

Hands-On AI Projects for the Classroom

A Guide for Computer Science Teachers



ISTE



Hands-On AI Projects for the Classroom

A Guide for Computer Science Teachers

About ISTE

The International Society for Technology in Education (ISTE) is a nonprofit organization that works with the global education community to accelerate the use of technology to solve tough problems and inspire innovation. Our worldwide network believes in the potential technology holds to transform teaching and learning.

ISTE sets a bold vision for education transformation through the ISTE Standards, a framework for students, educators, administrators, coaches and computer science educators to rethink education and create innovative learning environments. ISTE hosts the annual ISTE Conference & Expo, one of the world's most influential edtech events. The organization's professional learning offerings include online courses, professional networks, year-round academies, peer-reviewed journals and other publications. ISTE is also the leading publisher of books focused on technology in education. For more information or to become an ISTE member, visit iste.org. Subscribe to ISTE's YouTube channel and connect with ISTE on Twitter, Facebook and LinkedIn.

Related Resources

AI in the Classroom: Strategies and Activities to Enrich Student Learning by Nancye Blair Black

ISTE online course, *Artificial Intelligence and Their Practical Use in Schools*

To see all books available from ISTE, please visit iste.org/books

To see all courses available from ISTE, please visit iste.org/isteu



©2024. This work is licensed under a Creative Commons Attribution 4.0 International License.

Contents

Foreword	4
Introduction	5
What Is AI?	5
Why Is It Important to Teach About AI in Your Courses?	6
Considerations for Developing and Implementing AI Projects	7
How to Use This Guide	9
PROJECT 1	
Programming With Machine Learning	12
Project Overview	12
Preparation	15
Instructions	16
Extensions	21
PROJECT 2	
AI-Powered Players in Video Games	22
Project Overview	22
Preparation	24
Instructions	25
Extensions	32
PROJECT 3	
Using AI for Robotic Motion Planning	33
Project Overview	33
Preparation	35
Instructions	36
Extension	45
PROJECT 4	
Machine Learning as a Service	46
Project Overview	46
Preparation	48
Instructions	48
Extensions	53
Glossary	54
APPENDIX A	
Unpacking Artificial Intelligence	56
What Is AI?	56
How Do I Know If a Robot or Other Technology Has Artificial Intelligence?	57
What Is Machine Learning?	57
How Do Neural Networks Work?	58
What Is Natural Language Processing?	58
What Is Generative AI?	59
What Types of Ethical Considerations Surround AI?	59
APPENDIX B	
Alignment to ISTE Standards and AI4K12 Five Big Ideas in AI	60
Development Team	62



Foreword

Welcome to the *Hands-On AI Projects for the Classroom* series, a set of guides for teachers who are seeking instructional and curricular resources about artificial intelligence (AI) for various grade levels and across a range of subject areas.

We know that the jobs of the future will increasingly demand knowledge of how to leverage and collaborate with AI as a tool for problem-solving. Unfortunately, most students today are not on a trajectory to fill those jobs. To prepare students, all educators need to understand the implications, applications, and creation methods behind AI. After all, teachers are the most important link in developing the new generation of AI-savvy learners, workers, and leaders.

That's why ISTE has partnered with General Motors (GM) to lead the way regarding AI in education. Anticipating the explosion of interest in AI in education, we teamed up with GM to create scalable professional learning experiences to help educators bring AI to their classrooms in relevant ways, and to support students' exploration of AI-related careers.

These guides are an extension of our work and feature student-driven AI projects curated from educators in the field, as well as strategies to support teachers in implementing the projects in a variety of K-12 classrooms. The projects engage students in both unplugged and technology-infused activities that explore key facets of AI technologies.

The *Hands-On AI Projects for the Classroom* series is just one of the resources ISTE is creating to help educators implement powerful AI projects to prepare students for their futures.

We are convinced that the language of future problem-solving will be the language of AI, and that educators must accelerate their understanding of AI in order to guide the next generation. We are here to help you make that happen!

Joseph South
ISTE + ASCD Chief Innovation Officer



Introduction

What Is AI?

AI pervades learning, working, and living in the modern world. In fact, AI technologies are being developed and applied across all fields of study—from science and government to language acquisition and art. We believe that, in order to be successful in school and in life, *all* K-12 students need a foundational understanding of what AI is, how it works, and how it impacts society. We also believe students need to learn to use AI tools effectively and ethically in their academic lives and beyond. Because of this, AI education is important across *all* subject areas, not just computer science classes.

Yet, even if we believe that, most of us as K-12 educators and education leaders have not had much education in AI ourselves. After seeing the hype about AI in the news and social media, you might find yourself wondering: What exactly is AI? And if you are, you are not alone. In fact, even professionals in the field of AI do not always agree on the answer. Nevertheless, it is important to know what we mean in this guide when we refer to AI.

According to John McCarthy, who first coined the term, artificial Intelligence is “the science and engineering of making intelligent machines, especially intelligent computer programs” (McCarthy, J., 2007)¹. A technology powered by AI is capable of such things as using sensors to meaningfully perceive the world around it, of analyzing and organizing the data it perceives, and of autonomously using that data to make predictions and decisions.

In fact, the autonomous decision-making nature of AI technologies is part of what helps us to distinguish technologies that are and are not AI. For example, autonomous decision-making separates the non-AI automatic doors at your grocery store—which do use sensors to perceive, but open in response to simple if-then conditional statements—from AI-powered, self-driving cars that use sensors to perceive and analyze visual data, represent that data as a map of the world, and make time-sensitive, life-and-death decisions about which direction to move in next, and at what speed.

At their best, AI technologies accomplish tasks that are difficult or impossible for humans to accomplish by themselves. While early AI made decisions based on a preprogrammed set of data and actions, many newer AI technologies use machine learning to improve based on novel data as it is presented. When trained well, AI software is able to efficiently and effectively process, recognize patterns in, and extrapolate conclusions from large data sets across various fields of study. Some AI tools can even use what they have learned to generate new examples of data, text, art, and code based on the patterns that were detected. Similarly, robots powered by AI have the potential to complete tasks that are physically complicated, demanding, or even dangerous for their human counterparts. The projects in this guide and in the other volumes of the *Hands-On AI Projects for the Classroom* series reveal these capabilities to K-12 students across various subject areas and grade levels.

You can learn more about AI and access supporting resources in **Appendix A: Unpacking Artificial Intelligence**.

¹ McCarthy, J. (2007). What is artificial intelligence? Retrieved from jmc.stanford.edu/articles/whatisai/whatisai.pdf



Why Is It Important to Teach About AI in Your Courses?

Over the last decade, the majority of articles about the use of AI in K–12 education focused on two general areas: automating administrative tasks, such as taking attendance and grading assignments or increasing student performance through AI-supported assessment, personalized learning, and increasing engagement in typically mundane rote learning. Recently, attention has shifted to generative AI tools like ChatGPT, prompting both potential time-saving planning applications for teachers and concerns about what will happen when students use these types of tools to generate art, essays, or code.

However, these conversations barely scratch the surface when it comes to AI’s potential for impacting students’ lives—not only in the classroom but throughout their daily activities. The driving purpose of this guide is to look beyond the kinds of strategies mentioned above to consider not only how AI makes life easier at a superficial level, but also what students need to know and understand about AI to ensure they become thoughtful users and even creators of these powerful tools.

This guide is for educators who teach computer science. Once the stuff of science fiction, AI innovations now permeate nearly every facet of modern software development and computational thinking. For example:

- Machine learning powers applications that run classification features, such as spam filters, and predictive features, such as shopping and entertainment recommenders.
- Bots and non-player characters (NPC) in video games use AI to demonstrate human-like behaviors and improve performance.
- AI automation software is combined with robotics to efficiently perform tasks in a warehouse or drive autonomous vehicles.
- Complex machine learning models are abstracted into user-friendly interfaces for customization and integration into websites and apps by organizations and companies both large and small.

As AI is increasingly recognized as a fundamental building block for computing solutions, K–12 computer science students who may someday be writing AI algorithms, training AI models, and developing AI applications should receive more advanced instruction on the workings and challenges of this evolving technology. The projects and resources in this guide provide detailed scaffolding to connect foundational AI knowledge with the computer science concepts and standards that middle and high school computer science teachers are already familiar with. Each project is an entry point for teachers and students to co-learn and expand their knowledge of the field of AI, and each project provides possible extension paths for taking that learning further. As computer science students engage with the projects in this guide, they will discover their own potential to use AI as a computing tool to advance their ability to develop software that solves problems in their life, community, and world.



Considerations for Developing and Implementing AI Projects

This guide provides student-driven projects that can directly teach subject area standards in tandem with foundational understandings of what AI is, how it works, and how it impacts society. Several key approaches were taken into consideration in the design of these projects. Understanding these approaches will support both your understanding and implementation of the projects in this guide, as well as your own work to design further activities that integrate AI education into your curriculum.

Our Student-Driven Approach

The projects in this guide use a student-driven approach to learning. Instead of simply learning *about* AI through videos or lectures, the students completing these projects are active participants in their AI exploration. In the process, students work directly with innovative AI technologies, participate in “unplugged” activities that further their understanding of how AI technologies work, and create various authentic products—from machine learning models to video games—to demonstrate their learning.

Each project’s student-driven activities are divided into three sections: Getting Started, Take a Closer Look, and Culminating Performances.

Getting Started activities hook students’ interest, activate prior knowledge, and introduce them to the project’s objectives.

Take a Closer Look activities develop students’ AI understanding by providing students with scaffolded, guided learning activities that make connections between AI concepts and subject-area content. Students will learn key vocabulary, discover and analyze how real-world AI technologies work, and apply AI tools as they relate to subject-area problems.

Culminating Performances challenge students to synthesize their learning, complete a meaningful performance task, and reflect on the societal impact of what they have learned.

Moreover, in this guide, students’ exploration of AI is framed within the standards, concepts, and depth that would be appropriate to computer science courses. Depending on the level of your students and the amount of time you have available, you might complete the entire project from Getting Started to Culminating Performances, you might pick and choose from the listed activities, or you might take students’ learning further by taking advantage of the additional extensions and resources provided for you. For students with no previous experience with AI education, exposure to the guided learning activities alone will create an understanding of their world that they likely did not previously have. And for those with some background in computer science or AI, the complete projects and resources will still challenge their thinking and expose them to new AI technologies and applications across various fields of study.

In addition to modifying which project activities you implement, you can also modify the projects themselves as needed to support learning at various grade and ability levels. You might provide simpler explanations and vocabulary definitions; assign students to work as individuals, small groups, or a whole class; or adjust the output of the Culminating Performance to better suit their abilities. For example, the Programming with Machine Learning project can be completed by students in either middle school or high school; however, older students should be presented with a deeper understanding of how machine learning works and how it can be integrated into computational solutions. Early and repeated success with these and other AI learning activities can encourage students to continue their exploration into important field-relevant AI applications in the future.

Frameworks and Standards

When making decisions about what to teach about AI in K-12 classrooms, we recommend considering related educational standards and frameworks. In terms of frameworks for teaching AI, this guide references the Five Big Ideas in AI (shown in Figure 1).

THE FIVE BIG IDEAS IN AI

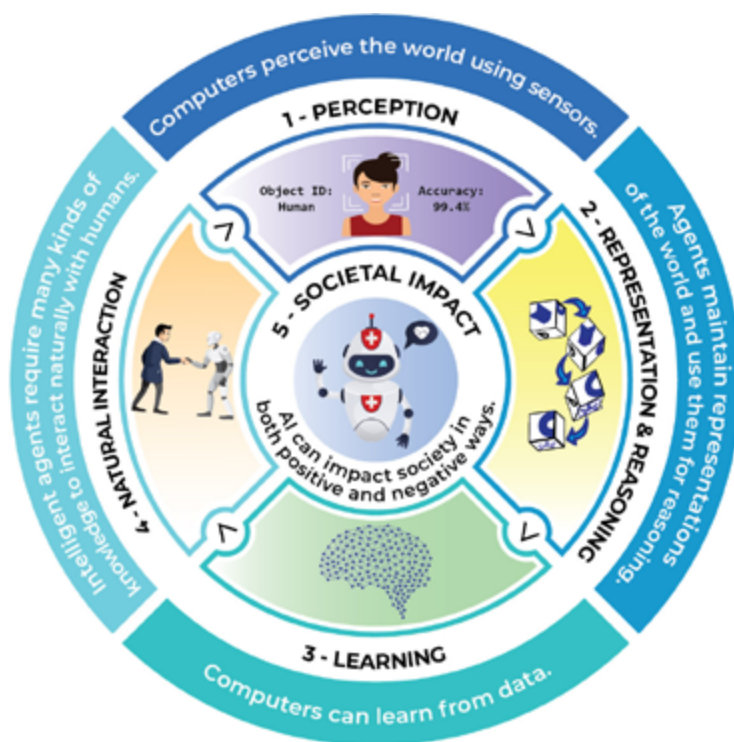



FIGURE 1. Five big ideas in AI. Credit: AI4K12 Initiative. Licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



The Five Big Ideas in AI serve as an organizing framework for the national AI in K-12 education guidelines developed by the **AI4K12 Initiative**. These guidelines articulate what all K-12 students should learn about AI. Each of the projects in this guide illuminates one or more of the first four foundational concepts—perception, representation and reasoning, learning, and natural interaction—as well the societal impact that the concept has in the context of the project.

Additionally, the ISTE Standards and Computational Thinking Competencies can help frame the inclusion and development of AI-related projects in K-12 classrooms. The **ISTE Standards for Students** identify the skills and knowledge that K-12 students need to thrive, grow, and contribute in a global, interconnected, and constantly changing society. The **Computational Thinking Competencies for Educators** identify the skills educators need to successfully prepare students to become innovators and problem-solvers in a digital world. Together, the standards and competencies can give us a language and lens for understanding how these AI projects fit into the greater goal of teaching all students to become computational thinkers. Each of this guide's projects will indicate alignment points with both the ISTE Standards for Students and the Computational Thinking Competencies.

Finally, another way to think about technology use in these student-driven projects is with the SAMR model developed by Dr. Ruben Puentedura. This model classifies the use of technology into four categories: Substitution, Augmentation, Modification, and Redefinition. While uses of technology at the substitution and augmentation level might enhance learning or the performing of tasks, uses at the modification and redefinition level transform the learning experience or task into something that was previously inconceivable, difficult, or even impossible. Many of the activities in this guide will push students' use of technology to the modification and redefinition levels. And while other activities might have students engage with AI technologies conceptually through unplugged activities, or work with AI technologies at the substitution or augmentation level of SAMR, each of the new understandings students walk away with will empower them to understand, use, and possibly even create AI technologies that will fundamentally redefine the way humans live and work.

How to Use This Guide

There are many courses, workshops, seminars, and other learning opportunities both online and offline that focus on the fundamentals of AI. There are also resources that target very tech-savvy educators who have backgrounds in AI concepts and the programming skills necessary to teach students how to code AI-based projects. However, when it comes to the educators who are themselves in the early stages of learning about AI, very little is available to help them transfer what they are learning into meaningful, student-driven classroom activities. That's where the *Hands-On AI Projects for the Classroom* series of guides comes in.

Each guide in this series offers information and activity suggestions that educators can use—regardless of their own experience and background—to ensure their students are afforded opportunities to engage in meaningful activities related to AI. Each guide consists of three parts: Introduction, Projects, and Appendices. Let's briefly review each section.



Introduction

Each of the first four guides in the Hands-On AI Projects for the Classroom series is directed toward a specific group of educators: elementary, secondary, teachers of electives, and computer science teachers. The fifth guide supports all educators in teaching K-12 students about ethical considerations and AI. In addition to this How To section, the introductory section of each guide includes the following information:

- An overview of the *Hands-On AI Projects for the Classroom* series
- A discussion entitled “What Is AI?”
- An explanation of how AI fits into the context for that guide
- Considerations for designing and implementing AI-related projects

Project Design

For ease of use, every project in each of the guides is designed using a consistent format, as follows:

Project Overview

The project overview offers an explanation of what the project is, how it ties to research-based standards, and what students will learn and be able to do as a result of completing the project. Specific sections include a brief overview of the project; the subject, target grades, and estimated duration of the project; objectives for the project; and a listing of relevant standards addressed, such as the ISTE Standards for Students, ISTE Computational Thinking Competencies, AI4K12 Five Big Ideas in AI, and content-area standards.


Preparation

Preparation provides the information educators need in order to put the project into action with students. This section includes a list of materials required for project completion; a list of supporting resources for the educator, if applicable; and a list of planning tasks to complete prior to implementation, such as selecting tools, reviewing online resources, etc.

Instructions

Each project includes instructions for:

- Getting Started activities that hook students’ interest, activate prior knowledge, and introduce them to the project’s objectives.
- Take a Closer Look activities that develop students’ AI understanding by providing students with scaffolded, guided learning activities that make connections between AI concepts and subject area content.
- Culminating Performances that challenge students to synthesize their learning, complete a meaningful performance task, and reflect on the societal impact of what they’ve learned.



While we have provided links to resources to support these activities, in most cases, these activities could be successfully implemented with a variety of similar tools. Moreover, new or improved tools may become available in coming years. Consider the tools and resources listed in the guides simply as suggestions.

Additionally, the inclusion of any material is not intended to endorse any views expressed, or products or services offered. These materials may contain the views and recommendations of various subject-matter experts as well as hypertext links to information created and maintained by other public and private organizations. The opinions expressed in any of these materials do not necessarily reflect the positions or policies of ISTE. ISTE does not control or guarantee the accuracy, relevance, timeliness, or completeness of any outside information included in these materials.

Moreover, prior to using any of the cited resources with students, it is imperative that you check the account requirements for each resource against your school/district student data privacy policy to ensure the application complies with that policy. In addition, some resources' Terms of Service may require parental permission to be COPPA and FERPA compliant for students younger than thirteen years of age.

Extensions

Extensions include strategies and resources for expanding or enhancing the project to support extended student learning.

Glossary and Appendices

Glossary

The glossary includes definitions for terms found in the projects that may be unfamiliar or need explanation for students.

Appendix A: Unpacking Artificial Intelligence

Appendix A provides basic explanations and resources for understanding and teaching fundamental AI concepts.

Appendix B: Alignment to ISTE Standards and AI4K12 Big Ideas

This section provides a high-level overview of how the projects in all five guides in the *Hands-On AI Projects for the Classroom* series align with the ISTE Standards for Students, ISTE Computational Thinking Competencies, and AI4K12 Five Big Ideas in AI.



PROJECT 1

Programming With Machine Learning

Whether they realize it or not, machine learning is integrated into many of the applications students use every day. Powering tools from Netflix to autocorrect, this AI technology is used to quickly process data, personalize the user experience, and make tasks easier.



This project would be great for expanding students' exposure to AI fundamentals in middle school CS courses. This project opens the door to have conversations around cultural bias in AI and the need for diversity in data sets. Especially considering incidents around racial inequity and injustices, stressing the importance of how AI is only as good as the data provided for machine learning is crucial. This emphasis would open the door to rich discussions and enable student cultural relevancy.

—Susan Forget, STEM & PLTW Teacher, Sabin Middle School

Project Overview

In this project, students will learn what machine learning is and how it works. Then they will apply this knowledge to the development of a program that uses a machine learning model they have trained. In the process, they will see how useful machine learning can be to developing today's most effective software solutions.

SUBJECT

Computer science

ESTIMATED DURATION

6–8 hours

TARGET GRADES

6–12

OBJECTIVES

At the end of the project, students will be able to:

- Train a machine learning model.
- Understand sources and implications of sampling bias in datasets.
- Use a machine learning model in the development of a software program.

VOCABULARY

artificial intelligence

bias

classification model

confidence level

data

dataset

decision tree

features

labels

machine learning

model

natural language understanding

sampling bias

supervised learning

test data

training data

STANDARDS

ISTE Standards for Students

1.1 Empowered Learner

- c. Students use technology to seek feedback that informs and improves their practice and to demonstrate their learning in a variety of ways.

1.3. Knowledge Constructor

- b. Students evaluate the accuracy, perspective, credibility and relevance of information, media, data or other resources.

1.5. Computational Thinker

- b. Students collect data or identify relevant data sets, use digital tools to analyze them, and represent data in various ways to facilitate problem-solving and decision-making.
- d. Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

1.7. Global Collaborator

- b. Students use collaborative technologies to work with others, including peers, experts or community members, to examine issues and problems from multiple viewpoints.

ISTE Computational Thinking Competencies

5.1. Computational Thinking

- b. Learn to recognize where and how computation can be used to enrich data or content to solve discipline-specific problems and be able to connect these opportunities to foundational CT practices and CS concepts.

5.3. Collaborating Around Computing

- a. Model and learn with students how to formulate computational solutions to problems and how to give and receive actionable feedback.
- b. Apply effective teaching strategies to support student collaboration around computing, including pair programming, working in varying team roles, equitable workload distribution and project management.

Big Ideas of AI

2. Representation and Reasoning

Agents maintain representations of the world and use them for reasoning.

3. Learning

Computers can learn from data.

5. Societal Impact

AI can impact society in both positive and negative ways.

CSTA K-12 Computer Science Standards

2-DA-08: Collect data using computational tools and transform the data to make it more useful and reliable.

2-DA-09: Refine computational models based on the data they have generated.

2-IC-20: Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

2-IC-21: Discuss issues of bias and accessibility in the design of existing technologies.

3A-AP-12: Create computational models that represent the relationships among different elements of data collected from a phenomenon or process.

3A-IC-25: Test and refine computational artifacts to reduce bias and equity deficits.

3A-IC-26: Demonstrate ways a given algorithm applies to problems across disciplines.

3B-AP-08: Describe how artificial intelligence drives many software and physical systems.

3B-AP-09: Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.

Preparation

MATERIALS

- Computer(s) or tablet(s) with internet connection for accessing tools and resources online.
- Website: [Machine Learning for Kids](#)

SUPPORTING RESOURCES FOR EDUCATORS

- Resource: ["Explaining Machine Learning with Decision Trees"](#)
- Resource: ["Explaining ML with Neural Networks"](#)

ADVANCED PREPARATION

- Set up your class on the website [Machine Learning for Kids](#) (ML4K). For complete instructions for creating your free IBM Cloud account and ML4K teacher account, visit the [help page](#). You may want to have students log into their accounts the day before beginning Activity 2 to confirm they can successfully access the platform.
- Set up a whole-class "Make Me Happy" ML4K project for Activity 3. We suggest you use the "whole-class project" feature of ML4K for this project so the entire class can contribute to the model and crowd-source the data. When you create the project, select the "Whole-class project?" option or, after you create the project, click the "Share" button to share it with the class. A video tutorial is available [here](#).
- Review the supporting materials for the "Titanic Survivors" and "Make Me Happy" ML4K projects.
 - ["Titanic Survivors"](#) Teacher Guidance Document
 - ["Titanic Survivors"](#) Student Worksheet
 - ["Make Me Happy"](#) Teacher Guidance Document
 - ["Make Me Happy"](#) Student Worksheet
- Select an ML4K project for your students' Culminating Performance.

Instructions

GETTING STARTED

Activity 1: Activating Prior Knowledge

In this activity, students will activate prior knowledge about **machine learning** by examining app functionalities they are likely familiar with, although they may not have considered how the tools work. The examples provided explore real-world scenarios that use machine learning to process various **data** types: text, numbers, images, and sound. If needed, students can do a quick search to learn more about the **AI** technology that powers them.

1. Provide students with a high-level introduction to machine learning. Emphasize that machine learning models can work with many types of data. Supporting resources can be found in Appendix A: Unpacking Artificial Intelligence.
2. For each of the real-world examples below, display and have students discuss the following questions in a small group; then share their answers with the whole class.
 - What is the purpose of this machine learning technology?
 - What type of data would the technology need to analyze to learn how to do that task?
 - What type of data does the technology collect to do the task for the user?
 - How do you think that machine learning technology works?

Following are real-world examples of machine learning for small group discussion. Also included are optional extension questions for whole-group discussion.

Text data. Predictive text in email, on smart devices, or during search queries. (Extend thinking: How does predictive text know what should come next? How does it learn your writing style? How does it know how to spell your last name?)

Numeric data. Map apps that determine the shortest route by distance or time. (Extend thinking: How do map apps use numeric values to calculate time for travel? How do they know whether to allot more time for heavy traffic?)

Image data. Facial recognition software used for social media tagging or unlocking smartphones. (Extend thinking: What type of facial features does the software use to recognize a specific person?)

Sound data. A music recognition app, like Shazam, that tells you the title and artist of a song that is playing. (Extend thinking: What types of features is the app identifying in the song? If multiple artists have covered the same song, how would the AI know the difference?)

TAKE A CLOSER LOOK

Activity 2: Train a Machine Learning Model in Machine Learning for Kids

Machine Learning for Kids (ML4K) is a free tool, based on APIs from IBM Watson, that allows students to train and use machine learning models. In this activity, students will work as a class to create and discuss two models, one using a preexisting **dataset** and one using a dataset that the students create. Students do not need to build a programming project for this activity, but will instead focus on learning how to provide data to a machine learning model, train a model, and test a model.

NOTE: The following walk-through asks students to consider reasons why people would or wouldn't survive the Titanic sinking. If you think this topic may be upsetting for students in your class, you can skip this example and teach the concepts of supervised learning, labels, features, models, decision trees, and confidence level through only the second model, "Make Me Happy".

1. Work through the "Titanic Survivors" project as a class. This project uses a provided dataset to develop a predictive machine learning model based on a decision tree.
 - Distribute or display the first 13 steps from the student worksheet for the "**Titanic Survivors**" project.
 - Walk through the first 9 steps together as a class. These steps set up an ML4K "Titanic Survivors" project template and display the training data set. Once the two labels ("survived" and "did_not_survive") and the data are displayed, use the visual to teach the following concepts:
 - This activity uses supervised learning to train a machine learning model to classify data. **Supervised learning** is a form of machine learning in which the trainer provides the AI with **labels** for each of the items in the training data. The AI then analyzes the data in each label group to identify patterns in the **features** (i.e., defining attributes) and creates a model.
 - In this case, the **model** will be represented by a **decision tree** that looks for each of the defining features that it identifies in the data. For example, a model for classifying animal data with the label "elephant" may identify features that reflect four legs; large, floppy ears; and a trunk.
 - Have students analyze the training data provided and look for patterns. Then hold a class discussion using the questions provided in step 10 of the student worksheet.
 - Follow steps 11–13 to train and test the model on the "Learn & Test" page. The results provided predict whether a person with a certain set of characteristics would have survived or not survived the sinking of the Titanic as well as the AI's confidence level with that prediction. Explain that the **confidence level** is the probability that the item has been matched with the correct label.
 - Click on the "Describe your model!" button to see the machine learning model's decision tree. Ask students: Based on the information in the decision tree, what are the features in the model that are associated with the "survived" label? Which features are associated with the "did_not_survive" label?

2. Work through the “**Make Me Happy**” project as a class. The “Make Me Happy” project collects user-created data to create a classifier model that looks at whether a sentence is a compliment or an insult.
 - Prior to the lesson, be sure to set up and share a “whole-class project” version of “Make Me Happy” so that students can collaboratively contribute data from each of their individual accounts.
 - Walk through steps 7–10 of the “Make Me Happy” student worksheet together as a class. These steps open the project and add the labels “kind_things” and “mean_things” to the class project. Explain that this time you will be creating a classification model that will use **natural language understanding** to identify features to classify statements as compliments or insults.
 - Have students log in to their own accounts, open the class “Make Me Happy” project from their list of projects, and click *Train* to access the label buckets. Have each student add one sentence with an extremely nice compliment to the “kind_things” bucket and one sentence with a school-appropriate insult to the “mean_things” bucket. This will be the **training data** used to teach the AI model. Have each student add another compliment and insult to a collaborative document (digital, paper, or whiteboard), which will be used as **test data**. Once all of the data are collected, train the model, then test it with the list of test data to see how it performs. Let students know that you will discuss some reasons why a model might do a good or poor job at a task in the next activity.
 - Conclude this activity by having students explain how the “Make Me Happy” model works using the terms supervised learning, labels, features, models, decision trees, confidence level, classification model, training data, and test data.

Activity 3: Data and Sampling Bias

In this activity, students will consider ways that data sampling for training and testing data can affect the output of a machine learning model. Then they will return to the datasets for their “Make Me Happy” model and make improvements to refine the datasets.

1. Ask students: Are you happy with the way the “Make Me Happy” model performed? Was it always accurate? Could it have performed more accurately?
2. Tell students that an AI does not have opinions or thoughts of its own, but instead can only make decisions based on the data that it learns from. Because machine learning models learn from training data, the quality of the training data sample directly determines the quality of the model. **Sampling bias**, which would lead to an inaccurate model, is caused by having a dataset that does not accurately represent the labels. A quality dataset has the following characteristics:
 - Enough data: The AI needs enough examples to be able to identify patterns in the features of the data. The amount of data needed depends on the specific performance goal for accuracy at the task. The more accuracy needed, the more training data is needed.

- **Accurate data:** The AI needs to be given enough of the right types of examples to understand all of the correct features of items that should be accurately given a particular label. This means that if there are examples of that label that are missing or misleading, the AI will likely not identify those correctly in test data. Some examples of sampling bias include the following.
 - If you were training the AI to identify insects, but you only trained it on examples of ants and beetles, it would likely not be able to identify a praying mantis as an insect because some insect features would be missing from the data. Likewise, if the model will be used by many people in many countries, but insects from just one country are represented, the AI might not be able to recognize insects from other locations.
 - If you trained the AI using a data set in which all of the images of insects were taken in the grass but the images of non-insects were taken in a variety of places, the AI might identify the grass as a feature for the insect label. If the model was tested with an image of a dog in the grass, it might label it as an insect.
 - If there is a feature that could apply to both labels, but is only shown in one, that would mislead the AI. For example, if all of the non-insect training images were also of non-animals, then the AI could possibly mistake any animals with legs, heads, etc. as an insect when the model is tested.
 - If the training data includes significantly more examples of one label than another, then the AI might learn that it is more common and, therefore, incorrectly select that label more often.
- 3. Prompt your students to reflect on the collaborative experience of creating the “Make Me Happy” model. Ask students to identify the elements of the training and test datasets they created that worked well. Then ask them what types of things they could do to improve the model’s results. Record and display their answers. Have students go back into their “Make Me Happy” project, make the dataset improvements they identified, and test the model to see if the results have improved. Have students continue to refine the dataset until the model is able to correctly identify sentences consistently and with a high degree of confidence.
- 4. *Optional:* Extend this learning activity by examining the four examples of machine learning tools in Activity 1 and discussing possible sources and impacts of sampling bias. You may even want to have students go one step further by researching current events that describe incidents where sampling bias led to negative outcomes in the use of machine learning models in applications. While this extension will reveal a downside of using machine learning in applications, the students should focus on the importance of avoiding sampling bias, and the role that people play in the outcome of the machine learning model.

CULMINATING PERFORMANCES

Activity 4: Programming with Machine Learning

To synthesize their learning from this project with other learning from their computer science course, students will use **pair programming** to complete an ML4K project in which they train and integrate a machine learning model into a Scratch, App Inventor, or Python program.

1. **ML4K** provides a variety of project worksheets with training and coding walk-throughs as well as corresponding datasets. Select and assign one or more ML4K supervised learning project options for your students to complete. Recommendations for beginner ML4K projects based on some common computer science course topics include: Cybersecurity—Face Lock; Data Science—Journey to School; Internet of Things—Smart Classroom; and Game Development—Snap!
2. When students have completed the programming portion of this activity, have them answer the following questions with their partner. Then discuss each of these questions as a whole class.
 - When testing your model, how well do you think the app performed? Why do you think that it performed well or not well?
 - What sampling biases did you identify in your dataset? What did you do to improve your dataset and the model's performance?
 - Do you think using a machine learning model made your program more useful or effective than programs that do not use machine learning? Why or why not?
 - What would be another type of program or solution that you could create using the same model you've already trained? How could you expand your model to work with multiple types of programs or solutions?

Activity 5: Reflect

In this activity, students discuss the following questions to reflect on their learning and consider the societal impact of using AI technologies in everyday applications.

- What are some consequences of using machine learning to power applications/apps?
- Now that you know more about machine learning and sampling bias, what questions would you ask yourself before trusting a machine learning tool, such as predictive text, facial recognition, or a product recommender?

Extensions

Here are two ways to expand students' experience in programming with machine learning:

1. Have students work individually or through pair programming to use an **iterative design process** to define a problem of their own to solve with a machine learning application. Students should collect and prepare a dataset; train a machine learning model in ML4K; integrate that model into an original program in Scratch, App Inventor, or Python; and discuss the ethical considerations and societal implications of their solution. For example, students might apply what they learned from the COVID-19 global pandemic to solve ongoing problems related to public health, such as:
 - An app that will turn on a light with voice command, so you can wash your hands without touching the switch.
 - A program that uses images from a video to monitor how well people are practicing personal hygiene (e.g., sneezing into their elbows, covering their mouths when coughing, washing hands after contacting frequently touched surfaces) or social distancing in public places (e.g., staying 6 feet apart and limiting the number of people in a gathering).
 - A program that analyzes the demographic and health information (e.g., age, height, weight, preexisting conditions, etc.) alongside symptoms of people who test positive for a particular bacteria or virus to predict the likelihood that other people who contract the disease will display those specific symptoms based on their specific demographic and health information.
2. Have students compare and contrast two machine learning systems. Train an image classifier model with a training dataset in ML4K as well as another machine learning image recognition system like **Teachable Machine** or Amazon Rekognition (accessible free through **AWS Educate**). Test each of the models with the same set of test images. Is one system easier to train than another? Do they produce the same outputs? Do they have the same confidence levels? Which system is the most accurate?
3. Extend students' experience integrating AI models into programming projects through **Machine Learning for Kids pre-trained models** extensions or MIT's **PoseBlocks** programming platform. Both of these platforms provide students with the opportunity to integrate various extensions harnessing pre-trained AI models, from speech-to-text to face detection, into block-based Scratch programming projects. Both platforms also include an option for students to train their own machine learning classification models in Google's Teachable Machine, then integrate them directly into their Scratch program, expanding the types of models that students are able to train and work with in this project. MIT also provides a five-day **Dancing with AI** curriculum to support students' exploration of training models and thoughtfully integrating them into Scratch projects.



PROJECT 2

AI-Powered Players in Video Games

Games have provided a critical platform for the development of modern artificial intelligence dating back as early as 1949, when Claude Shannon calculated the number of branching moves in chess at 10^{120} and published “Programming a Computer for Playing Chess” (1950). Because games are naturally engaging, problem-solving experiences, they are an ideal arena for explorations in AI and the power of computers to both solve problems and simulate a variety of behaviors. Even the simple AI integration in some of today’s games can provide realistic bots and non-player characters for an improved user experience.



Fred Rogers once said “Play is often talked about as if it were a relief from serious learning. But for children, play is serious learning. Play is really the work of childhood.” The games students play now already use sophisticated and complex AI and will only continue to get better and more advanced as people study and work with AI systems. During the development of this project, we had a considerable amount of conversation about AI playing against human players, how the AI makes the game better, and how AI makes playing the game more engaging. As teachers explore this project and this whole guide, I hope that they remember that “play is serious learning.”

— Mark Gerl, Technology Teacher, The Galloway School

Project Overview

In this project, students will explore how different AI algorithms are used to power or play various types of video games. They will examine the code of several simple games to see how computers play them and then improve their performance based upon their human opponents’ choices. Finally, students will program an original game application with an AI character or object.

SUBJECT

Computer science

TARGET GRADES

9–12

PREREQUISITES

Basic coding skills, preferably with a text-based coding language.

ESTIMATED DURATION

5–8 hours

OBJECTIVES

At the end of the project, students will be able to:

- Explain common uses of AI in gaming.
- Implement an artificial intelligence algorithm to play a game against a human opponent.

VOCABULARY

artificial intelligence

behavior tree algorithm

bot

Finite State Machine (FSM) algorithm

generative AI

Monte Carlo Tree Search (MCTS) algorithm

neural network

non-player character (NPC)

reinforcement learning

STANDARDS

ISTE Standards for Students

1.2. Digital Citizen

- c. Students demonstrate an understanding of and respect for the rights and obligations of using and sharing intellectual property.

1.5. Computational Thinker

- d. Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

1.6. Creative Communicator

- b. Students create original works or responsibly repurpose or remix digital resources into new creations.

ISTE Computational Thinking Competencies

5.4. Creativity & Design

- d. Create CS and CT learning environments that value and encourage varied viewpoints, student agency, creativity, engagement, joy and fun.

5.5. Integrating Computational Thinking

- c. Use a variety of instructional approaches to help students frame problems in ways that can be represented as computational steps or algorithms to be performed by a computer.

AI4K12 Five Big Ideas in AI**2. Representation and Reasoning**

Agents maintain representations of the world and use them for reasoning.

3. Learning

Computers can learn from data.

4. Natural Interaction

Intelligent agents require many kinds of knowledge to interact naturally with humans.

5. Societal Impact

AI can impact society in both positive and negative ways.

CSTA K-12 Computer Science Standards

3A-AP-15: Justify the selection of specific control structures when tradeoffs involve implementation, readability, and program performance, and explain the benefits and drawbacks of choices made.

3A-AP-16: Design and iteratively develop computational artifacts for practical intent, personal expression, or to address a societal issue by using events to initiate instructions.

3B-AP-09: Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.

Preparation

MATERIALS

- Computer(s) or tablet(s) with internet connection for accessing tools and resources online.
- *Optional:* Note cards and pens to support the iterative design process in the culminating experience.

SUPPORTING RESOURCES FOR EDUCATORS

- Book: *Invent Your Own Computer Games with Python, 4th Edition*, chapter 10: Tic-Tac-Toe and chapter 15: The Reversegram Game
- Article: "10 Games That Have Successfully Integrated Artificial Intelligence"
- Video Playlists: "AI and Games"
- Article: "Reinforcement Learning"

Instructions

GETTING STARTED

Activity 1: Activating Prior Knowledge

In this activity, students will hold a class discussion to activate prior knowledge about computer players in the games they play.

1. Hold a class discussion in which students consider the use of AI in games they currently play.
 - What games do you play that have a computer player, either when you play against the computer, or when you have computer characters as teammates or opponents? Do you think that computer player is powered by AI? Why or why not?
 - **Bots** are AI agents that can interact with computer systems or users. In video games, bots can be programmed to independently play a game or to compete as an opponent to, or a teammate for, a human player. **Non-player characters (NPC)** are characters or objects in a game that are not controlled by a human and do not play the game. AI is used to power the behaviors of many advanced NPCs. How do you think an AI playing relatively simple games (e.g., checkers, chess, or Starcraft II) is different from AI powering characters in video games (e.g., Madden NFL, FIFA, World of Warcraft, or Starfield)? Do you think the AI would need different skills when playing a game against a human versus when simulating how a human (or game character) would act?
2. Share this artificial intelligence definition by software developer and author AI Sweigart² with students: "An artificial intelligence (AI) is a computer program that can intelligently respond to the player's moves" (2016). Ask students whether using this definition would change any of their answers as to whether characters in games would be considered AI. Then tell students that during this project they will learn more about the evolving role of AI in video games.

TAKE A CLOSER LOOK

Activity 2: Introduction to How AI Players Work in Games

In this activity, students consider various types of AI bots and NPCs, and why game developers would incorporate them into video games.

1. Have students watch the video "[AI Playing Games #12](#)." Then, independently or in small groups—or using a [jigsaw instructional strategy](#)—have students read one or the following articles. Each of these resources describes various uses of AI in games.
 - "[Artificial Intelligence in Video Games](#)"
 - "[AI in Video Games: Toward a More Intelligent Game](#)"
 - "[Use of Artificial Intelligence in Video Games](#)"

² Sweigart, A. (2017). Invent your own computer games with Python (4th ed.). San Francisco, CA: No Starch Press.

- “AI-driven NPCs are living lives and planning parties all on their own”
- “Best AI games 2023: Released and upcoming”

As students watch and read, have them use a graphic organizer to write down what they learn about the four listed types of AI technologies used in video games. Key points may include:

- A **Finite State Machine (FSM)** algorithm:
 - FSM is a relatively simple AI algorithm that works from a finite, specific list of all possible events a bot or NPC can experience in the game.
 - FSM is not optimal for every game, because it offers limited game challenges.
 - FSM is easy to outsmart, because it becomes repetitive.
- A **Monte Carlo Tree Search (MCTS)** algorithm:
 - In MCTS, the AI visualizes all possible bot moves, then based on those moves, considers what moves a human would make in response and what possible moves it could make in response to the human's move. After weighing the options, it makes the best choice.
 - MCTS is used in many strategy games.
 - Because visualizing the total number of options takes a lot of time and computing power, you can program the AI to pick from a smaller random number of choices to optimize speed.
- A **behavior tree** algorithm:
 - A behavior tree controls the flow of decision-making. It works by asking questions, starting with the first sequence node in a parent tree and moving through a series of sequential decisions; the leaves at the end of the branches are commands that tell the bot or NPC what to do. The AI player would continue to progress along the tree until it fails or doesn't meet a criteria, causing it to revert back to the original sequence node. When that sequence either succeeds or fails it returns the result to a parent node of all possible sequences.
 - A behavior tree allows a bot or NPC to respond to cues in the environment or to human player behaviors.
 - Complex behavior trees with cues to unlock new behaviors can create the illusion that the character is learning or adapting, even without the use of a neural network.
- A **machine learning** algorithm:
 - Machine learning uses **neural networks** to process information through a series of input nodes and hidden layer nodes to analyze the input and produce an output. Some of these AI technologies use generative AI models, such as large language models, to generate NPC behaviors and conversations.
 - Some neural networks that power bots and NPCs typically have a clear goal to achieve and are often given rewards or punishments as they attempt to achieve it (i.e., **reinforcement learning**). If the output achieves the desired goal, the AI succeeds and proceeds to the next goal in the game.

- Those that use **generative AI** algorithms simulate human-like personality traits, autonomous goal-seeking, conversational interactions with both human and NPC players, and have memory about game play. These NPCs may convincingly seem like real human players.

2. Discuss the 4 types of AI technologies that have led to new evolutions in more complex games.

- What do all of these types of AI technologies have in common? Possible answer: In all types, an AI makes decisions based on certain cues or conditions, not randomly.
- How do FSM, MCTS, and behavior trees differ from neural networks? Possible answer: An AI that uses FSM, MCTS, and behavior trees makes decisions based on choices or paths that have been provided to it. An AI that uses neural networks/machine learning can be trained to get better at tasks over time and may even complete tasks in ways that the trainers did not imagine.
- How does the use of each of these technologies for the AI affect the complexity of the bot or NPC in the game? Possible answer: FSM is the most simple; characters can only do a few things. With time, MCTS can process billions or trillions of possible play paths and make the best possible choice. Behavior trees allow for the illusion of learning from the player, giving a more realistic experience. Neural networks allow for the greatest possibility of complexity and realism, but also take the most effort to train to act in desired ways.

3. After the students have identified the characteristics of these four types of AI in games, have them work in small groups to identify or research examples of AI-powered bots or NPCs that are examples of all four types. Possible answers include:

- FSM-powered bots or characters: ghosts from Pac-Man; demons from DOOM; creepers, zombies, and spiders in Minecraft; gym leaders in Pokemon.
- MCTS-powered bots or characters: computer players in digital chess, poker, Go, or Civilization.
- Behavior tree-powered bots or characters: NPCs in World of Warcraft, Stardew Valley, or Animal Crossing; aliens in Alien: Isolation; computer players in Halo 2, Far Cry 4, and BioShock Infinite.
- Machine learning-powered bots or characters: TD-gammon; DeepMind's AlphaStar playing StarCraft II; IBM Watson playing Jeopardy; Mar-IO playing the game Super Mario Brothers; OpenAI's agent playing Minecraft; cast of Inworld Origins; and the owl in Meet Wol.

NOTE: Neural network- and generative AI-powered bots or characters are not yet common, but will likely increase significantly over the next several years.

4. Optional: Let students know that the use of AI with bots and NPCs in games is rapidly advancing. Break students into small groups and have each group perform a quick research activity to find 5–10 facts about one of the topics below that reveals emerging uses of AI in video games. Then have groups share what they learned.

- AI bots that use machine learning to emulate human play styles, such as FIFA or Madden games (possible resource: "[Gridiron Notes: Madden NFL 22 Gameplay Deep Dive Discussion](#)")

- AI bots that use machine learning to get better at the game by actively learning from its human players (possible resource: [QuickDraw](#))
- Curiosity-driven AI bots like the one that plays Super Mario Brothers (possible resource: "[Curiosity-Driven Learning: AI agents exploring without looking at any scores](#)")
- A comparison between AI bots that are increasing in sophistication to be more difficult or realistic versus AI bots that are designed to be "stupider" in order for human players to be able to beat the game (possible resources: "[League of Legends AI Bots are Getting a Major Reboot](#)," "[Todd Howard says that Starfield's ship AI sucks on purpose so players can actually hit stuff: 'You have to make the AI really stupid'](#)")
- Any other AI-powered bot or NPC topics of your choosing

Activity 3: Developing an AI Computer Player

In this activity, students will take their conceptual knowledge about how humans play games and start to think about how a program could be written to have a computer play a game intelligently. In the process, students will examine two programs written in Python. Even if Python is not their standard coding language, students with background knowledge in basic coding skills and text-based languages should be able to understand the simple commands and comments provided.

1. Have students play ten rounds of Rock Paper Scissors against each other. Discuss: As humans, how do you play the game? Were your choices always random? Did they go in a pattern? Did you try to adapt to what the other player did? As you played longer, did you change your strategy for playing?
2. Tell students that in this activity, they will be writing and improving a pseudocode program for a computer player to play Rock Paper Scissors.
 - Have student pairs use natural language pseudocode to write a program for a computer player to play Rock Paper Scissors against a human player.
 - Next, have students play at least 10 rounds of **Rock Paper Scissors, Version 1** against a computer using [Trinket.io](#). Tell them to review the provided Python code and think about how it compares to the pseudocode they wrote. Ask:
 - How is the computer player choosing what move to play? Possible answer: It's not really "choosing;" it's just generating a random selection from line 15 and 16.
 - What is the computer's strategy for how to win? Possible answer: There is no strategy. Winning is pure luck.
 - Is this an example of artificial intelligence? Possible answer: No, because there is no autonomous, perception-based decision making. The computer player does not respond intelligently to the human player's moves.
 - Then have students play at least 10 rounds of **Rock Paper Scissors, Version 2** against a computer using [Trinket.io](#). Tell them to review the provided Python code and think about how it compares to both the pseudocode they wrote and the first Rock Paper Scissors code they reviewed.

- How is this computer player choosing what to play? Possible answer: First, at random. Then, after five rounds of play, lines 29–44 identify the human player’s most-played moves. The computer chooses the move to beat that move (e.g, if the human player throws rock most, then the AI will throw paper).
- What is the computer’s strategy for winning? Possible answer: Assuming humans subconsciously throw a “favorite” move more often; the computer throws the move that will beat that move more often.
- Is this an example of artificial intelligence? Possible answer: Using AI Sweigart’s definition, we can say that this program demonstrates a simple AI, since the computer player tracks and responds to the human player’s most often played moves. As the human player adjusts their play style, the computer player adapts as well.
- What data need to be collected to defeat a human player? Where is it in the code? Possible answer: Lines 14 and 15 build an array of moves, and lines 16–18 count the number and type of moves.
- Is this the only way to have a computer player respond intelligently to a human player in Rock Paper Scissors? Possible answer: No, other options include tracking the moves of all players to improve response; tracking play of the best player and simulating him/her; identifying patterns in how humans select their moves.
- What data need to be collected for a computer to simulate a specific human player? Possible answer: To simulate a person, the computer player would need to collect more rounds of play and look for patterns as well as counts. It would need to save tracking data for many games to an external file or database with a human player identifier.
- Allow student pairs time to improve their pseudocode games to make sure their computer player intelligently responds to the human player. Have pairs compare and contrast their final approach with another pair. Discuss the various ways students came up with to have a computer player intelligently play Rock Paper Scissors, including which is most human-like.

Activity 4: Programming AI for Computer Players

In this activity, students will look at two other simple—but more complex than Rock Paper Scissors—games played by a computer player. The code for these two programs comes from AI Sweigart’s book *Invent Your Own Computer Games with Python*, 4th Edition (2017). In this activity, students should be thinking about how the simple AI is programmed and what type of AI game approach is used: FSM, MCTS, behavior tree, or neural network. Depending on the students’ experience, the size of the class, and the amount of time you have, students can complete one or both of these options as individuals, small groups, or as a whole class. Again, while these programs are written in Python, students with background knowledge in basic coding skills and text-based languages should be able to understand the simple commands and comments provided.

1. Have students play **Tic-Tac-Toe** against a computer using Trinket.io. Have them read through the code, and then direct students to examine lines 90–117 to see how the computer player’s AI is programmed.
 - How is the computer player choosing what to play? Possible answer: The computer is using a decision tree about the current state of the board. Based on the state (sequentially: if it can win the game on that turn, if the opponent can win, or if the corners, center, or sides are free), it makes a move.
 - What is the computer’s strategy for playing? Possible answer: If it can’t win the game on that turn, it will block the opponent from getting three in a row, or pick the best available space.
 - What type of algorithm is used to program the AI? Possible answer: A simple FSM algorithm.
2. Have students play **Reversegram** (a version of Reversi or Othello) against a computer using Trinket.io. Direct students to examine lines 162–82 to see how the computer player’s AI is programmed.
 - How is this program choosing what to play? Possible answer: Lines 162–82 are the decision trees the computer uses to determine which moves will result in the highest number of flips.
 - What is the strategy for playing? Possible answer: The first strategy is to capture a corner (168–71); the next is to select moves that have the highest resulting flips (174–82).
 - What type of algorithm is used to program the AI? Possible answer: This is a simple Monte Carlo Tree Search, branching optimal moves out of all possible moves.

Activity 5: AI in Games and User Experience

In this activity, students will consider how the design of AI-powered bots and NPCs affects the user experience.

1. Discuss the following as a class using a cooperative learning structure, such as **numbered heads together** or **think-pair-share**.
 - How do you feel playing against a computer versus playing against a person? Does it matter if you know whether you are playing against a computer or a person?
 - Do you think that users would rather play against an AI-powered bot that is portrayed as a human character versus as a robot or other nonhuman character?
 - If you didn’t know you were playing against an AI, what clues might tell you that the opponent was an AI?
 - If the AI were always to win, would you think it is a well-designed AI player? Why or why not? If not, how might you approach better designing that AI player?
 - How can you use what you’ve learned from looking at these programs to improve the design of computer-controlled characters and objects in your own games?

CULMINATING PERFORMANCES

Activity 6: Programming a Game With an AI Bot

In this activity, student pairs will use an **iterative design** process to write a game program in which a computer player will play against a human player. While the programs they examined in this project were in Python, they can write this program in any language they work with in class.

NOTE: If this project is implemented in an AP Computer Science course, consider adding additional program requirements, such as the use of a list, a procedure, or an algorithm with multiple control structures.

1. Direct students to use an iterative design process to program a game in which a computer player plays against a human player. Students can remix the code from any of the four programs provided in this project or create a fully original program. If students use any code that was written by someone else, they must provide appropriate acknowledgement through citation or attribution. Some ideas for this game include:
 - Write to expand on Rock Paper Scissors or Tic-Tac-Toe by changing the complexity of the game (e.g., extra hand signs in Rock Paper Scissors, or a larger board in Tic-Tac-Toe), or by changing the way that the computer player adapts to the human player's moves.
 - Write to have an AI play more aggressively or more defensively.
 - Write to increase or decrease the AI's win-loss ratio so that the human can choose an easy or hard level of difficulty.
 - Write two AIs that can play against humans or each other.
 - Write to simulate your personal play style and match your average win-loss ratio.
 - Write to simulate someone else's play style and match their average win-loss ratio.
2. Once students have completed their programs, hold a gallery walk in which students have time to play others' games and provide peer feedback on the user experience, including something they like and something they might improve.

Activity 7: Reflect

In this activity, students should discuss the following questions to reflect on their learning and consider the personal and societal impact of AI in games:

- What do you see as the most powerful or game-changing effect of integrating AI into bot and NPC development and behavior?
- Do you think that gaming environments will eventually have AI-powered characters that are so realistic that you don't know whether it is a human or AI? Why or why not?
- Where do you see the future of AI-powered bots and NPCs in video games going?

Extensions

Following are four ways to expand students' exploration of AI-powered players in video games.

1. Students can learn more about writing games with AI in Python through Al Sweigart's books, *Invent Your Own Computer Games with Python*, 4th Edition, and *Making Games with Python & Pygame*, which are available to read for free on his website, inventwithpython.com.
2. Students can learn more about the use of neural networks and reinforcement learning to train AI players by watching the "[Let's Make an AI that Destroys Video Games \(LAB\) #13](#)". They can take that learning even further by completing the corresponding [hands-on lab](#). Both of these resources walk students through the process of reinforcement machine learning and its application for developing quality AI players.
3. AI technologies are doing more in games than just controlling bots and non-player characters. Expand students' background around the use of AI in game design by exploring other uses, such as:
 - Procedural content generation, used in games like Minecraft and No Man's Sky, in which AI is used to create new landscapes, dialogue, levels, and other elements during game play.
 - Rendering more realistic details, such as texture, lighting, physics, or motion animation.
 - Adaptive gameplay, like that used in Middle-earth: Shadow of War, which adjusts the difficulty and pacing of the game to be personalized to the player's skill level and performance.
 - AI-generated music that matches the player's actions and playstyle in real-time.
4. Extend students coding experience with AI bots and NPCs in games by completing the Programming with Machine Learning project from this guide. In that project, students use Machine Learning for Kids (ML4K) to train their own machine learning models, which can be integrated into coding projects. To tie their learning into this project, select one of the following [ML4K machine learning projects](#) for the culminating performance:
 - Pac-Man - Use machine learning and decision trees to train a Pac-Man player to avoid ghost bots.
 - Noughts & Crosses - Train an AI in a Tic-Tac-Toe style game to learn how to beat the user.
 - Rock, Paper, Scissors - Train an image classifier to recognize hand gestures of the user. Optionally, extend this ML4K project by iterating on the AI computer player's decision-making algorithm to improve game play.
5. Students who have taken Advanced Placement math and computer science courses may want to explore the use of AI to design NPCs within an industry tool through [Unity's online course: Artificial Intelligence for Beginners](#).



PROJECT 3

Using AI for Robotic Motion Planning

Ask the average person about AI and they will likely mention robots. In fact, many may even voice misconceptions that all robots are AI or that AI technology is the same as robotics.



AI is one of many rapidly evolving technologies—from drones, to smart cars, to automated machines used in manufacturing. This AI project empowers students to thrive as critical thinkers, problem-solvers, and innovators as they build a solid foundation in AI and robotics.

— David Lockett, STEM & IT Teacher, Bok Academy

Project Overview

In the light of the above, this project aims to help students distinguish between AI-powered robots and simple automation, and to expose students to the functions and capabilities that are typically unique to AI-powered robots. Students will explore these ideas through discussions, research, and simulations, and then apply their new knowledge to consider a real-world problem and develop a small-scale simulation of an AI robotics solution.

SUBJECT

Computer science and robotics classes.

TARGET GRADES

8–12

PREREQUISITES

Basic coding skills

ESTIMATED DURATION

8–12 hours

OBJECTIVES

At the end of the project, students will be able to:

- Discern between robots that do and do not have AI capabilities.
- Explain how AI-powered robots use perception and reasoning for motion planning.
- Examine fundamental ethical considerations related to AI and robotic motion planning.

VOCABULARY

artificial intelligence (AI)
AI agent
artificially intelligent robot
autonomous
ethical
machine learning

motion planning algorithm
neural network
robot
sensor
sensor fusion
sequential decision-making

STANDARDS

ISTE Standards for Students

1.1. Empowered Learner

- d. Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.

1.4. Innovative Designer

- a. Students know and use a deliberate design process for generating ideas, testing theories, creating innovative artifacts or solving authentic problems.
- c. Students develop, test and refine prototypes as part of a cyclical design process.

1.5. Computational Thinker

- d. Students understand how automation works and use algorithmic thinking to develop a sequence of steps to create and test automated solutions.

ISTE Computational Thinking Competencies

5.1. Computational Thinking

- e. Recognize how computing and society interact to create opportunities, inequities, responsibilities and threats for individuals and organizations.

5.2. Equity Leader

- b. Construct and implement culturally relevant learning activities that address a diverse range of ethical, social and cultural perspectives on computing and highlight computing achievements from diverse role models and teams.

5.4. Creativity & Design

- c. Guide students on the importance of diverse perspectives and human-centered design in developing computational artifacts with broad accessibility and usability.

AI4K12 Five Big Ideas in AI

1. Perception

Computers perceive the world using sensors.

2. Representation and Reasoning

Agents maintain representations of the world and use them for reasoning.

3. Learning

Computers can learn from data.

4. Natural Interaction

Intelligent agents require many kinds of knowledge to interact naturally with humans.

5. Societal Impact

AI can impact society in both positive and negative ways.

CSTA K-12 Computer Science Standards

2-CS-02: Design projects that combine hardware and software components to collect and exchange data.

2-IC-2: Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

3A-AP-13: Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-IC-2: Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.

3B-AP-08: Describe how artificial intelligence drives many software and physical systems.

3B-AP-09: Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.

Preparation

MATERIALS

- One or more AI-powered robots, such as Zumi, GoPiGo with Google Cloud Vision API, AutoAuto, AlphaAI, Loona, or ClicBot.
- Computer(s) or tablet(s) with internet connection for accessing tools and resources online.
- Tool: **CarLab**

SUPPORTING RESOURCES FOR EDUCATORS

- Article: "**What's the Difference Between Robotics and Artificial Intelligence**"
- Article: "**How Robots Work: Robots and Artificial Intelligence**"
- Article: "**Why Sensor Fusion is the Key to Self-Driving Cars**"
- Article: "**From Computational Thinking to Computational Action**"

ADVANCED PREPARATION

- Learn about the sensors and motion-planning functionality of the AI-powered robot you will be using with your class.
- Experiment with the **CarLab** simulation and review steps 5–9 in Activity 3.

Instructions

GETTING STARTED

Activity 1: The Characteristics of an AI Robot

In this activity, students will activate prior knowledge as they consider different types of robots and identify the characteristics that distinguish an **AI robot**.

1. Ask students to list robots that they have used themselves or seen in the news or popular media. If some of the robots they mention are fictional, have them label them as such.
2. Tell students that some robots have artificial intelligence while others do not. Give them the criteria below to evaluate each robot on their list. If students need help, let them know that if a robot does have AI capabilities, you may be able to tell because it can do things like: recognize specific objects or faces, navigate around objects on its own, classify or distinguish between objects, understand or speak in a human language, recognize or express emotions, or improvise when encountering something unexpected. In these ways, the **autonomous** decisions made by AI are more advanced than the simple automation of a task (performed in a prescribed sequence of steps), which non-AI robots are frequently used for.
 - Criteria 1: An AI robot must be able to perceive the world around it.
 - Criteria 2: An AI robot must be able to analyze and organize the data it perceives.
 - Criteria 3: An AI robot must be able to reason and make autonomous decisions based on data it perceives.

TAKE A CLOSER LOOK

Activity 2: Sensors—How Do Robots Perceive and Understand

Sensors allow AI-powered robots to perceive the natural world. In this activity, students will research common sensors used in robotic systems. Then they will examine the class's AI-powered robot to identify its sensors and capabilities.

1. An AI-powered robot uses **sensors** to perceive the world around it. Distribute a graphic organizer or have each student create a table to support their research of robot sensors. The graphic organizer should include three topics: the name of the sensor, the data that the sensor perceives, and how an AI-powered robot might be able to use that data. Using a **jigsaw instructional strategy**, assign one or more unique robot sensors, such as lidar, contact or touch sensors, or barometers, to each individual or small group. Have students research their assigned sensors and record their findings in their table. For example, ultrasonic sensors measure the distance to a target using reflected high frequency sound waves. This sensor's data can be used by an AI-powered

robot to determine that an object has passed in front of it and how far away the object is. Once students have completed their research, have them report their findings to the whole class. They should also add information garnered from other students' presentations to their tables. At the end of the activity, all students will have information about each of the sensors researched.

- Next, have students complete a robot scavenger hunt to examine the hardware, software, and documentation for the AI-powered robotics platform they are using in your classroom and identify the sensors available. For each sensor, have students identify the name of the sensor, the data the sensor collects, and how the data is represented by the robot for the user to see. For example, a table for **Zumi the Robot Car** might look like this:

Sensor	Data	Data Representation For the User
Pi Camera	Visual data. The computer processes it as pixels but displays it in the form of images and video.	Saved on the computer (Raspberry Pi Zero board) as a .jpg file and can be opened in any image file application.
Gyroscope	Rotational motion (how much it is turning) as numeric angles for the x, y, and z axis.	Use <code>update_angles()</code> function to get angles for x, y, and z axis. Angle values range from 1 to 360.
Accelerometer	Acceleration values for each axis to find Zumi's orientation with respect to the strongest force being applied to Zumi (gravity).	Use <code>get_orientation()</code> function to get orientation state value: -1 = unknown 0 = probably falling or moving between states 1 = camera straight up 2 = camera facing down 3 = on right side 4 = on left side 5 = wheels on floor 6 = wheels facing up (upside down) 7 = accelerating faster than 1g
6 IR Sensors	Distance values to a target using reflecting infrared light waves. Sensors are in the front right, bottom right, back right, bottom left, back left, and front left.	Use <code>get_IR_data()</code> function to get one of the 6 sensor values which range from a value between 0 and 255.

3. As students complete these activities, they may notice there are different types of sensors that accomplish the same task, such as detecting the distance from the robot to an object. Ask students why they think there are different types of sensors used to accomplish the same task. Possible answer: Many different sensors are used to collect the same data to create a system of redundancy and to increase accuracy. When data are collected from multiple sources, it is more reliable. Instead of assuming the ultrasonic sensor is reading an accurate distance, you can compare the value with an infrared (IR) sensor to ensure its accuracy. Changing conditions (like weather) can also affect certain sensor values. The more redundancy of measurement, the more reliable the system. Some sensors also measure data differently. Lidar and cameras can both detect objects. Lidar can see how far away something is and its general shape, but cannot see finer details like color and other two-dimensional details. Cameras can see an object's general shape and all the finer details (color, writing, etc.), but do not know how far away objects are. Robotic systems like self-driving cars combine these two sensors to create a better system for object detection. The ability of an AI to combine perception from multiple sensors into one model is called **sensor fusion**.
4. Finally, have students select one of the robot's sensors and describe one way the data perceived can be used to accomplish a task with AI. Possible answer: AI can be used with the camera's image data to detect objects and determine the difference between those objects; it can differentiate between pedestrians, cyclists, and other cars on the road, for example.

Activity 3: Self-Driving Car Simulation—Motion Planning and Obstacle Avoidance

In this activity, students will interact with an AI-powered self-driving car simulator to understand how an AI represents and reasons about the information it perceives.

1. A self-driving car is an example of an AI robot. Ideally, self-driving cars would be able to perceive the world around them and navigate roads safely with little or no human input.
2. Project one or both of the following videos that show both a camera view and the AI representation of the Cruise autonomous vehicle navigating in difficult scenarios involving emergency vehicles, bicycles, and pedestrians: "[Watch Cruise Self-Driving Car Maneuver Around Emergency Vehicles in San Francisco](#)," and "[Watch a Self-Driving Car Safely Maneuver Around Cyclists and Scooters in San Francisco](#)." Ask students what they observe about the sensors, perception, and navigation in the videos. Point out to students that it isn't enough for the autonomous vehicle to perceive with sensors; it has to be able to interpret and represent that perception, as well as reason, to decide how to act on what it perceives. Tell students that in this activity they will learn more about AI representation and reasoning.
3. Run and project the **CarLab** self-driving car simulator for the students to see. Explain the following:
 - CarLab is a simulator that allows the user to train a self-driving car's **neural networks** to safely navigate a track by avoiding walls and other vehicles. In the simulator, the self-driving car—called Network Car—is represented as a sports car. There are several smaller cars simultaneously driving on the track that the user cannot control or program.

- For an AI to make a decision with the information it perceives, it needs to collect data, represent the data, and reason about the best next action to take. Self-driving cars collect a variety of data via sensors and process the data through several algorithms that control various aspects of decision-making. In this activity, students will consider the perception, representation, and reasoning needed for a car's **motion planning algorithm**. A motion planning algorithm is a form of **sequential decision-making** in robotics to support movement and navigation from one point to another. In motion planning, robots reason to make a series of small decisions about what to move and where to go in order to safely navigate or perform other tasks.
4. Ask students: What data would a car's motion-planning algorithm need to perceive, represent, and reason about? What types of sensors would gather those data?
- Possible answer:* The car's motion-planning algorithm will receive input from cameras, lidar, and other sensors about where the lanes and other cars are located. It will then use sensor fusion to combine those input data, which are represented as an occupancy grid showing the free space around the vehicle.
5. Direct students to open the CarLab self-driving car simulator. Walk students through the following steps to create their own copy of the simulator.
- Click the "Fork" button at the top of the repl.it programming environment to create a copy.
 - Click the "Run" button at the top of the repl.it programming environment to run the program.
 - Right-click on the "Car View" window's title bar and select "Layer" then "Top," shown in Figure 2.
 - Right-click the "GridWorld" window's title bar and select "Maximize," shown in Figure 3.
 - When properly launched, the simulator should resemble Figure 4. If at any point students make a mistake during any of the steps above, they can reload the link and start over.

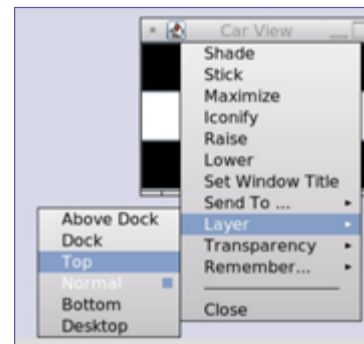


FIGURE 2. Arrange CarView window in CarLab simulator.

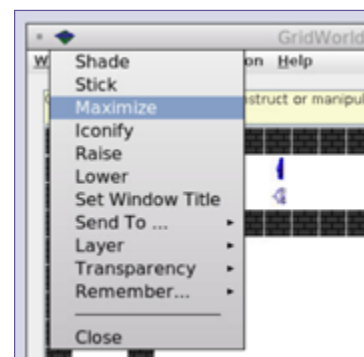


FIGURE 3. Maximize GridWorld window in CarLab simulator.

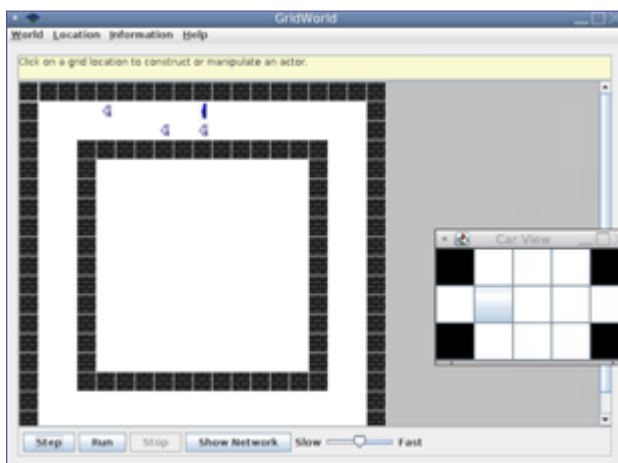


FIGURE 4. CarLab simulator window view.

6. Help students make a connection between the answers they provided in the previous discussion and the CarLab simulation. Explain to students how the self-driving car in the simulator represents the data it perceives:

- The Network Car's perception data are represented in this simulator as a Car View occupancy grid. The grid displays the car's location as a gray cell. The other cells depict the car's perception of the lanes on each side as well as its own lane ahead and behind. Unoccupied cells are represented in white; occupied cells are represented in red; and cells that are outside the car's view are represented in black. An example is shown in Figure 5.
- Direct students to click the "Run" button beneath the track (inside the GridWorld window) to see how the occupancy grid changes when the Network Car is driving. Then, have students click the "Step" button to see how the occupancy grid changes step-by-step as the Network Car navigates the track. In each step, help students describe what the occupancy grid is depicting and identify whether a wall or car is in each occupied cell.

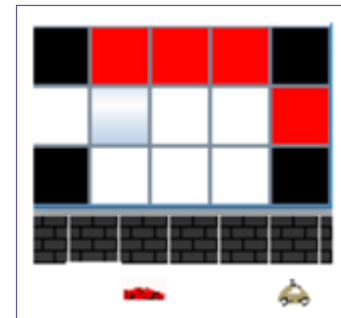


FIGURE 5. Car View occupancy grid in CarLab simulator.

7. Tell students they will next examine and train the neural networks that make up the Network Car's motion planning algorithm.

- The Network Car's motion planning algorithm uses **machine learning** to determine how to move and avoid obstacles in various situations. The three neural networks analyze the provided training data to identify patterns and create a machine learning model that it can use to determine when to move forward, when to turn, and when to change lanes. If students need more background information on how machine learning works, supporting resources can be found in [Appendix A: Unpacking Artificial Intelligence](#).
- Direct students to the NetworkCarTrainer.java file in the left hand Files panel. NetworkCarTrainer is the only file they will be editing in this activity. The three neural networks that CarLab uses to support the Network Car's motion planning algorithm appear in the program as Turning Network, Forward Network, and Lane Change Network. In the previous step, students may have noticed that the Network Car only drives around in a tiny circle. This is happening because, while all of the networks have been provided with some starter data, only the Forward Network—which tells the car when it is safe to drive forward—has been given enough training data to successfully navigate the track. In this activity, students provide the Turning Network and the Lane Change Network with additional training data so that the Network Car will navigate the track more successfully.
- The neural networks in CarLab use training data that consists of three dimensional arrays. Students edit data in the inner two arrays. Each pair of inner arrays is a single training sample that represents a specific scenario. For example, in forwardData, the first array represents the three cells directly in front of the car. A 0 indicates the cell is unoccupied; a 1 indicates the cell is occupied. The second array indicates the target output of the network, whether the car should move forward. A 0 indicates the car should move forward; a 1 indicates it should not. For example, in the scenario in which a small car is located three cells ahead, the Network Car should move forward ($\{\{0,0,1\},\{0\}\}$), but in the two scenarios where a small car is located in either of the next two cells, the Network Car is told not to move forward ($\{\{1,0,0\},\{1\}\}$ and $\{\{0,1,1\},\{1\}\}$).

The neural network learns from this training dataset that it is only safe for the Network Car to move forward when there are two open cells between it and the car in front of it.

8. Training the Turning Network: To train the Turning Network, students enter turnData representing the occupancy of the grid and whether the car should turn.

- When editing the two inner arrays, the first array in turnData represents locations in the occupancy grid. Indexes for each training sample are in order from the left side of the car, in front of the car, and the right side of the car, as shown in Figure 6. A 1 indicates the cell is occupied; a 0 indicates the cell is unoccupied. For example, the scenario depicted in Figure 6 would be represented by the array {1,1,1,0,0,1,0,0,0}. The second array that represents the target output of the network, whether the Network Car should turn, uses a 0 to indicate that it should turn and a 1 to indicate that it should not turn. Given the scenario in Figure 6, the car should not turn; therefore, the second array should read {1}. The full three dimensional array for this training scenario would be {{1,1,1,0,0,1,0,0,0},{1}}.

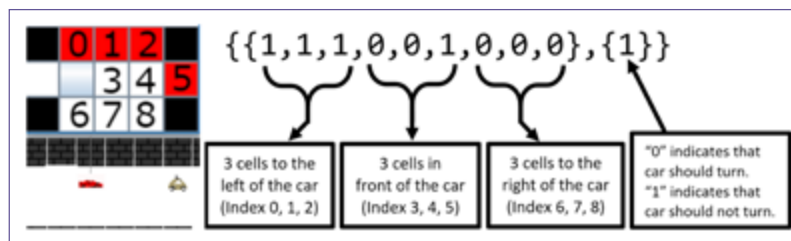


FIGURE 6. Editing TurnData arrays in CarLab simulator.

- If students were programming a car without machine learning, they would need to provide it with all 512 (i.e. 2^9) possible scenarios the car could encounter and what movement it should make. With machine learning, students can train the Network Car to know when to turn with approximately 20–25 pieces of training data that represent a variety of possible scenarios.
- Direct students to the NetworkCarTrainer.java file, where they can see the TurnData dataset. Currently the Network Car is trained on three data points in lines 7–9. The code includes another 17 data points that can be used for training data, but are currently commented out in lines 10–27.
- Have students add training data to the Turning Network by deleting the two backslashes (//) at the start of each commented line. They can choose to use some of the data provided, all of the data provided, or even add additional data of their own. Each time students change the training data, they will need to rerun the entire program to see the changes in the way the Network Car drives. Students can do this by following a process similar to the one they used at the start of the activity: clicking the “Run” button at the top of the programming environment, right-clicking on the “Car View” window’s title bar and selecting “Layer” then “Top,” and right-clicking the “GridWorld” window’s title bar and selecting “Maximize.” Then, they can click the “Run” button beneath the track (inside the GridWorld window) to start the self-driving car simulation. Students should make changes until the Network Car only turns when it can do so safely. The Network Car may still crash because it needs to change lanes, but they will address this next.



PROJECT 3

Using AI for Robotic Motion Planning

- As students adjust the training data, have them observe the changes, then discuss as a class how the additional training data changed the way the car drives.
9. *Training the Lane Change Network:* Next, students will train the Lane Change Network. This self-driving car simulator is programmed so that when the Network Car is driving in any cell, if the Network Car is told not to turn and not to drive forward, it will then check to see if it needs to change lanes to the left or to the right.
- In the real world, self-driving cars must not only make decisions about which direction to drive in, but also how to avoid obstacles. Have students add 3–5 rock obstacles to their track. To add rocks, students click on the track and select `info.gridworld.actor.Rock()`. The Network Car must be able to respond to these situations without crashing or driving in circles. Have students test their car to see how successfully it is able to navigate the track with added obstacles.
 - The Network Car needs to learn to change lanes appropriately to successfully avoid cars and other obstacles. Have students examine the track and the occupancy grid. Ask: Of the cells in the occupancy grid, which cells' data would the Network Car need to perceive and reason with in order to know if it should change lanes to the left? To the right? Have students justify their responses with specific scenarios about the location of walls and cars using those data.
 - Explain that this particular Lane Change Network determines whether the Network Car can safely change to the left or right lane by evaluating the status of the three cells to that side, perceiving whether or not they are occupied, and deciding if it is safe to move there. For example, to decide if it will change lanes to the left, the Network Car will analyze the status of the cell directly to its left, the cell it would move into (diagonally to the front left), and the cell in front of the cell it would move into to determine if those cells are occupied. These cells are represented by indexes 0, 1, and 2 in Figure 6. Since the network does not distinguish between walls, rocks, or small cars, the scenarios presented in the training data must account for an occupied space being occupied by any static or moving object. The Lane Change Network then mirrors this process when making a decision about changing lanes to the right.
 - Have students create their own training `laneChangeData` for the Lane Change Network using the three dimensional array format `{{x,x,x},{x}}`. The indexes of the first inner array represent the occupancy of the three cells on the side the network is analyzing. A 1 indicates the cell is occupied; a 0 indicates the cell is unoccupied. The second inner array that represents the target output of the network, whether the Network Car should change lanes, uses a 0 to indicate that it should change lanes and a 1 to indicate that it should not. There are 2^3 , or 8, possible occupancy combinations. One piece of training data is already provided: `{{0,0,0},{0}}`, which states that if all of the cells to that side are unoccupied, then it is safe to change lanes. Students should enter no more than five additional training scenarios to try to get the Network Car to successfully decide when to change lanes.
 - Each time students change the training data, they will need to rerun the entire program to see changes in the way the Network Car drives. Students can do this by following a process similar to the one they used at the start of the activity: clicking the "Run" button at the top of the programming environment, right-clicking on the "Car View" window's title bar and selecting "Layer" then "Top," and right-clicking the "GridWorld" window's title bar and selecting "Maximize." Then, they can click the "Run" button beneath the track (inside the GridWorld window) to start the self-driving car simulation. Students should make

changes until the Network Car only changes lanes when it can do so safely. Have students test their own Network Car with obstacles and other small cars, and iterate to improve the training data to get it to drive as successfully as possible.

10. Have students test each other's motion planning algorithms by adding new rock obstacles to the course and observing how well the Network Car is able to navigate the track. Have students provide peer feedback about what did and didn't work well, then iterate upon their data to improve the models as needed.
11. Conclude this activity with a class discussion.
 - This simulation defines safe driving as not crashing while traveling around the track. How much data did the car need to safely navigate the track? What might be the effect of having too little data? What might be the effect of having too much data? What other criteria for safe driving might a motion planning algorithm need to account for to be considered successful?
 - How does using machine learning for motion planning improve navigation for self-driving cars?
 - Based on your experience with this activity, what issues with perception and motion planning might arise in the development of AI-powered robots of various kinds?

Activity 4: Ethics Around AI Motion Planning

In this activity, students will consider **ethical** questions that surround allowing robots to make autonomous decisions.

NOTE: This activity asks students to consider tough decisions about the behavior of self-driving cars when encountering life-threatening scenarios. If you think this topic may be upsetting for students in your class, you may want to skip the Moral Machine simulation or the Ethics Around AI Motion Planning activity.

1. To introduce the topic, project the video "**The ethical dilemma of self-driving cars-Patrick Lin.**" Then discuss the following as a class.
 - In the situation described, would you prioritize your safety over everyone else's by hitting the motorcycle? Would you minimize danger to others by not swerving, even if you would hit the large object and potentially die? Would you take the middle ground by hitting the SUV since it's less likely the driver will be injured?
 - In comparison to what you would do, what should a self-driving car do? Think about if you were the motorcyclist or the SUV driver.
 - What is the difference between a "reaction" (human driver's split-second response) and a "deliberate decision" (driverless car's calculated response)?
 - Programming a car to react in a certain way in an emergency situation could be viewed as premeditated homicide. Do you think this is a valid argument? Why or why not?
2. Direct students to imagine that they are hired to help program the navigation rules for a new self-driving car. Ask them what rules they might consider giving the car. Then introduce them to the activity **Moral Machine**, which is a platform for public participation in and discussion of the human perspective on machine-made

moral decisions. When students click the red “Start Judging” button, they will be given 13 randomly selected scenarios to evaluate and respond to. After completing all 13 scenarios, students can review their results and how their responses compare to others who have participated in the simulation. Wrap up the activity by discussing the following as a class.

- The Moral Machine simulation represents one way that engineers could train AI robots to make decisions—by crowdsourcing the public’s decisions and having the AI act accordingly. What do you think would be the pros and cons of that method of decision-making? What would happen if people responded in ways that negatively affect some groups of people more than others? What might be other ways to train an AI to make these tough decisions?
- How does it make you feel to know an AI might be making these decisions instead of a human? If you found out that having an AI make these decisions resulted in safer roads and workplace environments, would that change your opinion?

CULMINATING PERFORMANCES

Activity 5: Programming an AI-Powered Robot

In this culminating performance, students will work in pairs or small groups, using an **iterative design process**, to define a problem that can be solved with an AI-powered robot. They will develop a small-scale simulation of the solution with the class robot.

1. Tell students that they will work in pairs or small groups to develop an AI-powered robotic solution to a real-world problem.
2. Direct students to brainstorm, research, and define a problem that might be solved with an AI-powered robot. For example, a sheep farmer may want an alternative to using sheepdogs to herd sheep, or a visually-impaired user may want an automated way to know if there are obstacles ahead. Ideally, students should identify a problem in their own home, school, or community.
3. Have students develop a solution that uses sensors for perception and involves a motion planning algorithm for navigation. For example, a robot sheepdog might gather stray sheep, or a robot guide dog might notify their owner if there is an obstacle in the path ahead.
4. Using the class AI-powered robot and its programming interface, students will develop a small scale simulation as a prototype of the solution, then test and iterate their solution. As part of the prototype, they should be able to describe what role the AI plays in their solution. For example, if your students are using a **Loona** robot:
 - A robot sheepdog might collect a ball representing a sheep and bring it back to one location representing a sheep pen. The AI would perform visual recognition to identify whether a particular ball is or is not a sheep as well as motion planning to navigate around the map and avoid obstacles.
 - A robot guide dog might navigate a designated path through a room and sound an alarm if there are obstacles detected in the path. The AI combines computer vision with motion planning to navigate around the room and avoid obstacles.

5. Have students present their final simulations, explaining the problem, solution, and role of AI. If possible, invite community stakeholders who might be impacted by these problems and solutions to watch and provide feedback on the presentations.

Activity 6: Reflect

In this activity, students should discuss the following questions to reflect on their learning and consider the societal impact of AI on robotics.

- What do you think would be the impact of having an AI-powered robot completing the task in your simulation, as opposed to having a human (or animal) complete the task?
- What parts of your project (information that needed to be perceived or tasks that needed to be performed) were easy for the AI-powered robot to accomplish? What parts were difficult?
- What ethical dilemmas might arise if an AI-powered robot is used to solve the problem you identified or complete the task you simulated?

Extension

Following are three ways to expand students' exploration of the use of AI in robotics:

- If this project piqued students' interest in the capabilities of AI-powered robots, have them research what companies and organizations are currently doing with the technology. An article like "[27 AI Robotics Companies Driving Innovation](#)" would be a good starting point. To continue building on what they have already learned, students should ask research questions like: What is the purpose of the project or technology? What is the AI doing? What types of sensors are involved in the AI's perception? What type of reasoning is the AI performing? What are the ethical implications and societal impacts of the AI project or technology?
- Students can take their ability to identify and solve problems with AI-powered robots to the next level by competing in the [World Artificial Intelligence Competition for Youth](#).
- You can also extend students' thinking about the ethics and societal impact of using AI-powered robots through the "Laws of AI" project found in *Hands-on AI Projects for the Classroom: A Guide for Secondary Teachers*.



We designed this project to make sure it would work cooperatively with teachers' existing AI robotics hardware platforms of choice and to allow students to explore real-world AI robotics solutions.

— Joe Mazzone, Computer & Software Engineering Teacher,
William M. Davies, Jr. Career and Technical High School



PROJECT 4

Machine Learning as a Service

As artificial intelligence technologies become more integrated into many apps and websites used in daily life, the demand for software developers who understand and can use machine learning tools is increasing.

Project Overview

In this project, students explore industry tools and applications of AI through the lens of Machine Learning as a Service (MLaaS). Students gain hands-on experience by experimenting with demos of several image recognition services. Then they develop their own image recognition machine learning model using Teachable Machine.

SUBJECT

Computer science

ESTIMATED DURATION

4–5 hours

TARGET GRADES

8–12

OBJECTIVES

At the end of the project, students will be able to:

- Explain the benefits of MLaaS.
- Describe how an MLaaS image recognition tool works and how it might be used to solve a real-world problem.
- Develop a practical machine learning model using an MLaaS tool.

VOCABULARY

deep learning
face detection
facial analysis
image recognition
machine learning

Machine Learning as a Service (MLaaS)
model (machine learning)
neural network
object detection
transfer learning

STANDARDS

ISTE Standards for Students

1.1. Empowered Learner

- d. Students understand the fundamental concepts of technology operations, demonstrate the ability to choose, use and troubleshoot current technologies and are able to transfer their knowledge to explore emerging technologies.

1.4. Innovative Designer

- c. Students develop, test and refine prototypes as part of a cyclical design process.

1.5. Computational Thinker

- b. Students collect data or identify relevant data sets, use digital tools to analyze them, and represent data in various ways to facilitate problem-solving and decision-making.

ISTE Computational Thinking Competencies

5.1. Computational Thinking

- e. Recognize how computing and society interact to create opportunities, inequities, responsibilities and threats for individuals and organizations.

5.2. Equity Leader

- e. Communicate with students, parents and leaders about the impacts of computing in our world and across diverse roles and professional life, and why these skills are essential for all students.

AI4K12 Five Big Ideas in AI

3. Learning

Computers can learn from data.

5. Societal Impact

AI can impact society in both positive and negative ways.

CSTA K-12 Computer Science Standards

2-DA-09: Refine computational models based on the data they have generated.

2-IC-20: Compare tradeoffs associated with computing technologies that affect people's everyday activities and career options.

3A-AP-13: Create prototypes that use algorithms to solve computational problems by leveraging prior student knowledge and personal interests.

3A-IC-24: Evaluate the ways computing impacts personal, ethical, social, economic, and cultural practices.

3B-AP-09: Implement an artificial intelligence algorithm to play a game against a human opponent or solve a problem.

3B-IC-26: Evaluate the impact of equity, access, and influence on the distribution of computing resources in a global society.

Preparation

MATERIALS

- Computer(s) or tablet(s) (one per student or group) with internet connection for accessing tools and resources online.
- Teacher computer and projector.
- One **KWL chart** per student. Students may create their own, or you can distribute premade charts.
- Digital stock photographs. Openly licensed stock images can be found through various online sources, such as **Search Creative Commons**, **Pics4Learning**, and **Kaggle's computer vision datasets**.

SUPPORTING RESOURCES FOR EDUCATORS

- Article: "**Teachable Machine From Google Makes It Easy To Train And Deploy ML Models**"
- Article: "**What is Machine Learning as a Service (MLaaS)?**"
- Article: "**What is Deep Learning?**"

ADVANCED PREPARATION

- Familiarize yourself with each of the MLaaS platforms used in the project.
- If students will be using Amazon Rekognition or **Runway ML** during their exploration of image recognition tools, you should establish **AWS Educate** and/or RunwayML accounts and become familiar with their platforms ahead of time. RunwayML would also need to be downloaded onto student computers.
- Prepare a set of stock photographs in at least two categories to use for training data and testing data while demonstrating **Teachable Machine** in Activity 4.

Instructions

GETTING STARTED

Activity 1: KWL Chart

In this activity, students will use a **KWL chart** to reflect on what they know and want to know about the **machine learning** topics in this project. They will return to this chart later in the project to add what they have learned.

1. Display a KWL chart for the class. Post the phrase "machine learning" at the top. Explain how a KWL chart is used.
2. Pass out a KWL chart to each student, or have students draw their own.



PROJECT 4

Machine Learning as a Service

3. Direct students to work individually to fill out the K column—what students know—by listing facts, terms, or ideas that they know about machine learning. Then have students share some examples of what they wrote, and add them to the displayed class chart.
4. Next, have students fill out their individual W column—what students want to know about machine learning. Then have students share some examples of what they wrote and add them to the displayed class chart.
5. Let students know that they will be learning about machine learning in this project and will return to these charts throughout the project to see what they have learned.

TAKE A CLOSER LOOK

Activity 2: What Is Machine Learning as a Service (MLaaS)?

In this activity, students learn about the purpose and applications of **Machine Learning as a Service (MLaaS)**.

1. Tell students that people, organizations, and corporations all over the world are solving problems in new and innovative ways due to the availability of AI tools. If students are unfamiliar with AI and machine learning, provide students with a high-level introduction to those concepts using resources from Appendix A: Unpacking Artificial Intelligence.
2. Project from 00:49–03:09 of the video “**Intro to Machine Learning (ML Zero to Hero—Part 1)**.” Following the video, emphasize that machine learning can solve problems that may be too complex to code by hand, or machine learning can lessen the amount of time it takes to write a program with complex rules. This is because, instead of programming all of the possibilities and solutions for a situation, using machine learning, you can show an AI lots of examples and let it identify the rules or patterns on its own. The AI uses the rules or patterns to create a machine learning **model** that it then uses to make predictions about novel data.
3. Share the story of Makoto Koike, a Japanese systems designer and farmer who developed an AI-powered solution to sort specialty cucumbers for his family farm. Resources: “**How a Japanese Cucumber Farmer Is Using Deep Learning and TensorFlow**,” “**TensorFlow powered cucumber sorter by Makoto Koike**.”
4. Explain to students that in the early days of machine learning, the development of AI tools was restricted to professional computer scientists, but now there are machine learning services available that allow people from diverse backgrounds to develop AI solutions. Machine learning as a service (MLaaS) is a term that describes a variety of cloud-based machine learning platforms, such as image analysis, recommender systems, chatbots, or translation. These platforms can be integrated into apps, websites, and other applications. MLaaS allows users to quickly get started with machine learning solutions through code-free, user-friendly interfaces and pre-trained models. Major MLaaS platforms include Amazon Machine Learning, Microsoft Azure Machine Learning, Google Cloud Machine Learning, and IBM Watson Machine Learning.



Activity 3: Experimenting with MLaaS Tools

In this activity, students will learn about machine learning through hands-on experience with two or more industry tools.

1. Have students experiment with two or more of the following MLaaS solutions, providing whole demonstrations as needed. Each of these solutions provide **image recognition** services such as **face detection** (AI that identifies whether there is a face in an image), **object detection** (AI that detects what type of objects are in an image), and **facial analysis** (AI that detects facial details like gender, age, or emotions). Tell students that these are examples of machine learning models that are available for people to integrate into apps or programs that they create. Have students explore the following.

NOTE: To keep from sharing personally identifiable information, students should only upload stock photos, such as those found at [Search Creative Commons](#) and [Pics4Learning](#).

- **Google Vision API.** Have students navigate to the Google Vision AI page and scroll down to the Try the API section where they can experiment with a demo of the Vision API. Have students upload an image from their computer or browse online and add at least one stock photograph to see how the machine learning model analyzes various aspects of the image (e.g., face or object detection) and displays the results.
- **Microsoft Azure's AI Vision.** Have students navigate to the Vision Studio page to find a collection of demos allowing them to experiment with the Azure AI vision tools. Students can explore how the various tools perform with image recognition tasks like face detection or image analysis using the provided images. Some demos, such as "Detect common objects in images", also allow students to upload stock photographs.

Optional: Two additional MLaaS tools can be used by students for image recognition, but these require accounts and/or downloads: Amazon Rekognition and RunwayML. Accessible through the Amazon Web Services console and free through [AWS Educate](#), Amazon Rekognition provides user-friendly demos of their image recognition services like face and object detection. Through the downloadable platform [Runway ML](#), students can look through pre-trained models to find those that perform face or object detection, such as Face-Landmarks. Test the models by uploading an image for analysis.

2. After students have tried at least two of the AI tools, hold a class discussion to compare and contrast the features and interfaces of the different services. Ask students to identify ways that different MLaaS tools—such as face detection, object detection, and facial analysis—can be used in applications.
3. Have students return to their KWL charts and list things that they have learned in the L column. Then have students share some examples of what they wrote and add them to the displayed class chart. As students share, review key concepts and correct any misconceptions that arise.



Activity 4: Building on an MLaaS Tool

In this activity, students will see how an MLaaS tool can be used to create a machine learning model that performs a custom task.

1. Introduce students to **Google's Teachable Machine** and demonstrate how the tool works by creating an image project. Simple tutorials for gathering examples and training the model can be found on the homepage. Model for students how they can find and use stock photographs for their image data.

NOTE: We recommend that you use stock photos for this activity. If you choose to use your webcam or personal photographs, please check Teachable Machine's terms of use and privacy policy against your school/district student data privacy policy to ensure the application complies with that policy.

2. Your introduction and demonstration should emphasize these key machine learning concepts:
 - Teachable Machine provides a user-friendly platform for training image recognition models. It is based on an open-source machine learning platform called TensorFlow.
 - TensorFlow uses neural networks for deep learning. **Neural networks** are modeled after the human brain. While a brain uses neurons and synapses to process data, neural networks use layers of nodes with directed connections. **Deep learning** algorithms use many layers of nodes to progressively identify both lower- and higher-level features in the input. For example, in image recognition, lower layers might identify pixel features related to the specific classes, such as faces or objects. Some of these connections are more important than others, so they have more weight in determining the outcome.
 - Just like people, machines learn through experience. As a machine processes a set of data, it recognizes patterns, assigns more weight to the most important information, learns to process inputs in order to develop the most accurate outputs, and creates a model from which to make future predictions or decisions.
 - Teachable Machine's Image and Pose Projects build on top of a Tensorflow neural network that has been pre-trained to process images. Teachable Machine uses **transfer learning** to apply the new classes and data the user inputs as the last layer or step of the neural network.
 - Just like the other MLaaS tools that were examined, these models can be trained and customized more effectively, efficiently, and cheaply than developing a neural network from scratch.
 - The step-by-step process for creating a model in Teachable Machine is: create an Image or Pose Project, define classes, upload or input several image samples for each class, train the machine learning model, test the model with new images to see how accurately the model is able to correctly recognize and classify novel images, and iterate to improve the model.
3. Have students return to their KWL charts and list things that they have learned in the L column. Then have students share some examples of what they wrote and add them to the displayed class chart. As students share, review key concepts and correct any misconceptions that arise.



CULMINATING PERFORMANCES

Activity 4: Develop a Solution Using MLaaS

In this culminating performance, students ideate, develop, test, and refine a machine learning model that uses image recognition to address a real-world challenge.

1. Have students work individually or in pairs to consider possible image recognition applications, such as the ones listed below, and select one to develop.
 - Recognizes specific hand symbols, such as those for rock, paper, and scissors.
 - Recognizes objects in images, such as endangered species.
 - Recognizes facial expressions for emotions, such as happiness and sadness.
 - Recognizes different physical poses, such as yoga poses.
2. Then they should use Teachable Machine and an **iterative design process** to create an Image or Pose Project, define classes, upload or input several image samples for each class, train the machine learning model, test the model with new images to see how accurately the model is able to correctly recognize and classify novel images, and iterate to improve the model.

NOTE: We recommend that students use stock photos for this activity. If you would like to allow students to use their webcam or personal photographs, students should still refrain from using their faces, and you should check Teachable Machine's terms of use and privacy policy against your school/district student data privacy policy to ensure the application complies with that policy.

3. Once students have created their model, each student or pair should demonstrate their working model for the class, explain how the model works, and describe one possible real-world application for the model.

Activity 5: Reflect

In this activity, students should discuss the following questions to reflect on the societal impact of MLaaS.

- How do you think the availability of MLaaS will impact software development?
- In what way might these tools democratize the development of AI applications? In what ways might they centralize data with large corporations?
- How do you think that MLaaS will impact the job market in the next ten to twenty years? How might you learn more about pathways to careers in MLaaS?



Extensions

Following are four ways to expand students' exploration of MLaaS.

1. Students can extend their understanding of MLaaS by integrating Teachable Machine models into programming activities. For example, students can use their models in Scratch block-based programs using the Teachable Machine extension in MIT's **PoseBlocks** platform or using the **Tensorflow integration** in the Machine Learning for Kids Scratch platform. Additionally, if students have access to an Arduino and a laptop with a webcam, they can take their experimentation with Teachable Machine further through the **Google Tiny Sorter** physical computing experiment. Similar to the cucumber sorter presented in this project, this experiment uses object detection to identify, classify, and physically sort cereal and marshmallows.
2. To provide students with a fuller understanding of deep learning neural networks, you can share the "**But What Is a Neural Network? Deep Learning, Chapter 1**" video by 3Blue1Brown. Additionally, students can advance their understanding of neural networks and deep learning by using **Google's TensorFlow Playground**. Have students try each of the four datasets, progressively advancing from simpler classification data to the most complex classification data, represented by the spiral. Even with the most basic dataset, students should be able to observe the images of what each neuron is outputting and which neurons are given more weight (based on the thickness of the lines) as the model is trained. As the data gets more complex, students will need a combination of more hidden layers, more neurons per layer, more feature property input and/or more training time (epochs) to accurately train the model. Students should adjust settings to see how quickly they can get the model to match the data.
3. Advanced students with experience coding in Python (or a similar text-based language) can work directly with Google's TensorFlow platform and the Keras API to build and train a neural network that performs image recognition based on the MNIST database.
 - Start by sharing these TensorFlow videos, which introduce the basic idea of training a neural network with Python and TensorFlow: "**Intro to Machine Learning (ML Zero to Hero—Part 1)**" and "**Basic Computer Vision with ML (ML Zero to Hero—Part 2)**."
 - Then have students work through the "**Basic classification: Classify images of clothing**" tutorial. This tutorial provides students with an interactive notebook and step-by-step instructions for building, training, and evaluating a neural network that classifies images. An overview of all TensorFlow tutorials can be found at www.tensorflow.org/tutorials.
4. Some students may want to spend more time exploring industry AI tools and developing the skills needed for machine learning careers. MLaaS providers often offer tutorials and training programs for potential developers. For example, students who want to further pursue **Google's machine learning tools** can use their free developer courses and guides. Students ages fourteen and up who are interested in further exploring Amazon's machine learning services can sign up for a student account at **AWS Educate** to access training materials and a wider array of machine learning tools. And IBM offers several courses and resources about using their AI tools on their **SkillsBuild students** page.



Glossary

AI agent: an entity that uses sensors and actuators to autonomously act on its environment and achieve goals.

artificial intelligence (AI): the science and engineering of creating computer programs that can imitate human intelligence.

artificially intelligent robot (AI robot): a robot that is able to use sensors to collect information and make autonomous decisions about how to complete a task even in a changing environment.

autonomous: having the capacity to act independently or without external control.

behavior tree algorithm: a branching model that controls the flow of decision making through prescribed responses to external cues.

bias: preference for or against an idea or thing.

bot: an AI agent that can interact with computer systems or users (e.g. play video games).

classification model: a mathematical representation of how to categorize data into classes based on common features.

confidence level: the probability that the item has been matched with the correct label.

data: information.

dataset: collection of data.

decision tree: a branching flowchart with nodes, branches, and leaves that symbolically represents a series of tests and classification labels.

deep learning: a machine learning algorithm that uses many layers of nodes to progressively identify both lower- and higher-level features in the input.

ethical: morally right.

face detection: AI that identifies whether there is a face in an image.

facial analysis: AI that detects facial details like gender, age, or emotions.

feature: unique measurable property.

Finite State Machine (FSM) algorithm: a relatively simple AI model that prescribes behaviors for each state in a finite, specific list of all possible states a bot or non-player character can experience in a game.

generative AI: a type of machine learning algorithm, such as a generative adversarial network (GAN) or a generative pre-trained transformer (GPT), that can generate new data points (e.g., text, images, video) based on its training data.

image recognition: the ability of a computer program to analyze the pixels in an image and identify objects, people, or other subjects.

label: what a machine learning model is trying to predict, such as a class, category, or value.


machine learning (ML): a subset of AI involving the study of algorithms and models that machines use to perform a task without explicit instructions.

Machine Learning as a Service (MLaaS): a variety of cloud-based machine learning platforms, such as image analysis, recommender systems, chatbots, or translation, that can be integrated into apps, websites, and other applications.

model (machine learning): a mathematical representation of a dataset developed by AI.

Monte Carlo Tree Search (MCTS) algorithm: an AI model that uses probabilities of winning playouts from a representation of all possible moves to determine the best next move.

motion planning algorithm: a form of sequential decision making in robotics to support movement and navigation from one point to another.



natural language understanding: an AI technology used to interpret human language.

neural network: a computer system modeled after the human brain that use layers of nodes with weighted, directed connections to learn to perform tasks.

non-player characters (NPC): characters or objects in a game that are not controlled by a human.

object detection: AI that detects what type of objects are in an image.

reinforcement learning: a form of machine learning in which an AI has a clear goal to achieve and learns by receiving rewards or punishments as it makes a sequence of decisions to achieve that goal.

robot: a machine that is able to perform complex tasks automatically.

sampling bias: in machine learning, a preference for or against an idea or thing caused by having a dataset that does not accurately represent the labels.

sensor: a device that allows a machine to perceive the natural world.

sensor fusion: the ability of an AI to combine perception from multiple sensors into one model.

sequential decision making: making a series of decisions.

supervised learning: a form of machine learning in which the trainer provides the AI with labels for each of the items in the training data.

test data: examples used to verify the accuracy of a machine learning model.

training data: examples used to teach a machine learning model.

transfer learning: creating a new machine learning model by tweaking a previously trained neural network.



APPENDIX A

Unpacking Artificial Intelligence

This section provides basic explanations of fundamental AI concepts referenced in the *Hands-On AI Projects for the Classroom* series of guides, along with resources for supporting instruction.

According to John McCarthy, who first coined the term, artificial intelligence is “the science and engineering of making intelligent machines, especially intelligent computer programs” (McCarthy, 2007). A technology powered by AI is capable of such things as using sensors to meaningfully perceive the world around it, of analyzing and organizing the data it perceives, and of autonomously using those data to make predictions and decisions.

AI technologies are sometimes classified as narrow and general AI. Narrow AI makes decisions about a specialized task, sometimes even based on a specific dataset of preprogrammed actions. The DeepBlue chess program that beat a human world champion in 1996, Apple’s Siri, and self-driving cars are all examples of narrow AI. In contrast, general AI could hypothetically learn and adapt to perform any task and solve any problem that a human being can. General AI does not currently exist, but there are many examples of it in fiction, such as “WallE” and Baymax from “Big Hero 6.”

Learn More

Video: “[What is AI?](#)”

Video: “[What is Artificial Intelligence \(or Machine Learning\)?](#)”

Video: “[What’s intelligent about artificial intelligence](#)”

Article: “[What Is Artificial Intelligence?](#)” by John McCarthy

Resource: “[How AI Works](#)”

Resource: “[Glossary of Artificial Intelligence Terms for Educators](#)”

Curriculum: “[AI4ALL’s Open Learning Curriculum](#).” This free curriculum provides activities to teach students what AI is, what types of AI exist, and how to identify AI in the world around them.



How Do I Know If a Robot or Other Technology Has Artificial Intelligence?

Some robots and computer programs have AI, while others do not. A robot or software solution that has AI capabilities can do things such as recognize specific objects or faces, navigate around objects or complex maps on its own, classify or distinguish between objects, interact naturally with humans, understand or speak in a human language, recognize or express emotions, or improvise when encountering something unexpected. In these ways, the autonomous decisions made by AI are more advanced than simple automation of a task (performed a prescribed sequence of steps), which even non-AI robots and software are frequently used for. As the cost of technology decreases and the capabilities of AI technologies increase, we will likely see increased AI use across most devices and software.

Learn More

Article: [“What’s the Difference Between Robotics and Artificial Intelligence”](#)

Article: [“How Robots Work: Robots and Artificial Intelligence”](#)

What Is Machine Learning?

Machine learning, a subset of AI, is the study of algorithms and models that machines use to perform a task without explicit instructions. Machine learning algorithms improve with experience. Advanced machine learning algorithms use neural networks to build a mathematical model based on patterns in sample “training” data. Machine learning algorithms are best used for tasks that cannot be completed with discrete steps, such as natural language processing or facial recognition.

Learn More

Video: [“Intro to Machine Learning \(ML Zero to Hero—Part 1\)”](#)

Video: [“How Does Machine Learning Work? Simply Explained”](#)



How Do Neural Networks Work?

Artificial neural networks are currently modeled after the human brain. While a brain uses neurons and synapses to process data, neural networks use layers of nodes with directed connections. Some of these connections are more important than others, so they have more weight in determining the outcome. Just like people, machines with neural networks learn through experience. As a machine processes a set of data, it recognizes patterns, assigns more weight to the most important information, learns to process inputs in order to develop the most accurate outputs, and creates a model from which to make future predictions or decisions. There are many types of neural networks, each with different design, strengths, and purposes.

Learn More

Video: "[Neural Networks and Deep Learning #3](#)"

Playlist: "[Neural Networks](#)"

Article: "[What Is Deep Learning?](#)"

Resource: "[Overview of GAN Structure](#)"

Article: "[What is GPT?](#)"

What Is Natural Language Processing?

Natural language processing (NLP) is the AI technology used to understand and interact with humans' natural language. NLP powers technologies such as voice experiences and assistants, text predictors, grammar checks, text analyzers (such as spam filters), and language translators.

Learn More

Video: "[Natural Language Processing #7](#)"

Article: "[A Simple Introduction to Natural Language Processing](#)"

Article: "[A Complete Guide to Natural Language Processing](#)"

Video: "[How Do Chatbots Work? Simply Explained](#)"

Article and video: "[What Are Chatbots?](#)"

Article: "[Chatbot vs ChatGPT: Understanding the Differences & Features](#)"

Video: "[How Chatbots and Large Language Models Work](#)"



What Is Generative AI?

Generative AI is a type of machine learning that uses advanced algorithms, such as a generative adversarial network (GAN) or a generative pre-trained transformer (GPT), in order to create new data. Based on what they have learned from training data, generative AI tools can generate text, images, video, music, code, and other types of media.

Learn More

Video: ["Introduction to Generative AI"](#)

Video: ["How Dall-E 2 and Other AI Art Generators Create Images From Text | WSJ"](#)

Article: ["Generative Artificial Intelligence in education: What are the opportunities and challenges?"](#)

Article: ["What Kids Need To Know About Generative AI: Unleash Your Creativity!"](#)

What Types of Ethical Considerations Surround AI?

All AI technologies are developed by humans. Whether they have been preprogrammed with a set of rules, or use training data to learn, they will have bias based on human input and decision-making. It is important that students understand that AI decisions are not objective, as well as to understand which stakeholders might benefit from certain biases in the technologies. Moreover, many AI technologies collect, store, and apply personally identifiable information about users. Students should be aware of privacy concerns related to these technologies.

Learn More

Video: ["Teach AI | Prepare our students for the future"](#)

Video: ["Algorithmic Bias and Fairness #18"](#)

Resource: [UNESCO's "Ethics of Artificial Intelligence"](#)

Report: ["The Ethical Framework for AI in Education"](#)

Article: ["Artificial Intelligence and Ethics: Sixteen Challenges and Opportunities"](#)

Video: ["Do you know AI or AI knows you better? Thinking Ethics of AI"](#) (version with multilingual subtitles)

Video: ["The ethical dilemma of self-driving cars—Patrick Lin"](#)

Video: ["The danger of AI is weirder than you think | Janelle Shane"](#)


Curriculum: ["An Ethics of Artificial Intelligence Curriculum for Middle School Students"](#)

APPENDIX B

Alignment to ISTE Standards and AI4K12 Five Big Ideas in AI

The following tables provide a big-picture view of how the projects in each guide align with the ISTE Standards for Students, ISTE Computational Thinking Competencies, and AI4K12 Five Big Ideas in AI.

Guide	Elementary				Secondary				Electives				Computer Science				Ethics			
Project	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
ISTE STANDARDS																				
1.1 Empowered Learner	x	x					x			x	x	x	x		x	x		x	x	
1.2 Digital Citizen					x			x			x			x			x	x	x	x
1.3 Knowledge Constructor	x		x	x		x	x	x			x		x				x	x	x	x
1.4 Innovative Designer		x	x				x		x	x					x	x			x	x
1.5 Computational Thinker			x	x	x		x		x		x		x	x	x	x	x			x
1.6 Creative Communicator					x	x		x			x			x					x	
1.7 Global Collaborator							x					x	x						x	
5.1 Computational Thinking				x	x	x	x		x	x	x	x	x		x	x	x	x	x	x
5.2 Equity Leader					x	x	x	x							x	x	x	x	x	x
5.3 Collaborating Around Computing	x			x			x					x	x							
5.4 Creativity & Design	x	x	x	x				x	x	x	x			x	x		x		x	x
5.5 Integrating Computational Thinking		x	x				x		x	x				x						x



Guide	Elementary				Secondary				Electives				Computer Science				Ethics			
Project	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
AI4K12 FIVE BIG IDEAS IN AI																				
1. Perception	x	x			x					x		x			x				x	
2. Representation and Reasoning	x		x	x			x		x			x	x	x	x				x	
3. Learning	x			x		x	x				x	x	x	x	x	x	x	x	x	x
4. Natural Interaction	x				x	x				x		x		x	x				x	
5. Societal Impact	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x

Development Team

Authors

Nancye Blair Black

Susan Brooks-Young

Content Contributors

Jared Amalong, Sacramento County Office of Education/AI4K12 Initiative, AI Subject Matter Knowledge

Mark Gerl, The Galloway School, *AI-Powered Players in Video Games*

Joe Mazzone, William M. Davies, Jr. Career and Technical High School, *Programming with Machine Learning, Using AI for Robotic Motion Planning*

Joseph South, International Society for Technology in Education (ISTE)

Other Contributors

Leah Aiwohi, Kauai High School

Susan Forget, Sabin Middle School

David Lockett, Bok Academy

Annie Ning, International Society for Technology in Education (ISTE)

Yolanda Ramos, formerly with International Society for Technology in Education (ISTE)

Emily Reed, International Society for Technology in Education (ISTE)

Jonathan Ringenberg, University of Nebraska Omaha

Cassandra Woodall, formerly with International Society for Technology in Education (ISTE)

Ramsey Young, University of Nebraska Omaha