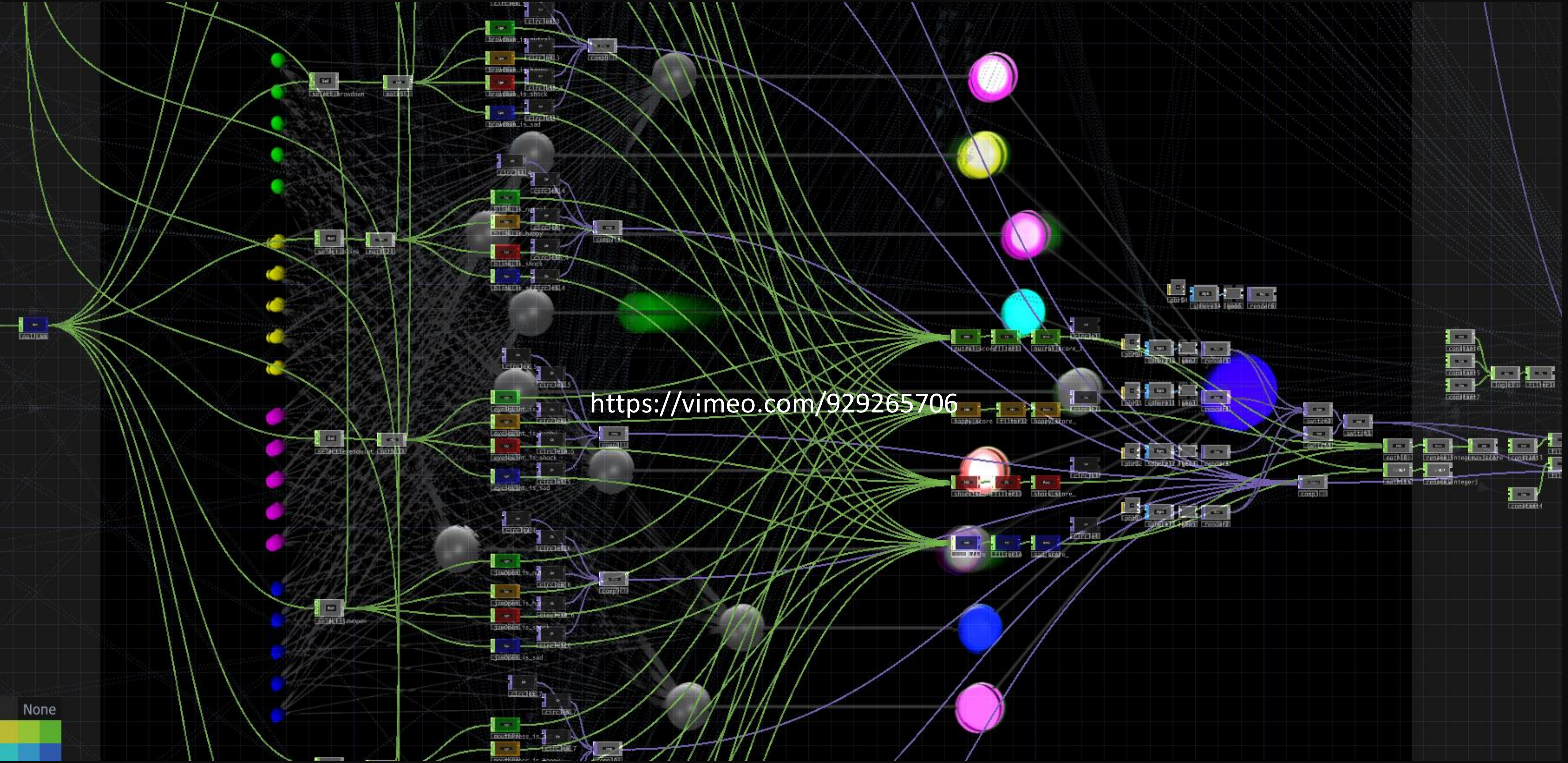
Visualization of a system of conversation
(live installation)

This is a post-human's approach to the concept of cyborg in posthumanism. The actors within any network tend to interact within an open system, which includes a feedback loop. This dialectical ping-pong results in mutual growth and development of the two actors. I will call this dialectical interrelation a system of conversation.

Parisa Salimi



https://vimeo.com/929265706

My initial idea revolves around exploring conversational design, its implications, and its potential within the context of the modern era. At the heart of this exploration lies the question of the relationship between beings, particularly between humans and machines, and how we can navigate concepts like transhumanism and posthumanism.

I aim to redefine the conventional hierarchical relationship between humans and machines, envisioning them as equal actors engaged in a symbiotic partnership rather than one dominating the other. This approach challenges the outdated object-subject dualism and instead fosters a relationship that cultivates mutual growth and understanding.

Central to this idea is the concept of the human being as a cyborg, seamlessly integrated with machine components, neither controlled by them nor controlling them. It's a response to the prevailing technological landscape and the fear of intelligent systems replacing or controlling us, turning humans into adversaries of their own creations.

Contrary to viewing creations as mere lines of code dependent on their programmers, as if they cannot even introduce the coder or the audience with the element of surprise within the growth. This stands in contrast to the trend of interactive art, which often falls short of true interactivity and growth. This is reactive art, called interactive by mistake!
a reactivity without evolving or surprising.

In pursuit of a more meaningful interaction, I'm exploring a parallel possibility in which the two actors in this cybernetic, post-humanistic relationship can engage in conversation. This conversation may sometimes have a goal, and at other times, it may not. What I find important is that at any moment you pause the conversation and reflect on the last phrase spoken, it's not the same as it was a minute or an hour ago. It has evolved; it has progressed.

Addressing the fear of the unknown inherent in technologies like neural networks, I suggest making these systems more transparent and visible, allowing humans to comprehend their intricacies and appreciate their marvels. This transparency fosters a deeper connection between humans and machines, enhancing the conversational dynamic.

Another aspect worth mentioning is the concept of black boxing, which contributes to the fear surrounding technological advancements. The fear of the unknown, particularly evident when working with neural networks. Nowadays, almost every aspect of technology involves neural networks, often with hidden layers in between. While some neural networks may seem simple, their inner workings remain obscure. Even if the programmer understands the network's structure, they may not comprehend the intricacies of the libraries they utilize. By making neural networks more explicit and visible, we can demystify their complexity. This transparency allows humans to witness both the chaos and wonder within these systems, almost as if they were alive. As if they were one of us.

In practical terms, my idea involves a system where data input triggers a neural network, producing a visualization of the process that makes it more tangible for humans to react to. Through a backpropagation function, the machine learns from its mistakes, updating itself to improve accuracy. Crucially, both human and the machine react to each other's outputs, transforming the interaction into a genuine conversation where both sides contribute and evolve. Additionally, the hidden layers within the intelligence system become visible, allowing humans to understand and engage with the inner workings of the machine.

The project process

I stepped foot into learning about machine learning and the most important concepts. I found out about the types of intelligent networks and the layers in between. To have a better understanding, I started training simple ML models like text and image classification with linear regression, sigmoid function, and decision tree. I figured if I am creating a creature that is going to develop, then I'm going to learn about how to give it some sort of life.

This was where I learned about backpropagation and how to train the network to make better predictions.

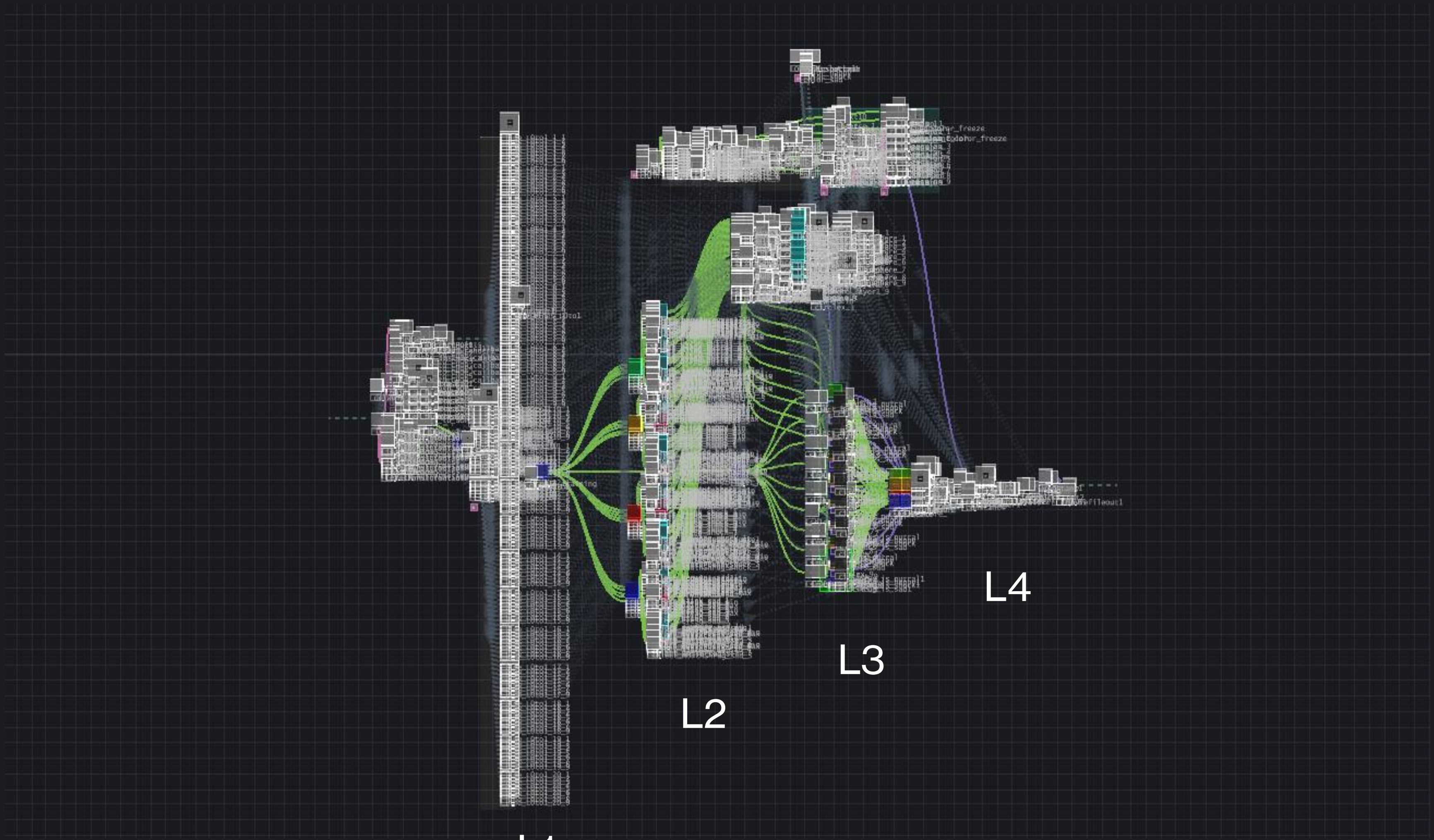
According to my research, machine learning network visualizations come in two types: those illustrating the training process and those showing real-time decision-making.

Existing visualizations predominantly focus on training, but I was intrigued by the prospect of showcasing live decision-making.

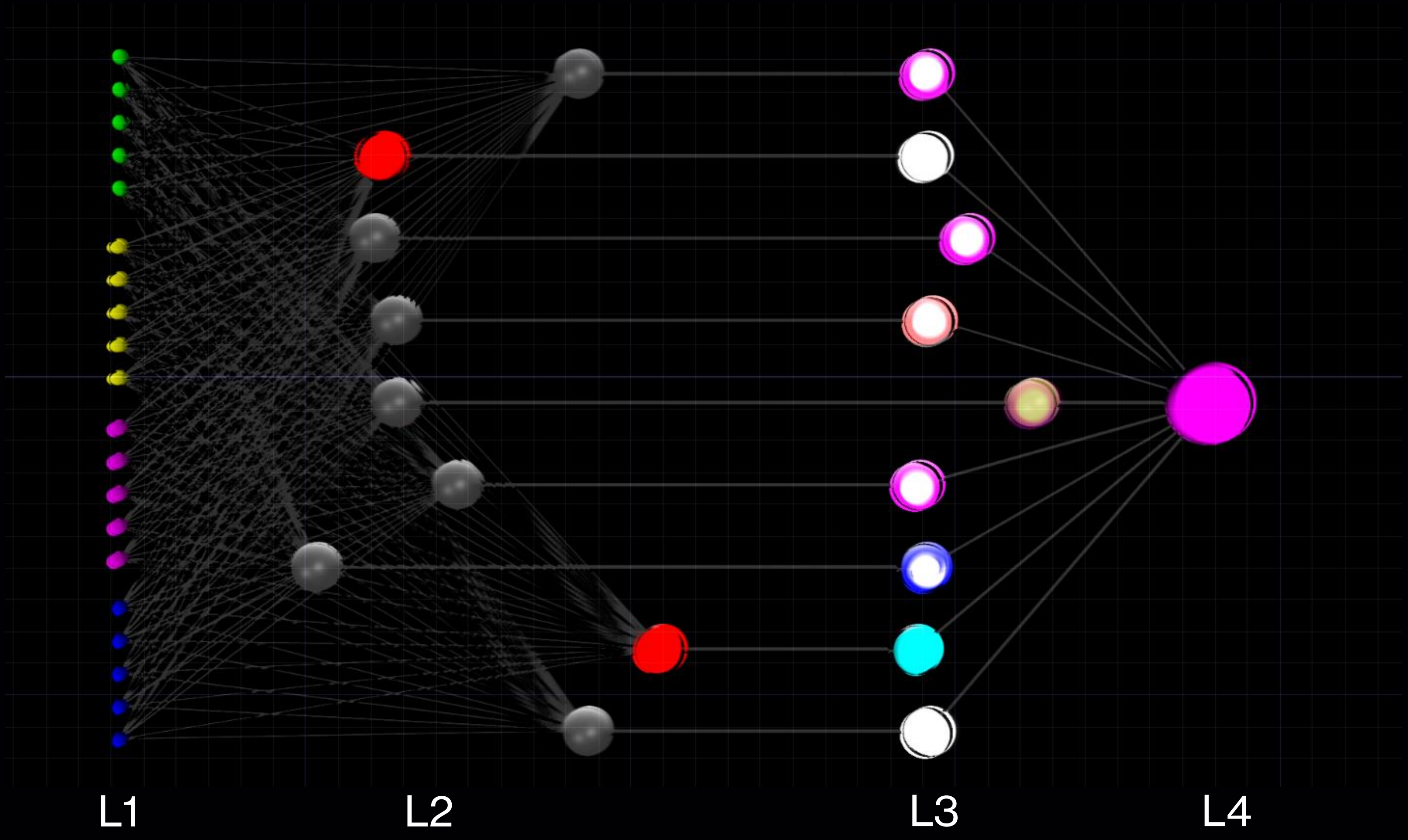
Given my intention to define a training function, I ultimately decided to visualize both processes.

I devided the two processes into four layers:

The first layer is associated with learning, and the other two show the decision-making. Additionally, one layer is added in between for better visual understanding of the network.



This is how the underlying neural network responsible for both the training and decision-making processes looks like

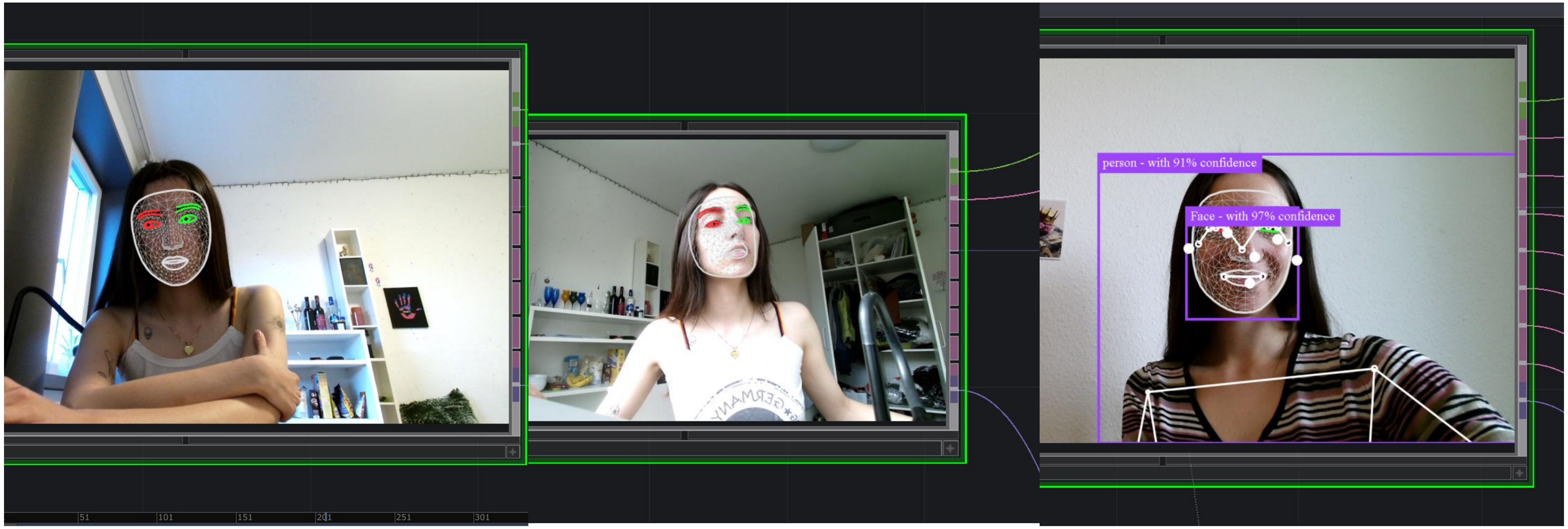


This is the final visualization of the neural network

The first layer (input training data):

Every neural network has a dataset that was trained on. In my case, the dataset is seemingly short, containing 20 individual snapshots recorded by me.

Each data is the facial feature information collected from an image, and each bundle of five snapshots is associated with one emotion. There are five neurons (input information) for each emotion, which are neutral, happy, shock, and sadness.



The second layer (the properties):

Contains nine separate neurons, each representing a specific facial feature. This layer was not necessary but was added for the tangibility of the backpropagation process for the human being. It shows how each dataset is connected to each facial feature and can affect all of them. It also is updated live once the dataset is updated in the training process.

The third layer (the decision-making):

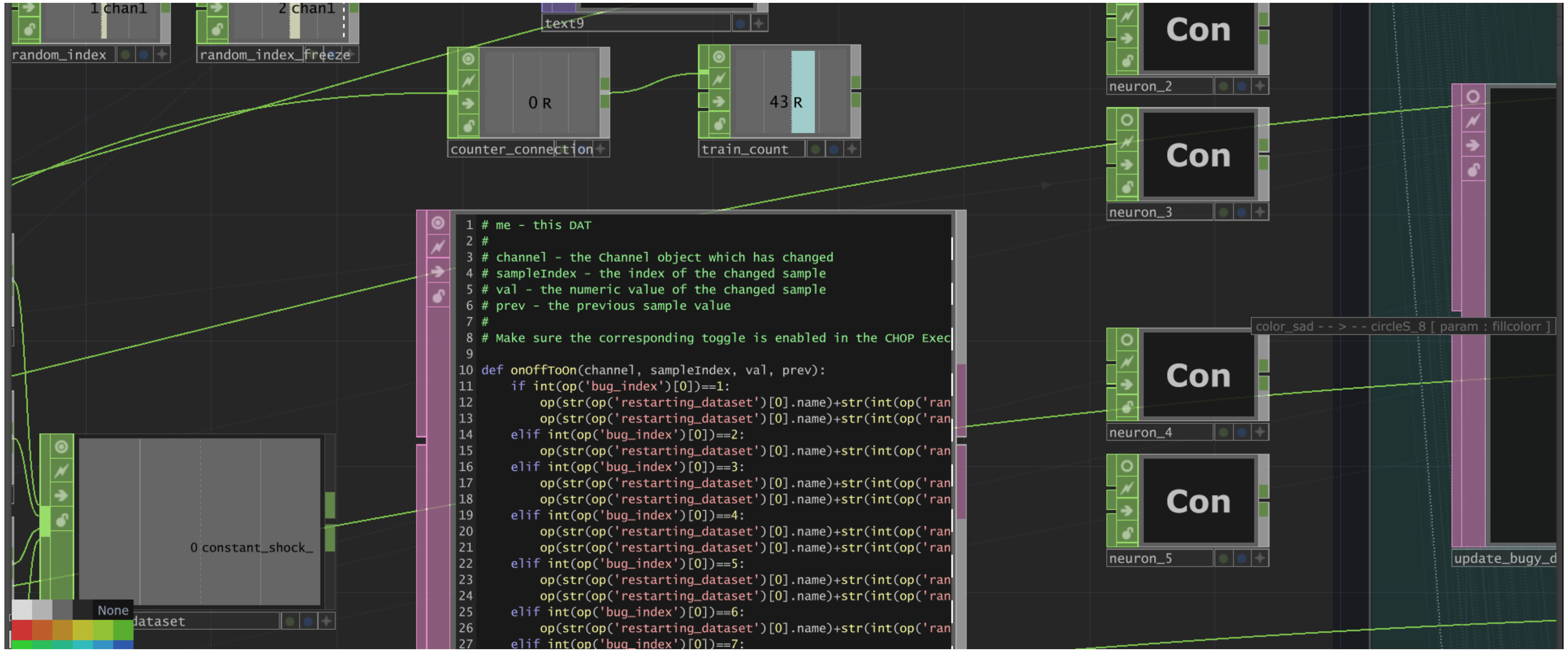
In this layer, the neurons are directly connected to the feature properties layer, showing how based on the datasets and the properties, each feature is associated with one emotion. For example, based on the properties and the shape of the mouth, the network checks to see if the mouth matches the mouth samples in each data bundle. If it matches, then the neuron turns to the color of that emotion.

The fourth layer (the prediction):

Is the sum of all the individual neurons in the layer before. If most neurons are suggesting blue, then the prediction will be that the emotion is blue.

The training

The training is done in two parts. One is the initial training, and the other one happens live within the interaction with the human.



```
# me - this DAT
#
# channel - the Channel object which has changed
# sampleIndex - the index of the changed sample
# val - the numeric value of the changed sample
# prev - the previous sample value
#
# Make sure the corresponding toggle is enabled in the CHOP Exec

def onOffToOn(channel, sampleIndex, val, prev):
    if int(op('bug_index')[0])==1:
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
    elif int(op('bug_index')[0])==2:
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
    elif int(op('bug_index')[0])==3:
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
    elif int(op('bug_index')[0])==4:
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
    elif int(op('bug_index')[0])==5:
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
    elif int(op('bug_index')[0])==6:
        op(str(op('restarting_dataset')[0].name)+str(int(op('ran
    elif int(op('bug_index')[0])==7:
```

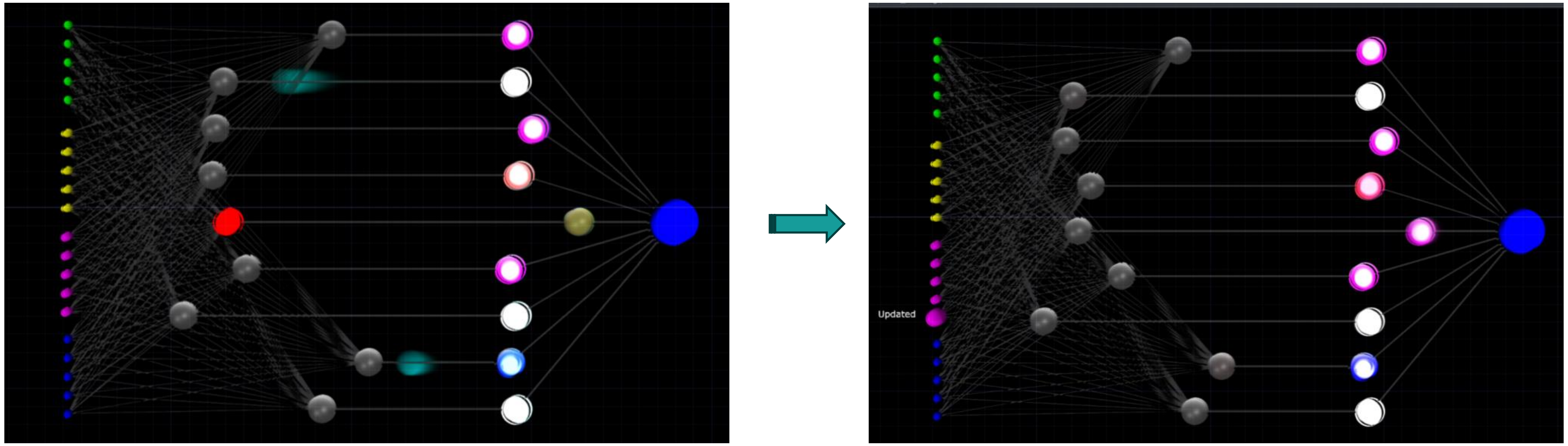
The initial training:

I used a MediaPipe library from Python, which gave me the live data of a face, including all of its features as numbers. I then took five photos posing in each emotion. Then, I broke the data read from the image, which was the facial features data (for example, one was how open the jaw is), into the ones that are going to be affected in each emotion the most. As an example, the nose stays almost the same in each emotion, and learning this from going through machine learning tutorials, it is only going to make the network biased.

The features ended up being nine, and I have extracted from them the range in which each of them is associated with each emotion. For example, the mouth shrink number is between 0.12 to 0.312 in sadness, based on my own dataset. So, if the number in live interaction is within that range, the network suggests that the mouth is in the sadness range.

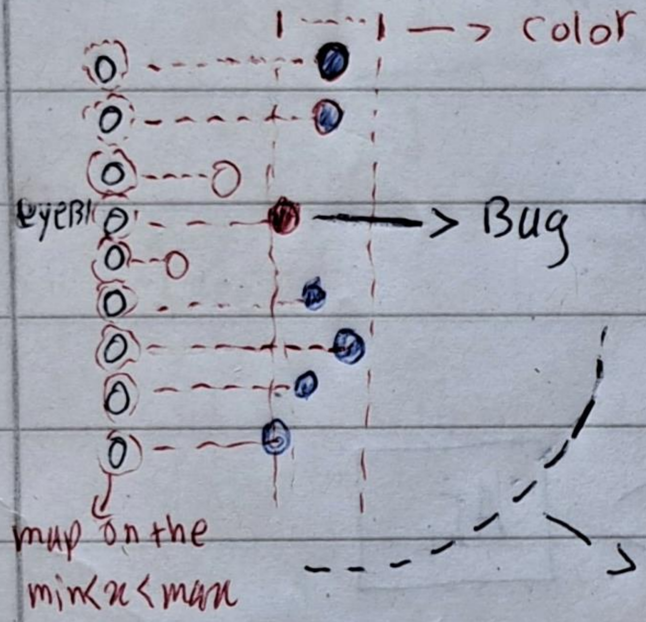
The live training:

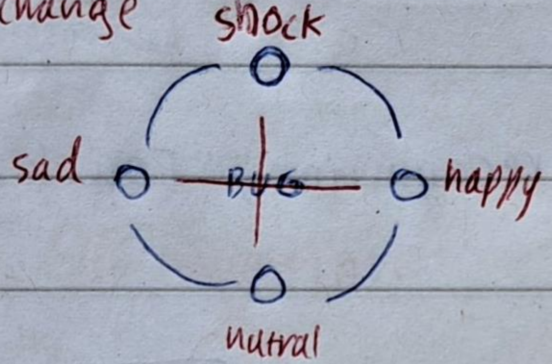
Since the prediction that is made is based on the color that is dominant in the third layer, there are always other minor colors within. If more than n neurons have the same color, then if only one neuron has the exact opposite color, it is spotted as a bug. The color of the bug and the index of the neuron are recorded, traced back into the first layer, and then a random snapshot among the corresponding datasets is updated for its similar feature. As an example, if the majority of neurons in the third layer suggest the color purple (shock emotion), and only the fourth neuron, which is responsible for the eyebrow shape (if more than one, then maybe it's not a bug), is green, which is the complementary color, then one data snapshot among the five snapshots of the shock emotions is going to swap its old values for the eyebrow shape with a live new one.



A little dive into the programming

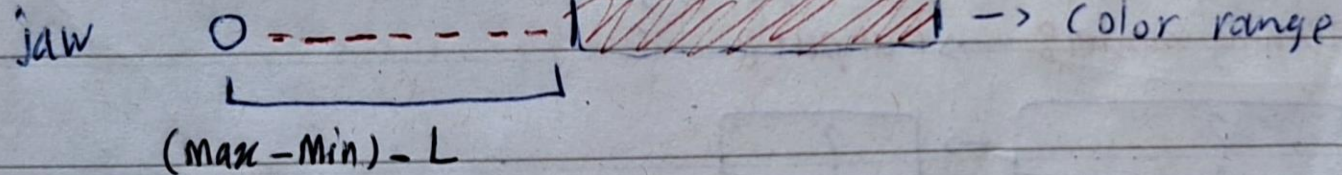
TD Visualisation : (not a NN for training, but for Desicion making)



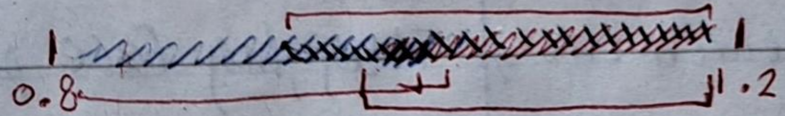


if only one Bug :
retake photo & classify
as the new data

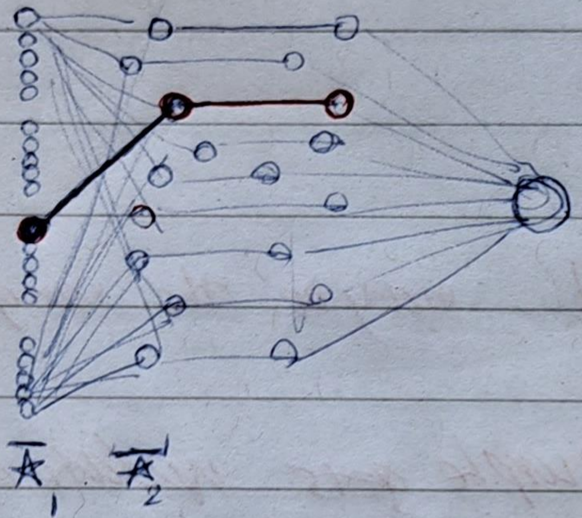
GUOURUB=L



ex : (0-2) L=(0.8-1.2) G=(0.9-1.1) O=(0.8-1.15) R=(0.9-1.2)



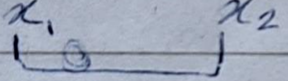
Data B.M prediction



$\bigstar_2 \rightarrow$ x = (op('range) max - min) / 2

Scale = Live Data maped

if RGB [0-9] opposite main RGB
and more than 6 in range(RGB) main



$\rightarrow$ jawopen_N_range

R G B 1-R 1-G 1-B

min ~~(R & 1-R)~~ $\rightsquigarrow$ R r = R ± 0.1

if 1 node with r in range R±0.1 **and** >6 nodes with r in range (1-R) ±0.1 and abs((1-R) - R) ~~X~~ = 4 or
1 node with r in range (1-R)±0.1 and >6 nodes with r in range R±0.1 and abs((1-R)-R) > 4 :

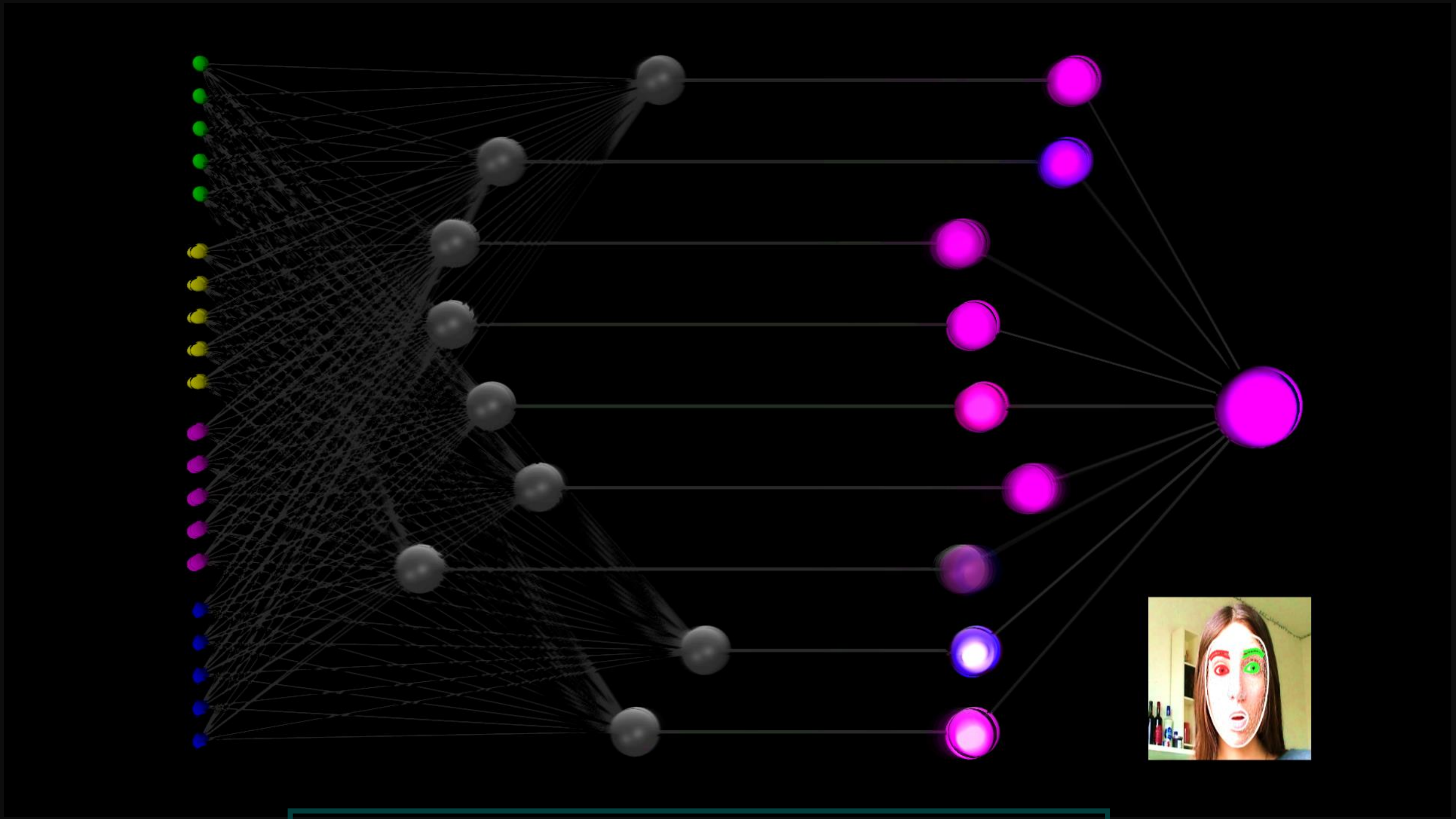
0.2

The logic of the program is implemented within the TouchDesigner software using a combination of Python and node-based programming.

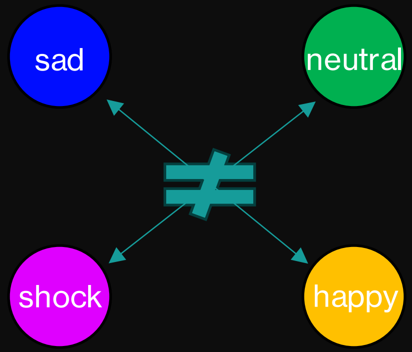
The most challenging function was the backpropagation, where I attempted to write a function to identify and record the data (bug color, bug index and dominant color). I devised a method to identify the bug by utilizing the neurons' RGB channels. This led me to decide to replace the random colors associated with each emotion with two sets of complementary colors. The function now only identifies the complementary values. For example, if the dominant color is R=0.5, G=0.5, B=0, and one neuron has R=0, G=0, B=1, first, a function intensifies the colors, resulting in R=1, G=1, B=0 and R=0, G=0, B=1. Then, if more than 'n' neurons are identified with the same value, the code checks to see if there is one neuron that meets this condition:

$$(R(\text{dominant})-R) + (G(\text{dominant})-G) + (B(\text{dominant})-B) = 3$$

The visualization was created using the same logic but replacing the logical outputs with simple graphics.



●	●	●	●
sad	shock	happy	neutral


sad
neutral
shock
happy

Quotes and reference theories

Systems philosophy

László explains that the new systems view of organized complexity went one step beyond the Newtonian view of organized simplicity which reduced the parts from the whole, or understood the whole without relation to the parts. The relationship between organizations and their environments can be seen as the foremost source of complexity and interdependence. In most cases, the whole has properties that cannot be known from analysis of the constituent elements in isolation.

Actor–network theory

Actor–network theory (ANT) is a theoretical and methodological approach to social theory where everything in the social and natural worlds exists in constantly shifting networks of relationships. It posits that nothing exists outside those relationships. All the factors involved in a social situation are on the same level, and thus there are no external social forces beyond what and how the network participants interact at present. Thus, objects, ideas, processes, and any other relevant factors are seen as just as important in creating social situations as humans.

Actor–network theory tries to explain how material–semiotic networks come together to act as a whole; the clusters of actors involved in creating meaning are both material and semiotic. As a part of this, it may look at explicit strategies for relating different elements together into a network so that they form an apparently coherent whole. These networks are potentially transient, existing in constant making and remaking. This means that relations need to be repeatedly "performed" or the network will dissolve.

Humans and non-humans are both independent and the product of their interrelation.

However, this is different from the idea of flat ontology, which in this case could be a critique of Graham Harman's object-oriented ontology and Bruno Latour's actor-network theory.

Flat ontology is a model for reality that says that all objects, even those that are imagined, have the same degree of being-ness as any other object. No object is more a subject than any other. All subjects are simply objects.

Object-oriented ontology

In metaphysics, object-oriented ontology (OOO) is a 21st-century Heidegger-influenced school of thought that rejects the privileging of human existence over the existence of nonhuman objects. This is in contrast to what it calls the "anthropocentrism" of Kant's philosophy by proposing a metaphorical Copernican Revolution, which would displace the human from the center of the universe like Copernicus displaced the Earth from being the center of the universe. Object-oriented ontology maintains that objects exist independently (as Kantian noumena) of human perception and are not ontologically exhausted by their relations with humans or other objects. For object-oriented ontologists, all relations, including those between nonhumans, distort their related objects in the same basic manner as human consciousness and exist on an equal footing with one another.