer will negatively affect the total iterations needed for training as the number of weight changes are greater.

The outputs are real valued, i just manually force them to be binary using a basic step function. Remember that your neural net will only be as good as the training data, and you have control over the training data.

***Bobby** , March 2, 2009 at 9:31 pm*
Reply
"A neural net is by definition a classifier" I don't think so. I think it is more general, it is a function approximater. It is capable of learning a real valued output function.

What I meant was that support vector machines and Bayesian networks are often preferred to neural networks for classification, boosting is also a common technique for classification.

The project I am working on is a reinforcement learning algorithm, and I am only using back propagation to match a small set of seed data.

I know that adding a second layer makes training much slower, but the function I am trying to learn comes from a complicated space, and the network is empirically unable to learn to match it with a single layer.

Trevor Standley ***, March 4, 2009 at 3:50 am***
Reply
a neural network is a pattern identifier and as such can be seen as a classifier. Whether the separate classes can be approximated by a function isn't guaranteed. I don't have the literature handy but my professor said there is academic literature that proves that you should never need more than a single hidden layer in any NN.

If you are having trouble training the network it might be advisable to change your activation function or training method rather than adding an extra layer. There is also a possibility that you have too many hidden neurons in your hidden layer, there is no method to calculate the optimum architecture of a NN.

I don't have much experience with bayesian networks or svm' so I cant comment on that.

Bobby ***, March 4, 2009 at 1:52 pm***
Reply
"I don't have the literature handy but my professor said there is academic literature that proves that you should never need more than a single hidden layer in any NN."

While it is true that any single valued function can be approximated to an arbitrary degree of accuracy using a single hidden layer, the decision regions used for classification are defined by multi-valued functions, and with only a single hidden layer it is not possible to represent concave decision regions or regions with holes (a bullseye for example requires 2 hidden layers to represent).

The representation power of the neural network is very important for certain problems, especially ones in which the output of the NN is to be a vector of real values.

Trevor Standley ***, March 9, 2009 at 10:30 pm***
Reply
please sir… i cant see the C++ code.. im a v source oriented person.. a piece of source code is all i need.. i have be reading the first part as well i.. thanx for tute…

Asitha ***, April 28, 2009 at 8:35 am***
Reply
a neural network

Lal ***, May 25, 2009 at 9:10 am***
Reply
Awesome article! Although I've read several introductory texts about neural networks, none them actually showed any calculations, strategies, or gave sample code; congratulations on a fine piece of work!

***Kenton* , June 4, 2009 at 8:01 am**
Reply
I you want to make it even faster you should consider programming in asm.

***David* , June 11, 2009 at 12:04 am**
Reply
bro, this tute is awesome..but i can't download your source code…kindly post it again in this article or email me at ememeke@yahoo.com…
thanks and more power…get neuralized…

***kim* , June 25, 2009 at 5:31 am**
Reply
how would you tackle an optical character recognition? i mean you assumed 1 for a letter a and 0 for the rest. but i dont think it is valid for an OCR. can you help pls? i really like your work.

***Xer* , July 29, 2009 at 10:26 am**
Reply
see Neural Networks in action:
http://www.sharktime.com/snn

***SharkTime* , August 18, 2009 at 1:25 am**
Reply
its nice , but since i agree with you that a simple solution is a best solution, so why not explain how a simple OR gate is implemented or a simple caharcter recognition problem show how the nenetwwork is traine using simple c languge code , anyway thanking you, for your valueable ideas

***pankaj* , September 19, 2009 at 1:09 pm**
Reply
honestly both those are really simple and I'm sure you can figure it out, I'm not in the habit of spoon feeding people.

this tutorial has source code that is pretty simple to understand.

***Bobby* , September 19, 2009 at 1:22 pm**
Reply
Hi Bobby!

First, I would like to thank you for this great blog. I find it best among others I have found online. Actually, I have used your example in implementing my master thesis. Of course, I have to optimize some things, but generally, i used your idea. I wanted to ask – why not use momentum for batch learning? I think it could be used, if error is less than previous epoch error…

***Dalcek* , October 28, 2009 at 1:15 pm**
Reply
The momentum term is added to stochastic/online learning to overcome the fluctuations in the sign of the error, the neurons keep going back and forth unlearning and relearning with every new piece of data and so momentum helps to curb that fluctuation.

Offline/batch learning doesn't have this problem so the momentum term is unnecessary since all the weight changes are averaged before being applied. Also due to the weights being averaged, you cannot assume that the pushing the error in the same direction as on the past epoch will be a better solution.

I hope this helps to answer your question?

Bobby* , *October 29, 2009 at 2:38 pm
Reply
Yes, thanks! I use your example, had to change it.. It works well for simple functions.. both online and batch learning, but unfortunately for me, doesn't work well for problem i have to solve, some predictions.

Average relative error is about 20%, where maximum allowed is 10. I am not sure why, but i am trying to figure out. I think I should try with bias separated for each neuron.

I am not sure if the way you implemented is the same as without neurons. Are you?

Dalcek* , *October 30, 2009 at 12:47 pm
Reply
hi Bobby
do you have any mtlab code for back propagation batch learning ?i have problem with it i write this program but instead of reduced the error ,it increases the error could u pls help me what is the wrong?

zara* , *November 21, 2009 at 5:30 pm
Reply
The main rule in OOP is "treat everything as an object".
So your program isn't OO and that's true – is very simple, like this network.
Try adding more algorithms, even layers nad you will see…

qwe* , *December 24, 2009 at 3:26 pm
Reply
And if your project will grow, then you will be rewriting your "simple" code…

qwe* , *December 24, 2009 at 3:30 pm
Reply
Weight could be an object to. Lets see..
Add some parameters to the weights, like delta,gradient,updateValue(all needed in RPROP). You will create 3 arrays of it, I just add 3 variables and getters/setters to it… Simpler?

qwe* , *December 24, 2009 at 3:38 pm
Reply
yeh, sure OO is one way to go about it but i personally prefer data driven approaches.

The problem is that in the OO methodology you always think of a single object whereas in many cases you have hundreds/thousands/millions of mini objects.

instead of just having an array of ints for the weights you know have object with their own methods, all adding overhead and extra memory, also to update the object instead of running a single function on the array you know have to store an array/list of weight objects, then run the get function on each object (i.e. look up the memory address of the get function, then get the memory address of the weight and return that), that's begging for cache misses. To improve performance you would also like to have all the objects sequentially stored in memory and you'd have to add extra code for that, etc,

etc…

so you propose for example replacing an array of int's lets say 256 weights (1mb of memory) with objects for each weight so 1mb + overhead of the classes, plus another array/list (even more memory) to store the addresses of the weight object so another 1mb of memory, and this just for the weights.

you also want to add in functions for get and set further slowing things down and adding overhead, and what is the benefit? You get a more OO and slower network, a much larger codebase, slower compilation times and more than double the memory usage.

adding parameters to the weights can simply be achieved by using a mixin class to a skeleton NN and some simple inheritance.

I've written some pretty large projects and OO is something that has its place but mustnt be take to extremes! Unfortunately idiot lecturers that only know theory drive the thought into students head that OO is the ultimate programming approach, and barely even touch data driven or other approaches.

Bobby* *, December 24, 2009 at 4:44 pm
Reply
hey, checked out your code and have to say thanks for the example/tut. it has helped out immensely. noticed someone was complaining about the load function. when you dynamically allocate memory yous gots to clean up after yourself buddy.

```
add: delete cstr;
after: t=strtok (cstr,",");
```

in your loadWeights method. otherwise memory leaks galore.

thanks again.

btb, dragon age cannot under any circumstances be compared with bg2. it's just not right.

dragon_age_sucks* *, February 10, 2010 at 8:11 am
Reply
whoa my bad, add that after that while loop.. should be like this:

```
while ( t!=NULL ){
weights.push_back( atof(t) );
t = strtok(NULL,",");
i++;
}
delete cstr;
```

I guess I was in too much of a dragon age hate rage to notice the memory was still very much needed for that token splicer. take care.

dragon_age_sucks* *, February 10, 2010 at 8:49 am
Reply
while I will agree Dragon Age isnt BG2, but i cant think of a since RPG in the last 5 years that's come as close, and honestly compared to the last few RPGs to be released DA was epic. The last proper RPG

I've played was Yhe Witcher and before that it was Vampire:bloodlines and TOEE. Not a lot of decent RPGs lately, so when something like DA comes out I'm thankful!

As for the memory management thing, you are completely right!!! I'll update the code when I get to work. Haha, these kinda things slip past me when I rush to try and write the tutorial code. Especially the V2 code, I basically retrofitted V1 in around an hour.

I dont even wanna look at any of the other code atm, since they'll probably be other errors or things that can be improved.

Bobby ***, February 10, 2010 at 9:22 am***
Reply
right, well if d&d didn't sell off their use rights exclusively to Atari maybe dragon age could have been something more comparable to bg1/2. Or at the least maybe we would see a bg3 considering its loyal fanbase. BG set the bar far too high and I haven't seen an RPG come even close to the standards it has set (for me). I'll trade bigger and better graphics for a game similar to BG any day. infinity engine & sprites ftw!

dragon_age_sucks ***, February 10, 2010 at 7:30 pm***
Reply
sorry to double post again, quick question tho. your code takes doubles as its input, and returns binary. Is this common for a NN? I'm planning on using binary input to return a character (an OCR). So the opposite would seem more appropriate. I read somewhere that I should use -.5 to .5 for pixel representation, it's supposedly more efficient for an NN? It just seems like a double for each input that is simply representing a bit is highly inefficient. I'd rather use bitwise operations to get them crammed down into a couple bytes, instead of a couple bytes per bit. And then I guess just assign each char a bit representation for the outputs? a=000,b=001,etc.

dragon_age_sucks ***, February 10, 2010 at 7:53 pm***
Reply
TBH I'm not much of a fan of the fourth ed D&D rules, I liked the specialization of 2nd ed.

for the NN, the output layers outputs are clamped to 1 and 0, so each category you wish to categorize stuff into will have a binary representation.

so just assign a binary number to each category.

so-> 000, 010, 011, 100, etc…

Bobby ***, February 10, 2010 at 8:34 pm***
Reply
Hi,

I understand the code but I couldn't not find out how to test it?

Could you please help?

Thanks!

Sule ***, March 9, 2010 at 9:10 pm***
Reply

> Trevor Standley :
> *"I don't have the literature handy but my professor said there is academic literature that proves that you should never need more than a single hidden layer in any NN."*

While that is technically true, it requires that you have a hidden neuron for each possible input state, which relly just reduces the NN to a resource hogging lookup table. For true generalization you often need multiple hidden layers. In real world applications 2 hidden layers with 5 nodes each will do a far better job of classifying that a single hidden layer with 10 nodes.

Anthony , April 30, 2010 at 1:24 am
Reply
hi
ive got a question, how can i test your nn?
i mean for ex. i've got 10inputs and 1 output, i've created nn, and now i want to find a result for some input combination.

luke , June 1, 2010 at 4:12 pm
Reply
got it

luke , June 1, 2010 at 8:31 pm
Reply
Thanks for posting this, very helpful for C++ source code!

Coder , June 24, 2010 at 7:36 pm
Reply
Hi, Bobby,

Very Nice! Browsed through your code, even for a C++ novice like me it's very accessible.

Question: with arrays, would you not run into memory limitations if the network gets larger? I am looking for a framework to start with, yours might be it, but if later on I find out that I should have be using pointers instead?

My ultimate goal: object recognition and classification by window-scanning an image using a stacked network of generic trained 3 layer network and an ART-CNP classifier on top.

Ha… that will be a while.

Thanks, Jan

Jan de Lange , July 4, 2010 at 3:06 pm
Reply
I am using dynamically allocated arrays in my code so memory limitations aren't an issue (I am using "pointers"). In my description i meant array as the basic data structure over something like a vector or list.

Bobby , July 4, 2010 at 5:01 pm
Reply
hello BOBBY

can you help me how can I use your code as a classifier to detect a trained frontal face in a video sequence?

Afshin ***, July 17, 2010 at 4:48 pm***
Reply

The code provided is just to demonstrate a working NN, while you can easily take the code and slot it into your application, that is the smallest step required for what you want to do. In your case you will need to somehow convert the face image data into a suitable numeric format for the neural network to train upon and hope that your data set is of good quality and your pre-processing is robust enough to be able to accurately represent certain facial data correctly and distinctly.

If you can convert your data to some sort of numeric format you can easily plug the data into my code and train on it. If trained successfully, the NN should then be able to handle unseen data.

Bobby ***, July 18, 2010 at 4:45 pm***
Reply

nice post, thx

Lern3R ***, August 15, 2010 at 4:49 am***
Reply

Hi Bobby,

very nice blog, one of the best if not the best.
Maybe you can help with a problem I have some time already with representing certain input data into neurons. I am dealing with forex market, and am trying to represent the moving price as regard to a constant line and it's affection on the price. For instance, if the price is moving towards a constant line, I want to represent this with the distance from the line, but then the price can also cross the line, and then that should be represented as a minus distance. Is there another way to do it? because I want to represent a certain truth that I think exists, that the nearer the price gets to a certain horizontal line, it's behaviour becomes different than when it's far away from that line.

Hope that it's not too much,

thank you for your great work.

Amir ***, September 3, 2010 at 12:11 am***
Reply

Hi,

When it comes to data pre-processing there really isn't any standard technique, you need to try various representations for your input data and see what works best. You're pretty much asked the single hardest thing about NN, implementing them is easy getting the right data inputs is not.

Try using the distance from the line initially and if your results aren't satisfactory, then I would suggest trying to use something else like the gradient of the price graph at a particular time (you said the price changes so it should easy be graphable or perhaps the delta between the current price and the previous price. I honestly have no idea what would work and am blindly throwing out suggestions.

Best of luck tho, and I hope you come right with it!

Bobby ***, September 3, 2010 at 9:08 am***
Reply
Great work Bobby

Vic ***, September 10, 2010 at 7:20 pm***
Reply
Hi Bobby,

Thanks for this clear tutorials and code. I just need to check something. Your main function ends after training the network. If I need to extend the code to query the network with unseen data. Shall I use the CBackProp::feedForwardPattern()?

Thank you.

Essam ***, September 29, 2010 at 8:58 pm***
Reply
yeh, exactly, just feed forward any unseen data and the network will return the classification of the data.

Bobby ***, September 29, 2010 at 9:38 pm***
Reply
One more question …
The CBackProp::feedForwardPattern() function returns an array of integers. Does this mean that your implementation is only suitable of classification problems? How can I use it for regression problems?

Essam ***, October 7, 2010 at 5:07 pm***
Reply
Another question..
Is there any way to create the NN with more than one hidden layer?

Essam ***, October 7, 2010 at 9:37 pm***
Reply
Essam, the code I have for the NN is just a basic example to cover designing and building your own network. If you wish to add in another hidden layer it's really simple but it will require some modification. As for the outputs, I've never really done any sort of regression analysis and am not sure of the suitability of a NN for that sort of problem.

At the end of the day my tutorials simple cover the basic theory of NNs, if you wish to use them for any real life situation, i highly suggest doing some additional reading and not using my example code as it is just that, example code. It hasnt been extensively profiled or tested and is just there to be used as a strating point!

Bobby ***, October 8, 2010 at 8:31 am***
Reply
Thank you Bobby for the clarification.

Essam ***, October 8, 2010 at 7:06 pm***
Reply
Hey Bobby,

Thanks again for the awesome post, twas quite helpful indeed.

I am trying to use and modify your code to accept arbitrary data patterns which must be formatted to fit into the program. I'm currently having some trouble rewriting or understanding your method for your code above.
You said there was a single output, letter classification, yet in the source you have an array of target values from which i figured to be the last three columns in the data file.
if you could please shed some light on the method or underlying reasoning behind your choice i would greatly appreciate it.

Thanks and regards.

yates , October 11, 2010 at 11:23 am
Reply
yeh there is a single output from the network, the letter classification but this output consists of three values since in my network a neuron can only ever be on or off. So we can think of each neuron as a binary value. I'll be honest, looking at the test data, i'm not really sure how it classifies each letter, i think it groups them together but i cant remember according to what the grouping is performed.

The 3 outputs give a total 7 different categories but what those categories are I'm not entirely sure, haha. If we wanted to do letter recognition we would need at 5 outputs to cover the entire range of 26 letters. I hope that answers your question.

Bobby , October 12, 2010 at 3:23 pm
Reply
yeah thanks man,
that is the info that i was hoping to get cos im trying to figure out how i should classify my data based on the dataset that you used above.
I checked out the original dataset from the url you provided in your blog and it seems they have the actual letter as an output value…so your data is changed but you say you cant remember how it classifies it exactly? is it nothing to do with each letter's ascii binary values perhaps? just a guess.
But anyways its cool, that's fine man, im thinking about it as i go along and hopefully i'l find a decent way of classifying the target data 🙂
thanks

yates , October 14, 2010 at 9:01 pm
Reply
very nice program sir
can u help me for how take the weight in application program

kitti , November 2, 2010 at 10:05 am
Reply
Hey,

i am getting erros with your code. I am using eclipe as environment, but somehow i am getting error in neuralNetwork.h file.

Line; " friend neuralNetworkTrainer; "

Error; " Multiple markers at this line
– friend declaration does not name a class or function
– a class-key must be used when declaring a friend

– Line breakpoint: neuralNetwork.h [line: 35]
– Syntax error ”

what should i do?

Edwin Ranjitkumar , December 4, 2010 at 9:13 pm
Reply
Edwin:
You should put a key word ‘class’ before name of class (neuralNetworkTrainer), so finally it will be: friend class meuralNetworkTrainer.

I have also a question related with vector of features. I’m implemented it in Matlab, but unfortuantelly my implementation give results that are different in relation with those in database. Can anybody show/sent me a implementation of feature extraction?Ofcourse reults are associated with style of letters, but i think that something is wrong with my implementation.

Matthew , December 22, 2010 at 10:08 am
Reply
How much overhead would using OOP really cost? It produces very readable code and if done correctly, almost no overhead; and from my own independent tests, no loss in performance.

Mind you this is based off my own OOP Backprop ANN, I’d be happy to share it with you to exchange knowledge on fast code; as that is my pet peeve also :).

Daniel , January 11, 2011 at 2:38 pm
Reply
honestly the overhead will probably be minimum, but in such a simple case there is no real benefit to it! Having a custom node class that simply stores two floats value and a link to a activation function is a bit pointless, you’ll add in a lot of extra complexity for no benefit.

There is a situation in which performance can be significantly affected, if you create each node in your NN with a “new” call then you have no guarantee that nodes will be create sequentially in memory (they wont be). Every time you perform activation function calls and weight updates you will incur more cache misses than with the simple value array since they may offer a degree of precaching when accessed.

At the end of the day an OO approach will work just as well and may not be much slower but what does the added complexity offer you? did it really make it that much simpler to understand? I feel my implementation is pretty simple. You now have a bunch of extra custom types and the need to worry about creation and deletion of these types and so on. Sure its not a headache but why do if you dont need to? 🙂

I believe in simplicity and minimalism in code, it makes it easier to debug as well…

Bobby , January 11, 2011 at 8:08 pm
Reply

> Minimalistic (BP) Neural Network
>
> That is one I re-wrote last night and today, I must agree with after comparing it with my OOP comparitive network, it is over 400 lines shorter, runs faster, and easier to debug in the long run.

Ultimately I had to write as I am hoping to use it on a microcontroller; just have to change the memory management to C malloc among other things.

Daniel , January 12, 2011 at 9:55 am
Reply

Hi Daniel,

I hate your bracketing style!! hahahaha 😉

I'm glad it worked out for you, so it is faster than the OO implementation? Can I ask by how much, I figured that there would have been a slight improvement but i didnt expect it to be more than 5~10%.

I use OO approaches when it is suitable to do so and data driven approaches when they are more suitable. Its all about using the right approach for the particular problem. I think programming is as much an art as a science, and a good programmer needs to be able to adapt to various problems he encounters.

I guess a part of the problem is that colleges tend to praise OO as the ultimate programming techniques (at least that is the case at our CS department) and students are taught to think in an OO fashion from the start.

If you change new to malloc, delete to free and the cout's to printf's I think you network will be fully c and should be perfect for an embedded controller 🙂

Bobby , January 12, 2011 at 10:15 am
Hahaha. K&R is no fun, and it just bloats the code. I've used 1TBS for a while; its great :).

Added an XOR in there, it results in ~230 iterations on average.
As for the differences between this and the OOP variant (I can post for you), with 20,000,000 iterations going for them (insane I know), I racked up about 10s on average compared to 12s (OO) on average.

There results varied a lot, however the OOP variant would complete the XOR problem on average in 500 iterations, so I think my non-OOP variant has a few fixes for the training algorithm (noted this during refactoring).

In which case, I'd say equal all around.

Daniel , January 12, 2011 at 10:53 am
I loved how you simplified the code. Very useful for use in microcontrollers.

I compiled your source after changing random() to rand() and randomseed() to srand(time(NULL)) using MINGW compiler. Because I couldn't find the definitions of the former. The program doesn't train this way…just goes on and on with a blank output console. So, I think the problem is with random weight generation.

I downloaded rmath.h from the stk++ project but that didn't help either.
Could you tell me what the solution is?

Abhipray , September 27, 2011 at 5:59 pm

I have no idea, its simply enough to debug tho so just step through the NN feedforward and

BP steps with a debugger. NNs are only as good as the data you feed them, if you put in uncategorizable data or a really small data set then it will not train.

Bobby* *, September 27, 2011 at 6:02 pm
Oh never mind, I just used Bobby's idea of range for initialization!

Abhipray* *, September 27, 2011 at 6:11 pm
@Abhipray
Apologies mate! rmath.h was my own custom header, nothing to do with STK++.

The difference between my random() and rand() was that it default returned a float between 0 and 1. rand() returns a number between 0 and RAND_MAX!

That is why you were having huge problems, it would take so long for the network to overcome the huge weights, because they are no where near the problem space.

Try using this: (0.5 – ((float)rand() / RAND_MAX)) * 2;

regards,

Daniel* *, September 28, 2011 at 7:12 am
Personally i cant handle it when the braces arent aligned with one another, it makes debugging deeply nested code a PITA. At the end of the day i personally write more verbose code that takes up more space but is easier to read and understand. at the end of the day it all gets compiled the the same machine code 🙂

Bobby* *, January 12, 2011 at 11:01 am
Reply
Heh, I'm the opposite (obviously), I have trouble debugging the code if its not nested like that; but I find it interesting that you can see problems with the bracing, yet use if statements without a code block!
Tbh, the code I pasted is pretty verbose, if you understand the basics of an ANN, then you can read that very easily.

Daniel* *, January 12, 2011 at 2:18 pm
Reply
Hey,

How do you test the network following training?
So once you have your weights (I created a new csv file), what do you do with them to test that they're correct?

Alistair* *, January 19, 2011 at 9:04 pm
Reply
I dont understand your question. Your weights get calculated during the training and are validated against the training set and generalization sets.

You use the validation set after training to test the NN's accuracy on completely unseen data.

Bobby* *, January 22, 2011 at 9:30 am
Reply

I'm not entirely sure what you're using for the generalization/validation data sets, I can see you're extracting the information from the csv file, but my C++ is kind of rusty and I'm having trouble understanding exactly what fields you are using for each training set. Can you explain?

Ron* *, January 22, 2011 at 8:48 pm
Reply
hey ron,

each line of the CSV contains the input values (the values for the input nodes) as well as the required output values.

The training, generalization and validation sets all contain the same type of data. I.E. the input values and what the output should be. Basically you take the CSV file and seperate it into the three sets as described in the tut.

As for the actual data, I cannot remember what the format was in the test file I provided, you have to remember I wrote this tutorial around 3 years ago and cannot really remember finer details like that. Looking at the code it seems that the first 16 values are the input data values and the last 3 are the desired values for the output neurons. Hope this helps!

Bobby* *, January 22, 2011 at 10:16 pm
Reply
hi

can u please post a visual studio code for object recognition using neural network

aythin* *, January 28, 2011 at 7:26 pm
Reply
I dont have the time nor the desire to do that… the tutorials and code supplied should be more than enought to help you get started and to do it yourself… I am not in the habit of spoonfeeding people. If you dont understand something ask, but dont expect me to do your projects or work for you.

Bobby* *, January 30, 2011 at 8:25 pm
Reply
hi:)…………

hey your post is so helping a lot for my project in OCR.
i have a doubt in input data, i.e, "letter-recognition-2.csv". what is this file all about (i know its a feature vector but which feature??) ?? what is the input & target data in this file??

Santhosh R* *, February 2, 2011 at 5:45 pm
Reply
all of this is explained in the post (under Training Data and Training Problem), and I've answered this question several times in the comments already… Maybe you should read the post before trying to get the code to work!

Bobby* *, February 2, 2011 at 6:19 pm
Reply
This is just nitpicking really, in an otherwise awesome article…
…but you say "minima's".

Minimum is the singular form of the plural minima.
There is no such word as "minima's", and even if there was – the apostrophe " ' " either denotes possession or concatenation, such as "Bobby's guitar", or "they're" (they are).

Sorry! (and again, both your NN articles are brilliant)

***Ozzah** , February 14, 2011 at 6:02 am*
Reply
hello sir, its excellent but i have some doubts in this code if my no of Input neuron is 13 and no of hidden neuron is 2 and no of out put neuron is 1 then i'm getting 31 weights how? i cnat understand this and here tesing is implemented if i want to get a result for a particular value (prediction) of my inputs what i want to do. please i'm in urgent to work on it sir. please reply me or mail me kannanjayan2007@gmail.com . Thanks in advance.. Expecting your reply sir.

***KANNAN J** , February 22, 2011 at 3:42 pm*
Reply
I have seen a lot NN programs, this is a very nice one.

Just a question regarding to example. The example in the code has 3 outputs (16 inputs, 10 hidden neurons and 3 outputs), but in your tutorial you said it should 1.

Thanks,

San

***San** , February 26, 2011 at 3:17 am*
Reply
yeh the example code does some other classification, like i said in the earlier comments. I wrote it over 2 years ago and I honestly cant remember what exactly it was classifying 😦

It was never intended as a fully fledged "take&use" NN but rather as an example…

***Bobby** , February 26, 2011 at 6:29 am*
Reply
> I think you encoded 0, 1, 2, 3, …, 6, 7 with 3 binary codes. I'm thinking you probably grouped 26 letters (original data) into 8 category.
>
> Do you think the output can be just one neuron and it is an integer distributed from 1 to 26? Just like the input neurons.
>
> San
>
> ***San** , February 26, 2011 at 6:06 pm*
> ***Reply***

the output from a single neuron can only be a 1 or a 0, due to the clamping of the activation function. Now that I think about it, i think that the NN in the example classify all the vowels in seperate categories and then all the consonants in one categories.

If you wish to do per letter classification you will either need 26 output neurons or 5 output neurons and use binary to distinguish the category from the output neurons.

***Bobby** , February 26, 2011 at 6:38 pm*

Reply

I got it. It all makes sense now. Very nice work.

Thanks,

San

***San* , February 27, 2011 at 4:13 am**

Reply

Hello

I'd like to ask, if I discard the clamping function and modify counting accuraccy, is it possible to get as output real numbers between 0 to 1, cause I have the 5 intiger as input and output is distributed values from 0 to 1. Or are there other functions that I need to modify? till now I only reached 16% accuraccy so I don't know where the problem is.
Thank you very much.

***dominika* , March 10, 2011 at 1:05 pm**

Reply

the clamping function is what make the classification of unseen data possible. What the 1 or 0 represents is the neuron firing.

If you are having problems with accuracy, either try and improve the quality/pre-processing of your input data or start playing around with the NN parameters and architecture.

Put in a 100 hidden neurons and a really really small learning rate, remove momentum and see if the accuracy improves. If it doesnt I'd suggest examining your input data.

***Bobby* , March 10, 2011 at 4:19 pm**

a NN is a classifier, what are you trying to achieve? If I were you I would identify distinct categories for the values between 0 and 1. IE. 0.1 sized intervals and use an output neuron for each category!

***Bobby* , March 10, 2011 at 4:22 pm**

Thank you very much for answer, so if I understand it correctly, the output neuron can give me only 0 or 1? I'm trying to teach NN to count from 5 integers 1 output, but if I do it your way, I would have to have 7200 output neurons and I doubt that it is the best way. What if I translate the real output to integer number and than to binary number, would it work?

thank you very very much

Dominika

***dominika* , March 10, 2011 at 6:18 pm**

@dominika

why are you even using a neural netowrk for that task?! NNs are used to classify input data into various categories. in your case you want to have it distinguish a massive number of categories, each one only having a single possible entry. Basically you are converting a NN into a massive if statement:

if (1) return 1
else if (2) return 2

etc…

NN's are not suitable for your problem at all…

Bobby , March 10, 2011 at 6:34 pm
Reply

Hello

But I read in some papers about NN, that it can be used for this task also, and I'm using it cause There may be about 30000 cases, but I have only about 1000 cases of data to learn from and it is nonlinear system so I hoped NN can learn from these 1000 cases how to count it, and then I can use it for other also unseen cases.

dominika

dominika , March 10, 2011 at 7:09 pm
Reply

Regarding OO neural networks, I think I'd take it further and implement neural networks using a graph library, like Boost Graph Library or something similar. Just because a neural network is a graph, and that's how I roll 🙂

IMO, the OOP paradigm is very suitable for neural networks. Things like SVMs? I'd think the Functional paradigm would be more appropriate. And the Procedural paradigm isn't good for anything (j/k).

Ed , March 13, 2011 at 1:14 am
Reply

hi guys
i am doing my project in neural networks with feed forward back propagation algorithm in c-language.
i found difficulty in calculation of mean square error for three outputs and i dono how to train this algorithm in c-language ……………………?
any one please help me to complete my project on time….

suresh v , March 30, 2011 at 8:56 am
Reply

I suggest you visit a website such as StackOverflow for programmer specific questions…
http://stackoverflow.com/

Daniel , March 31, 2011 at 4:38 am
Reply

Thank you bobby, and I should say, this post from 2008 and it keeps its update. I'm in computer engineering 4th year. This semester I'm takin ANN course and this post really helps me. What I saw in lecture , you implemeted same things…

See you at new posts.. 🙂 byby

gokhan koylu , April 3, 2011 at 2:58 pm
Reply

I found your tutorial to be amazing. After taking a class in AI, and extensive research, it was finally your tutorial that gave me the necessary "A Ha!" moment that until then had eluded me.

I am trying to use a Neural Network to classify data inputs of around 42000 points. I am required to use MATLAB (which in and of itself is a performance drain) and I programmed in a OO method. (I know, I know… But honestly, I can't think of a better way to keep a code modular, and to enable the best approach to debugging. Sorry…)

My problem is, my code appears to take an extremely long time to run. The initialization of the synapses between the inputs and the hidden layer (Haven't even gone past five nodes for the hidden layer) takes what feels to be about ten minutes. I haven't even moved through the length of time that it takes for the initial feedforward. My question is, is this to be expected for inputs of this size? I have heard of parsing techniques such as PCA or a t-test threshold to identify nodes of interest and then just use those, but this seems like it could lose some of the data that makes Neural Networks so powerful. Any thoughts or suggestions?

***Daniel P.** , July 1, 2011 at 12:04 am*
Reply
I think in your case matlab is the performance problem. Matlab is very limited and slow. My dad (a math's prof) implemented an image processing algorithm in matlab but had some severe performance issues as well as limitations. For example the max image he could input was 640×480 and that took around 15~20minutes to process. I rewrote the same algorithm in C# and was able to process 2MP images in around 5 seconds. Matlab has absolutely massive performance overheads that make it unuseable for pretty much anything past a quick proof of concept.

I dont know much about PCA or anything like that but what i can suggest is try to preprocess your input data. 42000 input points is a huge number and there must be some way to reduce the dimensions of the input data vector. In my experience pre-processing has the greatest effect on NN performance.

***Bobby** , July 1, 2011 at 10:45 pm*
Reply
Hello Bobby, I have a quick question. What if I want to predict the output of a continuous function(y=x^2 + Sin(x)) instead of binary classification problem. What changes i need to do in the logic of your code. Thanks.

***Md Kashif** , July 18, 2011 at 4:00 pm*
Reply
I dont see the point of predicting the output of a known function? Just evaluate it to get the exact answer. If you are looking to optimize the function then rather look at particle swarm techniques.

Neural nets are primarily used for data classification, i.e. figuring out what category that data belongs to.

***Bobby** , July 18, 2011 at 4:11 pm*
Reply
Hi Bobby,
Thanks for your great Blog.
I am new in NN and studied your post here. I have very simple question:
I made one NN model include 3 inputs , 1 output , 1 hidden layer. Now I have 6 weights.
My question is how use this weight for using to predict my new input and test my model?
Regards,

rmforexReza ***, July 25, 2011 at 11:03 am***
Reply

Well you have to train your network on some sort of input data. There are various training methods, the one I describe here is just simple backpropagation of the errors. NNs are not useful unless you have a good data set on which to train them as well as making sure that the dataset can actually be categorized wihtout too much overlap.

I think you question also signifies that you havent read through the post in detail as everything regarding the weights and training is discussed in detail.

Bobby ***, July 25, 2011 at 3:52 pm***
Reply

Hi Bobby

Thanks for a great piece of code and an excellent tutorial. I've actually converted it into C#, mainly because my projects are .net. I'll share it if anyone is interested. I use it to predict horse racing results; early days but so far very good results!

owen gunter ***, August 23, 2011 at 11:00 pm***
Reply

Ok, I'll bite.

What was the criteria and information you fed the ANN to choose the best 'horse', and what was your training data.
And what do you mean, "good results" 😛

Daniel ***, August 24, 2011 at 2:31 am***
Reply

I have a basic formula that calculates the horses performance over the last 3 races, I also use (C)ourse, (D)istance, CD, trainer and jockey success ratio over last 14 days. All of this is gathered automatically from the racing websites so it's very automated, Basically I store the data on a daily basis, make a prediction and when the results come in the current days race data is the future training data (so the training data is 'refreshed' on a daily basis). I repeat/cycle this for 60 days so as the going changes the database can take this into consideration.

The training database has been going since 10 August and has 18000 records. I have several 'systems' I am evaluating in terms of betting/profitability strategy to determine which is the most profitable. But based on ANN the results are:
August: 2 wins 10 races +30.5pts
Sept: 1 win 5 races +22.0pts
* based on SP upto 4 September

owen gunter ***, September 4, 2011 at 11:57 pm***

Hi Bobby,

Thanks for the great tutorial.

I ran your code and it worked successfully.

I couldn't really understand how to interpret the results, like i want to know that I have given the input as say '1' and the NN output now is '1' or '2'.

Also the i/p and o/p is difficult to understand.

The problem I am working on is, I have a set of observed values of co-ordinates x,y and I have corresponding actual co-ordinates X,Y. Now I have to design a NN which can give me an input/output machine to approximate the error or an equation of error surface.

I have Following questions,
1> I have 1 input co-ordinate pair corresponding 1 output co-ordinate pair and 100 such pairs. Now here the output is also changing, so there will be just one epoch to adjust weights
for a given pair. Will that be enough to train the network.

2> What approach can I have to use your code.

I know this sounds like you spoon feeding me, but I will appreciate any help you can provide or direction you can give me.

Cheers,

Paras

Paras ***, September 5, 2011 at 11:59 am***
Reply
I'm kind of wondering a few things. I'm not sure if you will read this message or not, but if you do and take the time to answer, my thanks.

I find it hard to understand the concept of NN, but I'm trying, and I have a few questions:
1) Why is the shuffle_vector commented in the source code?
2) Why are there 16 patterns and 3 targets when you said there would only be finding the difference between a and other letters, and vowels.
3) Kinda the same the 2) one, why only 1 output when there are 2 variants?

Wituri ***, September 24, 2011 at 11:24 am***
Reply
1) That's a mistake 🙂 I probably was doing some testing and left it commented out. I cant really remember, the code is nearly 3 years old at this stage.
2/3) There are 16 inputs as the training set i used defines letters as 16 float numbers. The outputs are binary i.e. 1 or 0, on or off… So each output defines a category. is it A? is it a vowel? etc…

Bobby ***, September 25, 2011 at 11:08 pm***
Reply
Thanks for your answer mate.
I found this a pretty useful and easy to comprehend code around NNs.

Wituri ***, September 27, 2011 at 7:15 am***
Thank you for such a great tutorial!

I wanted to test the program on another csv file that has just 6 patterns. It's a very simple pattern of a single input and 8 outputs. The 8-digit output is the binary representation of the square of the input integer.

So, I ran the program after making a few changes. I used 10 hidden neurons but that barely made a difference. The network never trains! It says training complete in the first step itself.
Can you explain where I am going wrong?

Thank you in advance.

Abhipray* , *September 25, 2011 at 10:07 pm
Reply

i think a training set with only 6 entries is too small further more NN are primarily used for classification of data which can be easily seperable not for function fitting. If you use a NN for a classification problem and with a large enough data set then you will see some benefit from the technique.

Bobby* , *September 25, 2011 at 11:12 pm
Reply

Thank you for explaining but does that mean there is any minimum required for the number of data patterns? or is the minimum experimental?

I am working on using NN to train a robot to understand my voice and to scan and classify alphabets(OCR). I just want to know how big my experimental data set should be to get a good level of accuracy?
Also, are there any guidelines for choosing the number of hidden neurons?

Abhipray* , *September 25, 2011 at 11:35 pm
To be honest NNs are very limited in their uses. I'm not very sure whether they would be the best option for your specific problem. There are various other types of classifiers out there that may potentially be better for your application e.g. support vector machines. The thing with these kinds of problems is that there is no real right answer to the problem, you need to try several approaches and see what works.

Choosing the number of hidden neurons is also largely trial and error. If you knew what the distribution of your dataset looked like you could see how many seperating hyperplanes you need and allocate that many hidden neurons. In practice, the dataset is often unknown and so you need to guess at the number of hidden neurons necessary. I'd say pick a random number, train with that many hidden neurons, then increase and decrease the number and retrain, that should give you a very rough indication if adding more hidden neurons helps or makes matters worse and you can go on from there.

Bobby* , *September 26, 2011 at 12:20 am
Perhaps a bit intense, but I started off this summer using Bobby's blog to learn about Neural Networks, so perhaps you can start inspecting further research:

http://yann.lecun.com/exdb/publis/

Dr. LeCun created something called LeNet (I believe, look through his papers circa 1990) which used a combination of statistical parsing and ANNs to perform OCR for banks on checks written. You may find this a place to get an idea of what you are trying to do may require. (I personally like his Helpful Tips paper which centered on implementing tanh as your sigmoid and using the second derivatives rather than the extrema to train your ANN [http://yann.lecun.com/exdb/publis/#lecun-98b] as well as the implementation of Boosting which I believe Bobby describes in the comments somewhere.)

Bobby, I don't think I ever got a chance to thank you for the assistance this website provided in getting me rolling in applying ANNs in statistical analysis, but Thank you.

djpeirano* *, September 26, 2011 at 12:32 am
Reply

Pingback: C++ Back Propagation Neural Network Code v2 « Taking Initiative: Bobby Anguelov's Blog

Hey, is there an SVN or Git repo for this? I made some changes and added a CMakeLists file to build it in Linux and wanted to post you my changes. All I really did was add some includes that Linux required (mainly rand functions). I also changed void main to int main per cpp standards (ISO that is).

JimBlogger* *, October 3, 2011 at 4:44 pm
Reply

Hi
We want use your library in our project, but no matter which parameters we change we get poor output (maximum was 20%). Theme of our project is recognition of recorded speach samples (digits), we have input vector long of 455 double type numbers, and target digit (int 0-9). Input file contains around 830 vectors, that gives 456 columns and 830 rows. We decided to put 455, 200, 10 neurons in layers, but we tried many more combinations. Where should we look for mistake?

M&K* *, October 28, 2011 at 3:13 pm
Reply

M&K, have you tried training with different layer layout? I did an NFL prediction tool and had to modify hidden layers to get my error rate down. I went from a 30% test result down to about 15%…..still not great but it might help in your situation.

Jim@ExampleCode.org* *, October 28, 2011 at 3:17 pm
Reply

Great delivery. Sound arguments. Keep up the great effort.

Search Engine Optimization Tips* *, February 22, 2012 at 4:17 pm
Reply

So what is the problem with doing this without back propagation? Allowing the network to learn solely through inputs alone. That's how we do it, why can't a artificial neural network do it?

Ryan Smith* *, February 26, 2012 at 9:50 pm
Reply

Back propagation is how the net is updated. The neural net wouldn't learn without it; it'd just be static (after a neural net is trained, you'd stop back propagation). This is a classic ANN which were meant to just work, and not simulate an actual neural network. There are other kinds of ANNs, and other update policies. Restricted Boltzmann Machines are recurrent neural networks that perform very well on some problems, and don't use back propagation. HMAX is a neural network that is modeled after the visual cortex.

Me* *, February 27, 2012 at 2:03 am
Reply

I've trained the NN on my datasets and now I want to use the weights of the trained NN on testing on other user given data? (I'm using NN to do regression not clssification, so is you model suitable for regression as well?)

mario , April 29, 2012 at 1:58 pm
Reply

How odd-
I was messing about with the code, and changed the number of input nodes to 24 just out of monkey curiousity. Sure didn't expect:

```
Data File Read Complete >> Patterns Loaded: 20000

Neural Network Training Starting:
==========================================================================
LR: 0.01, Momentum: 0.8, Max Epochs: 500
24 Input Neurons, 16 Hidden Neurons, 1 Output Neurons
==========================================================================

Epoch :0 TSet Acc:5.225%, MSE: 0.142681 GSet Acc:10.025%, MSE: 0.101812
Epoch :1 TSet Acc:58.525%, MSE: 0.0364501 GSet Acc:91.35%, MSE: 0.00485578
Epoch :2 TSet Acc:96.4583%, MSE: 0.00301354 GSet Acc:99.475%, MSE: 0.00234604
Epoch :3 TSet Acc:99.6333%, MSE: 0.00163814 GSet Acc:99.8%, MSE: 0.00170634

Training Complete!!! – > Elapsed Epochs: 500
Validation Set Accuracy: 99.975
Validation Set MSE: 6.97053e-005

— END OF PROGRAM —
```

total changes from d/l:
changed line 25 of main.cpp to
neuralNetwork nn(24, 16, 1);

sort of expected it'd blow up or perform identically to the install settings.

pathwaystoknowledge , May 3, 2012 at 8:10 pm
Reply

obviously I've also changed the hidden layer to 16 here, but behavior's the same with hidden layer at 8

pathwaystoknowledge , May 3, 2012 at 8:14 pm
Ah! slowly the light bulb starts to go on.

I tried setting max epoch to 250, desired accuracy to 98, then looping by 4's over number of input and hidden nodes. output is #input nodes, #hidden nodes, epochs to termination

```
Data File Read Complete >> Patterns Loaded: 20000
4,4,250
4,8,250
4,12,250
4,16,250
4,20,250
4,24,250
8,4,250
8,8,250
```

```
8,12,250
8,16,250
8,20,250
8,24,250
12,4,250
12,8,250
12,12,250
12,16,250
12,20,250
12,24,250
16,4,250
16,8,250
16,12,250
16,16,250
16,20,250
16,24,250
20,4,2
20,8,2
20,12,2
20,16,2
20,20,2
20,24,2
24,4,2
24,8,2
24,12,2
24,16,2
24,20,2
24,24,2
— END OF PROGRAM —
```

(this, incidentally, takes a really freaking long time – 90 minutes on my machine)

Nothing like data 8cD

pathwaystoknowledge , May 3, 2012 at 9:30 pm

thank you Bobby, I've add into my project for an CRS(Coin Recognition System), for a pre grade assignment I've given you the credits regards from Peru

enrique , July 30, 2012 at 4:21 am

Reply

Hello Bobby,
Im a novice to ANN and using ur code/explanation to understand the BPNN. My question is: after training the network using the alphabet example and saving the weights identified, how do I use it to classify/predict the class of an "unknown/unlabeled" alphabet attribute?
Detail:
training data: since I have 26 alphabets, I presume 26 classes which makes my number of outputs 5 (2pow5=32 classes) so for instance letter "a" = 00000, "b" = 00001, "c" = 00010, … "z" = 11010. The outputs (0,0,0,0) are added to the end of each occurance of letter "a" in the training data for example.
So after training I get a set of weighted values (which I save in file) and the result (Epoch,Training Set Accuracy, Generalization Set Accuracy,Training Set MSE, Generalization Set MSE) of 250 epochs is:

(Epoch,Training Set Accuracy, Generalization Set Accuracy,Training Set MSE, Generalization Set MSE)
250, 66.8306, 64.375, 0.040653, 0.0474078.

Now how do I use this weighted values and/or Epoch, TSA, GSA, TSMSE, GSMSE, for identifying the class an unlabeled unknown alphabet say “h”?

Any person’s help in explanation is highly appreciated 😉 thanks

***luhfluh** , July 30, 2012 at 11:42 pm*
Reply

if you’ve trained the network properly, then the weights at the end of the training should be used with the new input. i.e. in a really simplified version (3 neurons with two inputs, and using a tanh sigmoid instead of the logistic function): the network equation would look like output = tanh(tanh(input1*weightn11+input2*weightn12) + tanh(input1*weightn21+input2*weightn2)). So to “run” the network on any new input, just replace the input(s) with the new input (scaled to between 0 and 1) and use the most up to date weights (i personally prefer evolutionary algorithms (just add random noise to the weight, if the fitness function is better, keep the new weight, else reject it. rinse repeat) to BP as BP is so slow and prone to local minima).

***nick emblow** , August 27, 2012 at 2:11 pm*
Reply

sorry i must correct myself, i left out the weights for the final activation: the network sum should look like this: output = tanh(weightn31*tanh(input1*weightn11+input2*weightn12) + weightn32*tanh(input1*weightn21+input2*weightn2)).

***nick emblow** , August 27, 2012 at 2:14 pm*

How you saved weights in file? Please reply,

***MaX** , April 1, 2013 at 1:45 pm*
Reply

Hi everyone, This is code for training the network so I need code for the testing data to classify input data. My classes is 2 for skin and non-skin segment classification. Who can suggest to do it.

***pun_tgif@hotmail.com** , September 2, 2012 at 5:18 pm*
Reply

“the simplest solution is usually the best solution” cool & “idiot professors that know the theory but can’t code their way out of a paper bag” man you absolutely right i have seen many of them. Great tutorial.

***Shan** , September 15, 2012 at 6:56 am*
Reply

I’ve only recently taken an interest in machine learning and really appreciate the time you’ve taken to give such a thorough intro. I don’t agree with the gripes about OOP, though. Much of the allure to object oriented design is the efficiency of messaging between objects. A well designed structure will run very fast; in many cases good inheritance can make the execution even faster than a procedural solution by taking the conditionals out of run time. The overhead for method calls and stack frames can be negligible, especially if the training happens at once and only loads a reference to weights for execution. And if you’re for minimal code, there’s no beating extending an already existing class.

Finally, I know there’s the “if it was hard to write it should be hard to read” camp, and that’s unavoidable sometimes, but in general OOD is easier to read and I think that’s important. You blame the university mentality for it, but this factor is even more important in industry. If you ever want to get promoted and move on there has to be someone else who can make sense of what you've written! You can't be the only one able to maintain it.

Just my thoughts, I really like appreciate the help you've given here. I'm going to start working on some mad science (probably mad, object-oriented science). Thank you.

steve , September 15, 2012 at 9:49 am
Reply
I appreciate the article, and learned a lot from it. The data sets in your updated version 2 code however, are a bit confusing. In version one, you had three different spread sheets that had binary values if it was a letter 'a', or if it was a vowel. The updated code has a single spreadsheet with the 16 attributes, and then three additional columns.of 1's and 0's. These columns do not seem to fit the "is it 'a'", or the "is it a vowel" criteria that you mention in your post. For instance, Row one, which corresponds to 'T' has the three columns as 1, 1, and 0. Row two, which corresponds to the letter 'I", also has 1,1, and 0. Since these last three rows are not differentiating the letters T with I, what do they mean?

Calvin , January 21, 2013 at 11:07 pm
Reply
"Whether the separate classes can be approximated by a function isn't guaranteed. I don't have the literature handy but my professor said there is academic literature that proves that you should never need more than a single hidden layer in any NN. "
How old is he? That literatur is from 1980 and is true only for linear function. In late 80, on MIT some guys prove it is not true for sigmoidal function(and other function).

revvv , March 14, 2013 at 11:20 am
Reply
Updated NN code not working with codeblocks on windows

max , March 19, 2013 at 2:14 pm
Reply
Friend function is showing error!!

max , March 19, 2013 at 2:15 pm
Reply
Thanks Bobby. I want to ask can we store the weights after the termination of the program ? I tried to write them into file, but i realize that i am not a great coder. Can you help in here? Thanks in anticipation.

MaX , April 1, 2013 at 1:35 pm
Reply
Hi Bobby,

first, thanks for a great tutorial. Super clear and helpful.

I was able to train the network and get the "weights" file but I'm not sure how to feed forward a new set of data.

* Is this as simple as dropping an excel file in the "Resource Files" folder in the visual studio project?
* Also, once I have the file of data i'd like to feed forward, what do I do next? execute the

neuralNetwork.ccp file?

Forgive me if those are simple questions. I have a good understanding of NN theory and I've written OO programs in C++ before but never in Visual Studio.

Thank you in advance.

Fadi.

fadi , May 23, 2013 at 4:36 pm
Reply
Bobby, thanks for a great tutorial and sample code!

I have noticed only one issue in your implementation. I believe you misunderstand stochastic backprop as opposed to online backprop. In both cases, the weights are updated immediately with each training sample. However they are not synonymous.

In online backprop, the training patterns are presented once and only once – they are not kept in memory. In stochastic backprop, the samples are drawn randomly from the training set (hence the name "stochastic"). Your code always loops through the training samples in the same sequence. This can result in sub-optimal training of the network due to peculiarities of the training sequence. For example, suppose the last few samples are outliers. These will tend to dominate the training per epoch since they always update last. I have been warned by experienced researchers to never train a network using repeated sequential patterns in this manner. By instead drawing random samples, the training data becomes a random variable.

Your code is correct for batch backprop. The batch algorithm works because all the weight deltas are added together in one update and hence are equally weighted.

Cheers!

Brian

Brian , August 1, 2013 at 9:37 pm
Reply
Reblogged this on i like computers.

BORN2CODE , September 19, 2013 at 3:19 pm
Reply
Hey Bobby,

Thanks for this great, definitely informative and real-world-useful BPNN tutorials.

I'm currently doing a small BPNN project (3653 instances x 19 features) which cannot provide acceptable results, given the stopping conditions you've suggested. With series of review on my code and properly-normalized dataset which yielded OK, I'm now inclined of suspecting the dataset itself for containing irrelevant features.

My query is –> are all features which I believe useful in the process be included in the dataset? IS there such a thing which we have to validate, as in scientifically check, each and every feature with respect to the dependent feature before including them into the dataset?

Thanks in advance.

Noel ***, October 10, 2013 at 3:18 am***
Reply
great article.it,s really help me a lot.thanks.

neelam ojha ***, October 24, 2013 at 2:24 pm***
Reply
Fantastic goods from you, man. I've understand your stuff previous
to and you are just extremely excellent. I actually like
what you've acquired here, certainly like what you're stating and the way in which you say it.
You make it enjoyable and you still take care of to keep it smart.
I can not wait to read much more from you. This is really a tremendous web site.

Tyson Gugliuzza ***, December 26, 2013 at 6:57 pm***
Reply
Hi Bobby, do you have a license to use your neural network implementation in commercial applications? I cannot find it in the source zip.
Thanks,
Vladimir

Vladimir Parfenov ***, April 9, 2014 at 10:44 pm***
Reply
Today, I went to tthe beachfront with myy kids. I found a
sea shell and gave it to my 4 year old daughter and said
"You can hear the ocean if you put this to your ear." She put the shell to her eear and screamed.
There was a hermit crab inside and it pinched heer
ear. She never wans to go back! LoL I know this is totallyy off topic but I had to tell
someone!

Le Petit Chaperon Vert ***, May 4, 2014 at 9:26 pm***
Reply
I'm really enjoying the theme/design of your web site.
Do you ever run into any browser compatibility problems?
A number of my blog readers have complained about my blog not working correctly
in Explorer but looks great in Firefox. Do
you have any recommendations to help fix this issue?

Olivia(TM) 2013 Wall (calendar) download mobi ***, June 11, 2014 at 5:03 pm***
Reply
Wonderful beat ! I would like to apprentice at the same time as you amend your website, how could i subscribe for a
blog website? The account helped me a acceptable deal.
I had been tiny bit acquainted of this your broadcast provided vivid transparent concept

PCI Express System Architecture download mobi ***, June 13, 2014 at 12:51 am***
Reply
This website was… how do you say it? Relevant!! Finally I've found something that helped
me. Appreciate it!

A Guide to Financial and Commercial Property Management using QuickBooks (Manage Properties with QuickBooks) (Manage Properties with QuickBooks) download pdf , June 13, 2014 at 9:59 pm
Reply
Magnificent goods from you, man. I've understand your stuff previous to and you are just too wonderful.
I really like what you've acquired here, certainly like what you are stating and the way in which you say it.

You make it enjoyable and you still take care of to keep it sensible.
I can not wait to read far more from you. This is really a tremendous site.

The Breakfast Revolution: Recipes From Outside the Cereal Box pdf , June 14, 2014 at 10:19 am
Reply
Heya i'm for the first time here. I came across this board and I find It truly useful & it helped me out a lot. I hope to give something back and aid others like you helped me.

Look At This , July 19, 2014 at 4:23 am
Reply
Pingback: Stock market live today - binary options neural network
how can i compile this code in my c++ compiler or c compiler.
i got confused. Are there any shortcut to compile this code easily??
thanks in advance..

niraj , October 12, 2014 at 2:56 pm
Reply
Line 297 neuralNetwork.cpp, in the comment you put:"//set weights between input and hidden" while it should be "//set weights between hidden and output".
I know it's obvious but, you know…

Chri , December 3, 2014 at 7:47 pm
Reply
I ran the downloaded code. After 150 epochs it only achieves 69% accuracy. This is not good enough to be useful. How can this be increased.

Mike , January 2, 2015 at 6:26 pm
Reply
can we apply this code for shape detection??? how can we create other csv file for different shape??

Parul , March 7, 2015 at 12:10 pm
Reply
Do you have any code for recognize shapes using BPN???

Pearl , March 13, 2015 at 8:17 am
Reply
OOP is another paradigm for programming, neither better nor worst than the other ones (say "the procedural" for example). Two things:

1.- OOP (clases, inheritance and encapsulation) doesn't waste (uses) any extra memory. Even you can tweak it for best results (mostly with the constructors).

2.- OOP is a nicer view of a system. In the context of OOP you can think of neurons as an independant entity, which executes a job, given some inputs. Ultimately that's a neuron, isn't it? I guess you faced a bloated code in which the very important things were obfuscated for someone who wanted to seem smart. But please, don't blame the OOP.

Once I've said that in favor of the OOP, congratulations for your post, this is really amazing. Keep posting!!

fjrg76 , March 18, 2015 at 6:41 am
Reply
Finally someone who speaks some sense on OOP, I code computer games and totally agree with the needless OOP rant!

Danny , April 27, 2015 at 11:43 am
Reply
stry t ers era affect powerful fl ing will tinggal article power s

Tempat Kursus seo online , June 13, 2015 at 3:29 am
Reply
iles web efazio r there address e schedule mak usedn yahoo.com origina kursus website terbaik tempat kursus komputer

kursus website terbaik , June 17, 2015 at 6:52 am
Reply
Hi,
I plan to make use of your code for data of electronic devices, where i get data along with weights, i have 4 devices along with weight for each hour. Couple of questions

1. I see input data is of size 16 and output data is of size 3. Why we need 3 values of outputs ?
2. Since these NI and NO are just variables, instead of passing 16 and 3. I can pass 4 for NI and 1 for NO ?

Regards Piyush

piyush , June 28, 2015 at 4:10 am
Reply
Just wanted to send a huge thanks for the tutorial and code! I'm starting to learn more about NN, particularly using them for natural language processing, and it's extremely useful to read through your tutorial w/ matching source code. Thanks!

adamwulf , July 3, 2015 at 8:48 am
Reply
Somos el servicio tecnico de electrodomésticos de confianza en la Comunidad de Madrid.

Deandrea Pursifull , December 13, 2015 at 3:31 pm
Reply
Hello Bobby's I am running out of worry now, please help!!! me out on the source code and materials on authorship attribution system using neural network(The Algorithm is Back propagation)

orah123 , December 20, 2015 at 5:34 pm
Reply
Hi Bobby, totally impressed by your approach (first version even). And as for the OO "war" I totally

agree with your stance, unecessary things are unecessary and a burden only. Structure and clearity are fundamentals though. In my limited view, which for some application areas may benefit from a good OO support, especially in large systems with numerous fairly separable areas. The further away from HW, high speed and numbercrunching you get the more I suppose you want to go that way.
tnx again
Ohh, one question; have you made any rules of thumb/formulas for how you system scales in terms of Ops (with diff.# of in/out/hidden)(Im a bit lazy 🙂 ?

/Georg /still on my way into using your code

georg , January 5, 2016 at 7:01 pm
Reply
friend ,i have designed a c++ code for optimization using quad tree and the robot avoids the obstacle and go to the source to destination ,but i want to use NEURAL NETWOR here to further optimise it ,how can i do ,please sugest

Ramakantachoudhury , February 23, 2016 at 7:49 am
Reply
What compiler used to run this code?

shah , August 2, 2016 at 8:15 pm
Reply
I just updated the version one with ability to use for multi-class classification and dynamically changing the number of hidden layer and their corresponding neuron number.
So please let me know if I should cite your code or any existing article anywhere.
I will probably upload it til next week on the github.
Tell how to reference your code just in case.

Hadi SaadatDoorabi , August 20, 2016 at 5:02 pm
Reply

> it's the github link for the upgraded version:
> https://github.com/hadisaadat/Simple-Neural-Network.git
>
> ***Hadi SaadatDoorabi , August 20, 2016 at 6:43 pm***
> ***Reply***

Sorry for my noob question . I already took a look at the source code and I do not understand why 3 "outputs" ? How could we represent all letters with 3 outputs ?

PoP , October 26, 2016 at 8:41 am
Reply
Pingback: [Solved]: How do you determine the inputs to a neural network? – Ignou Group
Teacher Bobby? EP

garguntia , May 13, 2017 at 3:49 pm
Reply