Structured Outline – HNR 499 Final Project

From Equations to Engines: Translating Mathematical Foundations of AI into Motorcycle Part Recognition

---

Abstract

- One-paragraph overview of the project's arc: theory → implementation → real-world application.
- Key concept: *AI as math in action* — showing how linear algebra, calculus, and probability underpin machine learning.
- Summary of workflow:
  - Built and deconstructed MNIST models (in six abstraction stages).
  - Applied the same mathematical reasoning to a YOLOv8 motorcycle part identifier.
- Highlight core finding: as abstraction decreases, understanding and control increase.
- End with metrics: MNIST ≈ 98.8% accuracy; YOLOv8 mAP50 ≈ 0.98.

---

1. Introduction

Goal: establish context, purpose, and relevance.

- Restate original HNR 401 proposal: *bridging the gap between math theory and AI coding practice.*
- Explain why understanding *how* models learn (not just *that* they learn) matters.
- Personal motivation: connecting mathematical theory to a mechanical engineering context (Kawasaki Ninja 250 rebuild).
- Outline of paper structure: math foundations → controlled experiments (MNIST) → applied implementation (YOLOv8).
- Transitional note: "By the end, the same calculus that adjusts neural weights in a digit recognizer identifies motorcycle parts."

---

2. Mathematical Foundations of Machine Learning

Purpose: anchor later work in explicit mathematical reasoning.

2.1 Linear Algebra — Representing Data

- Matrices and vectors as the fundamental data structure.
- MNIST digits = 28×28 matrices; each convolution = matrix multiplication.
- Discuss PCA and dimensionality reduction (from Meeting 1 notes).
- *Insert Figure:* example of 3×3 convolution kernel applied to 5×5 patch (from Model 6).

2.2 Calculus — Learning Through Gradients

- Gradient descent as optimization engine; loss minimization.
- Backpropagation = chain rule applied repeatedly.
- *Insert Example:* derivative path from Model 5 where manual backprop was coded.
- Mention role of learning rate and convergence.

2.3 Probability & Statistics — Judging Performance

- Output probabilities via softmax.
- Metrics: accuracy, precision, recall, mAP.
- Statistical validation: how overlapping train/val sets inflate performance (ethical transparency).
- *Insert Table:* summary of metrics definitions.

---

3. Progressive De-Abstraction: From Black Box to Bare Math

Theme: showing how each layer of abstraction removed reveals the underlying mathematics.

3.1 Model 1 – High-Level CNN (Keras)

- Uses predefined layers; achieves 98 %+ accuracy quickly.
- “Push-button AI”: the baseline demonstration of functionality.
- Talking point: efficiency vs. transparency.

3.2 Model 2 – Manual Training Loop

- Introduced tf.GradientTape(); explicit loss and gradient computation.
- Now visible: how weights update every batch.
- Reflect: learning feels mechanical, not magical.

3.3 Model 3 – Raw TensorFlow Ops

- Direct control of convolutions, ReLU, pooling, softmax.
- Introduced Glorot initialization; connect to variance control in math.
- Shows complete visibility of matrix operations.

3.4 Model 4 – Pure NumPy Softmax Regression

- Single-layer, fully manual implementation.
- Demonstrates that even a simple linear model can learn via algebraic operations.
- Accuracy ≈ 91 %; limited but transparent.

3.5 Model 5 – Two-Layer MLP with Manual Backpropagation

- Every derivative step computed manually.
- Connect directly to calculus section — partial derivatives driving optimization.
- Discuss role of hidden layer and nonlinear transformation (ReLU).

3.6 Model 6 – Manual Convolution Visualization

- Hand-built edge and blur filters.
- Illustrates what CNNs "see."
- *Insert Figures:* horizontal/vertical edge detection, pooling comparison.

3.7 Reflection Table

- Columns: Model #, Abstraction Level, Math Visibility, Accuracy, Key Takeaway.
- Emphasize: as abstraction ↓, insight ↑.

---

4. Application Phase — Motorcycle Part Identifier (YOLOv8)

Purpose: demonstrate transfer from theory to engineering practice.

4.1 Motivation & Dataset

- Problem: identifying small, similar components during a rebuild.
- Collected 486 images across 15 classes (list major ones).
- *Insert Figure:* labeled dataset sample (piston vs. valve spring).
- Discuss manual labeling and folder organization (images/train + labels/train).

4.2 Model Architecture

- Explain YOLO ("You Only Look Once") approach: simultaneous detection + classification.
- Compare to MNIST CNNs:
    - MNIST = single object classification
    - YOLO = multi-object detection with bounding boxes
- Components: backbone → neck → head.
- *Insert Diagram:* YOLO pipeline annotated with matrix operations.

4.3 Training Process

- Config: 60 epochs, imgsz = 768, batch = 16, AdamW optimizer.
- *Insert Table:* key hyperparameters.
- Note use of automatic mixed precision (AMP) and cosine LR scheduler.
- Mention Google Colab environment, GPU (Tesla T4).

4.4 Results

- mAP50 ≈ 0.983, mAP50-95 ≈ 0.746.
- Per-class precision/recall values (summarize top and bottom).
- Discuss why overlapping train/val sets raise reported accuracy.
- *Insert Graph:* accuracy and mAP vs epoch (Training vs Validation).

4.5 Comparison to Hand-Drawn Models

- Similarities: convolutional feature extraction, gradient descent, probabilistic output.
- Differences: YOLO adds spatial localization (bounding boxes + objectness score).
- Analogy: MNIST identifies “what,” YOLO identifies “what and where.”
- Reflect on re-encountering the same math in a new domain.

---

5. Ethical & Practical Reflections

- Data ethics: personal vs. public data; small dataset limitations.
- Reproducibility and transparency — importance of documenting all training settings.
- Model bias and overconfidence in small sample sizes.
- Safety/utility of AI in mechanical contexts (assistive vs. replacement).
- Tie back to HNR program theme: responsible innovation.

---

6. Future Work

- Separate train/val sets, add validation augmentation (flip, rotation, brightness).
- Expand dataset beyond 486 images.
- Experiment with mobile exports (TorchScript / ONNX).
- Potential integration: AR-guided rebuild assistant.
- Publish as open-source educational dataset.

---

7. Conclusion

- Recap journey: abstract math → hands-on implementation → practical application.
- Central argument: mathematical literacy enables deeper AI control and creativity.

- Personal reflection: from “seeing AI as magic” to “understanding it as math.”
- Forward-looking statement: the same process could apply to other engineering systems.

---

8. References & Appendices

- **References:** (MLA-style list from your HNR 401 proposal.)
- **Appendices:**
  - Code excerpts (MNIST and YOLO).
  - Tables of accuracy/mAP per model.
  - Screenshots of training logs.
  - YOLO data.yaml structure for reproducibility.