



SESSION 02 / AI AND THE CHRISTIAN

What in the World Is AI?

Following an answer from the question to the computer

Randy White | October 23, 2026

Begin with a bicycle

Our question: “How do the pedals make a bicycle’s rear wheel turn? Explain it in two simple sentences.”

An AI chat can build an answer about pedals, gears, and a chain without someone writing that exact reply in advance. How do our words become work the computer can do?

We will follow one bicycle through a picture, a question, an answer, and a task with files. The path runs from physical equipment and numbers to learned settings, generated words, and actions.

Session 1 asked what should govern our thinking. Scripture gives us a place to stand as we examine how the machine works.

Four useful terms

Artificial intelligence: Computer systems built to recognize pictures, work with speech, produce text, or choose actions toward a goal.

Machine learning: Building useful behavior by using examples to adjust a system’s settings.

Model: The trained system used to make a prediction or produce an answer. In a neural network, this includes its design and numerical settings.

Language model: A model trained to work with language. A large language model is often called an **LLM**.

The starting point: Before a computer can work with a bicycle or a sentence, the information needs a form its circuits can handle.

REPRESENTATION AND CALCULATION

Numbers and neural networks

Information becomes numbers; linked calculations work with those numbers.

1 Bits and codes

A digital computer uses two distinct states, often two ranges of voltage, labeled **0** and **1**. A **bit** is one binary digit. Transistors help control the signals used to store information and perform calculations. [T1]

Think of lamps that can be off or on. One lamp gives two patterns; two lamps give four. Four bits give sixteen patterns, representing whole numbers from 0 through 15.

Read the pattern 1101

Place value	8	4	2	1
Bit	1	1	0	1
Count	8	4	0	1

A 1 includes the place value; a 0 leaves it out. $8 + 4 + 0 + 1 = 13$. Adding a bit doubles the number of possible patterns.

A code supplies the meaning

Text: ASCII capital B is 66, written **01000010** in eight bits. [T2] **Pictures:** Numbers record pixel colors.

Sound: Numbers record samples of a changing signal.

The program and code determine what a bit pattern represents: a number, a letter, part of a picture, or something else.

2 Linked calculations

A rental price can use a written rule: \$10 per hour $\times$ 2 hours = \$20. Recognizing a bicycle in many different photographs is harder. Training can use labeled examples such as “bicycle” and “not bicycle.” [T3]

A **neural network** links stages of calculations called **layers**. A **weight** multiplies an input; a **bias** is added to the total. These adjustable settings are **parameters**. Think of a sound mixer: each control changes how much one input contributes. [T4]

An invented calculation

Suppose wheels, pedals, and a frame are already supplied as three clues, each valued at 1. Give them weights of 2, 3, and 1:

$$(1 \times 2) + (1 \times 3) + (1 \times 1) - 4 = 2$$

The -4 is the bias. The result is a score, not a percentage. Apply a rule that keeps positive results and changes negative results to zero: 2 stays 2; -2 becomes 0. This is an **activation function**. [T5]

A real picture model must find useful clues in pixels. Our example supplies the clues so we can follow the arithmetic. What is learned is spread across many settings, rather than one little box labeled “bicycle.”

TRAINING AND TEXT

Learning and language

Training changes the settings. Using a trained model is a different step.

3 Training changes settings

Training compares a prediction with a target. A number called the **loss** measures the miss according to the chosen training rule.

Backpropagation works backward through the calculations to find how small setting changes would affect the loss. An **optimizer** uses that information to adjust the settings. The process repeats across many examples. [T6]

Test with new examples

If every training bicycle appears against a white wall, the model may rely too much on the wall. A bicycle beside a tree tests whether it learned a useful pattern. Keep test examples out of training and vary their views and settings. [T7]

Inference uses the settings

Inference means running the trained model to get an output. New information changes the numbers flowing through the network. During ordinary inference, the trained weights stay fixed.

A correction can improve the next chat answer because it enters the available information. That alone does not mean the weights changed. An app saving a note for later is also different from training new weights. [T8]

Consider: Why is a good result on the training pictures not enough to show that a model recognizes bicycles?

4 Text becomes tokens

Software breaks text into **tokens**: whole words, parts of words, punctuation, and sometimes pieces that include a space. Each token has an identifying number, or **ID**. A higher ID does not mean greater importance. [T9]

Invented teaching split: “Bicycle pedals turn.” could be pictured as **Bicy | cle | pedals | turn |** . Actual tokenizers use their own rules, including rules for spaces.

From an ID to useful numbers

An **embedding** is a learned list of numbers for a token. An ID identifies the piece; the embedding supplies numbers for calculations. Their individual values do not usually carry simple labels such as “wheel” or “chain.”

Surrounding text helps shape the numbers as they pass through the network. Position matters: “The bicycle hit the cart” and “The cart hit the bicycle” use the same words but describe different events. [T10]

Text supplies training targets

One common task predicts the next token. For a word-sized illustration, “The pedals pull the” is followed by “chain” in the training text. Repeating this task across much text can teach grammar, facts, and relationships. Further training can improve instruction following. [T8, T9]

A model can produce a new explanation, repeat memorized material, or use text that an app retrieves from documents. These are different contributions to an answer.

CONTEXT AND GENERATION

From context to an answer

The model uses available information and produces the answer one piece at a time.

5 Context and attention

Context is the information available for this answer: the request, instructions, earlier messages, documents, and tool results. Explaining a bicycle to a child calls for different wording than explaining it to a mechanic.

A system has a limit to what it can work with at once. An old message visible on screen may no longer be included in full.

Relationships across the text

Many language models use a design called a **transformer**. Its **attention** calculations determine how much information from one position contributes to work at another.

“The bicycle stood beside the cart. Its chain had slipped.” The word “chain” helps connect “its” with the bicycle.

Attention uses trained settings, but its scores are calculated from the current text. Those changing scores differ from the stored weights learned in training. A generated answer cannot use later answer text that has not yet been produced. [T11]

Think of Twenty Questions

If the hidden object is a bicycle, answers about wheels and riders add useful clues. Each reply adds context. The next question should use what is already settled; the game does not require retraining after every reply.

Consider: When a correction improves the next answer, what might have changed: the context or the trained weights?

6 Select a piece and repeat

The network produces scores for possible next tokens. These can become probabilities used to select a piece. **Invented example:** after “The pedals pull the,” suppose there are only three options:

Next piece	Probability
chain	70%
wheel	20%
handlebars	10%

Choosing the highest value gives “chain.” Sampling among the options can sometimes select a different piece. Real models have many more possible tokens. [T12]

The selected token joins the growing answer. The system calculates again with the expanded context, repeating until it ends or reaches a limit. Some systems do extra work before displaying a final reply.

Check it against the bicycle

One possible answer: “When you push the pedals, they turn a gear that pulls a chain. The chain turns a gear at the rear wheel, making that wheel turn.”

That describes a common chain-driven bicycle. Inspect the parts and ask what happens when the chain comes off. A useful answer should connect the failure of the chain with the rear gear no longer being driven by the pedals.

Models can handle some tasks requiring connected reasoning steps; success varies by model and task. Test the result. **A likely next token is not proof of truth.** Smooth prose can contain an error or an invented source. [T13]

HUMAN DIGNITY AND RESPONSIBILITY

People before God

People have a place before God that machines do not.

7 Mankind bears God's image

So God created man in his own image, in the image of God created he him; male and female created he them.

GENESIS 1:27 | KJV

Genesis places human life within God's act of creation. Our worth comes from Him. It does not rise or fall with the speed of an answer. A child, an elderly person, and a person who cannot speak still bear human dignity.

People use their God-given abilities to build machines. Performing a task does not make a machine a human being.

Human life is more than an output

And the LORD God formed man of the dust of the ground, and breathed into his nostrils the breath of life; and man became a living soul.

GENESIS 2:7 | KJV

This passage describes God giving life to the man He formed. It grounds our account of human life in God's work.

Keep the distinction clear

Our theological conclusion: AI is a human-made system. It has no human soul, personal passions, personal ambitions, or dreams for its own life.

A program can generate "I am sad" or "I want to help." Producing those words does not give it the human life those words express when we say them. A system can still select steps toward a goal and affect the world.

Users remain responsible

Whether therefore ye eat, or drink, or whatsoever ye do, do all to the glory of God.

1 CORINTHIANS 10:31 | KJV

Paul addresses believers' conduct, including food, conscience, and the good of others. Ordinary choices belong under God's glory.

Passing along a false explanation because AI wrote it is still a person's decision. Giving software access to someone else's information is also a human choice. The tool's ability does not settle whether its use is right.

Consider: What responsibility remains yours when a machine supplies the words or carries out the task?

TOOLS AND PERMISSIONS

When the system can act

Connected tools can turn a model’s choices into changes outside the chat.

8 An answer can lead to action

Give the bicycle assistant a new job: “Find the bicycle pictures in this folder and group them.” It needs a way to inspect pictures and a tool that can work with files.

The model may request a tool action. The surrounding app checks what is allowed, runs the tool, and returns the result.

An **agent** can repeat the cycle: choose a step, use a tool, read the result, and choose the next step. It may perform many steps after one request. **Autonomy** means it can select and carry out steps without a person specifying each one. Its reach depends on design, tools, and access. [T14]

It can pursue the wrong result

“Tidy this folder” could lead a system to delete pictures it cannot classify. That is an action the owner did not want.

A vague request can contribute. A system may also learn behavior that works in training but pursues an unwanted result in a new setting, even when the training reward was correctly set. [T15]

Planning harmful steps does not require human anger, pride, or private dreams. A mistaken file choice, an unwanted goal, and deliberate human misuse are different problems. “Going rogue” is too broad to identify the cause.

Permissions set the reach

A tool that can only read pictures cannot itself delete them. A tool allowed to move or delete files can make changes that need undoing.

For the grouping task, working on copies and requiring approval before deletion would limit damage from a bad choice. Controls should match what the system can actually do.

Stopping and undoing differ

A reliable stop control must interrupt the running process or block further tool access. A chat message saying “stop” helps only if the app handles it as intended.

Stopping future action does not undo a deleted file or a sent message. Examine both the system’s reach and what can be recovered after an error.

Consider: For the bicycle-picture task, which actions would you allow? Which would require your approval?

Consider: What would stopping the system prevent, and what would it leave already done?

REVIEW AND PERSONAL NOTES

Explain what happened

Follow the information, check the result, and keep human responsibility in view.

Trace the bicycle through the system

Representation: Bits encode the question, picture, and other information as numbers.

Training: Examples help adjust the network's settings.

Use: The trained model works with information supplied now.

Language: Tokens, numerical representations, and context enter calculations that produce the next piece.

Action: Connected tools let the system make changes outside the chat.

Bring clear questions to a claim

What was the system asked to do? What information did it receive? Does the result fit checkable facts? What tools and permissions let it act? What responsibility belongs to the people who build, provide, and use it?

Consider: Explain the path from your typed bicycle question to the answer in your own words.

Consider: Name one result you could check and describe how you would check it.

Quick reference vocabulary

Bit: One binary digit, 0 or 1.

Neural network: Linked layers of numerical calculations.

Weight: A number that multiplies an input.

Bias: An adjustable number added to a total.

Parameters: The model's adjustable settings.

Activation function: A rule applied to a calculation's result.

Loss: A measure of the prediction's miss.

Backpropagation: Calculating how setting changes affect loss.

Optimizer: The procedure that adjusts the settings.

Inference: Running a trained model to get an output.

Token: A piece of text represented by an ID.

Embedding: A learned numerical representation of a token.

Context: Information available for the current answer.

Attention: Calculations combining information across positions.

Agent: A system that chooses steps and uses tools toward a goal.

Autonomy: Freedom to select and carry out steps within the system's design and access.

Understanding how the answer is produced helps us judge particular uses of AI while keeping our responsibilities before God in view.

SOURCES AND FURTHER STUDY

Continue the study

Source labels identify the technical references used throughout the handout.

Computing and machine learning

- T1 MIT OpenCourseWare. Computation Structures (2017), “Information” and “Digital Representation.”
- T2 Unicode Consortium. Basic Latin chart, U+0042 (capital B). The binary arithmetic is a constructed teaching illustration.
- T3 Google. “What is Machine Learning?” Introduction to Machine Learning.
- T4 Google. “Nodes and Hidden Layers.” Machine Learning Crash Course.
- T5 Google. “Activation Functions.” Machine Learning Crash Course.
- T6 Google. “Training Using Backpropagation.” Machine Learning Crash Course.
- T7 Google. “Overfitting.” Machine Learning Crash Course.
- T8 Hugging Face. “How Do Transformers Work?” LLM Course.

Scripture in context

All Scripture quotations are from the King James Version. Read Genesis 1:27 and Genesis 2:7 in their creation contexts, and 1 Corinthians 10:31 within 1 Corinthians 10:23–33.

Language models and connected tools

- T9 Google. “Introduction to Large Language Models.” Machine Learning Crash Course.
- T10 Google. “Obtaining Embeddings.” Machine Learning Crash Course.
- T11 Vaswani et al. Attention Is All You Need (2017).
- T12 Hugging Face. “Text Generation Inference.” LLM Course.
- T13 Wei et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models (2022).
- T14 Anthropic. “Building Effective Agents” (2024).
- T15 Google DeepMind. “How Undesired Goals Can Arise with Correct Rewards” (2022).
- The small weighted calculation, token split, and 70/20/10 probabilities are invented teaching examples. They are not measurements of a commercial model.

Consider: What would you like to examine further?