Oregon State University

Portfolio

Master's in Computer Science
2023-2025

Name: Sanketh Karuturi
ONID: 934482940
Email: karutusa@oregonstate.edu

TABLE OF CONTENTS

SECTION 1: KNOWLEDGE OF CORE CONCEPTS.

1.1 Implementing a Neural Network for Digit Identification

Building a neural network from scratch to identify handwritten digits was a transformative experience that deepened my understanding of machine learning fundamentals. The neural network model in this project was trained on a subset of the MNIST dataset, which is widely used in machine learning research. The Fig 1 shown below is a visualization of some example images from this dataset:

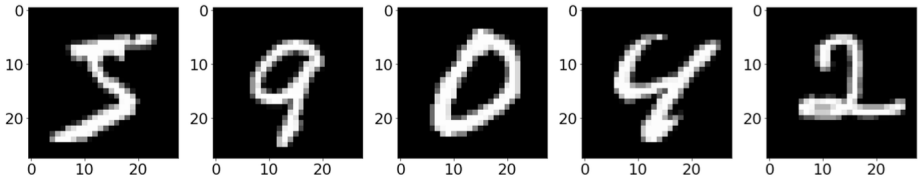

Fig 1. Plot of a Subset of Images From the MNIST Dataset

Each digit was represented as a 28×28 grayscale image, which I flattened into a 784-dimensional feature vector. With 10 possible class labels (digits 0–9), my goal was to train a feed-forward neural network that could accurately classify new handwritten digits based on the predicted value. I structured the network with multiple hidden layers, utilizing weight matrices and bias vectors to transform inputs through linear operations and non-linear activations. The final layer employed a softmax function shown in Equation 1 to convert raw scores into probability distributions over the 10 classes, where each pj|x represents the predicted probability for class j.

$$p_{j|x} = \frac{e^{z_{3j}}}{\sum_{c=0}^{9} e^{z_{3c}}} \tag{1}$$

By structuring the network this way, I ensured that it could learn complex patterns and generalize well to unseen data. To optimize the network, I implemented backpropagation to compute gradients and used stochastic gradient descent (SGD) for weight updates. The loss function chosen was cross-entropy loss, The objective is to measure the model's error by computing the negative log-likelihood, which quantifies how well the predicted probabilities align with the actual labels. For a given example, this is expressed as:

$$\mathcal{L}(\theta) = -\sum_{i=1}^{N} \log p_{y_i|x_i} \tag{2}$$

where the loss sums over all N examples in the dataset. This function was minimized using SGD to iteratively adjust the model parameters θ, thereby improving classification accuracy by increasing the probability of the correct class for each input. The neural network training process involved forward propagation to compute outputs and backward propagation to adjust weights based on errors. Initially, I experimented with different activation functions, testing Sigmoid and ReLU. The Sigmoid function suffered from the vanishing gradient problem as shown in Fig 2, which slowed learning, especially in deeper networks. Switching to ReLU activation shown in Fig 5 significantly improved performance by enabling efficient gradient flow and faster convergence.

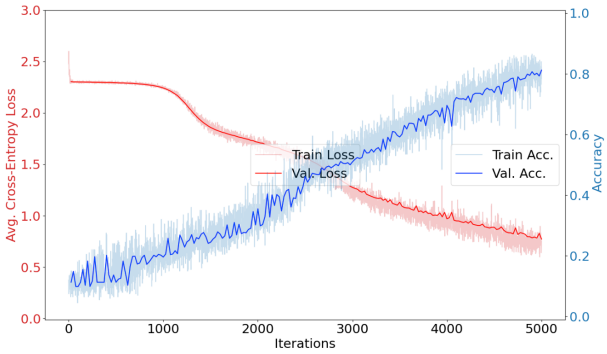

Fig 2. 5-layer with Sigmoid Activation with 0.1 step size

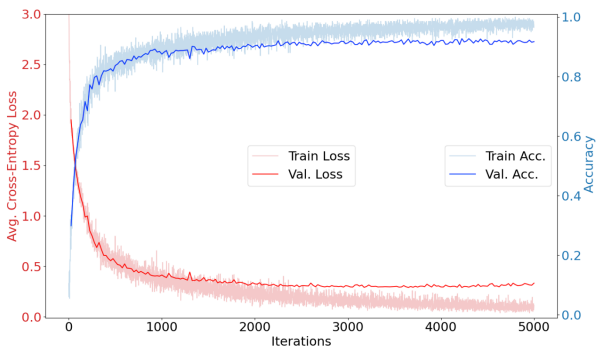

Fig 3. 5-layer with ReLU Activation

I analyzed the effects of different hyperparameter choices, including batch size and learning rate, which significantly impacted the model's ability to learn efficiently. The final model, trained with 6 layers, 256 neurons per layer, and ReLU activations, achieved an impressive 98.8% classification accuracy. Once the model was fine-tuned, I generated predictions for the test set and submitted them to a Kaggle competition, where my classifier successfully outperformed the Teaching Assistant (TA) model on the private leaderboard, and I received extra credit points on this project. Link to my Neural network code:
https://github.com/Sankethprasad09/Machine_Learning/tree/main/Neural%20Networks

This project reinforced key machine learning concepts and provided valuable hands-on experience in designing, training, and optimizing neural networks. Through multiple iterations, I gained insights into the impact of activation functions, the importance of hyperparameter tuning, and the role of initialization in model performance. Beyond implementation, I also learned how to visualize loss curves and interpret training dynamics, which are essential for debugging deep learning models. This experience has solidified my confidence in building deep learning models from scratch and has prepared me to tackle more advanced AI challenges in my career.

1.2 Monte Carlo Simulation Using CUDA for Performance Optimization

My understanding of parallel computing was significantly enhanced through the implementation of a Monte Carlo simulation using CUDA. Monte Carlo methods are widely used in computational sciences to solve problems through repeated random sampling, and leveraging GPU parallelism can drastically improve their efficiency. This project required simulating a physics-based scenario in which a golf ball starts moving from the top of a ski jump as shown in Fig 4. at an initial height of BeforeY. It rolls down until it reaches a height of AfterY, where it launches horizontally. The ball then travels through the air until it contacts the ground. A hole with a radius of RADIUS is positioned at a horizontal distance of DistX from the ski jump's end. Will the golf ball land inside the hole?

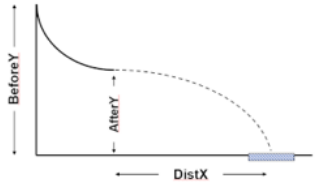

Fig 4. Golf Ball Motion from a Ski Jump to Target

Unlike traditional CPU-based Monte Carlo simulations, where iterations are executed sequentially, CUDA enables parallel execution by distributing computation across thousands of threads. The key to efficiency in this project was optimizing the number of threads per block (BLOCKSIZE) and the number of total trials (NUMTRIALS) to achieve high throughput. My implementation tested BLOCKSIZE values of 8, 32, 64, 128, and 256, combined with NUMTRIALS values ranging from 1024 to 2,097,152, running on an NVIDIA DGX system.
The core of the Monte Carlo simulation involved calculating the horizontal displacement of the ball upon landing. Using fundamental kinematic equations, the horizontal distance traveled by the ball after leaving the ski jump was determined by Equation 3.

$$vx = \sqrt{2.* GRAVITY * (beforey - aftery)}$$

$$t = \sqrt{\frac{2.* aftery}{GRAVITY}}$$

$$dx = vx * t \quad \textbf{(3)}$$

Where v is the horizontal velocity component and t is the time of flight. Given the uncertainties in measurements, these calculations were performed repeatedly with randomized inputs following normal distributions. Each GPU thread was responsible for a single Monte Carlo trial, where it computed whether the ball landed within the hole radius based on the calculation. The output was stored in an array where successes were marked as 1 and failures as 0. After execution, the CPU aggregated these results to compute the final probability. The results showed that

performance increased as BLOCKSIZE grew, peaking at an optimal point before plateauing due to resource contention as shown in Fig 5. The recorded performance, measured in MegaTrials/Second, demonstrated that larger NUMTRIALS values led to linear scaling of performance shown in Fig 6, as more trials leveraged the GPU's computational capacity efficiently. The highest recorded performance was 21,707.85 MegaTrials/Second at BLOCKSIZE 256 with 2,097,152 trials.

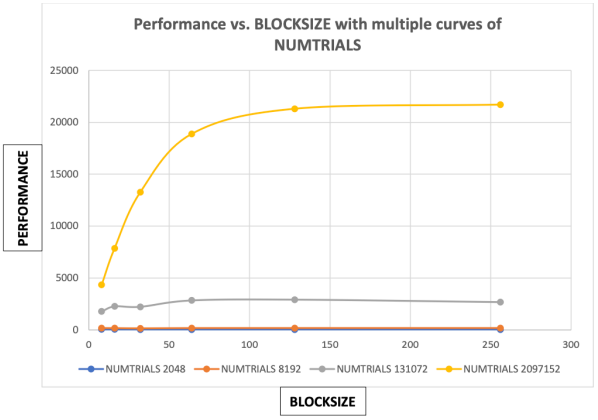

Fig 5. Performance vs BLOCKSIZE graph with NUMTRIALS curves

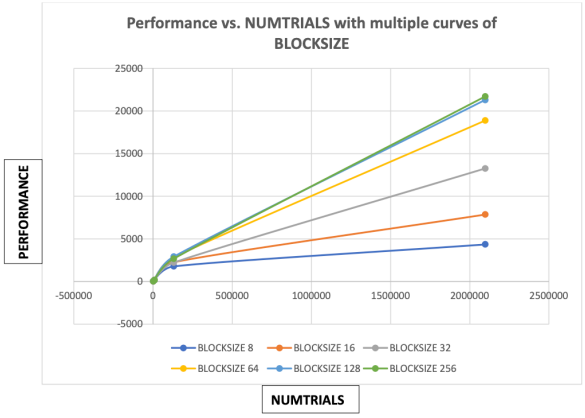

Fig 6. Performance vs NUMTRIALS graph with BLOCKSIZE curves

This project reinforced my understanding of how CUDA enables massive parallel execution, drastically outperforming CPU-based approaches. It also demonstrated key optimization principles, such as balancing thread workload, minimizing memory access bottlenecks, and understanding hardware limitations. These lessons will be invaluable for designing high-performance parallel applications in my career to maximize performance and minimize the computational cost associated with the task.

1.3 Computer Graphics: Shaders and Animation

Throughout my master's program, I have taken three core courses in Computer Graphics, Computer Shaders, and Computer Animation, each contributing significantly to my understanding of visual computing. My coursework and projects have allowed me to explore the full spectrum of computer graphics, from fundamental rendering techniques to advanced shading and animation. My animation project, Alien Soldier's Redemption, provided a hands-on opportunity to integrate these concepts, utilizing shaders, physics simulations, and keyframed animations to create an engaging 3D narrative. One of the key components of this project was shader programming, where I implemented custom vertex and fragment shaders in Blender to enhance the realism of materials and lighting as shown in Fig 7. The Phong shading model was applied and I used Equation 4 to dynamically compute light interactions, ensuring realistic reflections and depth perception.

$$I_p = k_a i_a + \sum_{m \in \text{lights}} \left(k_d (\hat{L_m} \cdot \hat{N}) i_{m,d} + k_s (\hat{R_m} \cdot \hat{V})^{\alpha} i_{m,s} \right) \tag{4}$$

where the ambient, diffuse, and specular components are combined to create natural lighting effects. The monster's armor, for example, featured a custom metallic shader, which responded dynamically to environmental lighting, giving it a reflective and battle-worn appearance.

Fig 7. Lighting and Shader-Driven Animation in a Dynamic Scene

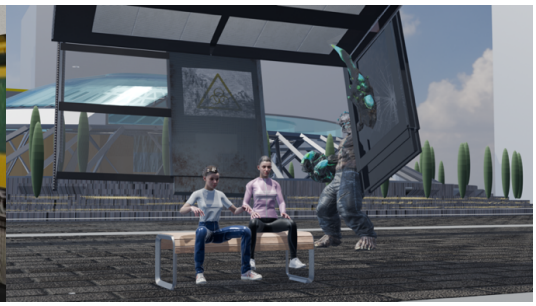

Fig 8. Physics-Based Destruction at the Train Station

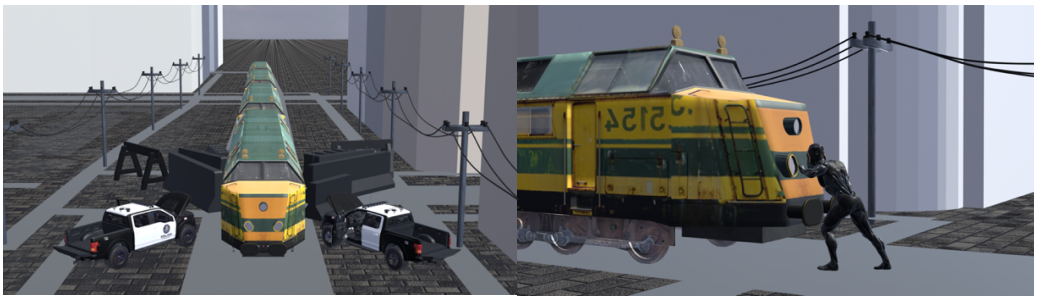

Fig 9. Blender rigid body engine to simulate realistic collisions *Fig 10.* Alien Soldier trying to stop the breakless moving train

In addition to shading, rigid-body physics simulations were used to create destruction effects, particularly when the villain sabotages the train station as shown in Fig 8. The Blender rigid-body engine was employed to generate realistic collisions, ensuring that the barricades scattered naturally upon train's impact as shown in Fig 9. This was complemented by procedural animation techniques, where the graph editor shown in Fig 11 was used to fine-tune object trajectories for smooth transitions and lifelike motion.

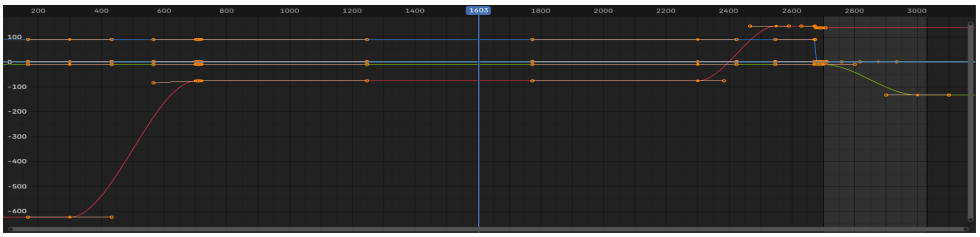

Fig 11. Graph editor in Blender for smooth and lifelike movements.

Character animation played a critical role in conveying emotion and realism. Utilizing the 12 principles of animation, I carefully implemented techniques such as squash and stretch (to exaggerate movement), anticipation (to prepare the audience for action), and follow-through (to create natural motion flow). One example of this was the alien soldier's entry to stop the moving train as shown in Fig 10, where motion curves were adjusted to achieve realistic momentum and eventual slowing down of the train.

The animation workflow also required careful synchronization of camera and character actions. Using Blender's Non-Linear animation editor shown in Fig 12, I crafted smooth dynamic transitions from one character pose to another by accurately adjusting the blend-in value to make pose transitions look very natural. For camera, I used blender's path constraints and interpolation methods for tracking shots that followed key moments, such as the train's out-of-control descent and the alien soldier's intervention. These cinematic techniques ensured that the animation remained engaging and visually compelling.

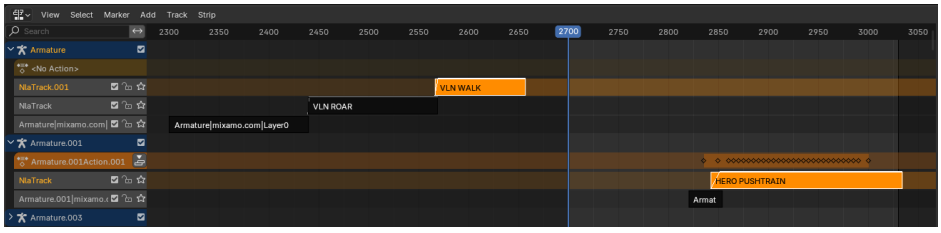

Fig 12. Non-Linear Animation editor for extremely smooth transition from pose-to-pose animation.

This project was a culmination of computer graphics, shader programming, and animation techniques, demonstrating my ability to merge technical precision with artistic storytelling. This experience reinforced my understanding of rendering optimizations, shader physics, and animation sequencing, skills that are crucial for my career in computer graphics and animation.

Link to my animation: https://media.oregonstate.edu/media/t/1_rh8hkmj9

SECTION 2: INTEGRATION OF OTHER DISCIPLINES IN COMPUTER SCIENCE

The ability to integrate knowledge from various other disciplines is essential in computer science, as many real-world solutions require expertise beyond just coding and optimizing algorithms. My graduate studies have provided me with numerous opportunities to integrate knowledge from other disciplines like mathematics and psychology to core areas of computer science, such as computer graphics and human-computer interaction. These interdisciplinary approaches have enabled me to design seamless 3D simulations and user-centered applications, which are critical skills for a career in computer graphics and human-computer interaction.

2.1 Mathematics

Mathematics is a field that is integral to computer graphics, which provides the foundational tools for modeling, transforming, and rendering 3D objects. A comprehensive understanding of 3D rotations was essential for me to accurately depict the object orientations and movements within a virtual environment (Blender). During one of my projects involving 3D Animation, I encountered the issue of "gimbal lock," which is a phenomenon that occurs when the axes of a multi-dimensional mechanism align in a certain way, causing a loss of one degree of freedom of movement. This was evident when animating Optimus' robotic hands shown in Fig 13. To resolve this, I implemented the quaternion-based rotation to systematically adjust and track rotational states.

Fig 13. Gimbal lock issue in my 3D animation project (Joe Rogan Podcast on Mars).

Fig 14. Animation scene after implementing quaternions-based rotations

In 3D space, rotations can be represented using various mathematical formulas. A rotation matrix is a 3×3 transformation matrix with a determinant of 1, which is used to perform rotations in Euclidean space. Each rotation matrix corresponds to a rotation about a specific axis by a given angle. For example, In Equation 5, the rotation matrix Rz(θ) represents a counterclockwise rotation by an angle θ about the z-axis as shown in Fig 15:

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \textbf{(5)}$$

Applying this matrix to a vector v = [x, y, z] T results in a new vector v′, which is the original vector rotated by θ around the z-axis. I have widely used rotation matrices due to their straightforward implementation and computational efficiency.

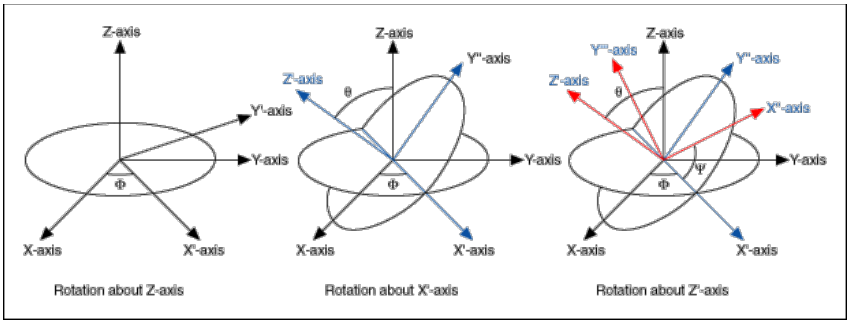

Fig 15. Coordinate rotation with Euler angles ϕ, θ, and ψ (rotation order Z-X-Z) (National Instruments, n.d.)

Based on Euler's rotation theory, a coordinate rotation can be represented using the Euler angles ϕ, θ, and ψ, as shown in Fig 15. These angles describe a rotation by three consecutive rotations about specified axes, typically referred to as roll, pitch, and yaw. A common convention is the Z-X-Z rotation sequence, where an object is first rotated around the z-axis by ϕ, followed by a rotation around the new x-axis by θ, and finally around the new z-axis by ψ. The composite rotation matrix R for this sequence is obtained by multiplying the individual rotation matrices as shown in Equation 6.

$$R = R_z(\psi)R_x(\theta)R_z(\phi) \tag{6}$$

While Euler angles are easy to visualize and very intuitive, they can suffer from gimbal lock as mentioned above. Quaternions, however, avoid gimbal lock by representing rotations as a single entity in four-dimensional space rather than as sequential axis rotations. Instead of applying separate transformations, quaternions use spherical linear interpolation (SLERP), allowing for continuous and smooth rotation without encountering singularities. A quaternion q is defined as:

$$q = w + xi + yj + zk \tag{7}$$

where w, x, y, z are real numbers, and i, j, k are the quaternion units. For rotation purposes, a unit quaternion with a norm of 1 shown in Equation 8 is used, and it can be expressed as:

$$q = \cos\left(\frac{\theta}{2}\right) + \sin\left(\frac{\theta}{2}\right)(xi + yj + zk) \tag{8}$$

Here, θ represents the rotation angle, and [x, y, z] is the unit vector along the axis of rotation. To apply a rotation to a vector v using a quaternion q, the following operation is performed:

$$\mathbf{v}' = q\mathbf{v}q^{-1} \tag{9}$$

By implementing quaternion-based rotation, I significantly improved the animation fluidity of the robotic hands. In Figure 13, the original gimbal lock issue resulted in restricted motion and unnatural rotation artifacts. After transitioning to quaternion-based rotation, the movements were smooth, continuous, and free from singularity issues, as demonstrated in Figure 14. The mathematical difference is also reflected in Figure 15, where the Z-X-Z sequence was replaced by quaternion interpolation, preventing loss of degrees of freedom.

By integrating rotation matrices and quaternions, I successfully controlled the robot orientations in this animation project. This knowledge is critical for developing realistic simulations in robotics, game development, and virtual reality. Mathematics remains a cornerstone of computer graphics, providing precise and efficient solutions for real-world animation challenges.

Link to my animation project: https://media.oregonstate.edu/media/t/1_51tixaeq

While mathematics provides the computational foundation for modeling and simulations, psychology complements this by enabling the design of intuitive and engaging user experiences. Together, these disciplines help create comprehensive solutions for the contemporary real-world challenges.

2.2 Psychology

One of the major challenges in mobile application development is engaging users and it requires the understanding of their psychological tendencies. While I was designing an iOS mobile application “Study Buddy”, I used color psychology shown in Fig 16. to design features that evoke emotions and promote engagement. The app aimed to create an intuitive and motivating study experience for users by carefully selecting color schemes aligned with psychological theories. The following color choices were intentionally applied throughout *Study Buddy* to enhance user experience:

- Red: Colored the project deadlines red as it conveys urgency and energy.
- Blue: Used in the calendar and scheduling features to establish a sense of calmness, trust, and organization, which helps reduce user stress while planning study sessions.
- Green: Incorporated into progress tracking and achievement badges to represent balance and growth, reinforcing the user’s sense of accomplishment and motivation to continue studying.
- Yellow: Highlighted key actions like reminders to grab attention and evoke a sense of urgency without overwhelming the user.
- Orange: Applied to call-to-action buttons to create warmth and enthusiasm, subtly encouraging users to interact with the app more frequently.

Fig 16. The Psychology of Color: Emotional and Behavioral Impacts of Colors in Branding and Design (PySquad, 2024)

By integrating psychological principles, I transformed *Study Buddy* from a functional tool into an engaging and user-centered application. These efforts not only improved the user experience, but also prepared me to apply HCI principles in future projects.

Finally, incorporating mathematics and psychology into my work has equipped me with the tools to tackle complex challenges in computer science. From solving technical issues like gimbal lock to designing user-friendly applications, these interdisciplinary skills are vital for success in my career in computer graphics and human-computer interaction.

Link to my mobile application: https://www.figma.com/design/studybuddy

SECTION 3: ETHICS

Ethical decision-making is a fundamental pillar of my career in computer science, particularly as a future software engineer specializing in AI and data security. Every day, professionals in this field face ethical dilemmas related to data privacy, artificial intelligence (AI), and responsible research, with consequences that extend beyond technical implementations to impact individuals, businesses, and society. Upholding integrity and trust requires a strong ethical foundation and deliberate ethical choices. One of the most critical areas where this applies is data privacy, where handling sensitive information, such as personal identifiers, financial records, and health data, demands strict adherence to data protection laws and ethical guidelines. To safeguard user privacy and maintain confidentiality, I must ensure secure data storage and processing while complying with regulations such as the General Data Protection Regulation (GDPR) and the California Consumer Privacy Act (CCPA). By prioritizing security and responsible data management, I can help prevent misuse or exploitation of personal information.

One ethical challenge I encountered while I was developing a deep learning model for facial recognition as part of a Deep Learning CV (computer vision) project. During evaluation, I observed disparities in accuracy across different demographic groups, with the model performing significantly better on lighter-skinned individuals compared to darker-skinned individuals. This bias arose due to imbalanced training data, where the dataset contained disproportionately fewer samples from underrepresented groups. Recognizing the ethical implications of deploying a biased model, I implemented corrective measures such as data augmentation, synthetic sample generation, and bias-aware loss functions to improve fairness. I further ensured model transparency by incorporating explainable AI techniques (e.g., Grad-CAM visualizations) to identify which facial features contributed to classification decisions. This experience reinforced my commitment to responsible AI development, where fairness, accountability, and unbiased performance are prioritized to prevent real-world harm and ensure ethical deployment of deep learning technologies.

In software development, ethics influences decisions regarding intellectual property, security practices, and responsible coding. Writing secure and efficient code is not just a technical requirement—it is an ethical responsibility to protect users from security vulnerabilities and potential cyber threats. Daily tasks such as reviewing security protocols, implementing encryption methods like hashing, and following secure coding guidelines will ensure that my work upholds ethical standards. Ensuring software accessibility for diverse user groups, including individuals with disabilities, is another key ethical consideration guiding my design and development choices.

In addition to technical ethics, my professional responsibilities include research integrity and ethical collaboration. Avoiding plagiarism, crediting sources, and reporting results transparently are crucial to maintaining credibility in research. Ethical peer review, honest data reporting, and responsible authorship are daily practices that uphold the integrity of the field. By embedding ethical decision-making into my daily work—whether in AI development, secure coding, or research integrity—I will actively contribute to a more responsible and equitable technological landscape. Upholding ethical principles is not just a professional obligation but a key factor in building trust and long-term success in my computer science and AI career.

CITI Program Certification – Responsible Conduct of Research:

I have successfully completed the CITI Program Responsible Conduct of Research (RCR) course. This course covered essential topics such as authorship, collaborative research, conflicts of interest, data management, peer review, plagiarism, and research misconduct. By completing this certification, I have strengthened my understanding of ethical research practices and integrity in academic and professional settings.

CITI PROGRAM

Completion Date 03-Feb-2025
Expiration Date 03-Feb-2029
Record ID 67773703

This is to certify that:

Sanketh Karuturi

Not valid for renewal of certification through CME.

Has completed the following CITI Program course:

Responsible Conduct of Research
(Curriculum Group)
Responsible Conduct of Research (All Disciplines)
(Course Learner Group)
1 - Basic Course
(Stage)

Under requirements set by:

Oregon State University

CITI
Collaborative Institutional Training Initiative

101 NE 3rd Avenue, Suite 320
Fort Lauderdale, FL 33301 US
www.citiprogram.org

Generated on 03-Feb-2025. Verify at www.citiprogram.org/verify/?w1f738a9d-13d7-4fda-a546-00618a11f842-67773703

SECTION 4: PROFESSIONAL GOALS

My primary professional goal is to secure a software engineer position specializing in artificial intelligence by June 1,2025. As a Computer Science graduate student at Oregon State University, I have gained expertise in deep neural networks and Intelligent agents, and I want to leverage these skills in an industry setting. To achieve my professional goal, I will refine my knowledge in deep learning frameworks such as TensorFlow and PyTorch, implement a real time semantic segmentation of Aerial (Satellite) imagery using U-Net Architecture with ResNet-50 encoder by this term end (March 21,2025) and complete at least three more AI-driven projects by May 2025. Additionally, l will participate in two Kaggle AI competitions to benchmark my problem-solving capabilities against industry standards. I will also network with professionals in the AI field through career fairs and LinkedIn and apply to at least 10 AI-focused full-time positions each month starting in March 2025, ensuring I secure a role that aligns with my expertise and career aspirations.

In the next five years, I aim to advance my career by working on AI-driven automation in industries such as autonomous systems and get promoted to a leadership role. To solidify my expertise, I will contribute to impactful AI projects, publish at least one research paper in a reputable journal like IEEE, and obtain certifications such as the TensorFlow Developer Certification and AWS Machine Learning Specialty Certification by January 2027. I will also attend at least two AI conferences annually to stay updated on industry trends and build connections with leading experts in the field. These steps will ensure I continuously develop my technical and analytical skills while establishing credibility within the AI community.

To ensure steady progress toward my goals, I will follow a structured plan by tracking my milestones every four months, allowing me to evaluate my progress three times a year and make necessary adjustments based on feedback from mentors and industry professionals. Additionally, I will enroll in specialized AI courses, such as IBM's Generative AI, and participate in workshops that provide hands-on experience with emerging technologies like Multimodal AI and Natural Language Processing (NLP) with Transformers. By maintaining a disciplined approach, refining my skills, and actively engaging with the AI community, I will position myself for a leadership role in AI-driven software engineering by 2029, achieving both personal and professional growth.

REFERENCES

1.National Instruments. (n.d.). *3D Cartesian coordinate rotation—Euler VI*. Retrieved January 24, 2025, from https://www.ni.com/docs/en-US/bundle/labview-api-ref/page/vi-lib/analysis/coordinate-llb/3d-cartesian-coordinate-rotation-euler-vi.html

2. PySquad. (2024, October 22). *Designing with purpose: The psychological keys to effective UI/UX*. Medium. https://medium.com/@pysquad/designing-with-purpose-the-psychological-keys-to-effective-ui-ux-c636e72fa6ec