

Beyond Fluency: AI Literacy and AI Capability

Giorgio Natili

Introduction to the Whitepaper

When we set out to map how the world is learning to live with AI, I wanted voices that stand with one foot in the engine room and one in the classroom. Giorgio Natili is exactly that. By day he builds AI systems at enterprise scale. He also sits on panels in front of first-generation students, veterans and people retraining after layoffs, and answers the simplest and hardest question of all: where do I start?

In *Beyond Fluency*, Giorgio names something many of us feel but rarely say out loud: being able to use a tool is not the same as being able to judge it. Fluency, literacy and capability are three different things, and confusing them is expensive, for individuals and for organisations. His answer is practical, free of product pitches, and honest about what these tools cannot do.

This paper is Giorgio's contribution to the Global AI Literacy Project 2026. His insights form Chapter 4 of the Global AI Literacy Study, which we will present on 22 October 2026 at the House of Lords in London. I am grateful for his generosity and clarity, and I hope this paper gives you what it gave me: a sharper sense of where judgment begins.

Anastassia

Global AI Literacy Project 2026/ CEO and Founder of AI Edutainment GmbH

About the author



Giorgio Natili is Senior Director and Head of AI Engineering at Oracle, where he leads engineering teams across the United States, EMEA and South Asia. Over more than 25 years he has held engineering leadership roles at Amazon, Capital One and Mozilla. He serves as Vice President of ISSNAF, the Italian Scientists and Scholars in North America Foundation, and as an invited adviser to the United Nations Department of Economic and Social Affairs on data and AI sovereignty. He is the author of four books on software development and AI, holds two U.S. patents in AI-driven data protection, and chairs an advisory panel at Tacoma Community College. Born in Rome, he is based in the United States.

Beyond Fluency: AI Literacy and AI Capability

Giorgio Natili

Oracle Cloud Support Head of AI Engineering, Vice President ISSNAF, and
United Nations Adviser on Sovereign AI

ORCID ID: 0009-0000-6342-3922.

1. Before the Tools: Why AI Literacy Comes First

AI literacy matters because the tools arrived before our understanding of them did. Most working people now have a capable AI model, the trained system behind a chatbot, a click away, and many use one every day. Few have been shown what these systems are, what they can and cannot do, or how to tell when an answer deserves their trust. That gap determines who gets more out of the tools, who is quietly misled by them, and who can recognize when a smaller, cheaper tool, or no AI at all, is the better choice.

Literacy is the ability to make those calls, and it is not a destination. The tools change every few months, and what you learn about one product becomes less relevant as the products change. What lasts is the ability to understand their limits, evaluate what comes back, and recognize when the tool is the wrong choice. The name for that ability is **judgment**, and it is the concept this paper keeps coming back to.

I saw the gap clearly this spring at Tacoma Community College, a public college south of Seattle, Washington, whose students include first-generation students, veterans, people retraining after layoffs, and professionals updating skills that have begun to age. I was one of several panelists, close enough to see how the answers landed.

The students weren't skeptical about AI. They wanted to understand how to use it effectively and what they needed to learn to remain competitive. Most were already using a chatbot, mainly as a slightly better search engine. What they were missing was an understanding of the layer underneath: what the model is actually doing, why it can return a confident but false answer, and why the same prompt can produce different results. They had access to AI, and many could use it, but they had little basis for judging its output or deciding how much to trust it. Then the questions came, and from very different starting points, they converged on the same one: *What should I learn to be competitive, and where do I start?*

That sounds like a question about skills. But answering it requires separating three ideas that are often treated as interchangeable: fluency, literacy, and capability. *Fluency* is being able to operate a tool, to give it instructions and get something useful back. *Literacy* is understanding enough about the tool and its limitations to evaluate what it produces and decide whether, when, and how it should be used. *Capability* is what you can actually do with it, from using a chatbot to designing a system around one.

The three overlap, but they are not the same. A person can become increasingly capable with AI without developing equivalent judgment about where and how that capability should be used. Conversely, understanding AI does not mean everyone needs to learn how to build or architect AI systems. How much capability you need depends on what you need to do. Literacy determines whether you can make that choice intelligently.

There is no shortage of material about AI. The problem is finding a path through it that does not assume technical expertise or lead back to a particular product. What is missing for many learners is a path clear enough to start now, honest about what the tools cannot do, and useful even if they never become AI specialists.

A panel gives a question like *Where do I start?* only a few minutes. The rest of this paper is the longer answer: first, what it means to become literate enough to judge these systems, and then how far you may choose to develop the capability to use, master, build with, or architect them.

2. The Trap

My first answer to the students was the obvious one: start with the tools you already have and learn to use them well. That drew a follow-up almost immediately. One participant said the tool they had wasn't giving them much to work with. The answers were wrong often enough that they had stopped trusting it, so what exactly was there to learn?

The question conflated two things: the tool's capability and the user's capability.

The tool does matter. Free tiers may provide access to a less capable model or impose tighter limits on context, memory, or features. Paying can give you a more capable system and more room to work with it. But that distinction is easy to overcome.

A subscription buys you a more capable instrument, not a more capable operator.

The other part of the answer became more interesting. Even a limited model can perform differently depending on the context and instructions it receives. One simple way to discover this is to ask the model to improve or the question you are about to give it, a practice called meta-prompting. The technique itself is a matter of fluency, not literacy. The more important realization is what it reveals: the quality of an AI response depends not only on the model, but also on the context and instructions the user provides.

When I explained this, the room went quiet. It wasn't the quiet of people who were lost; it was the quiet of people realizing that part of what they had attributed to the tool was actually part of the interaction with it.

That distinction is the trap. When AI works badly, an inexperienced user may conclude that the model is simply bad. When it works well, the same user may conclude that the model understands more than it actually does. In either case, fluency alone provides little basis for explaining the result. The problem is not simply learning how to get a better answer. It is learning enough about the system to understand why its performance changes and enough about its limitations to know when the answer should be trusted.

The same problem appears at organizational scale. A small team may adopt AI for their first automated workflows; a large organization may purchase one of the most capable “frontier” models offered by AI providers. In both cases, it is easy to start with the tool rather than the problem: choosing more capability before defining what the system needs to accomplish, how success will be measured, or the consequences of failure. The result is the same mistake at a different scale—acquiring capability without the literacy needed to evaluate or direct it.

AI makes this distinction unusually important because its interface hides much of what determines its behavior. A chatbot may present little more than a text box, but the answer can depend on the model being used, the context it has been given, what information it can access, and the instructions shaping its response. Learning the interface is therefore not the same as understanding the system behind it. That is why product-specific fluency is useful but fragile. Interfaces change. Models change. Features appear and disappear. What transfers is the ability to understand the limits of the system, evaluate its output, manage the information it receives, and decide how much trust or responsibility to place in it.

There is also a reason fluency is what gets taught: it is what can be measured. Usage, engagement, and subscriptions are easy to count; understanding is not, and no one who sells a tool is rewarded for teaching people when not to use it. That is not a conspiracy. It is where the incentives point, and it explains why so much of the available material stops at the interface.

This leaves us with two related but different questions. How well do you understand AI and how to judge its use? And what are you actually capable of doing with it? The first is a question of literacy; the second is a question of capability. The distinction matters because the two do not necessarily develop together. To make that distinction useful rather than abstract, we need a way to describe how those two dimensions develop—and how someone can advance in one without advancing equally in the other.

By AI literacy, I mean something specific, and I hold to this definition throughout: the ability to recognize and understand what AI is and where its limits are, to evaluate its outputs critically for accuracy and bias, and to use it safely and ethically while keeping a

human in a position of judgment and oversight. Fluency lets you operate a tool; literacy lets you decide whether, when, and which tool to use, and when to refuse one.

3. Two Axes: Literacy and Capability

The distinction between literacy and capability becomes clearer if we treat them as two related dimensions rather than as a single path.

The first describes a person's developing relationship with AI: Access → Fluency → Understanding → Judgment. **Access** means you have the tool. **Fluency** means you can operate it effectively. **Understanding** means you know enough about how it works to evaluate what it gives you. **Judgment** means you can decide when, where, how, and whether to use it at all.

These are not four equal stages of literacy. Access is the starting condition; having a tool makes everything else possible but says nothing about whether you can use or understand it. Fluency adds operational competence. Understanding and Judgment move beyond operation toward literacy: knowing enough about the system to evaluate its output and make informed decisions about its use.

This is where much of the confusion around AI begins. We have mistaken access for competence, and competence for literacy. Having a chatbot does not mean you know how to use it. Knowing how to use it does not mean you understand it. And understanding AI is not mainly a matter of learning more commands; it is a matter of judgment.

The second dimension is Capability: what you can actually do with the tools. It develops through four broad stages, named for the work you do at each. **Using the tools** means working with AI on everyday tasks. **Mastering the tools** means managing more complex, multi-step work across tools while keeping human judgment in control. **Building with the tools** means creating reusable skills (reusable actions a system can call), agents (programs that carry out multi-step work with some independence), and workflows that can carry out parts of that work with increasing independence (several of those working together toward one goal). **Architecting with the tools** means designing the larger systems that let a model serve real users reliably.

The two dimensions are related, but they do not necessarily develop at the same rate. Someone can become highly capable at using or even building with AI without developing equivalent understanding or judgment. Conversely, someone may develop strong AI literacy without ever needing to build an agent or architect an AI system. The capability a person needs depends on the work they need to do. The literacy they need depends on the decisions they are responsible for making.

Fluency and **using the tools** are where the two dimensions overlap most directly: knowing how to operate the tool and being able to operate it are essentially the same skill. Beyond that point, the relationship becomes more important rather than less. Greater capability creates more consequential choices, which makes understanding and judgment increasingly important. But greater capability does not automatically produce either one.

This paper's framework¹ is author-developed. It is informed by, but does not restate, the **AI Fluency Framework** developed by Rick Dakan and Joseph Feller. Their 4D framework describes four connected competencies for working effectively, ethically, and safely with AI: **Delegation**, deciding whether, when, and how AI should be involved; **Description**, communicating goals, context, and constraints clearly enough to guide the system; **Discernment**, evaluating the usefulness, accuracy, and fit of what the system produces; and **Diligence**, taking responsibility for the consequences of AI-assisted work. [AI Fluency framework, *Dakan and Feller*].

The two frameworks answer different questions. The 4D framework describes the competencies involved in human–AI collaboration. This paper separates two dimensions that are often taught and measured together: **AI literacy**, meaning the depth of understanding and judgment a person brings to AI use, and **AI capability**, meaning what that person can practically do with AI systems. It therefore uses fluency in a deliberately narrower sense: the operational ability to provide useful instructions and obtain useful results. On that narrower definition, Description corresponds most directly to this paper's Fluency stage. The other 4D competencies extend beyond operation alone: Delegation requires deciding whether to use AI and what authority it should have; Discernment requires evaluating output for accuracy and fitness for purpose; and Diligence requires accountability, transparency, and responsible use. In this paper's model, those competencies primarily map to the later literacy stages of Understanding and Judgment.

This is **not a disagreement** with the 4D framework. It is a choice of analytical emphasis. The 4D framework treats mature AI fluency as a unified set of practical competencies. This paper unbundles that maturity into two axes so that a learner, educator, or organization can distinguish between becoming more capable with AI and becoming better able to evaluate, govern, and refuse its use. A person may advance in one dimension without advancing equally in the other. That separation is the paper's original contribution.

Literacy therefore does not precede **Capability** in a simple sequence; it *guides* it. Capability describes what you can do with AI. **Literacy** helps determine which

¹ **Framework Note.** The Access → Fluency → Understanding → Judgment literacy progression and the Using → Mastering → Building → Architecting capability progression are original analytical frameworks developed for this paper. They are not validated psychometric scales, maturity models, or taxonomies adopted from Dakan and Feller, Anthropic, or another external source.

capabilities are worth developing, when to use them, what limits to place on them, and how to recognize when something has gone wrong. The two can reinforce each other: practice develops **Fluency**, failures can deepen understanding, and greater capabilities create new situations in which judgment must be exercised. But judgment does not arise automatically from greater use or greater technical skill.

That distinction is central to everything that follows. The next four sections examine the progression from access through fluency and understanding to judgment. The four capability sections that follow them—using, mastering, building, and architecting—then show how increasing technical capability draws on those forms of literacy without guaranteeing them.

4. Access

Access is the starting condition for everything that follows. You cannot develop fluency, understanding, or judgment about a tool you cannot use. But access itself should not be confused with literacy.

Having an AI tool available tells you very little about what you actually have. Informed access means knowing which system you are using, which model is responding, what limits apply to your account or tier, what the tool can reach beyond the conversation (your files, the web, or nothing), and what happens to the information you give it. None of this means you can use the tool well. It simply means you know what you are starting with.

For a student, Access may look like a free chatbot on a phone, used much like a search engine, with little awareness of which model is answering or what happens to the information entered into it. For a small business, it may look like several employees using personal AI accounts with no shared understanding of which tools are approved, what company information is being pasted into them, or where that information goes.

Access makes AI use possible; adoption tells us that people have begun using it. Neither tells us how well they understand what they are using.

Why it matters. Adoption has moved faster than understanding. In early 2026, about half of U.S. adults reported using AI chatbots, up from roughly a third two years earlier, and 44% reported having used ChatGPT; looking things up was among the most common uses [*Pew Research Center, Americans and AI 2026*]. Those figures show how quickly access has spread. They do not tell us whether users understand the systems they are using, which is precisely the distinction this paper is trying to preserve.

How to develop it. If you are already using AI, begin by finding out what you are actually using. Identify where AI is available to you: at work, inside the software you already use, or through tools you access directly. Then write down five things: the tool

you use, the model behind it, the account or tier you are on, what it can and cannot access, and what happens to the data you provide. It is a simple step toward AI literacy because it replaces vague familiarity with an informed starting point.

5. Fluency

Fluency is the ability to operate the tool effectively: knowing how to give it enough context and instruction to produce a useful result. It is the first step beyond access and the point where literacy and capability overlap most directly. On the capability axis, it corresponds to *Using the Tools*.

Prompting, meta-prompting, and choosing a more capable model can all improve Fluency. They do not, by themselves, demonstrate **Understanding** or **Judgment**. Fluency reaches its limit when you know how to get useful results from the tool but do not yet understand why it behaves as it does, where its limits come from, or how those limits should shape your decisions.

For a student, fluency looks like prompts that improve with practice and the discovery that phrasing and context change the answer. For a small business, it may look like the person who has become the office's AI expert: fast with the interface and good at getting useful results, but less certain about why the tool works when it works, or why it fails when it fails.

Why it matters. Fluency delivers real gains, which helps explain why so much AI training concentrates here. In a preregistered experiment with 453 college-educated professionals, access to ChatGPT cut the time spent on writing tasks by 40% and raised graded quality by 18%, with weaker writers benefitting particularly strongly [Noy & Zhang, *Science*, 2023]. Among 5,172 customer-support agents, using an AI assistant, researchers found an average 15% increase in issues resolved per hour [Brynjolfsson, Li & Raymond, *QJE* 2025].

Those findings matter because Fluency is not trivial. Learning to use AI effectively can make people faster and, on some tasks, improve the quality of their work. But those results come from tasks on which the tools performed well. They do not tell us whether a fluent user can recognize a task on which AI performs poorly, identify a plausible but false answer, or decide when the tool should not be used. Those questions require something beyond operational skill.

How to develop it. Fluency develops through practice. Use AI repeatedly on real tasks, pay attention to how changes in context and instruction affect the result, and keep track of what works. Repeated, intentional use builds fluency. But practice alone does not explain why the system behaves as it does or where its limitations come from. For that, fluency has to give way to understanding.

6. Understanding

Understanding is knowing why an AI system behaves as it does, so you can recognize where its strengths come from and where its limits begin. It does not require computer-science expertise or knowledge of every implementation detail. The required depth should match the decisions being made and their consequences. Someone deciding how to use AI in their work needs a different level of technical understanding than an AI system, but both need a mental model adequate to the decisions they are responsible for making.

At a minimum, that means understanding enough about how the system produces an output to explain why the same tool may perform well in one situation and fail in another. It means knowing what influences its responses, what information it has and does not have, why it can produce confident mistakes, and which conditions should cause confidence in its output to decrease. Risk is part of that understanding: not simply knowing that a system can fail, but having some idea of how it can fail and what the consequences may be.

A useful starting point is how a language model produces an answer. Most modern language models use an architecture called a **transformer** to predict the next **token**—a word, part of a word, or other small unit of text—based on patterns learned during training and the context provided in the current interaction. **Training** allows the model to learn statistical patterns from large amounts of data; it does not turn the model into a database of verified facts. The distinction matters because a model can produce language that is coherent and convincing without establishing that what it says is true.

Understanding also requires distinguishing the model from the larger system around it. The model itself is only one component of the AI product a user encounters. A system may retrieve information from the web or private documents, give the model access to external tools, retain information from earlier interactions, or impose controls on what the system is allowed to do. Some capabilities that appear to belong to the model therefore come from components surrounding it.

That distinction becomes increasingly important as systems become more capable. An answer based on private documents may depend on **retrieval**, which supplies outside information to the model. Taking an action may depend on external **tools** and orchestration software that coordinates how they are used. Restrictions on behavior may depend partly on guardrails built around the model. The point is not that every AI-literate person must master these mechanisms. It is that a person should understand enough to ask the right questions: *Where did this information come from? What part of the system produced this behavior? What does the system actually know or have access to? Where could it fail?*

Why it matters. Fluency helps to produce a useful result; understanding helps with interpreting that result. It gives a basis for asking why the result occurred, what shaped it, where its limitations come from, and how much confidence it deserves. For a student, that may mean understanding why an AI-generated answer should not automatically be treated as a fact. For a professional, it may mean recognizing that an apparently authoritative response depends on information or components whose reliability must be evaluated separately.

Research helps explain why that distinction matters. Confident mistakes cannot simply be treated as an ordinary software bug that will necessarily disappear with the next model update. Researchers at OpenAI and Georgia Tech argue that current training and evaluation methods can reward models for guessing rather than expressing uncertainty, helping explain why plausible falsehoods can be stated fluently [Kalai, Nachum, Vempala & Zhang, *arXiv:2509.04664*, 2025]. The “jagged frontier” discussed in Section 9 shows another difficulty: tasks that appear similarly demanding can fall on opposite sides of a model’s competence. Automation bias adds a human problem: people can place too much trust in automated output, particularly when that output appears confident and polished.

Understanding does not eliminate those risks. It makes them visible enough to manage. A user who understands why fluent output can still be wrong is better equipped to verify it, recognize uncertainty, choose appropriate safeguards, and decide when human review is necessary. That is where understanding begins to support judgment.

How to develop it. Start with the level of understanding your decisions require. For many users, a conceptual explanation of how language models generate responses, why they can produce plausible errors, and how the model differs from the surrounding system is enough to establish a useful foundation. Visual explainers of transformers and token prediction can make those mechanisms concrete without requiring mathematics. [Transformer Explainer; visual transformer explainers].

People whose work carries greater technical responsibility can go deeper. Introductory courses such as DeepLearning.AI’s *Generative AI for Everyone* or *Generative AI with Large Language Models* cover how models are trained and deployed and where they fail without requiring extensive mathematics [DeepLearning.AI generative AI courses]. For people building or debugging AI systems, Fast.AI’s *Practical Deep Learning for Coders* or a university-level study of transformers and deep learning provides implementation-level understanding [Fast.AI; university deep learning syllabi]. The goal is not to know everything about AI. It is to know enough about the system in front of you to understand what it can reasonably support, recognize where your knowledge ends, and make the decisions for which you are responsible.

7. Judgment

The final stage of literacy is **judgment: deciding whether, when, and how AI should be used, how much confidence its output deserves, and where human responsibility must remain.** Understanding explains what a system is doing, what shapes its behavior, and where its limitations come from. Judgment applies that understanding to a decision.

Judgment begins with accepting accountability for how AI is used and for decisions made based on its output. An AI system can generate, recommend, predict, and increasingly act, but it cannot assume responsibility for the consequences of its use. That responsibility remains human.

This is the principle I refer to as **Digital Dignity**²: people should retain meaningful authority over, and accountability for, decisions that affect people, even when an AI system generates, recommends, or carries out the work. The term has been used more broadly in discussions of dignity, agency, privacy, representation, and accountability in digital environments; I use it here in this narrower operational sense. The principle aligns with established human-rights and AI-governance principles concerning human dignity, meaningful human oversight, and ultimate human responsibility. It is not presented as a settled legal doctrine or empirically validated construct. [Natili, *The Human Blueprint*, *human-blueprint.ai*; UNESCO *Recommendation on the Ethics of AI*].

Judgment means deciding when a system is **fit for purpose**, that is, suitable for the task and the consequences of getting it wrong; when its output requires verification; when bias, uncertainty, or missing information should reduce confidence; when human oversight is necessary; and, crucially, when AI is the wrong tool for the task.

Understanding a system is not enough if that knowledge does not shape how it is used. The same capability may be acceptable in one context and unacceptable in another because the people affected, the consequences of failure, or the information involved are different. Ethics provides principles for making those choices. Laws, organizational policies, professional standards, and security requirements may constrain the decision, but they do not replace judgment. A system can satisfy a formal requirement and still raise questions about fairness, privacy, autonomy, or human dignity.

Why it matters. Greater AI capability increases both what can be accomplished and what can be delegated. That makes the quality of decisions around its use more important, not less. A fluent user may know how to obtain a convincing answer. A user with greater understanding may know why that answer could be wrong. Judgment determines whether to trust, verify, modify, reject, or act on it.

² **Terminology note.** “Digital dignity” is used in wider scholarship, policy, and public discourse to describe the protection of human dignity in digital environments, including autonomy, privacy, fair representation, accountability, explanation, contestability, and remedy. This paper uses the term more narrowly to focus on meaningful human authority and retained accountability in AI-assisted decisions.

For a student, this may mean deciding whether AI supports learning or replaces it. For a professional, it may mean deciding whether an AI recommendation can inform a decision or requires independent evidence. For someone designing an AI system, it may mean deciding what the system may do autonomously and where a human must remain involved.

Judgment is therefore not a checklist of ethical principles or regulatory requirements. It is the ability to make and defend decisions about AI use based on an understanding of the system, the context in which it operates, the possible consequences, and the values at stake.

Two people may understand the same technical limitation and still reach different decisions because the consequences and ethical obligations differ. The key test of Judgment is whether you can explain the decision: why you chose a particular boundary, what evidence supports it, and which values or consequences shaped it.

How to develop it. Judgment develops through decisions, not through tool use alone. Take real or realistic situations and force an explicit choice: should AI be used here at all? What should it be allowed to do? What must be verified? What happens if it is wrong? Who could be affected? Who remains accountable? Then state the reasoning behind the decision and revisit it when the facts change. Ethics, regulation, security, and policy become resources to consult in that process rather than separate subjects to memorize. The goal is not to eliminate uncertainty. It is to make AI decisions deliberately, explain why they were made, and remain accountable for their consequences.

8. From Knowing to Doing

The progression from access through fluency, understanding, and judgment describes a developing relationship with AI: whether you have access to a tool, whether you can operate it effectively, whether you understand enough to evaluate its behavior, and whether you can make informed decisions about its use. Literacy becomes consequential when that knowledge informs what you do.

The four capabilities that follow shift the focus from what you understand to what you can do with AI. They also show why the two dimensions cannot be separated completely. As capability increases, decisions about reliability, risk, delegation, and responsibility become more consequential, drawing increasingly on understanding and judgment.

The capabilities are ordered by what they ask of you, not by rank. Using comes first because the other capabilities build on it. Mastering, building, and architecting represent increasing levels of technical capability, but they are not destinations everyone needs to

reach. How far you go depends on what you need to accomplish and the responsibilities you carry.

The sections that follow show where capability and literacy reinforce each other—and where greater capability can expose the absence of understanding or judgment.

9. Using the Tools

Using the tools is the first capability, and the only one with a fixed place in the order: you cannot master, build with, or architect around something you have never used. In practice, it means working with a consumer AI system such as ChatGPT, Claude, or Gemini on everyday tasks: drafting an email, summarizing a document, brainstorming options, or organizing information.

It also extends to the small acts of thinking that fill a day: turning a tangle of notes into a to-do list, reorganizing scattered thoughts into an outline, pressure-testing an idea before committing to it. At this stage, AI is less an oracle than a fast, tireless surface for externalizing and reorganizing your own thinking.

Why it matters. For a working professional, *using the tools* is where the largest and easiest gains may be found, particularly on work the person already understands. The evidence is more modest than the marketing claims but still substantial. A large field study of customer-support agents found that access to a generative AI assistant increased issues resolved per hour by about 15% on average, with the largest gains among less-experienced workers [Brynjolfsson, Li & Raymond, QJE 2025]. A Harvard–BCG field experiment found that consultants using GPT-4 on suitable tasks completed them 25.1% faster, completed 12.2% more tasks, and produced work rated more than 40% higher quality [Dell’Acqua et al., Organization Science, March 2026]. But the same study revealed the limitation. AI was strong on some tasks and weak on others that appeared comparably difficult, a pattern the researchers called a “jagged technological frontier.” On a complex task outside GPT-4’s capability frontier, consultants using AI were 19 percentage points less likely to reach the correct answer than those working without it. AI did not simply help less; in that case, using it made performance worse.

This creates a calibration problem. Users need to determine not only whether AI can help with a task but also whether it actually did. Across three preregistered studies involving 2,691 participants, researchers found that people overestimate the time and effort AI saves on simple tasks and continued using it even when it saved no time, an effect they call the “**efficiency-gain illusion**” [Yu et al., arXiv:2605.22687, 2026]. Both findings point the same way: feeling helped is not evidence of being helped, and the skill at this stage is to check one against the other. [Horowitz & Kahn, International Studies Quarterly, 2024].

For students, the same tension appears in a different form. AI can support learning, but it can also replace the thinking that learning is meant to develop. A randomized trial of 334 university students found that access to an AI tutor improved test performance without reducing reading effort [*Fischer, Rau & Rilke, IZA DP 18338, 2025*]. Other studies suggest that how the tool is used matters: structured, scaffolded AI tutors can improve learning, while less constrained use may encourage overreliance and weaken performance when AI support is removed [*Kestin et al., Scientific Reports, 2025*; *Bastani et al., PNAS, 2025*].

The distinction is important. For a professional, AI may help perform work the person already knows how to evaluate. For a student, some of the work being delegated may be the very reasoning the student is supposed to develop. Assignments exist to build—an effect strongest for the least-prepared learners [*Springer, Italian Economic Journal, 2025*; *MDPI, Societies, 2025*]. The literacy question is therefore not simply whether AI makes the task easier, but what the person is gaining, and what they may be giving up, by using it.

How to learn it. Pick one real, ongoing task you are already responsible for—a report due next week, a backlog of customer emails, a course module you are working through—and use AI with it repeatedly for two weeks. Work with the actual task and its actual mess rather than a tutorial exercise. Pay attention to how the model responds to changes in context and instruction, where it drifts, and what it invents. Keep a short log of what worked and what did not. Record the time actually spent, not the time you think you saved, and compare the quality of the result with what you would have produced without AI. The purpose is not merely to become faster at prompting. It is to begin calibrating your impression of the tool against its actual performance.

This capability depends most directly on fluency, but the evidence above shows why fluency alone is insufficient. The more consequential the task, the more using AI draws on understanding and judgment: understanding where the system may fail, and judgment about whether to trust, verify, modify, or reject its output.

10. Mastering the Tools

Mastering the tools means moving beyond a single interaction and learning to manage AI across a complete workflow: managing context and continuity, recovering when a tool fails, checking intermediate results, and coordinating work across chat, code execution, file handling, voice, or other tools.

At this stage, orchestration is still primarily human. The user decides which tool should handle each part of the work, moves information between them, evaluates the results, and determines what happens next. This is not yet automation described in *Building with the Tools*. It is a multi-step process held together by the user's own judgment.

Why it matters. As a workflow becomes more complex, the burden on the person managing it changes. As you manage multiple agents or multi-step processes, you encounter what some call the “Orchestration Tax”—the reality that while starting agents is easy, your own cognitive bandwidth does not scale in parallel. Research on working memory and multitasking suggests why this becomes difficult: attention is limited, and switching repeatedly among competing tasks can degrade memory and performance [*Wagner, Heavy media multitasking and reduced memory; Allen et al., sustained attention and vigilance decrement in multitasking*].

The hardest part of the work can therefore shift from producing each individual component to integrating the whole. At this stage, the user is the connective tissue. The more work that is distributed across tools, the more important it becomes to preserve context, check what each tool has produced, and prevent small errors from propagating through the workflow.

This is where a practitioner begins designing around failure rather than merely being surprised by it. For an individual, it is the step from using AI for a draft to managing the process that produces a deliverable you would put your name on. For a small business, it is the step from an experiment to a workflow someone can depend on. For a student, it is the point at which AI can become part of a study process rather than a single-shot answer machine. In each case, greater capability brings greater responsibility to review what the tools produce and decide what can safely be delegated. Crucially, this shift elevates the operator’s role from passive to ethical arbiter. A responsible operator is the primary filter for accuracy, bias, and fairness, performing a constant, implicit moral audit of the machine’s judgment.

Mastery also includes learning not to use more capability than a task requires. A simple, repetitive task such as reformatting, classifying, or extracting information may not require the most capable model available. Testing the same task with a lighter model can reveal whether the additional capability improves the result. Matching the model to the task can reduce cost and latency and, depending on the systems involved, resource use. The larger principle is right-sizing: use the capability the task requires rather than assuming that the most powerful available model is always the appropriate choice.

How to learn it. Take one workflow you run every week and try to carry it end-to-end with AI, from raw input to finished output. Watch precisely where it breaks. Where is context dropped? Where does an error enter the process? Which results require verification? Where does moving between tools create additional work rather than remove it? Those failures become the curriculum at this stage.

Keep the human workflow explicit as you refine it: what each tool does, what information passes between them, what you check, and where you make the decision to continue,

revise, or stop. The goal is not simply to use more tools. It is to manage a more complex process without losing control of the result.

Mastering draws more heavily on understanding and judgment than direct use does, because the user is now responsible not only for individual outputs but for how they fit together. Understanding helps explain where failures come from; judgment determines what to delegate, what to check, and whether to use the workflow at all. Better orchestration skills can support those decisions, but they can't make them for you.

11. Building with the Tools

Building with the tools begins when some of the coordination that was performed manually at the mastering stage is transferred into the system itself. Instead of deciding every time which tool to use, what information to pass to it, and what should happen next, you begin encoding those decisions into a repeatable workflow.

The terminology around this work changes quickly, but a few distinctions are useful. A **skill** is a bounded action a system can perform when asked. An **agent** can pursue a goal across multiple steps, choosing among available actions with some degree of autonomy. More complex *configurations* may combine several skills and agents into a larger workflow. The labels matter less than two questions: How much can the system decide on its own, and what information or state does it retain as it works?

Those questions matter because autonomy changes the nature of failure. If you ask a chatbot for a summary and it gets something wrong, you can catch the error before doing anything with it. If an agent reads incoming messages, classifies them, updates a project record, and drafts a response before you intervene, one mistaken decision can propagate through several steps. Building therefore means designing not only what the system should do when things go right, but what should happen when they do not.

Consider a simple inbox workflow. At the mastering stage, you might ask AI to summarize messages, compare the summary with project documents, and draft a status update, moving the information between tools yourself. At the building stage, you could create a workflow that does those steps automatically: collect relevant messages, retrieve the appropriate project information, prepare a draft, and pause for approval before sending anything.

The important part is not that the workflow uses an “agent.” It is the decisions encoded around it: what triggers the process, what information it may access, what it is allowed to do without permission, what must be verified, what happens when information conflicts, and where a human must intervene. In mastering, the human is the connective tissue. In building, some of that connective tissue becomes part of the system.

Why it matters. Automation can multiply AI's value because a useful process no longer has to be reconstructed by hand each time. On well-defined workflows, organizations report meaningful reductions in effort and improvements in productivity [*Bain, Technology Report 2025; Forrester, Total Economic Impact of Agentic AI Solutions 2026; McKinsey, The State of AI 2026*]. But those gains are task specific. They should not be treated as evidence that more automation automatically produces more productivity.

On standardized, well-scoped work, agentic systems can cut cost-per-task by large multiples — one 2026 benchmark synthesis reports reductions on the order of 9–66x and time savings near 6.4 hours per knowledge worker per week [*Bain, Technology Report 2025*], “[*McKinsey, Generative AI and Productivity 2025*]”, “[*Forrester, Agentic Automation Total Economic Impact 2026*]. Those are gains on repeatable tasks, not a claim that any individual becomes nine times more productive.

Evidence from software development makes the point particularly clearly. In a randomized study of experienced open-source developers working in familiar repositories, METR found that AI tools increased completion time by about 19%, even though participants expected to be faster and afterward believed they had been faster [*METR, arXiv:2507.09089*]. METR later reported that newer estimates were affected by selection bias and redesigned the experiment, so the result should not be generalized beyond the setting studied [*METR, We Are Changing Our Developer Productivity Experiment Design, 2026*].

In a randomized controlled trial, METR found that experienced open-source developers using early-2025 AI tools took about 19% longer on real tasks in their own repositories, even though they predicted a speedup and believed afterward they had been faster [*METR, arXiv:2507.09089*]. The broader lesson is the same one that appeared at the using stage: capability does not automatically translate into value. At the building stage, however, the consequences are larger because mistakes can now be repeated automatically. A workflow that saves five minutes once is convenient. A workflow that makes the same wrong decision hundreds of times is a system failure.

Building also makes right-sizing more consequential. A routine step such as classification, extraction, or reformatting may not require the most capable model available. A more demanding step may. Testing different models against the actual requirements of each part of the workflow can reduce cost and latency without sacrificing quality. The principle remains the same: match capability to the task rather than assuming that more capability is always better.

Most importantly, building turns questions of judgment into design decisions. What may the system do autonomously? What information may it access? What requires verification? When should it stop rather than guess? When must a person approve the

next action? Who can challenge or reverse what it has done? These decisions determine how human authority and accountability are preserved as more work is delegated to the system.

Through 2026, building with agents is shifting from a specialist skill to an increasingly valuable capability in technical roles and a potential competitive edge for small businesses, particularly as organizations pay greater attention to cost governance and sustainability reporting [Gartner, *Agentic AI Maturity Model 2026*]. For students entering AI-affected fields, the ability to build a small agent to accomplish a defined task is moving from the research lab to the practical skills employers may increasingly look for [LinkedIn, *2026 Emerging Skills Report*].

How to learn it. Start with one repetitive workflow whose inputs, outputs, and success criteria you already understand. Map the steps before automating them. Then automate the smallest useful portion and observe what happens. Add autonomy only when the existing process is reliable enough to justify it. The method is more important than the particular platform: start small, measure what happens, make failures visible, and add capability only in response to a demonstrated need.

For conceptual grounding and hands-on practice, several accessible resources provide useful starting points. Fast.AI's practical courses introduce the foundations behind modern AI systems, while Hugging Face's free Agents Course moves into building working agents with frameworks such as *smolagents*, *LlamaIndex*, and *LangGraph*. DeepLearning.AI's short courses provide more concrete on-ramps: "AI Agents in LangGraph," for example, walks through building an agent helps visualize which work the model performs and which work is handled by the surrounding system. Others courses cover agentic RAG and agent evaluation. Documentation from providers such as Anthropic and OpenAI goes further into tool use, function calling, orchestration, and standards such as MCP for connecting models to external tools [Fast.AI, *Practical Deep Learning*], [DeepLearning.AI, *AI Agents in LangGraph*].

These resources teach the mechanics of building, but mechanics are only part of the capability. As a system becomes more autonomous, understanding and judgment become increasingly important. Understanding helps identify which part of the system produced a failure and why. Judgment determines what the system may do on its own, what it must ask before doing, what requires verification, and where a human must remain involved. A configuration can act; only a person can answer for the action.

12. Architecting with the Tools

Architecting with the tools means designing the system around the model rather than working primarily with the model itself. It is genuinely engineering-grade work. Building

creates AI-enabled workflows; architecting creates the larger system that makes those workflows reliable, observable, governable, and maintainable in production.

The components are often discussed separately—retrieval-augmented generation (RAG), advanced prompting, recursive or multi-step calls, LLM-as-a-judge, LLM-as-a-verifier, and guardrails—but architecting means understanding how they function together as a **single architecture**.

One useful way to see that architecture is by what its components are responsible for. Context determines what information the model receives: RAG can supply current, private, or task-specific knowledge, while techniques such as reranking, hybrid and graph-based retrieval, and agentic retrieval help determine which information reaches the model. These approaches belong to the broader practice of **context engineering**: deliberately managing the information available to a model at inference time [Atlan; *Towards Data Science*, 2026].

Direction and action determine what the system does with that context. Structured prompting, few-shot examples, recursive or multi-step calls, **ReAct-style** patterns, tool use, and function calling can break complex work into steps and allow the system to act beyond generating text. Protocols such as MCP and A2A can help connect models, tools, and other agents.

Evaluation determines whether the result is good enough to continue. Evaluation harnesses, task-specific tests, model-based evaluators such as LLM-as-judge, and verification steps can compare outputs against defined criteria. **Controls** determine what the system is permitted to do, when it must stop, and when human approval is required. **Observability** records what happened: the context supplied, sources retrieved, tools called, decisions made by the surrounding software, evaluations performed, and outputs produced.

Architecting is the work of making those pieces operate as a coherent system. The bare model is only one component within it.

Why it matters. This is the difference between a prototype that demos well and a system people can depend on in production. A demonstration needs to work long enough to show what is possible. A production system has to keep working when inputs vary, components fail, information conflicts, requirements change, and people depend on the result.

That is also where **auditability and provenance** become architectural requirements. An auditable system should preserve enough evidence to reconstruct what happened: what information was available, which sources were retrieved, which tools were called, what actions were taken, what checks were performed, and where human approval entered the process. Provenance records where information came from and how it

moved through the system. Together, these mechanisms make a system more traceable and its behavior more open to inspection.

They should not be confused with perfect **explainability**: RAG does not automatically create provenance; the architecture must preserve and expose the sources it retrieves. An LLM used as an evaluator does not automatically make an evaluation reliable; the evaluator itself must be tested. Guardrails can enforce defined boundaries, but those boundaries still have to be chosen, maintained, and monitored. Architecture does not remove the need for judgment. It turns many judgments into explicit design decisions.

This is where ethics becomes part of engineering rather than commentary around it. Decisions about what the system may access, what it may do autonomously, what evidence it must preserve, when it must defer to a person, and how an affected person can challenge an outcome can be designed into the architecture. Regulation may specify some of those requirements, but the engineering need exists even when regulation does not. A system people depend on should be designed so that its behavior can be inspected, failures can be investigated, and responsibility does not disappear inside the machinery.

For a student or professional entering an AI-affected technical field, architecting is an advanced path where formal systems knowledge earns its keep. Accessible models, orchestration frameworks, and evaluation tools have lowered the barrier to experimentation, but they have not eliminated the engineering required to turn those components into something robust. Access to the ingredients is not the same as mastery of the recipe. Production systems still require the ability to reason about reliability, security, data, evaluation, failure, observability, and the interactions among components.

How to learn it. Be honest about the investment. Architecting production AI systems is not a capability acquired through a few tutorials. It requires sustained study and practice in the systems surrounding the model: retrieval and data architecture, evaluation, orchestration, security, observability, reliability, and the design of human oversight.

Not everyone needs to reach this level, and most people will not. The earlier capabilities carry most of the everyday value. But anyone designing AI systems that real people depend on takes on responsibility for this level of rigor. If that expertise does not exist internally, the responsible choice is to bring in someone who has it.

Architecting draws on every part of AI literacy because technical decisions at this level become decisions about what the system will do repeatedly and at scale. It is judgment written into the system: decisions about evidence, oversight, boundaries, and accountability made once so that they hold every time. It presupposes every stage of

literacy. But architecture cannot supply the reason to build the system in the first place; that decision comes before the architecture, and it belongs to judgment.

13. Where to start

The question the students asked at Tacoma Community College was straightforward: *What should I learn to remain competitive, and where do I start?* It is an individual question, but the problem behind it appears at organizational scale as well.

A student may have a powerful AI assistant available and use it mainly as a better search engine, without understanding enough about the system to judge its answers or recognize what else it can do. A small-business owner or manager can make the inverse mistake: acquiring more AI capability than the organization needs without knowing how to evaluate whether the additional capability improves the work. The student underuses a capability they do not yet understand; the business may overbuy one it cannot yet evaluate. They are making the same error from opposite ends.

The common problem is not lack of access. Both have access, and both may develop considerable fluency. What is missing is the literacy needed to connect capability to purpose: understanding what the system can and cannot do, evaluating whether it is producing value, and exercising judgment about when and how it should be used.

That is why the answer to *What should I learn?* is different from *How technically capable should I become?* Everyone who works seriously with AI benefits from greater understanding and judgement. Not everyone needs to build agents or architect AI systems. The appropriate level of capability depends on the work; the necessary depth of literacy depends on the decisions and consequences involved.

For someone starting out, the practical answer is to begin with the tools available, learn to use them effectively, and develop enough understanding to recognize their strengths and limitations. From there, develop the capabilities the work requires—whether that means mastering more complex workflows, building with AI, or, for those whose work demands it, architecting the systems around it. At every level, understanding and judgment should guide the capability rather than follow it.

The particular tools worth learning will continue to change. The ability to evaluate them, decide which capabilities are appropriate, and take responsibility for their use is what makes that learning durable.

14. Conclusion

This paper began with a question from a room of students who wanted to know what to learn, and it ends with an answer that is shorter than the paper. We have mistaken access for competence, and competence for literacy. Having a tool is not the same as

knowing how to use it. Knowing how to use it is not understanding it. And understanding it is not, in the end, a matter of commands; it is a matter of judgment.

Two dimensions should be kept distinct. The progression from Access through Fluency and Understanding to Judgment describes a person's developing relationship with AI. Capability runs from Using through Mastering and Building to Architecting and describes what a person can do with it. Fluency and using are where the two dimensions overlap most directly, but literacy and capability continue to interact beyond that point. Greater capability creates more consequential choices, making understanding and judgment increasingly important. Literacy does not simply come before capability; it helps guide which capabilities are worth developing and how to use them.

The evidence reads the same direction. The gains are real but bounded, and in some settings they are particularly large for less experienced users. The failures are often failures of calibration: people cannot always tell where the tool helps, how much it helps, or when it has led them astray. That is why Judgment, understood as the ability to make decisions you can explain and defend, is the most durable outcome of AI literacy. The tools will change; the need to evaluate them will not.

That leaves three responsibilities. Educators should teach beyond the products and put Understanding and Judgment on the syllabus beside prompting, because a course that stops at fluency concentrates on the part most likely to change. Small businesses should define the problem before choosing the model and match the capability to the task, because more capability is not necessarily more value. And every reader should develop the literacy their decisions require and the capability their work requires, rather than assuming progress along one dimension automatically produces progress along the other.

The people who asked the question deserve that much: not a better trick for this season's tools, but a way of standing in relation to the tools that will still hold when the tools are different.

15. Resources

1. AI Literacy, Adoption, and AI Fluency

1. Americans and AI 2026: Chatbots, Smart Devices and Views on Impact — Pew Research Center — <https://www.pewresearch.org/internet/2026/06/17/americans-and-ai-2026-chatbots-smart-devices-and-views-on-impact/>
2. Americans and AI 2026: Chatbots, Smart Devices and Views on Impact — Pew Research Center (PDF report) —

https://www.pewresearch.org/wp-content/uploads/sites/20/2026/06/PI_2026.06.17_Americans-and-AI_REPORT.pdf

3. AI Fluency Framework — Dakan and Feller — <https://aifluencyframework.org/>

4. The AI Fluency Framework — Anthropic (PDF) — <https://www-cdn.anthropic.com/334975cdec18f744b4fa511dc8518bd8d119d29d.pdf>

2. Generative AI Productivity, Calibration, and Automation Bias

5. Experimental evidence on the productivity effects of generative artificial intelligence — Noy and Zhang, *Science* (2023) — <https://doi.org/10.1126/science.adh2586>

6. Generative AI at Work — Brynjolfsson, Li, and Raymond, *The Quarterly Journal of Economics* (2025) — <https://doi.org/10.1093/qje/qjae044>

7. Generative AI at Work — National Bureau of Economic Research working-paper version — <https://www.nber.org/papers/w31161>

8. Navigating the Jagged Technological Frontier: Field Experimental Evidence of the Effects of Artificial Intelligence on Knowledge Worker Productivity and Quality — Dell’Acqua et al., Harvard Business School Working Paper No. 24-013 — <https://www.hbs.edu/faculty/Pages/item.aspx?num=64700>

9. The efficiency-gain illusion: People underestimate the rate of AI use and overestimate its benefits on simple tasks — Yu et al., arXiv:2605.22687 (preprint, 2026) — <https://arxiv.org/abs/2605.22687>

10. Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity — Becker et al., arXiv:2507.09089 (preprint, 2025) — <https://arxiv.org/abs/2507.09089>

11. Measuring the Impact of Early-2025 AI on Experienced Open-Source Developer Productivity — METR research summary — <https://metr.org/blog/2025-07-10-early-2025-ai-experienced-os-dev-study/>

12. We are Changing our Developer Productivity Experiment Design — METR update on the limitations of subsequent estimates (2026) — <https://metr.org/blog/2026-02-24-uplift-update/>

13. Bending the Automation Bias Curve: A Study of Human and AI-Based Decision Making in National Security Contexts — Horowitz and Kahn, *International Studies Quarterly* (2024) — <https://doi.org/10.1093/isq/sqae020>

14. AI Safety and Automation Bias — Center for Security and Emerging Technology — <https://cset.georgetown.edu/publication/ai-safety-and-automation-bias/>

3. Hallucination, Learning, and Critical Thinking

15. Why Language Models Hallucinate — Kalai, Nachum, Vempala, and Zhang, arXiv:2509.04664 (preprint, 2025) — <https://arxiv.org/abs/2509.04664>
16. AI Tutoring Enhances Student Learning Without Crowding Out Reading Effort — Fischer, Rau, and Rilke, IZA Discussion Paper No. 18338 (2025) — <https://www.iza.org/publications/dp/18338/ai-tutoring-enhances-student-learning-without-crowding-out-reading-effort>
17. AI Tutoring Enhances Student Learning Without Crowding Out Reading Effort — Fischer, Rau, and Rilke (PDF) — <https://www.econstor.eu/bitstream/10419/335986/1/dp18338.pdf>
18. AI Tutoring Outperforms In-Class Active Learning: An RCT Introducing a Novel Research-Based Design in an Authentic Educational Setting — Kestin et al., Scientific Reports (2025) — <https://doi.org/10.1038/s41598-025-97652-6>
19. Generative AI without Guardrails Can Harm Learning — Bastani et al., Proceedings of the National Academy of Sciences (2025) — <https://doi.org/10.1073/pnas.2422633122>
20. AI Tools in Society: Impacts on Cognitive Offloading and the Future of Critical Thinking — Gerlich, Societies (2025) — <https://doi.org/10.3390/soc15010006>

4. Neuroscience, Cognition, and Multitasking

21. Heavy multitaskers have reduced memory — Stanford Report — <https://news.stanford.edu/stories/2018/10/decade-data-reveals-heavy-multitaskers-reduced-memory-psychologist-says>
22. The working memory costs of a central attentional bottleneck in multitasking — Vergauwe et al., Psychological Research — <https://link.springer.com/article/10.1007/s00426-021-01615-1>
23. The Mechanism for Allocating Limited Working Memory Resources — Ma et al., Nature Human Behaviour — <https://pmc.ncbi.nlm.nih.gov/articles/PMC9276901/>
24. Examining the Role of Task Requirements in the Vigilance Decrement in Sustained Attention Tasks — Thomson et al. — <https://pmc.ncbi.nlm.nih.gov/articles/PMC6109784/>
25. An examination of sustained attention during complex multitasking — Grier et al. — <https://pmc.ncbi.nlm.nih.gov/articles/PMC12504179/>

5. Agentic Systems, Orchestration, and Technical Resources

- 26. Architect Your Attention (The Orchestration Tax) — Addy Osmani — <https://addyosmani.com/blog/orchestration-tax/>
- 27. Towards a science of scaling agent systems: When and why agent systems work — Google Research — <https://research.google/blog/towards-a-science-of-scaling-agent-systems-when-and-why-agent-systems-work/>
- 28. n8n Documentation — n8n — <https://docs.n8n.io/>
- 29. Model Context Protocol — official documentation — <https://modelcontextprotocol.io/>
- 30. What is A2A protocol (Agent2Agent)? — IBM — <https://www.ibm.com/think/topics/agent2agent-protocol>
- 31. Anthropic documentation — Anthropic — <https://docs.anthropic.com/>
- 32. OpenAI API documentation — OpenAI — <https://platform.openai.com/docs>

6. Learning Resources and Course Examples

- 33. Practical Deep Learning for Coders — Fast.AI — <https://course.fast.ai/>
- 34. Generative AI for Everyone — DeepLearning.AI — <https://www.deeplearning.ai/courses/generative-ai-for-everyone>
- 35. Generative AI with Large Language Models — DeepLearning.AI — <https://www.deeplearning.ai/courses/generative-ai-with-llms>
- 36. Evaluating and Debugging Generative AI — DeepLearning.AI — <https://www.deeplearning.ai/alpha/short-courses/evaluating-debugging-generative-ai/>
- 37. Transformer Explainer — interactive visualization — Polo Club of Data Science / Georgia Tech — <https://poloclub.github.io/transformer-explainer>
- 38. Learning LLM Transformers with Interactive Visual Explanation and Exploration — Cho et al., CHI 2026 — https://minsuk.com/papers/transformer_explainer-chi26.pdf
- 39. Hugging Face Agents Course — Hugging Face — <https://huggingface.co/learn/agents-course>
- 40. AI Agents in LangGraph — DeepLearning.AI — <https://www.deeplearning.ai/short-courses/ai-agents-in-langgraph/>

7. Author Framework

41. The Human Blueprint — Giorgio Natili — <https://human-blueprint.ai/>