Image Recognition Project

H446/03 Programming Project

By Vilius Radavicius

Centre number: 54543
Candidate num: 5694

Contents

Analysis

About this project

This programming project will be an Image recognition program in which the neural net should recognise some basic items and objects within the scene and then it should be labelled, to show and prove that the program actually works.

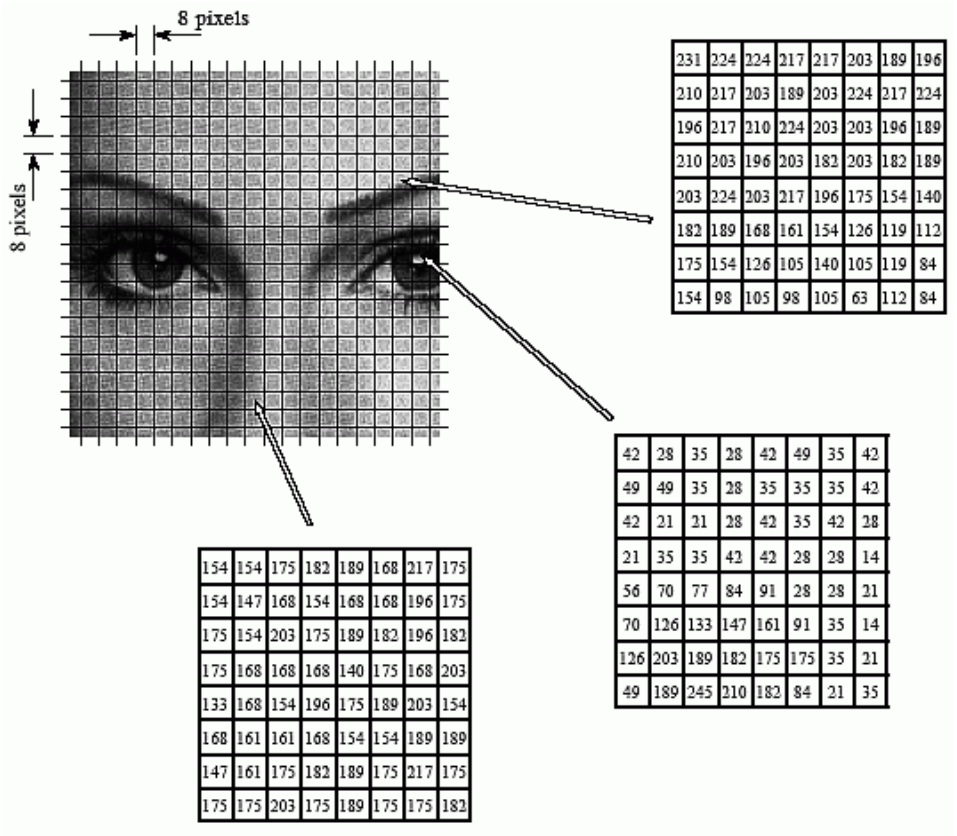
"Image recognition, in the context of machine vision, is the ability of software to identify objects, places, people, writing and actions in images. Computers can use machine vision technologies in combination with a camera and artificial intelligence software to achieve image recognition" - https://searchenterpriseai.techtarget.com/definition/image-recognition

This will be a complicated project and the reason for this is because I will need to know how neural networks work and I would also need the program to abstract the image and turn each pixel into a number so the neural net could work with it, but before I could start working on the neural net I need the program to convert each pixel into an int

One massive problem with this project is that the inputs for my neural net will determine on the size of an image, so I could either make it so the inputs change in which require some serious amount of work or make every image the same size by automatically resizing it, which would be much easier and require less work compared to the 1st option.

Identifying problems

One of the problems for this project would be to make the image recognisable by the neural net, so abstracting the image and turning it into black and white and then turning it into integer numbers between 0 and 255



As you can see this is how my image abstractor will work, it will grayscale the image and convert each pixel into an integer between 0 and 255, 0 meaning black and 255 meaning white, but even this will make the image recognition very inefficient and would require vast number of processing power and a lot of time to train the program.

Stakeholders

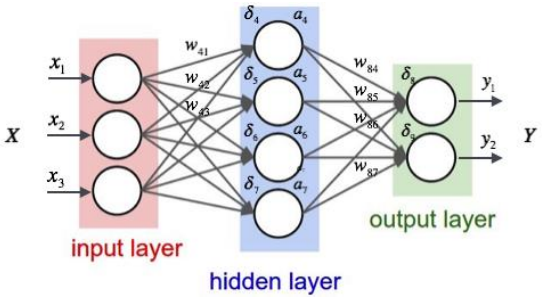
The stakeholders for my project will be **Byron Piper** and **Jordan Dillon** because they're very intrigued by this project, Byron has some experience in machine learning and is mostly interested in the machine learning part of the program whilst Jordan is mostly interested in the image recognition and he would like to use it for his other projects.

One of my stakeholders demanded "**real time image recognition**" to be implemented in my program, and I will do my best to fulfill the request once the program can recognise objects and people from images.

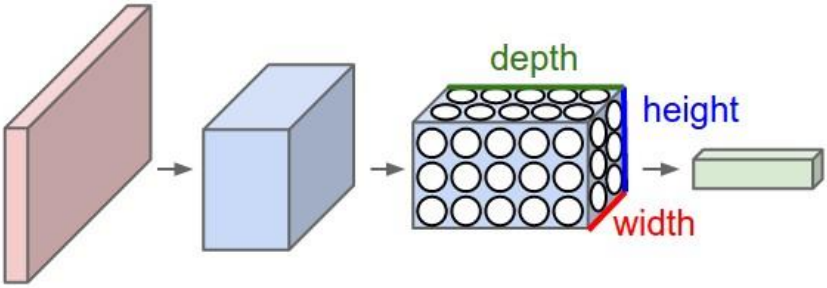
Research

I looked up how image recognition software work and what i've found was interesting, instead of using normal neural networks to train the image recognition it used something called "**convolutional neural networks**" - http://cs231n.github.io/convolutional-networks/
And instead of using a 1d array to store data I could use a 3d array and efficiently store data.

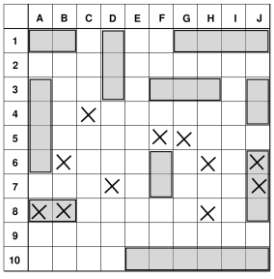
How normal neural networks work:



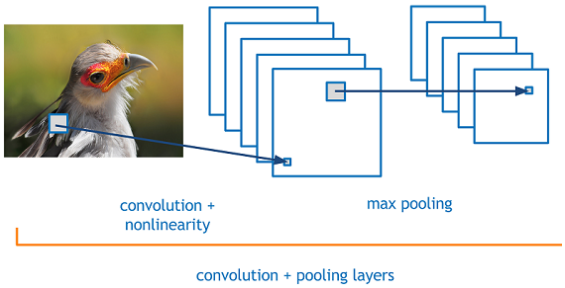
As you can see from the diagram a simple neural network has a single 1d array for input, hidden layer and the output layer, we could simply fetch each pixel from the image and feed it into this neural net but we covered this problem back in **Identifying problems.**



As you can see the input 3d array will have width and height so we could store the pixels of the image and we could easily select a pixel from the 3d array just like in a game of battleships, but for the depth we have a little issue, we could either store

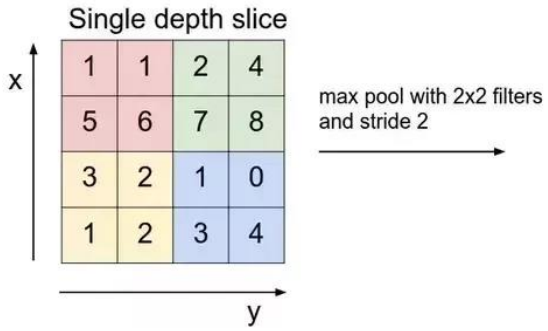


The RGB value by converting it into hex and then into int eg:
White = RGB(255,255,255) Hex(#FFFFFF) int(15463490)
Which would be a little more effort and will add in extra work for the program to do.
The other solution is to make the image grayscale and to store the value as a single number eg:
RGB(x,x,x) x= the value you store in the 3d array, so if x= 255 then the pixel will be white because RGB(255,255,255) is white.



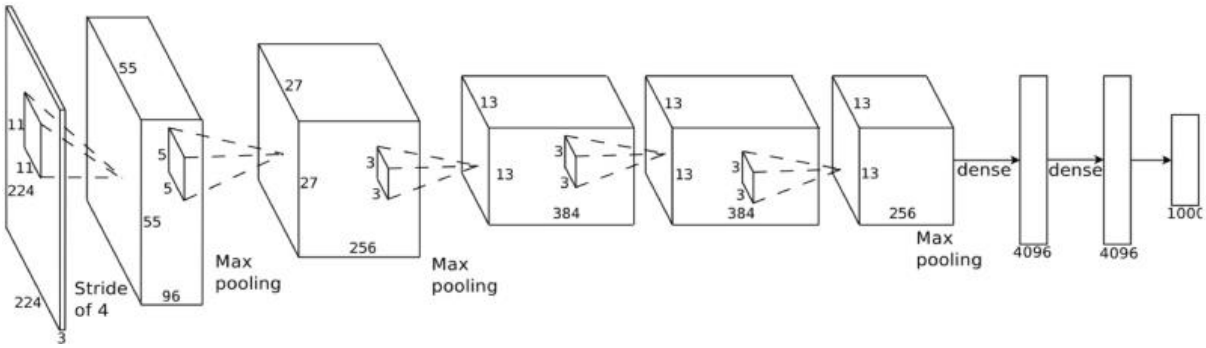
One of the things I don't understand about CNN image recognition is the multiple layers in max pooling, from the visual example of how CNN works it seems like it's reducing the resolution of an image and by my guess the multiple layers might indicate the type of filters, but i'll need to do some research in order for me to know for sure.

So after some research I found a very useful article on how max pooling works: https://www.quora.com/What-is-max-pooling-in-convolutional-neural-networks



I'll have to select the first 4 pixels and select the pixel with the biggest value using the max function, for example if we have 4 pixels 1,1,5,6 then 6 would be selected and added to the new bitmap so this transformed the 4 pixels into just 1. That's how the max pooling will work, now that we figured that out we would need to see how small should we make the images.

So i've done some research and I found this website https://medium.com/@ageitgey/machine-learning-is-fun-part-3-deep-learning-and-convolutional-neural-networks-f40359318721(1) and they had this diagram:



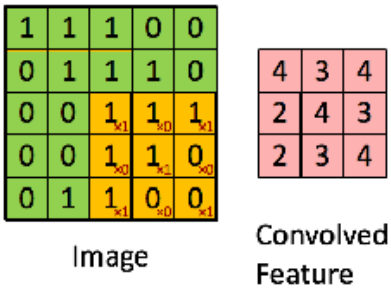
And if you can see they start with 224*224 image and end up with a 13*13 image while this still does contain a lot of inputs (169 inputs if we do the math) this still drastically reduces a lot of inputs and could speed up the whole process of recognising the image.

Another different way of doing this image abstraction process is by splitting up the image into smaller pieces, see link (1) and their idea's interesting because instead of

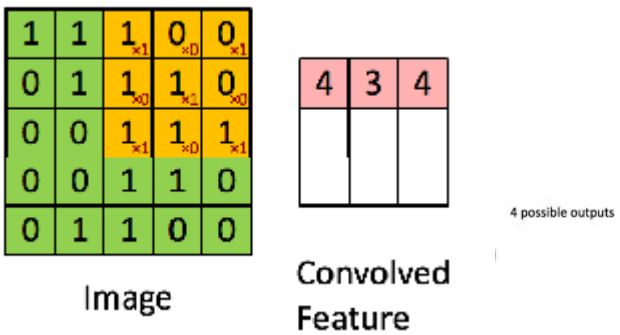
using a single neural network they used two one to pick up interesting features from the image and the other for recognising images, while this method could be very useful training another neural network would be time consuming.

So now that we know how max pooling works and how small the images are going to be we now need to find out what the convolution part of cnn image recognition does, and luckily enough I found another website that tells us how it all works: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

In a nutshell convolution works similarly to the max pooling, it has a small grid that scans through the whole image and creates another bitmap image, but instead of making the image smaller by reducing the pixels it will multiply the filter's pixel values to the pixel the filter's currently hovering over and this will make a totally unique image result and this would be useful to detect edges and giving the program a better understanding of the image so the program could be more accurate and prevent anomalies.

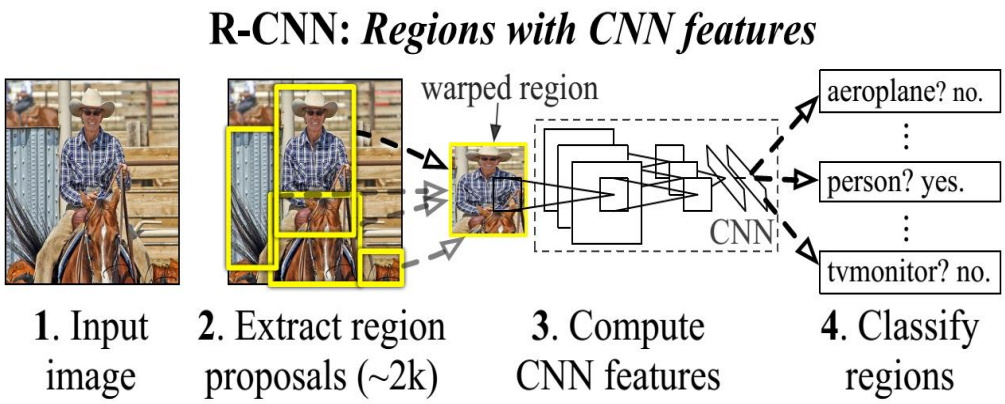


The abstracted image will narrow down to couple of outputs in which should represent the possibility of the object, 1 meaning 100% sure that the image is a specific item/object and 0 meaning 0% chance of the image containing that item, so we could set a specific output to represent the likelihood of that item existing in the image as we can see from the diagram above the output neuron(boat) value's 0.94 therefore the likelihood of a boat existing in the image is very likely.

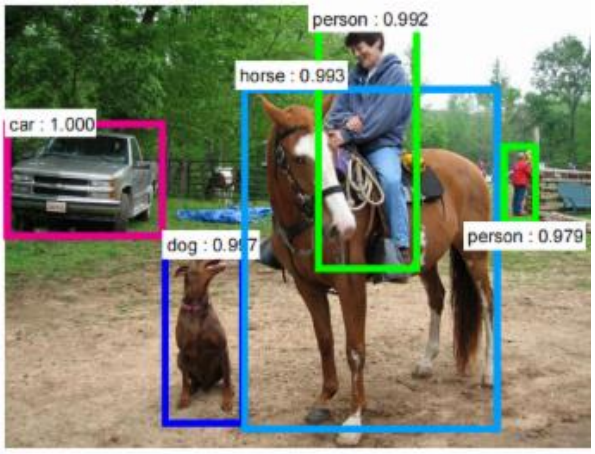


Box feature(object detection)

The box feature is a very important feature within the image recognition software, it's used to identify multiple real world objects which increases the chances of an accurate prediction, the reason for why the box feature make the image recognition program more accurate is because if you look in that first diagram, you can see that the the object detection detects a potential object and then crops out everything outside the boxes, leaving just the objects to be recognised, this should reduce the overall time to recognise the image because only the non important parts of the image will be taken out which reduces the size of the images and it keeps only the important regions of the image.



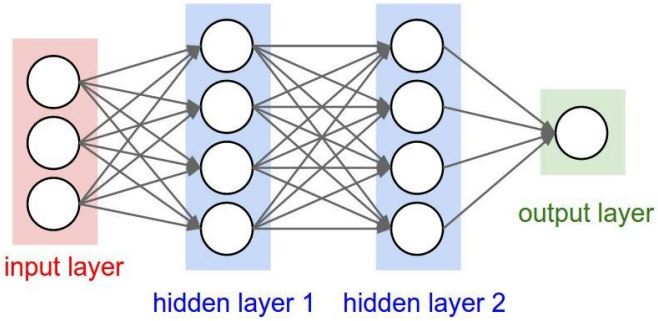
More about object detection
https://www.datacamp.com/community/tutorials/object-detection-guide



Neural networks

Now if you go back to link (1) we notice that in order for the program to recognise images we need something called a Neural network and so we first need to understand how Neural networks really work, so after some search I stumbled across this website https://medium.freecodecamp.org/want-to-know-how-deep-learning-works-heres-a-quick-guide-for-everyone-1aedeca88076 (2)
And this post explains how neural networks work, basically there's 3 main nodes

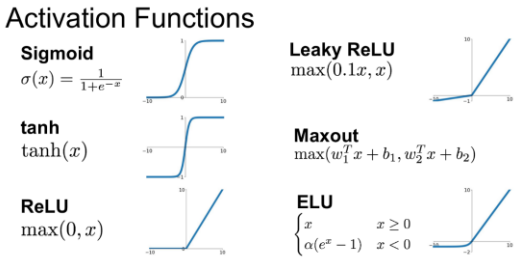
- Input node(where your input goes into in my case it will be the value of a pixel)
- Hidden node(where the learning bit comes in)
- Output node(where the final sum is outputted)



As we can see from the image above this is what a diagram of a neural network would look like, each line representing a weight in which it contain a value which was set up with a random number at the start of running the program and then the value of a weight would change by a process called Back propagation.

Back propagation:

Back propagation is the process of altering the weights to minimise the error as much as possible through Gradient descent



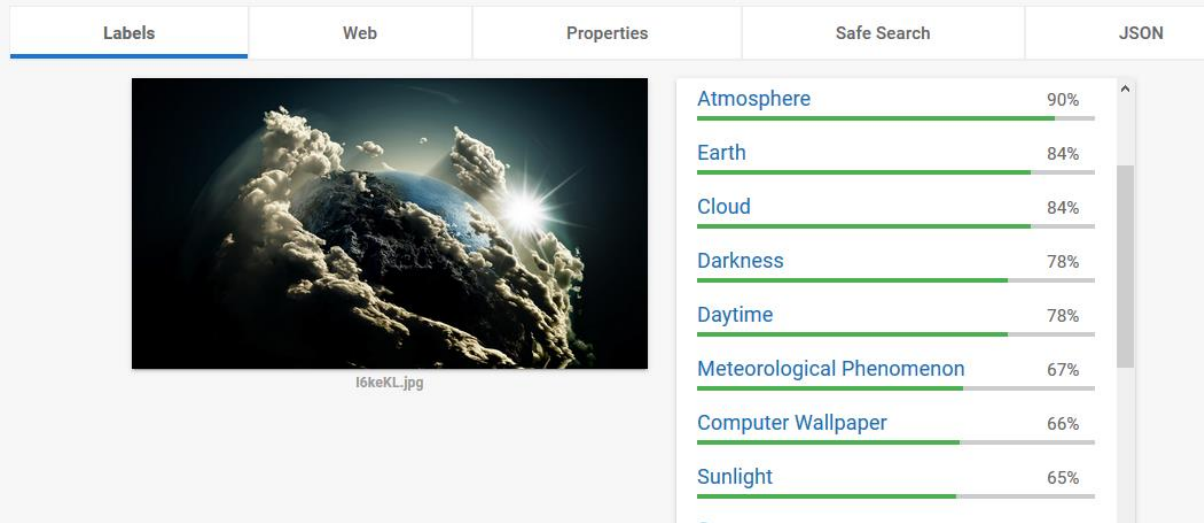
Gradient descent:
If we go back to (2) page they mentioned that the gradient descent just finds the best way to reduce the cost. "It works by **changing the weights** in small increments **after each data set iteration**. By computing the derivative (or gradient) of the cost function at a certain set of weight, we're able to see in which direction the minimum is.

Activation functions: There are many activations functions but these are mostly widely used for neural networks. Activation functions define the output of a node. https://medium.com/@shrutijadon10104776/survey-on-activation-functions-for-deep-learning-9689331ba092

Existing image recognition software

As part of my research I have to look up already existing image recognition software so I would know what my usp(unique selling point) would be, so after a quick search I came across an image recognition api that was made by google.

Cloud vision
Url: https://cloud.google.com/vision/



So Google's image recognition is pretty neat and easy to use while also displaying very accurate results as you can see through the screenshot above I uploaded an image of the earth and it pretty much describe most of the image.

Cloud vision also has a web tab in which it generates the tags that this image could have on the google search, see image below

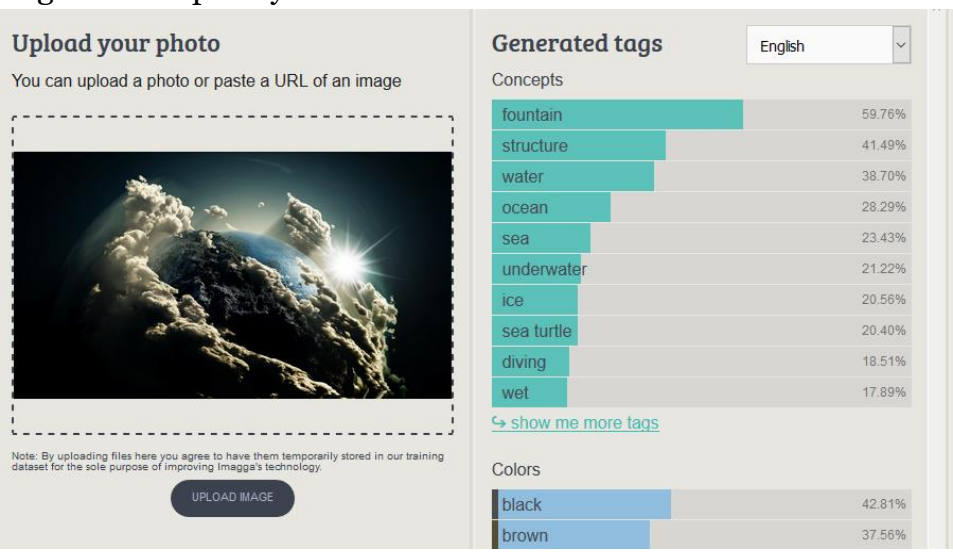
So this image recognition software is good and accurate and it gave me an idea, which is to display the percentage of the object detected so the user will know how certain the program is with its choice.

On the other hand Google's cloud vision recognition software doesn't label the parts of the image it identifies and that's something i'll try to implement.

Imagga

Url: https://imagga.com/

I disliked this image recognition website and the main reason for that is because it contained two main features from the Cloud vision api and it wasn't as accurate as the google's Cloud vision api but yet again I used the demo version, so the full version might be completely different to the one I used.

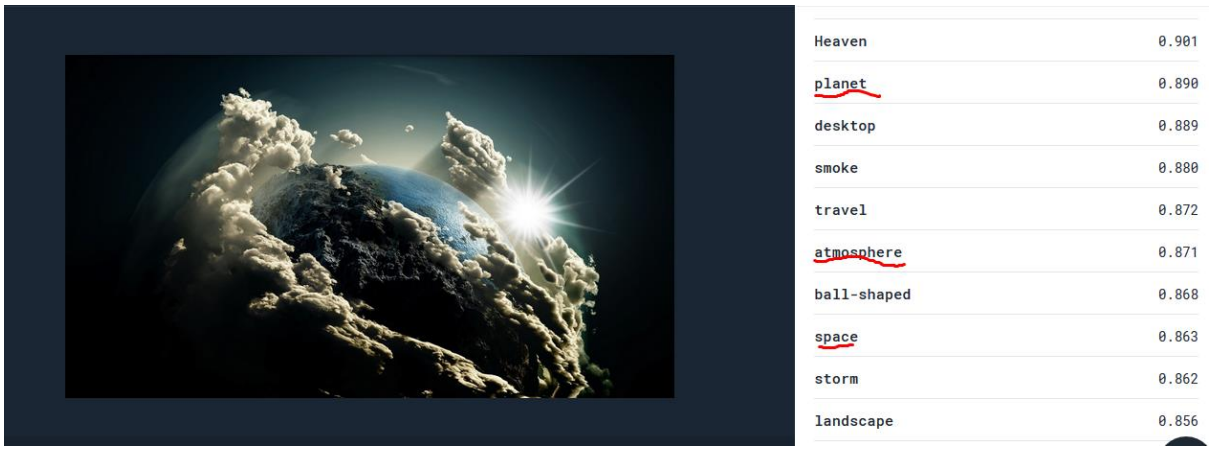


As you can see from the screenshot above, I used the same picture to make the test fair and it gave less accurate results to the Cloud vision api and it also had less features than the Cloud vision Api.

Clarifai

Url: https://clarifai.com/demo

Clarifai had very little features but it was highly accurate and specific, it was also the first to fully identify the image and yet again I used the same image to make the test fair.



So from what I found so far, these image recognition softwares have only a few features so to make my program stand out I would need to:

- **Have some extra features**
- **Make sure that it's very accurate**
- **Make it free(**because that would attract more people**)**

Conclusion:

So in order for my program to become successful and stand out from the other image recognition softwares i'll have to implement unique features and make it user friendly whilst also being fully customisable.

Requirements

The required features of my recognition program would be:

- **Customizability**(allows the user to customise my recognition program by changing the images it can detect)
- **Display**(my program will display what it can see and which objects it can detect)
- **Percentage**(my program should display the percentage of probability that the object is what it thinks it is)
- **Generate a box around the image**(this is just a fancy feature that tells the user if the program is selecting the objects precisely)
- **Ability to recognise objects**(the main feature of my program)
- **Realtime**(requested feature by one of my stakeholders, recognise objects in real time)

- **Ability to detect custom objects**(optional: a feature that lets the user to train my Ai to detect other types of objects)

Limitations

- **Specs**(if ran on a slow weak computer, the program might not work as fast)
- **Time**(I might run out of time and maybe not all features might be added)
- **AI**(If the neural network is not working as efficiently as the other image recognition AI's then my program might not be as accurate as I want it to be)
- **Image abstractor**(If I want to make my program run on slow computers then I will have to make the image abstractor more simpler which should speed the whole process up, but it will also make my program less accurate.)

Required specs

My image recognition program should require some descent hardware in order for it to work fast and efficiently so here's the following list of hardware specifications:

-Windows 7 or windows 10
-4gb Ram
-cpu with 2.4ghz or higher with 4 or more cores
-Integrated graphics

You could use less efficient hardware to run the program but that is the recommended specifications.

The reason for why it doesn't need a good graphics card is because this program will be made using only Visual studio and therefore it will not use up the graphics however it will use up your cpu and Ram.

Success Criteria

My image recognition program should consist in these following features:

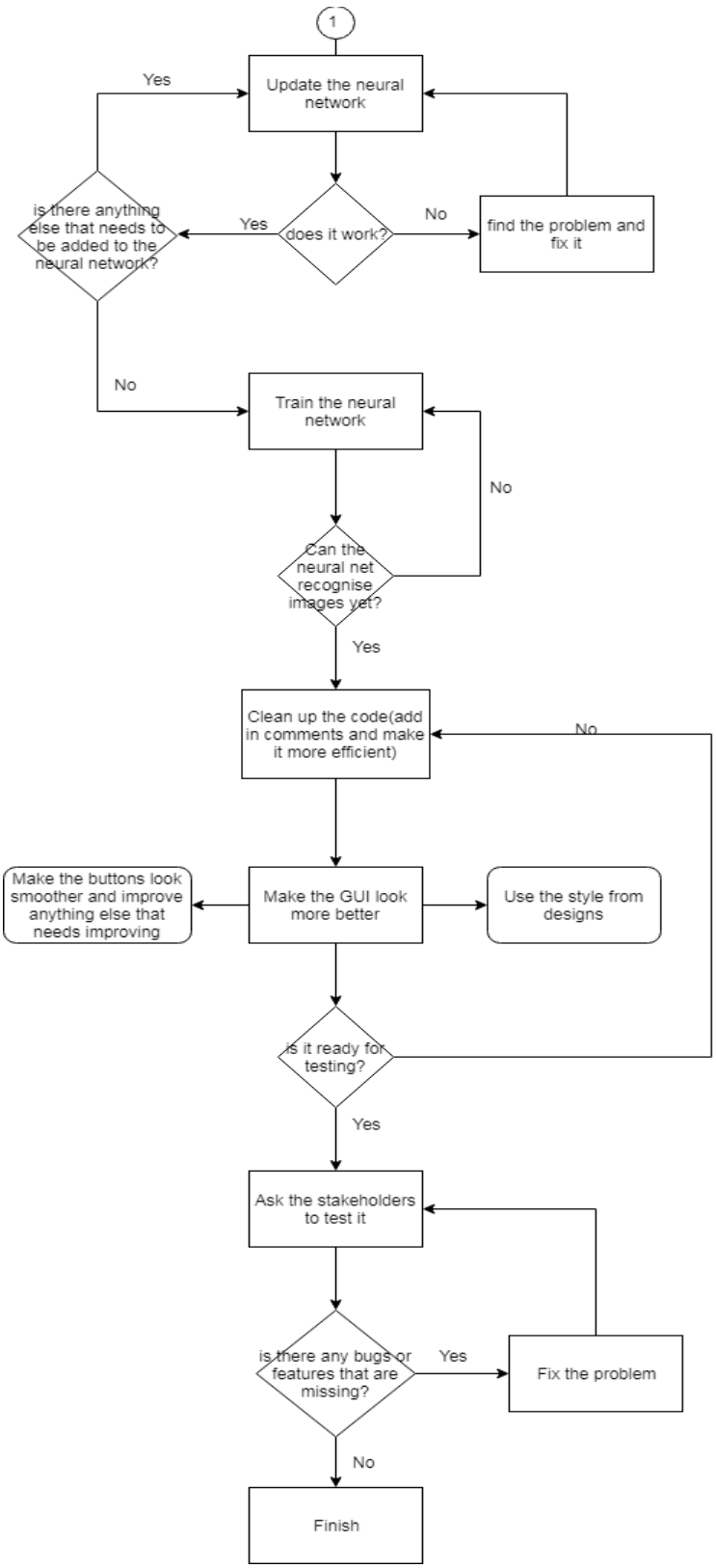
- **Image Abstractor working perfectly**(with all the filters applied)
- **Neural network fully functioning**(can be trained)
- **Display**(must display the input image and the abstracted image)
- **Generate boxes**(to show which object it has recognised and if the program has selected the object perfectly)

- **Show Percentages**(this will tell the user how sure the AI is with the object it has detected and recognised)
- **Customizability**(let's the program to be customizable for the users needs)
- **Recognising images**(image recognition program successfully recognising the image)
- **UI fully functioning**(so the user can use my program)

Design

Decomposition

For the decomposition I made a flow chart of every development stage I would have to go through in order for me to successfully create my image recognition program, and creating this flow chart is useful for problem decomposition in which one big problem can be split into multiple smaller problems.

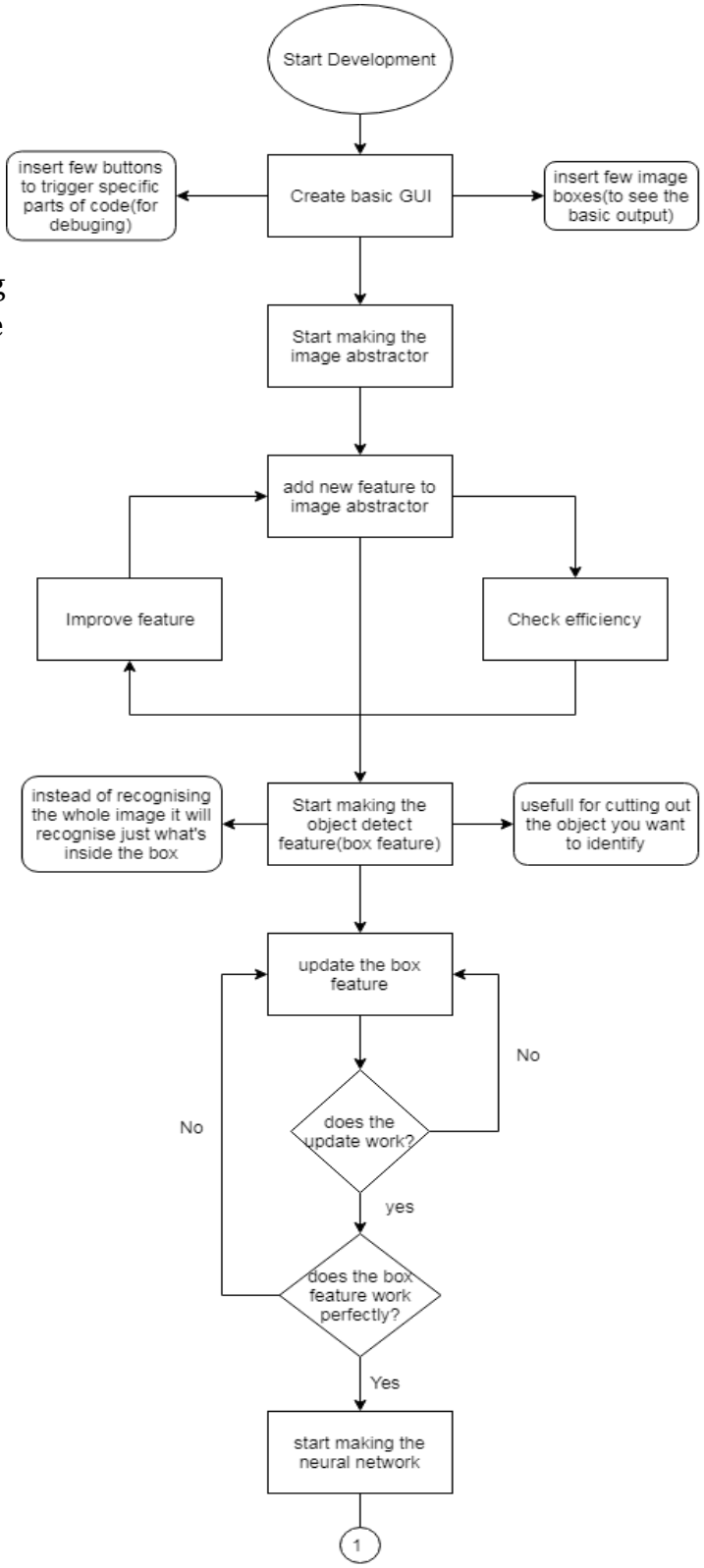


At the **1st stage** of development I will start by creating the basic GUI for debugging and checking if my features work how I want them to work.

On the **2nd stage** of development I would like to start on making the image abstractor so recognising images would be way more accurate **see page 7**. I've added a loop in the diagram to show that the image abstractor will require a lot of adjustments and tweaks so the image abstractor works perfectly.

The **3rd stage** would be the box feature which is also made in a similar style to the image abstractor, a lot of tweaks and bug fixes and testing to ensure that it works perfectly.

The **4th part** of my development will be the most complicated stage because it will require more
work and stages, as you can see in the diagram the neural network needs to be created, tested and trained.



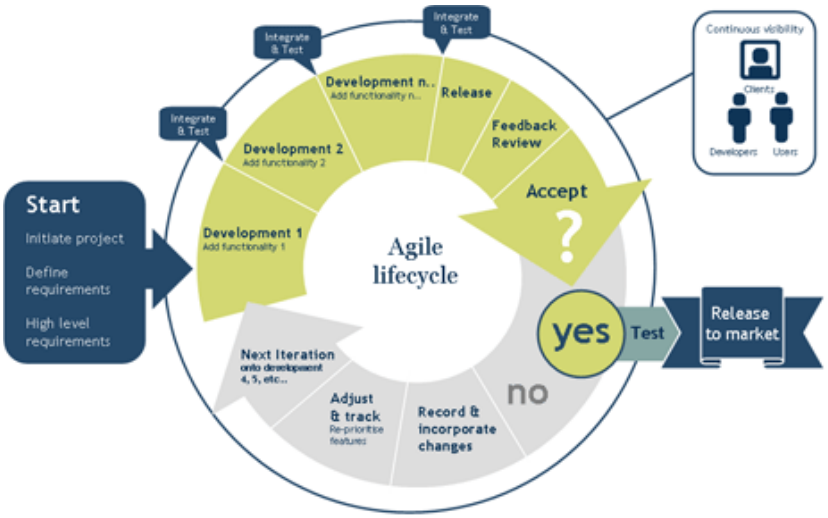
The **5th stage** of my development just involves me cleaning up the program, adding in comments and trying to make the program more efficient if possible.

On the **6th stage** of development I will have to create a better GUI if not already done, this stage is also useful
If I want my program to be easy to use.

The **7th stage** is the final stage and that's alpha testing, I will ask my stakeholders to test out my program and identify any bugs or problems my it might have, if no problems are identified and my stakeholders are happy with the program then this project could be considered as successful, but if there would be any bugs or problems, then I would do my best to try and fix/change the program to my stakeholders standards,
See success criteria.

Structure of solution

For my program I will be taking the object oriented approach and the reason for why I will be using this approach is because c# is the language I will be using to create my program and that language is an object oriented language, so in order for me to use this language I will have to take the object oriented approach. I will also have to use real world objects for my project and in order for me to use real world objects I will have to use the object oriented approach.
I will be taking the agile approach because this will make sure that the customer is happy with the final product produced and at the end of the whole development, the customer would be happy with its final product.



Planned versions:

Version	Features	Testing	Date Due
0.1	**GUI[**buttons for input, Image	Click the buttons and	End of october

	boxes for output, labels for simple output.] **Code[**setup the required variables and program the buttons to display the output**]**	see if basic output and input is functioning And if variables are set up	
0.2	**Code[**create a basic image abstractor, create a function to import pictures, program the image boxes to output abstracted images.**]**	Import a random picture and make sure that the basic image abstractor function works.	End of november
0.3	**Code[**create neural network runner and link the inputs to the neural network**]**	Check if running the neural network displays the correct output	End of December
0.4	**Code[**make neural network trainer and check if it works and reduces the cost**]**	See if training the neural network makes it smarter	End of January
0.5	**Code[**link up the image classifier, object detection and image abstractor together**]** **GUI[**implement any labels for debugging and testing if required**]**	Test if each of the features works and can recognise some of the basic images that the user can import	End of February
0.6	**Code[**add in a function to customise the main image recognition features**]** **GUI[**add in buttons to customise the image recognition features**]**	Test if the buttons open new forms for customising the image recognition features.	End of March
0.7	**Code[**implement more customisable functions, make object detector and image classifier customisable**]** **GUI[**make the whole program more pretty**]**	Test if the user can customise the image classifier and object detector	End of April

Algorithms

For my Image recognition program I will be using these algorithms to make my program fully functional:
This algorithm is the convolution stage of image abstraction section, different filters will be applied to the image, making it more abstract and accurate. Filters work by scanning each pixel and multiplying the rgb value to the the filter grid value and once that's done it will average the values and output just 1 pixel with filters applied. These

```
Convolution(bitmap pictr: int[3] filter)

        int[3] array:
        for(int i : filter.count(2) : ++i)              #loops for every filter ammount there is

                for(int y : pictr.height : ++y) #loops for each pixel in y axis

                        for(int x : pictr.width : ++x)  #loops for each pixel in x axis

                                for(int yy : filter.count(1) : ++yy)  # multiply each pixel with each segment of the filter

                                        for(int xx : filter.count(0) : ++xx)

                                                temp += filter[xx,yy,i] * pictr.getpixel[x + xx, y + yy, i]:

                                        end

                                end
                                temp / 9:

                                array[x,y,i] = temp:

                        end

                end

        end

        return(array.tobitmap()):

end
```

are the stages of how the abstraction stage will work, it will slowly turn the whole 300*300*3(x,y,rgb) image into a 1*1*256(x,y,filter) which we as people couldn't tell what the image is with just 256 different 1*1 pixels.

The max pooling algorithm if fairly simple since we are just taking the four pixels in the image adding them and averaging the

```
maxpooling(bitmap pictr)

        bitmap newpictr(pictr.width /2, pictr.height /2):

        for(int y : pictr.height/2 : ++y)

                for(int x : pictr.width/2 ++x)

                        temp = (pictr.getpixel[x,y] + [x+1,y] + [x,y+1] + [x+1,y+1])/4:
                        newpiictr.setpixel[x,y](temp):                                  #sets the new bitmap with the pooled pixel

                end

        end

        return(newpictr):

end
```

So what we should be left with is a 2* smaller image than what we started out with.

For my object detection algorithm which is used to track an object(see page 8), I decided to make it work like that, it would fetch the image and convert it into a 3d integer array, x-axis, y-axis, and R/G/B value.
Once that's done it will run the object detection neural net which can be trained to find objects locations and then lastly it will loop for every object that was found in the image and will run an image recognition to identify and make sure that it's that specific object which makes the whole program more accurate because this image recognition software will focus on just 1 type of object and if the object detection

picks something that's not the object it can be simply discarded after it's been runned on the image recognition.

```
                              #Object Detection Algorithm
ObjectDetect()
        bitmap pictr = imgbox1.image:
        int[3] imgpixel = new int[pictr.width.count,pictr.height.count,3]:   #int array of each pixel, x,y,R/G/B(1for red,2for green,3for blue)

        for(int y : pictr.height.count : ++y)
                for(int x : pictr.width.count: ++x)
                        imgpixel[x,y,1] = pictr.pixel(x,y,R)            #puts each pixel into an array
                        imgpixel[x,y,2] = pictr.pixel(x,y,G)
                        imgpixel[x,y,3] = pictr.pixel(x,y,B)
                end
        end

        list<> boxes = objectdetectneuralnet(imgpixel):
        foreach(boxes<num.count(),null,null,null,null> : ++i)                  # boxes<int number of objects,x,y,with,height>
                list<> results = imgrecognition(pictr.getpixels(boxes<null,i,i,i,i>))   #labelname,x,y,width,height
                if(results<i,null,null,null,null> != null)      #checks if the object detected is recognised
                        pictr.draw.selectivesquare(results<null,i,i,i,i>):  #draws bounding boxes on each object with label
                end
        end
end
```

Key variables and data structures

For my image recognition program I will be using the following key variables, data structures and classes:

- **3d integer array(would be for storing filter values eg x,y,filter**
- **Bitmaps(to retrieve/edit pixels of an image and to store images as a variable)**
- **2d double array(would be to store weights and hidden node values)**
- **1d double array(will be used to store input nodes and output nodes for my neural network)**

Key variables:

Data type	Name	Description
2d double array	weight	These are essential for neural networks and by storing them as a 2d array I would make the code a bit more tidy

		and since the neural network will be very large I will require a lot of weights.
2d double array	hidden	Hidden nodes are also very essential for neural networks and since there will be more than 4000 of them storing them in an array sounds appropriate thing to do.
3d integer array	filter	For the convolution a filter of x,y,filter num is required and since the filters will be 3x3x number of filters,storing them in an array would be a smart thing to do because yet again there could be over 100 different filters.
bitmap	pictr	I will be using this variable a lot for the image abstraction stage to store and edit the image

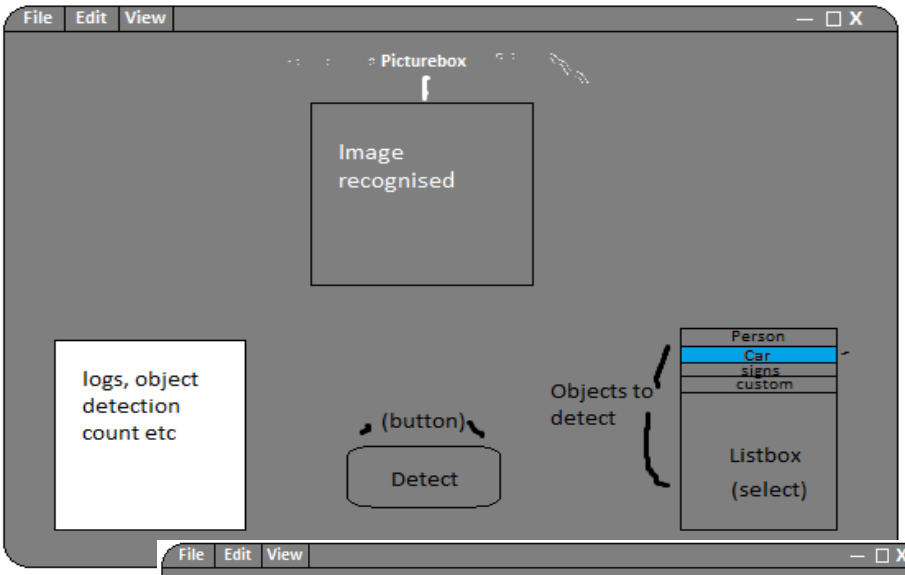
Classes:

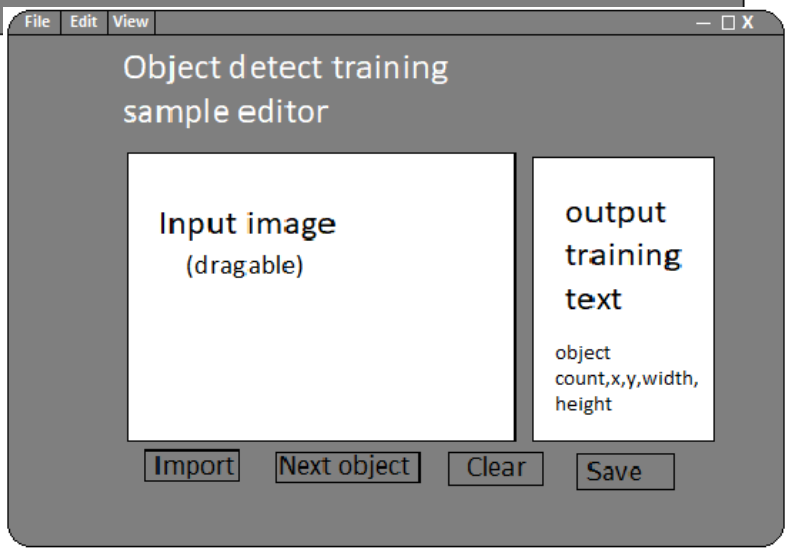
The main classes for my image recognition program would be:

- **Main**(for the main parts of my program including convolution,max pooling and running the trained neural network.
- **Object detect training sample editor**(for manually creating an object detection training samples)
- **Object detection training**(for training an object detector with a data sample of images and the object locations)
- **Image recognition training**(for training the main image recognition)
- **settings**(general settings page)

User Interface

Main menu:





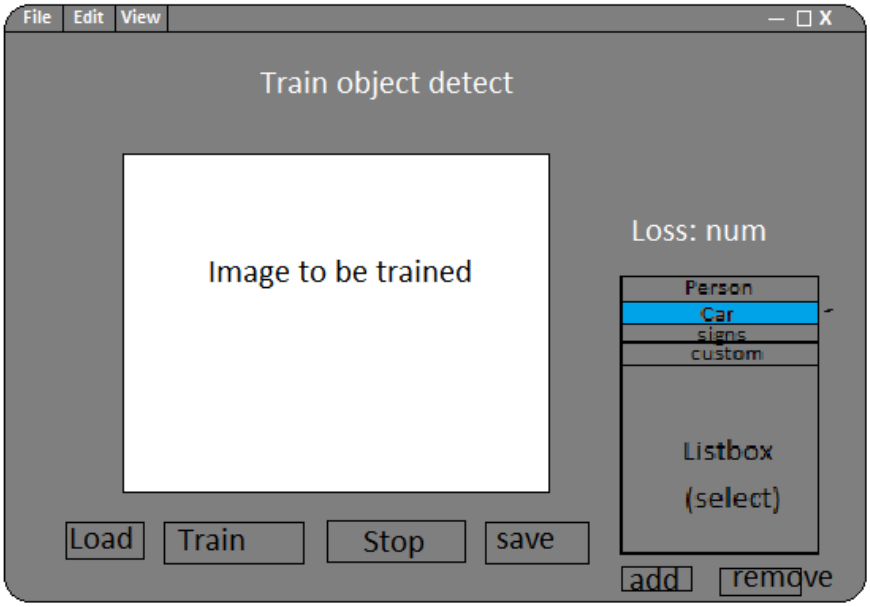
My main menu features user friendliness and customizability, I will have a listbox to select the images to detected which can be trained by the user.
Object detection sample editor, this would be used for editing images and creating them to train the object detection.

The input image would allow the user to select the objects in the image and it will generate the output.

The import button will allow the user to import and image, which would then be resized to a specific size. Next object button would train create another type of object to be trained on.
Lastly, the clear button clears any work you might of done and the save button saves the object into a file

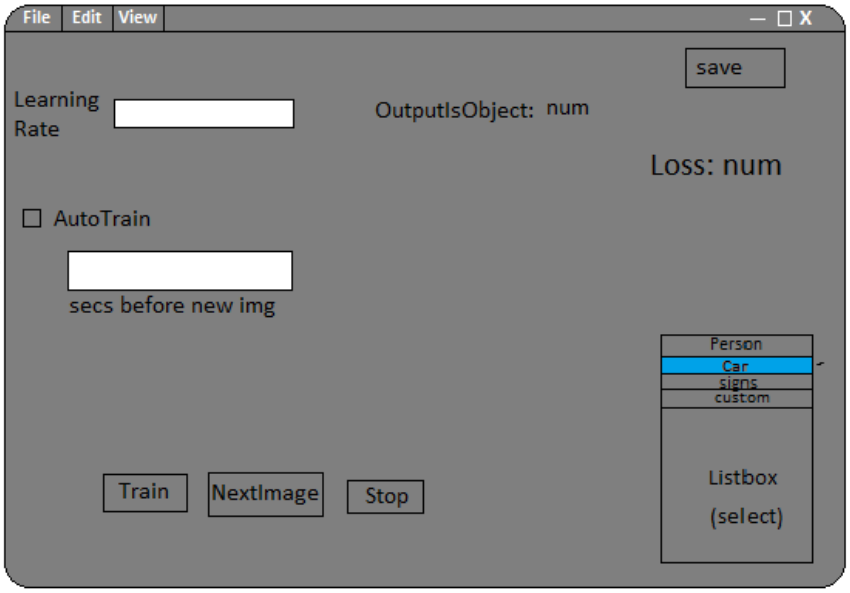
Object detector trainer:

The object detector trainer loads the training data and trains on the images.



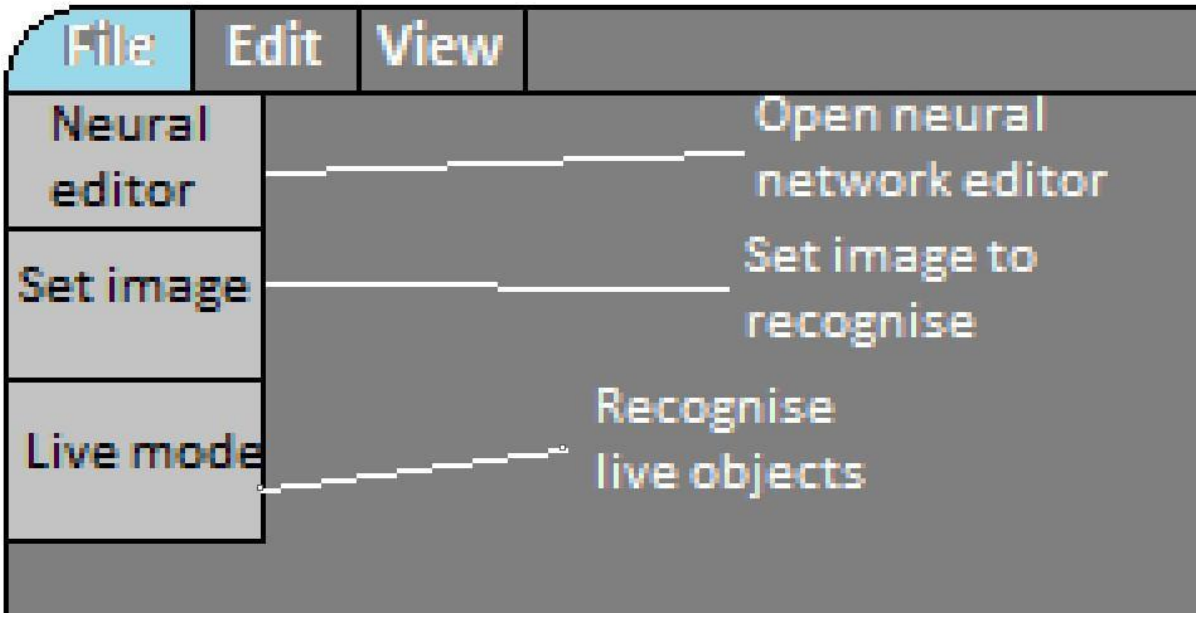
Train button starts the trainings and the stop button stops the training process, after the training is complete the person can either save the object detection or leave it unchanged.
The listbox and the add and remove button allows the user to remove certain objects to customise their object detector as much as they want.

Image recognition:



The main image recognition that recognises the images is slightly similar to the object detector but instead of detecting the location of the actual object it detects what the object is.

The empty box near label learning rate would be at how fast and rough do you want the image recognition to learn the images.
An auto train has been also added there and my program will also allow the user to choose at how fast the images will change, the faster it will change the smaller the chance the neural network has to adjust its weights, but if the time is too slow, then the neural network would reach its local minimum and time would be wasted.
As you can see it has a train button to start the training and allows the neural network to be trained by hand. Next image button does what it says and that is to chose another random image to be the training data, and a stop button when the user would feel like the neural network had enough training.



This is the basic functions that my program will have in the file menu, to go on basic image set mode, which is where the user imports an image and it gets recognised, live mode if the user will want to recognise objects on video or camera and lastly neural editor where the user can edit the object detector and the image recognition.

Conclusion:

I've made this type of design to allow the user to have as much customizability as they can, I want to also make this user friendly so I will make each of these functions easy to use and understandable.

Test data

In my first prototype version 0.1, I had to test if basic input and output is working and if the first time setup is fully functional, so I decided to make the following test data to see if it works exactly how I want it to work.

- Click the main detection button and see if it displays a simple image to see if it works.
- Use every function in toolbox and see if they all work and function
- See if first time setup works perfectly(if everything loads in)

Prototype version 0.2 would be about image abstraction(max pooling and convolution). I will need to see if these functions would work perfectly by checking:

- If maxpool function would sample down the image to the right size

- If convolution function would apply the filter to the image and output the right sized array

Prototype 0.3 would link up the the max pool function with the convolution function

- Check if the max pooling and convolutions follow the chosen algorithm
- Make the convolution and max pooling link up
- Check if the input image would be outputted as a completely abstracted image.

Prototype 0.4 this would be a very big update as it would require the main image recognition function to be made

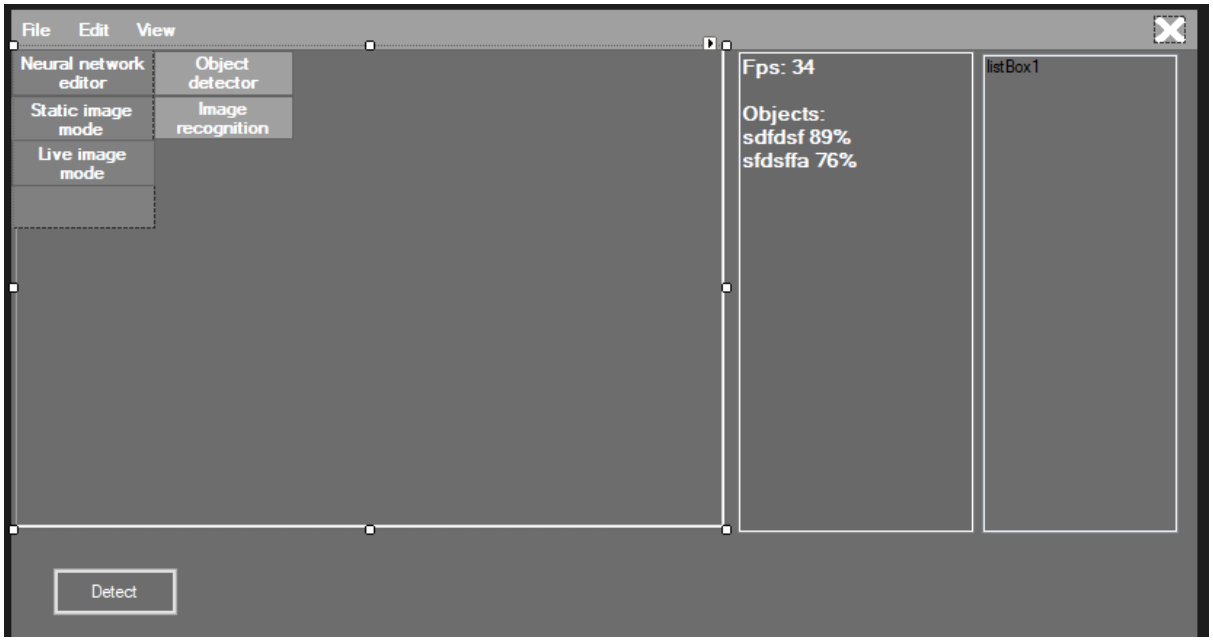
- Check if the image recognition recognises a simple image
- Check if it will recognise multiple objects in image

Development

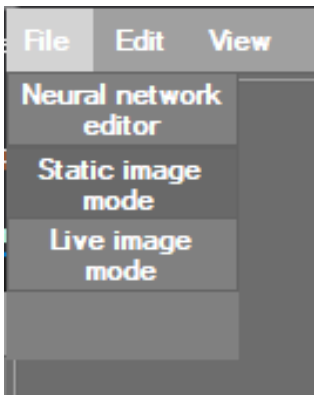
Version 0.1

Gui

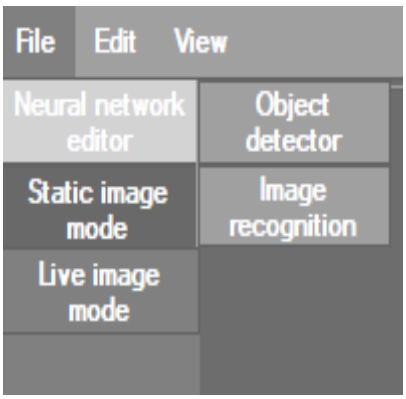
In my development I have made a slightly different design than the one I was originally planning on making.



The reason for why I made it different from the original design is because I feel like this design looks even better than the original one and it's easier to use since it doesn't have a complicated design.



In my development I've made a simple but nice feature, in which allows the user to have a nicer selection in the toolbar.
Which gives a nicer feeling to the GUI.

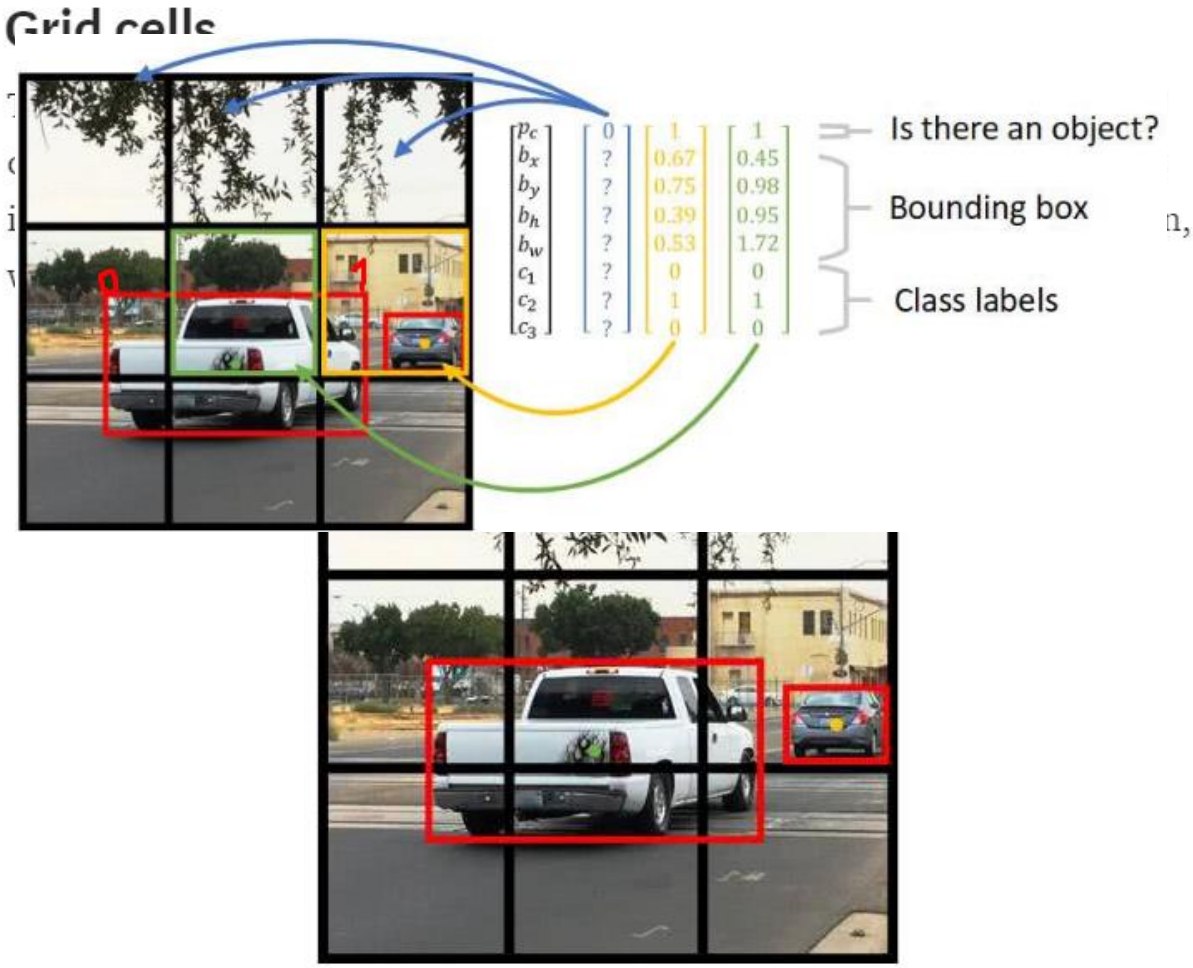


Research and planning

Later in my development I stumbled across a slight problem when I was setting up the variables, I first decided to store all my weights and filters as a text file since there's just so many of them and saving it in a text file will also mean that your training progress would be saved. My first problem was to find out the exact number of weights and filters I will require.
So I was first looking at rcnn and I found this link which and it made me think that r-cnn might not be the best architecture for my image recognition project. I then stumbled across Yolo(you only look once) architecture and it's way more efficient and faster than r-cnn because it merges the object detector and the image recognition

into one neural network. https://towardsdatascience.com/r-cnn-fast-r-cnn-faster-r-cnn-yolo-object-detection-algorithms-36d53571365e

So then I started looking up more about Yolo, and how it works and found this very useful article https://heartbeat.fritz.ai/gentle-guide-on-how-yolo-object-localization-works-with-keras-part-2-65fe59ac12d



Research and planning pt2

After a lot of research, I found out that the image would be split into a 19*19 grid cell and each grid cell is responsible for predicting a class and the location of the bounding box.

So as you can see in the example
Pc = the confidence level of the object if 0 then ignore that grid cell
Bx = the x coordinate in which the centre of the object corresponds to the top left corner of the grid cell

By = the y coordinate in which the centre of the object corresponds to the top left corner of the grid cell

Here's an example of how it works, since the centre is near the bottom of the cell the "By" would be 0.95 and since the centre is closer to the left of the cell the "Bx" would be 0.45

Bh = the height of the bounding box
Bw = the width of the bounding box
C1 = the confidence score of it being class 1(bike for example)
C2 = the confidence score of it being class 2(car for example) C3 = the confidence score of it being class 3(human for example).

Name	Filters	Output Dimension
Conv 1	7 x 7 x 64, stride=2	224 x 224 x 64
Max Pool 1	2 x 2, stride=2	112 x 112 x 64
Conv 2	3 x 3 x 192	112 x 112 x 192
Max Pool 2	2 x 2, stride=2	56 x 56 x 192
Conv 3	1 x 1 x 128	56 x 56 x 128
Conv 4	3 x 3 x 256	56 x 56 x 256
Conv 5	1 x 1 x 256	56 x 56 x 256
Conv 6	1 x 1 x 512	56 x 56 x 512
Max Pool 3	2 x 2, stride=2	28 x 28 x 512
Conv 7	1 x 1 x 256	28 x 28 x 256
Conv 8	3 x 3 x 512	28 x 28 x 512
Conv 9	1 x 1 x 256	28 x 28 x 256
Conv 10	3 x 3 x 512	28 x 28 x 512
Conv 11	1 x 1 x 256	28 x 28 x 256
Conv 12	3 x 3 x 512	28 x 28 x 512
Conv 13	1 x 1 x 256	28 x 28 x 256
Conv 14	3 x 3 x 512	28 x 28 x 512
Conv 15	1 x 1 x 512	28 x 28 x 512
Conv 16	3 x 3 x 1024	28 x 28 x 1024
Max Pool 4	2 x 2, stride=2	14 x 14 x 1024
Conv 17	1 x 1 x 512	14 x 14 x 512
Conv 18	3 x 3 x 1024	14 x 14 x 1024
Conv 19	1 x 1 x 512	14 x 14 x 512
Conv 20	3 x 3 x 1024	14 x 14 x 1024
Conv 21	3 x 3 x 1024	14 x 14 x 1024
Conv 22	3 x 3 x 1024, stride=2	7 x 7 x 1024
Conv 23	3 x 3 x 1024	7 x 7 x 1024
Conv 24	3 x 3 x 1024	7 x 7 x 1024
FC 1	-	4096
FC 2	-	7 x 7 x 30 (1470)

Research and planning pt3

As I was doing some even more research I came across an even more detailed and helpful explanation as it tells us how many filters and weights we will use for convolution and neural network stage. So as we can see: We would start of with 448*448*3 which is height*width*RGB, at the top of that list we have the first convolution with a stride of 2 which will move a 7*7 convolution filter by 2 pixels which reduces the image size by 2x more about convolution and max pooling check pages 5-6.

Since filters would have a random number between -1 and 2 I have to make a file for each different filter and the reason for why I will be storing these filters in a file is that the neural network finds patterns and so the filters should always stay the same because if it would change then it could impact the image recognition drastically.

So the filter files I would have would be:
64_7x7.txt
192_3x3.txt
128_1x1.txt
256_3x3.txt

256_1x1.txt
512_1x1.txt
512_3x3.txt
1024_3x3.txt

The reason for why I have not put a seperate file for a repeating filter is because instead of using the same size filter but in a different file and different filter numbers, I decided to re use the same sized filter since it would save up some space and it would just be unnecessary because if we apply the same filter twice, the output will still change.

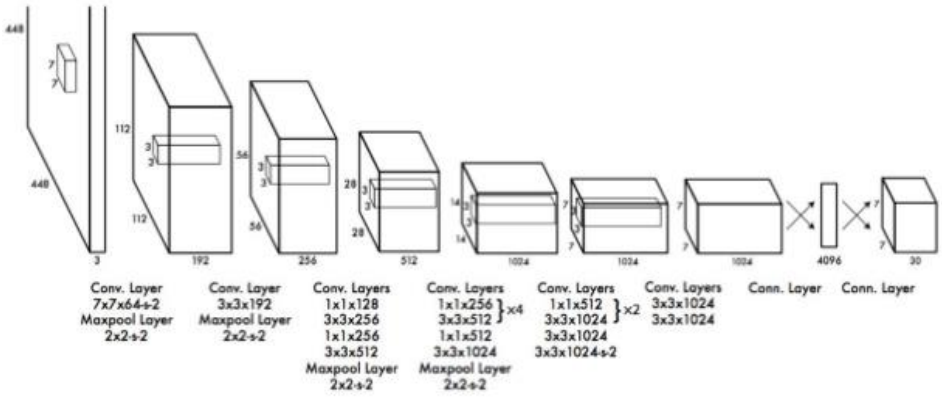
Research and planning pt4

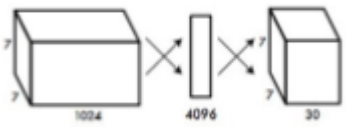
I will save the filters as text file and since there will be a couple of them, I will have to write a temporary function to create that text file with random filter values.
The way I will do it is by storing each filter on a new line for example:

3_3x3.txt would be
-1, 1, 0, 1, 1, 2, 2, 0, 0
0, 0, 1, 2, 2,-1,-1,-1, 0
1, 1, 2, -1,1, 2, 2, 0, 0
I've made it so that 2d matrices will become flatten into a 1d, so my plan is when the program would start it would open these text files and store these filters into a 2d integer array.

One last thing that I need to do before I can continue with the programming is to find out how many weights are needed for the neural network, more about neural networks see page 9.

If we look back at that table we can see that FC1 and FC2 are on there, so I was a bit confused since I didn't know what it meant, so then I looked up yolo algorithm and I found this detailed diagram.





If we look at the end we can see that the final convolved layer which is a 7x7x1024 will go into a 1 hidden node and a simple way of figuring out how many weights will be between the final convolved layer and the hidden node, is to multiply them with each other, since the final convoled layer would be the input part of the neural network and that 7x7x30 would be the output since it's right at the back.

Calculating weights

Since the input would be 1d integer array I will have to make that 3d integer array which is the convolved layer into a 1d integer array, so we just simply find the area of that 3d integer array layer so 7x7x1024 is 50176 so we will have a lot of inputs and as we can see that convolution and max pooling is essential for image recognition as it would be very very computationally expensive program.
So since we now have a 50176 input we can find out the first layer of weights by multiplying that number by 4096 which is the amount of hidden node in hidden layer. The first amount of weights on w1 is 205520896 and that's a lot but I still have to find the other weight layers size(w2) and to calculate that we just simply multiply the 4096 by 7*7*30 which would be 6021120 weights.

So we will have:

205520896 weights on w1
6021120 weights on w2

With a total of 211542016 different weights that need to be made and each weight will be randomly generated between 1 and 0 and in between so one example of a random weight is -0.23423442 and as we know, weights are basically the memory of a neural network so they are very important.
So the way I will store them is by adding all of the weights into 2 text files:

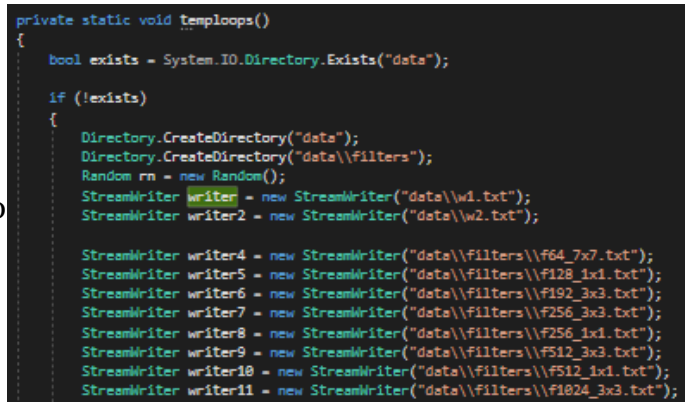
w1.txt

w2.txt

Each of the weights will be separated by a new line so when the program would load, it would store each of the weights into a 1d double array.

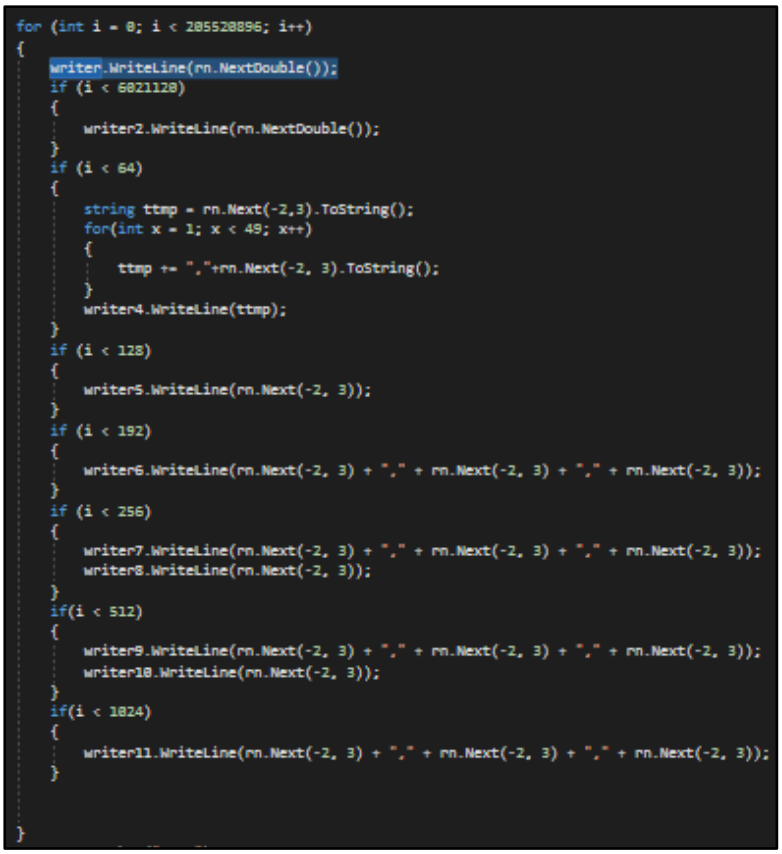
Generating weights temporary code

So I made a temporary function to be run so I would have the randomly generated filters and weights, I will later remove it as I would just simply add those files with the installation and this would allow the user to use different weights from other users to share their weights which would be trained on whatever object they train it on.



```
private static void temploops()
{
    bool exists = System.IO.Directory.Exists("data");

    if (!exists)
    {
        Directory.CreateDirectory("data");
        Directory.CreateDirectory("data\\filters");
        Random rn = new Random();
        StreamWriter writer = new StreamWriter("data\\w1.txt");
        StreamWriter writer2 = new StreamWriter("data\\w2.txt");

        StreamWriter writer4 = new StreamWriter("data\\filters\\f64_7x7.txt");
        StreamWriter writer5 = new StreamWriter("data\\filters\\f128_1x1.txt");
        StreamWriter writer6 = new StreamWriter("data\\filters\\f192_3x3.txt");
        StreamWriter writer7 = new StreamWriter("data\\filters\\f256_3x3.txt");
        StreamWriter writer8 = new StreamWriter("data\\filters\\f256_1x1.txt");
        StreamWriter writer9 = new StreamWriter("data\\filters\\f512_3x3.txt");
        StreamWriter writer10 = new StreamWriter("data\\filters\\f512_1x1.txt");
        StreamWriter writer11 = new StreamWriter("data\\filters\\f1024_3x3.txt");
```

The code is pretty simple, just checks if the folder exists and if it doesn't it will generate random weights and filters, but now that I think of it, I will keep the weight generator in case the user would like to train even more different objects but the filters will have to be kept the same otherwise other users won't be able to use their own trained weights on another device with different filters as the neural network finds patterns and learns to use the filters given.
I figured out a strategy to increase the speed and as I was programming the random weights and filter function, I decided to put it all into a 1 for loop, so instead of the function going through several loops it will go through just one.



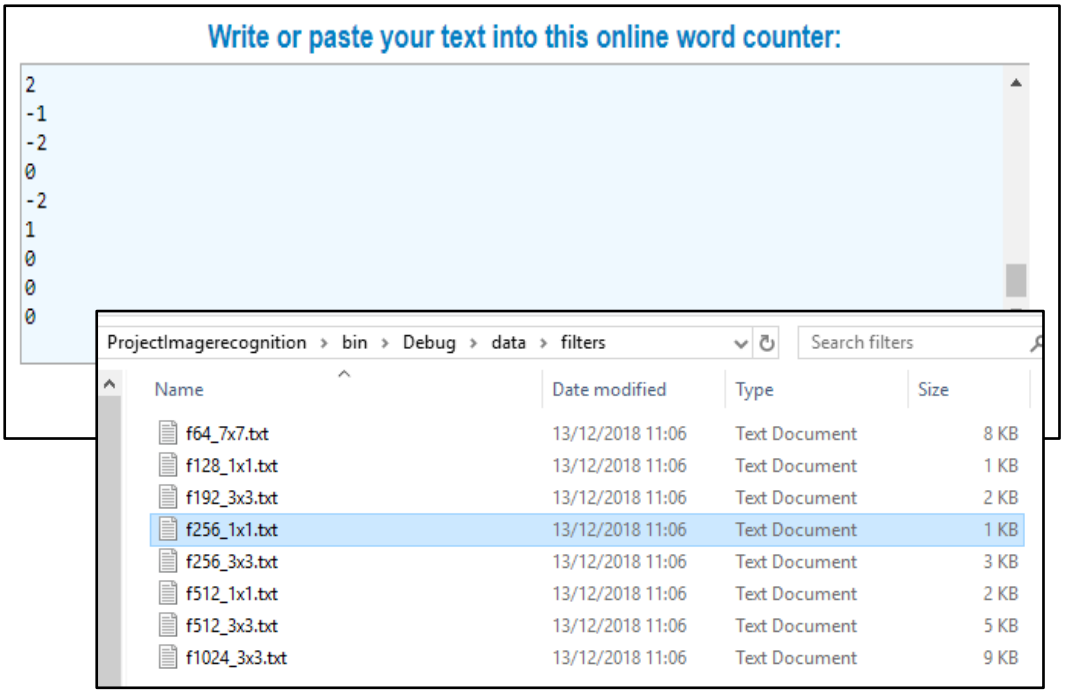
```
for (int i = 0; i < 205520896; i++)
{
    writer.WriteLine(rn.NextDouble());
    if (i < 6021120)
    {
        writer2.WriteLine(rn.NextDouble());
    }
    if (i < 64)
    {
        string ttmp = rn.Next(-2,3).ToString();
        for(int x = 1; x < 49; x++)
        {
            ttmp += ","+rn.Next(-2, 3).ToString();
        }
        writer4.WriteLine(ttmp);
    }
    if (i < 128)
    {
        writer5.WriteLine(rn.Next(-2, 3));
    }
    if (i < 192)
    {
        writer6.WriteLine(rn.Next(-2, 3) + "," + rn.Next(-2, 3) + "," + rn.Next(-2, 3));
    }
    if (i < 256)
    {
        writer7.WriteLine(rn.Next(-2, 3) + "," + rn.Next(-2, 3) + "," + rn.Next(-2, 3));
        writer8.WriteLine(rn.Next(-2, 3));
    }
    if(i < 512)
    {
        writer9.WriteLine(rn.Next(-2, 3) + "," + rn.Next(-2, 3) + "," + rn.Next(-2, 3));
        writer10.WriteLine(rn.Next(-2, 3));
    }
    if(i < 1024)
    {
        writer11.WriteLine(rn.Next(-2, 3) + "," + rn.Next(-2, 3) + "," + rn.Next(-2, 3));
    }

}
```

Testing

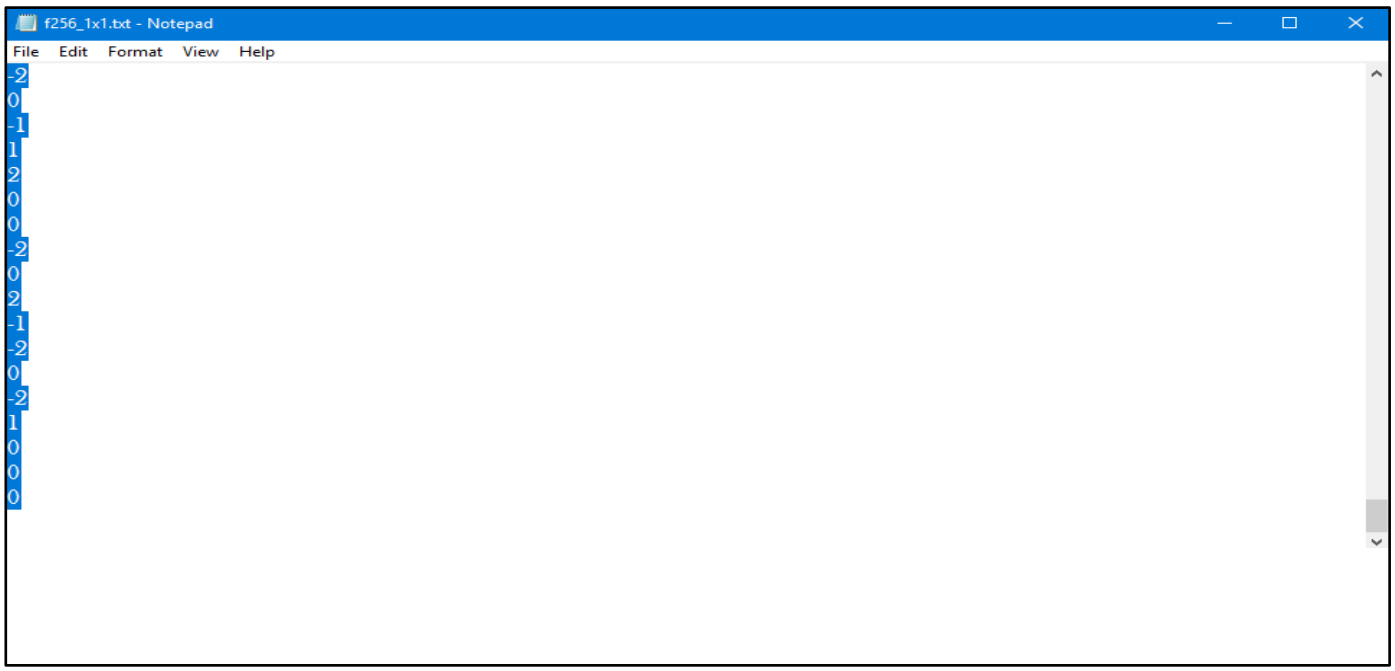
So now that I created the function I tested it by running my program and opening the text files where they would be stored in, but I experienced another slight problem which was to count if it's exactly the right amount but since I won't be counting 64

different lines I used an online word count which also counted the amount of new lines and it all worked just how I wanted it to work.

As you can see, all of the filters are there and stored just how I wanted them to be stored. I used an online word count



https://www.countofwords.com/ which was very helpful in making sure that I have made the right amount of filters.



Fetching data

Since all the filters and weights have been generated, I will have to temporarily store the data in my program and the data structures I will use are listed in page 20 but since I know more about how my program will have to work, I decided to use 1 dimensional double array for my weights and hidden layers since it would make my whole program easier to understand.

Key variables I will use to store generated data:

- **W1(weight 1d double array for neural networks)**
- **W2(weight 1d double array for neural networks)**
- **filter_64_7x7(filter 3d integer array for convolution)**
- **filter_192_3x3(filter 3d integer array for convolution)**
- **filter_128_1x1(filter 1d integer array for convolution)** I made this a 1 dimensional integer array because there will only be 128 different filters and the grid will only be the size of 1x1
- **filter_256_3x3(filter 3d integer array for convolution)**
- **filter_256_1x1(filter 1d integer array for convolution)**
- **filter_512_1x1(filter 1d integer array for convolution)**
- **filter_512_3x3(filter 3d integer array for convolution)**
- **filter_1024_3x3(filter 3d integer array for convolution)**

These are the variables, but one simple problem is that these variables default value would be 0 so we just need to assign each variable with the data in the text files.

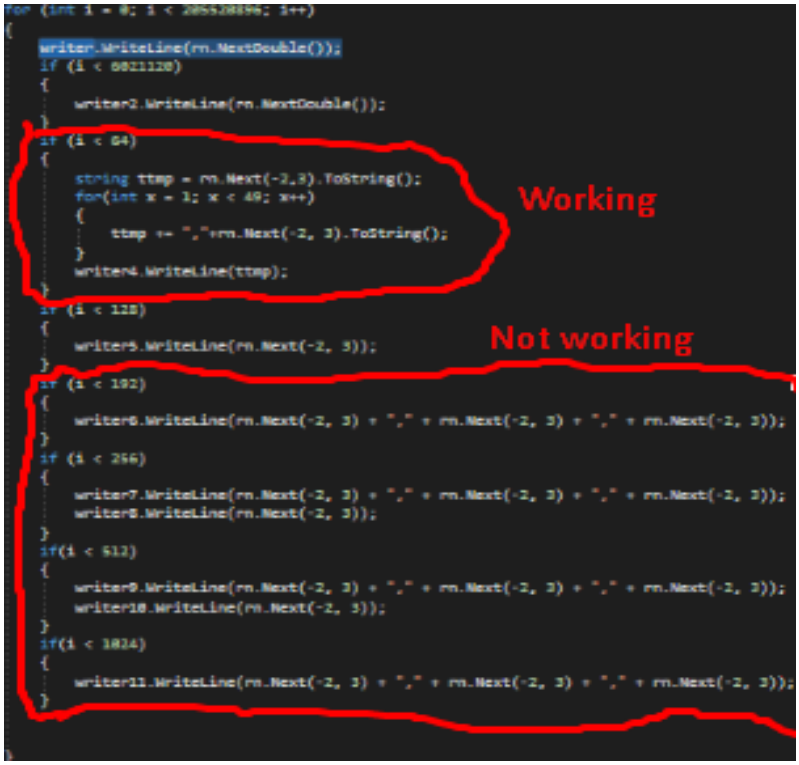
```
static double[] w1 = new double[205520896];
static double[] w2 = new double[6021120];
static double[,,] filter_64_7x7 = new double[64,7,7];
static double[,,] filter_192_3x3 = new double[192, 3, 3];
static double[] filter_128_1x1 = new double[128];
static double[,,] filter_256_3x3 = new double[256, 3, 3];
static double[] filter_256_1x1 = new double[256];
static double[] filter_512_1x1 = new double[512];
static double[,,] filter_512_3x3 = new double[512, 3, 3];
static double[,,] filter_1024_3x3 = new double[1024, 3, 3];
```

Fetching data Problem

As I was developing my fetch data function from files, I experienced a problem, one of the arrays were overflown and when I looked back at my notes, I realised a huge mistake that i've made

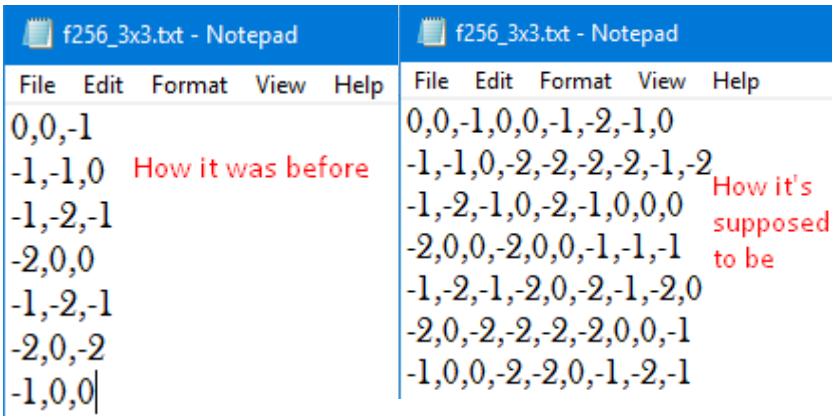
I realised that something within this code wasn't right, and then I checked the text files again, and as I thought, my generating function didn't work as it's supposed to work,

But after realising the cause of this bug, I used the working section of generating function and applied it to the rest of the filters.



I confirmed that each of the filter files are just the way I wanted them to be by individually counting the number of numbers in a single line.

Now that I've got all the filters and weights set up I will have to continue and create a function that fetches the data and stores it within the arrays.



Fetching data code

```
private static void fetchdata()
{

    Random rn = new Random(); //sets up the random number generator

    var readw1 = File.ReadLines("data\\w1.txt");//specifies the location of the file
    var readw2 = File.ReadLines("data\\w2.txt");//weights


    var readf64_7x7 = File.ReadAllLines("data\\filters\\f64_7x7.txt");//filters
    var readf128_1x1 = File.ReadAllLines("data\\filters\\f128_1x1.txt");
    var readf192_3x3 = File.ReadAllLines("data\\filters\\f192_3x3.txt");
    var readf256_1x1 = File.ReadAllLines("data\\filters\\f256_1x1.txt");
    var readf256_3x3 = File.ReadAllLines("data\\filters\\f256_3x3.txt");
    var readf512_1x1 = File.ReadAllLines("data\\filters\\f512_1x1.txt");
    var readf512_3x3 = File.ReadAllLines("data\\filters\\f512_3x3.txt");
    var readf1024_3x3 = File.ReadAllLines("data\\filters\\f1024_3x3.txt");
    int teemp = 0;//temporary variable
    loading = true;//for loading progress bar, activates it
    loading_max = 211543040;//specifies the size of progressbar6
    loading_value = 0;//sets the loading value back to 0 which is the start

    foreach (var line in readw1)//loops 205520896 times
    {

        w1[teemp] = Convert.ToDouble(line);//stores each line in w1.txt to a 1d double array
        teemp++;//incremeants the temporary value
        loading_value++;//increase the loading progressbar value
    }
```

```
foreach(var line in readw2)//loops 6021120 times, since that's how many weights and lines there are
{
    w2[teemp] = Convert.ToDouble(line);
    teemp++;
    loading_value++;
}

for (int i = 0; i < 1024; i++)//readf1024_3x3 = 1024 which is the ammount of loops this will go through
{
    loading_value++;


    if (i < 1024)//makes sure that the program doesn't execute the code within the if statement more than 1024 times
    {
        int tmpp = 0;//temporary value
        string[] word = readf1024_3x3[i].Split(',');//using similar code as the one with weights but since, the data is only fetched -
        //line by line, this function will have to split that 1 line into many seperate numbers
        for (int y = 0; y < 3; y++)
        {
            for (int x = 0; x < 3; x++)
            {

                filter_1024_3x3[i, y, x] = Convert.ToDouble(word[tmpp]);//stores the data into a 3d integer array
                tmpp++;
            }
        }
    }
```

```
    if (i < 256)
    {
        filter_256_1x1[i] = Convert.ToDouble(readf256_1x1[i]);
        int tmpp = 0;
        string[] word = readf256_3x3[i].Split(',');
        for (int y = 0; y < 3; y++)
        {
            for (int x = 0; x < 3; x++)
            {
                filter_256_3x3[i, y, x] = Convert.ToDouble(word[tmpp]);
                tmpp++;
            }

        }
    }
    if (i < 512)
    {
        filter_512_1x1[i] = Convert.ToDouble(readf512_1x1[i]);
        int tmpp = 0;
        string[] word = readf512_3x3[i].Split(',');
        for (int y = 0; y < 3; y++)
        {
            for (int x = 0; x < 3; x++)
            {
                filter_512_3x3[i, y, x] = Convert.ToDouble(word[tmpp]);
                tmpp++;
            }

        }
    }
}
```

Fetching data code prt2

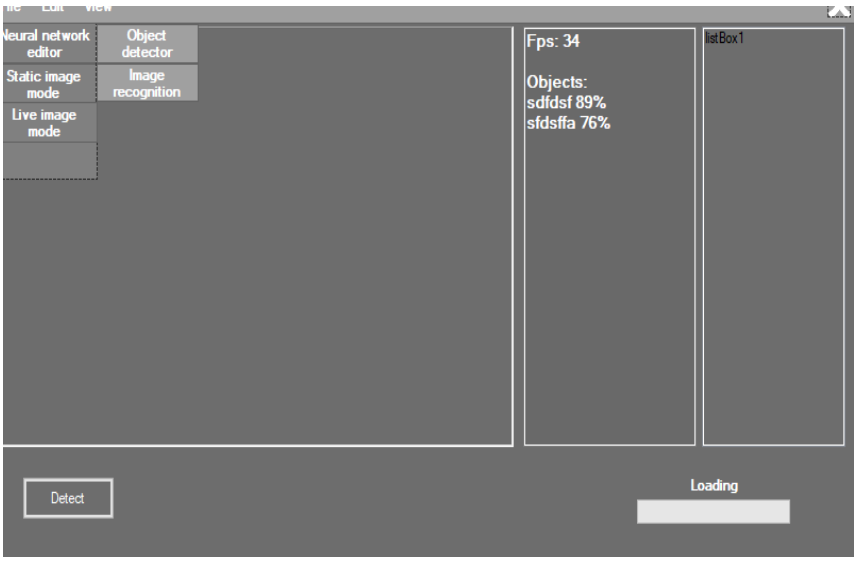
This is the whole function that loads in data into arrays and I did some tests later on by running it and displaying the values

As I made that function I noticed that it was missing something, since it took a while to load, I decided to create a loading progress bar, so the user would know if the program is close to finishing or not.

```
if (i < 64)
{
    int tmpp = 0;
    string[] word = readf64_7x7[i].Split(',');
    for (int y = 0; y< 7; y++)
    {
        for(int x = 0; x< 7; x++)
        {

            filter_64_7x7[i, y, x] = Convert.ToDouble(word[tmpp]);
            tmpp++;
        }
    }
}
if(i < 128)
{
    filter_128_1x1[i] = Convert.ToDouble(readf128_1x1[i]);//since filter_128_1x1 is only a 1x1 then there's no need to make a 3d int array so this -
    //segment of code will be used just like it was used with w1 and w2
}
if(i< 192)
{
    int tmpp = 0;
    string[] word = readf192_3x3[i].Split(',');
    for(int y = 0; y < 3; y++)
    {
        for(int x = 0; x<3; x++)
        {
            filter_192_3x3[i, y, x] = Convert.ToDouble(word[tmpp]);
            tmpp++;
        }

    }
```

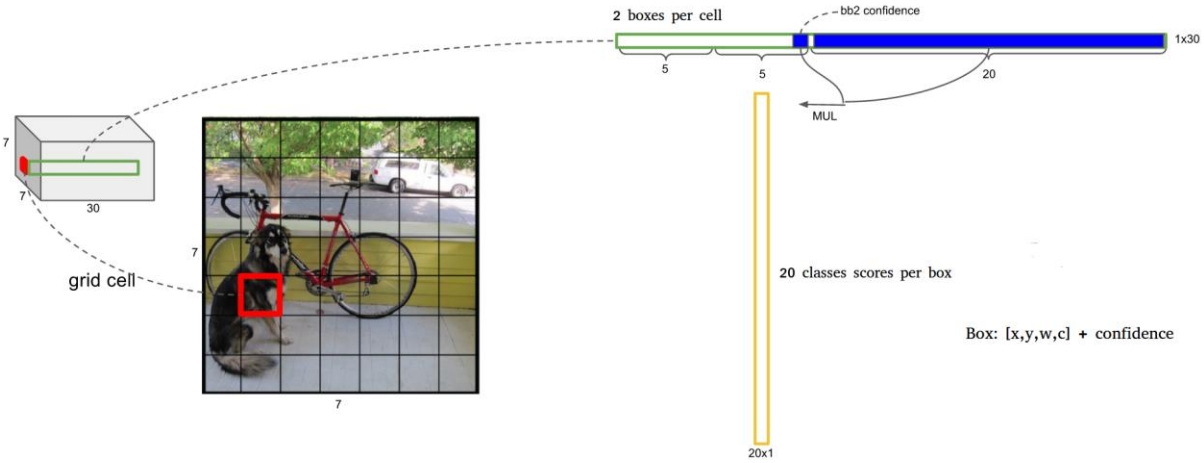
```
loading_value = 0;//sets the values back to default
loading_max = 0;
loading = false;//dissables loading progress bar
MessageBox.Show("Done");
```

The loading bar would show up only when something is being loaded and will take a while, once the progress bar finished loading, it will hide it self and show it self again when loading will be needed again.



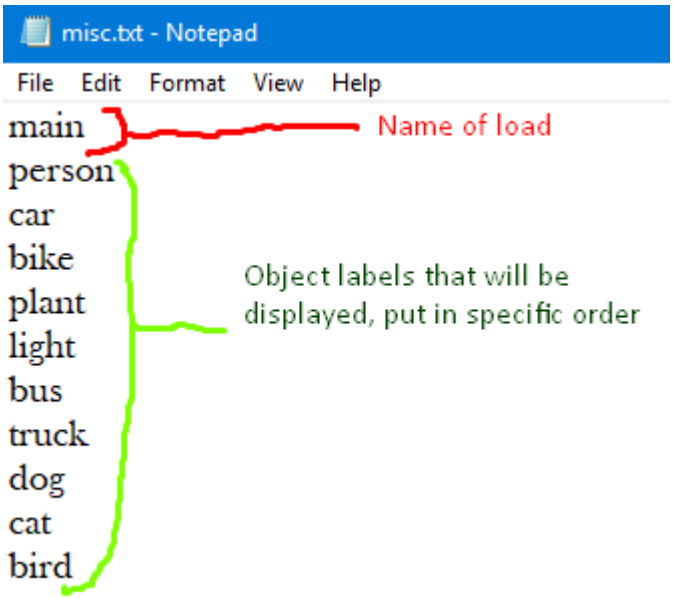
Listbox problem

As I was looking through my program, I wondered how will I make use of the listbox and I wanted to find out how many classes can the yolo algorithm detect, so I did some research and I found this diagram
https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/single-shot-detectors/yolo.html



As you can see in this diagram above there will be 20 different classes so 20 different objects to train, so I will make a seperate text file, to store the names of all class names and also a new button that allows the user to select which folder to load in which will be the main folder(data).
The data displayed on listbox would be the names of the objects so the user would know which classes are available and which load they have selected.

I personally made this with random objects as a template to how it should look like, it will be loaded in with the rest of the data files.



Listbox Code

So I made a 1d string array and set a value of 21.

```
static string[] objectlabel = new string[21]; //20 objects +1
```

20 of the spaces in this array were used for different objects and 1 for the main name of the load.

I later decided to set the destination of this file near the weight files.

```
var readmisc = File.ReadAllLines(location + "\\misc.txt");
```

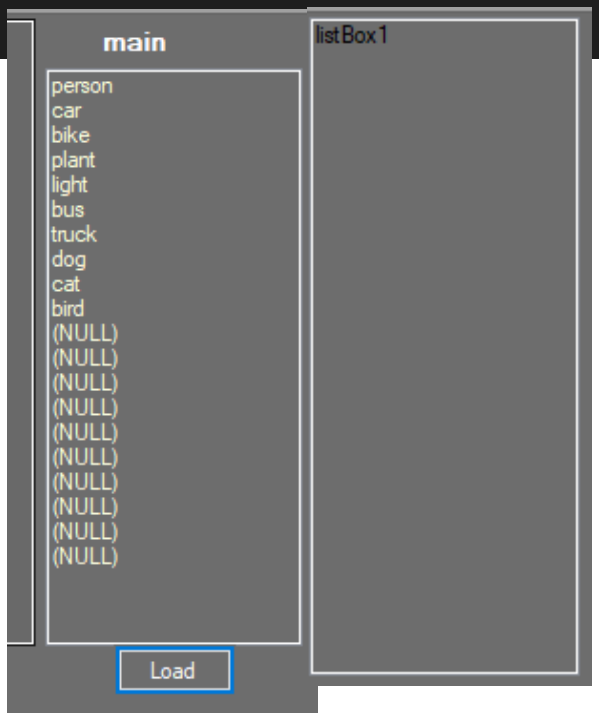
I later then programmed it to store the data into a global array which was defined above.

```
if(i < readmisc.Count())// loops around how many objects there are in a text file
{
    objectlabel[i] = readmisc[i];//stores it in another array
}
```

Lastly the values from the global array was placed into a listbox.

```
private void timer2_Tick(object sender, EventArgs e)
{
    if(listboxloaded == true)
    {
        listboxloaded = false;
        for(int i = 1; i< 21; i++)
        {
            if(objectlabel[i] == null)
            {
                listBox1.Items.Add("(NULL)");
            }
            else
            {
                listBox1.Items.Add(objectlabel[i]);//objects
            }
        }
        label1_objectfilename.Text = objectlabel[0]; // the main name of the file
        timer2.Enabled = false;
    }

}
```

As I ran my program and selected the folder, I successfully loaded all the object labels, since I didn't write the full 20 object, the rest of the spaces were set to (NULL) and this will later be set to ignore in my main image recognition.



Other changes

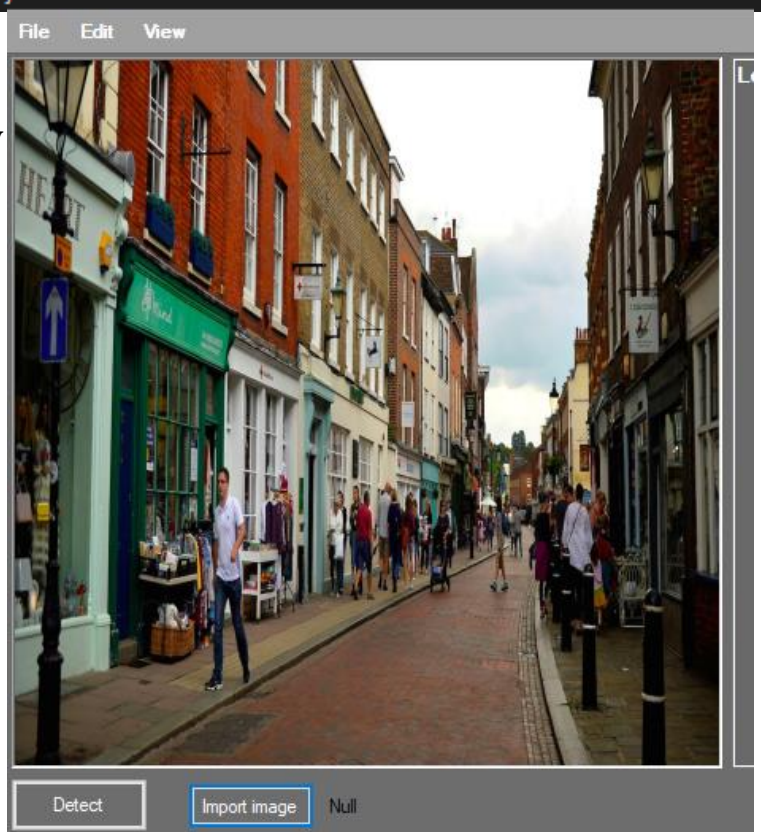
The other smaller changes to my program

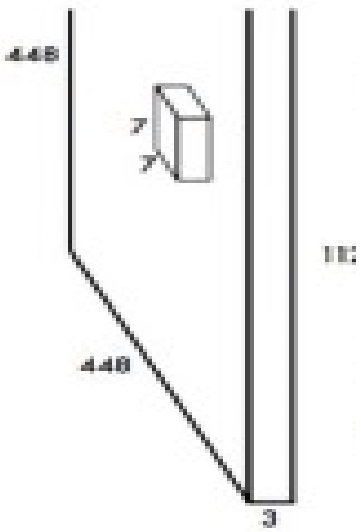
Added import button, to allow the user in importing their image.

This is the code I did for import image function and it selects any file but if the file is not an image then it will get rejected, but once an image is selected I had to make it so the image will be resized to a size of 448x448, since that's the size of an image used for the yolo algorithm.

```
private void button_import_Click(object sender, EventArgs e)
{
    MessageBox.Show("please select the image you want to import");
    OpenFileDialog fl = new OpenFileDialog();
    DialogResult rl = fl.ShowDialog();
    if(rl == DialogResult.OK)
    {
        try
        {
            bp = new Bitmap(fl.FileName);
            Bitmap resized = new Bitmap(bp, new Size(448, 448));

            pictureBox2_main.Image = resized;
        }
        catch (Exception)
        {
            MessageBox.Show("File cannot be loaded!");
        }
    }
    else
    {
        MessageBox.Show("Could not open the image!");
    }
}
```

Just to test out if my function works, I temporarily programmed import image to display the resized image.
As you can see the image is stretched and reason for that is because the image I used was 1576x788, so unless the image is exactly 448x448 it won't be stretched.





Version 0.1 Final testing

As I made a lot of changes to my program, this version sheet is outdated, since I thought that my image recognition will have a separate image classifier and object detector, but since I

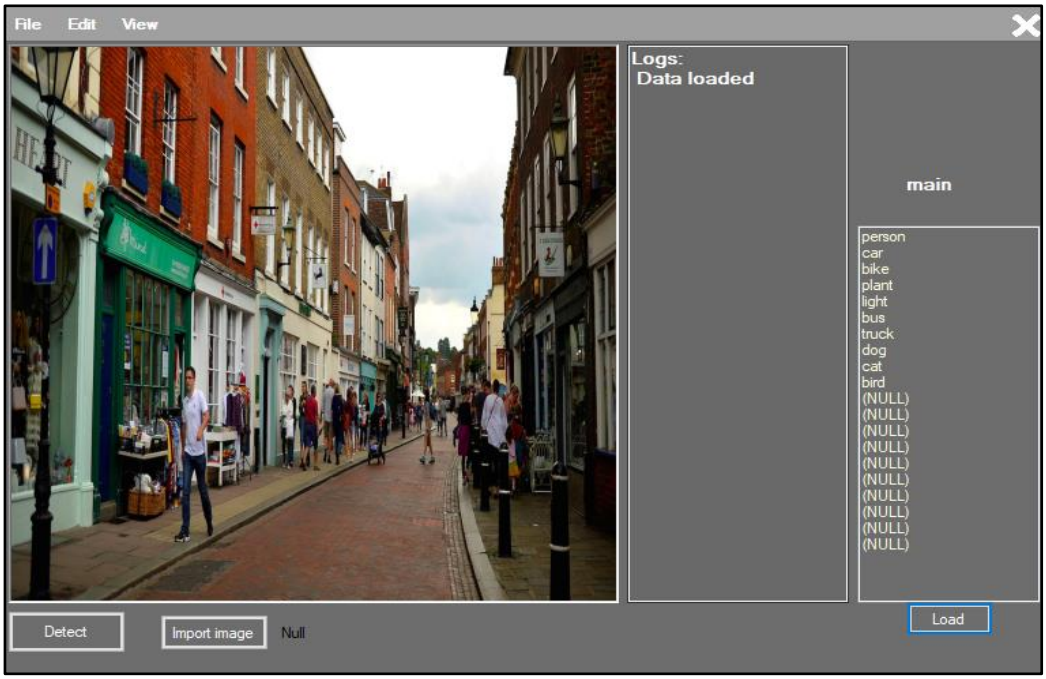
will be using yolo algorithm, image classifier and object detector is merged into one large neural network.

Version	Features	Testing	Date Due
0.1	GUI[buttons for input, Image boxes for output, labels for simple output.] Code[setup the required variables and program the buttons to display the output]	Click the buttons and see if basic output and input is functioning And if variables are setup	End of october

Was this version successful?

Yes, this version was successful as my program can now do more than what I planned it to do for version 0.1.

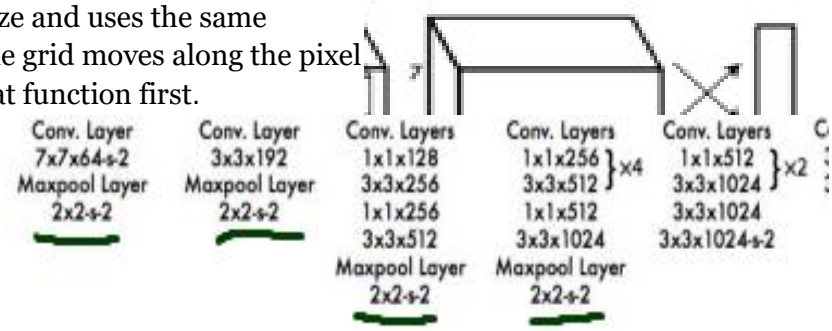
The basic GUI was changed than from how I first intended it to be, but since some certain changes, main picture box for example was stretched out since the image required for image recognition is exactly 448x448 in which changed the whole form size.



Version 0.2

In this version I will have to create a function to max pool and convolve the image into a specific size of 7x7x1024 which will then proceed to feed into the neural network for recognition and object detection.

If we look back at the yolo algorithm diagram we will see that max pooling is the same size and uses the same stride(2) which is by how far the grid moves along the pixel matrix so I decided to make that function first.



At page 18-19 I created pseudo code for the max pooling function and the main code something similar to this.

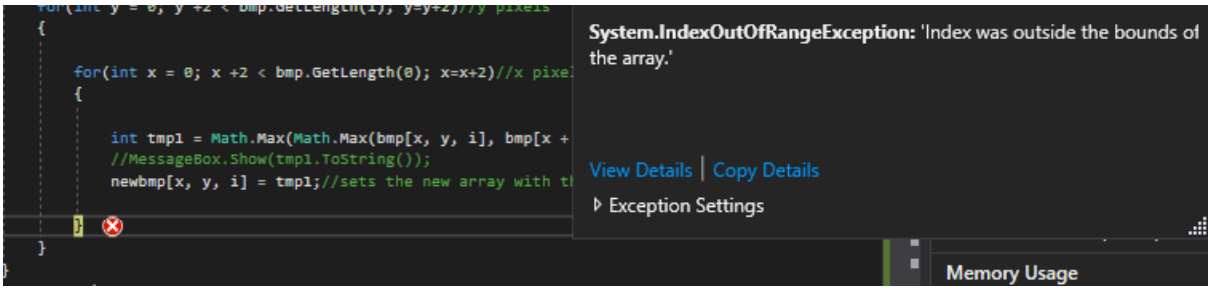
```
maxpooling(bitmap pictr)

        bitmap newpictr(pictr.width /2, pictr.height /2):

        for(int y : pictr.height/2 : ++y)

                for(int x : pictr.width/2 ++x)

                        temp = (pictr.getpixel[x,y] + [x+1,y] + [x,y+1] + [x+1,y+1])/4:
                        newpiictr.setpixel[x,y](temp):                    #sets the new bitmap with the pooled pixel

                end

        end

        return(newpictr):

end
```

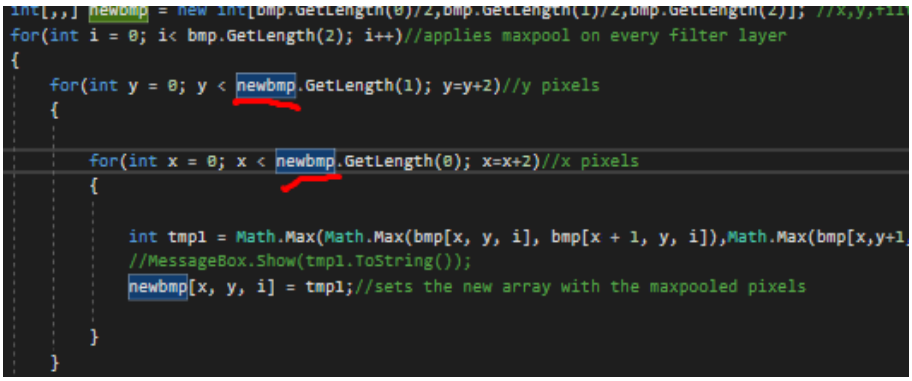
As the max pooling will be using arrays to store the pixels, I will have to use 3d integer arrays, for max pooling and convolution stage.

Code maxpool

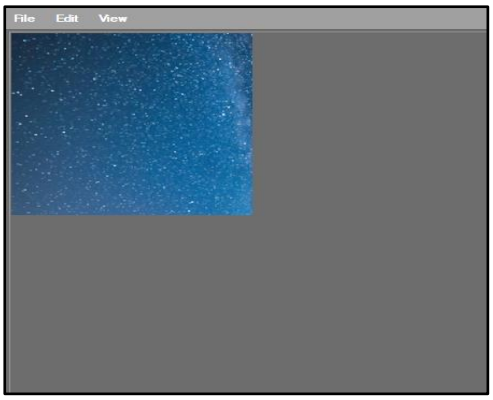
```
private int[,,] maxpool(int[,,] bmp)
{
    int[,,] newbmp = new int[bmp.GetLength(0)/2,bmp.GetLength(1)/2,bmp.GetLength(2)]; //x,y,filter
    for(int i = 0; i< bmp.GetLength(2); i++)//applies maxpool on every filter layer
    {
        for(int y = 0; y +2 < bmp.GetLength(1); y=y+2)//y pixels
        {

            for(int x = 0; x +2 < bmp.GetLength(0); x=x+2)//x pixels
            {

                int tmp1 = Math.Max(Math.Max(bmp[x, y, i], bmp[x + 1, y, i]),Math.Max(bmp[x,y+1,i],bmp[x+
                //MessageBox.Show(tmp1.ToString());
                newbmp[x, y, i] = tmp1;//sets the new array with the maxpooled pixels

            }
        }
    }
    return newbmp;
}
```

The max pooling function had some problems in the code above





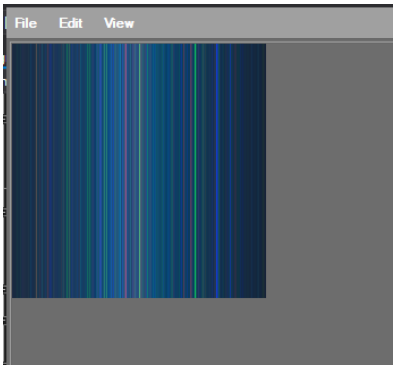
This one error, is usually thrown when I go over the size of the array
And this was the simple fix for the problem, since my for loops, runs the code the same amount of times as the size of bmp, which was 448 but the array size is half that(224) so I changed it to the size of newbmp, which is half the size of bmp and it worked, there wasn't any array errors, but as you fix a problem another one comes up.
Something else was wrong with the maxpool function, instead of scaling down the whole image it cropped out the rest and only left the top left side of the whole image.



Code maxpool pt2

As a method to fix that, I decided to add in 2 pointers, so instead of changing the values of the for loops, creating a seperate variable, that won't affect the amount of times, the for loops run.

```
int[,,] newbmp = new int[bmp.GetLength(0)/2,bmp.GetLength(1)/2,bmp.GetLength(2)]; //x,y,filter
int pointy = 0; //y pointer
int pointx = 0; //x pointer
for(int i = 0; i< bmp.GetLength(2); i++)//applies maxpool on every filter layer
{
    for(int y = 0; y < newbmp.GetLength(1); y++)//y pixels
    {

        for(int x = 0; x < newbmp.GetLength(0); x++)//x pixels
        {

            int tmp1 = Math.Max(Math.Max(bmp[pointx, pointy, i], bmp[pointx + 1, pointy, i]),Math.Max(bmp[pointx,pointy+1,i],bmp[pointx+1,pointy+1,i]));//
            //MessageBox.Show(tmp1.ToString());
            newbmp[x, y, i] = tmp1;//sets the new array with the maxpooled pixels
            pointx = pointx + 2;
        }
        pointx = 0;
    }   pointy = pointy + 2;
}
return newbmp;
```

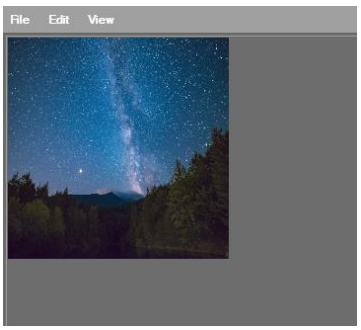


But as I debugged my program I ran into another bug.
The whole image becomes smudged, the logical reason for this might be that the maxpool function only, selected the first line of pixels and applied it throughout the whole image.

Then I found some small problems in my code, see bellow and after I did some minor changes again, my program worked just how I wanted it to.

```
private int[,,] maxpool(int[,,] bmp)
{
    int[,,] newbmp = new int[bmp.GetLength(0)/2,bmp.GetLength(1)/2,bmp.GetLength(2)]; //x,y,filter
    int pointy = 0; //y pointer
    int pointx = 0; //x pointer
    for(int i = 0; i< bmp.GetLength(2); i++)//applies maxpool on every filter layer
    {
        pointy = 0; Set back to 0 so no array problems
        for(int y = 0; y < newbmp.GetLength(1); y++)//y pixels
        {

            for(int x = 0; x < newbmp.GetLength(0); x++)//x pixels
            {

                int tmp1 = Math.Max(Math.Max(bmp[pointx, pointy, i], bmp[pointx + 1, pointy, i]),Math.Max(bmp[pointx,pointy+1,i],bmp[point
                //MessageBox.Show(tmp1.ToString());
                newbmp[x, y, i] = tmp1;//sets the new array with the maxpooled pixels
                pointx = pointx + 2;
            }
            pointy = pointy + 2;
            pointx = 0;
        }
        Before
    }
    return newbmp;
}
```

The whole maxpool functions just how I wanted it to, it makes the whole image twice as smaller just how it was displayed on the yolo algorithm diagram, so I can say that my maxpool function is now complete.



Code convolution errors and fix

This function will be slightly more complex than the max pool function as size of the grid will change and so will the stride on different points of the algorithm.

At first I needed to store an rgb 448x448 image into an array
So this function simply stores the pixel value of the image into a 448x448x64 integer 3d array and I did this by storing the red pixels, green pixels and blue pixels in a specific order.

```
{
    int[,,] img = new int[448, 448, 64];//448x,448y,rgb
    int ttmp = 0;
    for(int f = 0; f<64; f++)
    {
        for(int y = 0; y<448; y++)
        {
            for(int x = 0; x<448; x++)
            {
                Color cl = bp.GetPixel(x, y);
                switch (ttmp)
                {
                    case 0:
                        img[x, y, f] = cl.R;
                        ttmp++;
                        break;

                    case 1:
                        img[x, y, f] = cl.G;
                        ttmp++;
                        break;

                    case 2:
                        img[x, y, f] = cl.B;
                        ttmp = 0;
                        break;

                }

            }
        }
    }
```

Red pixels are stored in
(0,3,6,9,12,15,18,21 etc)
whilst the green pixels are stored in
(1,4,7,10,13,16,19,22 etc)
And lastly the Blue pixels were stored in
(2,5,8,11,14,17,20,23 etc).

Since neural networks can learn and adapt, they should be able to understand and find features that can be helpful for them to recognise the objects.

As I was thinking, doing that convolution in the same loop would be less computationally expensive, so I would create the first convolution step inside of that same for loop.

```
}
int tempor = 0; //temporary value to store the convolved value which will be passed into the array
for(int yy = 0; yy< 7;yy++)
{
    for(int xx = 0; xx < 7; xx++)//multiply image by filters
    {
        tempor += filter_64_7x7[f, xx, yy] * img[x + xx, y + yy, f];
    }
}
//MessageBox.Show(tempor.ToString());
img[x, y, f] = tempor;
```

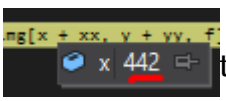
Exception Unhandled

System.IndexOutOfRangeException: 'Index was outside the bounds of the array.'

View Details | Copy Details

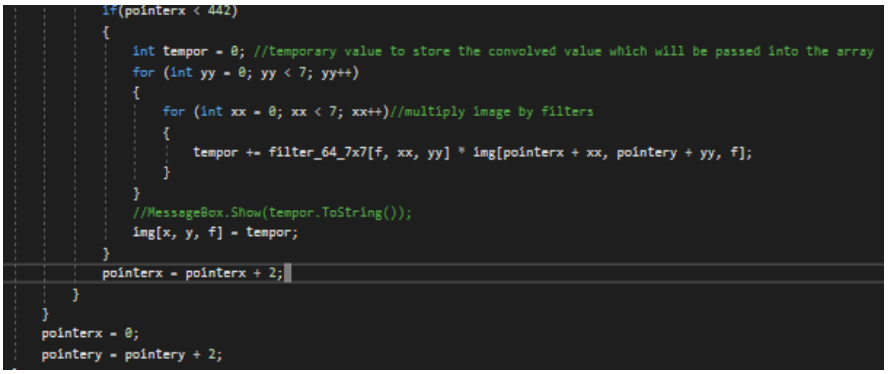
Exception Settings

As I coded the part for multiplying the filter and the image section I came into a slight common problem.
So as a method to fix this problem I needed to check each value and see what is making that error.



Since visual studio has a really useful debugger I was able to find out that x had the largest value of 442 which might be causing the problem, but as I realised, the stride has to also be by 2, making the output array 2x smaller.

So I applied a simple fix of adding 2 pointers(x and y) and an if statement to make sure that the pointer doesn't go over 442.



```
            if(pointerx < 442)
            {
                int tempor = 0; //temporary value to store the convolved value which will be passed into the array
                for (int yy = 0; yy < 7; yy++)
                {
                    for (int xx = 0; xx < 7; xx++)//multiply image by filters
                    {
                        tempor += filter_64_7x7[f, xx, yy] * img[pointerx + xx, pointery + yy, f];
                    }
                }
                //MessageBox.Show(tempor.ToString());
                img[x, y, f] = tempor;
            }
            pointerx = pointerx + 2;
        }
    }
    pointerx = 0;
    pointery = pointery + 2;
```

Code convolution errors and fix part 2

After running the code I discovered another problem which looked familiar and it was the same problem as the one on maxpool code, so I looked at the code to see what I might of done wrong and found one small mistake which was that the pointer y increments at the wrong place, so I changed it so it would be incremented once the x loop finishes instead of the y loop.

So I fixed the first problem but then later found another one, which I already knew the solution to it.
The first problem was the out of bounds array, which simply means that pointers are either too low or too big, in this case it was too big so I added an if statement to make sure that the pivots don't go higher than the size of the array.

```
            img2[x, y, f] = tempor;
        }
        pointerx = pointerx + 2;
    }
}
pointerx = 0;
pointery = pointery + 2;
```

Then I had another problem as once I ran it nothing new happened but output the same image as last time (see above) so I then later realised that the function was returning(img) rather than(img2) so I quickly changed it and then found another problem.

```
            if (pointerx < 442 && pointery < 442)
            {
                int tempor = 0; //temporary value to store the convolved value which will be passed into the array
                for (int yy = 0; yy < filter_64_7x7.GetLength(2); yy++)
                {
                    for (int xx = 0; xx < filter_64_7x7.GetLength(1); xx++)//multiply image by filters
                    {
                        tempor += filter_64_7x7[f, xx, yy] * img[pointerx + xx, pointery + yy, f];
                    }
                }
                //MessageBox.Show(tempor.ToString());
                img2[x, y, f] = tempor;
                pointerx = pointerx + 2;
            }
        }
        pointerx = 0;
        pointery = pointery + 2;
    }
}


return img2;
```

So after some thinking, I realised that converting the bitmap into an array will take that whole loop to complete and that the main convolution part was taking the array values before they even managed to be set from the image, so the main solution for this would be to create a seperate loop, so the first main loops can be finished.

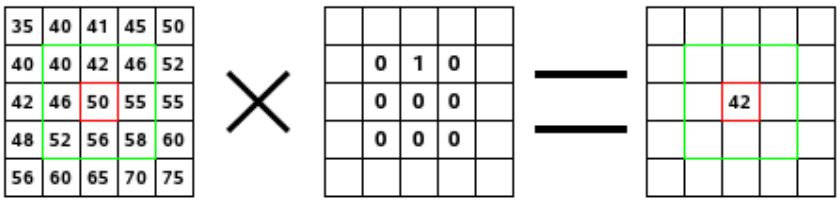
Finally after some minor bug fixing, my program has convolved an image, but one problem with this is that the convolved image is dark and hard to see, so there must be another problem within the code that is causing the convolved image to be dark.

Code convolution errors and fix part 3

As you can see I did some changes to fix some basic problems

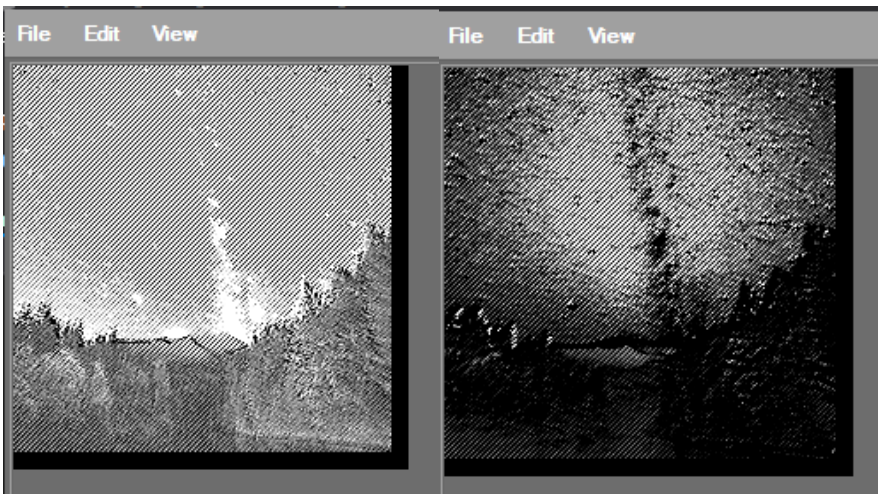
```
for(int f = 0; f<64; f++)
{
    for(int y = 0; y<442; y++)
    {
        for(int x = 0; x<442; x++)
        {
            int tempor = 0; //temporary value to store the convolved value which will be passed into the array
            for (int yy = 0; yy < filter_64_7x7.GetLength(2); yy++)
            {
                for (int xx = 0; xx < filter_64_7x7.GetLength(1); xx++)//multiply image by filters
                {
                    tempor += filter_64_7x7[f, xx, yy] * img[x + xx, y + yy, f];
                }
            }
            x++;
            tempor = tempor / 49; // averages the convolved value
            img2[pointerx, pointery, f] = tempor;//sets the final array with the convolved values
            pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis

        }

        pointerx = 0; //sets pointer x back to 0
        pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis
         y++;
    }
    pointery = 0;
    pointerx = 0;


}
return img2;
```

But as I was looking through my code and looked up how convolution works again I found a small mistake that I have made.
Convolution multiplies the numbers in the image and filter and then it adds them all together, but for my code, I averaged the numbers at the end.
So the simple fix for this is to remove the division part.




```
    {
        tempor += filter_64_7x7[f, xx, yy] * img[x + xx, y + yy, f];
    }
}
x++;
tempor = tempor / 49; // averages the convolved value
```

And just that one small change fixed the convolvolution function.



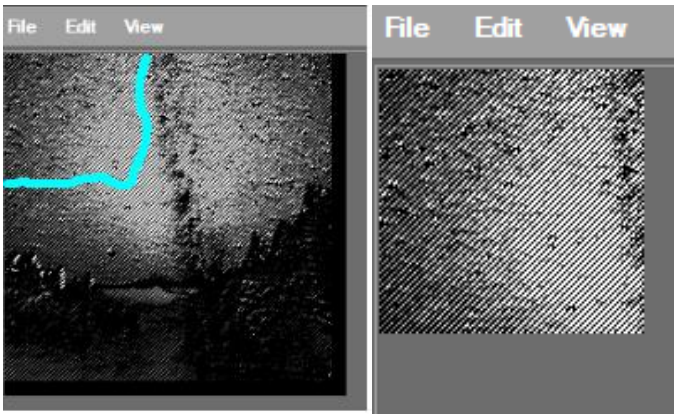
Just to see that it works perfectly, I chose a different filter and as you can see it detected different features than to the one that the 1st one did.

So now that the convolution part is working I will need to link it to the max pooling feature and make many more convolution functions so it can fully abstract the image into a 7x7x1024

Linking Convolution and maxpool problem

As I implemented the max pooling function again and tested it on the same image again, I found out that the problem might be the max pooling function, so I tried to figure out the problem, and as it looks like the max pooling feature only just cropped the top left part of the convolved image.

```
int[,,] tmp;
Bitmap bmp = new Bitmap(pictureBox2_main.Image);
int[,,] tempor = convolution64_7x7(bmp);
tmp = maxpool(tempor);
```



Yet again, the max pooling function should be working as I tested it out before and it was successfully working with images.

So then I was trying to figure out what was causing the problem and I decided to check if the temporary code to display the convolved images, as we won't see the convolved images because they would be stored in an integer array that would be passed on to the neural network for image recognition.

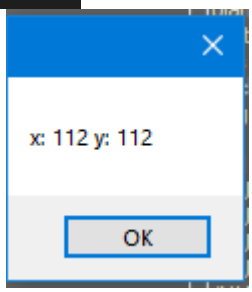
So as I took a look at that temporary code, I did not notice anything at first, but as I re read the whole code to see what might be causing the problem, I noticed a slight mistake.
These 7 small mistakes, instead of the pixel values from tmp being converted into an image, pixel values from tempor were being converted and that also explains why the image was only cropped out,
because as you can see in the for loop, it would only the number of times to the tmp size, which is half of tempor size.

```
private void button1_detect_Click(object sender, EventArgs e)
{
    int[,,] tmp;
    Bitmap bmp = new Bitmap(pictureBox2_main.Image);
    int[,,] tempor = convolution64_7x7(bmp);
    tmp = maxpool(tempor);

    Bitmap tmppp = new Bitmap(tmp.GetLength(1), tmp.GetLength(0));
    for(int y = 0; y < tmp.GetLength(1); y++)
    {
        for(int x = 0; x< tmp.GetLength(0); x++)
        {
            if(tempor[x,y,17] > 255)
            {
                tempor[x, y, 17] = 255;
            }
            if(tempor[x,y,17] < 0)
            {
                tempor[x, y, 17] = 0;
            }
            Color cl = Color.FromArgb(tempor
            tmppp.SetPixel(x, y, cl);
        }
    }
    pictureBox2_main.Image = tmppp;
}
```

So as I tested my program, it was working successfully as the whole convolved and max pooled image was displayed.
Since linking the max pool and convolution worked, I need to test one last thing before continuing with doing the rest of the algorithm.
I need to check the size of the outputted array.

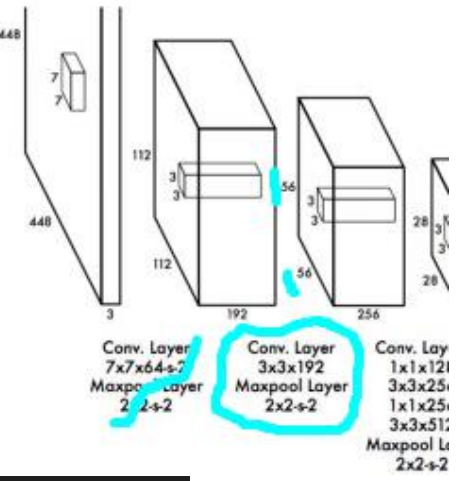
```
MessageBox.Show("x: " + tmp.GetLength(0).ToString() + " y: " + tmp.GetLength(1).ToString());
```

So I implemented this small code into my program which simply shows a message box of the size of the array



And as you can see it is exactly 112, which is exactly the size that it needs to be, according to the algorithm.

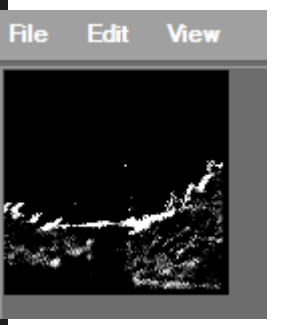
Convolution algorithm code



So as max pooling and convolution is working just how I wanted it to work, I will need to now continue creating the rest of the functions for image abstraction part of the code.
The next convolution size is 3x3x192 with just a stride of 1, so the easiest way of doing this is to copy the last function and edit it so it would suit this convolution.
So if we do some simple math, the convolution should not affect the size of the array but the maxpool would since 112/2 = 56, so this means that i'll need to make sure that the convolution function does not affect the array of the image pixel values.

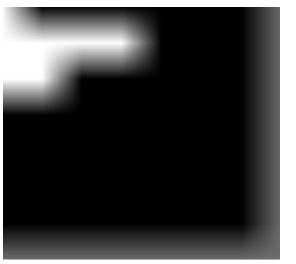
As I copied the last convolution function and edited it to suit this new one, we can see it working perfectly as the image became more abstract.
So now all I have to do is to apply the same thing throughout the whole algorithm.

```
private static int[,,] convolution192_3x3(int[,,] img)
{
    int[,,] img2 = new int[img.GetLength(0), img.GetLength(1), img.GetLength(2)];//224
    int pointerx = 0;
    int pointery = 0;
    int tmp = 0;
    for (int f = 0; f < img.GetLength(2); f++)
    {
        for (int y = 0; y < img.GetLength(1)-3; y++)
        {
            for (int x = 0; x < img.GetLength(0)-3; x++)
            {
                int tempor = 0; //temporary value to store the convolved value which will be passed into the array
                for (int yy = 0; yy < filter_192_3x3.GetLength(2); yy++)
                {
                    for (int xx = 0; xx < filter_192_3x3.GetLength(1); xx++)//multiply image by filters
                    {
                        tempor += filter_192_3x3[f, xx, yy] * img[x + xx, y + yy, tmp];

                    }
                }

                img2[pointerx, pointery, f] = tempor;//sets the final array with the convolved values
                pointerx++;//increments pointerx in which tells where in array would we store the convolved value x axi
            }
            pointerx = 0; //sets pointer x back to 0
            pointery++; //increments pointery in which tells where in array would we store the convolved value y axis

        }
        pointery = 0;
        pointerx = 0;
        tmp++;
        if (tmp >= 64)
        {
            tmp = 0;
        }
    }
    return img2;
}
```



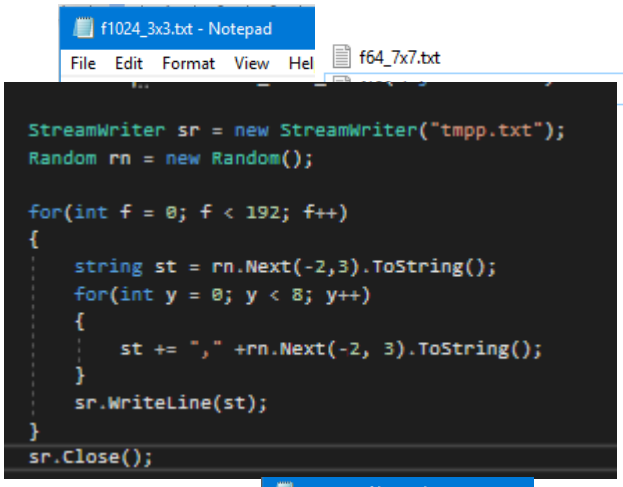
As I finished programming the rest of convolution functions, I decided to test it out and without any problems, it seemed to be working, but since the image would be really small, I enlarged it so it would be easier to see.
But one problem was, when I checked on filter 2, it was completely black, so out of curiosity, I opened the filter file and looked at filter 2
So I first thought that maybe it was just random, but as I took a look at all of the other filter values, they were all either 0 or -1 or -2, but no positives, which was a problem as the filters need to be between 2 and -2, so I checked the other filters to see if they are also affected and unfortunately, they were too affected but only the 3x3 filters, this suggest that something was wrong with my filter generation function, as I couldn't tell that there would be a problem with the filters

-1,-1,-2,0,-2,-2,-2,0,-1

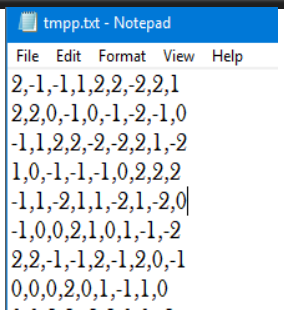


before until later in the stage of development, I successfully identified hopefully the last problem with the filters.

Filter problem fix

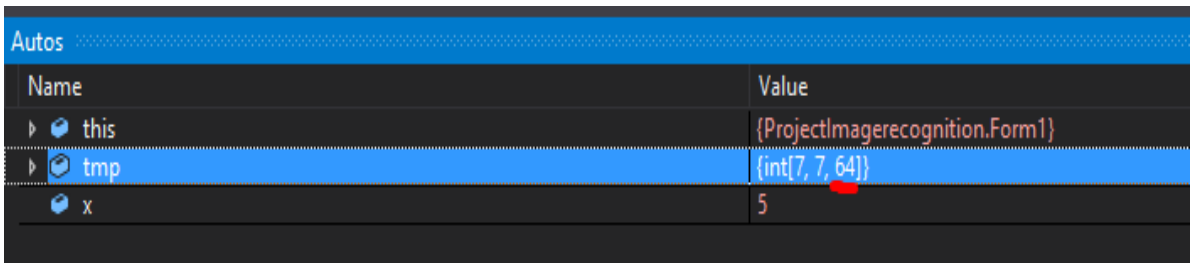


As the problem only affect a couple of filters, I can simply re create the filter generation function to generate a new set of filters. Since this program can load in any data folder with filters and weights, there would not be a problem if we change the filters.
So as I made this temporary code to just create a text file called tmpp.txt which should contain the 3x3x192 filters, which should be random.
As I ran the code, that small function created the text file and successfully made the filters random just how I wanted it to work.

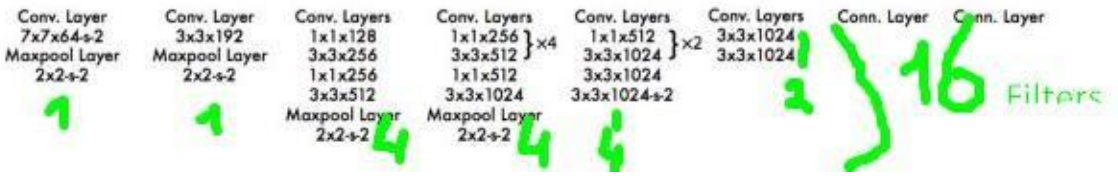


So now I just replaced the filter numbers with the ones from tmpp.txt and I just changed the f value in the for loop for the rest of the filters and did the same steps.

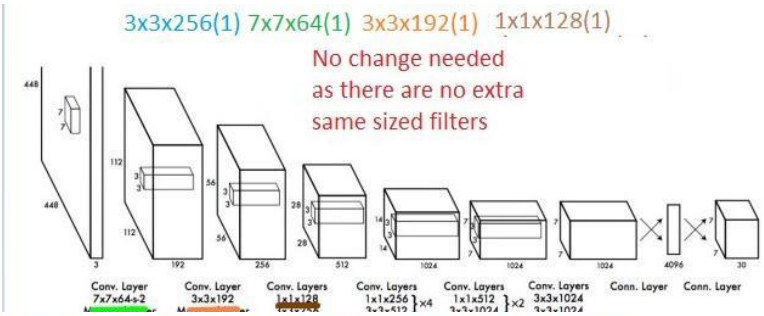
As I was debugging I ran into another slight problem



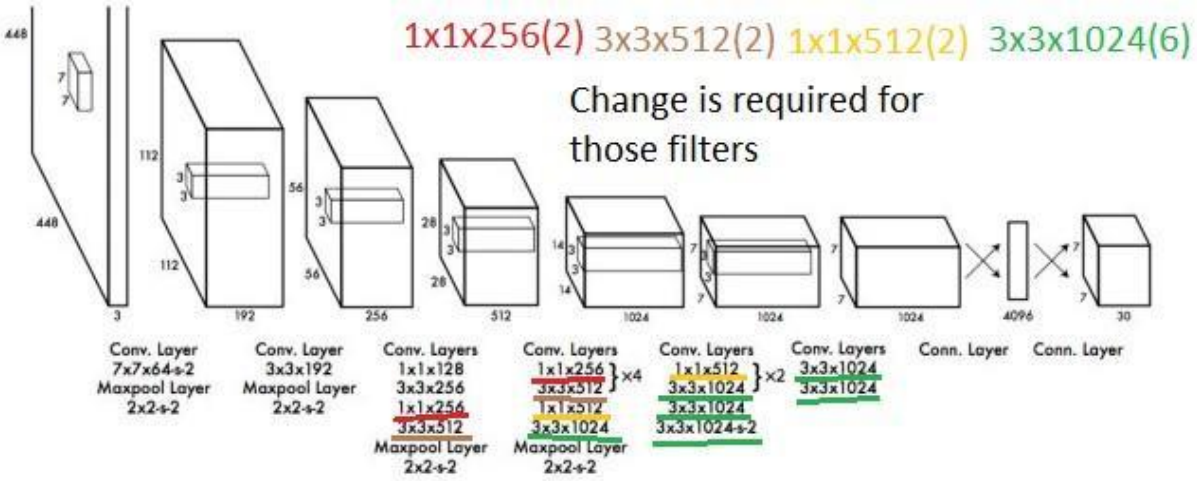
The problem is that the final convolved array is a 7x7x64 and not 3x3x1024
So the problem might be in the convolution functions, where it resizes the array from 64 filters to 192 and the reason for that is since I re used the same function but slightly adjusted it to suit the filter. So there would probably be the same problem for every convolution function apart from the 64.



So as we can see in the diagram, there would be 16 different filters but the filters in }x num are re used to the amount set.
Now to workout the filters needed to be changed, we first find the ones which don't need to be changed as they would be used only once and are already set up.



Filter problem fix pt2



These are the following filters that need to be changed, which would just mean to add in extra filters.

The main method for this would be to simply, make a 4d array for the filters that need changing.
So instead of the filter array representing [filternum,x,y] it does [filternum,x,y,filterarray] the other filters that didn't need changing are the same size and won't be changed as it's not needed to fix this problem.

```
static int[,,,] filter_512_3x3 = new int[512, 3, 3,2];
static int[,,,] filter_1024_3x3 = new int[1024, 3, 3,6];
```

Reason for why we need more different filters

Since we are using random filters, re using the same ones more than once would be a bad idea, as there is a bigger chance of it generating more 0's on images and always staying a 0 no matter the image and that is a problem since the image recognition would be less accurate and not functioning as well as it should.

So the best method in how we would use those extra filters is by adding even more filters into the filters text files. In other words,as filter 512x3x3 is affected, and only

needs 1 more extra layer, we would double the filter amount in the file, so instead of there being 512 filters in the text file there would be 1024.

Filter problem code

```
if (i < 1024)//doubled because of filter problem
{
    if(tem3 >= 512)
    {
        tem3 = 0;//makes sure that tem3 doesn't go over 512 which is the maximum
        tem3_1++;//increments the filter layer pivot
    }
    filter_512_1x1[tem3,tem3_1] = Convert.ToInt32(readf512_1x1[i]);//readf512_1x1 size is 1024 but it will read the first half if -
    //tem3_1 = 0 and would read the last half if tem3_1 = 1
    int tmpp = 0;
    string[] word = readf512_3x3[i].Split(',');//splits the filter values when it finds a ,
    for (int y = 0; y < 3; y++)
    {
        for (int x = 0; x < 3; x++)
        {
            filter_512_3x3[tem3, y, x,tem3_1] = Convert.ToInt32(word[tmpp]);
            tmpp++;
        }

    }
    tem3++;
}
```

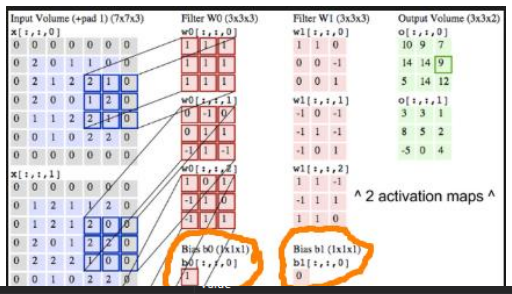
As you can see this would be very useful as I used same array to fix this problem instead of creating a seperate array.
So as I ran my program to see if extra filters have reduced the number of zero's which zero's would make the code slightly less accurate because multiplying anything by a zero would return a zero, which is why reducing the need for zero's is important, but as I checked the array values, as you can see there are still a lot of zero's.

Name	Value
tmp	{int[7, 7, 1024]}
[0, 0, 0]	0
[0, 0, 1]	0
[0, 0, 2]	0
[0, 0, 3]	0
[0, 0, 4]	0
[0, 0, 5]	0
[0, 0, 6]	0
[0, 0, 7]	-105015136
[0, 0, 8]	0
[0, 0, 9]	1042834432
[0, 0, 10]	717309952
[0, 0, 11]	0
[0, 0, 12]	0
[0, 0, 13]	0
[0, 0, 14]	0
[0, 0, 15]	0
[0, 0, 16]	0
[0, 0, 17]	0
[0, 0, 18]	0
[0, 0, 19]	0
[0, 0, 20]	-1025736512

After some research I discovered this diagram again, and one thing that I missed out was the bias and the bias can simply increment the number so in case if the number would be 0 the bias would make it into a 1. Since that diagram used a different bias, I thought that I would randomly set the bias for each filter one being 0 whilst the other 1.
So for random convolution functions I inserted this small code in order for the bias to work and reduce the number of 0's because even though there were still numbers which weren't 0's the recognition would be

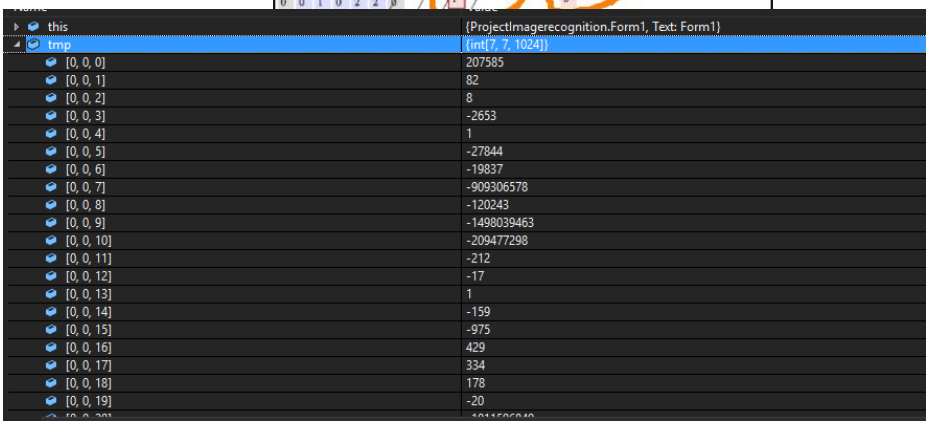
```
    }
    x++;
    tempor++;
    img2[pointerx, pointery, f] = tem
    pointerx++;//incremeants pointerx
}
```

really inaccurate as the neural network won't receive better input data.



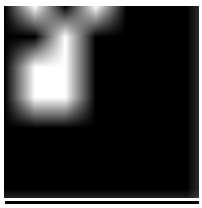
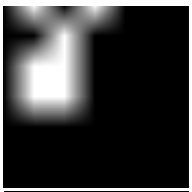
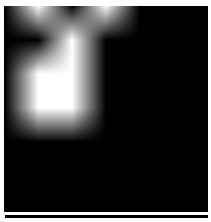
Convolution testing

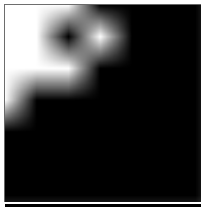
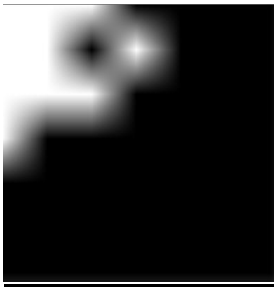
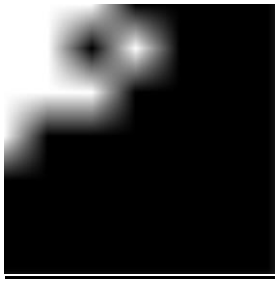
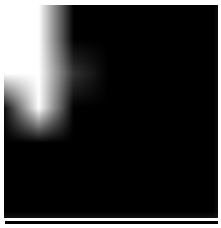
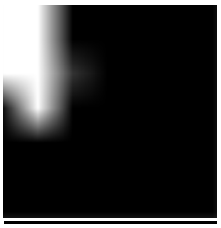
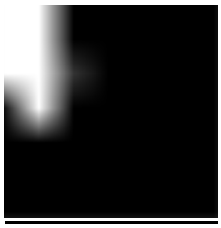
As you can see there is a huge difference in result just by using a bias, so I can confirm that the bias convolution stage of my program is done and ready for testing.



this	{ProjectImagerecognition.Form1, Text: Form1}
tmp	{int[7, 7, 1024]}
[0, 0, 0]	207585
[0, 0, 1]	82
[0, 0, 2]	8
[0, 0, 3]	-2653
[0, 0, 4]	1
[0, 0, 5]	-27844
[0, 0, 6]	-19837
[0, 0, 7]	-909306578
[0, 0, 8]	-120243
[0, 0, 9]	-1498039463
[0, 0, 10]	-209477298
[0, 0, 11]	-212
[0, 0, 12]	-17
[0, 0, 13]	1
[0, 0, 14]	-159
[0, 0, 15]	-975
[0, 0, 16]	429
[0, 0, 17]	334
[0, 0, 18]	178
[0, 0, 19]	-20

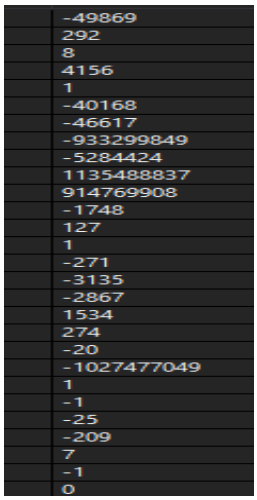
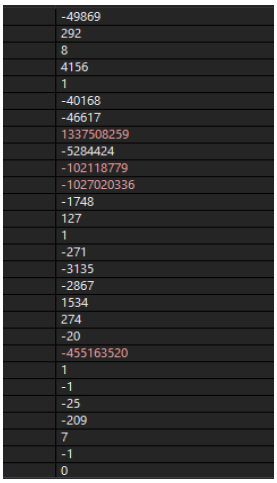
Testing image abstraction

Filter number	Image	Image 2	Image 3
None			
0			

1			
2			

Convolution problem

As the test data shows, each image displayed the same output in which suggests that something might be wrong with the convolution function, so in order to see what is causing this I decided to take out the bias and compare the result.

	Image1 output snippet	Image2 output snippet
With bias		

Without bias	0 0 0 0 0 0 0 -105015136 0 1042834432 717309952 0 0 0 0 0 0 0 0 0 -1025736512 0 0 0 0 0 0 0	0 0 0 0 0 0 0 173494656 0 794688768 -1869937600 0 0 0 0 0 0 0 0 0 -453538688 0 0 0 0 0 0 0

Convolution problem pt2

As of the convolution problem, I decided to try and find the problem within my code by experimenting with the convolution filters. So I passed on the image through many of the same filters to see if the same problem would happen again

```
int[,,] tmp;
Bitmap bmp = new Bitmap(pictureBox2_main

int[,,] tempor = convolution64_7x7(bmp);
tempor = maxpool(tempor);
//MessageBox.Show(tempor.GetLength(2).To
tmp = convolution192_3x3(tempor);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = maxpool(tmp);
//tempor = maxpool(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
tmp = convolution192_3x3(tmp);
```

the

So the following output is this:

		(white image)

[0, 0, 0]	-178023812(
[0, 0, 1]	-111200643
[0, 0, 2]	2004968644
[0, 0, 3]	-191143114
[0, 0, 4]	931392328
[0, 0, 5]	-131111649
[0, 0, 6]	1580124834
[0, 0, 7]	-128625759
[0, 0, 8]	-140407603
[0, 0, 9]	1977478839
[0, 0, 10]	-949744489
[0, 0, 11]	283436907
[0, 0, 12]	-129845280
[0, 0, 13]	-523677856
[0, 0, 14]	-868314234
[0, 0, 15]	363218800
[0, 0, 16]	-407285623
[0, 0, 17]	-190183757
[0, 0, 18]	-123763097
[0, 0, 19]	620467114
[0, 0, 20]	411377263

[0, 0, 0]	-200676823
[0, 0, 1]	337161606
[0, 0, 2]	-1410815086
[0, 0, 3]	-1443479253
[0, 0, 4]	-1284411164
[0, 0, 5]	2054965725
[0, 0, 6]	-70029797
[0, 0, 7]	2133254879
[0, 0, 8]	509919232
[0, 0, 9]	-161519376
[0, 0, 10]	-1418356048
[0, 0, 11]	-844365714
[0, 0, 12]	-1304677775
[0, 0, 13]	857019611
[0, 0, 14]	1431003573
[0, 0, 15]	-953636383
[0, 0, 16]	-1863264509
[0, 0, 17]	1533049757
[0, 0, 18]	-1300611072
[0, 0, 19]	569400252
[0, 0, 20]	-150545489

[0, 0, 0]	-1052999680
[0, 0, 1]	2040
[0, 0, 2]	34553
[0, 0, 3]	268566528
[0, 0, 4]	-1219657093
[0, 0, 5]	0
[0, 0, 6]	28697814
[0, 0, 7]	478991230
[0, 0, 8]	20922368
[0, 0, 9]	1637455099
[0, 0, 10]	1021406897
[0, 0, 11]	62221
[0, 0, 12]	0
[0, 0, 13]	16711680
[0, 0, 14]	-1275
[0, 0, 15]	-1073741824
[0, 0, 16]	-1073741824
[0, 0, 17]	-8290290
[0, 0, 18]	0
[0, 0, 19]	1067952324
[0, 0, 20]	0

As we can see, the output numbers change for each picture and that is exactly what we needed, this is the type of output that we should be seeing. How image recognition works, is it will find a pattern in these numbers, something that we as humans can't find, the neural network will identify the objects within the picture just with these types of numbers.

Convolution problem pt3

After removing and adding extra convolutions I came to realisation that filters with a 1x1 grid create this convolution problem, so after some thinking of why would it produce that error, I

just realised that 0's are the main problem and since those filters contain some 0's instead of a regular filter when they mutliply the numbers and add them up

	Value
[0, 0, 0]	765712576
[0, 0, 1]	-495545806
[0, 0, 2]	-467242224
[0, 0, 3]	-492627598
[0, 0, 4]	1615198795
[0, 0, 5]	1648921570
[0, 0, 6]	-1629081804
[0, 0, 7]	813379682
[0, 0, 8]	-944386290
[0, 0, 9]	747249703
[0, 0, 10]	581102666
[0, 0, 11]	312302884
[0, 0, 12]	-4209186
[0, 0, 13]	-1002805624
[0, 0, 14]	-121406714
[0, 0, 15]	140481644
[0, 0, 16]	-124009961
[0, 0, 17]	-1162040673
[0, 0, 18]	-36228074
[0, 0, 19]	941598596
[0, 0, 20]	-619739446
[0, 0, 21]	-951978298

```
tmp = convolution128_1x1(tmp);
tmp = convolution128_1x1(tmp);
tmp = convolution128_1x1(tmp);
tmp = convolution128_1x1(tmp);
*/
tmp = convolution256_3x3(tmp);
//tmp = convolution256_1x1(tmp,0);
tmp = convolution512_3x3(tmp,0);
 tmp = maxpool(tmp);

// tmp = convolution256_1x1(tmp,1);
// tmp = convolution256_1x1(tmp,1);//ERROR sets array to
//  tmp = convolution256_1x1(tmp,1);//4 times same filte
//  tmp = convolution256_1x1(tmp,1);
  tmp = convolution512_3x3(tmp,1);
  tmp = convolution512_3x3(tmp,1);//4 times same filter
 tmp = convolution512_3x3(tmp,1);
 tmp = convolution512_3x3(tmp,1);
// tmp = convolution512_1x1(tmp,0);
tmp = convolution1024_3x3(tmp,0);
tmp = convolution256_3x3(tmp);
tmp = convolution256_3x3(tmp);
tmp = maxpool(tmp);

// tmp = convolution512_1x1(tmp,1);
 //tmp = convolution512_1x1(tmp,1);
tmp = convolution1024_3x3(tmp,1);
tmp = convolution1024_3x3(tmp,1);
tmp = convolution1024_3x3(tmp,2);
tmp = convolution1024_3x3_s2(tmp,3);

tmp = convolution1024_3x3(tmp,4);
tmp = convolution1024_3x3(tmp,5);
```

Eg: 3x3 filter

Filter Image

1 0 0 30 2 20

2 2 2 X 10 0 10 = (1x30) + (0x2) + (0x20) + (2x10) + (2x0) + (2x10) + (1x5) + (1x11) +

1 1 0 5 11 12 (0x12) = 138

Eg: 1x1 filter

Filter Image

0 30 2 20

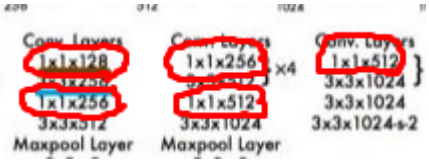
X 10 0 10 = (0x30) + (0x2) + (0x20) etc = 0

5 11 12

So as you see this is a simple math problem, which leaves static gaps in output, so whatever the image might be, that space would always be a 0, the reason for why not all of the output is a 0 is that in this 1x1 filter it is filled up with 2's 1's 0's -1's and -2's. One method to get rid of this problem is to simply not use 0's in 1x1 convolution filters.

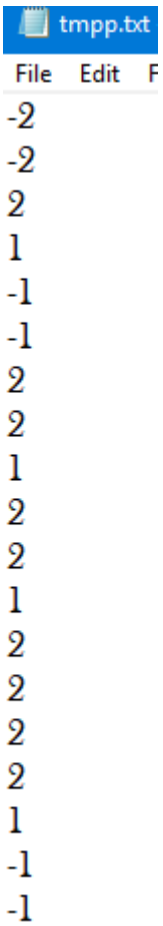
Convolution problem Fix

So the first step into fixing this issue is to identify the convolutions that need to be changed, since there is 5 of them, I will need to randomly generate 5 new series of filters.



```
StreamWriter sr = new StreamWriter("tmpp.txt");//temporary file
Random rn = new Random();

for(int f = 0; f < 128; f++)
{
    int num = rn.Next(-2, 3);//random number generated
    while(num == 0)//will loop until the random number isn't a 0
    {
        num = rn.Next(-2, 3);//number randomly generated
    }
    sr.WriteLine(num.ToString());//once there aren't any 0's the number gets written into the filter.
}
sr.Close();//saves the file
```

So as I coded the random filter generator I decided to manually place the random filters into their files.

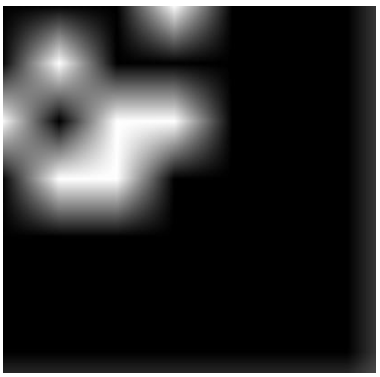


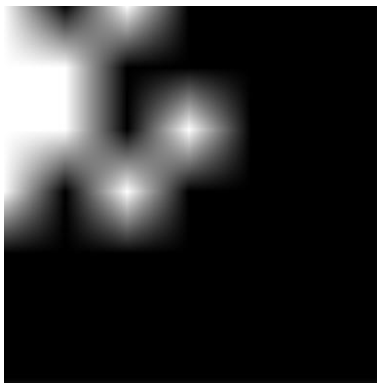
Convolution testing

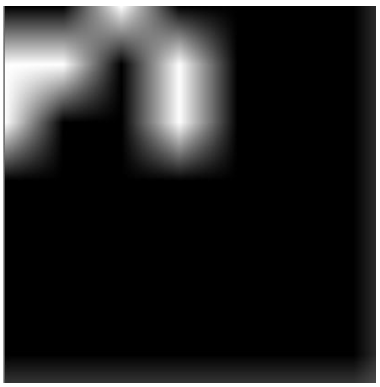
[0, 0, 0]	-2137851200
[0, 0, 1]	1446286464
[0, 0, 2]	1161153536
[0, 0, 3]	1111919744
[0, 0, 4]	-1329324800
[0, 0, 5]	965038736
[0, 0, 6]	-672465608
[0, 0, 7]	-1877986688
[0, 0, 8]	1959739776
[0, 0, 9]	-403277968
[0, 0, 10]	-401244824
[0, 0, 11]	-1779138688
[0, 0, 12]	752281024
[0, 0, 13]	2113296384
[0, 0, 14]	-1086275904
[0, 0, 15]	1497363232
[0, 0, 16]	425659968
[0, 0, 17]	-1509736476
[0, 0, 18]	275794368
[0, 0, 19]	-1550075264

[0, 0, 0]	542751808
[0, 0, 1]	-1862102400
[0, 0, 2]	202813440
[0, 0, 3]	-1441269888
[0, 0, 4]	856876800
[0, 0, 5]	-876156656
[0, 0, 6]	-1177031154
[0, 0, 7]	922841216
[0, 0, 8]	1388910976
[0, 0, 9]	-726227248
[0, 0, 10]	-209678448
[0, 0, 11]	-486085888
[0, 0, 12]	-644286848
[0, 0, 13]	-380547072
[0, 0, 14]	468940800
[0, 0, 15]	1525495936
[0, 0, 16]	-218330816
[0, 0, 17]	-1605551344
[0, 0, 18]	-385718720
[0, 0, 19]	-904442752

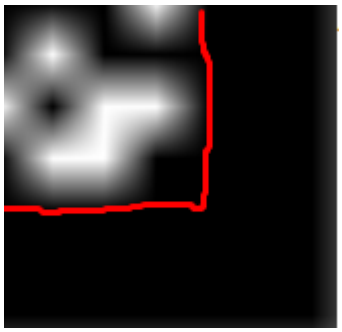
[0, 0, 0]	-2019086336
[0, 0, 1]	1642520192
[0, 0, 2]	1306041344
[0, 0, 3]	-407396480
[0, 0, 4]	-1049077568
[0, 0, 5]	-166054128
[0, 0, 6]	1251489684
[0, 0, 7]	-1724061872
[0, 0, 8]	2122798464
[0, 0, 9]	-1893925984
[0, 0, 10]	1396498248
[0, 0, 11]	1042751488
[0, 0, 12]	-951365120
[0, 0, 13]	-1393890304
[0, 0, 14]	-779010976
[0, 0, 15]	556444624
[0, 0, 16]	-379569784
[0, 0, 17]	1681833744
[0, 0, 18]	-627536192
[0, 0, 19]	705570688







Convolution mini bug

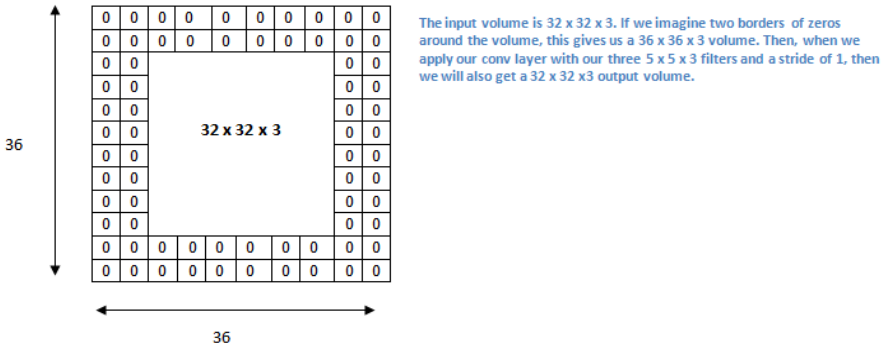


As the convolution filters work, one small bug was discovered.
Instead of the whole 7x7 square being used as output, only a small amount of it is.

The problem should be because of my code and that it slowly, increases the black border on the output image, the further the image goes down the convolution.

So I did some research and discovered this representation on how convolution should work, as you can see, during the convolution stage, the image would be placed in a centre of a larger image, with borders around it. My idea to making this work in my program is to make a very large 2d integer array, and to place the image in the middle of it. Since my algorithm will have a huge image to work with, because of the borders, it won't matter if we lose some of that border because at the end of the convolution, we would take that image out of the borders and use that.

So I simply made the array really large and used pointers to set the image around the middle of the array.



Convolution mini bug fix

So as a way of fixing this, I created the array big so it would not lose any detail of the image.

Since the neural network will require a 1d array, I also created a converter to convert the 3d integer array into a 1d integer array.

```
private static int[,,] convolution64_7x7(Bitmap bp)
{
    int[,,] img = new int[448+500, 448+500, 64];//448x,448y,rgb
    int[,,] img2 = new int[224+500, 224+500, 64];//224
    int ttmp = 0;

    int pointerx = 0;
    int pointery = 0;

    for(int f = 0; f<64; f++)
    {
        for(int y = 0; y<448; y++)
        {
            for(int x = 0; x<448; x++)
            {
                Color cl = bp.GetPixel(x, y);
                pointerx++;
                switch (ttmp)
                {
                    case 0:
                        img[pointerx, pointery, f] = cl.R;
                        ttmp++;
                        break;
                    case 1:
                        img[pointerx, pointery, f] = cl.G;
                        ttmp++;
                        break;
                    case 2:
                        img[pointerx, pointery, f] = cl.B;
```

```
for(int i = 0; i < 1024; i++)
{
    for(int y = 0; y < 7; y++)
    {
        for(int x = 0; x<7; x++)
        {
            inp[temp] = Math.Max(tmp[x, y, i],0);//stores a 3d integer array into a 1d integer array
            temp++;//increments the pointer
        }
    }
}
```

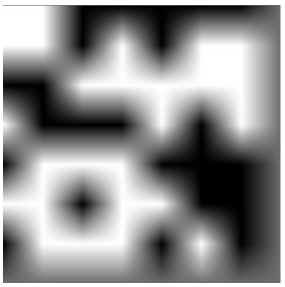
Convolution testing pt2

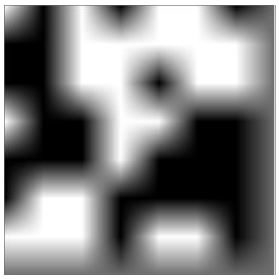
In order to see if there would be anymore other problem with the convolution function I decided to run another test on it and see if everything works just how it's supposed to.

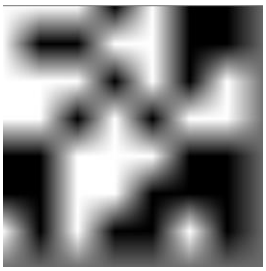
[0, 0, 0]	-1227494976
[0, 0, 1]	-954120832
[0, 0, 2]	922937344
[0, 0, 3]	-174650624
[0, 0, 4]	297308352
[0, 0, 5]	-433679344
[0, 0, 6]	-906650724
[0, 0, 7]	-371492224
[0, 0, 8]	313597312
[0, 0, 9]	-128769328
[0, 0, 10]	-373705320
[0, 0, 11]	-1094115200
[0, 0, 12]	1499815424
[0, 0, 13]	232430080
[0, 0, 14]	1769587456
[0, 0, 15]	-1331847424
[0, 0, 16]	-1062705032
[0, 0, 17]	-1494463380
[0, 0, 18]	-974368896
[0, 0, 19]	-6831232

[0, 0, 0]	242242944
[0, 0, 1]	1546545536
[0, 0, 2]	-1097383936
[0, 0, 3]	109435392
[0, 0, 4]	-1307205312
[0, 0, 5]	63584832
[0, 0, 6]	1042303562
[0, 0, 7]	-353023696
[0, 0, 8]	-948032640
[0, 0, 9]	-1710934896
[0, 0, 10]	281504872
[0, 0, 11]	737412288
[0, 0, 12]	-2134955072
[0, 0, 13]	1053894144
[0, 0, 14]	1754420672
[0, 0, 15]	-1796834528
[0, 0, 16]	-1472654384
[0, 0, 17]	1499855668
[0, 0, 18]	-735009152
[0, 0, 19]	-1213390592

[0, 0, 0]	-888355328
[0, 0, 1]	948014976
[0, 0, 2]	-1355997184
[0, 0, 3]	353564480
[0, 0, 4]	-70759872
[0, 0, 5]	1558150512
[0, 0, 6]	-442580600
[0, 0, 7]	-1740654176
[0, 0, 8]	209338752
[0, 0, 9]	-659861440
[0, 0, 10]	884245392
[0, 0, 11]	391267648
[0, 0, 12]	-1467068672
[0, 0, 13]	-1210950144
[0, 0, 14]	-472904192
[0, 0, 15]	1164189984
[0, 0, 16]	-281454176
[0, 0, 17]	1071453704
[0, 0, 18]	-1373179392
[0, 0, 19]	1313773952







0.2	**Code[**create a basic image abstractor, create a function to import pictures, program the image boxes to output abstracted images.**]**	Import a random picture and make sure that the basic image abstractor function works.

Since all of the outputs are different and similar to one another, I am certain that this was a successful test and no problems were found with the algorithm.

As this version can do all of that I can say that I successfully fully finished version 0.2

Version 0.3

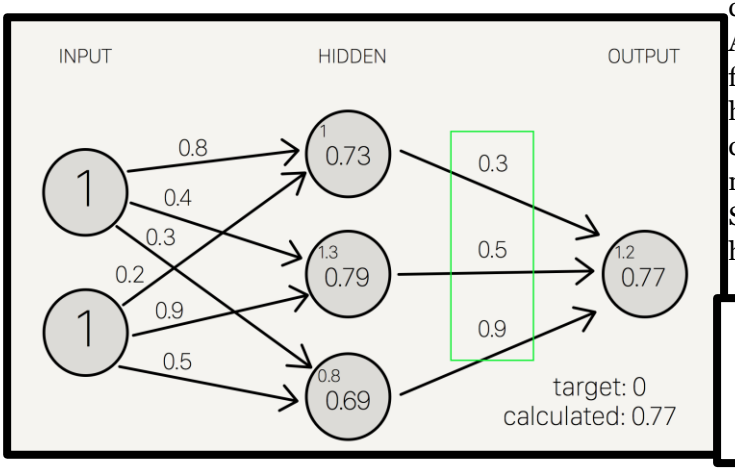
0.3	**Code[**create neural network runner and link the inputs to the neural network**]**	Check if running the neural network displays the correct output	For this version I have to simply create a neural network, just to run in what it's learnt.

https://towardsdatascience.com/deep-learning-feedforward-neural-network-26a6705dbdc7

So after a lot of research into neural networks I discovered this website and video which are great at explaining how neural networks work.

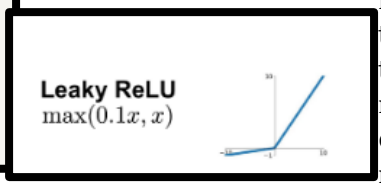
https://www.youtube.com/watch?v=GvQwE2OhL8I

For example if we had 2 inputs of 1 and 0 and 1 hidden layer and 1 output, the hidden layer = input1 * weight 1 + input2* weight 2 and the output would be calculated as the hidden layer * weight connected to the output and so on.



As I was doing even more research I found this flowchart showing that they had to use leaky relu after every convolution layer and even in the neural network section.

So I decided to implement leaky rely and how it works is quite simple, if a number is below 0.1 then set the number to 0.1 else do nothing to the number.



Since c# has math.max function in which it selects the highest number out of 2 numbers.

```
img2[pointerx, pointery, f] = Math.Max(tempor,0.1);
```

One problem with this, was that all of the convolution functions used integers rather than double which is needed since doubles can store numbers with decimal places whilst integers can only store full positive or negative numbers.

```
private static double[,,] convolution128_1x1(double[,,] img)
{
    double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 128];//128
    int pointerx = 0;
    int pointery = 0;
    int tmp = 0;
    for (int f = 0; f < 128; f++)
    {
        for (int y = 0; y < img.GetLength(1); y++)
        {
            for (int x = 0; x < img.GetLength(0); x++)
            {
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                tempor += filter_128_1x1[f] * img[x, y, tmp];

                img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
            }
            pointerx = 0; //sets pointer x back to 0
            pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

        }
        pointery = 0;
        pointerx = 0;
        tmp++;
        if (tmp >= img.GetLength(2))
        {
            tmp = 0;
        }
```

For every other convolution function I had to replace the integers with doubles and that solved the relu problem.

Neural network feedforward code

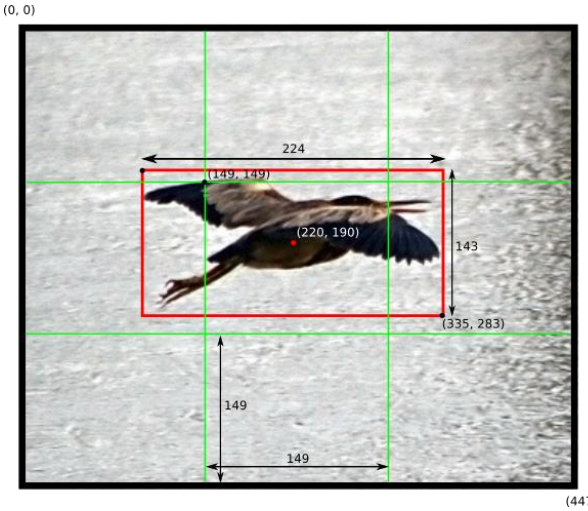
So after all of that research I finally made my neural network.
The 1st loops calculates each hidden layers of the neural network whilst the 2nd calculates the output using the calculated hidden layers and lastly the final loops stores the output into a 3d array ready to be displayed.

```
private static double[,,] neuralnetwork(double[] inp)
{


        double[] hiddenlayer = new double[4096];//size ofx the hidden layer
        double[] outp = new double[1470];//output size(7x7x30)
        double[,,] outp_3d = new double[7, 7, 30];//create 3d array
        int tmp = 0;
        for (int i = 0; i < 4096; i++)//hidden layer size
        {
            for (int x = 0; x < inp.Count(); x++)//7x7x1024 = 50176 inputs
            {
                hiddenlayer[i] += inp[x] * w1[tmp]; //works out the hidden layer's value
                tmp++;
            }
        }
        tmp = 0;
        for (int i = 0; i < 1470; i++)
        {
            for (int x = 0; x < 4096; x++)
            {
                outp[i] += hiddenlayer[x] * w2[tmp];
                tmp++;
            }
        }
        tmp = 0;
        for (int i = 0; i < 30; i++)
        {
            for (int y = 0; y < 7; y++)
            {
                for (int x = 0; x < 7; x++)
                {
                    outp_3d[x, y, i] = outp[tmp];//simply stores the 1d double array into a 3d double array
                    tmp++;
                }
            }
        }


        return outp_3d;
}
```

As I tested the neural network the output seemed a bit too large since the output I am looking for is really small see example below.

So now in order to see the problem with this I need to check the values of w1, w2, and the input, to see what might be causing this problem.

[0, 0, 0]	1300591150460848.5
[0, 0, 1]	1306065112038058.7
[0, 0, 2]	1289930383297361.3
[0, 0, 3]	1279658999532175.5
[0, 0, 4]	1271837129170189.5
[0, 0, 5]	1277287938239354
[0, 0, 6]	1302626750122364.7
[0, 0, 7]	1291988780156473.7
[0, 0, 8]	1288072118311908.5
[0, 0, 9]	1278332961057569.5
[0, 0, 10]	1288613989685081.2
[0, 0, 11]	1285459662738325.7



x = (220-149) / 149 = 0.48
y = (190-149) / 149 = 0.28
w = 224 / 448 = 0.50
h = 143 / 448 = 0.32

Neural network feedforward problem

Input	Weights1	Weights2
[1463] 0.1	[0] 0.865585901246213	[0] 0.431238025627675
[1464] 0.1	[1] 0.419450747044501	[1] 0.320723657179961
[1465] 0.1	[2] 0.0645071063491083	[2] 0.809186515309469
[1466] 0.1	[3] 0.583145906023283	[3] 0.907234704544411
[1467] 0.1	[4] 0.256724056907335	[4] 0.705268178929234
[1468] 0.1	[5] 0.321495310087453	[5] 0.243960940858331
[1469] 0.1	[6] 0.544590765863932	[6] 0.659031334174346
[1470] 0.5	[7] 0.333734096183318	[7] 0.671084465305826
[1471] 0.5	[8] 0.668330442005922	[8] 0.460226691542299
[1472] 0.5	[9] 0.389169093868308	[9] 0.853223776376445
[1473] 0.5	[10] 0.204424258882377	[10] 0.736121822957006
[1474] 0.5	[11] 0.448116755787338	[11] 0.696467979204128
[1475] 0.5	[12] 0.932438057350199	[12] 0.713424168393679

So possibly this can not be the problem, although I possibly found a problem with the input as it is repeating the numbers exactly 49 times which is 7x7 but this is not the cause for this problem.

[0]	701686115075.13879
[1]	588536253896.66052
[2]	696288960520.61475
[3]	529604914536.58649
[4]	788307887427.4519
[5]	651725669231.02
[6]	693180849356.61023
[7]	580002765477.74316
[8]	637369344039.591
[9]	626057263034.82556
[10]	674665963441.6521
[11]	655410675643.234
[12]	674278207987.22681

Since the hidden layer is calculated by hidden += inp * w1 in small amounts it would be fine, but since there are so many inputs and weights, the value of the hidden layer keeps rising and rising.
So this could mean that the neural network is working how it should be it just needs training so the output would be reduced.

```
for (int i = 0; i < 4096; i++)//hidden layer size
{
    for (int x = 0; x < inp.Count(); x++)//7x7x1024 = 50176 inputs
    {
        hiddenlayer[i] += inp[x] * w1[tmp]; //works out the hidden layer's value
        tmp++;
    }
}
```

Since the neural network needs to be trained in order to make the output correct, I decided to proceed to the next version.

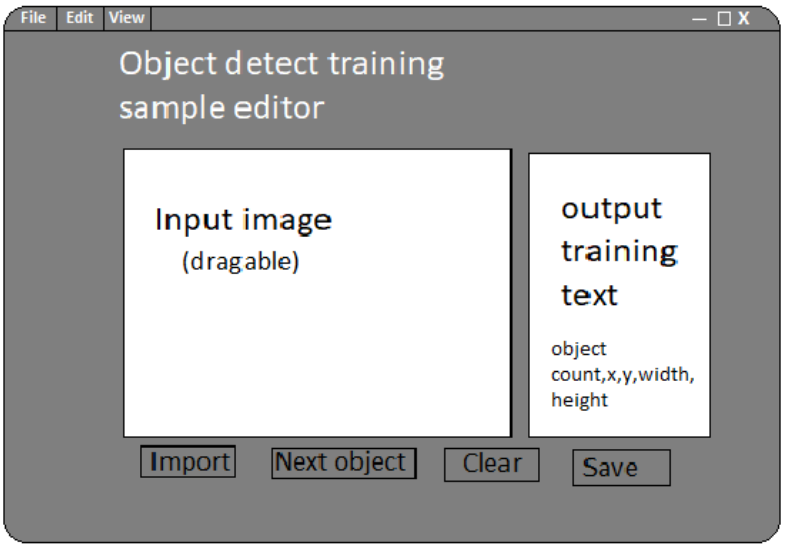
0.3	**Code[**create neural network runner and link the inputs to the neural network**]**	Check if running the neural network displays the correct output
0.4	**Code[**make neural network trainer and check if it works and reduces the cost**]**	See if training the neural network makes it smarter

Since version 0.3 can't be completed till version 0.4 doing version 0.4 and then going back to 0.3 to test would be the best approach.

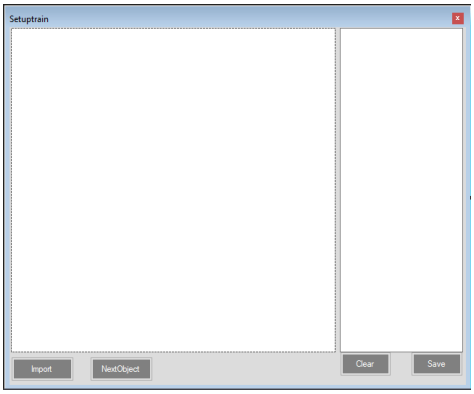
Version 0.4

In this version I have to create a neural network trainer and I will also make a training dataset maker as well so the user could create a training dataset of their own.

So for the dataset creator, this was the original design but I decided to make it different since the planned design did not look as good and wasted a lot of space.



This new design however looks much better and simpler than the last and therefore I will use the newer one.

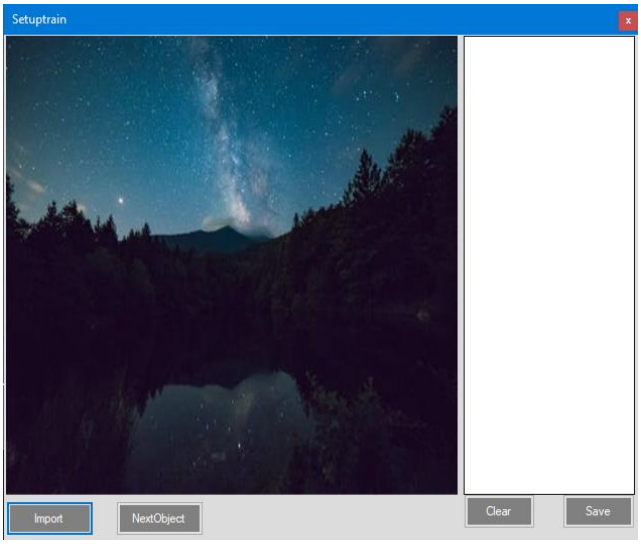


So now in order for the user to create a training dataset, I need to make the import button work and since it would work very similarly to the main program image importer then I can simply re use that code for this particular one.
So I slightly change the code to suit this form as it's using different object names.

I quickly tested to see if it all works perfectly and it does so we can say that the import button is working just

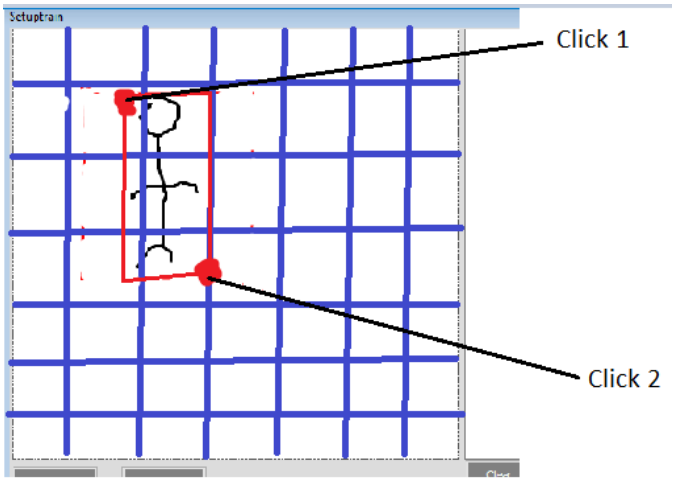
how it should be.

```
private void buttonImport_detect_Click(object sender, EventArgs e)
{
    Bitmap bp;
    OpenFileDialog fl = new OpenFileDialog();
    DialogResult rl = fl.ShowDialog();
    if (rl == DialogResult.OK)
    {
        try
        {
            bp = new Bitmap(fl.FileName);
            Bitmap resized = new Bitmap(bp, new Size(448, 448));

            pictureBoxImg.Image = resized;
        }
        catch (Exception)
        {
            MessageBox.Show("File cannot be loaded!");
        }
    }
    else
    {
        MessageBox.Show("Could not open the image!");
    }
}
```



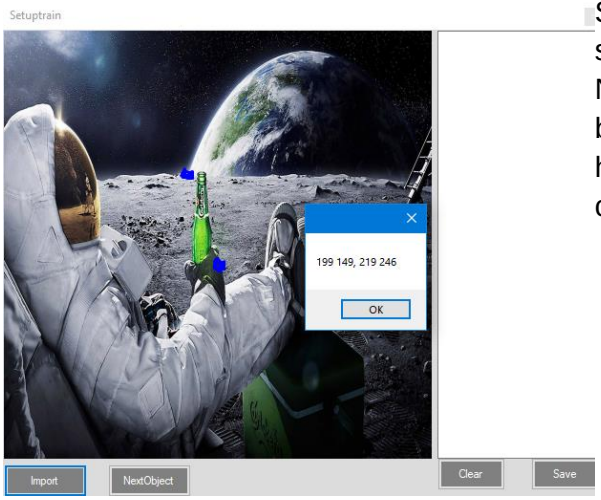
Data set maker planning and coding

Since I want the user to select an object within the image box, I decided to record the location of where the user clicks which would generate a pointer and the second click would also generate a pointer and those two pointers would create a virtual box depending where the user clicks.



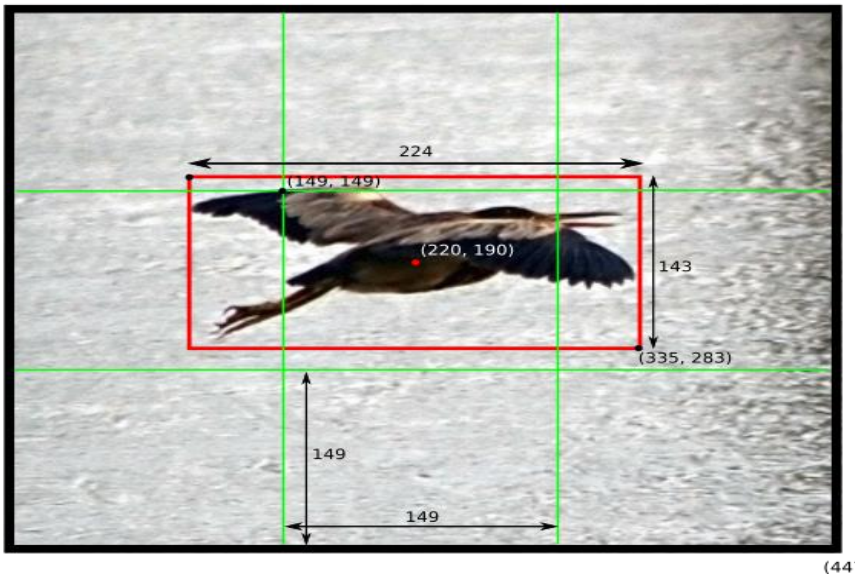
The code for this is pretty simple. When the user first clicks on the image, it would record the x and y of where the user clicked and it would store it into integer variables x and y but the second time the user clicks on the image, the location would be recorded as well but would be stored in integer xx and yy.

```
private void pictureBoxImg_MouseClick(object sender, MouseEventArgs e)
{
    if (selected == false)
    {
        selected = true;
        x = e.X;
        y = e.Y;

    }
    else
    {
        xx = e.X;
        yy = e.Y;
        selected = false;
        MessageBox.Show(x.ToString() + " " + y.ToString() + ", " + xx.ToString() + " " + yy.ToString());

    }
}
```



So as you can see it works perfectly as I selected the beer bottle within the image. Now all I need to do is to calculate the bx,by,bw,bh and luckily I found an extremely helpful example explaining how it needs to be calculated.

Data set maker planning pt2

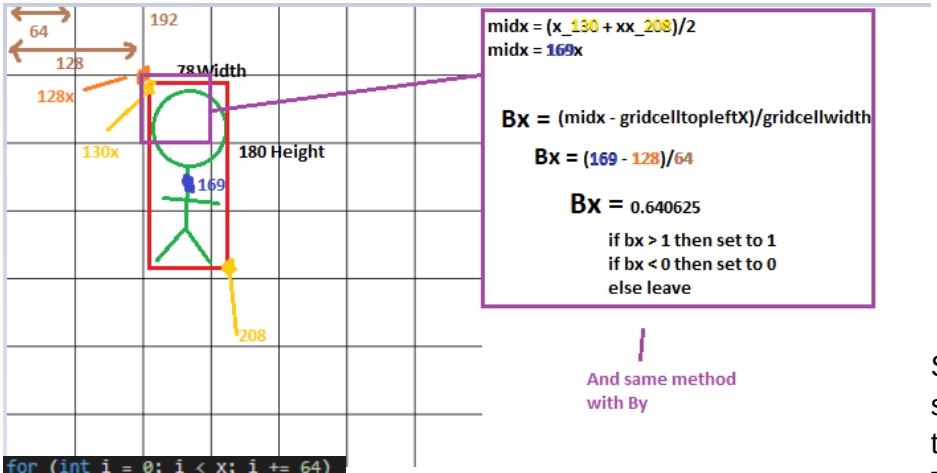


x = (220-149) / 149 = 0.48
y = (190-149) / 149 = 0.28
w = 224 / 448 = 0.50
h = 143 / 448 = 0.32

The reason for why this example is very helpful is because it tells you everything you need to know about calculating those 4 attributes.
I condensed it even more to make it easier to understand.
So the 1st thing needed to do is to calculate the midpoint of the object and to do that we just have to add up both of the x pointers that we stored and divide them by 2, giving us the midpoint(x) of the object.

```
(x+xx)/2 = midx
(y+yy)/2 = midy

bx = (midx - gridcelltopleftX)/gridcellwidth
by = (midy - gridcelltopleftY)/gridcellheight
bw = widthofobject / 448
bh = heightofobject / 448
ob1 = how confident is it in being object 1
```

Then to calculate bx and by I made a diagram as it's harder to explain




```
for (int i = 0; i < x; i += 64)
{
    pointer_x = i;
}
for (int i = 0; i < y; i += 64)
{
    pointer_y = i;
}
for (int i = 0; i< xx; i += 64)
{
    pointer_xx = i;
}
for(int i = 0; i< yy; i+= 64)
{
    pointer_yy = i;
}
```

So now the next step is to apply that to my code.
The first step was to locate every grid cell that the object is inside of. So how I did it was simply by retrieving the x location of where the player clicked and make a for loop that would check every x cell until the cells right side x is bigger than the location of selected x, this would tell us at which cell x is located in and we do the same thing for y, xx and yy

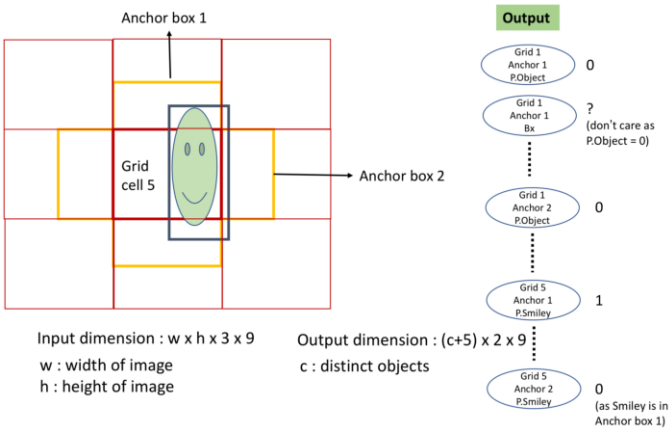
Data set maker code and research

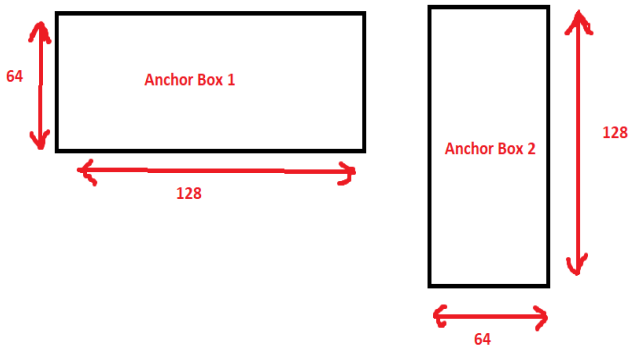
After working out the pointers, which are simply the top left corner x and y of cells that x,y and xx,yy were selected in. This would allow me to workout the location of every cell within the selected area.

```
double num1 = 64;//I did this to fix a simple bug since c# thought that 64 is an integer it automatically rounded up the number
for(int y = pointer_y; y<pointer_yy + 64; y+= 64)
{
    for (int x = pointer_x; x < pointer_xx + 64; x+= 64)
    {
        MessageBox.Show(x.ToString() + "      " + y.ToString());//for debuging
        expected[x/64, y/64, 0] =  (midx - x) / num1; //works out the bx
        expected[x / 64, y / 64, 1] = (midy - y) / num1;//works out the by
        MessageBox.Show(expected[x/64, y/64, 0].ToString() + "x " + expected[x/64,y/64,1].ToString() + "y ");//for debuging



    }
}
```

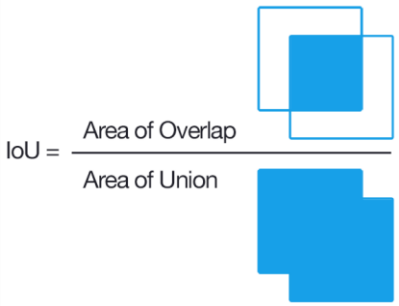
The next step into making my data set would be work out which anchor box the object mostly suits to.

To work out the sizes of my anchor boxes, I decided to use the similar size as in the example, since my 1 grid cell is 64 width and 64 height and in the example anchor box 1 and 2 take up to two grid cells. So to make this work,my anchor box 1 will be 128x64 whilst anchor box 2 will be 64x128 the reason why they are like that is because anchor box 1 might be more suitable for cars and any other wide object, whilst anchor box 2 would be more suitable for tall objects, in a nutshell anchor boxes are used to display more than one

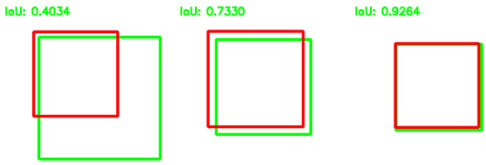




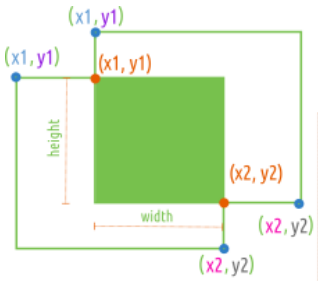
objects in a grid cell, in yolo v1 we will only have 2 anchor boxes therefore we will have only up to two objects max per box cell.



In order to see which object belongs to which anchor box, I needed to do some research and I've discovered IoU and the reason why IoU is used is because if we apply the IoU to the object and the 2 anchor boxes, the anchor box with the highest IoU will be chosen for that object.



Data set maker research pt2 code and



$(x1, y1) = (\max(x1), \max(y1))$
$(x2, y2) = (\min(x2), \min(y2))$

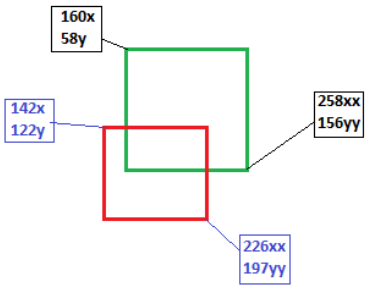
If width and height are both positive:
Overlap = width * height
= (x2-x1) * (y2-y1)
Else:
Overlapping Region = 0

After some many research of how to workout the IoU, I came across two very helpful websites in explaining how to work it all out.
So I decided to condense their explanation into an example explaining everything in how to work it out.
https://www.dlology.com/blog/gentle-guide-on-how-yolo-object-localization-works-with-keras-part-2/

https://www.geeksforgeeks.org/total-area-two-overlapping-rectangles/

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

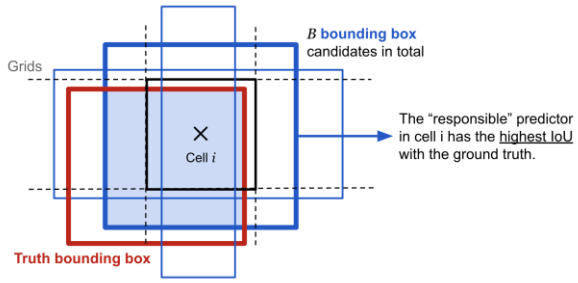
Area1 = abs(x - xx) * abs(y - yy)
Area2 = abs(2x - 2xx) * abs(2y - 2yy)
Area of Overlap = (min(xx, 2xx) - max(x, 2x)) * (min(yy, 2yy) - max(y,2y))
Area of Union = Area1 + Area2 - Area of Overlap
Iou = Area of Overlap/ Area of Union



Area1 = abs(160 - 258) * abs(58-156) // 98 * 98 = 9604
Area2 = abs(142-226) * abs(122-197) //84 * 75 = 6300

Area of Overlap = (min(258,226) - max(160,142)) * (min(156,197) - max(58,122)) //(226 - 160) * (156 - 122) = 66 * 34 = 2244
Area of Union = Area1 + Area 2 - Area of Overlap 9604 + 6300 - 2244 = 13660
IoU = 2244 / 13660
IoU = 0.16427525622254758418740849194729

Note: Abs converts negative numbers to positive, max picks the highest number out of two, min picks the smallest number out of two.

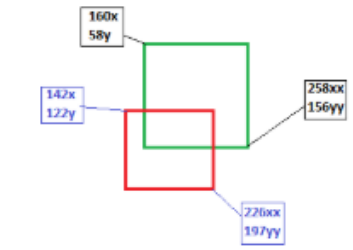


So now all I need to do is implement it to my code and test if it works.

Data set maker code and research pt3

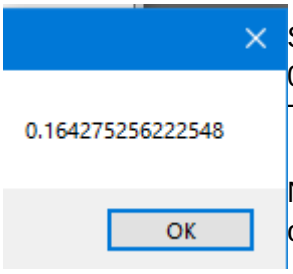
```
private static double IoUgenerator(double x, double y, double xx, double yy, double x_2, double y_2, double xx_2, double yy_2)
{
    double area1 = Math.Abs(x - xx) * Math.Abs(y - yy);//all explained in the documentation
    double area2 = Math.Abs(x_2 - xx_2) * Math.Abs(y_2 - yy_2);

    double areaofoverlap = (Math.Min(xx, xx_2) - Math.Max(x, x_2)) * (Math.Min(yy, yy_2) - Math.Max(y, y_2));
    double areaofunion = area1 + area2 - areaofoverlap;

    return areaofoverlap/areaofunion;
}
```

So I simply made a function that takes in the location of the 2 boxes/anchors and returns the calculated IoU.
In order to see if this function is fully working I calculated the IoU in the example above and we will use the same input data to see if it returns the same result.



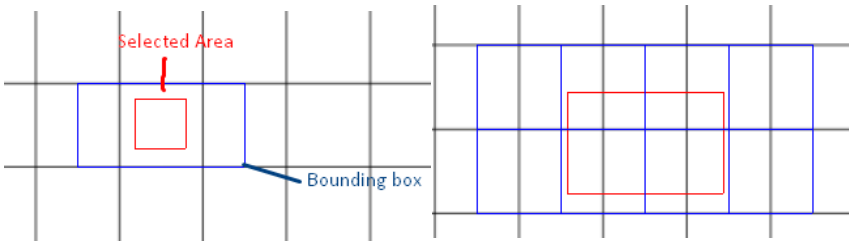
Area1 = abs(160 - 258) * abs(58-156) // 98 * 98 = 9604
Area2 = abs(142-226) * abs(122-197) //84 * 75 = 6300
Area of Overlap = (min(258,226) - max(160,142)) * (min(156,197) - max(58,122))
Area of Union = Area1 + Area 2 - Area of Overlap 9604 + 6300 - 2244 = 13660
IoU = 2244 / 13660

```
MessageBox.Show(IoUgenerator(160, 58, 258, 156, 142, 122, 226, 197).ToString());
```

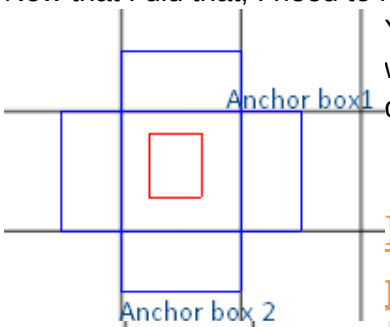


So as we can see, the IoU function is fully working as it outputted 0.1642etc
This means that the IoU calculating functions works just perfectly.

Next stage is to workout which bounding box is most suitable for the object, in each grid.

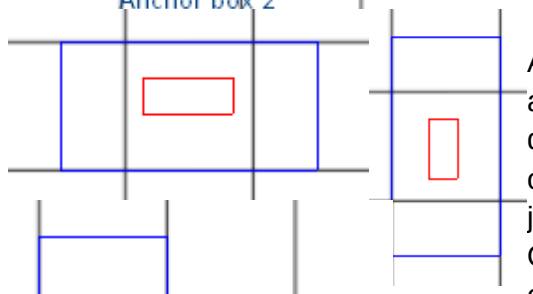


In order for me to know that the anchor boxes are working how they should be, I decided to program my data set to draw out the anchor boxes for the cells selected.
Now that I did that, I need to make one more anchor box (anchor box 2)

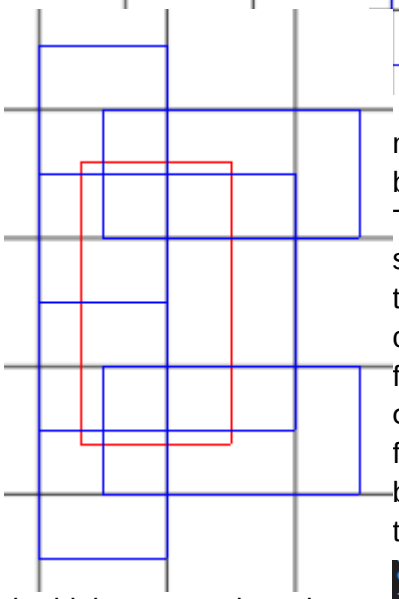


Yet again, it worked successfully as it displayed both of the wanchor boxes and they work for each grid cell except the ones on the edges.

Data set maker code and research pt4



As the two anchor boxes work, I needed display the anchor box chosen, for debugging and documentation purposes. So as you can see, the object selected (red box) has an anchor box chosen just for the shape of the object.
One bug I found in the anchor box function is that on big objects that go through some grids, some non correct anchor boxes are chosen, as all of the anchor boxes for object should be the same one.

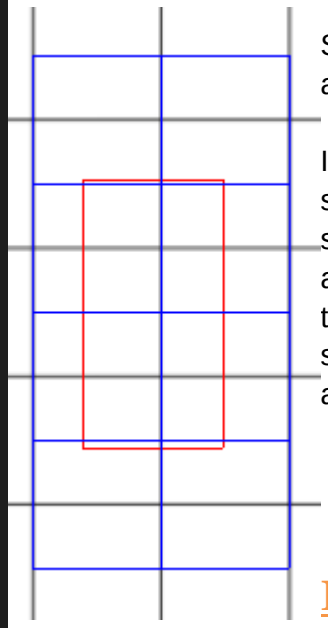


The code simply, uses the iou checker function to check 2 iou's for 2 anchor boxes and then selects the highest one, since there was a minor bug of both of the iou is the same but different shape, I decided to make another else if statement to check which shape matches the most with the bounding box.

```
if(anchorbox1iou > anchorbox2iou)// if anchorbox1 iou is bigger than anchorbox2 iou
{
    for (int i = ex; i < exx; i++)//draws the box
    {
        bmp.SetPixel(i, ey, Color.Blue);
    }
    for (int i = ey; i < eyy; i++)
    {
        bmp.SetPixel(ex, i, Color.Blue);
    }
    for (int i = ex; i < exx; i++)
    {
        bmp.SetPixel(i, eyy, Color.Blue);
    }
    for (int i = ey; i < eyy; i++)
    {

        bmp.SetPixel(exx, i, Color.Blue);
    }
```

```
else if (anchorbox2iou > anchorbox1iou)//if anchorbox2iou is greater than anchorbox1iou
{
    for (int i = rx; i < rxx; i++)
    {
        bmp.SetPixel(i, ry, Color.Blue);
    }
    for (int i = ry; i < ryy; i++)
    {
        bmp.SetPixel(rx, i, Color.Blue);
    }
    for (int i = rx; i < rxx; i++)
    {
        bmp.SetPixel(i, ryy, Color.Blue);
    }//DELETE DIS
    for (int i = ry; i < ryy; i++)
    {
        bmp.SetPixel(rxx, i, Color.Blue);
    }
}
```

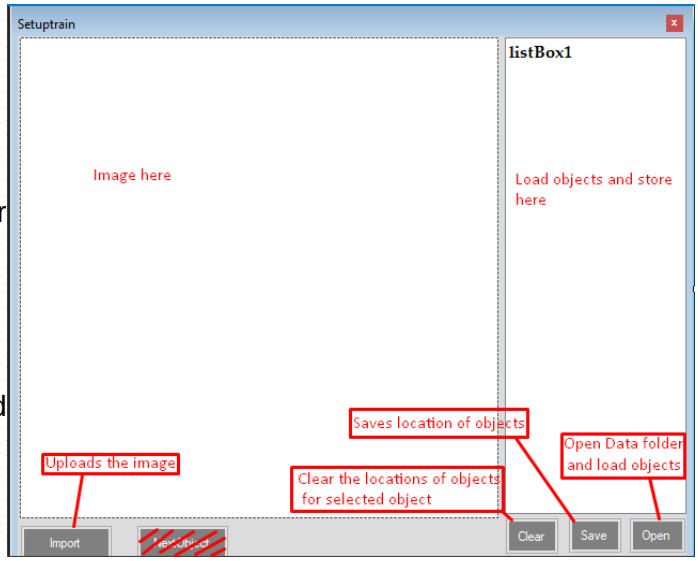
```
else if(Math.Abs(x-xx) > Math.Abs(y-yy))
{
    for (int i = ex; i < exx; i++)
    {
        bmp.SetPixel(i, ey, Color.Blue);
    }
    for (int i = ey; i < eyy; i++)
    {
        bmp.SetPixel(ex, i, Color.Blue);
    }
    for (int i = ex; i < exx; i++)
    {
        bmp.SetPixel(i, eyy, Color.Blue);
    }//DELETE DIS
    for (int i = ey; i < eyy; i++)
    {
        bmp.SetPixel(exx, i, Color.Blue);
    }
}
else if(Math.Abs(x - xx) < Math.Abs(y - yy))
{
    for (int i = rx; i < rxx; i++)
    {
        bmp.SetPixel(i, ry, Color.Blue);
    }
    for (int i = ry; i < ryy; i++)
    {
        bmp.SetPixel(rx, i, Color.Blue);
    }
    for (int i = rx; i < rxx; i++)
    {
        bmp.SetPixel(i, ryy, Color.Blue);
    }//DELETE DIS
    for (int i = ry; i < ryy; i++)
    {
        bmp.SetPixel(rxx, i, Color.Blue);
    }
}
```



So since this function chooses which anchor box is more similar to the object, I decided to disable the IoU and keep just the last two else if statements and by doing that, it successfully, worked as all of the anchor boxes are the same and therefore the whole object should be stored in the similar section of the array.

Data set maker code and research pt5

So before I go any further, I decided to make some changes to my programs layout, the reason for this is so I could save the objects and load them up and in case if the user makes a mistake, they can clear the selected objects.



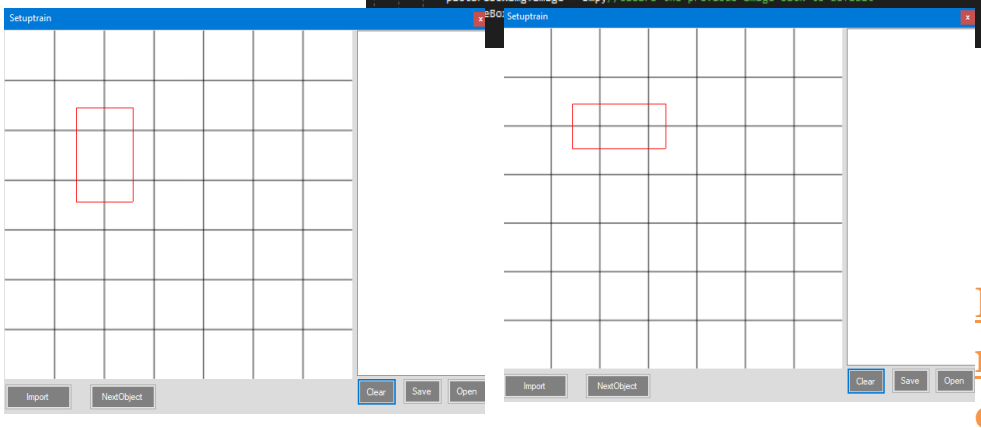
I decided to reuse the same code used for the main form, to load up the objects but instead of loading up the weights and filters I loaded up just the object list.

In order to make this program easier to use, I decided to implement this feature to draw the box as the cursor moves, in which it also allows the user to see where the box would be drawn once they click with their cursor.

```
private void buttonImport_detect_Click(object sender, EventArgs e)
{
    Bitmap bp;//creates bitmap variable
    OpenFileDialog fl = new OpenFileDialog();
    DialogResult rl = fl.ShowDialog();//opens up a dialog that allows the user to choose the file path
    if (rl == DialogResult.OK)//checks if file path is valid
    {
        try//in case if something goes wrong, the program won't crash but display that something is wrong with filepath
        {
            bp = new Bitmap(fl.FileName);//opens the image and stores it into bitmap variable
            default_bmp = new Bitmap(bp, new Size(448, 448));//resizes the image to 448x448 size
            bmp = default_bmp;
            pictureBoxImg.Image = default_bmp;//displays the image
        }
        catch (Exception)
        {
            MessageBox.Show("File cannot be loaded!");
        }
    }
    else
    {
        MessageBox.Show("Could not open the image!");
    }
}
```

This feature is not necessary, but I decided to use it anyway, as it is very helpful when trying to select an object.

```
private void pictureBoxImg_MouseMove(object sender, MouseEventArgs e)
{
    if(selected == true)//this is just to make it look nicer.
    {
        int xxx = e.X;
        int yyy = e.Y;
        if (xxx != previousx && yyy != previousy)
        {
            bitm = new Bitmap(bmp);//creates a copy of the image
            previousx = x;//to reduce lag, I decided to implement this so it would check if the location of the cursor has changed or not
            previousy = y;
            for (int i = x; i < xxx; i++)//draws box
            {
                bitm.SetPixel(i, y, Color.Red);
            }
            for (int i = y; i < yyy; i++)
            {
                bitm.SetPixel(x, i, Color.Red);
            }
            for (int i = x; i < xxx; i++)
            {
                bitm.SetPixel(i, yyy, Color.Red);
            }
            for (int i = y; i < yyy; i++)
            {
                bitm.SetPixel(xxx, i, Color.Red);
            }
            pictureBoxImg.Image = bmp;//clears the previous image back to default
```



Data set maker code and research pt6

As I needed to make the actual selection of the object and to record the location and work out all the other variable like bx,by etc for bounding boxes.
The first part was relatively simple, as the program records the x and y of the click and marks it as the 1st click was used.

```
private void pictureBoxImg_MouseClick(object sender, MouseEventArgs e)
{
    if (selected == false)//when the user clicks first time, the x and y of the location of the first selection click will be recorded
    {
        bmp = new Bitmap(pictureBoxImg.Image);//sets bmp to the image
        selected = true;//sets that the 1st click was used
        x = e.X;//obtains the x co-ordinates of the click
        y = e.Y;//obtains the y co-ordinates of the click
        pictureBoxImg.Image = bmp;

    }
```

```
else//2nd x and y click recorded in which will later on draw the selection and record the location of the object.
{
    pictureBoxImg.Image = bmp;
    xx = e.X;//get the co-ordinates of the 2nd click
    yy = e.Y;

    int pointer_xx = 0;
    int pointer_yy = 0;
    int pointer_x = 0;
    int pointer_y = 0;
    midx = (xx + x) / 2;//finds midpoint of x
    midy = (yy + y) / 2;//finds midpoint of y
    for(int i = x; i< xx; i++)//draws the selected box
    {
        bmp.SetPixel(i, y, Color.Red);//draws the top
    }
    for (int i = y; i < yy; i++)
    {
        bmp.SetPixel(x, i, Color.Red);//draws the left side
    }
    for (int i = x; i < xx; i++)
    {
        bmp.SetPixel(i, yy, Color.Red);//draws the bottom
    }
    for (int i = y; i < yy; i++)
    {
        bmp.SetPixel(xx, i, Color.Red);//draws the right side
    }
```

On the 2nd click, the 2nd x and y of the click is recorded, giving us the location of the top left corner of the object and the bottom right corner, in which we can simply draw the box once the player does a 2nd click.

I then needed to workout the cells location of the x,y and xx,yy in order to get the location of the anchor boxes that I could draw.

```
for (int i = 0; i < x; i += 64)
{
    pointer_x = i;
}
for (int i = 0; i < y; i += 64)
{
    pointer_y = i;
}
for (int i = 0; i< xx; i += 64)
{
    pointer_xx = i;
}
for(int i = 0; i< yy; i+= 64)
{
    pointer_yy = i;
}
```

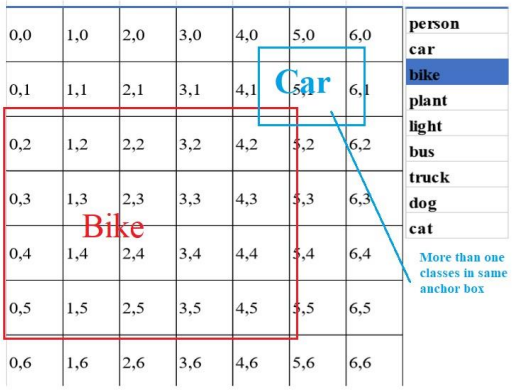
This part of the code is the most complex, as it makes sure that the bounding boxes don't go outside of the grid and causing an out of bounds error. This function also calculates other things like the bx,by,bw,bh and which bounding box is the most appropriate for the object.

```
double num1 = 64;//I did this to fix a simple bug since c# thought that 64 is an integer it automatically rounded up the number
for(int yd = pointer_y; yd<pointer_yy + 64; yd+= 64)
{
    for (int xd = pointer_x; xd < pointer_xx + 64; xd+= 64)
    {
        //MessageBox.Show((xd/64).ToString());
        int ax = Math.Max(xd- 32,0);
        int ay = yd;
        int axx = Math.Min(xd + 96,447);//anchor box 1
        int ayy = Math.Min(yd + 64,447);

        int rx = xd;//anchor box 2
        int ry = Math.Max(yd - 32,0);
        int rxx = Math.Min(xd + 64,447);
        int ryy = Math.Min(yd + 96,447);

        double anchorboxiou = IoUgenerator(ax, ay, axx, ayy, x, y, xx, yy);
        double anchorboxiou = IoUgenerator(rx, ry, rxx, ryy, x, y, xx, yy);

        if(Math.Abs(x-xx) > Math.Abs(y-yy))
        {


            expected[xd / 64, yd / 64, listboxindexnum + 5] = 1;
            bool boxerror = false;
            for (int i = 5; i < 14; i++)
            {
                if (expected[xd / 64, yd / 64, i] == 1 && i != listboxindexnum + 5)
                {
                    //MessageBox.Show("Can't fit another object in cell");
                    expected[xd / 64, yd / 64, listboxindexnum + 5] = 0;
                    boxerror = true;
                }
            }
```

Data set maker code and research pt7

```
if(boxerror == false)
{
    expected[xd / 64, yd / 64, 0] = 1;
    expected[xd / 64, yd / 64, 1] = Math.Min((Math.Max(((midx - xd) / num1), 0)), 1); //works out the bx
    expected[xd / 64, yd / 64, 2] = Math.Min((Math.Max(((midy - yd) / num1), 0)), 1);//works out the by
    expected[xd / 64, yd / 64, 3] = Math.Abs(x - xx) / 448.0;//works out the bw
    expected[xd / 64, yd / 64, 4] = Math.Abs(yy - y) / 448.0;//works out the bh


    //MessageBox.Show(expected[xd/64, yd/64, 3].ToString());
    for (int i = xx; i < xxx; i++)
    {
        bmp.SetPixel(i, yy, Color.Blue);
    }
    for (int i = yy; i < yyy; i++)
    {
        bmp.SetPixel(xx, i, Color.Blue);
    }
    for (int i = xx; i < xxx; i++)
    {
        bmp.SetPixel(i, yyy, Color.Blue);
    }//DELETE DIS
    for (int i = yy; i < yyy; i++)
    {
        bmp.SetPixel(xxx, i, Color.Blue);
    }
}
else
{
    MessageBox.Show("Can't fit another object in cell");
}

}
else if(Math.Abs(x - xx) < Math.Abs(y - yy))
{
    expected[xd / 64, yd / 64, listboxindexnum + 20] = 1;
    bool boxerror = false;
    for (int i = 20; i < 29; i++)
    {
        if (expected[xd / 64, yd / 64, i] == 1 && i != listboxindexnum + 20)
        {
            //MessageBox.Show("Can't fit another object in cell");
            expected[xd / 64, yd / 64, listboxindexnum + 20] = 0;
            boxerror = true;
        }
    }
```

I later discovered a small bug in which displayed the wrong object when saved, for example, I saved the location for a bike and the location for the car and for some reason, the car was still being displayed even if the bike was selected



So since the array was not that large but still hard to make sense of where something was stored, so this diagram labelled from 0,0 to 6,6 was pretty helpful as I would know which cells to look out for. So the problem was that 2 different objects using 1 anchor box in the same cell, wanted to be displayed. So I simply added in a little checker(boxerror) to see if there is more than one classes in same cell using same anchor box; If there is then the most recent object will be ignored and not saved. This worked perfectly as it did not allow too many objects in the same cells.

The rest of the code in that function is relatively similar to the rest, as it checks the 2nd anchor box if there is no box error and stores the location of the object and draws it as well.

On the else statement, it picks the 2nd anchor box but the else statement will only be triggered if the selected object is a perfect square, which is highly unlikely, but still possible and since I want the same anchor box for the whole object I decided to randomly pick an anchor box. Eg: if the whole square is perfectually a square with all sides the same size, then every cell inside the object will have the same anchor box(anchorbox 2).

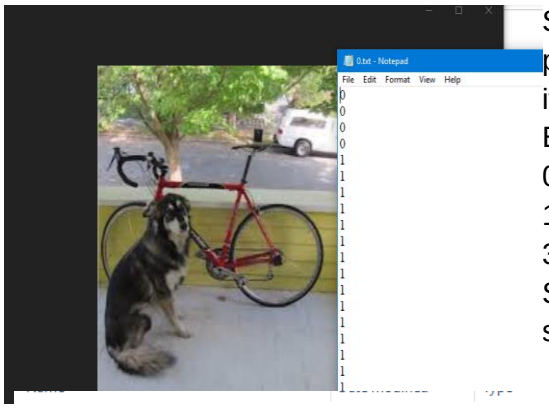
```
{
    expected[xd / 64, yd / 64, listboxindexnum + 20] = 1;
    bool boxerror = false;
    for (int i = 20; i < 29; i++)
    {
        if (expected[xd / 64, yd / 64, i] == 1 && i != listboxindexnum + 20)
        {
            //MessageBox.Show("Can't fit another object in cell");
            expected[xd / 64, yd / 64, listboxindexnum + 20] = 0;
            boxerror = true;
        }
    }

    if(boxerror == false)
    {
        expected[xd / 64, yd / 64, 15] = 1;
        expected[xd / 64, yd / 64, 16] = Math.Min((Math.Max(((midx - xd) / num1), 0)), 1); //works out the bx
        expected[xd / 64, yd / 64, 17] = Math.Min((Math.Max(((midy - yd) / num1), 0)), 1);//works out the by
        expected[xd / 64, yd / 64, 18] = Math.Abs(x - xx) / 448.0;//works out the bw
        expected[xd / 64, yd / 64, 19] = Math.Abs(yy - y) / 448.0;//works out the bh

        for (int i = rx; i < rxx; i++)
        {
            bmp.SetPixel(i, ry, Color.Blue);
        }
        for (int i = ry; i < ryy; i++)
        {
            bmp.SetPixel(rx, i, Color.Blue);
        }
        for (int i = rx; i < rxx; i++)
        {
            bmp.SetPixel(i, ryy, Color.Blue);
        }//DELETE DIS
        for (int i = ry; i < ryy; i++)
        {
            bmp.SetPixel(rxx, i, Color.Blue);
        }
    }
```

Data set maker code Misc changes part 8

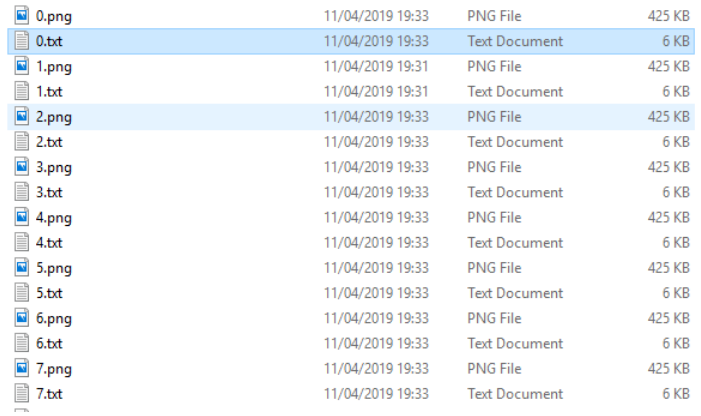
Since, I needed a way to save the files I decided to save the whole expected 3d array into a 1d array inside of a text file that will have the same filename as the image so linking the image file to the text file would be super easy and to also name these files in numbers starting from 0 and on forward.

```
private void buttonSave_Click(object sender, EventArgs e)
{

    int tmp = 0;
    while(File.Exists(location+"\\training\\" + tmp.ToString() + ".txt"))//stores the file once an available number for the file name was found
    {
        tmp++;
    }
    StreamWriter sr = new StreamWriter(location + "\\training\\" + tmp.ToString() + ".txt");//stores it in a text file
    for(int i = 0; i<30; i++)
    {
        for(int y = 0; y< 7; y++)
        {
            for(int x = 0; x< 7; x++)
            {
                sr.WriteLine(expected[x,y,i].ToString());
            }
        }
    }
    default_bmp.Save(location + "\\training\\" + tmp.ToString() + ".png",System.Drawing.Imaging.ImageFormat.Png);
    sr.Close();

}
```

As you can see, there will be a while loop in which would increment tmp(file name) until it finds an available file name in which it would then save the entire expected array.



So as I tested this function, as expected it works perfectly and even works better than I expected as if the a file would be missing eg:
Existing files:
0
1
3
Space 2 would be filled in as the while statement slowly increments until it finds an available space.



0.png	11/04/2019 19:33	PNG File	425 KB
0.txt	11/04/2019 19:33	Text Document	6 KB
1.png	11/04/2019 19:31	PNG File	425 KB
1.txt	11/04/2019 19:31	Text Document	6 KB
2.png	11/04/2019 19:33	PNG File	425 KB
2.txt	11/04/2019 19:33	Text Document	6 KB
3.png	11/04/2019 19:33	PNG File	425 KB
3.txt	11/04/2019 19:33	Text Document	6 KB
4.png	11/04/2019 19:33	PNG File	425 KB
4.txt	11/04/2019 19:33	Text Document	6 KB
5.png	11/04/2019 19:33	PNG File	425 KB
5.txt	11/04/2019 19:33	Text Document	6 KB
6.png	11/04/2019 19:33	PNG File	425 KB
6.txt	11/04/2019 19:33	Text Document	6 KB
7.png	11/04/2019 19:33	PNG File	425 KB
7.txt	11/04/2019 19:33	Text Document	6 KB

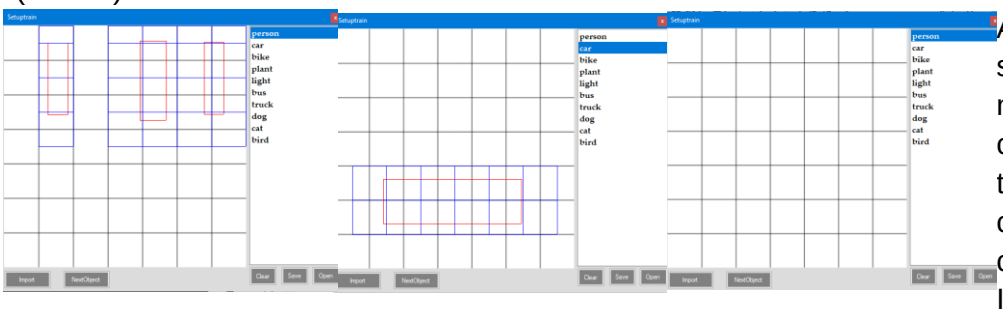
In this example I simply clicked the save button a lot of times to see if the function works well.

Data set maker code Misc changes part 9

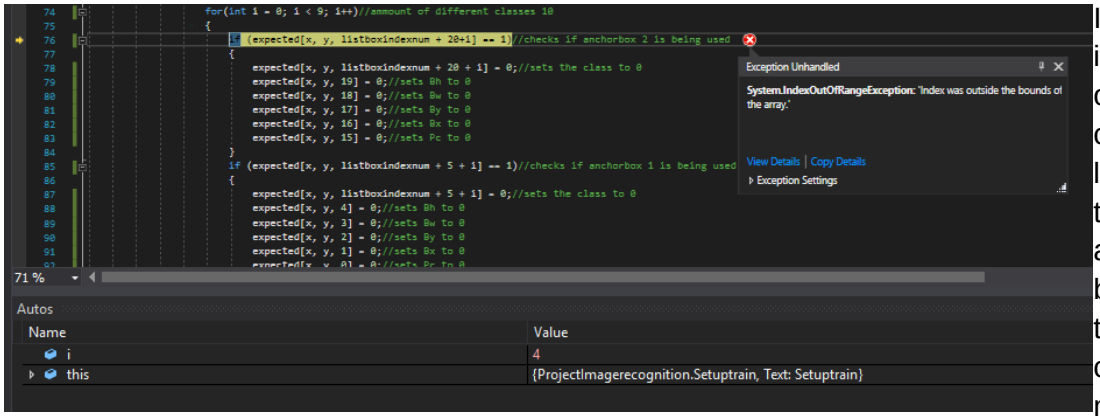
The clear function, in which is essential to clear any changes done in case of an error.

```
private void buttonClear_Click(object sender, EventArgs e)
{
    pictureBoxImg.Image = default_bmp;//removes drawn bounding boxes by setting the image to the default image that was imported
    for(int y = 0; y<7; y++)//since expected is 7x7x30 in output, I need to do a for loop 7x7 times
    {
        for(int x = 0; x<7; x++)
        {

            for(int i = 0; i < 9; i++)//ammount of different classes 10
            {
                if (expected[x, y, listboxindexnum + 20+i] == 1)//checks if anchorbox 2 is being used
                {
                    expected[x, y, listboxindexnum + 20 + i] = 0;//sets the class to 0
                    expected[x, y, 19] = 0;//sets Bh to 0
                    expected[x, y, 18] = 0;//sets Bw to 0
                    expected[x, y, 17] = 0;//sets By to 0
                    expected[x, y, 16] = 0;//sets Bx to 0
                    expected[x, y, 15] = 0;//sets Pc to 0
                }
                if (expected[x, y, listboxindexnum + 5 + i] == 1)//checks if anchorbox 1 is being used
                {
                    expected[x, y, listboxindexnum + 5 + i] = 0;//sets the class to 0
                    expected[x, y, 4] = 0;//sets Bh to 0
                    expected[x, y, 3] = 0;//sets Bw to 0
                    expected[x, y, 2] = 0;//sets By to 0
                    expected[x, y, 1] = 0;//sets Bx to 0
                    expected[x, y, 0] = 0;//sets Pc to 0
                }

            }
```

The code for the clear function is easy to understand as it looks through every cell and checks if that cell is taken and if it is, then the object's location and class gets set to 0(default).



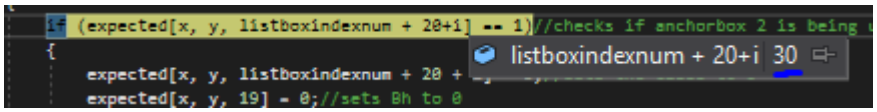
As I tested it, something did not work as It cleared both the person classes and the car classes. I would like to make it so the clear function would only clear out the class that was selected.



I tested it with other classes like truck and dog but in this case, my program threw this error in which is very familiar to me.

Index out of bounds error is cause when the program is trying to edit/view an item in the array that is outside the maximum size of the array. Eg: array size of 5 would not be able to give us the value of item 6

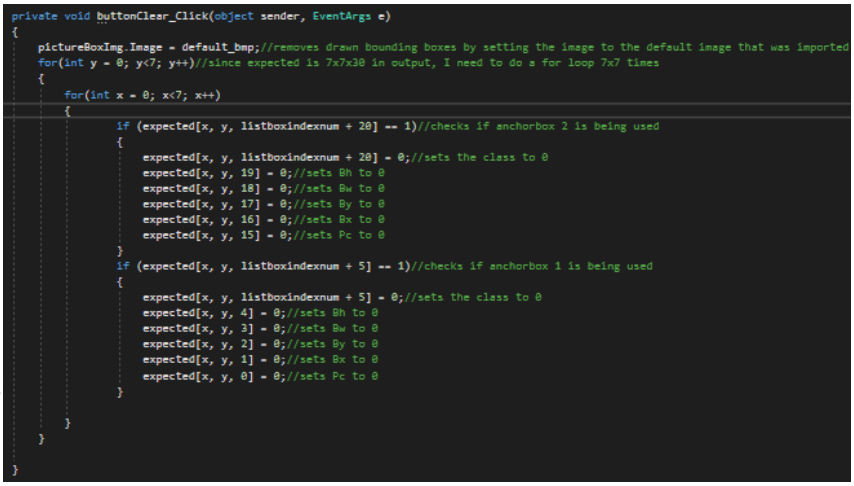
Data set maker code Misc changes part 10



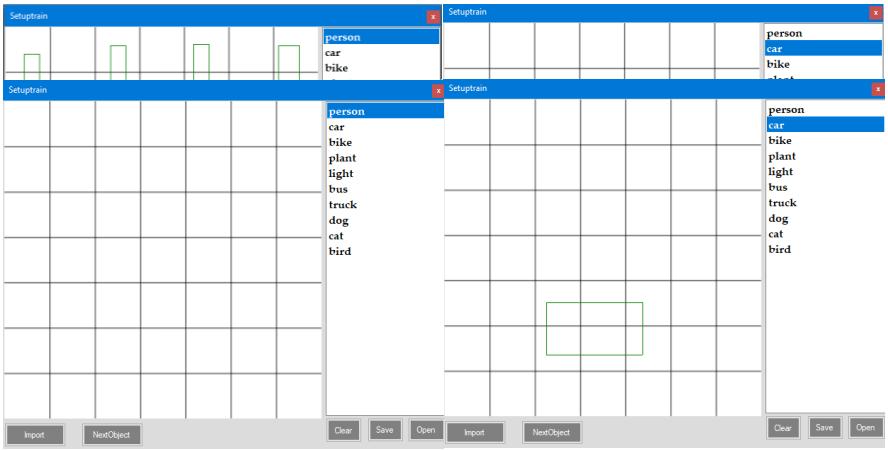
Thanks to visual studio debugging tools, I was able to find out the values of the variables at the time when the error was thrown. Since the expected array maximum item is 7x7x30(in array's terms that's 6x6x29 since arrays start with 0 instead of 1)

So as I took a look back at my code, I discovered an error and a non needed for loop(i).

Removing that for loop and i fixed the whole clear function and as I tested it, it works perfectly and no more further errors were discovered.



In the output bellow, person was cleared.



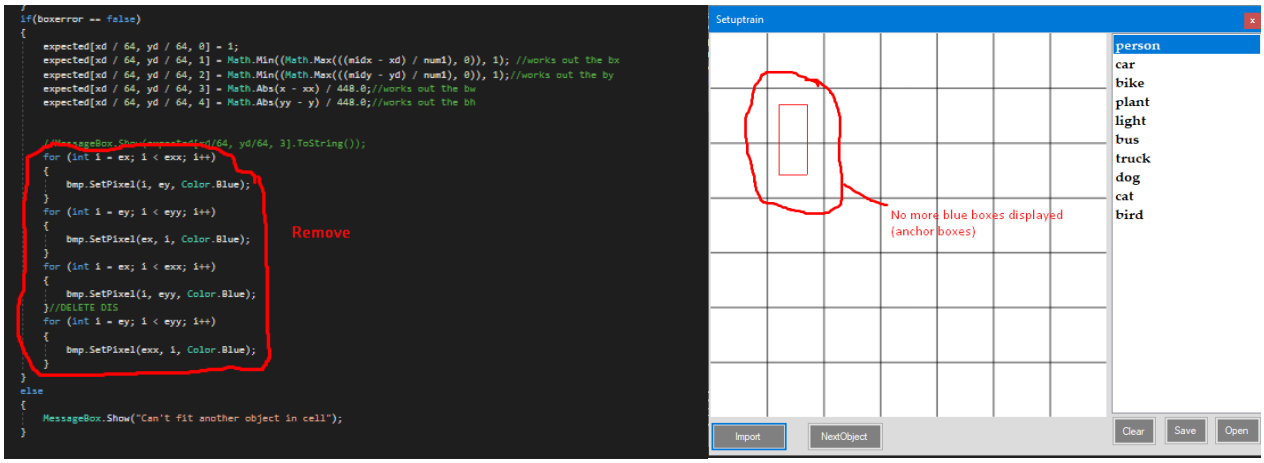
Before

After

Data set maker final

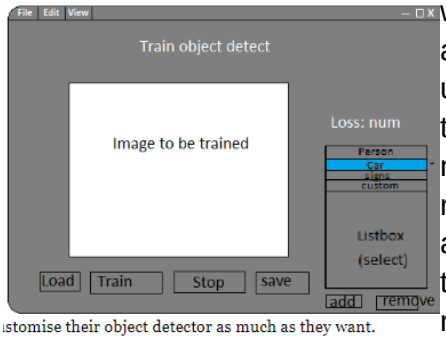
Since pretty much the whole Data set maker is functional, from opening image to selecting objects and saving the data I can confidently say that Data set maker is almost finished for proceeding with development of neural network trainer.

Other small little changes are still needed to be made to the Data set maker, for example getting rid of debugging features that were only used to identify the anchor boxes that were used.

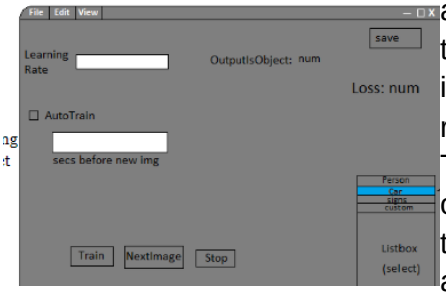


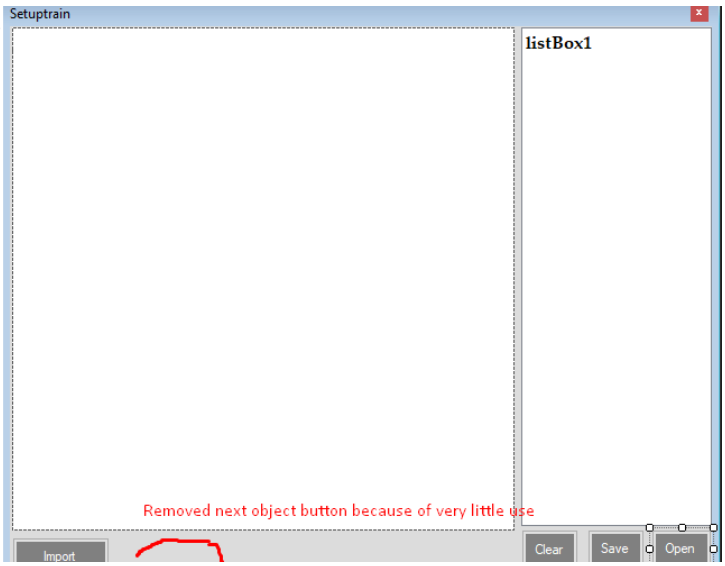
Neural network trainer

As I finished my data set maker I need to start on creating a neural network trainer in which would train the neural network using the data from data set maker and the end result would allow the user to train their neural network and able to recognise any object they train it to recognise.

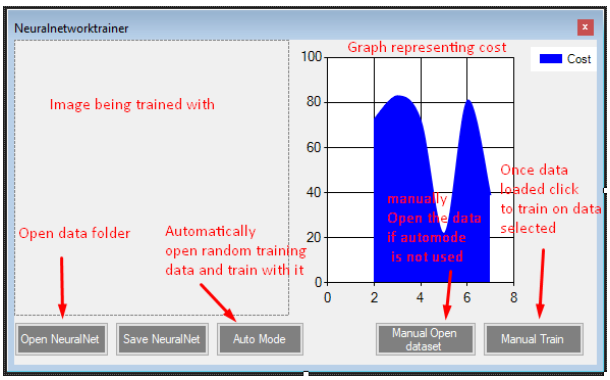


istomise their object detector as much as they want.





The first design of neural network trainer contained a few flaws as there is no need for the list box to exist and I might be adding other features like a graph showing improvements of the cost.

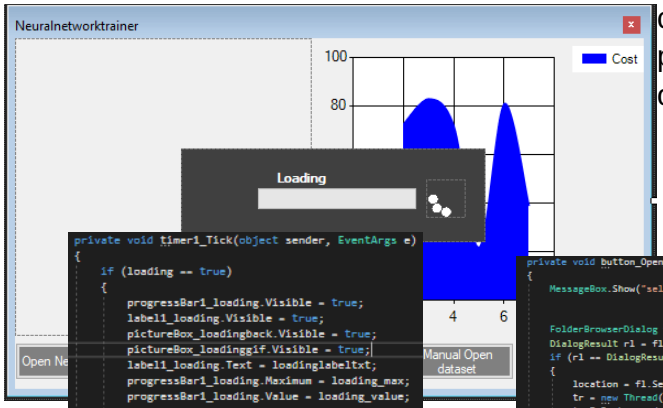


Instead of using the original design of my neural network trainer, I decided to make it more compact and add in a graph to display the progress of training.

Auto Mode and stop are on top of each other, once auto mode is clicked stop button will be replacing Auto Mode which saves up space and would make it look much better and easier to use.

For the Open NeuralNet button, it would not necessarily open Neural Network data but instead allow the user to select the data folder giving the location of training data folder and also the two neural network weights. Save button would also use the location provided by

open neural net button to save the weights without a need of asking the user to select the location again.
Since Neural network trainer will require for us to load up the weights, I decided to re-use the same loading progress bar.
As we will need to load up both of the weights(w1 & w2) I decided to also re-use the same code to load up the weights in my program in which would save up on a lot of time.




```
private void timer1_Tick(object sender, EventArgs e)
{
    if (loading == true)
    {
        progressBar1_loading.Visible = true;
        label1_loading.Visible = true;
        pictureBox_loadingback.Visible = true;
        pictureBox_loadinggif.Visible = true;
        label1_loading.Text = loadinglabeltxt;
        progressBar1_loading.Maximum = loading_max;
        progressBar1_loading.Value = loading_value;


    }
    else if (loading == false)
    {
        progressBar1_loading.Visible = false;
        label1_loading.Visible = false;
        pictureBox_loadinggif.Visible = false;
        pictureBox_loadingback.Visible = false;
    }
}
```

```
private void button_Open_Click(object sender, EventArgs e)
{
    MessageBox.Show("select the data folder");


    FolderBrowserDialog f1 = new FolderBrowserDialog();
    DialogResult r1 = f1.ShowDialog();
    if (r1 == DialogResult.OK && f1.SelectedPath != "")//checks if file was opened successfully
    {
        location = f1.SelectedPath.ToString();//location of the data folder
        tr = new Thread(load);//creates a thread in which would load up the weights without freezing the whole program
        tr.IsBackground = true;
        tr.Start();
        //var readw1 = File.ReadAllLines(location + "\\w1.txt");
        //string[] sr = new string[21];
        /*for (int i = 1; i < readw1.Count(); i++)
        {
            //listBox1.Items.Add(readmisc[i]);
        }*/
    }
    else
    {
        MessageBox.Show("Invalid file path");
    }
}
```

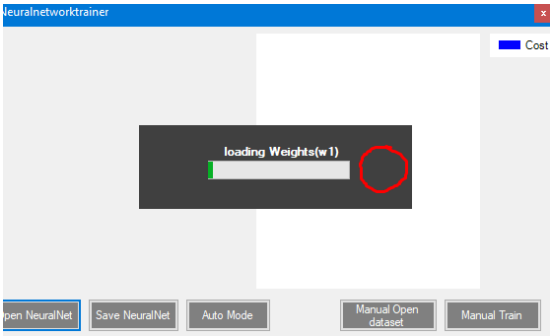
```
private void load()
{
    var readw1 = File.ReadLines(location + "\\w1.txt");//specifies the location of the file
    var readw2 = File.ReadLines(location + "\\w2.txt");//weights
    loading = true;
    loading_max = 211542016; //size of w1 + w2
    loading_value = 0;
    int teemp = 0;
    foreach (var line in readw1)//loops 205520896 times
    {

        w1[teemp] = Convert.ToDouble(line);//stores each line in w1.txt to a 1d double array
        teemp++;//incremeants the temporary value
        loading_value++;//increase the loading progressbar value
        loadinglabeltxt = "loading Weights(w1)";
    }
    teemp = 0;
    foreach (var line in readw2)//loops 205520896 times
    {

        w2[teemp] = Convert.ToDouble(line);//stores each line in w1.txt to a 1d double array
        teemp++;//incremeants the temporary value
        loading_value++;//increase the loading progressbar value
        loadinglabeltxt = "loading Weights(w2)";
    }
    loadinglabeltxt = "";
    loading_max = 0;
    loading_value = 0;
    loading = false;//dissables loading and sets all values back to default
    tr.Abort();
}
```

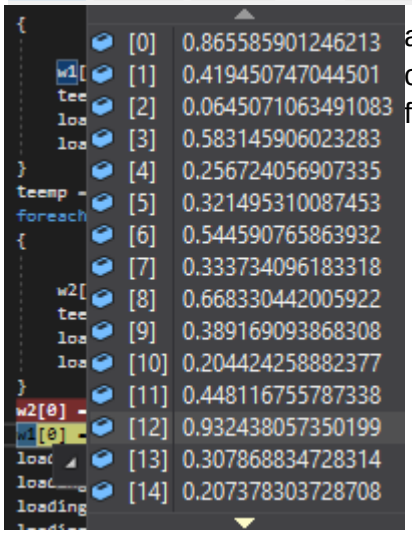
Neural network trainer code pt1

As stated before I re-used the weight loading function in which was not time consuming nor causing any problems.

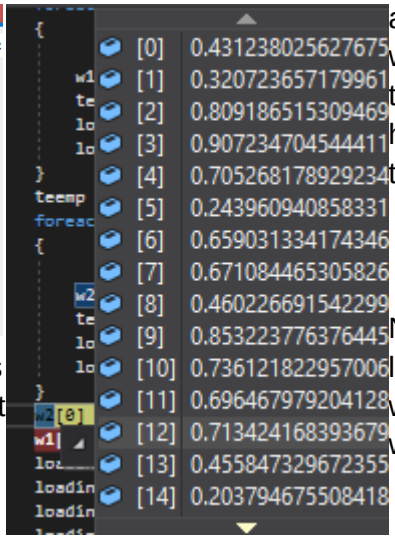
So as I finished coding the weight loading function, I decided to debug it and test if weights are actually loaded in and stored in arrays of w1 and w2.
When I first tested it, I discovered a very small bug in which did not matter at all, but I will still want to fix it later as the appearance of my program is important too.

Later when my program loaded up the weights, I used visual studio's debugging tools to check the values within the array w2, which is





and w2 has confirm that function

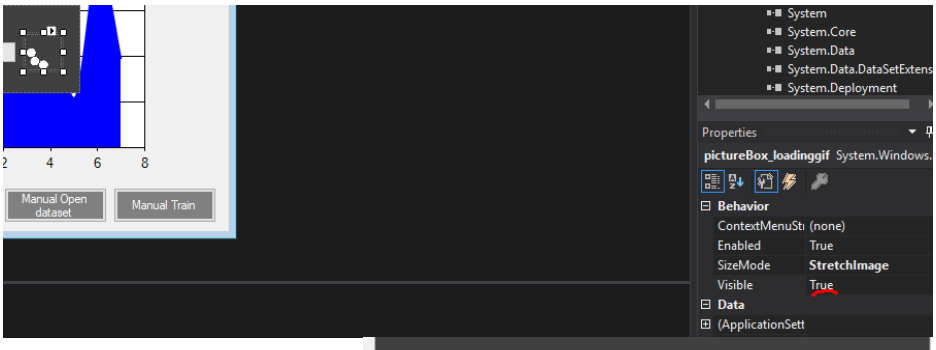


a success as the weights seem to be loaded just how I wanted them to load.

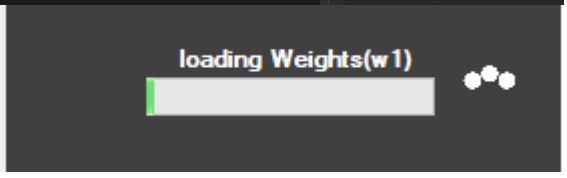
Now that both of w1 loaded in, I can weight loading works as expected.

Neural network trainer code pt2

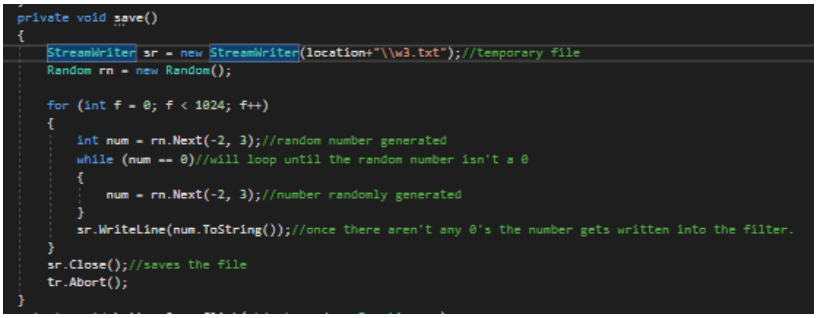
As I was troubleshooting my program, I decided to make the loading gif visible and in my code there is a function that makes it invisible if loading bar is not used.

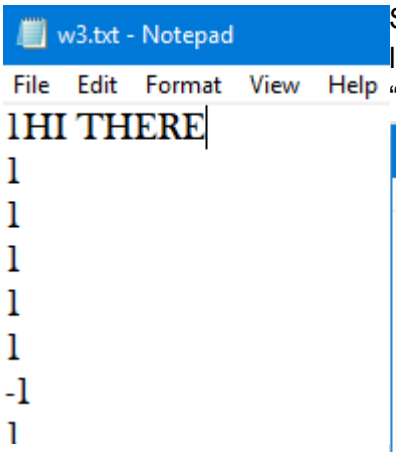


So as my program loads up, the loading gif will be hidden and as I was testing neural network

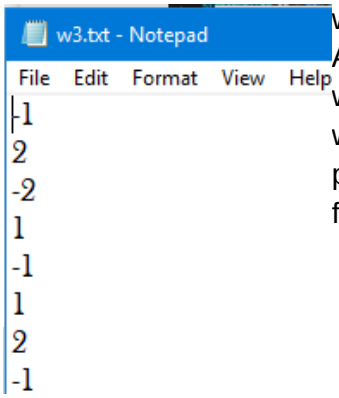


trainer, that one small changed managed to fix the issue.
For the save function in which should simply save w1 and w2, overwriting the previous w1 and w2.
So I re-used the same function that I used to generate random filter values but for testing purposes, I named a temporary file(w3) in the same location as w1 and w2 to see if everything would function as well as it should. One of the other reasons why I wanted to do this is to see if it would overwrite the file.





So I manually edited w3 and added “HI THERE” into the 1st line of the file and I would run the save function again to see if “HI THERE” would be removed in which would mean that it works.



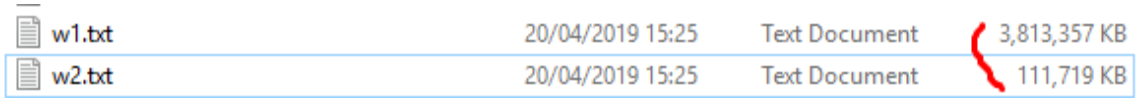
As “HI THERE” was removed it would mean that my save function is working and therefore allowing me to progress into finishing the save function.

Neural network trainer code pt2

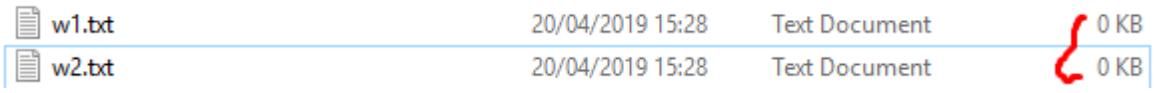
```
private void save()
{
    loading = true;
    loadinglabeltxt = "Saving Weights(w2)";
    loading_max = 211542016;
    loading_value = 0;
    StreamWriter sr = new StreamWriter(location+"\\w1.txt");//w1
    StreamWriter sr2 = new StreamWriter(location + "\\w2.txt");//w1
    Random rn = new Random();

    for (int f = 0; f < w1.Length; f++)
    {
        sr.WriteLine(w1[f].ToString());//once there aren't any 0's the number gets written into the filter.
        loading_value++;
    }

    for (int f = 0; f < w2.Length; f++)
    {
        sr2.WriteLine(w2[f].ToString());//once there aren't any 0's the number gets written into the filter.
        loading_value++;
    }
    sr.Close();//saves w1 file
    sr2.Close();//saves w2 file
    loading = false;
    loadinglabeltxt = "";
    loading_max = 0;
    loading_value = 0;
    saving = false;
    tr.Abort();
```

Now after some editing to my function, I made it so this function would save both files(w1 & w2) and I also added in some extra features in which will not allow the user to run the functions if the weights have not been loaded first and if it's not already saving.
I am certain that this should work but just in case, I will need to test it and see if it will save and work perfectly.
And as it seems to be working so as all of the security features to not break the program.

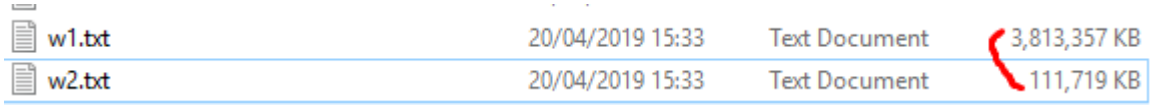
Before saving



During saving



After saving



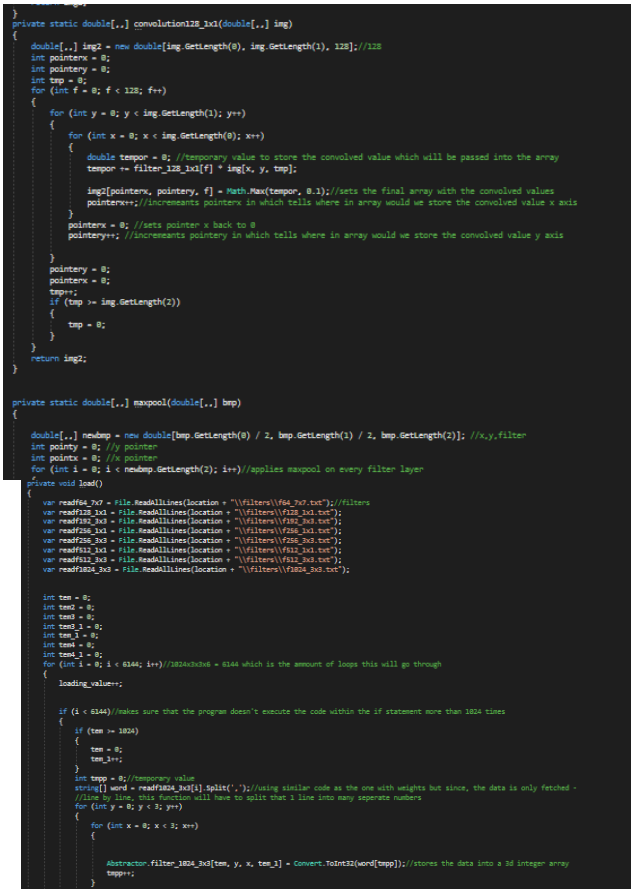
So it seems to me that during saving, the file gets set to as empty and then new data replaces the old one therefore overwriting the file.

Neural network trainer problem

During the development of my Neural network trainer, I stumbled across one big problem. In order for me to train the neural network, I will need the image abstracted into a 7x7x1024 array(50176 inputs). Since the abstraction takes a lot of time, making the whole training process really slow, I decided to abstract the image during the dataset maker stage which would still be time consuming, but it would speed up the training of neural networks and once the image is abstracted, it won't need to be abstracted again.

Solution:

Solution for this is quite simple, instead of having the image abstractor on the main form, I will have the image abstractor in a separate class in which could be accessed by the main form and even dataset maker.
So I created this class named Abstractor and then I simply copied the abstraction function from the main form into the abstractor class.
As I was copying the abstractor function I discovered another small problem that I would be facing and that is to load the filters. So I decided to instead of loading the filters in the main form and storing it into their array's I will change it so when the user loads all the weights, the filters will be loaded and stored in the Abstraction class. Same for data set maker in case if the user decides to use the dataset maker first instead of the main form.



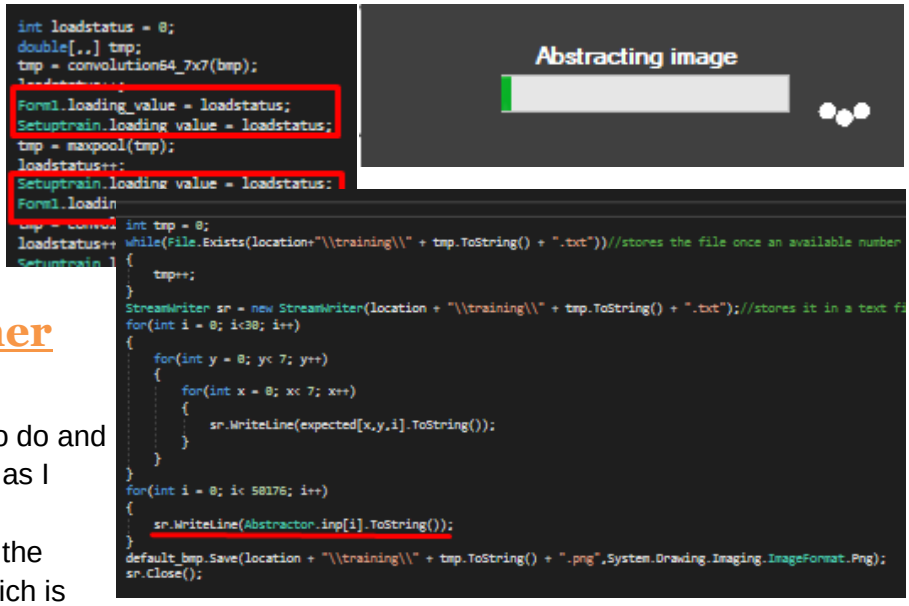
```
}
private static double[,,] convolution128_1x1(double[,,] img)
{
    double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 128];//128
    int pointerx = 0;
    int pointery = 0;
    int tmp = 0;
    for (int f = 0; f < 128; f++)
    {
        for (int y = 0; y < img.GetLength(1); y++)
        {
            for (int x = 0; x < img.GetLength(0); x++)
            {
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                tempor += filter_128_1x1[f] * img[x, y, tmp];

                img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                pointerx++;//increaments pointerx in which tells where in array would we store the convolved value x axis
            }
            pointerx = 0; //sets pointer x back to 0
            pointery++; //increaments pointery in which tells where in array would we store the convolved value y axis

        }
        pointery = 0;
        pointerx = 0;
        tmp++;
        if (tmp >= img.GetLength(2))
        {
            tmp = 0;
        }
    }
    return img2;
}


private static double[,,] maxpool(double[,,] bmp)
{

    double[,,] newbmp = new double[bmp.GetLength(0) / 2, bmp.GetLength(1) / 2, bmp.GetLength(2)]; //x,y,filter
    int pointy = 0; //y pointer
    int pointx = 0; //x pointer
    for (int i = 0; i < newbmp.GetLength(2); i++)//applies maxpool on every filter layer
{
private void load()
{
    var readf64_7x7 = File.ReadAllLines(location + "\\filters\\f64_7x7.txt");//filters
    var readf128_1x1 = File.ReadAllLines(location + "\\filters\\f128_1x1.txt");
    var readf192_3x3 = File.ReadAllLines(location + "\\filters\\f192_3x3.txt");
    var readf256_1x1 = File.ReadAllLines(location + "\\filters\\f256_1x1.txt");
    var readf256_3x3 = File.ReadAllLines(location + "\\filters\\f256_3x3.txt");
    var readf512_1x1 = File.ReadAllLines(location + "\\filters\\f512_1x1.txt");
    var readf512_3x3 = File.ReadAllLines(location + "\\filters\\f512_3x3.txt");
    var readf1024_3x3 = File.ReadAllLines(location + "\\filters\\f1024_3x3.txt");


    int tem = 0;
    int tem2 = 0;
    int tem3 = 0;
    int tem3_1 = 0;
    int tem_1 = 0;
    int tem4 = 0;
    int tem4_1 = 0;
    for (int i = 0; i < 6144; i++)//1024x3x3x6 = 6144 which is the ammount of loops this will go through
    {
        loading_value++;


        if (i < 6144)//makes sure that the program doesn't execute the code within the if statement more than 1024 times
        {
            if (tem >= 1024)
            {
                tem = 0;
                tem_1++;
            }
            int tmpp = 0;//temporary value
            string[] word = readf1024_3x3[i].Split(',');//using similar code as the one with weights but since, the data is only fetched -
            //line by line, this function will have to split that 1 line into many seperate numbers
            for (int y = 0; y < 3; y++)
            {
                for (int x = 0; x < 3; x++)
                {


                    Abstractor.filter_1024_3x3[tem, y, x, tem_1] = Convert.ToInt32(word[tmpp]);//stores the data into a 3d integer array
                    tmpp++;
                }
```

Since, the main function also has a load function that I later made, I decided to also re-use that function and add it in the dataset maker.

As abstraction function used few minutes in order to abstract the image, I used the loading screen function to display the progress of abstraction.

Since I could not retrieve the loading status from abstraction class during execution, I made it so the abstraction class would edit the loading value of form1 and dataset maker.

As expected, it worked perfectly.



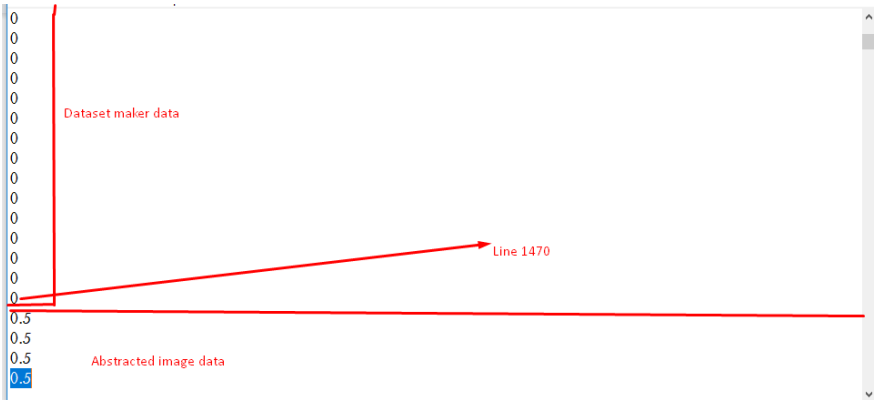
Dataset maker and neural network trainer changes

The saving was easy to do and no problems occurred, as I decided to stored the abstracted image after the dataset maker data which is after 1470th line since 7x7x30 = 1470.

```
int loadstatus = 0;
double[,] tmp;
tmp = convolution64_7x7(bmp);
Form1.loading_value = loadstatus;
Setuptrain.loading_value = loadstatus;
tmp = maxpool(tmp);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loadin
```

```
int tmp = 0;
while(File.Exists(location+"\\training\\" + tmp.ToString() + ".txt"))///stores the file once an available number
{
    tmp++;
}
StreamWriter sr = new StreamWriter(location + "\\training\\" + tmp.ToString() + ".txt");//stores it in a text fi
for(int i = 0; i<30; i++)
{
    for(int y = 0; y< 7; y++)
    {
        for(int x = 0; x< 7; x++)
        {
            sr.WriteLine(expected[x,y,i].ToString());
        }
    }
}
for(int i = 0; i< 50176; i++)
{
    sr.WriteLine(Abstractor.inp[i].ToString());
}
default_bmp.Save(location + "\\training\\" + tmp.ToString() + ".png",System.Drawing.Imaging.ImageFormat.Png);
sr.Close();
```

As you can see in the text file. Data was separated and saved successfully.



Once I finished again with the dataset maker, I went back on to creating the neural network trainer and before I can code the back propagation, I needed to load the data from the dataset maker and as we know that the abstracted image starts after like 1470 then we can just simply store both the dataset maker data and the abstracted image as well.

```
private void neuralnetbackprop()
{
    while (true)
    {
        if (stop == false)//while automode is still enabled keep looping
        {
            int f = 0;
            while (File.Exists(location + "\\training\\" + f.ToString() + ".txt"))
            {
                f++;
            }
            Random rn = new Random();
            f = rn.Next(0, f);
            while (!File.Exists(location + "\\training\\" + f.ToString() + ".txt"))
            {
                f = rn.Next(0, f);// checks if there are no empty spots.
            }
            MessageBox.Show(f.ToString());
            var readmisc = File.ReadAllLines(location + "\\training\\" + f.ToString() + ".txt");
            bp = new Bitmap(location + "\\training\\" + f.ToString() + ".png");
            pictureBox_outp.Image = bp;
            int tmp = 0;
            for (int i = 1470; i < readmisc.Count(); i++)
            {
                inp[tmp] = Convert.ToDouble(readmisc[i]);
                tmp++;
                //listBox1.Items.Add(readmisc[i]);
            }
            tmp = 0;
            for (int i = 0; i < 30; i++)
            {
                for (int y = 0; y < 7; y++)
                {
                    for (int x = 0; x < 7; x++)
                    {
                        expected[x, y, i] = Convert.ToDouble(readmisc[tmp]);
                        tmp++;
                    }
                }
            }
            h1 = new double[4096];
            outp = new double[1470];
```

Checks if such file exists and will keep checking till it finds a random file that exists

Loads abstracted image

Loads dataset data

Once all the data has been loaded, we need to get the output of the most recent version of neural network to work out the error rate so the weights can be altered to reduce the cost.

```
//<Forward propagation so we can get the output of neural net and workout the cost(error)>
tmp = 0;
for (int i = 0; i < 4096; i++)
{
    for (int x = 0; x < inp.Count(); x++)
    {
        h1[i] += inp[x] * w1[tmp];
        tmp++;
    }
}
tmp = 0;
for (int i = 0; i < 1470; i++)
{
    for (int x = 0; x < h1.Count(); x++)
    {
        outp[i] += h1[x] * w2[tmp];
        tmp++;
    }
}
double[,,] outp_3d = new double[7, 7, 30];
tmp = 0;
for (int i = 0; i < 30; i++)
{
    for (int y = 0; y < 7; y++)
    {
        for (int x = 0; x < 7; x++)
        {
            outp_3d[x, y, i] = outp[tmp];
            tmp++;
        }
    }
}
//<  >
```

Neural network trainer research and changes

loss function:

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$
$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$
$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3)$$

https://arxiv.org/pdf/1506.02640v5.pdf

As I was doing some research about how to work out the cost for my yolo algorithm and I stumbled across this function.

Instead of having a lot of different loss functions it will only have 2 per cell each cost for anchorbox(in which there's only 2 of)

This function works out the cost for px,bx,by.bh,bw and all 10 classes and then add them up to form just a single cost, same will happen to all other anchor boxes in which again is only 2 of them.

$$\left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

The first part of the function subtracts the output By the expected in this case it's x - x^(x being the output and x^ being the expected value)

```
double[,,] cost = new double[7, 7, 2];//0 and 1
double cost_average = 0;
for (int y = 0; y < 7; y++)
{
    for (int x = 0; x < 7; x++)
    {
        double cost_bx0 = Math.Pow((outp_3d[x, y, 1] - expected[x, y, 1]), 2);//anchorbox 1
        double cost_bx1 = Math.Pow((outp_3d[x, y, 16] - expected[x, y, 16]), 2);//anchorbox 2

        double cost_by0 = Math.Pow((outp_3d[x, y, 2] - expected[x, y, 2]), 2);//anchorbox1
        double cost_by1 = Math.Pow((outp_3d[x, y, 17] - expected[x, y, 17]), 2);//anchorbox2

        cost[x, y, 0] = Math.Round((cost_bx0 + cost_by0),2);
        cost[x, y, 1] = Math.Round((cost_bx1 + cost_by1),2);
```

I simply then applied that function into my program.

$$\left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

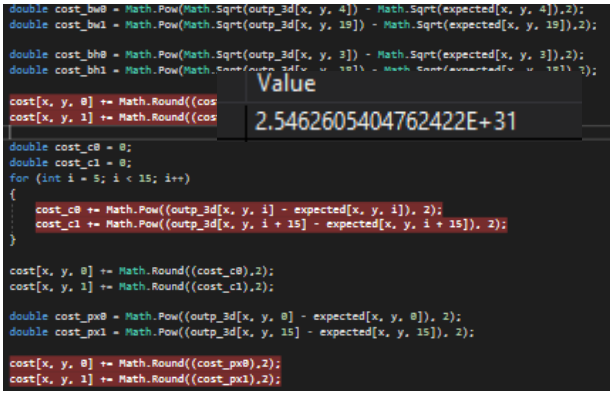
$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \quad (3)$$

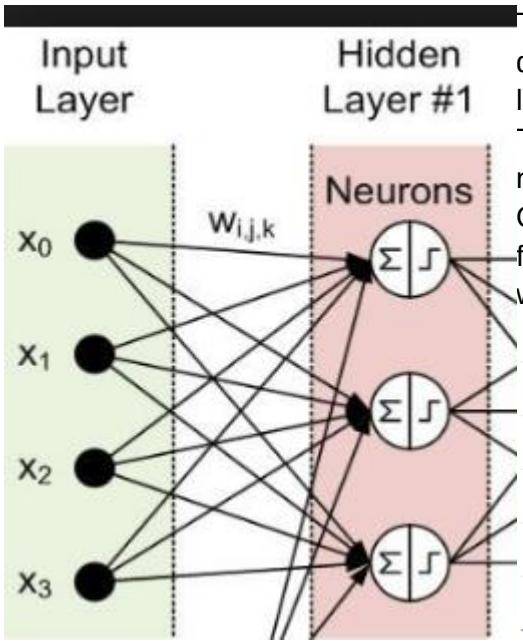
So was the same method applied for the other costs.

After the coding, I needed to do some testing to ensure that the cost would be working, but when I first tested it, I stumbled across a big problem.

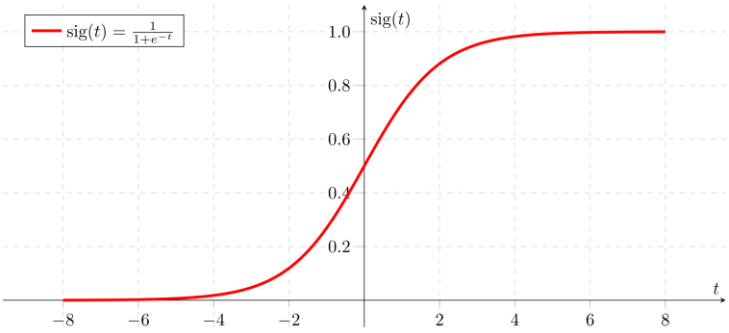
In which makes total sense considering the fact that my output is so large.

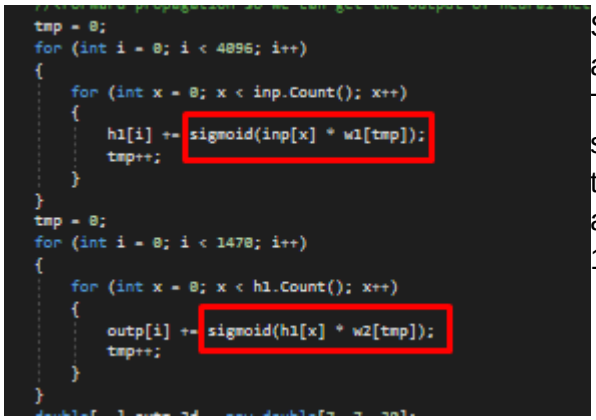


Neural network trainer problem

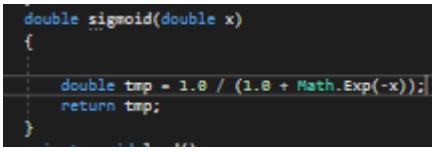


The cause for this problem is pretty clear, as I debugged before, since the neural network is so large and so many input and hidden layers are used The hidden layer would be just larger and larger the more and more input's there are.
One solution to this is to use another activation function for that layer. Instead of using leaky relu, I would be using sigmoid.





So I decided to apply a sigmoid function for h1 and output.
This function simply uses the equation above 1/1+e^-x.



So as you may see. The values are still Relatively high, but they are significantly smaller. Although the output is really high and another method will have to be Used.

H1

[0]	28377.697337725254
[1]	28382.969967857178
[2]	28383.670188247059
[3]	28362.795002223469
[4]	28354.044778889085
[5]	28374.132138714871
[6]	28363.624017130562
[7]	28356.363051353972
[8]	28368.006190534619
[9]	28386.402808019004
[10]	28367.057457964664
[11]	28377.842003383957
[12]	28378.149048136427
[13]	28362.244884547494
[14]	28377.815677308867

Outp

[0]	4095.8407607578538
[1]	4095.999955955484
[2]	4095.8062719519316
[3]	4095.9999999297725
[4]	4095.9999639829357
[5]	4095.927975183648
[6]	4095.9917425605722
[7]	4095.7229192987784
[8]	4095.8652992091211
[9]	4095.8338715297546
[10]	4095.7395735288032
[11]	4095.7929708529732
[12]	4095.9476825574334
[13]	4095.469839680386
[14]	4095.76269434328

So I did some research and stumbled across this website: https://medium.com/usf-msds/deep-learning-best-practices-1-weight-initialization-14e5c0295b94
It stated some dissadvantages with using a randomised weights, in which I have and that might be the case for why I am getting a really high output.

Neural

2. Initializing weights randomly

Initializing weights randomly, following standard normal distribution (`np.random.randn(size_l, size_l-1)` in Python) while working with a (deep)

$$\sqrt{\frac{2}{size^{[l-1]}}}$$

$$W^{[l]} = np.random.randn(size\_l, size\_l\text{-}1) * np.sqrt(2/size\_l\text{-}1)$$

network trainer problem and research

So I used the equation above to change the weights temporarily just to see if it would fix the problem.
Since the weights originally generated were random, I don't need to generate a new number.

Just as an extra change, I also decided to normalise my input, which has a lot of advantages, one of which is to make all inputs small since they will all add up to 1.
To normalise a number we need to add up all the numbers that will need to be normalised and then take each number and divide by that number, giving us the normalised sum.
Eg: normalised num1 = num1 / (num1+num2+num3+num4…) and so on…

```
for(int ff = 0; ff< inp.Count(); ff++)
{
    tmpp += inp[ff];
}
for(int ff = 0; ff< inp.Count(); ff++)
{
    inp[ff] = inp[ff] / tmpp;
}
```

```
        {
            h1[i] += inp[x] * (w1[tmp]) * Math.Sqrt(2.0/1.0);
            tmpp += h1[i];
            tmp++;
        }
    }

    tmp = 0;
    for (int i = 0; i < 1470; i++)
    {
        for (int x = 0; x < h1.Count(); x++)
        {
            outp[i] += h1[x] * (w2[tmp]) *Math.Sqrt(2.0/1.0);
            tmp++;
        }
```

So after testing, I have found some very successful results. The algorithm has indeed made h1 and outp better to deal with.

H1

[0]	0.75746252057729291
[1]	0.69773706834421456
[2]	0.81419421139388026
[3]	0.684701454107823
[4]	0.80557514586111345
[5]	0.71393226694645
[6]	0.67805838942414165
[7]	0.61609924863530019
[8]	0.68479872425197263
[9]	0.72408524689127074
[10]	0.64736761228353867
[11]	0.70142945544032576
[12]	0.70871470537781545
[13]	0.71931621152090319
[14]	0.71741352373494627

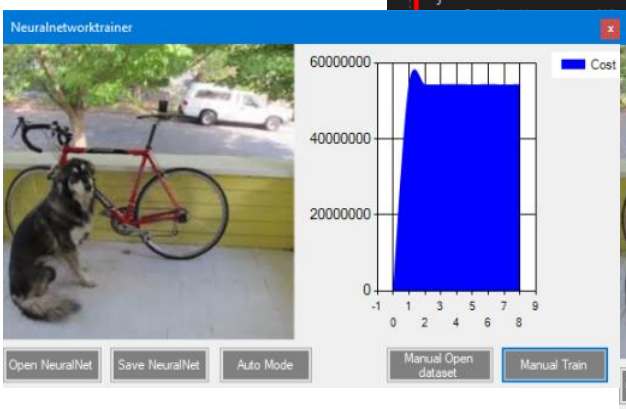
Outp

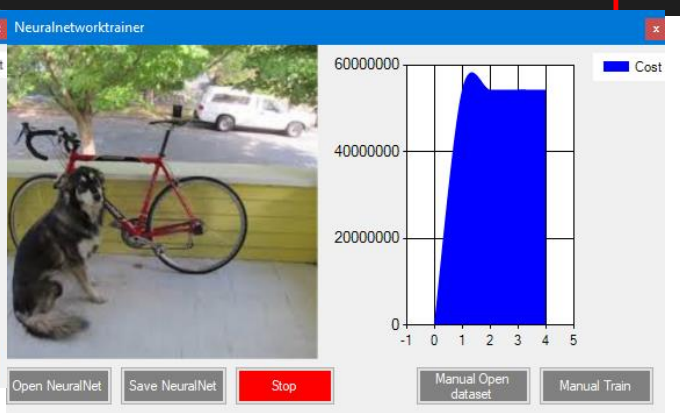
[0]	2057.4042277378826
[1]	2023.7113704552146
[2]	2064.9262119020432
[3]	2044.474723737344
[4]	2018.5864830763981
[5]	2039.5265658615679
[6]	2060.4582132688538
[7]	2056.6238013325205
[8]	2060.7489873572617
[9]	2050.4447086294931
[10]	2020.4023380520086
[11]	2041.5402002486628
[12]	2048.2969083820958
[13]	2066.5573503207911
[14]	2036.7046156226652

Neural network trainer problem

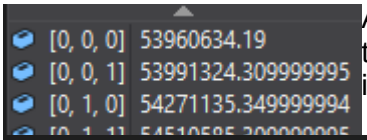
So I temporary implemented that equation to change the weights.

```
private void save()
{
    loading = true;
    loadinglabeltxt = "Saving Weights(w2)";
    loading_max = 211542816;
    loading_value = 0;
    StreamWriter sr = new StreamWriter(location+"\\w1.txt");//w1
    StreamWriter sr2 = new StreamWriter(location + "\\w2.txt");//w1
    Random rn = new Random();

    for (int f = 0; f < w1.Length; f++)
    {
        double n = w1[f] * Math.Sqrt(2.0 / 1.0);
        sr.WriteLine(n.ToString());//once there aren't any 0's the number gets written into the filter.
        loading_value++;
    }

    for (int f = 0; f < w2.Length; f++)
    {
        double n = w2[f] * Math.Sqrt(2.0 / 1.0);
        sr2.WriteLine(n.ToString());//once there aren't any 0's the number gets written into the filter.
        loading_value++;
    }
```





As you might see, the same cost is outputted, in which proves the fact that changing the weights have worked.

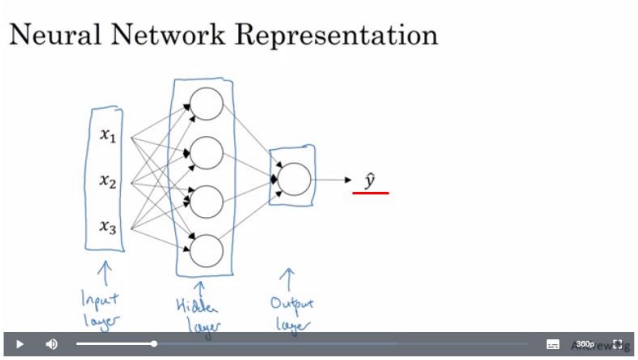


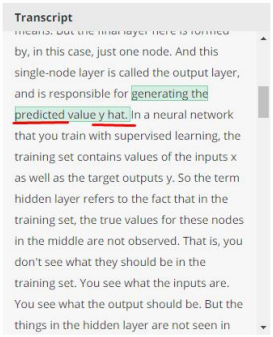
After I finished with changing the randomised weights, I saw that the average cost was too high, in which it could either mean that it's working or that something is wrong with my cost algorithm.

```
double cost_bx0 = Math.Pow((outp_3d[x, y, 1] - expected[x, y, 1]), 2);//anchorbox 1
double cost_bx1 = Math.Pow((outp_3d[x, y, 16] - expected[x, y, 16]), 2);//anchorbox 2

double cost_by0 = Math.Pow((outp_3d[x, y, 2] - expected[x, y, 2]), 2);//anchorbox1
double cost_by1 = Math.Pow((outp_3d[x, y, 17] - expected[x, y, 17]), 2);//anchorbox2
```

The problem might be the fact that, I misstakened y-hat & x-hat for expected value, rather than predicted.
https://www.coursera.org/lecture/neural-networks-deep-learning/neural-network-representation-GyW9e and from this video, it seems like I did. So the simple solution is to just swap the output and expected around.

$$\left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2\right]$$



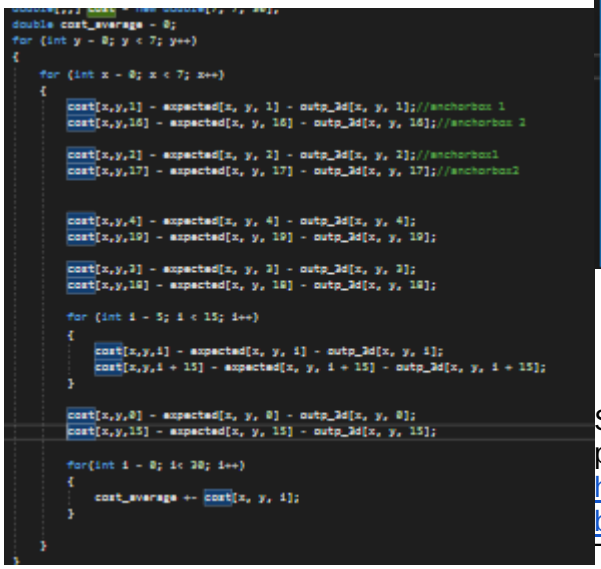

Transcript
by, in this case, just one node. And this single-node layer is called the output layer, and is responsible for generating the predicted value y hat. In a neural network that you train with supervised learning, the training set contains values of the inputs x as well as the target outputs y. So the term hidden layer refers to the fact that in the training set, the true values for these nodes in the middle are not observed. That is, you don't see what they should be in the training set. You see what the inputs are. You see what the output should be. But the things in the hidden layer are not seen in

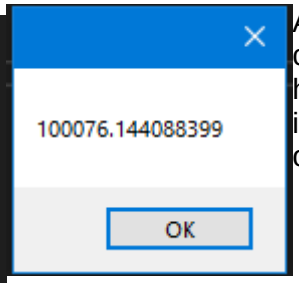
Neural network trainer problem

cost_bx0	4258968.54233694
cost_bx1	4134101.1761867986
cost_by0	4155567.5991715854
cost_by1	4196345.8646656526

The problem is that even after that, the cost is still super high, so my solution to this is to instead of adding up all the cost into 2 different ones, I will have 30 different ones for each cell, thus not adding them up at all but storing in a double 3d array of 7x7x30. One other

advantage to this is the backpropagation and that it would be easier to set the right cost to the right delta.
I changed around the code in order to make the cost much smaller as the previous one was significantly large.

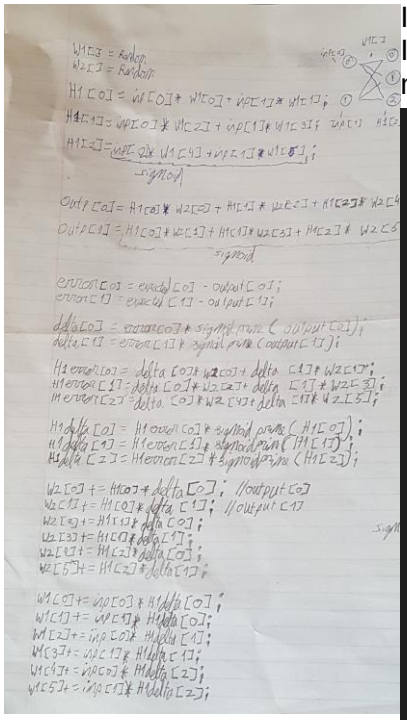




As I tested it, the average cost is still significantly high, but it might be used in backpropagation in order to minimise it.

Since everything else is set up, I will need to proceed on programming the back propagation
https://stevenmiller888.github.io/mind-how-to-build-a-neural-network/
There were a lot of interesting articles, but I found this the most useful as it explained a lot of things about back propagation, I also discovered this very helpful video in how to create a neural network, in which it explains a lot about how it all works.
https://www.youtube.com/watch?v=h3l4qz76JhQ
Essentially, we will need to do some forward propagation first to obtain the output.
HiddenLayer += weight1 * input
Output += HiddenLayer * weight2
Once we do get the output, a cost will need to be calculated and to calculate the cost we will need to:
Cost = Expected Output - Output
Now this is where backpropagation starts.
We will also need to calculate the following, depending on the neural network it might change but for an input layer, 1 hidden layer and an output.
We need to calculate the delta next and in order to do that:
Delta = cost *reluprime(output)
Then we will have to calculate the hiddenlayererror:
Hiddenerror = delta * weight2
And then:
Hiddendelta = hiddenerror * sigmoidprime(hidden)
Finally. Once we calculated all of these, we change the values of weights
we will adjust each weight to reduce the cost:
Weight1 += input * hiddendelta
Weight2 += hidden * delta

Neural network trainer coding



I created a pseudo code for a neural network.
In which was a good idea to do as it gave me the idea of how neural networks work.

```
double[..] delta = new double[7, 7, 30];
int temp = 0;
for(int i = 0; i< 30; i++)
{
    for(int y = 0; y < 7; y++)
    {
        for(int x = 0; x < 7; x++)
        {
            delta[x, y, i] = cost[x, y, i] * reluprime(outp[temp]);
            temp++;
        }
```

```
double[] h1error = new double[4096]; //h1
temp = 0;
for(int i = 0; i< 4096; i++)
{
    for(int fff = 0; fff< 30; fff++)
    {
        for(int y = 0; y< 7; y++)
        {
            for(int x = 0; x< 7; x++)
            {
                h1error[i] += delta[x,y,fff] * w2[temp];
                temp++;
            }
        }
    }
}
```

```
uble x)
{
    double tmp = 0;
    if (x > 0)
    {
        tmp = 1;
    }
    else
    {
        tmp = 0;
    }
    return tmp;
}
```

```
double[] h1delta = new double[4096];
for(int i = 0; i< 4096; i++)
{
    h1delta[i] = h1error[i] * reluprime(h1[i]);
}
temp = 0;
```

So after making the pseudocode, I created the code for backpropagation. One big problem was that the output was really high and that was a big disadvantage if you see examples above with a very large cost.

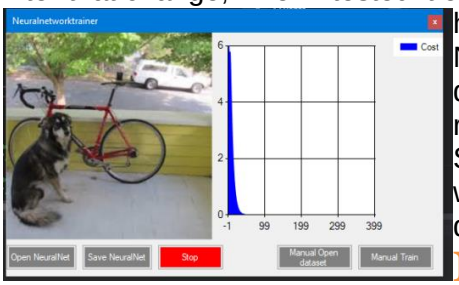
```
for(int fff = 0; fff< 30; fff++)
{
    for(int y = 0; y< 7; y++)
    {
        for(int x = 0; x< 7; x++)
        {
            for(int i = 0; i< 4096; i++)
            {
                w2[temp] += h1[i] * delta[x, y, fff]* 100000000000;
                temp++;
            }
        }
    }
}
```

```
            h1[i] += inp[x] * w1[tmp];
            tmpp += h1[i];
            tmp++;
        }
    }
    for(int i = 0; i< 4096; i++)
    {
        h1[i] = h1[i] / tmpp;
    }

    tmp = 0;
```

```
temp = 0;
for(int i = 0; i < 4096; i++)
{
    for(int fff = 0; fff < inp.Count(); fff++)
    {
        w1[temp] += (inp[fff] * h1delta[i])*1000000000;
        temp++;
    }
}
temp = 0;
```

So I normalised the hidden Layer In which made the hidden layer value very small so small that I required to Multiply the new weight by a Very very large number '*10000000000'

After that change, when I tested it once more. I received a very successful result after few hours.

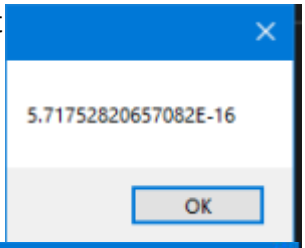


Note: the smaller the cost the better and this test was done only using one image thus making the training really fast.

Since I only wanted to see if the algorithm worked in which it did I could now test it with a slightly larger dataset.

Neural network trainer testing

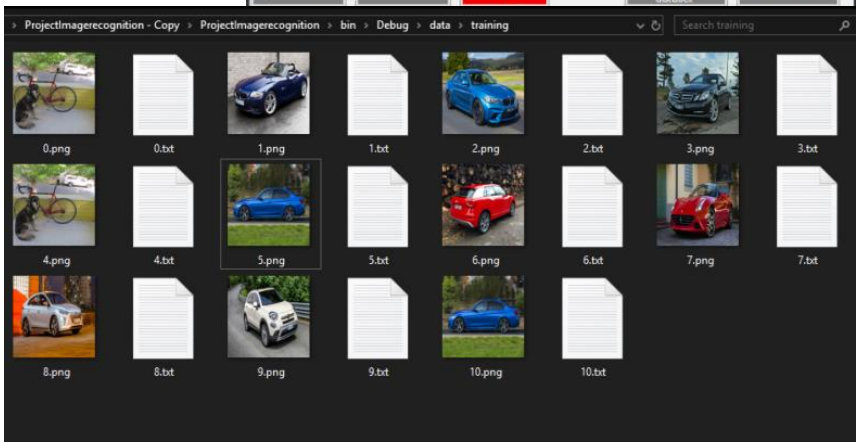
Just to prove how well the algorithm worked, this is the average cost of the test on previous page.



Since the last test was very successful I tried it with a couple of car pictures to see if it will also work.

Note: this is few minutes into the training the x part in graph represents how many cycles of training the neural network has gone through, whilst the y represents the cost.

This is the following dataset being used all made using my dataset maker.
Each image gets picked randomly for training.



Around an hour of training

More than an hour of training

Theoretically, after the cost becomes super low, I can simply just run the forward propagation on an image with a car and it would give me the location of that car, in which I would display the output like I did in dataset maker.

Since training will take a very long time and deadline is very close, I am not able to finish it by that time.

Testing

Testing

For testing, I will test my program using the test data that I previously made during design.

Test data

In my first prototype version 0.1, I had to test if basic input and output is working and if the first time setup is fully functional, so I decided to make the following test data to see if it works exactly how I want it to work.

- Click the main detection button and see if it displays a simple image to see if it works.
- Use every function in toolbox and see if they all work and function
- See if first time setup works perfectly(if everything loads in)

Prototype version 0.2 would be about image abstraction(max pooling and convolution). I will need to see if these functions would work perfectly by checking:

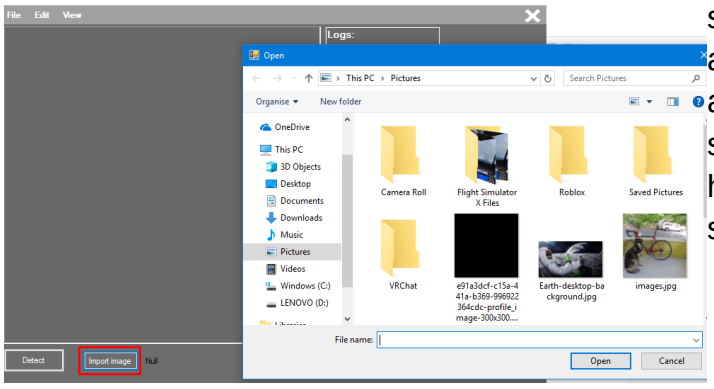
- If maxpool function would sample down the image to the right size
- If convolution function would apply the filter to the image and output the right sized array

Prototype 0.3 would link up the the max pool function with the convolution function

- Check if the max pooling and convolutions follow the chosen algorithm
- Make the convolution and max pooling link up
- Check if the input image would be outputted as a completely abstracted image.

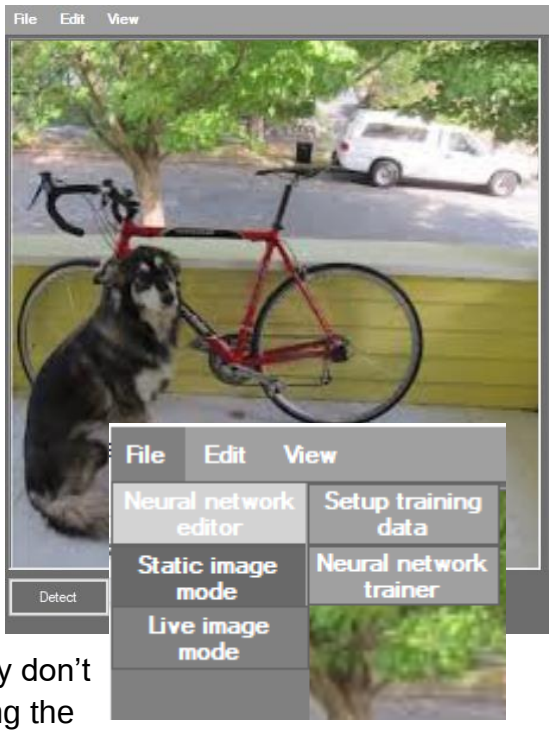
Prototype 0.4 this would be a very big update as it would require the main image recognition function to be made

- Check if the image recognition recognises a simple image
- Check if it will recognise multiple objects in image



So the first test data was to see if main detection button works and to see if an image is able to appear on screen. Since I did some changes to my program I have an import image button separate from the detection button.

Other than that I can say that it was a successful test.

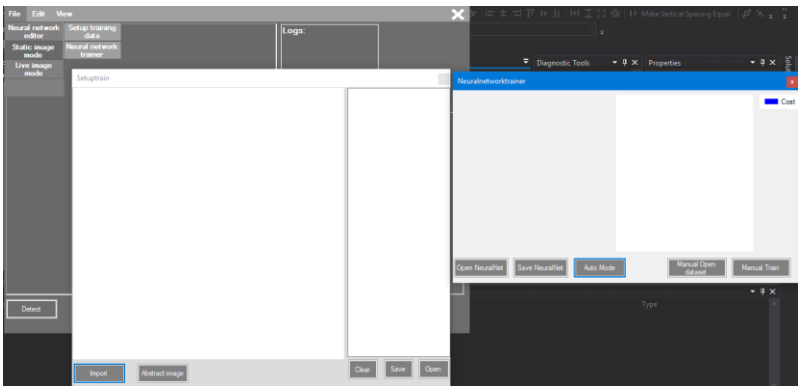


Testing 2

For the next test, I will be required to check if every function in toolbox works perfectly.

As I hover over file these functions appear and also setup training data & neural network trainer when I hover over neural network editor.
Static image mode and Live image mode currently don't do anything and possibly will never will considering the fact that my project takes a long time to process data and abstract images/load weights etc.
Edit & view also don't currently do anything as they are future features, that would allow the user to change the style of the program.
Another limitation of this is that sometimes on corners the setup training and neural trainer close which is a small bug.
Other than that, the tool box has some other effects when the mouse hovers over a button, it changes shade and it can successfully open up Setup training data form and Neural network trainer form.

As the training data stated, "Use every function in toolbox and see if they work and function" in which some of my buttons don't so I have yet not passed through that test yet.

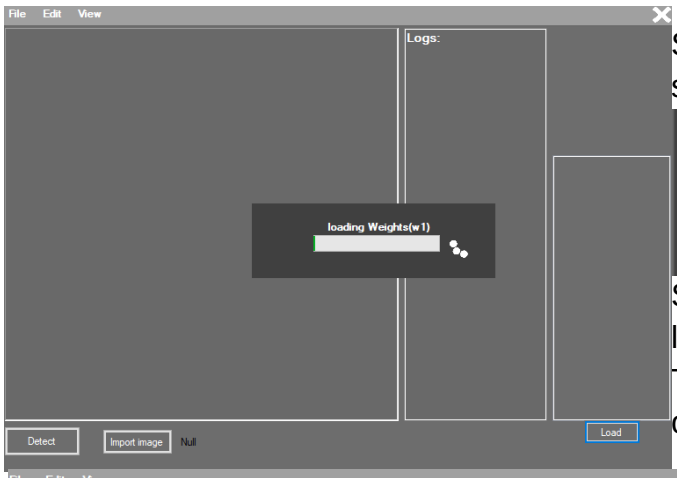


Testing 3

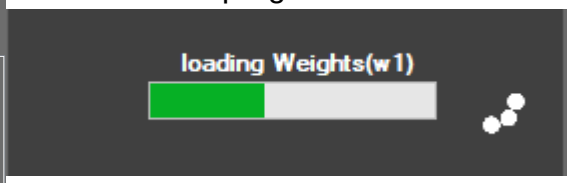
The next test that I need to do is

- See if first time setup works perfectly(if everything loads in)

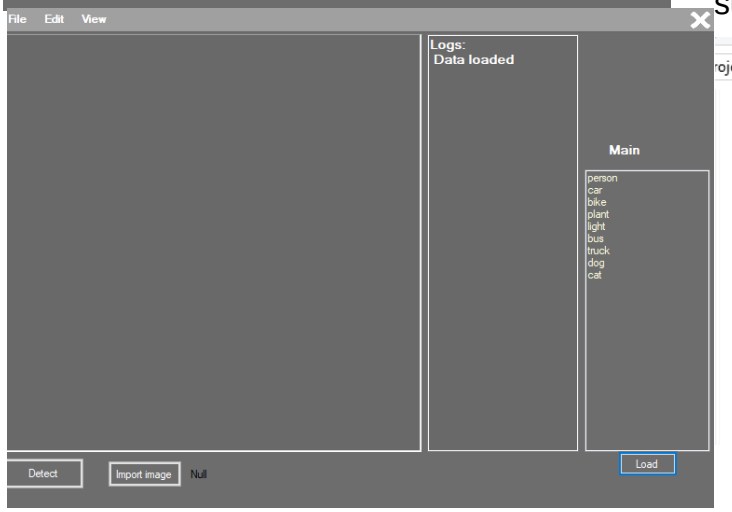
By that I need to see if all my weights and all my data will load in

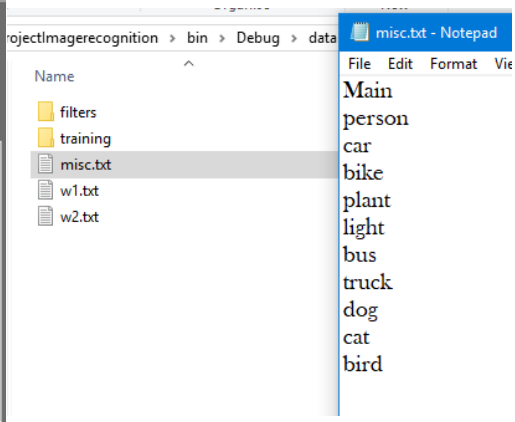


So far for the main form loading seems to be in progress

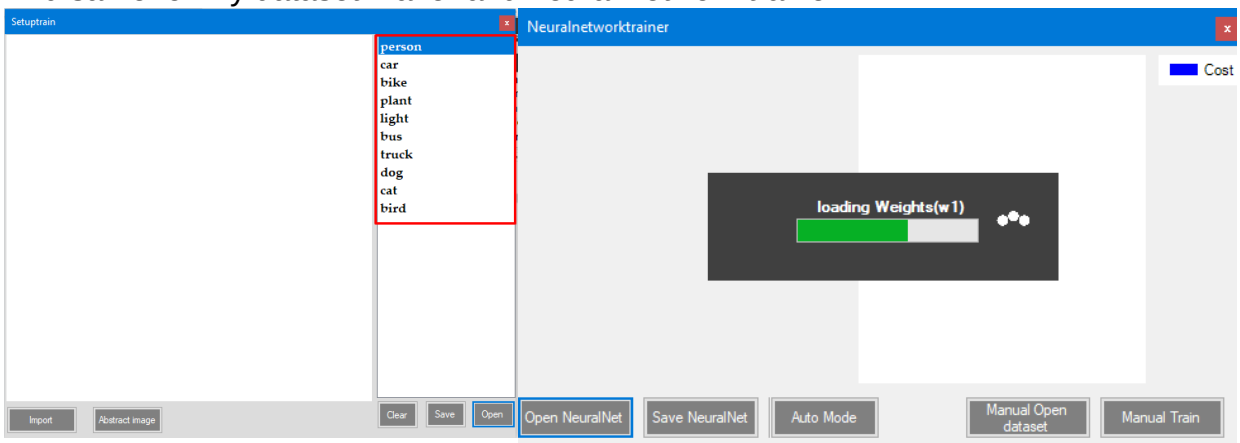


So after all the loading, the data is loaded and for even more proof The data loaded is the exact same one as in the data folder thus successfully loading in the data





And same for my dataset maker and neural network trainer.



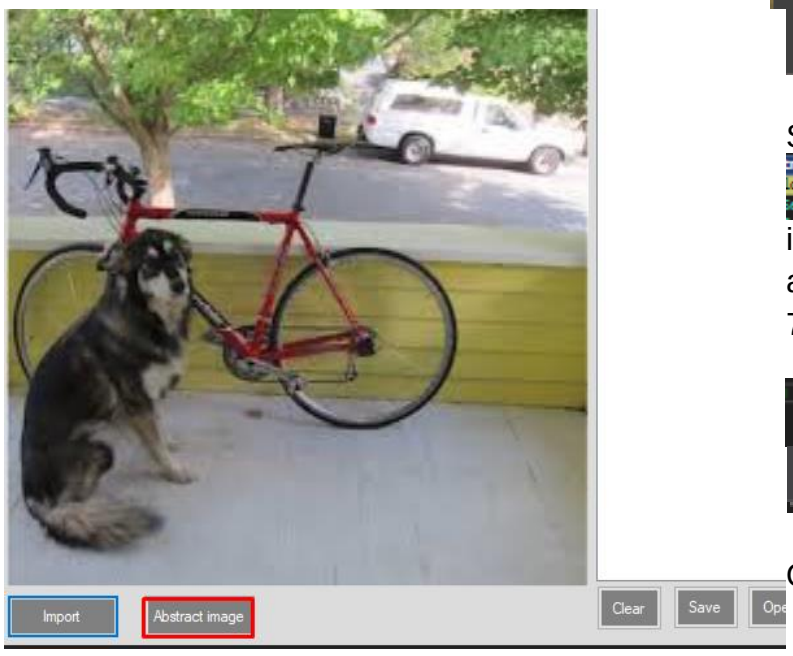
So as the data can load up successfully, I can now say that this test was successful.

Testing 4

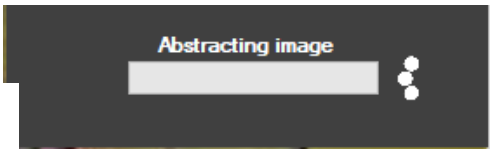
- If maxpool function would sample down the image to the right size
- If convolution function would apply the filter to the image and output the right sized array

rototype 0.3 would link up the the max pool function with the convolution function

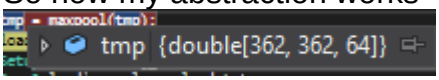
- Check if the max pooling and convolutions follow the chosen algorithm
- Make the convolution and max pooling link up
- Check if the input image would be outputted as a completely abstracted image.



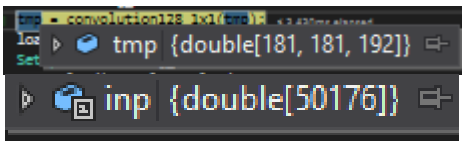
For this test I will be required to test out the image abstractor and see if it successfully abstracts an image.



So how my abstraction works



is it takes a 448x448 image and shrinks it down to 7x7x1024



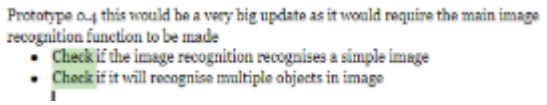
Or 50176 in 1d

As I displayed the abstraction in later development, I am certain that my image abstractor is working, only downside was that some filters were bad and in order for me to see which ones are causing the problem the most I would need to finish with neural networks and as I tested before on the cars

Around an hour of training

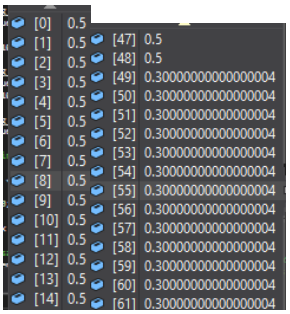
More than an hour of training

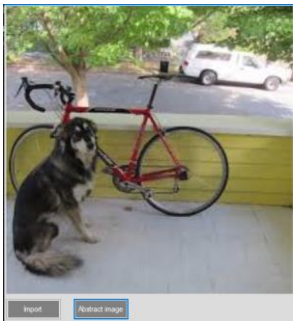
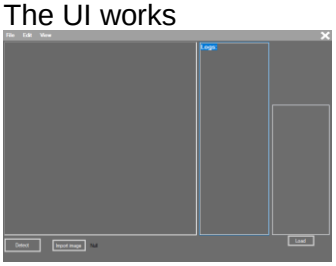
As you may see, the cost has decreased and the fact that i'm able to get the cost indicates the fact that my image abstractor is working.



For prototype 0.4, I am yet unable to test it as I have not yet created a method to display the output on the image and since the training takes a very very long time, I am unable to complete tests for prototype 0.4

Evaluation

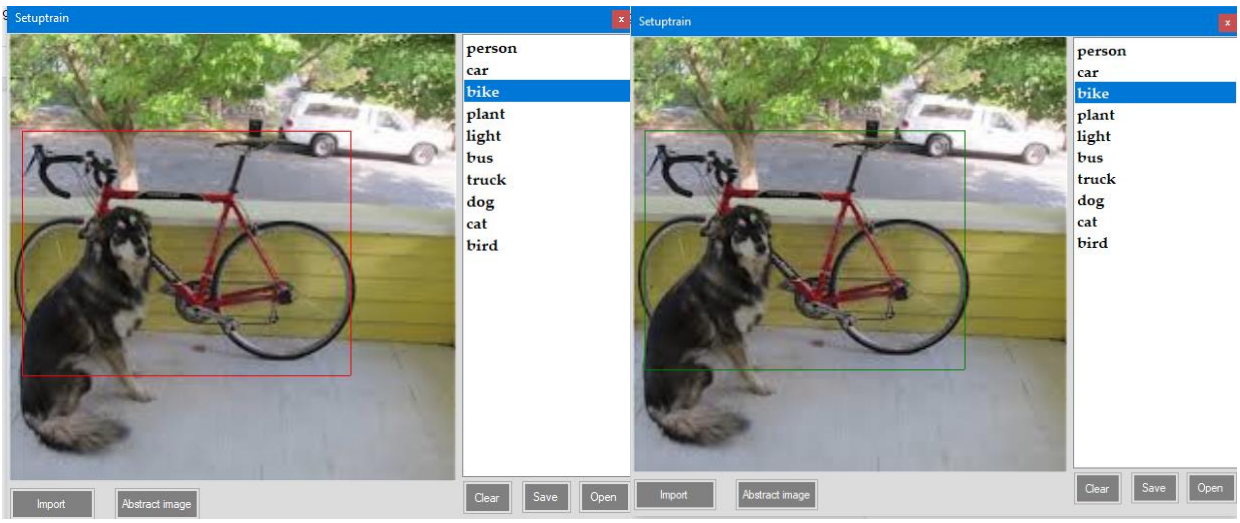
Success criteria	successful/partially successful/ unsuccessful	proof
Image Abstractor working perfectly(with all filters applied)	Partially successful	Some of the filters don't work as well. 
Neural network fully functional(can be trained)	Successful	More than an hour of training 
Must display the input image and the abstracted image	Partially successful	I am able to display the input image but I did not see the need to display the abstracted image

		
Generate boxes(show which object is has recognised)	Unsuccessful	Not completed
Show percentages(of the objects)	Unsuccessful	Not completed
Customizability	Unsuccessful	Not completed
UI fully functioning	Successful	The UI works 

The image abstractor would be working if I were to experimented a bit more with the filters it could possibly work.

The display will display just the input image and as I did not see the point of displaying the abstracted image too I decided to not implement it but only briefly during debugging.

As generating boxes and displaying them was unsuccessful due to running out of time, but I am certain that the boxes would work as I have done it similarly with the dataset maker.



Red = encrypted

Green = decrypted

In future development I would also focus on other parts like customizability, although it's not as important but it's still necessary for the user.

What could be done to improve my program

- Use a graphics processor for extra processing power in which would speed up the whole program substantially.
- Add in more user customizability features.
- Create the Bounding boxes display
- Make it more user friendly
- Make buttons more useful

Other Issues

Since my program requires a lot of processing power, then one of my programs limitations is that it requires a decent computer with more than 4gbs of ram.
This program will also require the user to have some very basic knowledge of neural networks in order to use this program properly.

Lack of help, is one of the other downsides of my program in which I will prevent that from happening in my future developments.

Code

Main form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Threading;
using System.Windows.Media;

namespace ProjectImagerecognition
{
    public partial class Form1 : Form
    {
        public Form1()
        {
```

```
    InitializeComponent();
}
bool selectedstatic = true;

static double[] w1 = new double[205520896];
static double[] w2 = new double[6021120];
static int[,,] filter_64_7x7 = new int[64,7,7];
static int[,,] filter_192_3x3 = new int[192, 3, 3];
static int[] filter_128_1x1 = new int[128];
static int[,,] filter_256_3x3 = new int[256, 3, 3];
static int[,] filter_256_1x1 = new int[256,2];//changed so more different filters
static int[,] filter_512_1x1 = new int[512,2];
static int[,,,] filter_512_3x3 = new int[512, 3, 3,2];
static int[,,,] filter_1024_3x3 = new int[1024, 3, 3,6];
public static int loading_value = 0;
static int loading_max = 0;
static bool loading = false;
static bool listboxloaded = false;
static string loadinglabeltxt = "loading";
static string location = "";
static Thread tr;

static string[] objectlabel = new string[21]; //20 objects +1
static Bitmap bp; //main image
private void pictureBox1_Click(object sender, EventArgs e)
{
    Application.Exit();

}

private void panel2_file_MouseLeave(object sender, EventArgs e)
{
    panel2_file.Visible = false;
    label1_file.BackColor = Color.FromKnownColor(KnownColor.ControlDark);
}

private void panel2_Neuraledit_MouseLeave(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = false;
}

private void pictureBox2_main_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = false;
    panel2_file.Visible = false;
    label1_file.BackColor = Color.FromKnownColor(KnownColor.ControlDark);
}

private void pictureBox2_main_Click(object sender, EventArgs e)
{
    StreamWriter sr = new StreamWriter("tmpp.txt");//temporary file
    Random rn = new Random();

    for(int f = 0; f < 1024; f++)
    {
        int num = rn.Next(-2, 3);//random number generated
        while(num == 0)//will loop until the random number isn't a 0
        {
            num = rn.Next(-2, 3);//number randomly generated
        }
        sr.WriteLine(num.ToString());//once there aren't any 0's the number gets written into the filter.
    }
    sr.Close();//saves the file
}

private void pictureBox2_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = false;

}

private void label1_file_Click(object sender, EventArgs e)
{

}
```

```
private void label1_file_MouseEnter(object sender, EventArgs e)
{
    panel2_file.Visible = true;
    label1_file.BackColor = Color.LightGray;
}

private void label1_file_MouseLeave(object sender, EventArgs e)
{
    label1_file.BackColor = Color.Gray;


}

private void label2_edit_MouseEnter(object sender, EventArgs e)
{
    label2_edit.BackColor = Color.LightGray;
    panel2_file.Visible = false;
    label1_file.BackColor = Color.FromKnownColor(KnownColor.ControlDark);
}

private void label3_view_MouseEnter(object sender, EventArgs e)
{
    label3_view.BackColor = Color.LightGray;
}

private void label3_view_MouseLeave(object sender, EventArgs e)
{
    label3_view.BackColor = Color.FromKnownColor(KnownColor.ControlDark);
}

private void label2_edit_MouseLeave(object sender, EventArgs e)
{
    label2_edit.BackColor = Color.FromKnownColor(KnownColor.ControlDark);
}

private void Form1_Load(object sender, EventArgs e)
{

    label1_staticimagemode.BackColor = Color.DimGray;

}

private void label1_neuralnetwork_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = true;
    label1_neuralnetwork.BackColor = Color.LightGray;
}

private void label1_neuralnetwork_MouseLeave(object sender, EventArgs e)
{
    label1_neuralnetwork.BackColor = Color.Gray;
    //panel2_Neuraledit.Visible = false;
}

private void label1_staticimagemode_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = false;
    label1_staticimagemode.BackColor = Color.LightGray;
}

private void label2_liveimagemode_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = false;
    label2_liveimagemode.BackColor = Color.LightGray;
}

private void label2_objectdetector_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = true;
}

private void label3_imagerecognition_MouseEnter(object sender, EventArgs e)
{
    panel2_Neuraledit.Visible = true;
```

```
}

private void label1_staticimagemode_MouseLeave(object sender, EventArgs e)
{
   if(selectedstatic == true)
   {
      label1_staticimagemode.BackColor = Color.DimGray;
   }
   else
   {
      label1_staticimagemode.BackColor = Color.Gray;
   }

}

private void label1_staticimagemode_Click(object sender, EventArgs e)
{
   selectedstatic = true;
   label1_staticimagemode.BackColor = Color.DarkGray;
   label2_liveimagemode.BackColor = Color.Gray;
}

private void label2_liveimagemode_Click(object sender, EventArgs e)
{
   selectedstatic = false;
   label2_liveimagemode.BackColor = Color.DarkGray;
   label1_staticimagemode.BackColor = Color.Gray;

}

private void label2_liveimagemode_MouseLeave(object sender, EventArgs e)
{
   if (selectedstatic == false)
   {
      label2_liveimagemode.BackColor = Color.DimGray;
   }
   else
   {
      label2_liveimagemode.BackColor = Color.Gray;
   }

}

private static void fetchdata()
{

   var readw1 = File.ReadLines(location +"\\w1.txt");//specifies the location of the file
   var readw2 = File.ReadLines(location+"\\w2.txt");//weights


   var readf64_7x7 = File.ReadAllLines(location+"\\filters\\f64_7x7.txt");//filters
   var readf128_1x1 = File.ReadAllLines(location + "\\filters\\f128_1x1.txt");
   var readf192_3x3 = File.ReadAllLines(location + "\\filters\\f192_3x3.txt");
   var readf256_1x1 = File.ReadAllLines(location + "\\filters\\f256_1x1.txt");
   var readf256_3x3 = File.ReadAllLines(location + "\\filters\\f256_3x3.txt");
   var readf512_1x1 = File.ReadAllLines(location + "\\filters\\f512_1x1.txt");
   var readf512_3x3 = File.ReadAllLines(location + "\\filters\\f512_3x3.txt");
   var readf1024_3x3 = File.ReadAllLines(location + "\\filters\\f1024_3x3.txt");
   var readmisc = File.ReadAllLines(location + "\\misc.txt");
   int teemp = 0;//temporary variable
   loading = true;//for loading progress bar, activates it
   loading_max = 211543040;//specifies the size of progressbar6
   loading_value = 0;//sets the loading value back to 0 which is the start

  foreach (var line in readw1)//loops 205520896 times
   {

      w1[teemp] = Convert.ToDouble(line);//stores each line in w1.txt to a 1d double array
      teemp++;//incremeants the temporary value
      loading_value++;//increase the loading progressbar value
      loadinglabeltxt = "loading Weights(w1)";
   }
   teemp = 0;
   foreach(var line in readw2)//loops 6021120 times, since that's how many weights and lines there are
   {
```

```
            w2[teemp] = Convert.ToDouble(line);
            teemp++;
            loading_value++;
            loadinglabeltxt = "loading Weights(w2)";
        }
        int tem = 0;
        int tem2 = 0;
        int tem3 = 0;
        int tem3_1 = 0;
        int tem_1 = 0;
        int tem4 = 0;
        int tem4_1 = 0;
        for (int i = 0; i < 6144; i++)//1024x3x3x6 = 6144 which is the ammount of loops this will go through
        {
            loading_value++;


            if (i < 6144)//makes sure that the program doesn't execute the code within the if statement more than 1024 times
            {
                if(tem >= 1024)
                {
                    tem = 0;
                    tem_1++;
                }
                int tmpp = 0;//temporary value
                string[] word = readf1024_3x3[i].Split(',');//using similar code as the one with weights but since, the data is only
fetched -
                //line by line, this function will have to split that 1 line into many seperate numbers
                for (int y = 0; y < 3; y++)
                {
                     for (int x = 0; x < 3; x++)
                     {


                        Abstractor.filter_1024_3x3[tem, y, x, tem_1] = Convert.ToInt32(word[tmpp]);//stores the data into a 3d
integer array
                        tmpp++;
                     }
                }
                tem++;
            }


            if (i < 64)
            {
                int tmpp = 0;
                string[] word = readf64_7x7[i].Split(',');
                for (int y = 0; y< 7; y++)
                {
                    for(int x = 0; x< 7; x++)
                    {


                       Abstractor.filter_64_7x7[i, y, x] = Convert.ToInt32(word[tmpp]);
                       tmpp++;
                    }
                }
            }
            if(i < readmisc.Count())// loops around how many objects there are in a text file
            {
                objectlabel[i] = readmisc[i];//stores data in another array
            }
            if(i < 128)
            {

                Abstractor.filter_128_1x1[i] = Convert.ToInt32(readf128_1x1[i]);//since filter_128_1x1 is only a 1x1 then there's no
need to make a 3d int array so this -
                //segment of code will be used just like it was used with w1 and w2
            }
            if (i< 192)
            {
                int tmpp = 0;
                string[] word = readf192_3x3[i].Split(',');
                for(int y = 0; y < 3; y++)
                {
```

```
                for(int x = 0; x<3; x++)
                {
                    Abstractor.filter_192_3x3[i, y, x] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }

            }
        }
        if (i < 256)
        {
            int tmpp = 0;
            string[] word = readf256_3x3[i].Split(',');
            for (int y = 0; y < 3; y++)
            {
                for (int x = 0; x < 3; x++)
                {
                    Abstractor.filter_256_3x3[i, y, x] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }

            }
            tem2++;
        }
        if (i < 512)
        {
            if (tem4 >= 256)
            {
                tem4 = 0;
                tem4_1++;
            }
            Abstractor.filter_256_1x1[tem4, tem4_1] = Convert.ToInt32(readf256_1x1[i]);
            tem4++;
        }


        if (i < 1024)//doubled because of filter problem
        {
            if(tem3 >= 512)
            {
                tem3 = 0;//makes sure that tem3 doesn't go over 512 which is the maximum
                tem3_1++;//increments the filter layer pivot
            }
            Abstractor.filter_512_1x1[tem3, tem3_1] = Convert.ToInt32(readf512_1x1[i]);//readf512_1x1 size is 1024 but it will
read the first half if -
            //tem3_1 = 0 and would read the last half if tem3_1 = 1
            int tmpp = 0;
            string[] word = readf512_3x3[i].Split(',');//splits the filter values when it finds a ,
            for (int y = 0; y < 3; y++)
            {
                for (int x = 0; x < 3; x++)
                {
                    Abstractor.filter_512_3x3[tem3, y, x, tem3_1] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }

            }
            tem3++;
        }
    }
    loadinglabeltxt = "loading";
    listboxloaded = true;
    loading_value = 0;//sets the values back to default
    loading_max = 0;
    loading = false;//dissables loading progress bar
    MessageBox.Show("Data fully loaded");

    tr.Abort();


}

private void timer1_Tick(object sender, EventArgs e)
{

    if (Abstractor.finished == true && Abstractor.trainorform == false)
```

```
    {
        loading = false;
        loadinglabeltxt = "";
        loading_max = 0;
        loading_value = 0;
        Abstractor.started = false;
        Abstractor.finished = false;
        neuralnetwork(Abstractor.inp);

    }
    if(loading == true)
    {
        progressBar1_loading.Visible = true;
        label1_loading.Visible = true;
        pictureBox_loadingback.Visible = true;
        pictureBox_loadinggif.Visible = true;
        label1_loading.Text = loadinglabeltxt;
        progressBar1_loading.Maximum = loading_max;
        progressBar1_loading.Value = loading_value;


    }
    else if(loading == false)
    {
        progressBar1_loading.Visible = false;
        label1_loading.Visible = false;
        pictureBox_loadinggif.Visible = false;
        pictureBox_loadingback.Visible = false;

    }
}

private void button2_load_Click(object sender, EventArgs e)
{
    timer2.Enabled = true;
    MessageBox.Show("please select the data folder");
    FolderBrowserDialog fl = new FolderBrowserDialog();
    DialogResult rl = fl.ShowDialog();

    if (rl == DialogResult.OK && fl.SelectedPath != "")
    {
        location = fl.SelectedPath.ToString();
        tr = new Thread(fetchdata);
        tr.IsBackground = true;
        tr.Start();
    }

}

private void timer2_Tick(object sender, EventArgs e)
{
    if(listboxloaded == true)
    {
        listboxloaded = false;
        for(int i = 1; i< 10; i++)
        {
            if(objectlabel[i] == null)
            {
                listBox1.Items.Add("(NULL)");
            }
            else
            {
                listBox1.Items.Add(objectlabel[i]);//objects
            }
        }
        label1_objectfilename.Text = objectlabel[0]; // the main name of the file
        textBox1_logs.Text += "\r\n Data loaded";
        timer2.Enabled = false;
    }

}

private void button_import_Click(object sender, EventArgs e)
{
    MessageBox.Show("please select the image you want to import");
```

```
    OpenFileDialog fl = new OpenFileDialog();
    DialogResult rl = fl.ShowDialog();
    if(rl == DialogResult.OK)
    {
        try
        {
            bp = new Bitmap(fl.FileName);
            Bitmap resized = new Bitmap(bp, new Size(448, 448));

            pictureBox2_main.Image = resized;
        }
        catch (Exception)
        {
            MessageBox.Show("File cannot be loaded!");
        }
    }
    else
    {
        MessageBox.Show("Could not open the image!");
    }
}

private void button1_detect_Click(object sender, EventArgs e)
{
    try
    {
        double[,,] tmp;
        Bitmap bmp = new Bitmap(pictureBox2_main.Image);

        double[] inp = new double[50176];// 1024x7x7 = 50176
        int temp = 0;//pointer

        Abstractor.trainorform = false;
        Abstractor.bmp = bmp;
        Abstractor abs = new Abstractor();
        Thread tr = new Thread(() => abs.main());
        tr.Start();
        loading = true;
        loadinglabeltxt = "Abstracting image";
        loading_max = 28;
        loading_value = 0;

    }
    catch(Exception a)
    {
        MessageBox.Show(a.Message);
    }


}
private void panel1_Paint(object sender, PaintEventArgs e)
{

}


private static double[,,] neuralnetwork(double[] inp)
{


        double[] hiddenlayer = new double[4096];//size ofx the hidden layer
        double[] outp = new double[1470];//output size(7x7x30)
        double[,,] outp_3d = new double[7, 7, 30];//create 3d array
        int tmp = 0;
        for (int i = 0; i < 4096; i++)//hidden layer size
        {
            for (int x = 0; x < inp.Count(); x++)//7x7x1024 = 50176 inputs
        {
                hiddenlayer[i] += inp[x] * w1[tmp]; //works out the hidden layer's value
                tmp++;
            }
        }
        tmp = 0;
        for (int i = 0; i < 1470; i++)
        {
```

```
                for (int x = 0; x < 4096; x++)
                {
                    outp[i] += hiddenlayer[x] * w2[tmp];
                    tmp++;
                }
            }
            tmp = 0;
            for (int i = 0; i < 30; i++)
            {
                for (int y = 0; y < 7; y++)
                {
                    for (int x = 0; x < 7; x++)
                    {
                        outp_3d[x, y, i] = outp[tmp];//simply stores the 1d double array into a 3d double array
                        tmp++;
                    }
                }
            }


            return outp_3d;
        }

        private void label2_objectdetector_Click(object sender, EventArgs e)
        {
            Setuptrain sr = new Setuptrain();
            sr.Show();
        }

        private void label3_imagerecognition_Click(object sender, EventArgs e)
        {
            Neuralnetworktrainer nw = new Neuralnetworktrainer();
            nw.Show();
        }
    }
}
```

Abstractor

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Drawing;
using System.Windows.Forms;
using System.ComponentModel;
namespace ProjectImagerecognition
{
    class Abstractor
    {
        public static int[,,] filter_64_7x7 = new int[64, 7, 7];
        public static int[,,] filter_192_3x3 = new int[192, 3, 3];
        public static int[] filter_128_1x1 = new int[128];
        public static int[,,] filter_256_3x3 = new int[256, 3, 3];
        public static int[,] filter_256_1x1 = new int[256, 2];//changed so more different filters
        public static int[,] filter_512_1x1 = new int[512, 2];
        public static int[,,,] filter_512_3x3 = new int[512, 3, 3, 2];
        public static int[,,,] filter_1024_3x3 = new int[1024, 3, 3, 6];
        public static Bitmap bmp = new Bitmap(448,448);
        public static double[] inp = new double[50176];
        //public static int loadstatus = 0;
        public static bool finished = false;
        public static bool started = false;
        public static bool trainorform = false; // checks which one called the abstraction function false = form1 true = setuptrain


        public void main()
        {
            int loadstatus = 0;
            double[,,] tmp;
            tmp = convolution64_7x7(bmp);
            loadstatus++;
            Form1.loading_value = loadstatus;
            Setuptrain.loading_value = loadstatus;
            tmp = maxpool(tmp);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
            tmp = convolution192_3x3(tmp);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
            tmp = maxpool(tmp);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
            tmp = convolution128_1x1(tmp);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
            tmp = convolution256_3x3(tmp);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
            tmp = convolution256_1x1(tmp, 0);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
            tmp = convolution512_3x3(tmp, 0);
            loadstatus++;
            Setuptrain.loading_value = loadstatus;
            Form1.loading_value = loadstatus;
```

```
tmp = maxpool(tmp);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;

tmp = convolution256_1x1(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution256_1x1(tmp, 1);//ERROR sets array to 0
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution256_1x1(tmp, 1);//4 times same filter
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution256_1x1(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_3x3(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_3x3(tmp, 1);//4 times same filter
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_3x3(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_3x3(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_1x1(tmp, 0);//broke
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution1024_3x3(tmp, 0);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = maxpool(tmp);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_1x1(tmp, 1);//2 times
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution512_1x1(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution1024_3x3(tmp, 1);//2 times
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution1024_3x3(tmp, 1);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution1024_3x3(tmp, 2);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;
tmp = convolution1024_3x3_s2(tmp, 3);
loadstatus++;
Setuptrain.loading_value = loadstatus;
Form1.loading_value = loadstatus;

tmp = convolution1024_3x3(tmp, 4);
```

```
    loadstatus++;
    Setuptrain.loading_value = loadstatus;
    Form1.loading_value = loadstatus;
    tmp = convolution1024_3x3(tmp, 5);
    loadstatus++;
    Setuptrain.loading_value = loadstatus;
    Form1.loading_value = loadstatus;

    int temp = 0;//pointer

    for (int i = 0; i < 1024; i++)
    {
        for (int y = 0; y < 7; y++)
        {
            for (int x = 0; x < 7; x++)
            {
                //MessageBox.Show(Math.Max(tmp[x, y, i], 0.1).ToString());
                inp[temp] = Math.Max(tmp[x, y, i], 0.1);//stores a 3d integer array into a 1d integer array
                temp++;//increments the pointer
            }
        }
    }
    finished = true;

}


private static double[,,] convolution64_7x7(Bitmap bp)
{
    int[,,] img = new int[448 + 500, 448 + 500, 64];//448x,448y,rgb
    double[,,] img2 = new double[224 + 500, 224 + 500, 64];//224
    int ttmp = 0;

    int pointerx = 0;
    int pointery = 0;

    for (int f = 0; f < 64; f++)
    {
        for (int y = 0; y < 448; y++)
        {
            for (int x = 0; x < 448; x++)
            {
                Color cl = bp.GetPixel(x, y);
                pointerx++;
                switch (ttmp)
                {
                    case 0:
                        img[pointerx, pointery, f] = cl.R;
                        ttmp++;
                        break;
                    case 1:
                        img[pointerx, pointery, f] = cl.G;
                        ttmp++;
                        break;
                    case 2:
                        img[pointerx, pointery, f] = cl.B;
                        ttmp = 0;
                        break;
                }
            }
            pointerx = 0;
            pointery++;
        }
        pointery = 0;
        pointerx = 0;
    }
    pointerx = 0;
    pointery = 0;
    for (int f = 0; f < 64; f++)
    {
        for (int y = 0; y < 448; y++)
        {
            for (int x = 0; x < 448; x++)
            {
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
```

```
            for (int yy = 0; yy < filter_64_7x7.GetLength(2); yy++)
            {
                for (int xx = 0; xx < filter_64_7x7.GetLength(1); xx++)//multiply image by filters
                {
                    tempor += filter_64_7x7[f, xx, yy] * img[x + xx, y + yy, f];
                }
            }
            x++;

            img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
            pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis

        }

        pointerx = 0; //sets pointer x back to 0
        pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis
        y++;
    }
    pointery = 0;
    pointerx = 0;


}
return img2;
}
private static double[,,] convolution192_3x3(double[,,] img)
{

double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 192];

int pointerx = 0;
int pointery = 0;
int tmp = 0;
for (int f = 0; f < 192; f++)
{
    for (int y = 0; y < img.GetLength(1) - 3; y++)
    {
        for (int x = 0; x < img.GetLength(0) - 3; x++)
        {
            double tempor = 0; //temporary value to store the convolved value which will be passed into the array
            for (int yy = 0; yy < filter_192_3x3.GetLength(2); yy++)
            {
                for (int xx = 0; xx < filter_192_3x3.GetLength(1); xx++)//multiply image by filters
                {
                    tempor += filter_192_3x3[f, xx, yy] * img[x + xx, y + yy, tmp];

                }
            }

            img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
            pointerx++;//increments pointerx in which tells where in array would we store the convolved value x axis
        }
        pointerx = 0; //sets pointer x back to 0
        pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

    }
    pointery = 0;
    pointerx = 0;
    tmp++;
    if (tmp >= img.GetLength(2))
    {
        tmp = 0;
    }
}
return img2;
}
private static double[,,] convolution256_3x3(double[,,] img)
{
double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 256];//224
int pointerx = 0;
int pointery = 0;
int tmp = 0;
for (int f = 0; f < 256; f++)
{
    for (int y = 0; y < img.GetLength(1) - 3; y++)
```

```
            {
                for (int x = 0; x < img.GetLength(0) - 3; x++)
                {
                    double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                    for (int yy = 0; yy < filter_256_3x3.GetLength(2); yy++)
                    {
                        for (int xx = 0; xx < filter_256_3x3.GetLength(1); xx++)//multiply image by filters
                        {
                            tempor += filter_256_3x3[f, xx, yy] * img[x + xx, y + yy, tmp];

                        }
                    }

                    img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                    pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
                }
                pointerx = 0; //sets pointer x back to 0
                pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

            }
            pointery = 0;
            pointerx = 0;
            tmp++;
            if (tmp >= img.GetLength(2))
            {
                tmp = 0;
            }
        }
        return img2;
    }
    private static double[,,] convolution256_1x1(double[,,] img, int e)//the array will be passed on and so will the filterarray to
be used.
    {
        double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 256];//128
        int pointerx = 0;
        int pointery = 0;
        int tmp = 0;
        for (int f = 0; f < 256; f++)
        {
            for (int y = 0; y < img.GetLength(1); y++)
            {
                for (int x = 0; x < img.GetLength(0); x++)
                {
                    double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                    tempor += filter_256_1x1[f, e] * img[x, y, tmp];

                    img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                    pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
                }
                pointerx = 0; //sets pointer x back to 0
                pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

            }
            pointery = 0;
            pointerx = 0;
            tmp++;
            if (tmp >= img.GetLength(2))
            {
                tmp = 0;
            }
        }
        return img2;
    }
    private static double[,,] convolution512_3x3(double[,,] img, int e)
    {
        double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 512];//224
        int pointerx = 0;
        int pointery = 0;
        int tmp = 0;
        for (int f = 0; f < 512; f++)
        {
            for (int y = 0; y < img.GetLength(1) - 3; y++)
            {
                for (int x = 0; x < img.GetLength(0) - 3; x++)
                {
```

```
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                for (int yy = 0; yy < filter_512_3x3.GetLength(2); yy++)
                {
                    for (int xx = 0; xx < filter_512_3x3.GetLength(1); xx++)//multiply image by filters
                    {
                        tempor += filter_512_3x3[f, xx, yy, e] * img[x + xx, y + yy, tmp];

                    }
                }

                img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
            }
            pointerx = 0; //sets pointer x back to 0
            pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

        }
        pointery = 0;
        pointerx = 0;
        tmp++;
        if (tmp >= img.GetLength(2))
        {
            tmp = 0;
        }
    }
    return img2;
}
private static double[,,] convolution512_1x1(double[,,] img, int e)
{
    double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 512];//128
    int pointerx = 0;
    int pointery = 0;
    int tmp = 0;
    for (int f = 0; f < 512; f++)
    {
        for (int y = 0; y < img.GetLength(1); y++)
        {
            for (int x = 0; x < img.GetLength(0); x++)
            {
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                tempor += filter_512_1x1[f, e] * img[x, y, tmp];

                img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis

            }
            pointerx = 0; //sets pointer x back to 0
            pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

        }
        pointery = 0;
        pointerx = 0;
        tmp++;
        if (tmp >= img.GetLength(2))
        {
            tmp = 0;
        }
    }
    return img2;
}
private static double[,,] convolution1024_3x3(double[,,] img, int e)
{
    double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 1024];//224
    int pointerx = 0;
    int pointery = 0;
    int tmp = 0;
    for (int f = 0; f < 1024; f++)
    {
        for (int y = 0; y < img.GetLength(1) - 3; y++)
        {
            for (int x = 0; x < img.GetLength(0) - 3; x++)
            {
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                for (int yy = 0; yy < filter_1024_3x3.GetLength(2); yy++)
                {
```

```
                for (int xx = 0; xx < filter_1024_3x3.GetLength(1); xx++)//multiply image by filters
                {
                    tempor += filter_1024_3x3[f, xx, yy, e] * img[x + xx, y + yy, tmp];
                }
            }

            img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
            pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
        }
        pointerx = 0; //sets pointer x back to 0
        pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

    }
    pointery = 0;
    pointerx = 0;
    tmp++;
    if (tmp >= img.GetLength(2))
    {
        tmp = 0;
    }
  }
  return img2;
}
private static double[,,] convolution1024_3x3_s2(double[,,] img, int e)
{
  double[,,] img2 = new double[img.GetLength(0) / 2, img.GetLength(1) / 2, 1024];//224
  int pointerx = 0;
  int pointery = 0;
  int tmp = 0;
  for (int f = 0; f < 1024; f++)
  {
    for (int y = 0; y < img.GetLength(1) - 3; y++)
    {
      for (int x = 0; x < img.GetLength(0) - 3; x++)
      {
        double tempor = 0; //temporary value to store the convolved value which will be passed into the array
        for (int yy = 0; yy < filter_1024_3x3.GetLength(2); yy++)
        {
          for (int xx = 0; xx < filter_1024_3x3.GetLength(1); xx++)//multiply image by filters
          {
            tempor += filter_1024_3x3[f, xx, yy, e] * img[x + xx, y + yy, tmp];
          }
        }
        x++;

        img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
        pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
      }
      y++;
      pointerx = 0; //sets pointer x back to 0
      pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

    }
    pointery = 0;
    pointerx = 0;
    tmp++;
    if (tmp >= img.GetLength(2))
    {
      tmp = 0;
    }
  }
  return img2;
}
private static double[,,] convolution128_1x1(double[,,] img)
{
  double[,,] img2 = new double[img.GetLength(0), img.GetLength(1), 128];//128
  int pointerx = 0;
  int pointery = 0;
  int tmp = 0;
  for (int f = 0; f < 128; f++)
  {
    for (int y = 0; y < img.GetLength(1); y++)
    {
      for (int x = 0; x < img.GetLength(0); x++)
      {
```

```
                double tempor = 0; //temporary value to store the convolved value which will be passed into the array
                tempor += filter_128_1x1[f] * img[x, y, tmp];

                img2[pointerx, pointery, f] = Math.Max(tempor, 0.1);//sets the final array with the convolved values
                pointerx++;//incremeants pointerx in which tells where in array would we store the convolved value x axis
            }
            pointerx = 0; //sets pointer x back to 0
            pointery++; //incremeants pointery in which tells where in array would we store the convolved value y axis

        }
        pointery = 0;
        pointerx = 0;
        tmp++;
        if (tmp >= img.GetLength(2))
        {
            tmp = 0;
        }
    }
    return img2;
}


private static double[,,] maxpool(double[,,] bmp)
{

    double[,,] newbmp = new double[bmp.GetLength(0) / 2, bmp.GetLength(1) / 2, bmp.GetLength(2)]; //x,y,filter
    int pointy = 0; //y pointer
    int pointx = 0; //x pointer
    for (int i = 0; i < newbmp.GetLength(2); i++)//applies maxpool on every filter layer
    {
        pointy = 0;
        for (int y = 0; y < newbmp.GetLength(1); y++)//y pixels
        {

            for (int x = 0; x < newbmp.GetLength(0); x++)//x pixels
            {
                double tmp1 = Math.Max(Math.Max(bmp[pointx, pointy, i], bmp[pointx + 1, pointy, i]), Math.Max(bmp[pointx,
pointy + 1, i], bmp[pointx + 1, pointy + 1, i]));//gets the maximum value of the 2x2 grid
                newbmp[x, y, i] = tmp1;//sets the new array with the maxpooled pixels
                pointx = pointx + 2;

            }
            pointy = pointy + 2;
            pointx = 0;

        }

        pointy = 0;
        pointx = 0;
    }

    return newbmp;
}




  }
}
```

Neural network trainer

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;
using System.IO;
using System.Threading;
namespace ProjectImagerecognition
{
    public partial class Neuralnetworktrainer : Form
    {
        public Neuralnetworktrainer()
        {
            InitializeComponent();
        }
        double cost = 0;
        int clicks = 0;
        string location = "";
        double[] w1 = new double[205520896];
        double[] w2 = new double[6021120];
        double[] h1 = new double[4096];
        double[] inp = new double[50176];
        double[] outp = new double[1470];
        double[,,] expected = new double[7, 7, 30];
        int loading_value = 0;
        int loading_max = 0;
        string loadinglabeltxt = "";
        bool loading = false;
        bool loaded = false;
        bool saving = false;
        bool stop = true;
        double chartnum = 0;
        double chartpoint = 0;
        double chartnum_prev = 0;
        double chartpoint_prev = 0;
        Bitmap bp = new Bitmap(448, 448);

        Thread tr;


        private void Neuralnetworktrainer_Load(object sender, EventArgs e)
        {

        }

        private void buttonAuto_detect_Click(object sender, EventArgs e)
        {
            if(loaded == true)
            {
                buttonAuto_detect.Visible = false;
                button_stop.Visible = true;
                stop = false;
                tr = new Thread(neuralnetbackprop);
                tr.IsBackground = true;
                tr.Start();

            }
            else
            {
                MessageBox.Show("Please load the weights first");
            }
            chart_cost.Series["Cost"].Points.AddXY(clicks,cost);
            cost--;
        }

        private void neuralnetbackprop()
        {
            while (true)
            {
                if (stop == false)//while automode is still enabled keep looping
                {
                    int f = 0;
                    while (File.Exists(location + "\\training\\" + f.ToString() + ".txt"))
                    {
                        f++;
                    }
                    Random rn = new Random();
                    f = rn.Next(0, f);
                    while (!File.Exists(location + "\\training\\" + f.ToString() + ".txt"))
```

```
{
    f = rn.Next(0, f);// checks if there are no empty spots.
}
//MessageBox.Show(f.ToString());
var readmisc = File.ReadAllLines(location + "\\training\\" + f.ToString() + ".txt");
bp = new Bitmap(location + "\\training\\" + f.ToString() + ".png");
pictureBox_outp.Image = bp;
int tmp = 0;
for (int i = 1470; i < readmisc.Count(); i++)
{
    inp[tmp] = Convert.ToDouble(readmisc[i]);
    tmp++;
    //listBox1.Items.Add(readmisc[i]);
}
tmp = 0;
for (int i = 0; i < 30; i++)
{
    for (int y = 0; y < 7; y++)
    {
        for (int x = 0; x < 7; x++)
        {
            expected[x, y, i] = Convert.ToDouble(readmisc[tmp]);
            tmp++;
        }
    }
}
h1 = new double[4096];
outp = new double[1470];
//<Forward propagation so we can get the output of neural net and workout the cost(error)>
double tmpp = 0;
for(int ff = 0; ff< inp.Count(); ff++)
{
    tmpp += inp[ff];
}
for(int ff = 0; ff< inp.Count(); ff++)
{
    inp[ff] = inp[ff] / tmpp;
}
tmp = 0;
tmpp = 0;
for (int i = 0; i < 4096; i++)
{
    for (int x = 0; x < inp.Count(); x++)
    {

        h1[i] += inp[x] * w1[tmp];
        tmpp += h1[i];
        tmp++;
    }
}
for(int i = 0; i< 4096; i++)
{
    h1[i] = h1[i] / tmpp;
}

tmp = 0;
for (int i = 0; i < 1470; i++)
{
    for (int x = 0; x < h1.Count(); x++)
    {
        outp[i] += h1[x] * w2[tmp];
        tmp++;
    }
    }
double[,,] outp_3d = new double[7, 7, 30];
tmp = 0;
for (int i = 0; i < 30; i++)
{
    for (int y = 0; y < 7; y++)
    {
        for (int x = 0; x < 7; x++)
        {
            outp_3d[x, y, i] = outp[tmp];
            tmp++;
        }
```

```
    }
}
//<   >
double[,,] cost = new double[7, 7, 30];
double cost_average = 0;
for (int y = 0; y < 7; y++)
{
    for (int x = 0; x < 7; x++)
    {
        cost[x,y,1] = expected[x, y, 1] - outp_3d[x, y, 1];//anchorbox 1
        cost[x,y,16] = expected[x, y, 16] - outp_3d[x, y, 16];//anchorbox 2

        cost[x,y,2] = expected[x, y, 2] - outp_3d[x, y, 2];//anchorbox1
        cost[x,y,17] = expected[x, y, 17] - outp_3d[x, y, 17];//anchorbox2


        cost[x,y,4] = expected[x, y, 4] - outp_3d[x, y, 4];
        cost[x,y,19] = expected[x, y, 19] - outp_3d[x, y, 19];

        cost[x,y,3] = expected[x, y, 3] - outp_3d[x, y, 3];
        cost[x,y,18] = expected[x, y, 18] - outp_3d[x, y, 18];

        for (int i = 5; i < 15; i++)
        {
            cost[x,y,i] = expected[x, y, i] - outp_3d[x, y, i];
            cost[x,y,i + 15] = expected[x, y, i + 15] - outp_3d[x, y, i + 15];
        }

        cost[x,y,0] = expected[x, y, 0] - outp_3d[x, y, 0];
        cost[x,y,15] = expected[x, y, 15] - outp_3d[x, y, 15];

        for(int i = 0; i< 30; i++)
        {
            cost_average += cost[x, y, i];
        }

    }
}


cost_average =  cost_average / 30;

//MessageBox.Show(cost_average.ToString());
chartnum = Math.Round(cost_average,3);
chartpoint++;
tmp = 0;


//Backpropagation


double[,,] delta = new double[7, 7, 30];
int temp = 0;
for(int i = 0; i< 30; i++)
{
    for(int y = 0; y < 7; y++)
    {
        for(int x = 0; x < 7; x++)
        {
            delta[x, y, i] = cost[x, y, i] * reluprime(outp[temp]);
            temp++;
        }
    }
}

double[] h1error = new double[4096]; //h1
temp = 0;
for(int i = 0; i< 4096; i++)
{
    for(int fff = 0; fff< 30; fff++)
    {
        for(int y = 0; y< 7; y++)
        {
            for(int x = 0; x< 7; x++)
            {
```

```
                    h1error[i] += delta[x,y,fff] * w2[temp];
                    temp++;
                }
            }
        }
    }

    double[] h1delta = new double[4096];
    for(int i = 0; i< 4096; i++)
    {
        h1delta[i] = h1error[i] * reluprime(h1[i]);
    }
    temp = 0;


    for(int fff = 0; fff< 30; fff++)
    {
        for(int y = 0; y< 7; y++)
        {
            for(int x = 0; x< 7; x++)
            {
                for(int i = 0; i< 4096; i++)
                {
                    w2[temp] += h1[i] * delta[x, y, fff]* 100000000000;
                    temp++;
                }
            }
        }
    }


    temp = 0;
    for(int i = 0; i < 4096; i++)
    {
        for(int fff = 0; fff < inp.Count(); fff++)
        {
            w1[temp] += (inp[fff] * h1delta[i])*1000000000;
            temp++;
        }
    }
    temp = 0;



    }
    else
    {
        break;//stop looping since automode is off
    }
  }
}
double relu(double x)
{

  double tmp = Math.Max(0.01, x);
  return tmp;
}
double reluprime(double x)
{
  double tmp = 0;
  if (x > 0)
  {
    tmp = 1;
  }
  else
  {
    tmp = 0;
  }
  return tmp;
}
private void load()
{
  var readw1 = File.ReadLines(location + "\\w1.txt");//specifies the location of the file
  var readw2 = File.ReadLines(location + "\\w2.txt");//weights
  loading = true;
```

```
    loading_max = 211542016; //size of w1 + w2
    loading_value = 0;
    int teemp = 0;
    foreach (var line in readw1)//loops 205520896 times
    {

        w1[teemp] = Convert.ToDouble(line);//stores each line in w1.txt to a 1d double array
        teemp++;//incremeants the temporary value
        loading_value++;//increase the loading progressbar value
        loadinglabeltxt = "loading Weights(w1)";
    }
    teemp = 0;
    foreach (var line in readw2)//loops 205520896 times
    {

        w2[teemp] = Convert.ToDouble(line);//stores each line in w1.txt to a 1d double array
        teemp++;//incremeants the temporary value
        loading_value++;//increase the loading progressbar value
        loadinglabeltxt = "loading Weights(w2)";
    }


    for(int i = 0; i < 1470; i++)
    {

    }
    loadinglabeltxt = "";
    loading_max = 0;
    loading_value = 0;
    loading = false;//dissables loading and sets all values back to default
    loaded = true;
    tr.Abort();
}
private void save()
{
    loading = true;
    loadinglabeltxt = "Saving Weights(w2)";
    loading_max = 211542016;
    loading_value = 0;
    StreamWriter sr = new StreamWriter(location+"\\w1.txt");//w1
    StreamWriter sr2 = new StreamWriter(location + "\\w2.txt");//w1
    Random rn = new Random();

    for (int f = 0; f < w1.Length; f++)
    {
        //double n = w1[f] * Math.Sqrt(2.0 / 1.0);
        sr.WriteLine(w1[f].ToString());//once there aren't any 0's the number gets written into the filter.
        loading_value++;
    }

    for (int f = 0; f < w2.Length; f++)
    {
        //double n = w2[f] * Math.Sqrt(2.0 / 1.0);
        sr2.WriteLine(w2[f].ToString());//once there aren't any 0's the number gets written into the filter.
        loading_value++;
    }
    sr.Close();//saves w1 file
    sr2.Close();//saves w2 file
    loading = false;
    loadinglabeltxt = "";
    loading_max = 0;
    loading_value = 0;
    saving = false;
    tr.Abort();
}
private void button_Open_Click(object sender, EventArgs e)
{
    if(saving == false)
    {
        MessageBox.Show("select the data folder");


        FolderBrowserDialog fl = new FolderBrowserDialog();
        DialogResult rl = fl.ShowDialog();
        if (rl == DialogResult.OK && fl.SelectedPath != "")//checks if file was opened successfully
```

```
        {
            location = fl.SelectedPath.ToString();//location of the data folder

            tr = new Thread(load);//creates a thread in which would load up the weights without freezing the whole program
            tr.IsBackground = true;
            tr.Start();
            //var readw1 = File.ReadAllLines(location + "\\w1.txt");
            //string[] sr = new string[21];
            /*for (int i = 1; i < readw1.Count(); i++)
            {
                //listBox1.Items.Add(readmisc[i]);
            }*/
        }
        else
        {
            MessageBox.Show("Invalid file path");
        }
    }
    else
    {
        MessageBox.Show("Saving in progress, please wait...");
    }

}

private void timer1_Tick(object sender, EventArgs e)
{
    if (loading == true)
    {
        progressBar1_loading.Visible = true;
        label1_loading.Visible = true;
        pictureBox_loadingback.Visible = true;
        pictureBox_loadinggif.Visible = true;
        label1_loading.Text = loadinglabeltxt;
        progressBar1_loading.Maximum = loading_max;
        progressBar1_loading.Value = loading_value;


    }
    else if (loading == false)
    {
        progressBar1_loading.Visible = false;
        label1_loading.Visible = false;
        pictureBox_loadinggif.Visible = false;
        pictureBox_loadingback.Visible = false;
    }
    pictureBox_outp.Image = bp;
    if(chartnum_prev != chartnum || chartpoint_prev != chartpoint)
    {
        chart_cost.Series["Cost"].Points.AddXY(chartpoint, chartnum);
        chartnum_prev = chartnum;
        chartpoint_prev = chartpoint;
    }

}

private void button_Save_Click(object sender, EventArgs e)
{
    if(loaded == true)
    {
        saving = true;
        tr = new Thread(save);
        tr.IsBackground = true;
        tr.Start();
    }
    else
    {
        MessageBox.Show("Please load up the weights first");
    }
}

private void button_stop_Click(object sender, EventArgs e)
{
    button_stop.Visible = false;
    buttonAuto_detect.Visible = true;
```

```
            stop = true;

        }

        private void pictureBox_outp_Click(object sender, EventArgs e)
        {

        }
    }
}
```

Dataset maker

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;
using System.Threading;
namespace ProjectImagerecognition
{
    public partial class Setuptrain : Form
    {
        public Setuptrain()
        {
            InitializeComponent();
        }
        bool selected = false;
        int x = 0;
        int y = 0;
        int xx = 0;
        int yy = 0;
        int midx = 0;
        int midy = 0;
        Bitmap bmp;
        Bitmap bitm;
        Bitmap default_bmp;
        int previousx = 0;
        int previousy = 0;
        string location = "";
        int listboxindexnum = 0;
        double[,,] expected = new double[7, 7, 30];
        int[,,] filter_64_7x7 = new int[64, 7, 7];
        int[,,] filter_192_3x3 = new int[192, 3, 3];
        int[] filter_128_1x1 = new int[128];
        int[,,] filter_256_3x3 = new int[256, 3, 3];
        int[,] filter_256_1x1 = new int[256, 2];//changed so more different filters
        int[,] filter_512_1x1 = new int[512, 2];
        int[,,,] filter_512_3x3 = new int[512, 3, 3, 2];
        int[,,,] filter_1024_3x3 = new int[1024, 3, 3, 6];
        public static int loading_value = 0;
        static int loading_max = 0;
        static bool loading = false;
        static bool listboxloaded = false;
        static string loadinglabeltxt = "Abstracting image";

        private void buttonImport_detect_Click(object sender, EventArgs e)
```

```
        {
            Bitmap bp;//creates bitmap variable
            OpenFileDialog fl = new OpenFileDialog();
            DialogResult rl = fl.ShowDialog();//opens up a dialog that allows the user to choose the file path
            if (rl == DialogResult.OK)//checks if file path is valid
            {
                try//in case if something goes wrong, the program won't crash but display that something is wrong with filepath
                {
                    bp = new Bitmap(fl.FileName);//opens the image and stores it into bitmap variable
                    default_bmp = new Bitmap(bp, new Size(448, 448));//resizes the image to 448x448 size
                    bmp = default_bmp;
                    pictureBoxImg.Image = default_bmp;//displays the image
                }
                catch (Exception)
                {
                    MessageBox.Show("File cannot be loaded!");
                }
            }
            else
            {
                MessageBox.Show("Could not open the image!");
            }
        }

        private void buttonNextObject_Click(object sender, EventArgs e)
        {

        }

        private void buttonClear_Click(object sender, EventArgs e)
        {
            pictureBoxImg.Image = default_bmp;//removes drawn bounding boxes by setting the image to the default image that
was imported
            for(int y = 0; y<7; y++)//since expected is 7x7x30 in output, I need to do a for loop 7x7 times
            {
                for(int x = 0; x<7; x++)
                {
                        if (expected[x, y, listboxindexnum + 20] == 1)//checks if anchorbox 2 is being used
                        {
                            expected[x, y, listboxindexnum + 20] = 0;//sets the class to 0
                            expected[x, y, 19] = 0;//sets Bh to 0
                            expected[x, y, 18] = 0;//sets Bw to 0
                            expected[x, y, 17] = 0;//sets By to 0
                            expected[x, y, 16] = 0;//sets Bx to 0
                            expected[x, y, 15] = 0;//sets Pc to 0
                        }
                        if (expected[x, y, listboxindexnum + 5] == 1)//checks if anchorbox 1 is being used
                        {
                            expected[x, y, listboxindexnum + 5] = 0;//sets the class to 0
                            expected[x, y, 4] = 0;//sets Bh to 0
                            expected[x, y, 3] = 0;//sets Bw to 0
                            expected[x, y, 2] = 0;//sets By to 0
                            expected[x, y, 1] = 0;//sets Bx to 0
                            expected[x, y, 0] = 0;//sets Pc to 0
                        }

                }
            }

        }

        private void buttonSave_Click(object sender, EventArgs e)
        {


            int tmp = 0;
            while(File.Exists(location+"\\training\\" + tmp.ToString() + ".txt"))//stores the file once an available number for the file
name was found
            {
                tmp++;
            }
            StreamWriter sr = new StreamWriter(location + "\\training\\" + tmp.ToString() + ".txt");//stores it in a text file
            for(int i = 0; i<30; i++)
            {
                for(int y = 0; y< 7; y++)
```

```
        {
            for(int x = 0; x< 7; x++)
            {
                sr.WriteLine(expected[x,y,i].ToString());
            }
        }
    }
    for(int i = 0; i< 50176; i++)
    {
        sr.WriteLine(Abstractor.inp[i].ToString());
    }
    default_bmp.Save(location + "\\training\\" + tmp.ToString() + ".png",System.Drawing.Imaging.ImageFormat.Png);
    sr.Close();

}

private void pictureBoxImg_MouseClick(object sender, MouseEventArgs e)
{
    if (selected == false)//when the user clicks first time, the x and y of the location of the first selection click will be recorded
    {
        bmp = new Bitmap(pictureBoxImg.Image);//sets bmp to the image
        selected = true;//sets that the 1st click was used
        x = e.X;//obtains the x co-ordinates of the click
        y = e.Y;//obtains the y co-ordinates of the click
        pictureBoxImg.Image = bmp;

    }
    else//2nd x and y click recorded in which will later on draw the selection and record the location of the object.
    {
        pictureBoxImg.Image = bmp;
        xx = e.X;//get the co-ordinates of the 2nd click
        yy = e.Y;

        int pointer_xx = 0;
        int pointer_yy = 0;
        int pointer_x = 0;
        int pointer_y = 0;
        midx = (xx + x) / 2;//finds midpoint of x
        midy = (yy + y) / 2;//finds midpoint of y
        for(int i = x; i< xx; i++)//draws the selected box
        {
            bmp.SetPixel(i, y, Color.Red);//draws the top
        }
        for (int i = y; i < yy; i++)
        {
            bmp.SetPixel(x, i, Color.Red);//draws the left side
        }
        for (int i = x; i < xx; i++)
        {
            bmp.SetPixel(i, yy, Color.Red);//draws the bottom
        }
        for (int i = y; i < yy; i++)
        {
            bmp.SetPixel(xx, i, Color.Red);//draws the right side
        }


        pictureBoxImg.Image = bmp;
        for (int i = 0; i < x; i += 64)
        {
            pointer_x = i;
        }
        for (int i = 0; i < y; i += 64)
        {
            pointer_y = i;
        }
        for (int i = 0; i< xx; i += 64)
        {
            pointer_xx = i;
        }
        for(int i = 0; i< yy; i+= 64)
        {
            pointer_yy = i;
        }
```

```
            double num1 = 64;//I did this to fix a simple bug since c# thought that 64 is an integer it automatically rounded up the
number
            for(int yd = pointer_y; yd<pointer_yy + 64; yd+= 64)
            {
               for (int xd = pointer_x; xd < pointer_xx + 64; xd+= 64)
               {
                  //MessageBox.Show((xd/64).ToString());
                  int ex = Math.Max(xd- 32,0);
                  int ey = yd;
                  int exx = Math.Min(xd + 96,447);//anchor box 1
                  int eyy = Math.Min(yd + 64,447);

                  int rx = xd;//anchor box 2
                  int ry = Math.Max(yd - 32,0);
                  int rxx = Math.Min(xd + 64,447);
                  int ryy = Math.Min(yd + 96,447);

                  //double anchorbox1iou = IoUgenerator(ex, ey, exx, eyy, x, y, xx, yy);
                 // double anchorbox2iou = IoUgenerator(rx, ry, rxx, ryy, x, y, xx, yy);

                  if(Math.Abs(x-xx) > Math.Abs(y-yy))
                  {


                     expected[xd / 64, yd / 64, listboxindexnum + 5] = 1;
                     bool boxerror = false;
                     for (int i = 5; i < 14; i++)
                     {
                        if (expected[xd / 64, yd / 64, i] == 1 && i != listboxindexnum + 5)
                        {
                           //MessageBox.Show("Can't fit another object in cell");
                           expected[xd / 64, yd / 64, listboxindexnum + 5] = 0;
                           boxerror = true;
                        }
                     }
                     if(boxerror == false)
                     {
                        expected[xd / 64, yd / 64, 0] = 1;
                        expected[xd / 64, yd / 64, 1] = Math.Min((Math.Max(((midx - xd) / num1), 0)), 1); //works out the bx
                        expected[xd / 64, yd / 64, 2] = Math.Min((Math.Max(((midy - yd) / num1), 0)), 1);//works out the by
                        expected[xd / 64, yd / 64, 3] = Math.Abs(x - xx) / 448.0;//works out the bw
                        expected[xd / 64, yd / 64, 4] = Math.Abs(yy - y) / 448.0;//works out the bh


                        //MessageBox.Show(expected[xd/64, yd/64, 3].ToString());
                        /*for (int i = ex; i < exx; i++)
                        {
                           bmp.SetPixel(i, ey, Color.Blue);
                        }
                        for (int i = ey; i < eyy; i++)
                        {
                           bmp.SetPixel(ex, i, Color.Blue);
                        }
                        for (int i = ex; i < exx; i++)
                        {
                           bmp.SetPixel(i, eyy, Color.Blue);
                        }//DELETE DIS
                        for (int i = ey; i < eyy; i++)
                        {
                           bmp.SetPixel(exx, i, Color.Blue);
                        }*/
                     }
                     else
                     {
                        MessageBox.Show("Can't fit another object in cell");
                     }

                  }
                  else if(Math.Abs(x - xx) < Math.Abs(y - yy))
                  {
                     expected[xd / 64, yd / 64, listboxindexnum + 20] = 1;
                     bool boxerror = false;
                     for (int i = 20; i < 29; i++)
                     {
                        if (expected[xd / 64, yd / 64, i] == 1 && i != listboxindexnum + 20)
```

```
            {
                //MessageBox.Show("Can't fit another object in cell");
                expected[xd / 64, yd / 64, listboxindexnum + 20] = 0;
                boxerror = true;
            }
        }
        if(boxerror == false)
        {
            expected[xd / 64, yd / 64, 15] = 1;
            expected[xd / 64, yd / 64, 16] = Math.Min((Math.Max(((midx - xd) / num1), 0)), 1); //works out the bx
            expected[xd / 64, yd / 64, 17] = Math.Min((Math.Max(((midy - yd) / num1), 0)), 1);//works out the by
            expected[xd / 64, yd / 64, 18] = Math.Abs(x - xx) / 448.0;//works out the bw
            expected[xd / 64, yd / 64, 19] = Math.Abs(yy - y) / 448.0;//works out the bh

            /*for (int i = rx; i < rxx; i++)
            {
                bmp.SetPixel(i, ry, Color.Blue);
            }
            for (int i = ry; i < ryy; i++)
            {
                bmp.SetPixel(rx, i, Color.Blue);
            }
            for (int i = rx; i < rxx; i++)
            {
                bmp.SetPixel(i, ryy, Color.Blue);
            }//DELETE DIS
            for (int i = ry; i < ryy; i++)
            {
                bmp.SetPixel(rxx, i, Color.Blue);
            }*/
        }
        else
        {
            MessageBox.Show("Can't fit another object in cell");
        }

    }
    else
    {
        expected[xd / 64, yd / 64, listboxindexnum + 20] = 1;
        bool boxerror = false;
        for (int i = 20; i < 29; i++)
        {
            if (expected[xd / 64, yd / 64, i] == 1 && i != listboxindexnum + 20)
            {
                //MessageBox.Show("Can't fit another object in cell");
                expected[xd / 64, yd / 64, listboxindexnum + 20] = 0;
                boxerror = true;
            }
        }

        if(boxerror == false)
        {
            expected[xd / 64, yd / 64, 15] = 1;
            expected[xd / 64, yd / 64, 16] = Math.Min((Math.Max(((midx - xd) / num1), 0)), 1); //works out the bx
            expected[xd / 64, yd / 64, 17] = Math.Min((Math.Max(((midy - yd) / num1), 0)), 1);//works out the by
            expected[xd / 64, yd / 64, 18] = Math.Abs(x - xx) / 448.0;//works out the bw
            expected[xd / 64, yd / 64, 19] = Math.Abs(yy - y) / 448.0;//works out the bh

            /*for (int i = rx; i < rxx; i++)
            {
                bmp.SetPixel(i, ry, Color.Blue);
            }
            for (int i = ry; i < ryy; i++)
            {
                bmp.SetPixel(rx, i, Color.Blue);
            }
            for (int i = rx; i < rxx; i++)
            {
                bmp.SetPixel(i, ryy, Color.Blue);
            }//DELETE DIS
            for (int i = ry; i < ryy; i++)
            {
                bmp.SetPixel(rxx, i, Color.Blue);
            }*/
```

```
                    }
                    else
                    {
                        MessageBox.Show("Can't fit another object in cell");
                    }


                }
                pictureBoxImg.Image = bmp;

                //MessageBox.Show(x.ToString() + "      " + y.ToString());//for debuging
                //expected[ex, ey, 0] =  (midx - x) / num1; //works out the bx
               // expected[ex, ey, 1] = (midy - y) / num1;//works out the by
                //MessageBox.Show(expected[ex, ey, 0].ToString() + "x " + expected[ex,ey,1].ToString() + "y ");//for debuging


            }
        }
        selected = false;
    }
}



    private static double IoUgenerator(double x, double y, double xx, double yy, double x_2, double y_2, double xx_2, double
yy_2)
    {
        double area1 = Math.Abs(x - xx) * Math.Abs(y - yy);//all explained in the documentation
        double area2 = Math.Abs(x_2 - xx_2) * Math.Abs(y_2 - yy_2);

        double areaofoverlap = (Math.Min(xx, xx_2) - Math.Max(x, x_2)) * (Math.Min(yy, yy_2) - Math.Max(y, y_2));
        double areaofunion = area1 + area2 - areaofoverlap;

        return areaofoverlap/areaofunion;
    }

    private void pictureBoxImg_MouseMove(object sender, MouseEventArgs e)
    {
        if(selected == true)//this is just to make it look nicer.
        {
            int xxx = e.X;
            int yyy = e.Y;
            if (xxx != previousx && yyy != previousy)
            {
                bitm = new Bitmap(bmp);//creates a copy of the image
                previousx = x;//to reduce lag, I decided to implement this so it would check if the location of the cursor has
changed or not
                previousy = y;
                for (int i = x; i < xxx; i++)//draws box
                {
                    bitm.SetPixel(i, y, Color.Red);
                }
                for (int i = y; i < yyy; i++)
                {
                    bitm.SetPixel(x, i, Color.Red);
                }
                for (int i = x; i < xxx; i++)
                {
                    bitm.SetPixel(i, yyy, Color.Red);
                }
                for (int i = y; i < yyy; i++)
                {
                    bitm.SetPixel(xxx, i, Color.Red);
                }
                pictureBoxImg.Image = bmp;//clears the previous image back to default
                pictureBoxImg.Image = bitm;//sets the image with the drawn box
            }
        }
    }

    private void button_Open_Click(object sender, EventArgs e)
    {
        MessageBox.Show("please select the data folder");
        FolderBrowserDialog fl = new FolderBrowserDialog();
        DialogResult rl = fl.ShowDialog();
```

```
            if (rl == DialogResult.OK && fl.SelectedPath != "")
            {
               location = fl.SelectedPath.ToString();
               var readmisc = File.ReadAllLines(location + "\\misc.txt");
               string[] sr = new string[21];
               for(int i = 1; i < readmisc.Count(); i++)
               {
                   listBox1.Items.Add(readmisc[i]);
               }
                listBox1.SelectedIndex = 0;
                load();
            }
            else
            {
               MessageBox.Show("Invalid file path");
            }



        }
        private void load()
        {
            var readf64_7x7 = File.ReadAllLines(location + "\\filters\\f64_7x7.txt");//filters
            var readf128_1x1 = File.ReadAllLines(location + "\\filters\\f128_1x1.txt");
            var readf192_3x3 = File.ReadAllLines(location + "\\filters\\f192_3x3.txt");
            var readf256_1x1 = File.ReadAllLines(location + "\\filters\\f256_1x1.txt");
            var readf256_3x3 = File.ReadAllLines(location + "\\filters\\f256_3x3.txt");
            var readf512_1x1 = File.ReadAllLines(location + "\\filters\\f512_1x1.txt");
            var readf512_3x3 = File.ReadAllLines(location + "\\filters\\f512_3x3.txt");
            var readf1024_3x3 = File.ReadAllLines(location + "\\filters\\f1024_3x3.txt");


            int tem = 0;
            int tem2 = 0;
            int tem3 = 0;
            int tem3_1 = 0;
            int tem_1 = 0;
            int tem4 = 0;
            int tem4_1 = 0;
            for (int i = 0; i < 6144; i++)//1024x3x3x6 = 6144 which is the ammount of loops this will go through
            {
                loading_value++;


                if (i < 6144)//makes sure that the program doesn't execute the code within the if statement more than 1024 times
                {
                    if (tem >= 1024)
                    {
                        tem = 0;
                        tem_1++;
                    }
                    int tmpp = 0;//temporary value
                    string[] word = readf1024_3x3[i].Split(',');//using similar code as the one with weights but since, the data is only
fetched -
                    //line by line, this function will have to split that 1 line into many seperate numbers
                    for (int y = 0; y < 3; y++)
                    {
                        for (int x = 0; x < 3; x++)
                        {


                            Abstractor.filter_1024_3x3[tem, y, x, tem_1] = Convert.ToInt32(word[tmpp]);//stores the data into a 3d integer
array
                            tmpp++;
                        }
                    }
                    tem++;
                }


                if (i < 64)
                {
                    int tmpp = 0;
                    string[] word = readf64_7x7[i].Split(',');
```

```
            for (int y = 0; y < 7; y++)
            {
                for (int x = 0; x < 7; x++)
                {


                    Abstractor.filter_64_7x7[i, y, x] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }
            }
        }
        if (i < 128)
        {

            Abstractor.filter_128_1x1[i] = Convert.ToInt32(readf128_1x1[i]);//since filter_128_1x1 is only a 1x1 then there's no
need to make a 3d int array so this -
            //segment of code will be used just like it was used with w1 and w2
        }
        if (i < 192)
        {
            int tmpp = 0;
            string[] word = readf192_3x3[i].Split(',');
            for (int y = 0; y < 3; y++)
            {
                for (int x = 0; x < 3; x++)
                {
                    Abstractor.filter_192_3x3[i, y, x] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }

            }
        }
        if (i < 256)
        {
            int tmpp = 0;
            string[] word = readf256_3x3[i].Split(',');
            for (int y = 0; y < 3; y++)
            {
                for (int x = 0; x < 3; x++)
                {
                    Abstractor.filter_256_3x3[i, y, x] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }

            }
            tem2++;
        }
        if (i < 512)
        {
            if (tem4 >= 256)
            {
                tem4 = 0;
                tem4_1++;
            }
            Abstractor.filter_256_1x1[tem4, tem4_1] = Convert.ToInt32(readf256_1x1[i]);
            tem4++;
        }


        if (i < 1024)//doubled because of filter problem
        {
            if (tem3 >= 512)
            {
                tem3 = 0;//makes sure that tem3 doesn't go over 512 which is the maximum
                tem3_1++;//increments the filter layer pivot
            }
            Abstractor.filter_512_1x1[tem3, tem3_1] = Convert.ToInt32(readf512_1x1[i]);//readf512_1x1 size is 1024 but it will
read the first half if -
            //tem3_1 = 0 and would read the last half if tem3_1 = 1
            int tmpp = 0;
            string[] word = readf512_3x3[i].Split(',');//splits the filter values when it finds a ,
            for (int y = 0; y < 3; y++)
            {
                for (int x = 0; x < 3; x++)
                {
```

```
                    Abstractor.filter_512_3x3[tem3, y, x, tem3_1] = Convert.ToInt32(word[tmpp]);
                    tmpp++;
                }

            }
            tem3++;
        }
    }
}

private void listBox1_MouseClick(object sender, MouseEventArgs e)
{
    listboxindexnum = listBox1.SelectedIndex;
    bmp = new Bitmap(default_bmp);
    pictureBoxImg.Image = default_bmp;
    for (int y = 0; y < 7; y++)
    {
        for (int x = 0; x < 7; x++)
        {

            if (expected[x, y, 15] == 1.0 && expected[x, y, 16] != 0.0 && expected[x, y, 16] != 1.0 && expected[x, y, 17] != 0.0
&& expected[x, y, 17] != 1.0 && expected[x, y, listboxindexnum + 20] == 1)
            {
                double bx = (expected[x, y, 16] * 64.0) + (x * 64.0);//works out the centre location of the object x
                double by = (expected[x, y, 17] * 64.0) + (y * 64.0);//works out the centre location of the object y
                double bw = (expected[x, y, 18] * 448.0);//works out the width of the object
                double bh = (expected[x, y, 19] * 448.0);//works out the height of the object
                int obx = Math.Abs(Convert.ToInt32((bw / 2.0) - bx));
                int oby = Math.Abs(Convert.ToInt32((bh / 2.0) - by));

                int obxx = Math.Abs(Convert.ToInt32((bw / 2.0) + bx));
                int obyy = Math.Abs(Convert.ToInt32((bh / 2.0) + by));


                for (int i = obx; i < obxx; i++)
                {
                    bmp.SetPixel(i, oby, Color.Green);
                }
                for (int i = oby; i < obyy; i++)
                {
                    bmp.SetPixel(obx, i, Color.Green);
                }
                for (int i = obx; i < obxx; i++)
                {
                    bmp.SetPixel(i, obyy, Color.Green);
                }
                for (int i = oby; i < obyy; i++)
                {
                    bmp.SetPixel(obxx, i, Color.Green);
                }
            }

            if (expected[x, y, 0] == 1.0 && expected[x, y, 1] != 0.0 && expected[x, y, 1] != 1.0 && expected[x, y, 2] != 0.0 &&
expected[x, y, 2] != 1.0 && expected[x, y, listboxindexnum + 5] == 1)
            {
                double bx = (expected[x, y, 1] * 64.0) + (x * 64.0);
                double by = (expected[x, y, 2] * 64.0) + (y * 64.0);
                double bw = (expected[x, y, 3] * 448.0);
                double bh = (expected[x, y, 4] * 448.0);
                int obx = Math.Abs(Convert.ToInt32((bw / 2.0) - bx));
                int oby = Math.Abs(Convert.ToInt32((bh / 2.0) - by));

                int obxx = Math.Abs(Convert.ToInt32((bw / 2.0) + bx));
                int obyy = Math.Abs(Convert.ToInt32((bh / 2.0) + by));

                for (int i = obx; i < obxx; i++)
                {
                    bmp.SetPixel(i, oby, Color.Green);
                }
                for (int i = oby; i < obyy; i++)
                {
                    bmp.SetPixel(obx, i, Color.Green);
                }
                for (int i = obx; i < obxx; i++)
                {
```

```
                    bmp.SetPixel(i, obyy, Color.Green);
                }
                for (int i = oby; i < obyy; i++)
                {
                    bmp.SetPixel(obxx, i, Color.Green);
                }

            }
        }

    }
    pictureBoxImg.Image = bmp;

}

private void buttonabstract_Click(object sender, EventArgs e)
{
    Abstractor.trainorform = true;
    Abstractor.bmp = default_bmp;
    Abstractor abs = new Abstractor();
    Thread tr = new Thread(() => abs.main());
    tr.Start();
    loading = true;
    loadinglabeltxt = "Abstracting image";
    loading_max = 28;
    loading_value = 0;
}

private void timer1_Tick(object sender, EventArgs e)
{
    if (Abstractor.finished == true  && Abstractor.trainorform == true)
    {

        loading = false;
        loadinglabeltxt = "";
        loading_max = 0;
        loading_value = 0;
        Abstractor.started = false;
        Abstractor.finished = false;


    }
    if (loading == true)
    {
        progressBar1_loading.Visible = true;
        label1_loading.Visible = true;
        pictureBox_loadingback.Visible = true;
        pictureBox_loadinggif.Visible = true;
        label1_loading.Text = loadinglabeltxt;
        progressBar1_loading.Maximum = loading_max;
        progressBar1_loading.Value = loading_value;


    }
    else if (loading == false)
    {
        progressBar1_loading.Visible = false;
        label1_loading.Visible = false;
        pictureBox_loadinggif.Visible = false;
        pictureBox_loadingback.Visible = false;

    }
}
  }
}
```