DR.-ING. MARK SCHUTERA

AGENTIC AI AND AGENT SYSTEMS

UNFINISHED LECTURE NOTES

PUBLISHED BY UNFINISHED LECTURE NOTES

First printing, July 2025

THE HOTTEST NEW PROGRAMMING LANGUAGE IS ENGLISH.

ANDREJ KARPATHY

Contents

Introduction

These unfinished lecture notes analyze agent systems founded on Large Language Models, examining their theoretical foundations, architectural design, and operational mechanisms. We investigate how LLMs function as cognitive substrates for autonomous agents and their integration of reasoning, action and observation capabilities through structured interaction interfaces.

THESE NOTES ARE A LIVING THING, subject to continuous updates and improvements—and have only recently seen the light of day. They are not yet complete and are rather intended to showcase the look and feel of my lecture notes, but I hope you will come to find them useful nonetheless.

Foundations of Agentic AI

Language model evolution proceeded through discrete developmental phases culminating in contemporary agentic systems. Pre-2017 methodologies employed statistical models and distributed representations—Word2Vec[1] and GloVe[2]—alongside primitive conversational systems like eliza[3].

The Transformer architecture[4] introduced in 2017 revolutionized nlp through self-attention mechanisms, enabling efficient parallel processing and long-range dependency modeling.

The Transformer's self-attention mechanism[5] facilitated Large Language Model development (2018-2021), characterized by parametric scale and sophisticated training paradigms: pre-training, few-shot learning, and reinforcement learning from human feedback.

The 2022 watershed marked by ChatGPT's deployment transformed llms from research artifacts to mainstream applications. GPT-3's[6] emergent reasoning capabilities established language models as inference engines.

Agent Formalization and Autonomy

An agent constitutes a computational system leveraging an ai model to reason, plan, and interact with environments through external tool execution to achieve specified objectives.

> The distinguishing characteristic of *agency* represents the capacity for autonomous action based on internal reasoning processes. This formalization encompasses both reactive and deliberative behavioral patterns.

This taxonomic progression from passive processing to autonomous workflow orchestration represents escalating agency levels. level 4 autonomy emerges when agents initiate and control execution cycles, including meta-agent spawning and coordination.

[1] Word2Vec pioneered distributed word representations, enabling semantic similarity computation through vector arithmetic operations.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013. URL https://arxiv.org/abs/1301.3781

[2] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. *EMNLP*, 2014. URL https://nlp.stanford.edu/pubs/glove.pdf

[3] J. Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966. doi: 10.1145/365153.365168. URL https://doi.org/10.1145/365153.365168

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Łukasz Kaiser, and I. Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017. URL https://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf

[5] Attention enables models to weigh the relevance of different input elements when processing each token, fundamentally improving context understanding.

[6] T. B. Brown, B. Mann, N. Ryder, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. URL https://arxiv.org/abs/2005.14165

Full autonomy manifests when agents can recursively instantiate other agents, creating hierarchical control structures that mirror biological neural organization.

Autonomy	Description	Implementation Pattern
LEVEL 0	Passive output processing	`process_llm_output(llm_response)`
LEVEL 1	Conditional control flow	`if llm_decision(): path_a() else: path_b()`
LEVEL 2	Dynamic function invocation	`run_function(llm_chosen_tool, llm_args)`
LEVEL 3	Iterative process control	`while llm_should_continue(): execute_step()`
LEVEL 4	Meta-agent orchestration	`if llm_trigger(): spawn_agent()`

Table 1: Hierarchical Agent Autonomy Levels

Language Models as Cognitive Substrates

Large Language Models constitute the foundational cognitive architecture for modern agent systems through their capacity for pattern recognition, contextual reasoning, and linguistic generation. Training on vast textual corpora enables LLMs to internalize linguistic patterns, semantic relationships, and domain-specific knowledge structures[7]. This emergent capability transforms static language models into dynamic reasoning engines.

[7] T. B. Brown, B. Mann, N. Ryder, et al. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020. URL https://arxiv.org/abs/2005.14165

Tokenization represents the fundamental computational unit whereby LLMs decompose linguistic input into discrete, processable elements—typically words or subword units.

Token vocabularies typically exceed 30,000 elements, balancing computational efficiency with semantic granularity.

Figure 1: **Playground**–Tokenization process by merging in Corpora.

Self-supervised learning exploits implicit textual structure, enabling training on unlabeled corpora:

$$\mathcal{D} = \{(x_t, \tilde{y}_t)\}_{t=1}^{N} \quad \text{where} \quad x_{t+1} = \tilde{y}_t$$

The attention mechanism computes similarity scores between queries and keys, then applies these weights to corresponding values:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}$$

Query (Q): Things I am looking for (adjectives).

Key (K): What do I contain? (adjective).

Value (V): What information is stored? ("awesome").

Attention enables the model to capture **long-range** dependencies within text sequences. The **query** of each individual token determines which other tokens receive higher attention weights. This mechanism supports efficient **batch processing** on GPUs. The output

of the attention layer is a **contextualized representation** of the input tokens, reflecting their relationships in context.

The attention operation computes how well each query matches the keys, then uses these scores to aggregate the corresponding values, producing a context-aware representation for each token.

The normalization factor $\sqrt{d_k}$ prevents dot products from growing excessively large, which would cause softmax to converge to one-hot distributions, reducing model expressiveness.

...

Index