***Spring 2025 course announcement***

***MATH 392: Intro to Neural Networks***

***Instructor: Arvind Suresh (arvindsuresh@arizona.edu)***

***Credits: 3 (counts toward the ‘Application Course” requirement for math majors)***

***Prereqs: MATH 223***

***Can potentially be waived if student has some experience with multi-variable functions or matrices/vectors; please contact the instructor to enquire!***

***(Note: more or less time will be spent as needed to cover the requisite mathematical content, the pre-requisites are mainly to ensure a reasonable pace can be maintained.)***

***Coding prereqs: Experience with Python is useful but not needed because a lot of code will already be provided, and we will make systematic use of Github Copilot to get by with minimal coding.***

***How to sign up: Email instructor to verify pre-requisites (can skip this if you’ve taken MATH 223), then fill out enrollment form in MATH Rm 108 (main office in the math building).***

***Email math-academics@arizona.edu for more assistance in signing up.***

***When: Mon-Wed, 11 – 12:15 pm***

***Where: Modern Languages, Rm 201.***

***Goal:***

***To provide students with a self-contained introduction to the mathematics and practical implementation of neural networks, which are a fundamental class of machine learning models that underlie modern AI’s like ChatGPT.***

***Reasons why you might like to enroll:***

***• You are interested in AI and would like a self-contained introduction covering the basics.***

***• You are considering internships/jobs in the field of machine learning/AI and would like to get started building a portfolio with projects to showcase to potential employers.***

***• You enjoy courses that blend theory (from math) with practice (coding).***

***• You are a math major looking to fill the “Application Course” credit.***

***Learning Objectives:***

**• Understand the key mathematical concepts used in neural networks, including linear algebra, gradient descent, and backpropagation.**

**• Learn to build and implement simple neural networks using libraries like PyTorch.**

**• Analyze and evaluate neural network models, with an emphasis on model optimization, regularization, and hyperparameter tuning.**

**• Gain experience in the research method (namely, asking questions and being able to hunt down answers or resources).**

**• Prepare for independent research by developing the ability to approach problems related to neural networks and machine learning with a solid mathematical framework.**

***Course features:***

***• Math concepts will be motivated by asking natural questions about datasets provided by the instructor.***

***• Every week, students will learn key topics and immediately engage with the material through hands-on coding exercises (Jupyter notebooks prepared before-hand by the instructor).***

***• Assignments will primarily consist of mini-projects, implemented in Python using industry-standard packages (sklearn and PyTorch).***

***• Students will maintain a GitHub repository containing their mini-projects and final project.***

***• Students will learn to use GitHub Copilot for Education (free and powerful AI code assistant) to write code with minimal effort.***

***• Heavy emphasis will be placed on the process of doing research, namely, asking lots of interesting questions and collaborating with peers on projects.***

***Schedule by week***

***(tentative, might spend more time on math/coding portions depending on the background and interests of students)***

***Week 1: Introduction to Machine Learning and Neural Networks***

***Topics***

***1.*** ***Overview of Machine Learning***

***•*** ***Introduction to supervised learning.***

***•*** ***Applications and importance of machine learning.***

***2.*** ***Universal Approximation Theorem***

***•*** ***Statement and implications for neural networks.***

***3.*** ***Case Study: Image Classification in PyTorch***

***•*** ***Example of a simple feedforward neural network for image classification.***

***•*** ***Code demonstration of building, training, and evaluating a model on a small dataset (e.g., MNIST).***

***Learning Objectives***

***•*** ***Understand supervised learning as a key machine learning paradigm.***

***•*** ***Appreciate the theoretical significance of the universal approximation theorem.***

***•*** ***Gain initial exposure to neural network implementation in PyTorch.***

***Hands-On Example***

***•*** ***Use PyTorch to demonstrate:***

***•*** ***Loading and visualizing a dataset.***

***•*** ***Defining a simple network architecture.***

***•*** ***Training the model.***

***•*** ***Evaluating and visualizing predictions.***

***Week 2: Mathematical Foundations – Linear Algebra (I)***

***Topics***

***•*** ***Vectors and Matrices***

***•*** ***Definition and operations: addition, scalar multiplication, and dot products.***

***•*** ***Matrix multiplication and its geometric interpretation.***

***•*** ***Identity matrix and inverse matrix.***

***•*** ***Applications in Neural Networks***

***•*** ***Representing data as vectors (e.g., input features as vectors).***

***•*** ***Matrix multiplication in neural network computations (e.g., inputs and weights).***

***•*** ***The role of the dot product in calculating activations.***

***Learning Objectives***

***•*** ***Understand vector and matrix operations.***

***•*** ***Be able to apply matrix multiplication to represent neural network calculations.***

***Hands-On Session***

***•*** ***Implement basic matrix operations using NumPy.***

***•*** ***Demonstrate how a neural network layer computes its output using matrix multiplication.***

***Week 3: Mathematical Foundations – Basic Statistics***

***Topics***

***•*** ***Descriptive Statistics***

***•*** ***Mean, median, and mode.***

***•*** ***Variance, standard deviation, and range.***

***•*** ***Interquartile range and outliers.***

***•*** ***Correlation and Covariance***

***•*** ***Covariance: measuring the relationship between two variables.***

***•*** ***Correlation coefficient: normalizing covariance for easier interpretation.***

***•*** ***Interpretation of correlation in datasets.***

***•*** ***Importance of Data Distribution***

***•*** ***Normal distribution and its significance in data analysis.***

***•*** ***Skewness and kurtosis.***

***Learning Objectives***

***•*** ***Understand and calculate key statistics: mean, variance, correlation, etc.***

***•*** ***Learn to interpret statistical measures and their impact on modeling.***

***Hands-On Session***

***•*** ***Calculate statistics for a sample dataset using Python (NumPy/Pandas).***

***•*** ***Visualize distributions using histograms and box plots.***

***Week 4: Mathematical Foundations – Basic Probability***

***Topics***

***•*** ***Introduction to Probability***

***•*** ***Basic probability rules: addition, multiplication, and conditional probability.***

***•*** ***Probability distributions: uniform, normal, and binomial distributions.***

***•*** ***Random variables: discrete vs continuous.***

***•*** ***Bayes’ Theorem and Conditional Probability***

***•*** ***Conditional probability and Bayes’ theorem.***

***•*** ***Applications in machine learning, especially in classification.***

***•*** ***Expectation and Variance in Probability***

***•*** ***Expected value of a random variable.***

***•*** ***Variance and standard deviation in probability distributions.***

***Learning Objectives***

***•*** ***Understand fundamental probability concepts: conditional probability, random variables, and distributions.***

***•*** ***Learn how to apply Bayes’ theorem and basic probability rules in data analysis.***

***Hands-On Session***

***•*** ***Use Python to simulate random variables and calculate basic probabilities.***

***•*** ***Generate and visualize probability distributions.***

***Week 5: Introduction to Loss Functions in Machine Learning***

***Topics***

***1.*** ***What is a Loss Function?***

***•*** ***Definition and role in machine learning.***

***•*** ***Difference between loss and cost (loss per sample vs average loss).***

***2.*** ***Common Loss Functions (MSE for regression, cross-entropy for classification)***

***3.*** ***Visualizing Loss***

***•*** ***Loss landscapes and gradients.***

***•*** ***Intuition behind minimizing loss during training.***

***4.*** ***Connecting Loss to Optimization***

***•*** ***Gradient descent as a method to minimize loss.***

***•*** ***Backpropagation as a mechanism to compute gradients in neural networks.***

***Learning Objectives***

***•*** ***Understand the purpose of loss functions in machine learning.***

***•*** ***Learn about commonly used loss functions for regression and classification.***

***•*** ***Appreciate the connection between loss functions and optimization.***

***Hands-On Session***

***•*** ***Explore different loss functions using synthetic datasets.***

***•*** ***Use Python to compute MSE and Cross-Entropy Loss on example datasets.***

***•*** ***Visualize how changes in predictions affect the loss values.***

***Week 6: Linear Regression and Its Connection to Neural Networks***

***Topics***

***•*** ***Linear Regression Overview***

***•*** ***Basics of fitting a line to data using least squares.***

***•*** ***Loss function for regression: Mean Squared Error (MSE).***

***•*** ***Multivariable linear regression: extending to multiple input features.***

***•*** ***Link to Neural Networks***

***•*** ***Neural networks as generalizations of linear models.***

***•*** ***Understanding linear layers in neural networks.***

***Learning Objectives***

***•*** ***Understand the mechanics of linear regression and how it minimizes loss.***

***•*** ***Recognize how linear models relate to the structure of neural networks.***

***Hands-On Session***

***•*** ***Implement linear regression from scratch using Python.***

***•*** ***Use Scikit-learn to quickly fit a linear regression model to a dataset and visualize results.***

***Week 7: The Perceptron – Concepts, History, and the XOR Problem***

***Topics***

***1.*** ***The Perceptron***

***•*** ***Concept: a single-layer binary classifier.***

***•*** ***Mathematical formulation: input, weights, bias, activation function.***

***•*** ***Training via weight updates: the perceptron learning rule.***

***2.*** ***History of the Perceptron***

***•*** ***Invented by Frank Rosenblatt in 1958.***

***•*** ***Early promise and excitement in artificial intelligence.***

***3.*** ***The XOR Problem***

***•*** ***Demonstration of a linearly non-separable problem.***

***•*** ***Limitations of the perceptron in solving non-linear problems.***

***•*** ***Introduction to multi-layer networks as a solution.***

***Learning Objectives***

***•*** ***Understand the perceptron as a foundational building block for neural networks.***

***•*** ***Appreciate the historical significance of the perceptron and its limitations.***

***•*** ***Recognize the need for non-linearity and multi-layer networks.***

***Hands-On Session***

***•*** ***Implement the perceptron algorithm for a simple linearly separable dataset.***

***•*** ***Visualize the decision boundary for the perceptron.***

***•*** ***Attempt to classify XOR data and discuss why it fails.***

***Week 8: Introduction to Multilayer Perceptrons (MLPs)***

***Topics***

***•*** ***What is a Multilayer Perceptron?***

***•*** ***Extension of the perceptron with hidden layers.***

***•*** ***Activation functions enabling non-linear transformations.***

***•*** ***Why MLPs Solve the XOR Problem***

***•*** ***Representing non-linear decision boundaries with multiple layers.***

***Learning Objectives***

***•*** ***Understand the architecture and function of an MLP.***

***•*** ***Recognize how MLPs overcome the limitations of single-layer perceptrons.***

***Hands-On Session***

***•*** ***Implement an MLP for the XOR problem using PyTorch.***

***•*** ***Train and evaluate the model, visualizing the learned decision boundary.***

***Week 9: Introduction to Optimization via Gradient Descent***

***Topics***

***•*** ***Basics of Gradient Descent***

***•*** ***Concept of gradients and their role in optimization.***

***•*** ***Learning rate and its impact.***

***•*** ***Variants of Gradient Descent***

***•*** ***Batch, Stochastic, and Mini-Batch Gradient Descent.***

***Learning Objectives***

***•*** ***Understand how gradient descent is used to optimize loss functions.***

***•*** ***Learn the differences between gradient descent variants.***

***Hands-On Session***

***•*** ***Visualize gradient descent on a 2D function.***

***•*** ***Use PyTorch’s autograd to implement parameter updates in a simple example.***

***Week 10: Backpropagation***

***Topics***

***•*** ***Backpropagation: its role in computing gradients and training neural networks.***

***•*** ***Applying backpropagation to single-layer networks.***

***•*** ***Extending backpropagation to multi-layer networks using the chain rule.***

***Learning Objectives***

***•*** ***Understand how backpropagation computes gradients layer by layer.***

***•*** ***Learn its application in training multi-layer neural networks.***

***Hands-On Session***

***•*** ***Manual implementation of backpropagation for a single-layer network.***

***•*** ***Use PyTorch to observe backpropagation in a multi-layer perceptron.***

***Week 11: Activation Functions***

***Topics***

***•*** ***The purpose of activation functions: enabling non-linearity in neural networks.***

***•*** ***Common activation functions: Sigmoid, Tanh, ReLU, and their modern variants.***

***•*** ***Choosing the right activation function for a task.***

***Learning Objectives***

***•*** ***Understand the importance of activation functions in neural networks.***

***•*** ***Learn the properties, advantages, and limitations of commonly used activation functions.***

***Hands-On Session***

***•*** ***Visualize and compare activation functions.***

***•*** ***Experiment with different activation functions in a PyTorch model.***

***Week 12: Regularization in Neural Networks***

***Topics***

***•*** ***Why Regularization?***

***•*** ***Overfitting and its causes in machine learning models.***

***•*** ***Common Regularization Techniques:***

***•*** ***L1 and L2 regularization (penalizing large weights).***

***•*** ***Dropout: randomly deactivating neurons during training.***

***•*** ***Data augmentation: enhancing the dataset to improve generalization.***

***Learning Objectives***

***•*** ***Understand the importance of regularization in training neural networks.***

***•*** ***Learn and apply various regularization techniques.***

***Hands-On Session***

***•*** ***Implement L2 regularization and dropout in a PyTorch model.***

***•*** ***Experiment with how regularization impacts model performance on a small dataset.***

***Week 13: Model Evaluation and Hyperparameter Tuning***

***Topics***

***•*** ***Model Evaluation***

***•*** ***Train, validation, and test splits.***

***•*** ***Common evaluation metrics: accuracy, precision, recall, F1-score.***

***•*** ***Visualizing performance with confusion matrices.***

***•*** ***Hyperparameter Tuning***

***•*** ***Key hyperparameters: learning rate, batch size, number of layers, etc.***

***•*** ***Grid search and random search.***

***•*** ***Introduction to advanced techniques like Bayesian optimization.***

***Learning Objectives***

***•*** ***Learn to evaluate models effectively using appropriate metrics.***

***•*** ***Understand the impact of hyperparameters on model training and performance.***

***Hands-On Session***

***•*** ***Evaluate a trained model using performance metrics in PyTorch.***

***•*** ***Perform a simple grid search for hyperparameter tuning.***