VR-ANN: Visualization of Artificial Neural Network Models in Virtual Reality

Roy Oberhauser[0000-0002-7606-8226]
Computer Science Dept.
Aalen University
Aalen, Germany
e-mail: roy.oberhauser@hs-aalen.de

Abstract – **Artificial Neural Networks (ANNs or NNs) are used in Deep Learning (DL), a subset of Machine Learning (ML). And yet, especially to novices or infrequent users, ANNs can seem abstract and mathematical, and not readily accessible and understandable. Furthermore, a model's configuration and output results may not be comprehensible and obvious. To make ANN models more accessible and support comprehension and analysis even for large models, this paper contributes our VR-ANN solution concept for immersive ANN visualization in Virtual Reality (VR). Its feasibility is demonstrated with a prototype, while a case-based evaluation provides insights into its capabilities and potential for supporting ANN model building, comprehension, analysis, and collaboration.**

Keywords – artificial neural networks; visualization; virtual reality; deep learning; machine learning.

I. INTRODUCTION

Machine Learning (ML) is a subset of Artificial Intelligence (AI) that focuses on having machines learn from data, improving their performance over time without reprogramming [1]. A learning algorithm can optimize its model's parameters to improve its performance, which can improve pattern detection, predictions, decisions, etc. Various models can be applied in the area of ML. Artificial NNs (ANNs), referred to as just Neural Networks (NNs) in this paper, are inspired by biological NNs and consist of nodes (referred to as artificial neurons) connected via weighted links or edges (i.e., synapses), with other nodes and are aggregated into layers. Numbers are used to represent signals, while a node's activation function determines a neuron's output based on its inputs and their associated link weights. Between the input and output layer are intermediate (hidden) layers. Dense layers are fully-connected to the preceding layer. Dropout layers address overfitting by randomly setting a fraction of the input units to that layer to 0 during training. Hidden layers are any that are neither input nor output layers. Deep Learning (DL) is a branch of ML utilizing deep NNs having multiple hidden layers. Convolutional Neural Networks (CNNs) are a type of DL network especially relevant for image processing. Feedforward NNs (FNNs) are a type of NN where all neurons are fully connected to the next layer with unidirectional information flow. Recurrent NNs (RNNs) are a type of NN for processing sequential data where its order matters.

Visualization of NN models can support comprehension, analysis, and learning, and while various tools support 2D, there has been relatively little investigation into the potential offered by Virtual Reality (VR). To address a comprehensive visualization of NN models, this paper proposes and investigates an immersive VR experience. In prior work, we investigated the application of VR to various other areas. A selection of our prior VR-related contributions include: in the Software Engineering (SE) space, VR-SDLC [2] models development lifecycles, VR-Git [3] models Git repositories, VR-DevOps [4] models Continuous Development pipelines, VR-SBOM [5] models Software Bill of Materials (SBOM) and software supply chains, and VR-EA+TCK [6] and VR-EvoEA+BP [7] exemplify enterprise modeling and business processes. This paper contributes a solution concept towards immersive visualization of ANNs in VR. A prototype demonstrates its feasibility, while a case-based evaluation provides insights into its capabilities and potential for supporting ANN model building, comprehension, analysis, and collaboration.

The structure of this paper is as follows: the next section discusses related work. Section 3 describes our solution. Section 4 presents our realization and is followed by our evaluation in Section 5. And finally, a conclusion is provided.

II. RELATED WORK

In their survey of the application of XR to AI, Hirzle et al. [8] screened 2619 publications (2017-2021) and reviewed 311 in depth. They found only seven papers that applied XR to AI problems (2.3%), five of which visualize AI methods for immersive analytics, or to improve the understanding of neural networks for non-expert users by visualizing them in VR (the rest are discussed below). The authors state XR "methods are promising to facilitate the interaction with neural networks for novices." The 2025 survey by Yim and Su [9] of K-12 learning tools for AI examined 46 papers, but and makes no mention any XR/VR tool. The 2021 survey by Reiners et al. [10] identified 36 papers that combined XR and AI, of which only two were non-domain-specific and related to visualizing DL in XR (discussed below). The survey on AI in VR by Inkarbekov et al. [11] identified a research gap for more user-friendly, intuitive, and adaptable VR tools that can accommodate complex and high-dimensional AI models. From these surveys we conclude that VR-based visualization of AI has not been thoroughly investigated nor readily adopted and remains relatively unexplored.

VR-related NN visualization work includes Bellgardt et al. [12], who depict convolutional ANNs in VR as node-link diagrams by stacking circular layers for a robotic image processing case. InteractML [13] is a node-based tool for creative practitioners to train an ML model for movement

interactions in VR using real-time gesture demonstrations. AIive [14] uses a force-directed graph visualization and sonification to enable VR users to manipulate NN parameters via virtual hands and auditory feedback. Towards non-experts, Meissler et al. [15][16] depict a simplified CNN model in 3D in a closed room virtual environment, with information in 2D anchored to different areas of the room. Schreiber and Bock [17] use the Unreal Engine to display a NN in 3D, whereby connections between layers are not visualized. While VR is mentioned in the title, there is no mention of VR or immersion in the paper, which focuses on 3D. Queck et al. [18] create a virtual room in VR with areas providing CNN information. VR4DL [19] can train and test CNNs with a focus on biomedical image classification; it is not intended to be generic for arbitrary CNNs, and is tailored to users with little to no knowledge of ML. DeepVisionVR [20] visualizes CNNs in VR with a focus on image processing.

Common 2D NN model visualization tools include (TensorFlow) Deep playground, TensorBoard, Netron, Comet, and neptune.ai.

III. SOLUTION CONCEPT

Our VR-ANN solution concept is shown (in blue) in the ML area relative to our other prior VR solutions in our conceptual map of Figure 1. Our generalized VR Modeling Framework (VR-MF) (detailed in [21]) is the foundation, which provides a domain-independent hypermodeling framework, which addresses the VR aspects of visualization, navigation, interaction, and data integration. Our VR-based solutions specific to the Enterprise Architecture (EA) and Business Process (BP) space (EA & BP) include: VR-EA [21] for mapping EA models to VR, VR-BPMN [22] for BPMN models, VR-EAT for enterprise repository integration, VR-EA+TCK [6] for knowledge and content integration, and VR-EvoEA+BP [7] for EA evolution and business process animation, VR-ProcessMine, and VR-SBOM [5]. Solutions in the SE and Systems Engineering (SysE) areas include: VR-Git [3], VR-GitCity, and VR-GitEvo+CI/CD for git-related solutions, VR-DevOps [4], VR-V&V (Verification and Validation), VR-TestCoverage, VR-SDLC [2], VR-ISA for informed software architectures, and VR-UML and VR-SysML for software and systems modeling.

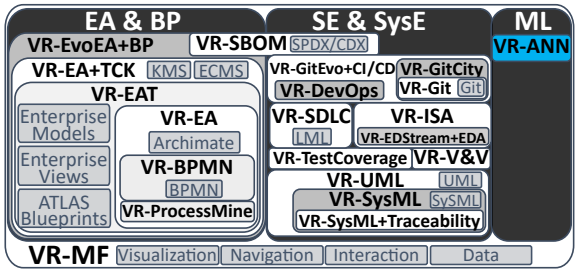

Figure 1. Conceptual map of our various published VR solution concepts with VR-ANN highlighted in blue.

A. Visualization in VR

To better delimit the context of a NN model in a multi-NN VR space, each NN model is visualized within an outlined 3D transparent Boundary Box (BB) placed on a *hyperplane*. This enables further additional portfolio of NNs to be visualized contemporaneously. Contained within the NN BB are outlined transparent 3D BBs, each representing a NN layer, which vertically delineates a set of spherical nodes (neurons). Lines are used as connectors to indicate which nodes are connected between layers.

B. Navigation in VR

Dual navigation modes are incorporated in our solution: default gliding controls for fly-through VR, while teleporting instantly places the camera at a selected position in space. While teleporting can be potentially disconcerting, it may reduce the likelihood of VR sickness.

C. Interaction in VR

User-element interaction is supported primarily through VR controllers and the incorporation of a VR-Tablet. The VR-Tablet provides detailed context-specific element information. Various tabs in the VR-Tablet enable support for loading, training, executing, or configuring NN models and viewing related graphs. Since for our examples no text entry and keyboard were required, a virtual keyboard was not included. However, our implementation can be readily enhanced with a virtual keyboard for text entry using laser pointer key selection, as demonstrated in our other VR solutions. A small control sphere is placed as an affordance at the bottom front corner on the boundary of the hyperplane for dragging, collapsing (to reduce visual clutter), or expanding a NN BB.

IV. REALIZATION

For our prototype realization, the VR visualization aspects were implemented using Unity in C#, referred to as our frontend, as shown in Figure 2. It connects via a Socket to our backend Data Hub, which is based on the hexagonal (or ports-and-adapters) design pattern and is implemented in Python. It integrates and stores all data via PyMongo to a MongoDB database in JSON/BSON format.

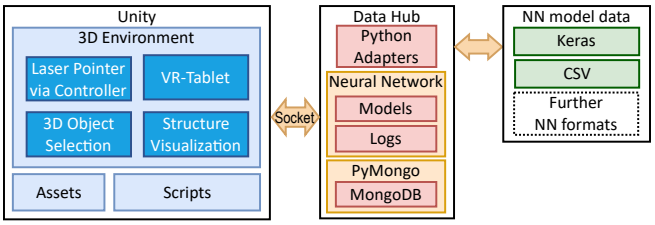

Figure 2. VR-ANN logical architecture.

NN model support is implemented within the Data Hub in Python. The Keras API was used as a high-level DL API primarily due to its popularity and flexibility, since it supports multiple backends, such as TensorFlow, PyTorch, OpenVINO, and JAX (Python library for high-performance ML). Initially for prototyping, the Sequential model with various layer types placed in sequence is supported, but support for additional models can be readily added.

Data can be imported from or exported to the common keras format (.keras extension), which is a zip archive containing JSON-based configuration, H5-based state file

containing layers and weights, model weights, and metadata. Additionally, the CSV file format is supported. Further formats can be readily supported via adapters. The models can either be pretrained, loaded, and executed in VR (load and execute), or can be (re)configured and trained in a VR session via our Model Builder support mode.

To exemplify the internal interaction, a sequence diagram for a VR-centric training session is shown in Figure 3. Initially, the list of available NN projects is retrieved from the Data Hub. If the user creates a new model and starts a training session, then the layer data is sent to the Data Hub, the training inputs are retrieved, and a model is created and trained via the Keras API, training values are logged, and the model is saved or exported. The acquired layer data is returned to Unity via the socket, and is visualized as a model, which can be further analyzed.

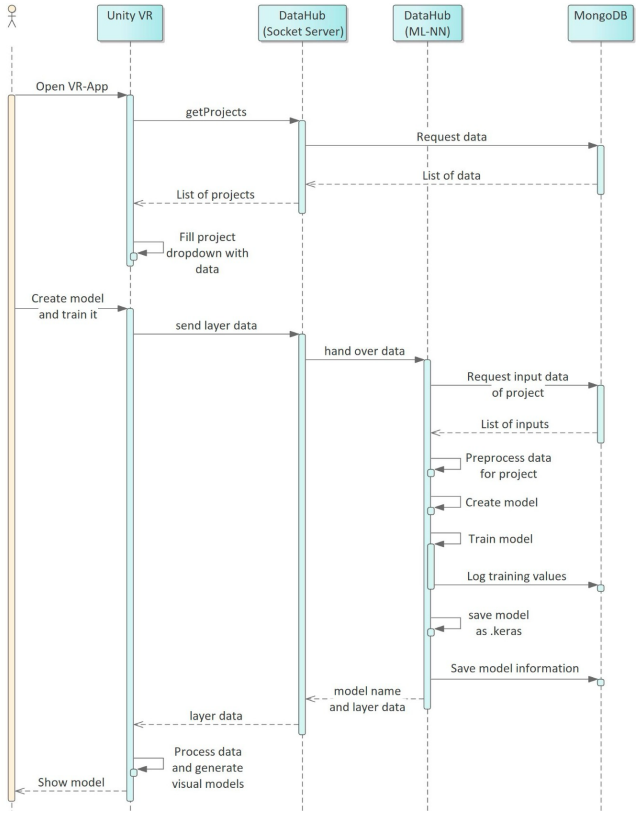

Figure 3. VR-ANN sequence diagram for a VR-centric training session.

To support VR interaction, we implemented the VR-Tablet as shown in the bottom right in Figure 4. The tab groups Load, Training, Execution, Display options, and Graphs are shown, as is the scrollbar. Here, under the selected Display options tab, the epoch slider is shown as well as options to show or color connections. Also shown in this figure, the model layer BBs top side provide layer information (number and type) and metrics (number of neurons) and containing spheres as neurons evenly spaced along a single vertical plane with slots for the next best-fitting square matrix, filled from the bottom left (upper right may have empty slots), while dropout layers are depicted as an opaque slab. Connectors are shown by default in blue; the diameter of the lines indicate the relative weighting to the next layer.

In execution, the most active (top five) routes as nodes and connectors are colored (darker to lighter) green. Connectors can optionally be colored (iterated list of 32 colors) to more easily follow a connector between nodes.

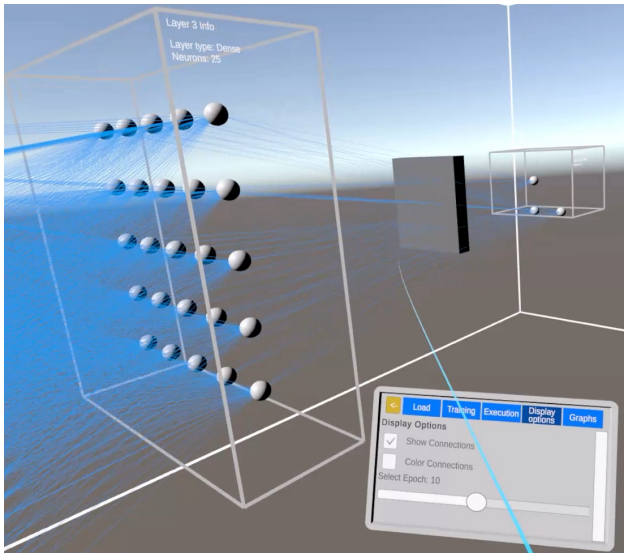

Figure 4. BBs and VR-Tablet showing tab groups and input options.

To support Model Builder mode, opaque 3D boxes represent layers ordered from left to right, each of which offers appropriate options based on type, as shown in Figure 5. To avoid requiring text input and simplify interaction, the default values can be adjusted with plus/minus buttons and dropdown lists offer selection options. Layers can be removed by an X button at the top right, and new layer types can be inserted via the VR-Tablet. The ordering of the layers can be adjusted by dragging the boxes. RNN support was not yet implemented. All applicable parameters can be selected in the VR-Tablet, including epochs, learning rate, optimizer, loss function, etc.

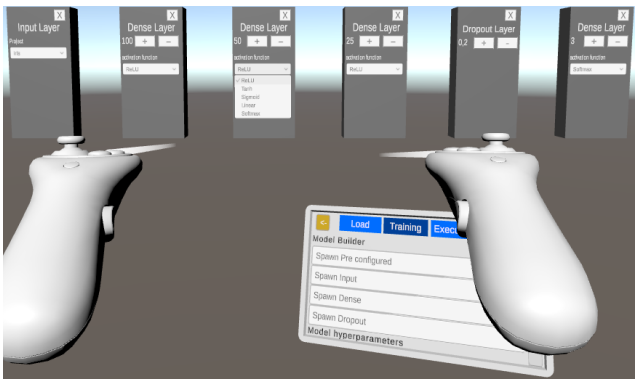

Figure 5. Model Builder for defining and configuring a NN.

Support for spawning two graph types was implemented: a loss graph and an accuracy graph, each of which also offer a corner sphere affordance for flexible placement or collapsing.

V. EVALUATION

The evaluation of our VR-ANN solution concept is based on the design science method and principles [24], in particular a viable artifact, problem relevance, and design evaluation (utility, quality, efficacy). A case study is used based on the following scenarios: comparison to 2D visualization, build and train support, analysis support, and scaling support.

A. 2D vs. VR-ANN Visualization

To visually compare a typical NN tool's 2D visual representation of a NN to VR-ANN, an FNN is shown in the TensorFlow Playground in Figure 6. The equivalent NN in VR-ANN is shown in Figure 7. The VR-ANN model is immersively accessible, and can be readily investigated and analyzed. Much of the information seen in the 2D playground is available in VR by interacting with the VR-Tablet or elements (layer, nodes). Given many connections in 2D, we contend it to be more straightforward to immersively follow a connector in VR separated spatially in 3D space.

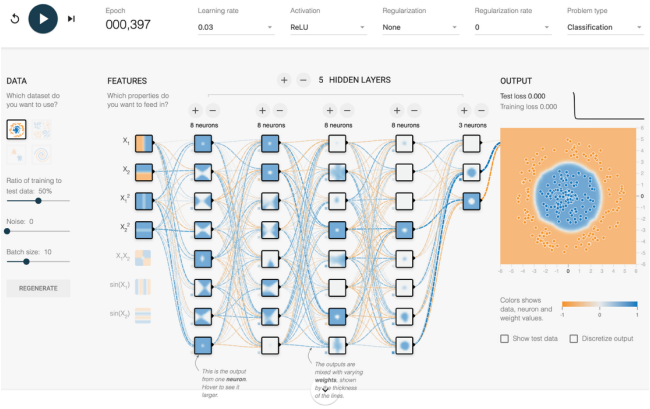

Figure 6. Screenshot of a NN model in TensorFlow playground [25].

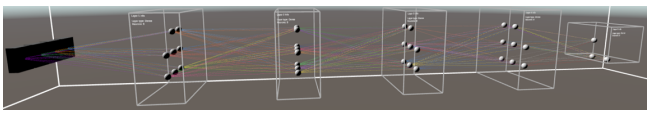

Figure 7. Equivalent NN in VR-ANN containing layers as BBs of connected nodes.

B. Build and Train NN Model Support

Support for building NNs was shown in Figure 5. Via the VR-Tablet, a preconfigured existing model can be loaded, or additional layer types flexibly. Once trained, the model

C. Analysis Support

To demonstrate analysis support, we utilize an Iris flower dataset available on Kaggle [26]. For this, 50 samples each of three Iris species (Setosa, Versicolor, Virginica) are classified in the output based on four properties: SepalLengthCm, SepalWidthCm, PetalLengthCm, PetalWidthCm. After building and training, the entire model with all layer types and neurons is visualized within a BB as was shown in Figure 7. This helps in comprehension of the total number of layers and their type, size, and ordering. Metrics are provided on each layer BB as seen in the upper right of Figure 8. As shown, the connections between neurons can be optionally colored to help differentiate them when immersively following a connection.

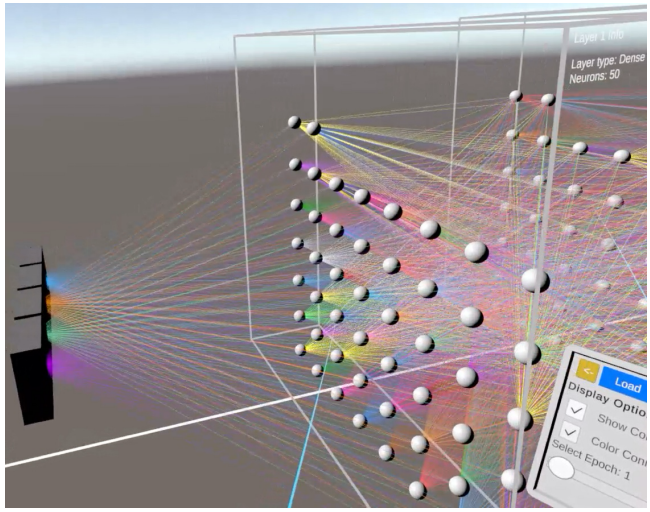

Figure 8. Colored connections between neurons.

To support more detailed analysis, selecting a single node will cause only connections related to that node to be shown and all others to be hidden, as seen in Figure 9. Here, the weight values of all the connected input nodes are displayed above each input node in the previous layer, metadata about the selected node is shown in a panel above that node, such as bias and output values, and output connections are visible.

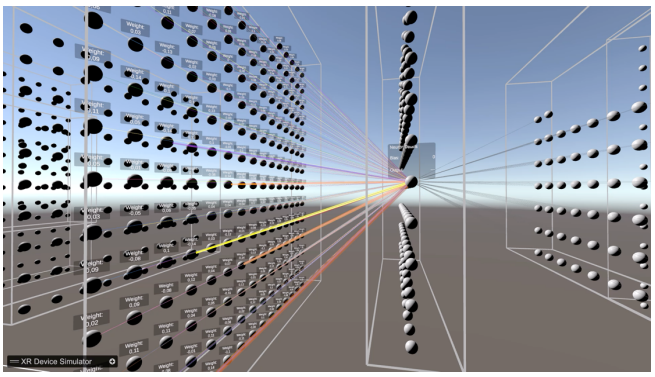

Figure 9. Node selection shows details related to that node.

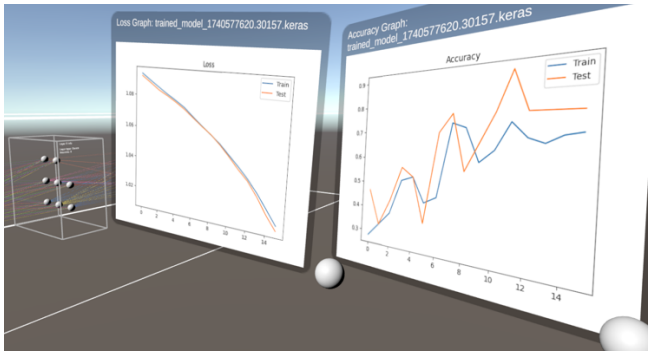

Figure 10. Loss (center) and Accuracy (right) graphs relative to epochs.

Beyond the data available in the VR-Tablet, rather than just have the graphs in the VR-Tablet, a loss graph and/or accuracy graph can be spawned (and moved or collapsed via the affordances) and retained with the model context as shown in Figure 10. Thus, when multiple similar models or slightly different configurations are loaded in VR, one can more readily determine the differences.

The inputs can be adjusted as shown in Figure 11 (left), and via the VR-Tablet the model executed. The top classification result for this data set can be seen in Figure 11 (right). After NN execution, the top five most frequent activation paths (pathways, routes) are indicated via (darker-to-lighter) green nodes and connectors as shown in Figure 12.

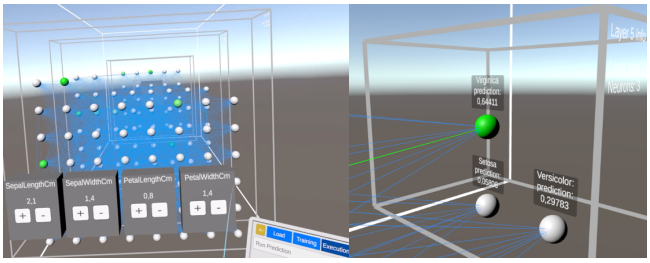

Figure 11. Execution on inputs (left) and result in the output layer (right).

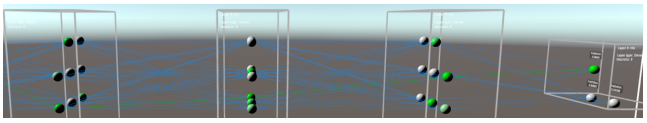

Figure 12. Overall top primary active routes (green nodes and connectors).

D. Scaling Support

To demonstrate the scaling ability of VR to support large NN models, Figure 13 depicts a mid-sized model that consists of ten (8 hidden) layers with 432 neurons counts (4, 100, 50, 25, 100, 50, 25, 50, 25, dropout, 3). Figure 14 shows a large model consisting of 982 neurons across ten (8 hidden) layers (4, 100, 200, 25, 100, 200, 100, 50, 200, 3) with the connectors colored. Our principle is to initially depict a model's reality with its inherent complexity. However, as shown in the analysis case, via immersion, selection of an element of interest, display filtering such as turning off connectors, visual overload can be addressed. Based on the stakeholder's interest and intentionality, comprehension or issue analysis for large models can be supported and stakeholder collaboration opportunities with a common immersive model utilized.

VI. Conclusion

This paper described our VR-ANN contribution, a solution concept towards immersive visualization of ANNs in VR. Our prototype demonstrates its feasibility. The case-based evaluation provides insights into its capabilities and potential for immersively supporting the comprehension, building, configuring, training, and analysis of ANN models and related stakeholder collaboration. The scaling case showcased its ability to immersively depict large models, which could benefit more advanced users when the models become much larger than what current 2D models can readily display, or when investigating anomalies or issues. Future work includes RNN and CNN support, additional framework and format support, and a comprehensive empirical study.

Acknowledgment

The author would like to thank Daniel Godeck for his assistance with the design, implementation, and screenshots.

References

[1] S. Russell and P. Norvig, Artificial Intelligence: A Modern Approach, Global Edition, 4th ed. Pearson Education, 2021. ISBN 978-0-1346-1099-3.

[2] R. Oberhauser, "VR-SDLC: A Context-Enhanced Life Cycle Visualization of Software-or-Systems Development in Virtual Reality," In: Business Modeling and Software Design (BMSD 2024), LNBIP, vol 523, Springer, Cham, 2024, pp. 112-129, https://doi.org/10.1007/978-3-031-64073-5_8.

[3] R. Oberhauser, "VR-Git: Git Repository Visualization and Immersion in Virtual Reality," 17th Int'l Conf. on Software Engineering Advances (ICSEA 2022), IARIA, 2022, pp. 9-14.

[4] R. Oberhauser, "VR-DevOps: Visualizing and Interacting with DevOps Pipelines in Virtual Reality," Nineteenth International Conference on Software Engineering Advances (ICSEA 2024), IARIA, 2024, pp. 43-48.

[5] R. Oberhauser, "VR-SBOM: Visualization of Software Bill of Materials and Software Supply Chains in Virtual Reality," In: Business Modeling and Software Design (BMSD 2025), LNBIP, vol 559, Springer, Cham, 2025, pp. 52-70, https://doi.org/10.1007/978-3-031-98033-6_4.

[6] R. Oberhauser, M. Baehre, and P. Sousa, "VR-EA+TCK: Visualizing Enterprise Architecture, Content, and Knowledge in Virtual Reality," In: Business Modeling and Software Design (BMSD 2022), LNBIP, vol 453, Springer, 2022, pp. 122-140. https://doi.org/10.1007/978-3-031-11510-3_8.

[7] R. Oberhauser, M. Baehre, and P. Sousa, "VR-EvoEA+BP: Using Virtual Reality to Visualize Enterprise Context Dynamics Related to Enterprise Evolution and Business Processes," In: Business Modeling and Software Design (BMSD 2023), LNBIP, vol 483, Springer, 2023, pp. 110-128, https://doi.org/10.1007/978-3-031-36757-1_7.

[8] T. Hirzle et al., "When XR and AI Meet - A Scoping Review on Extended Reality and Artificial Intelligence," In: Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23), ACM, Article 730, 2023, pp. 1–45. https://doi.org/10.1145/3544548.3581072.

[9] I. Yim and J. Su, "Artificial intelligence (AI) learning tools in K-12 education: A scoping review," Journal on Computers in Education, Vol. 12, pp. 93–131, 2025, https://doi.org/10.1007/s40692-023-00304-9.

[10] D. Reiners, M. R. Davahli, W. Karwowski, and C. Cruz-Neira, "The combination of artificial intelligence and extended reality: A systematic review," Frontiers in Virtual Reality, 2, 721933, 2021, doi: 10.3389/frvir.2021.721933

[11] M. Inkarbekov, R. Monahan, and B. A. Pearlmutter, "Visualization of ai systems in virtual reality: A comprehensive review," arXiv preprint, 2023, arXiv:2306.15545.

[12] M. Bellgardt, C. Scheiderer and T. W. Kuhlen, "An Immersive Node-Link Visualization of Artificial Neural Networks for Machine Learning Experts," 2020 IEEE International Conf. on Artificial Intelligence and Virtual Reality (AIVR), IEEE, 2020, pp. 33-36, doi: 10.1109/AIVR50618.2020.00015.

[13] C. Hilton et al., "InteractML: Making machine learning accessible for creative practitioners working with movement interaction in immersive media," In: Proceedings of the 27th ACM Symposium on Virtual Reality Software and Technology (VRST '21), ACM, Article 23, 2021, pp. 1–10. https://doi.org/10.1145/3489849.3489879.

[14] Z. Lyu, J. Li and B. Wang, "AIive: Interactive Visualization and Sonification of Neural Networks in Virtual Reality," 2021 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), IEEE, 2021, pp. 251-255, doi: 10.1109/AIVR52153.2021.00057.

[15] N. Meissler, A. Wohlan, N. Hochgeschwender, and A. Schreiber, "Using Visualization of Convolutional Neural Networks in Virtual Reality for Machine Learning Newcomers," 2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR), IEEE, 2019, pp. 152-1526, doi: 10.1109/AIVR46125.2019.00031.

[16] A. Wohlan, N. Hochgeschwender, and N. Meissler, "Visualizing Convolutional Neural Networks with Virtual Reality," In: Proc. 25th ACM Symposium on Virtual Reality Software and Technology (VRST '19). ACM, Article 100, pp. 1–2, 2019, https://doi.org/10.1145/3359996.3364817

[17] A. Schreiber and M. Bock, "Visualization and exploration of deep learning networks in 3D and virtual reality," In: Human-Computer Interaction International 2019 (HCII 2019) Posters, Proc. 21st International Conference on HCI (HCII 2019), CCIS, Springer International Publishing, 2019, pp. 206-211.

[18] D. Queck, A. Wohlan, and A. Schreiber, „Neural Network Visualization in Virtual Reality: A Use Case Analysis and Implementation," In: Human Interface and the Management of Information: Visual and Information Design (HCII 2022), LNCS, Springer, 2022, vol 13305, pp. 384-397, https://doi.org/10.1007/978-3-031-06424-1_28.

[19] K. VanHorn and M. C. Çobanoğlu, "Democratizing AI in biomedical image classification using virtual reality," Virtual Reality, 26(1), pp. 159-171, 2021, https://doi.org/10.1007/s10055-021-00550-1

[20] C. Linse, A. Hammam, and T. Martinetz, "A walk in the black-box: 3D visualization of large neural networks in virtual reality," Neural Computing and Applications, vol. 34, no. 23, pp. 21237-21252, 2022.

[21] R. Oberhauser and C. Pogolski, "VR-EA: Virtual Reality Visualization of Enterprise Architecture Models with ArchiMate and BPMN," In: Business Modeling and Software Design (BMSD 2019), LNBIP, vol. 356, Springer, Cham, 2019, pp. 170-187, https://doi.org/10.1007/978-3-030-24854-3_11.

[22] R. Oberhauser, C. Pogolski, and A. Matic, "VR-BPMN: Visualizing BPMN models in Virtual Reality," In: Shishkov, B. (ed.) Business Modeling and Software Design (BMSD 2018), LNBIP, vol. 319, Springer, 2018, pp. 83–97, https://doi.org/10.1007/978-3-319-94214-8_6.

[23] R. Oberhauser, "VR-GitCity: Immersively Visualizing Git Repository Evolution Using a City Metaphor in Virtual Reality," International Journal on Advances in Software, 16 (3 & 4), 2023, pp. 141-150.

[24] A.R. Hevner, S.T. March, J. Park, and S. Ram, "Design science in information systems research," MIS Quarterly, 28(1), 2004, pp. 75-105.

[25] [Online]. Available from: https://playground.tensorflow.org 2025.08.01

[26] [Online]. Available from: https://www.kaggle.com/code/ranjeetjain3/visualization-machine-learning-deep-learning 2025.08.01

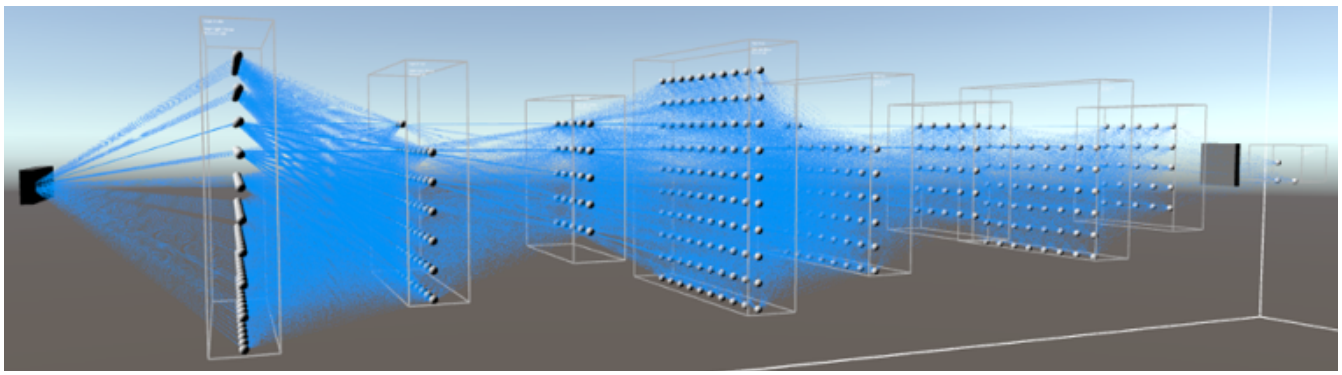

Figure 13. Mid-sized model consisting of 8 hidden layers and 432 neurons.

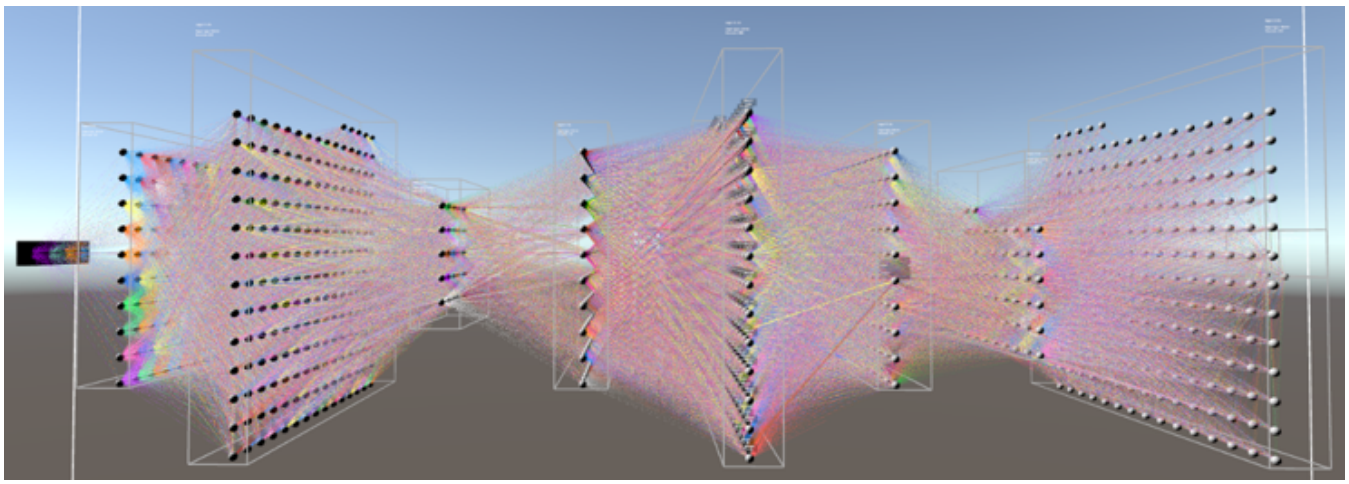

Figure 14. Large model (8 hidden layers and 982 neurons) with colored connections.