LLMs and AI Agents

USP 410/510 | Spring 2026

Dr. Liming Wang

Portland State University

Part 1: Why This Matters

Today

- What LLMs are good at in practice
- How Karpathy recommends using them (as of 2025)
- What the HDSR article found about data analysis workflows
- Takeaways
- How we will use AI tools in this course

Core Claim

LLMs are not just chatbots

- They are **general-purpose interfaces** to text, code, files, and tools
- They can help with **reading, writing, coding, summarizing, translating, and analysis**
- They are most useful when treated as **collaborators**, not oracles
- In this course, the goal is **better thinking and faster iteration**
- The goal is **not** outsourcing judgment

A Good Mental Model

Think of an LLM as a system that can:

- Predict plausible next text
- Work over large amounts of context
- Use tools such as **search**, **Python**, and **code editors**
- Generate drafts very quickly
- Can still produce confident nonsense (always verify its output)

Implication: LLM output is a **starting point for inspection**, not the endpoint

Work with LLM

- Prompt, prompt engineering
- Context window, the size limit
- Thinking vs non-thinking mode
- Tool use
- Mode of interaction: chatbot vs agent

Part 2: How Karpathy Uses LLMs

Karpathy's Practical Use Cases

From the video, several use patterns stand out:

- **Search and explanation** for unfamiliar topics
- **Reading companion** for papers, books, and technical documents
- **Python interpreter / data analysis** for calculations and code execution
- **Coding in agentic IDEs** with project files in context
- **Image and multimodal input** for extracting and interrogating information
- **Custom workflows** through persistent instructions and reusable prompts

Reading With LLMs

Karpathy's workflow is simple and strong:

- Start with a **summary** of the paper or chapter
- Read the source yourself
- Ask questions when something is unclear
- Use the model to unpack unfamiliar vocabulary or background knowledge
- Go back to the original text to verify understanding

This is especially useful when reading outside your domain

Tool Use Changes the Game

An LLM with tools is different from an LLM alone

- Plain chat can draft answers
- **Search** can pull in current information
- **Python / data analysis** can compute, plot, and transform files
- **IDE agents** can read and edit real code in a project
- **Memory / custom instructions** can reduce repeated setup work

The useful question is not "Which chatbot is best?"

It is "What tool setup fits this task?"

Coding With LLMs

Karpathy's message is clear: for serious coding, context matters

- Web chat is often too disconnected from the actual project
- IDE-based tools can see **files, folders, errors, and terminal output**
- This makes them better for:
  - editing existing code
  - debugging
  - refactoring
  - generating small apps and scripts
- You still need to read the code and test the result

Coding Agents

As of April 2026, Coding agents (OpenAI's Codex, Anthropic's Claude Code, and Google's gemini cli) are very capable of the software development and data science tasks.

- Agency: autonomous decides steps to take: plan, execution, collect user inputs
- Tool use: select which tool to use
- Skill: read in instructions for specific tasks on the fly, e.g. office skills

Prompting Patterns That Actually Help

Better prompts usually include:

- **Task**: what you want
- **Context**: what the model should know
- **Constraints**: format, scope, audience, tools, assumptions
- **Examples**: especially for structure and style
- **Verification request**: ask it to explain assumptions, edge cases, or failure points

Karpathy emphasizes **few-shot prompting** when you want consistent output

Part 3: What the HDSR Article Found

The Article's Setup

Evkaya and de Carvalho evaluate ChatGPT's **Data Analysis** workflow as a quantitative copilot

- Upload a dataset
- Ask for summaries, plots, and models
- Let the system generate and run Python code
- Inspect both the outputs and the interpretations

Their key position: useful, but only with **human critique and oversight**

Where It Worked Well

The article shows clear strengths:

- Easy onboarding from common file types like **CSV** and **XLSX**
- Fast generation of **descriptive statistics**
- Helpful suggestions for **next analytical steps**
- Strong support for **basic visualizations**
- Ability to move from questions to runnable Python without requiring the user to code directly

Where It Failed

The same article also documents important failure modes:

- Mislabeling or misdescribing plots
- Interpreting numbers incorrectly even when the figure looks reasonable
- Choosing weak chart types for the task
- Suggesting analyses that do not really fit the data
- Inconsistent results across repeated prompts

Good-looking output is not the same as valid analysis

Example Failure: Visualization

One reported problem:

- A histogram was described as being on a **log scale**
- But inspection of the underlying figure showed it was actually on the **original scale**

Lesson: always check the **axes, labels, encodings, and units**

Example Failure: Interpretation

Another reported problem:

- The correlation heatmap displayed one value
- The accompanying written interpretation reported a different value
- The visualization looked acceptable
- The explanation was still wrong

Lesson: in quantitative work, verify claims against the actual output

Human-in-the-Loop Workflow

The article strongly supports a workflow like this:

1. Use the model to draft an analysis plan
2. Let it generate code or summaries
3. Inspect the data, figures, and assumptions yourself
4. Re-prompt to fix, narrow, or clarify
5. Treat the result as provisional until checked

What This Means for Data Science

LLMs can lower barriers to entry

- Nonprogrammers can do more than before
- Programmers can work faster
- Iteration becomes cheaper
- Exploration becomes easier

But the hard parts remain hard:

- problem formulation
- data quality judgment
- causal reasoning & research design
- validation
- communication

Part 4: How We Will Use AI in This Course

Course Norms

AI use is encouraged on assignments

- Use LLMs to **brainstorm**, **debug**, **explain**, and **speed up routine work**
- Do not use them to avoid understanding your own project
- You are responsible for every line of code, chart, and claim you submit
- If the model gives you an answer you cannot explain, you do not understand it yet

A Strong Workflow for This Class

For assignments and projects:

1. Start with your own question
2. Ask the model for a plan, not just an answer
3. Work in small steps
4. Run the code and inspect the output
5. Save intermediate results
6. Keep notes on what you changed and why

What To Ask LLMs For

Useful requests:

- "Explain this error message"
- "Write a first draft of this function"
- "Suggest three ways to visualize this variable"
- "What assumptions does this analysis make?"
- "Refactor this code to be clearer"
- "Help me interpret this output, and say what could be misleading"

What Not To Delegate

Be careful when asking the model to decide:

- whether the data is trustworthy
- whether a pattern is causal
- whether a map or chart is ethically appropriate
- whether the results make sense for Portland or Oregon policy contexts
- whether a conclusion is ready for public communication

These are analyst responsibilities

In-Class Exercise

Take 2024 Oregon Crashes Geodatabase and try three prompts:

1. Summarize number of crashes by severity
2. Visualize number of crashes by severity
3. **Verify and Critique** the result

Compare which prompt gives the most reliable and usable output

Key Takeaways

- LLMs are progressing incredibly fast, use the latest models whenever possible
- Create your own eval/benchmark for common tasks
- Evaluate different models from time to time
- Enable thinking + tool use
- In data science, **verification is part of the workflow**
- Your competitive advantage is not having AI access, but knowing **how to direct, check, and integrate** AI output

For Next Week

Read:

- Yu & Barter, Chapter 4: Data Preparation
- McKinney, Chapters 6-8

Do:

- Continue DataCamp DC1 if needed
- Install and test at least one LLM tool you can use for coding or analysis
- Bring one example next week of a good AI interaction and one bad one

Sources for today's slides:

- Karpathy, How I Use LLMs
- Evkaya & de Carvalho, Using ChatGPT for Data Science Analyses