UNC CHARLOTTE

The WILLIAM STATES LEE COLLEGE *of* ENGINEERING

Introduction to ML
Lecture 1: Course Introduction

Hamed Tabkhi
Department of Electrical and Computer Engineering,
University of North Carolina Charlotte (UNCC)
htabkhiv@uncc.edu

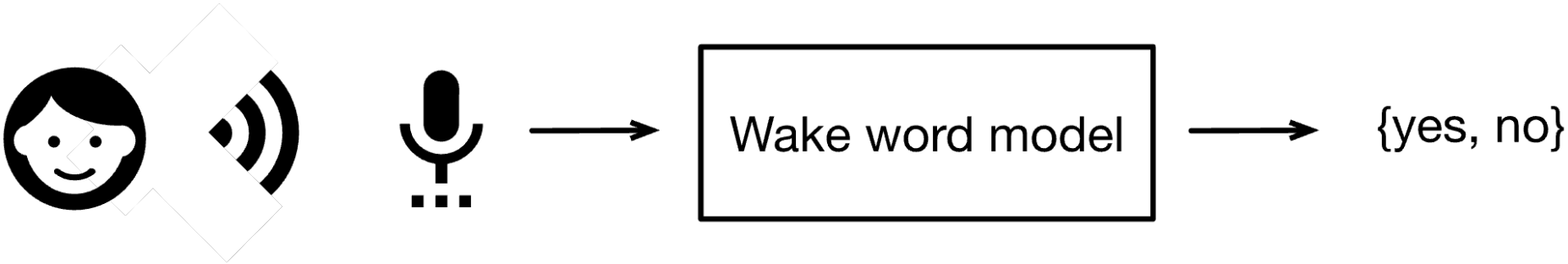
Reference Book for lectures and code samples:

https://d2l.ai/

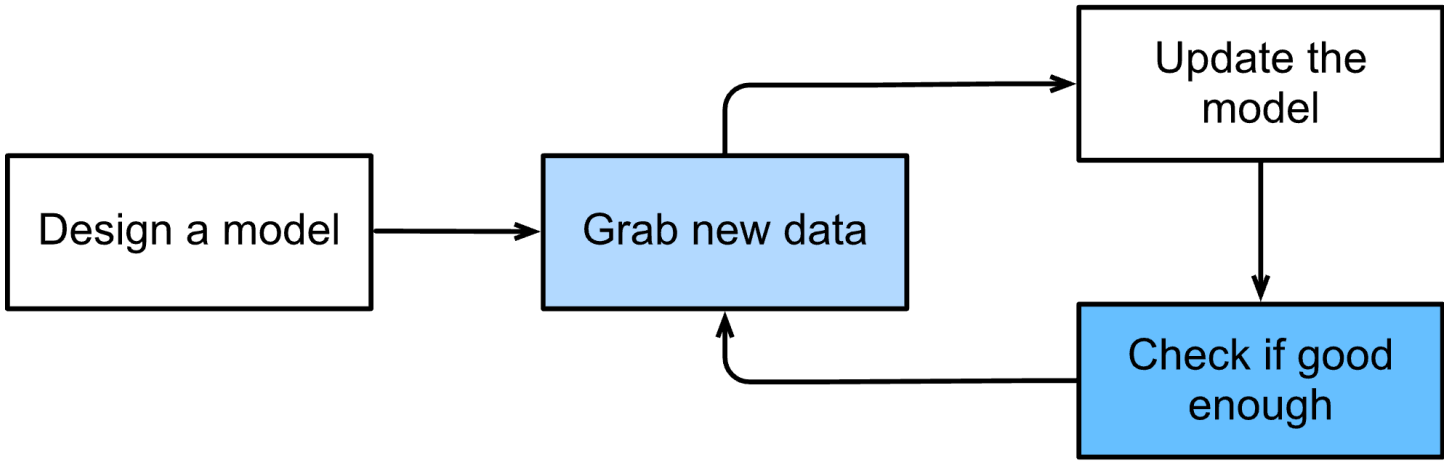
Motivational Example

Imagine just writing a program to respond to a *wake word* such as “Alexa”, “OK Google”, and “Hey Siri”. Try coding it up in a room by yourself with nothing but a computer and a code editor!

How would you write such a program from first principles?



A typical training loop



1.Start off with a randomly initialized model that cannot do anything useful.
2.Grab some of your data - e.g., audio snippets and corresponding {yes,no} labels.
3.Tweak the knobs to make the model perform better as assessed on those examples.
4.Repeat Steps 2 and 3 until the model is awesome.

Key components of a modern AI system

1. The *data* that we can learn from.
2. A *model* of how to transform the data.
3. An *objective function* that quantifies how well (or badly) the model is doing.
4. An *algorithm* to adjust the model's parameters to optimize the objective function.

1- Role of Data in Machine Learning

- They are the training examples – representative samples of real-world / problem we want to solve
- Generally, the more data we have, the easier our job becomes
  - Training more powerful models and rely less heavily on preconceived assumptions.
  - The regime change from (comparatively) small to big data is a major contributor to the success of modern deep learning.
  - Many of the most exciting models in deep learning do not work without large datasets.
  - Some others work in the small data regime but are no better than traditional approaches.

1- Role of Data in Machine Learning

- It is not enough to have lots of data and to process it cleverly.
- We need the *right* data.
- If the data is full of mistakes, or if the chosen features are not predictive of the target quantity of interest, learning is going to fail.
  - Cliché: *garbage in, garbage out*.

- One common failure mode occurs in datasets where some groups of people are unrepresented in the training data.
  - Imagine applying a skin cancer recognition system in the wild that had never seen black skin before.

- Failure can also occur when the data does not merely under-represent some groups but reflects societal prejudices.
  - For example, if past hiring decisions are used to train a predictive model that will be used to screen resumes, then machine learning models could inadvertently capture and automate historical injustices.

2- Role of Model

- Most machine learning involves transforming the data in some sense.
- By *model*, we denote the computational machinery for ingesting data of one type, and spitting out predictions of a possibly different type
- At core, models are statistical models that can be estimated from data.
- In this course, we stretch the limits of classical methods and focus on deep learning models.
  - Deep learning is differentiated from classical approaches principally by the set of powerful models that it focuses on.
  - These models consist of many successive transformations of the data that are chained together top to bottom, thus the name *deep learning*.

3- Objective Function

- We introduced machine learning as learning from experience.
  - By *learning* here, we mean improving at some task over time.
- But who is to say what constitutes an improvement?
  - In order to develop a formal mathematical system of learning machines, we need to have formal measures of how good (or bad) our models are.

- In machine learning, and optimization more generally, we call these *objective functions*.
  - By convention, we usually define objective functions so that lower is better.
  - You can take any function for which higher is better, and turn it into a new function that is qualitatively identical but for which lower is better by flipping the sign.
- Because lower is better, these functions are sometimes called ***loss functions***.

3- Objective Function

- When trying to predict numerical values, the most common loss function is *squared error*,
  - i.e., the square of the difference between the prediction and the ground truth target.
- For classification, the most common objective is to minimize error rate
  - i.e., the fraction of examples on which our predictions disagree with the ground truth.

- Some objectives (e.g., squared error) are easy to optimize, while others (e.g., error rate) are difficult to optimize directly, owing to non-differentiability or other complications.
  - In these cases, it is common to optimize a *surrogate* objective*.
- During training, we think of the loss as a function of the model's parameters and treat the training dataset as a constant.
  - We learn the best values of our model's parameters by minimizing the loss incurred

* Surrogacy is an arrangement, often supported by a legal agreement, whereby a woman agrees to delivery/labour for another person or people, who will become the child's parent after birth.

3- Objective Function

- Doing well on the training data does not guarantee that we will do well on unseen data.
- So, we will typically want to split the available data into two partitions:
  - the *training dataset* (or *training set*), for learning model parameters
  - the *test dataset* (or *test set*), which is held out for evaluation

- We typically report how our models perform on both partitions.
  - You could think of training performance as analogous to the scores that a student achieves on the practice exams used to prepare for some real final exam.
  - Even if the results are encouraging, that does not guarantee success on the final exam. Over the course of studying, the student might begin to memorize the practice questions, appearing to master the topic but faltering when faced with previously unseen questions on the actual final exam.
- When a model performs well on the training set but fails to generalize to unseen data, we say that it is *overfitting* to the training data.

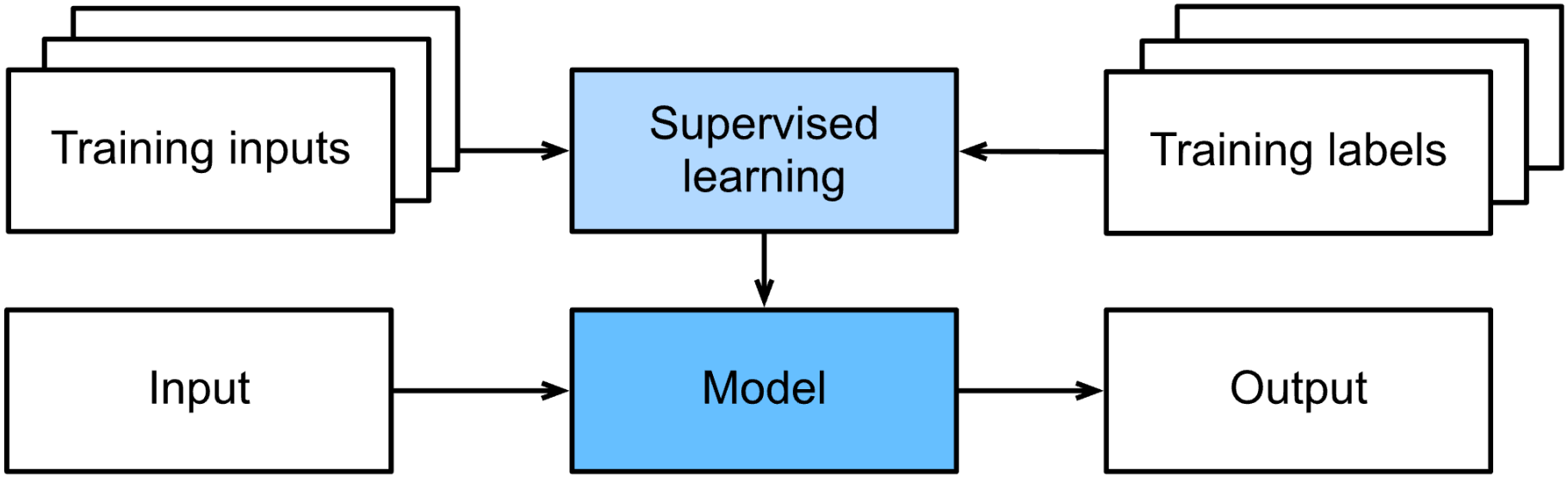
4- Optimization Algorithms (*gradient descent)*

- Once we have got some data source and representation, a model, and a well-defined objective function, we need an algorithm capable of searching for the best possible parameters for minimizing the loss function.
- Popular optimization algorithms for deep learning are based on an approach called ***gradient descent***.
  - In short, at each step, this method checks to see, for each parameter, which way the training set loss would move if you perturbed that parameter just a small amount.
  - It then updates the parameter in the direction that lowers the loss.

Key Machine Learning Paradigms

1. Supervised Learning
   - Regression
   - Classification
   - Sequence Learning
   - Tagging
   - Search
   - Recommender Systems
2. **Unsupervised / Self-supervised learning**
3. Interacting with an environment
4. Reinforcement learning

Supervised Learning



Supervised Learning

- Supervised learning describes tasks where we are given a dataset containing both features and labels and tasked with producing a model to predict the labels given input features.
- While all supervised learning problems are captured by the simple description “predicting the labels given input features”, supervised learning can take diverse forms and require tons of modeling decisions, depending on (among other considerations) the type, size, and quantity of the inputs and outputs.

Supervised Learning: Regression

- When labels take on arbitrary numerical values (even within some interval), we call this a *regression* problem. The goal is to produce a model whose predictions closely approximate the actual label values.
- Lots of practical problems are easily described as regression problems.
  - Predicting the rating that a user will assign to a movie can be thought of as a regression problem and
  - Predicting the length of stay for patients in the hospital is also a regression problem.
- A good rule of thumb is that any *how much?* or *how many?* problem should suggest regression, for example:
  - How many hours will this surgery take?
  - How much rainfall will this town have in the next six hours?

- Regression works with continuous labels (targets)

Supervised Learning: Classification

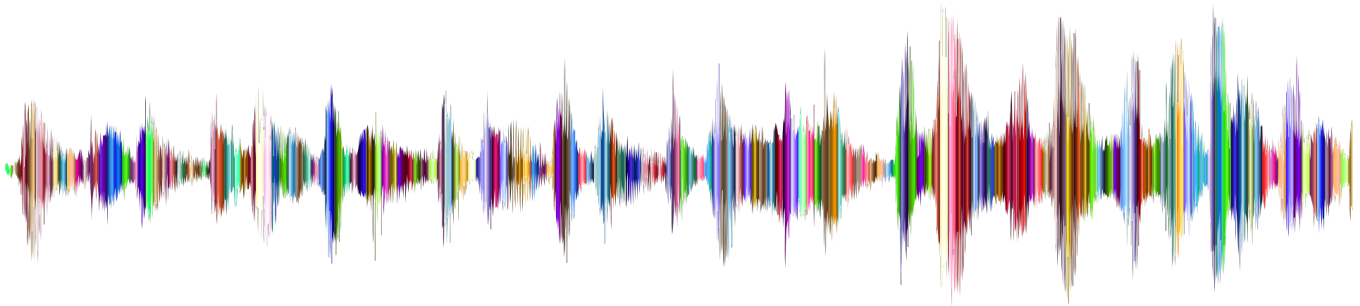
- In *classification*, we want our model to look at features, and then predict which *category* (sometimes called a *class*) among some discrete set of options, an example belongs.
- The simplest form of classification is when there are only two classes, a problem which we call *binary classification*.
  - For example, our dataset could consist of images of animals and our labels might be the classes {cat,dog}.
- Classification often expands to multiple/many classes
  - For handwritten digits, we might have ten classes, corresponding to the digits 0 through 9.
- While in regression, we sought a regressor to output a numerical value, in classification, we seek a classifier, whose output is the predicted class assignment.
- Classification works with Categorical labels

Supervised Learning: Taging

- Auto-tagging problems are typically best described as multi-label classification problems
- E.g. a donkey, a dog, a cat, and a rooster.

Supervised Learning: Sequence Learning

- These problems are among the most exciting applications of machine learning, and they are instances of *sequence learning*.
- They require a model to either ingest sequences of inputs or to emit sequences of outputs (or both).
- Examples include:
  - machine translation and speech-to-text transcription.
  - Speech to Speech translation
  - Broad range of computer vision applications: behavioral analysis, actions detection, anomaly detection over sequence of videos



Unsupervised Learning

- You could think of the supervised learner as having an extremely specialized job and an extremely dictatorial boss.
  - The boss stands over its shoulder and tells it exactly what to do in every situation until you learn to map from situations to actions.

- Considering the opposite situation, it could be frustrating to work for a boss who has no idea what they want you to do. The boss might just hand you a giant dump of data and tell you to *do some data science with it!*
  - This sounds vague because it is.
  - We call this class of problems *unsupervised learning*, and the type and number of questions we could ask is limited only by our creativity.

Some Unsupervised Learning Examples

- Can we find a small number of prototypes that accurately summarize the data?
  - Given a set of photos, can we group them into landscape photos, pictures of dogs, babies, cats, and mountain peaks? Likewise, given a collection of users' browsing activities, can we group them into users with similar behavior? This problem is typically known as *clustering*.

- Can we find a small number of parameters that accurately capture the relevant properties of the data?
  - The trajectories of a ball are well described by velocity, diameter, and mass of the ball. Tailors have developed a small number of parameters that describe human body shape fairly accurately for the purpose of fitting clothes. These problems are referred to as *subspace estimation*. If the dependence is linear, it is called *principal component analysis*.

- Is there a representation of (arbitrarily structured) objects in Euclidean space such that symbolic properties can be well matched?
  - This can be used to describe entities and their relations, such as "Rome" − "Italy" + "France" = "Paris".

Some Unsupervised Learning Examples

- Is there a description of the root causes of much of the data that we observe?
  - For instance, if we have demographic data about house prices, pollution, crime, location, education, and salaries, can we discover how they are related simply based on empirical data? The fields concerned with *causality* and *probabilistic graphical models* tackle such questions.

- Another important and exciting recent development in unsupervised learning is the advent of **deep generative models**.
  - These models estimate the density of the data p(x), either explicitly or *implicitly*.
  - Once trained, we can use a generative model either to score examples according to how likely they are, or to sample synthetic examples from the learned distribution.

Self-supervised Learning

- A major development in unsupervised learning, has been the rise of *self-supervised learning*, techniques that leverage some aspect of the unlabeled data to provide supervision.
- Self-supervised learning is a subset of machine learning techniques where the system learns to understand and work with data by teaching itself to fill in the blanks, so to speak. It's a form of unsupervised learning, but with a structure that mimics supervised learning but without the need for labeled data.
  - For text, we can train models to “fill in the blanks” by predicting randomly masked words using their surrounding words (contexts) in big corpora without any labeling effort.
  - For images, we may train models to tell the relative position between two cropped regions of the same image to predict an occluded part of an image based on the remaining portions of the image
  - To predict whether two examples are perturbed versions of the same underlying image.

- Self-supervised models often learn representations that are subsequently leveraged by fine-tuning the resulting models on some downstream task of interest.

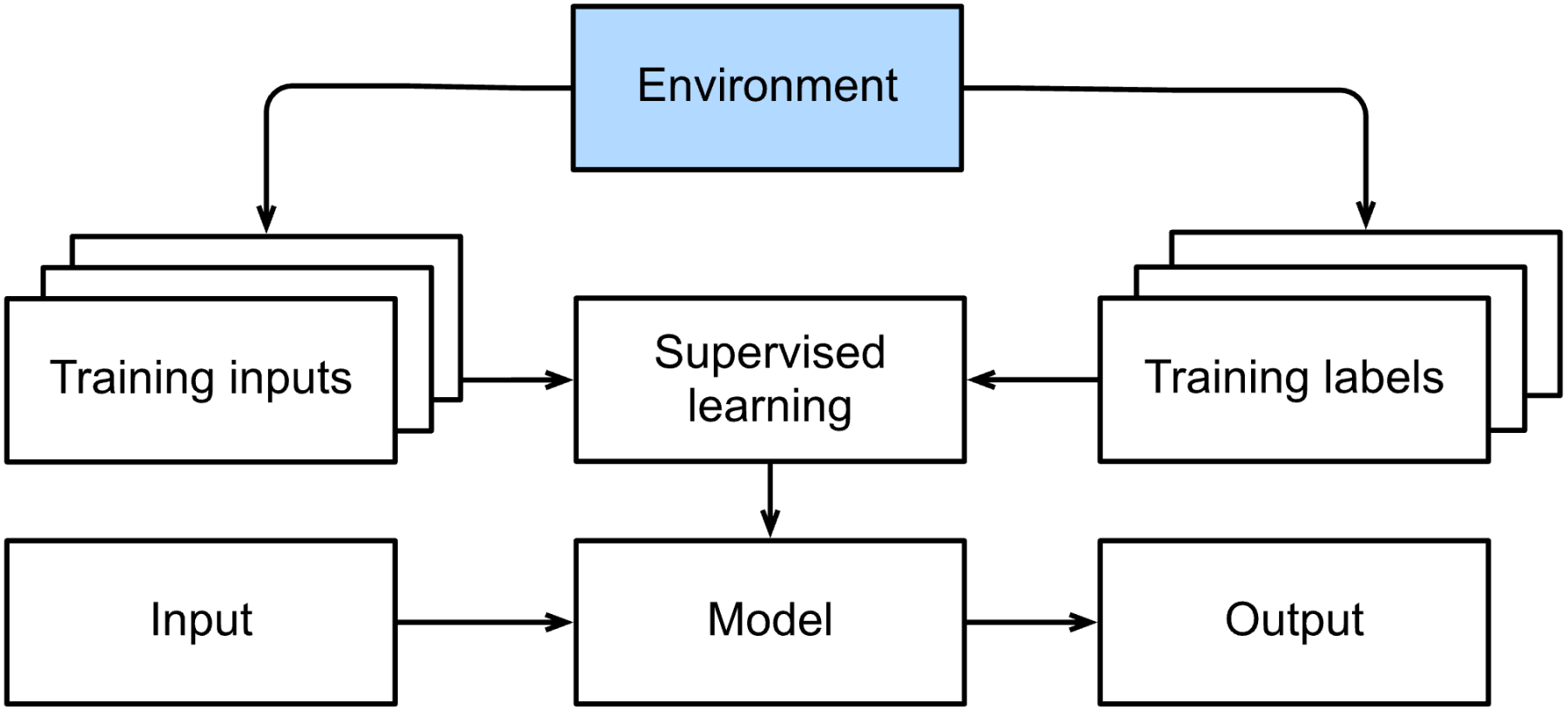
Key Benefits of Self-Supervised Learning

1. **Reduces Dependence on Labeled Data**:
   - Less need for expensive and time-consuming data labeling processes.
   - Makes use of abundant unlabeled data.
2. **Improves Learning Efficiency**:
   - Learns more robust features that can generalize better to new tasks.
   - Often results in models that perform better on a variety of tasks.
3. **Enhanced Understanding of Data**:
   - Enables models to learn more about the structure and nature of the data.
   - Helps in learning representations that capture underlying patterns in the data.
4. **Scalability**:
   - More scalable as it can leverage the vast amounts of available unlabeled data.
   - Particularly useful in domains where labeled data is scarce or expensive to obtain.
5. **Flexibility and Adaptability**:
   - Can be adapted to a wide range of domains and tasks.
   - Offers a versatile approach to handling different types of data (text, images, audio, etc.).

Interacting with an Environment: Offline Learning

- So far, we have not discussed where data actually comes from, or what actually happens when a machine learning model generates an output.
- That is because supervised learning and unsupervised learning do not address these issues in a very sophisticated way.
- *Offline Learning:* we grab a big pile of data upfront, then set our pattern recognition machines in motion without ever interacting with the environment again.
  - Because all of the learning takes place after the algorithm is disconnected from the environment, this is sometimes called ***offline learning***.

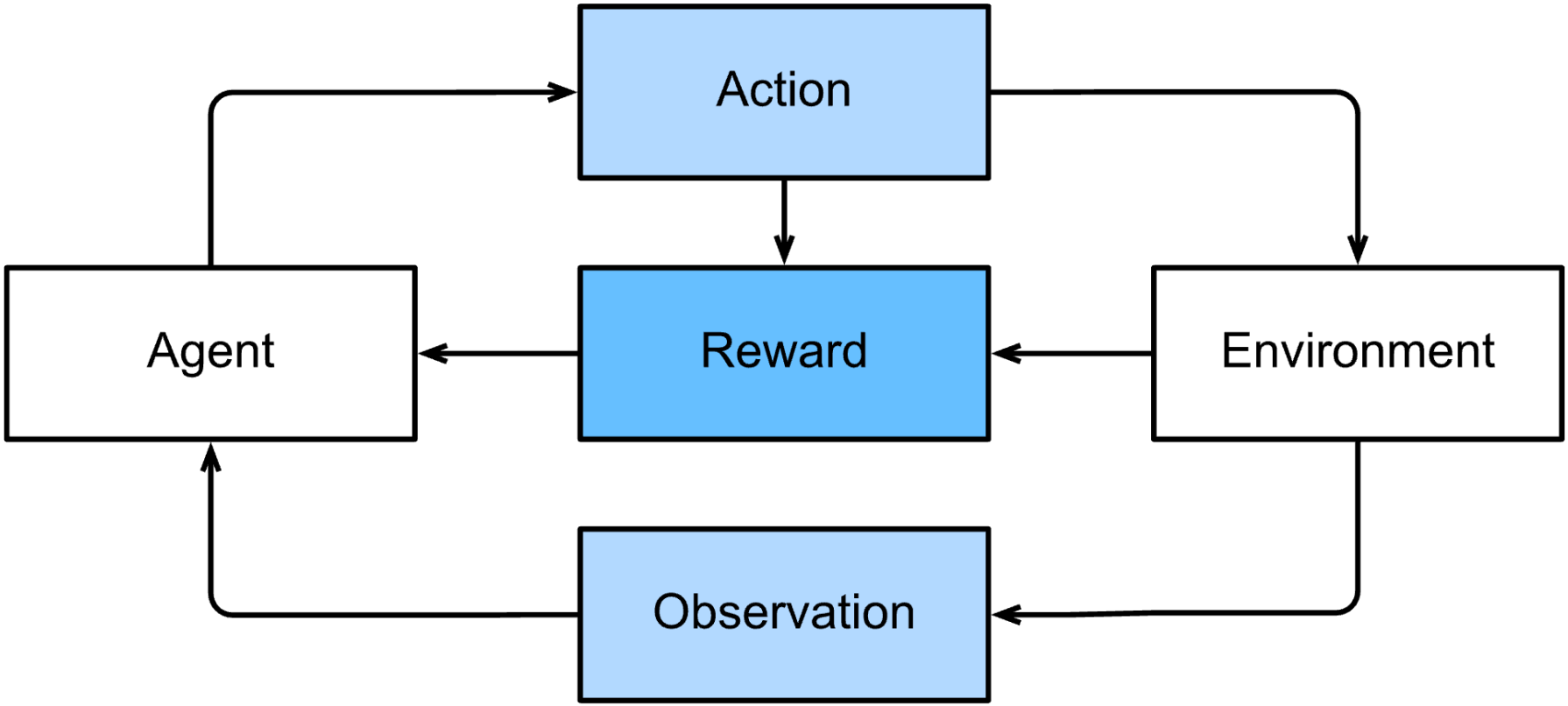
Interacting with an Environment: Online Learning



Interacting with an Environment

- If you grew up reading Asimov's Robot novels, then you might imagine artificially intelligent agents capable not only of making predictions, but also of taking actions in the world.
- We want to think about intelligent *agents*, not just predictive models.
- This means that we need to think about choosing *actions*, not just making predictions. Unlike mere predictions, actions actually impact the environment.

Reinforcement Learning



Reinforcement Learning

- Is applicable to robotics, to dialogue systems, and even to developing artificial intelligence (AI) for video games. *Deep reinforcement learning*, which applies deep learning to reinforcement learning problems, has surged in popularity.
- Reinforcement learning gives a very general statement of a problem, in which an agent interacts with an environment over a series of time steps.
  - The agent receives some *observation* from the environment and must choose an *action* that is subsequently transmitted back to the environment via some mechanism.
  - Finally, the agent receives a reward from the environment.
- The behavior of a reinforcement learning agent is governed by a *policy*.
  - In short, a *policy* is just a function that maps from observations of the environment to actions. The goal of reinforcement learning is to produce good policies.

Reinforcement Learning

- It is hard to overstate the generality of the reinforcement learning framework.
  - We can cast supervised learning problems as reinforcement learning problems. Say we had a classification problem.
  - We could create a reinforcement learning agent with one action corresponding to each class.
  - We could then create an environment which gave a reward that was exactly equal to the loss function from the original supervised learning problem.

- Reinforcement learning can also address many problems that supervised learning cannot.
  - For example, in supervised learning, we always expect that the training input comes associated with the correct label. But in reinforcement learning, we do not assume that for each observation the environment tells us the optimal action.
  - In general, we just get some reward. Moreover, the environment may not even tell us which actions led to the reward.

The Road to Deep Learning

Decade	Dataset	Memory	Floating point calculations per second
1970	100 (Iris)	1 KB	100 KF (Intel 8080)
1980	1 K (house prices in Boston)	100 KB	1 MF (Intel 80186)
1990	10 K (optical character recognition)	10 MB	10 MF (Intel 80486)
2000	10 M (web pages)	100 MB	1 GF (Intel Core)
2010	10 G (advertising)	1 GB	1 TF (Nvidia C2050)
2020	1 T (social network)	100 GB	1 PF (Nvidia DGX-2)

Questions?

UNC CHARLOTTE