MATH 392: Intro to Neural Networks

Arvind Suresh

Dec 11, 2024

What is AI, anyway?

- Modern AI's like ChatGPT are based on so called **Large Language Models** (LLMs).

What is AI, anyway?

- Modern AI's like ChatGPT are based on so called **Large Language Models** (LLMs).
- These are very sophisticated programs– given an input ("prompt"), they generate an output (a textual response, image, and so on).

What is AI, anyway?

- Modern AI's like ChatGPT are based on so called **Large Language Models** (LLMs).
- These are very sophisticated programs– given an input ("prompt"), they generate an output (a textual response, image, and so on).
- For text, output is generated word-by-word:
  The model is trained to generate the *most likely* word next, given the words that have come before.

What is AI, anyway?

- Modern AI's like ChatGPT are based on so called **Large Language Models** (LLMs).
- These are very sophisticated programs– given an input ("prompt"), they generate an output (a textual response, image, and so on).
- For text, output is generated word-by-word:
  The model is trained to generate the *most likely* word next, given the words that have come before.
- Thus, LLM's are in the business of making *predictions*.

What is AI, anyway?

- Modern AI's like ChatGPT are based on so called **Large Language Models** (LLMs).
- These are very sophisticated programs– given an input ("prompt"), they generate an output (a textual response, image, and so on).
- For text, output is generated word-by-word:
  The model is trained to generate the *most likely* word next, given the words that have come before.
- Thus, LLM's are in the business of making *predictions*.
- So, at its core, modern AI falls within the framework of **machine learning** (ML).

Framework of ML

- **Given**: a *dataset* (matrix):

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.
  - Each row is a *data point* (features and target).

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.
  - Each row is a *data point* (features and target).

- **Believe**: the target can be predicted from the features, i.e. there is a "one true function" F such that

$$Y = F(X_1, \ldots, X_m) + (\text{noise}).$$

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.
  - Each row is a *data point* (features and target).

- **Believe**: the target can be predicted from the features, i.e. there is a "one true function" F such that

$$Y = F(X_1, \ldots, X_m) + (\text{noise}).$$

- **Goal**: *Learn* $F \ldots$

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.
  - Each row is a *data point* (features and target).

- **Believe**: the target can be predicted from the features, i.e. there is a "one true function" F such that

$$Y = F(X_1, \ldots, X_m) + (\text{noise}).$$

- **Goal**: *Learn* $F \ldots$ i.e. find a *good approximation* to F.

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.
  - Each row is a *data point* (features and target).
- **Believe**: the target can be predicted from the features, i.e. there is a "one true function" F such that

$$Y = F(X_1, \ldots, X_m) + \text{(noise)}.$$

- **Goal**: *Learn* $F \ldots$ i.e. find a *good approximation* to F.
- **Model**: A class of functions that we belive might approximate F (e.g. linear, polynomial, logistic).

Framework of ML

- **Given**: a *dataset* (matrix):
  - Some columns $X_1, \ldots, X_m$ are *features/predictors*.
  - One column Y is the *target variable*.
  - Each row is a *data point* (features and target).

- **Believe**: the target can be predicted from the features, i.e. there is a "one true function" F such that

$$Y = F(X_1, \ldots, X_m) + (\text{noise}).$$

- **Goal**: *Learn* F . . . i.e. find a *good approximation* to F.

- **Model**: A class of functions that we belive might approximate F (e.g. linear, polynomial, logistic).

- **Training**: *Fit the model* to the given data to get the best approximation to F within our class.

The simplest example of a ML model

Choosing a model amounts to choosing a type of “formula” for our approximation of F.

Example (Linear models)

- **Goal**: Predict house prices (Y) from square footage (X).

The simplest example of a ML model

Choosing a model amounts to choosing a type of "formula" for our approximation of F.

Example (Linear models)

- **Goal**: Predict house prices (Y) from square footage (X).
- **Belief**: $Y = mX + b + \text{(noise)}$.

The simplest example of a ML model

Choosing a model amounts to choosing a type of "formula" for our approximation of F.

Example (Linear models)

- **Goal**: Predict house prices (Y) from square footage (X).
- **Belief**: $Y = mX + b + \text{(noise)}$.
- **Model parameters**: m and b.

The simplest example of a ML model

Choosing a model amounts to choosing a type of "formula" for our approximation of F.

Example (Linear models)

- **Goal**: Predict house prices (Y) from square footage (X).
- **Belief**: $Y = mX + b + \text{(noise)}$.
- **Model parameters**: m and b.
- **Training**: Find the values of m and b that best fit the data.

The simplest example of a ML model

Choosing a model amounts to choosing a type of "formula" for our approximation of F.

Example (Linear models)

- **Goal**: Predict house prices (Y) from square footage (X).
- **Belief**: $Y = mX + b + \text{(noise)}$.
- **Model parameters**: m and b.
- **Training**: Find the values of m and b that best fit the data. Get the best approximation to F among all linear functions:

$$f(X) = \hat{m}X + \hat{b}.$$

The simplest example of a ML model

Choosing a model amounts to choosing a type of "formula" for our approximation of F.

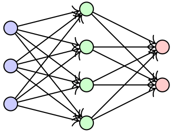
Example (Linear models)

- **Goal**: Predict house prices (Y) from square footage (X).
- **Belief**: $Y = mX + b +$ (noise).
- **Model parameters**: m and b.
- **Training**: Find the values of m and b that best fit the data. Get the best approximation to F among all linear functions:

$$f(X) = \hat{m}X + \hat{b}.$$

- **Prediction**: Given a house with a certain square footage x, predict the house price to be $y = f(x) = \hat{m}x + \hat{b}$.

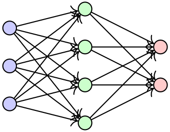
What are neural networks?



- A flexible class of ML models that combine linear and non-linear functions in *layers*:

  Input layer $\longrightarrow$ *Hidden layers* $\longrightarrow$ *Output layer*

What are neural networks?

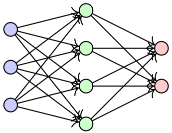


- A flexible class of ML models that combine linear and non-linear functions in *layers*:

  Input layer $\longrightarrow$ *Hidden layers* $\longrightarrow$ *Output layer*

- Inspired by biology– the data flow in a NN is similar to how neurons transmit info by electric impulses.

What are neural networks?

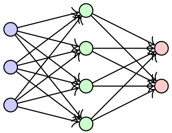


- A flexible class of ML models that combine linear and non-linear functions in *layers*:

  Input layer $\longrightarrow$ *Hidden layers* $\longrightarrow$ *Output layer*

- Inspired by biology– the data flow in a NN is similar to how neurons transmit info by electric impulses.
- The model parameters are called *weights*; they are trained using a process called *backpropagation*.

What are neural networks?



- A flexible class of ML models that combine linear and non-linear functions in *layers*:

  Input layer $\longrightarrow$ *Hidden layers* $\longrightarrow$ *Output layer*

- Inspired by biology– the data flow in a NN is similar to how neurons transmit info by electric impulses.
- The model parameters are called *weights*; they are trained using a process called *backpropagation*.
- LLMs like ChatGPT are based on neural networks with billions of weights!

Why are neural networks so useful?

There are two reasons, one theoretical, one computational.

Why are neural networks so useful?

There are two reasons, one theoretical, one computational.

- (Theoretical): They are *universal approximators*, which means that *any* real-world function F can (in principle) be closely approximated by a NN.

Why are neural networks so useful?

There are two reasons, one theoretical, one computational.

- (Theoretical): They are *universal approximators*, which means that *any* real-world function F can (in principle) be closely approximated by a NN.

- (Computational): They are *scalable*, which means that they can be trained on large datasets with many features.

Why are neural networks so useful?

There are two reasons, one theoretical, one computational.

- (Theoretical): They are *universal approximators*, which means that *any* real-world function F can (in principle) be closely approximated by a NN.

- (Computational): They are *scalable*, which means that they can be trained on large datasets with many features.

  Highly optimized hardware (GPUs) and software (e.g. PyTorch) allow for large-scale parallel computations.

MATH 392: Learning objectives

- Gain experience in the research method (asking questions and hunting for answers).

MATH 392: Learning objectives

- Gain experience in the research method (asking questions and hunting for answers).
- Develop a solid mathematical foundation to approach problems related to NNs: linear algebra, probability and statistics, gradient descent (vector calculus).

MATH 392: Learning objectives

- Gain experience in the research method (asking questions and hunting for answers).
- Develop a solid mathematical foundation to approach problems related to NNs: linear algebra, probability and statistics, gradient descent (vector calculus).
- Learn to build and implement simple NNs using libraries like PyTorch.

MATH 392: Learning objectives

- Gain experience in the research method (asking questions and hunting for answers).
- Develop a solid mathematical foundation to approach problems related to NNs: linear algebra, probability and statistics, gradient descent (vector calculus).
- Learn to build and implement simple NNs using libraries like PyTorch.
- Learn how to evaluate NN's (model optimization, regularization, and hyperparameter tuning).

MATH 392: Learning objectives

- Gain experience in the research method (asking questions and hunting for answers).
- Develop a solid mathematical foundation to approach problems related to NNs: linear algebra, probability and statistics, gradient descent (vector calculus).
- Learn to build and implement simple NNs using libraries like PyTorch.
- Learn how to evaluate NN's (model optimization, regularization, and hyperparameter tuning).
- Demonstrate your learning by completing a final project.

- **Motivation**: Develop math concepts by asking natural questions about datasets.

MATH 392: Course structure

- **Motivation**: Develop math concepts by asking natural questions about datasets.
- **In-class**: Learn key concepts $\longrightarrow$ engage with the material via hands-on coding exercises (Jupyter notebooks).

MATH 392: Course structure

- **Motivation**: Develop math concepts by asking natural questions about datasets.
- **In-class**: Learn key concepts $\longrightarrow$ engage with the material via hands-on coding exercises (Jupyter notebooks).
- **Coding**: Will make systematic use of GitHub Copilot for Education to write code with minimal effort.

MATH 392: Course structure

- **Motivation**: Develop math concepts by asking natural questions about datasets.
- **In-class**: Learn key concepts $\longrightarrow$ engage with the material via hands-on coding exercises (Jupyter notebooks).
- **Coding**: Will make systematic use of GitHub Copilot for Education to write code with minimal effort.
- **Assignments**: 4-5 Mini-projects implemented in Python using industry-standard packages, and one final project.

MATH 392: Course structure

- **Motivation**: Develop math concepts by asking natural questions about datasets.
- **In-class**: Learn key concepts $\longrightarrow$ engage with the material via hands-on coding exercises (Jupyter notebooks).
- **Coding**: Will make systematic use of GitHub Copilot for Education to write code with minimal effort.
- **Assignments**: 4-5 Mini-projects implemented in Python using industry-standard packages, and one final project.
- **GitHub**: Maintain a repository containing mini-projects and final project.

MATH 392: Course structure

- **Motivation**: Develop math concepts by asking natural questions about datasets.
- **In-class**: Learn key concepts $\longrightarrow$ engage with the material via hands-on coding exercises (Jupyter notebooks).
- **Coding**: Will make systematic use of GitHub Copilot for Education to write code with minimal effort.
- **Assignments**: 4-5 Mini-projects implemented in Python using industry-standard packages, and one final project.
- **GitHub**: Maintain a repository containing mini-projects and final project.
- **Research-spirit**: Heavy emphasis on asking interesting questions and collaborating with peers on projects.

MATH 392: Reasons why you might like to enroll

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

MATH 392: Reasons why you might like to enroll

- You are interested in AI and would like a self-contained introduction covering the basics.
- You enjoy seeing math concepts from different courses come together in a single course.

MATH 392: Reasons why you might like to enroll

- You are interested in AI and would like a self-contained introduction covering the basics.
- You enjoy seeing math concepts from different courses come together in a single course.
- You are considering internships/jobs in the field of ML/AI and would like to get started building a portfolio with projects to showcase to potential employers.

MATH 392: Reasons why you might like to enroll

- You are interested in AI and would like a self-contained introduction covering the basics.
- You enjoy seeing math concepts from different courses come together in a single course.
- You are considering internships/jobs in the field of ML/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).

MATH 392: Reasons why you might like to enroll

- You are interested in AI and would like a self-contained introduction covering the basics.
- You enjoy seeing math concepts from different courses come together in a single course.
- You are considering internships/jobs in the field of ML/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.

Closing remarks

I hope you will consider enrolling in MATH 392!

Please feel free to reach out to me at *arvindsuresh@arizona.edu* if you have any questions about the course, or simply want to chat about neural networks (or anything else mathematical, for that matter).

Thank you for your time!!