DOI: http://dx.doi.org/10.26483/ijarcs.v12i2.6700

ISSN No. 0976-5697

Volume 12, No. 2, March-April 2021

International Journal of Advanced Research in Computer Science

RESEARCH PAPER

Available Online at www.ijarcs.info

MODEL CREATION FOR AI IN A GRAPHIC PROGRAMMING ENVIRONMENT THROUGH OPEN-SOURCE BLOCK-BASED PROGRAMMING

AyseKok Arslan
Silicon Valley researcher, Oxford Alumni- Northern California

Abstract: This paper describes an open library based on an open-source site that empowers non-professionals to create machine learning programs. The library includes modeling blocks, explaining training and data validation, training and prediction. The purpose of this study is to provide those people who have no background in computer science or who have limited programming skills with tools for designing, training, testing, and using advanced machine learning models. To confirm the complexity of AI and its algorithm, students must see simple tests of integration with two neural networks. This paper concludes with recommendations for future work.

Keywords: Machine learning, visual editing, construction, neural nets, artificial intelligence

I. INTRODUCTION

Computerized imaging (CT) is key to the development of technologically advanced societies, and is very important for inclusion in studies [1]. To better understand CT, coursework should move from traditional classrooms to informal settings and partnerships.
Over the past decade mechanical learning (ML) systems have demonstrated outstanding skills including image recognition, translation, autonomous driving, speech recognition, medical image interpretation, writing, complimentary production, robotic control, game play, and much more. Students who experience the advantages, disadvantages, and weaknesses of these technologies may gain a deeper understanding than those who simply study technology. This view reflects Dewey's (1938) view of 'practical learning', which emphasizes the importance of experience and involvement. While a small fraction will continue to be AI researchers or engineers, the remaining students may be better prepared to contribute, in an informed way, to a rapidly changing society due to the impact of AI[2, 4].

This study develops some strategies on how to improve CT by understanding AI through a graphic design environment. After giving a brief review of the textbooks, go into the software rating description and how the teacher will use it. The last section provides some ideas for the whole process.

II. BACKGROUND

AI is seen as a technological innovation to transform our society, economy and jobs in the digital community, let alone its accelerated outcome during the Covid phase. Some of the most well-known examples of AI non-driving vehicles, chat chats, voice assistants, online search engines, robotic vendors, etc. Despite the destructive AI myth represented in films like Terminator or I, Robot, the fact is that nowadays clever algorithms contain a series of simple rules applied to a great series of numbers, and the result is called AI[3, 5].

This study examines basic statistics behind simple AI algorithms developed on open source platforms to promote computerized imaging (CT). The purpose is to raise awareness of the rules behind smart programs rather than reading or memorizing anything. Integrated software understanding functions for automatic integration, learning and predicting AI.

The term computational thinking (CT) was first coined by J. Wing in 2006[5, 7]. Phiko suggested that there should be a “global operating environment and a set of skills” to use “decomposition and decay” to deal with the complex tasks and thought of a computer scientist[2, 11]. In his seminal paper Wing (2008)[5, 6] aims to "promote public interest in psychological entertainment" and "spread the joy, fear and power of computer science. To counter criticism, Phiko (2008) published another paper in which he asked the following question: computer thinking (in children)? ". According to Wing (2008), CT will be an important part of children's education.

The term CT was first mentioned by Papert in his book "Mindstorms: Children, Computers, and Powerful Ideas" in 1980 [8, 14], and reappeared in 1996 in his book "An Exploration in the Space of Mathematics Educations" [7, 15] and was explained by Wing [11] in the above edition of the same name. The program's environmental program, with its iconic tortoise, and LEGO Mindstorm learning tools, developed by Seymour Papert, are widely known and regarded as the First Predecessors. The work of MIT media labs is a whole-kindergarten health team and Mitchel Resnick in the Scratch program area and its ideas for using the constructivist ideas and community of creators and creators formed by the current educational structures[8, 18].

Other scholars in the field, such as DiSessa, who developed the concept of “computational literacy” and Pea and Grover, who developed the concept of “procedural literacy”, were among the main opponents of Wing's name for CT[7, 19].

Contradicting these experts, Hu noted that, if CT "thinks about the removal of the process, then Jean Piaget's stages of Cognitive Development may suggest that this thinking ability cannot be successfully taught until adolescence", making it unsuitable for K-12 education. His argument is that CT is a combination of different ways of thinking, especially involving mathematical thinking[18m 20].

In addition, to suggest a strong correlation of statistical output, the definition of CT was added by Aho which focuses on the close relationship between algorithmic thinking and CT[18]. He says "CT should be imaginative processes involved in creating problems so that their solutions are represented as computational steps and algorithms." (page 2).

There is widespread consensus among computer scientists that it is very difficult to teach the basics of computer thinking, let alone AI (Kahn and Winters, 2018), when given the availability of a generally agreed method for transmitting multiple skills. AI modeling, algorithms and applications can be taught using simple tools such as simple as paper and pencil, traditional computer programming or hands-on-computer programming.

Given the current practice of transmitting CT skills in the form of code clubs or robotics workshops, technologies such as robots are regarded as mere toys. Indeed, the famous computer scientist Seymour Papert (1990) has shown in his studies that the use of the Lego robot can enable his students to hardware and control software that controls the senses and the environment[15]. Similarly, the use of cheap robots platforms to teach students how the neural network is trained and designed for the problem of moving robots can increase student engagement in AI-related topics.

When it comes to teaching CT coding, various editing tools equate to these methods at different levels: Scratch, Crickets, Karel the Robot, Alice, Game Maker, Kodu or Greenfoot, all based on Logos philosophy[12]. The graphic application areas are easy to use and allow the original experience to focus on design and construction, avoiding syntax system issues.

III. CONCEPTUAL FRAMEWORK

The purpose of this study is to provide those who do not have or have limited planning skills in the tools for designing, training, testing and using in-depth ML models. Relying on blockchain language has more benefits than simply providing "wrappers" to read JavaScript or Python. Blocks can be very readable without the typical costs involved in installing verbose commands.
This study harmonizes this definition of Csizmadia et al. and assumes that CT will allow anyone to:

1. understand what aspects of the problem can be solved with the tool,
2. Analyze whether a particular tool can solve a problem,
3. understand the limit of calculation tools,
4. convert calculator tool to reuse,
5. see new ways to use the tools, too
6. apply computer techniques to any domain.

Although robots are accepted as an effective way to teach students CT skills, this document introduces a comprehensive AI approach, based on the teaching of specific mathematics with the help of Scratch [18, 22]which will introduce students to more than one algorithm.

From an educational point of view, computer tools are able to deepen the study of mathematical and scientific content and the opposite is true [188]. Scratch today is one of the most influential sites, and has proven to be very effective in engaging and motivating young inexperienced students in programs (Papert, 1990).

IV. RELATED RESEARCH

Historically, computer use refers to mainframe computers for editing and writing hidden lines of text code that can be made. Learning a low-level programming language, such as BASIC established in 1964, meant remembering instructions and working in a solid text environment without feedback or help. That learning to think like a software engineer can be made easier and more playable is proved by Papert, who had the original idea of the Logo program in 1967 and made a line for Scratch's continued success[18, 24].

To encourage active participation during the learning process, Kahn and Winters (Kahn and Winters, 2018) have developed a visual library designed for high school students to create AI (Artificial Intelligence) applications. They first created "Snap!" blocks connected to AI cloud services for speech recognition and image compression and speech integration [13, 16]. They then offer block-based communication sites for a few in-depth learning models of services such as reading transfer, discovery, style transfer, and photo labeling. This paper introduces a framework framework that supports the definition of the development of in-depth learning models, their lost functions, their best practices, training parameters, and predictability (Craft2Learn, 2020a). In addition to the library itself, there are learning resources that include guidelines for working with sample projects.

In 2009, Denning wrote a very important and thought-provoking piece in the computer science community (CS), arguing that close communication between computer science and programs is problematic[24]. He emphasized that "re-packing" and "replacing the old idea with 'CS = CT'", reminded readers that CS is a broad and varied field and should be presented and understood as such. In his words, “Statistics are more important than CT. For this reason alone, CT looks like enough CS”[17, 19].

Barr and Stephenson also tried to find out what role the CS community plays in “bringing about practical thinking in education”[7, 8, 11]. Emphasizing complex and sometimes controversial relationships between these fields, on the other hand, allows CT to serve as "an automated problem-solving and transcendental solution" to solve complex problems around the world, while on the other hand, the very words field holders and somehow link CT with CS and, in particular, the system.

Finally, Barr and Stephenson show that the larger CS community can help to achieve structural change by providing "clear examples of how it works and can be incorporated into a variety of study areas".

Researchers Barcelos and Silveira are exploring the suggestion that CT, "as a means of consultation and problem solving", builds the existing relationship between maths and CS. The authors aim to draw on skills in the field and argue that it is possible to link different disciplinary therapies with CT skills, repeating the importance of a separate education for better future exchanges. One of the biggest challenges facing CT is the lack of cooperation between different countries and different sectors.

Lye et al. they tested a different angle to CT skills by reviewing their "teaching and learning ... systematically". This study places importance on the relationship between CT and planning skills, examining 27 and nine studies focusing on K-12 education. The focus on visual programming languages, like Scratch, is widespread, and the use of authentic problem “pertinent to the students” is very well received and should be added to rigorous reflection activities. The researchers found a problematically low percentage of studies in this age bracket combined with the lack of representative studies in classroom settings critiquing that there “seems to be an implicit assumption that learners can exhibit such computational practices and perspectives through pure self-discovery”.

The 2015 article makes the case move from CT to a computer-based process to "test the strength of visual learning support programs", examining the learning skills of elementary-school children working with e-textiles in Germany[11, 16]. To transform CT into computer simulations, the authors suggest five key elements of expanding CT, namely aesthetics, art, architecture, representation and architectural materials. The authors propose to preserve this very important form of education within classrooms and institutions, reminding all of us that "code should be accepted as a way of doing digital critical things"[12, 19].

When it comes to integrating CS into the curriculum, today's efforts largely incorporate the use of ML-based language-enabled language through platforms such as the Mechanized Learning for Kids (2020) website and the Cognimatesproject [18]involving blocks that provide easy access of various cloud AI services. This does not explicitly provide a plan for building neural networks - connectors dealing with the programs they offer for training and using neural networks only.

In the model provided in this paper, there is no black box implementation in JavaScript, and no reliance on complex APIs or cloud services. Enabling learning to see how ML works with blocks they are familiar with obviously has its advantages. However, legally it is very difficult to achieve the speed and proportion that these blocks can do. Also, blogs provide access to very powerful APIs which can be a great effort to completely replicate in a click-through environment. Ideally, students should be able to access libraries in order to be able to incorporate the most obvious functionality and performance in their projects as appropriate.

To give some examples, Google's Teachable Machine [15, 17]provides a web page where users can train the image classification system while TensorFlow provides a playground to collaborate, train, and test an in-depth neural learning net.

V. CODING EDITING BLOCKS

The block-based programming program is almost identical to Scratch's most popular Scratch program [Resnick et al. 2009]. All blocks in this library can be defined in JavaScript or in terms of other blocks ultimately dependent on JavaScript-defined blocks. It should be noted that this means that the source code does not need to be changed when it comes to implementing extensions in a clear environment.

The program can be run with JavaScript so it can run on any modern web browser without the need for any extensions or plugins.

A block-based programming program has several advantages over text-based programming. Because blocks only click together when they are compatible, eliminate the need to read syntax and, perhaps most importantly, eliminate the possibility of syntax errors or misspelled commands. When using a block, users do not have to memorize a large amount of the original language. Well-crafted learning activities can help students use blocks to reduce the burden of comprehension during the learning process [16]. While this reduces memory requirements for users and facilitates the availability of new functionality, it can be much slower than typing. The graphical encoding mode speaks to this by providing a keyboard path to search blocks.

Accordingly, Mitch Resnick's preschool life team added, in their 2009 paper "Scratch: Programming for All", the third foundation to establish "broad walls", to allow students more work to try and learn from. The three principles described translate well into other popular coding tools and inform CS teaching efforts in general.

V.I DEEP NEURAL NEETWORKS

Many of these basic neural network ideas are over fifty years old [13]but began to lead to thousands of useful applications over the past decade (Deepindex, 2020) from powerful search engines and large data acquisition data. Computation engines often exploit graphical processing units (GPUs) found on many digital devices that can reduce the time it takes to train a model.

Recent ML projects can exploit speedups using GPUs due to the advent of TensorFlow.js [17]. This is the use of TensorFlow, the popular machine API, in JavaScript. It can access the GPU of a portable computer, desktop computer, or phone using the WebGL interface [12] supported by all modern web browsers.

Neural nets only work with numbers. To work this with division functions, numbers are used to label. For example, if the text of a particular text is “denied”, “neutral”, or “agreed” this may be written as 0, 1, and 2. Training blocks convert text labels into “single hot inserts” (vectors with 1 and 1 0). Image input is usually converted to a pixel price list (either black and white (0 and 1), grayscale (0.0 to 1.0), or values.

Weights associated with neurons are initiated randomly. These weights are updated as the model is trained in data. In the supervised reading, the data included examples of the results associated with the input. During training the system adjusts the instruments in an effort to reduce the difference between its prediction and the desired results given to the datasets. Figure 1 provides one example.

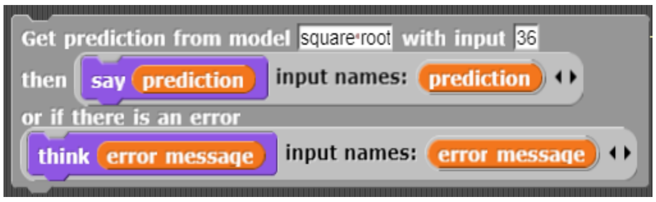

Get prediction from model square root with input 36
then say prediction input names: prediction
or if there is an error
think error message input names: error message

Figure 1. An example of a code-based environment (source: MIT Scratch tool)

While students may have trouble finding millions of labeled images and uploading them to a browser, professionally built models can take weeks of training in large photo collections. While many activities are therefore impossible for students to try, fortunately, there are still many exciting activities that do not require large data sets or computer resources.

Most neural net applications run on servers that receive data from the client and respond to the prediction. This allows service providers to host their models on very powerful servers, sometimes on special Hardware to speed up machine learning. Many business models rely on providing server performance.

There are some problems, though. Many people are concerned about privacy issues when using these services. Voice and video are often transmitted to servers. The cost of running the server is usually a barrier. Another disadvantage of using models on servers is that applications cannot respond like those operating locally on user devices. Also, server-based applications only work when the client has a fast and reliable network connection.

By using the user's device in the web browser these issues are avoided. One can download this app and its library and run everything without an Internet connection.

V.II Library

The click-to-click environment for natural animation is available in two ways:
(1) a set of blocks that can be inserted in any position or
(2) a project that includes conditions that reflect the use of blocks and informative comments. One can simply click on any conditions to use them.

V.III MODEL CREATION

A simple block building model is shown in Figure 2. Build a model called a ‘guessing relationship’ by inserting one connected to 100 neurons. Each of these connected neurons received 50 neurons connected to a single output neuron. When the program has finished creating the model the "say" block is run.

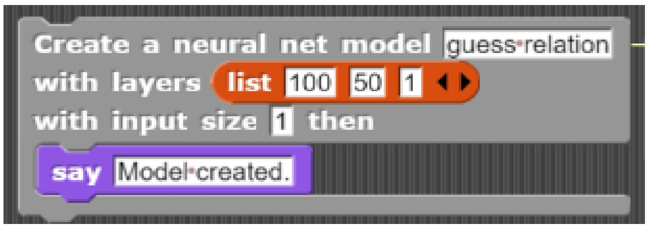

Create a neural net model guess relation
with layers list 100 50 1
with input size 1 then
say Model created.

Figure 2. Simple model building block (source: MIT Scratch tool)

Figure 3 shows an example of a full installed version of this block. It differs from the example in Figure 2 by describing how to perform and perform the desired loss function. Note that this is displayed using drop-down menus for easy use. Documents for these methods and functions are provided in the help menu item and in the program directory for more details. Examples of using this block to perform complex tasks are explained over time.

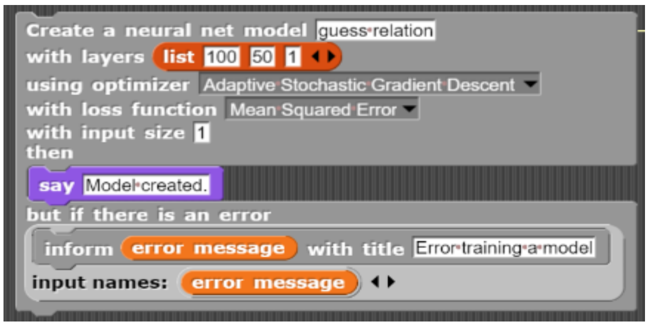
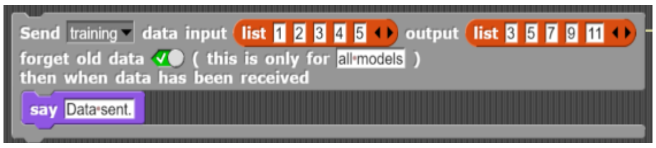

Create a neural net model guess relation
with layers list 100 50 1
using optimizer Adaptive Stochastic Gradient Descent
with loss function Mean Squared Error
with input size 1
then
say Model created.
but if there is an error
inform error message with title Error training a model
input names: error message

Figure 3. Fully installed block for model construction

V.IV TRAINING

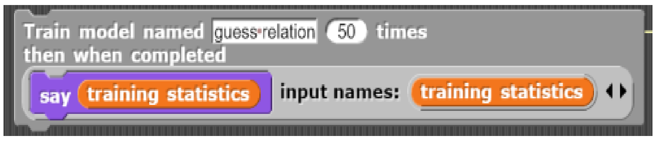
Once the model is built one can start training it. First, the training database needs to be clarified. A database has two lists with the same number of items: input and output. Lists can contain numbers, or numerical values (eg correlation or height blue to blue), or any number of column values. Output can also contain text strings that are converted into numbers internally. The verification data, if provided, does not affect weights during training and is used to provide prediction without the risk of overriding the test model.
In Figure 4, a database containing the first five digits is used as input and the result is calculated using 2 * n + 1 (similar to "Hello World"). This block can define a database or provide details added to the current database. Data sets can be found in all models for use or only integrated with a specified model.



```
Send training data input list 1 2 3 4 5 output list 3 5 7 9 11
forget old data ✓ ( this is only for all models )
then when data has been received
    say Data sent.
```

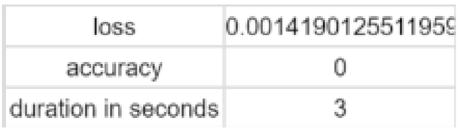
Figure 4. Specification training (source: MIT Scratch Tool)

After specifying the database one can start training. Figure 5 shows a very simple block sample for starting training. It asks for 50 repetitions of the training step and shows the figures from the training as shown in Figure 6.



```
Train model named guess relation 50 times
then when completed
    say training statistics    input names: training statistics
```

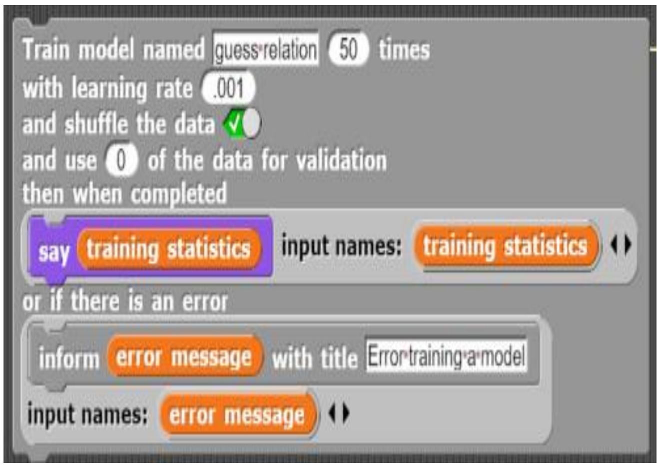
Figure 5. Sample block to start training (source: MIT Scratch tool)



loss	0.001419012551195
accuracy	0
duration in seconds	3

Figure 6. Sample training statistics

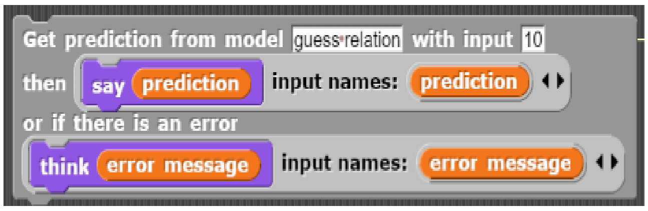
Figure 7 shows how to specify a reading value of .001, that data should be exchanged (to avoid any art objects from order), that no data should be used for verification. It also responds to any errors that occur.



```
Train model named guess relation 50 times
with learning rate .001
and shuffle the data ✓
and use 0 of the data for validation
then when completed
    say training statistics    input names: training statistics
or if there is an error
    inform error message with title Error training a model
    input names: error message
```

Figure 7. Full training start block block (source: MIT Scratch Tool)

Figure 8 shows a block requesting that the model predict the output given 10 as input. In this example, the "us" block will be moved to a number close to 21 (e.g., 2 * 10 + 1). There is a version of this block that accepts the input list and responds with a list of related predictions. If the model is trained to label input, output from a list of pairing labels and confidence scores.



```
Get prediction from model guess relation with input 10
then    say prediction    input names: prediction
or if there is an error
    think error message    input names: error message
```

Figure 8. Predict detection block from model (source: MIT Scratch Tool)

VI. RESEARCH METHOD

This section will show you the algorithms that can be made using this site like Scratch.
For all the algorithms set out in this section, the student's first task is to fill in the blanks provided by the lines between the lines of the code, written in the comments. That is, the reader does not have to create an algorithm or write the whole code; the code will be provided for the most part.

Some algorithms are specified as follows:

• K-means
• Neural Network

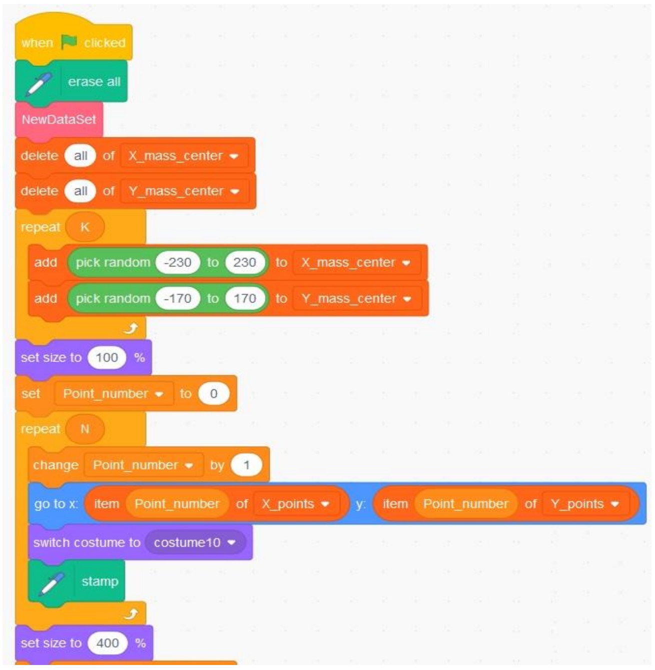
K-means

The K-means algorithm, developed in 1967 by MacQueen (19670, is one of the easiest learning methods for solving a well-known problem. as many collections as we want, the small rule of square error.

From the given cloud of N points and the small cloud of K mass centers, the purpose of this activity is for one to learn to combine points in K groups, by installing the program in Scratch. In order to make a merger, each point will be a group defined by the nearest center of the plural. Finally, each K collection will be colored differently.

Neural Networks

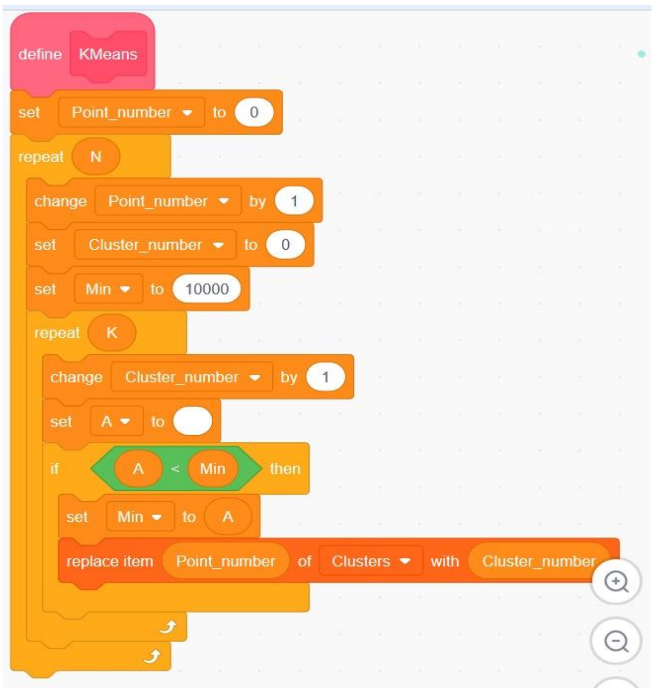
The basic premise of the neural network is to mimic the closely connected brain cells within a computer to discover, learn patterns, and make decisions in a human-like way. A key feature of this tool is that the neural network learns by itself. The program builder only needs to design the physical structure (output volume, input, hidden layers) and set very simple rules that include addition, duplication and output. They are based on perceptrons, produced in the 1950s and 1960s by scientist Frank Rosenblatt (1960), inspired by earlier work by Warren McCulloch and Walter Pitts (1970). It is important to know that neural networks are (usually) software simulations: they are done by setting up the most common computers. Computer simulations are simply collections of algebraic and mathematical simulations.
Typically, neural networks use repropagation type algorithms that require the use of alternatives. However, students can be equipped with this mathematical application in their subjects at the moment, so neural networks can be built on a logic, easily understandable task.
Different exercises are performed on neural networks. First, a simple neural network with two inputs and an outgoing neuron, is trained through the logic gate (see Figure 12). In each iteration, students will see the different weights gained by the neural network. Next, along with the OR logic gate will be used, and as a result, students will see how the adjustment limits change with this new data.



```
when [flag] clicked
erase all
NewDataSet
delete all of X_mass_center
delete all of Y_mass_center
repeat K
    add pick random -230 to 230 to X_mass_center
    add pick random -170 to 170 to Y_mass_center
set size to 100 %
set Point_number to 0
repeat N
    change Point_number by 1
    go to x: item Point_number of X_points y: item Point_number of Y_points
    switch costume to costume10
    stamp
set size to 400 %
```

Figure 12 - Initiates fine art search with hyper parameters (source: MIT Scratch Tool)

The first task is to complete the NewDataSet block system that will create a cloud of N points. The cloud should consist of N points with X links between (−230, 230) and Y links between (−170, 170) (see Figure 12).



```
define KMeans
set Point_number to 0
repeat N
    change Point_number by 1
    set Cluster_number to 0
    set Min to 10000
    repeat K
        change Cluster_number by 1
        set A to ( )
        if A < Min then
            set Min to A
            replace item Point_number of Clusters with Cluster_number
```

Figure 13. Exemplar task- Draft completed to create a cloud of points (source: MIT Scratch Tool)

Block KMeans maintains a large facility that will be allocated per point. To do so, the student must enter the code in the Euclidean distance calculation from one location to another in the variable centers A. The system will then find the smallest distance of all and complete the vector groups, containing the numbers of the collection where each point belongs.

Finally, ColourPoints displays cloud graphs of points and centers of magnitude, each with a color given to Vector Collections.

The AND / OR logic gate algorithm can be built using a large block, which starts with data, and two blocks, Neuron and ExecuteButton.

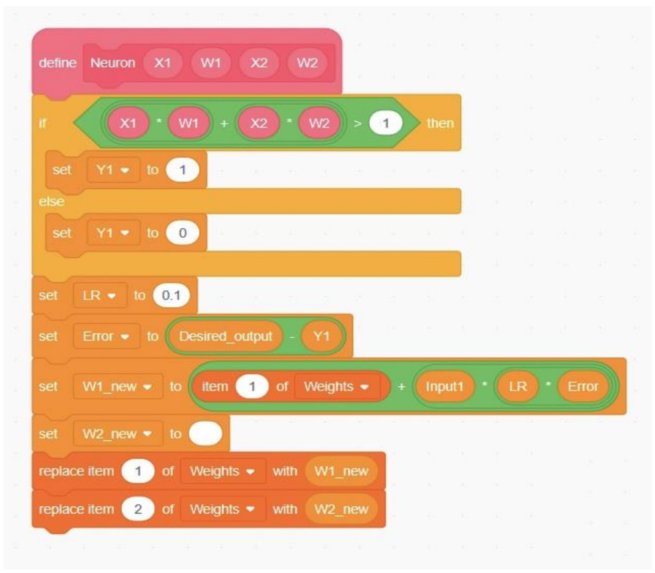
In order to code equations that describe the function of a network neuron, W2 needs to be updated as shown in Figure 13.

Students must train the neuron using two sets of data: one logic gate set AND one OR logic gate.

In addition, additional multilayer add-ons, Neuron1, Neuron2 and Neuron3 and ExecuteButton can be used.

The activity is designed to fill the gaps for the following activities:

• Complete N3 calculation
• W2 and W5 renewal



```
define Neuron X1 W1 X2 W2
if X1 * W1 + X2 * W2 > 1 then
    set Y1 to 1
else
    set Y1 to 0
set LR to 0.1
set Error to Desired_output - Y1
set W1_new to item 1 of Weights + Input1 * LR * Error
set W2_new to ( )
replace item 1 of Weights with W1_new
replace item 2 of Weights with W2_new
```

Figure 14. Work Sample. Unfinished task to calculate weight recovery (source: MIT Scratch Tool)

VII. RECOMMENDATIONS

The activities featured in this paper are questionable; students can get into maths involved in maths or Scratch code itself, or suggest new neural networks to deal with other problems. The use of K methods and neural networks enhances their performance in other formats, such as MS Excel as well.

The use of neural networks can help the student develop their skills to develop neuron networks through the gates of the AND and OR logic.

As a future project, first of all, interested participants can begin organizing an AI workshop to measure the success rate of goals in collaboration with academic researchers. Second, many programming languages such as popular

HTML should be investigated in order to perform more experiments, such as using neural data prediction networks.

VII. CONCLUSION

This paper introduces instructor-directed, easy-to-install activities that can be done using Scratch or similar click-through site. By illuminating the various problem-solving angles, as well as the challenges involved in teaching and learning how to think and solve problems in an effective and timely system, this study raised a series of artificial case studies and effective CT teaching using a block-based system.

It must be recognized that higher intelligence and thinking skills are also essential and powerful in a changing world. The goal should be to experiment with professional communication and find ways to integrate external knowledge, practical technology and useful technology into everyday life. To solve the most interesting challenges such as climate problems, individual minds need to be provided with the best tools and skills.

Last, but not least, "Today's students will face an endless stream of unknown, uncertain, and unexpected situations for the rest of their lives. Their success and happiness depend on their ability to think and act intelligently. "- Mitchel Resnick

REFERENCES

1. Çakiroğlu, Ü., Suiçmez, S. S., Kurtoğlu, Y. B., Sari, A., Yildiz, S., &Öztürk, M. (2018). Exploring perceived cognitive load in learning programming via Scratch. Research in Learning Technology, 26.
2. Druga, S. (2018) Growing up with AI: Cognimates: from coding to teaching machines., PhD diss., Massachusetts Institute of Technology.
3. eCraft2Learn (2020b) https:// ecraft2learn.github.io/ai/AI-Teacher-Guide/chapter-6.html
4. Google (2020a) https://experiments.withgoogle.com/teachable-machine
5. Harvey, B. &Mönig, J. (2010) Bringing "No Ceiling" to Scratch: Can One Language Serve Kids and Computer Scientists? Constructionism 2010 Proceedings, Paris, France.
6. Kahn, K. & Winters, N. (2017) Child-friendly programming interfaces to AI cloud services, Proceedings of EC-TEL 2017: Data Driven Approaches in Digital Education, 10474, 566-570.
7. Kahn, K, Megasari R., Piantari, E. &Junaeti, E. (2018) AI Programming by Children using Snap! Block Programming in a Developing Country, Proceedings of the EC-TEL Conference, Leeds, UK, September 2018.
8. Kahn, K. & Winters, N. (2020) Constructionism and AI: A history and possible futures, Proceedings of Constructionism 2020, Dublin, Ireland, 2020.
9. Loukatos, D., Kahn, K. &Alimisis, D. (2019) Flexible Techniques for Fast Developing and Remotely Controlling DIY Robots, with AI flavor. Proceedings of the Edurobotics 2018 Conference, Rome, Italy, October 2018. In Moro, M., Alimisis, D. and Iocchi, L., 2019. Educational robotics in the context of the maker movement, Springer.
10. Machine Learning for Kids (2020) machinelearningforkids.co.uk.
11. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., & Dean, J. (2013). "Distributed representations of words and phrases and their compositionality." In Advances in neural information processing systems, pp. 3111-3119.
12. Minsky, M. (2019). Inventive Minds: Marvin Minsky on Education. MIT Press. Minsky, M. &Papert, S. (1969). Perceptrons: An introduction to computational geometry. MIT press.
13. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., &Riedmiller, M. (2013). "Playing Atari with deep reinforcement learning." arXiv preprint arXiv:1312.560.
14. National Oceanic and Atmospheric Administration (2019). https://www1.ncdc.noaa.gov/pub/data/ghcn/daily/readme.txt
15. Papert, S. (1980) Mindstorms: Children, computers, and powerful ideas. Basic Books, Inc.
16. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B. and Kafai, Y. (2009), Scratch: programming for all. Communications of the ACM, 52(11), 60-67 DOI=http://dx.doi.org/10.1145/1592761.1592779.
17. Silver, D., Schrittwieser, J., Simonyan, K. Antonoglou, I., Huang, A, Guez, A., Hubert, T. (2017). "Mastering the game of go without human knowledge." Nature, 550, no. 7676: 354.
18. Smilkov, D., Thorat, N. Assogba, Y., Yuan, A., Kreeger, N., Yu, P., Zhang, K. (2019) "TensorFlow.js: Machine Learning for the Web and Beyond." arXiv preprint arXiv:1901.05350.
19. Stoyanov, M., (2018).HPJS: Hyper-parameter Optimization for JavaScript, Medium, https://medium.com/@martin_stoyanov/hpjs-hyperparameter-optimization-for-javascript- 8f78aa7a3368.
20. WebGL (2019) https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API World Health Organization (2019) http://apps.who.int/flumart/Default?ReportNo=16
21. Wolfram, S. (2017a) An Elementary Introduction to the Wolfram Language, Second Edition. Wolfram Media.
22. Wolfram, S. (2017b) Machine Learning for Middle Schoolers. Stephen Wolfram blog. http://blog.stephenwolfram.com/2017/05/machine-learning-for-middle-schoolers/.