Visualizing Machine Learning in 3D

Diego Rivera
drivera13@gaels.iona.edu
Iona College
New Rochelle, NY, USA

ABSTRACT

Understanding machine learning models can be difficult when the models at hand have many parts to them. Having a visual model can help aid in understanding how the model functions. A way to visualize these models is to use a 3D (three-dimensional) game development application. An application that will have an interactive element allowing the users to interact with the model (rotating and scaling it) and see changes at run-time. An interactive element will keep the users engaged, understanding, and seeing how a machine learning model looks and behaves. This paper describes the process of visualizing a machine learning model in a 3D application.

CCS CONCEPTS

• **Human-centered computing** → *Virtual reality*.

KEYWORDS

Neural Networks, Transformers, Interactive

ACM Reference Format:
Diego Rivera. 2022. Visualizing Machine Learning in 3D. In *28th ACM Symposium on Virtual Reality Software and Technology (VRST '22), November 29-December 1, 2022, Tsukuba, Japan.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3562939.3565688

1 INTRODUCTION

Machine learning can be a scary topic for some people as many see it as hard and advanced code that takes years to understand in order to implement it. Machine learning is not as hard nowadays, with many tutorials and applications dedicated to helping to create machine and deep learning models. Virtual Reality is something that has been in development for years, and many users seem to enjoy the virtual applications that were created. Machine and deep learning can be an application that will work well in virtual reality. Creating these models and making them interactive will allow machine learning to be open to the public and hopefully make create a new generation of students willing to work with machine learning. A virtual experience can provide knowledge and experience better than textbook work. A virtual experience can provide fun interactions to keep the user engaged.

VRST '22, November 29-December 1, 2022, Tsukuba, Japan

ACM ISBN 978-1-4503-9889-3/22/11.
https://doi.org/10.1145/3562939.3565688

2 RELATED WORK

There was a three-dimensional interactive neural network that was developed back in 2018. The program illustrated had a feature that allowed the user to see changes in real-time and have limited movement around the model. The program Stefsietz [Sietzen [n. d.], 2022] created was a CNN model that had to change parameters and the physical appearance of the model. The model was very well built, and the application worked as it intended. Stefsietz created the application via Unity and Json files which store the parameters and weights for the model that gets created in Unity.

3 METHODOLOGY

In developing the application, Unity is the main application. In order for the project to be compatible with virtual reality headsets, downloading packages and tool kits are necessary to have the game running properly. In the XR toolkit, inside toolkit, one needs to enable XR windows and Oculus compatibility in order to have a three-dimensional environment. In this case, the Android extension has to be downloaded from Unity's built-in modules. The reason for using an Android extension in this project is for the Meta Quest 2, which is classified as an Android in Unity. Barracuda is another package necessary in order for the ONNX file to work in this application. Once the necessary applications are downloaded properly, some settings need to be changed and modified for the application to run fine in the Meta Quest.

3.1 Convolutional Neural Network Scene

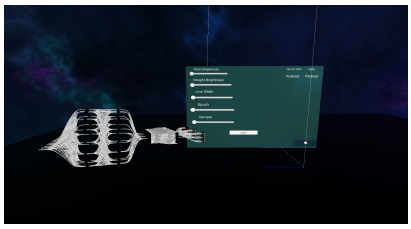

Figure 1: Demo of CNN visuals

Developing the CNN scene in Unity was built with the help of Stefsietz, who, with the help of his documentation and code, I was able to replicate and modify to fit the current version of Unity compared to the version of Unity Stefsietz used. Once the assets and scripts were debugged and re-scripted to fit in the current version of Unity, developing the environment was next. Using an action-based and a VR camera controller was needed to have the project running. At the start of the level, the model shown is the base model without any inputs, weights, or desired outputs. On the right of the player, the user interface elements will be placed that are responsive to the controllers. The user interface elements have a separate tab where

the actual output and predicted output are displayed. There is a slider on the other side where the user can change the parameters and visuals of the model. Additionally, there is a load button that allows the user to load up their own model and .json files to change the base model to their desired output. The model created from loading your own model is not as interactive as the base model. The model is still able to rotate and scale but can not be held on by the user. The rotation and scaling are made possible through an input system that reads the data the controller or device sends whenever a button, stick or touch registration is detected.

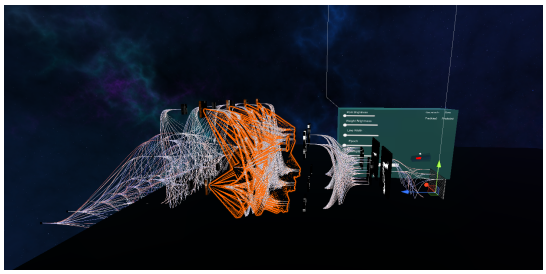

Figure 2: Demo of CNN visuals 2

3.2 Transformer Scene

Creating the Transformer model was a tedious task. I was able to find a basic tutorial[you 2021] on transformers and was able to build simple visual transformers that take photos and classify them. Once the model was created, simple training was done where I ran a few images into the model. In a batch of 100, only 60% of the batch was used in the model. Once the model was trained, it was exported using an ONNX file. ONNX file allows the model and its parameters to be transferred in the same file.

Figure 3: Demo of Transformer

After finding a way to convert the ONNX file to a JSON file, it should have been easy to implement to the CNN scene to have the Transformer model visually present there; however, the code did not allow the parameters, weights, and inputs from the transformer JSON file. In order to have a scene dedicated to the transformer model, a small interactive scene was created. This scene asks the user to drag a picture from the photo wall and place it on the photo holder. Here the program receives the image and sets it up in the ONNX model, which contains the parameters, weights, and inputs. In this visualization scene, the user can use the keyboard or the button to initiate the classification process, and the predicted value will be displayed on the user interface block. To have the ONNX file run in Unity and update during run-time, Barracuda is needed to allow the ONNX model to work. Barracuda[Unity [n. d.]], as stated before, is a package created by Unity to properly implement machine learning models into Unity. Barracuda will help show that the Transformer model does work and function properly. Once the package is implemented, and the ONNX file is able to run in Unity, now the model must be loaded from ONNX to Unity.

The code needed to load and run the model in run-time was found on the forum page, and other information was needed for the ONNX file to work properly and other tips for a better implementation of the model.

Figure 4: Demo of Transformer 2

4 CONCLUSIONS AND FUTURE WORK

The process of creating and developing visual representations of neural networks specifically a transformer model was a good challenge. Learning and understanding a deep learning model like a transformer helped me increase my knowledge about deep learning and the marvels it can do when trained correctly. The process of developing in unity would have been difficult and tedious in developing a transformer model. There are features I would have liked to add and make possible but due to time and limited understanding of virtual reality and deep learning, some of these features were not possible to make. Building the Transformer model helped me understand more and more about deep learning and what it takes to build a simple yet powerful model. For the CNN Scene, the process was a little more since the model originally was in an old Unity format, there was some cleaning up needed to do, some adjustments and changes in order to have the application running and interactive.

Future modifications that can be added or done is having the Barracuda scene functional with a working model. The model will behave similarly to the Stefsietz model. In addition to a working model, a model that allows the user to see the weights, inputs, and outputs will be displayed as a pop-up when a player looks at a specific spot in the model.

ACKNOWLEDGMENTS

This work was supported in part by a grant from the National Science Foundation, Research Experience for Undergraduates program (Award No. 2050532, Principal Investigator - Oyewole Oyekoya)

REFERENCES

[1] 2021. *Vision Transformer in Pytroch.* YouTube. https://www.youtube.com/watch?v=ovB0ddFtzzA&feature=emb_logo
[2] Stefan Sietzen. [n. d.]. *CNN Visualization Tool - The conception and implementation process of the VIS2 CNN project.* Retrieved October 1, 2022 from http://visuality.at/vis2/detail.html
[3] Stefan Sietzen. 2022. *Visual Analytics for Convolutional Neural Network Robustness.* Ph. D. Dissertation. Wien.
[4] Unity. [n. d.]. Introduction to Barracuda. https://docs.unity3d.com/Packages/com.unity.barracuda@1.0/manual/index.html