



Open Discussion #2

VENICE
2025

Theme

Revisiting LLMs - Semantic Mismatching, Licensing Scheme, and Uncertainty Estimation of LLMs

IARIA Congress 2025 & DigiTech 2025



Open Discussion #2

VENICE
2025

Coordinator



PETRE – AI-related activities

VALENCIA
November 2023

- Petre: 1980/90
 - Fuzzy-based resource allocation, Automatic knowledge incorporation, CAD/CAM Expert Systems,
 - Real-time embedded systems, Space/time thinking and processing, Multi-layers context-based meaning
- Petre: 1992: The First ITC Conference (Montreal), tutoring systems, self-adaptable Q&A professor-student systems (advanced Chatbots)
- Petre: 1997 Dartmouth, Mobile Intelligent Agents (Intelligent Grasshopping Polling)
- Petre: 1997-2000: Nomadic code, Mobile agents, (Grasshopper EU project)
- Petre: 2000-2010: Autonomous systems, Policy-driven systems, Intelligent systems (pushed to Patents, ITU, TMF, standards)
 - Capturing emerging properties, Variable pooling frequency, Self-adaptable decision policies, Reflexive-policies (Digital-Twins)
 - Routers embedded-AI (temporal logic in Syslog processing, policy-driven signal processing)
- AI-driven Selection of Content Servers based on Current Server Availability (dynamic availability, heuristics, real-time)
- Petre: 2010 - now (active observer and critic, panels, open discussions)



Petre DINI
petre@iaria.org

Prof. Dr. Petre Dini
IARIA, USA/EU

At large: <http://www.iaria.org/fellows/PetreDini.pdf>



Topics

VENICE
2025

- Hallucinations (on purpose - they are human - vs induced - by the model -)
- LLM Resource Consumptions (precision vs tailored precision)
- Semantic Mismatching (dentification & disambiguation)
- LLMs, SLMs, tinyLMs (resource vs accuracy)
- LLMs vs LCMs (asset management)
- Licensing Scheme and Revenue Steam (critical uncontrolled changes)
- Agentic Framework (conflicting goals, agents farms)
- Uncertainty Estimation of LLMs (entropy, fuzzy/weights)



Open

VENICE
2025

**THE STAGE IS
YOURS**



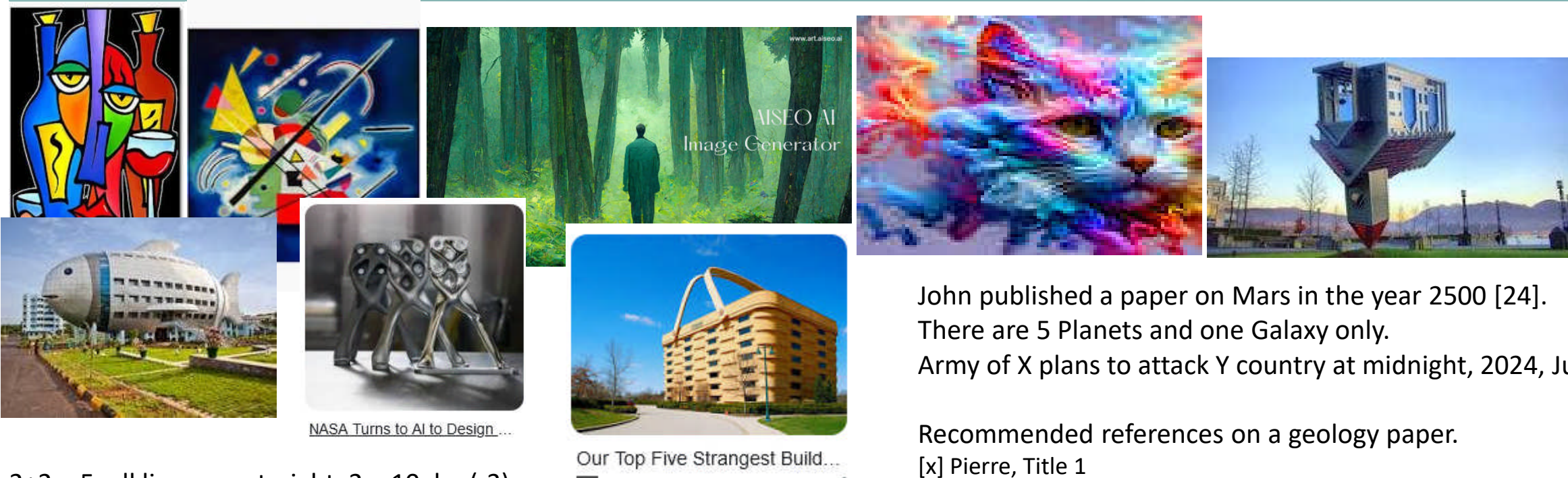
Back-up slides

**VENICE
2025**



Hallucinations

VENICE
2025



$2+2 = 5$, all lines are straight, $2 = 10$, $\log(-3) = x$
Horses ride a bicycle over the shining cloud of dust!!
Aliens will invest in The House of the rising Sun!
Alice's Adventures in the Wonderland

Metaverse
VR AVR
Immersion

John published a paper on Mars in the year 2500 [24].
There are 5 Planets and one Galaxy only.
Army of X plans to attack Y country at midnight, 2024, June 30.

Recommended references on a geology paper.

[x] Pierre, Title 1

[z] Jacobs, Title 2

[y] Stan, Title 3

Note: [x] doesn't exist

[z] has another title

[y] is a carpenter instructions book

LLMs-based: - very good summarization of information they are fed with, even only less than 1% validated as true
- very good mixed (4-5-6 ...) languages, correct punctuation, correct grammar, spelling correction on context-based intuition
- helpful at the informative level, like white papers, very quickly obtained and quite comprehensive
- assumes user's familiarity and experience with a given domain; see, selection an oscilloscope for 5G spectrum, financial aspects,...



Hallucinations - Q

VENICE
2025

Items under scrutiny

- a. Are human hallucinations more acceptable than machine hallucinations?
- b. Why can hallucinations of artificial machines occur?
- c. How to spot damageable hallucinations vs inoffensive hallucinations?
- d. How to improve the LLMs processes for minimizing (bad) hallucinations?



Resources - DeepSeek

VENICE
2025

DeepSeek's AI assistant surpassed OpenAI's ChatGPT in the Apple App Store
DeepSeek: DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via
Reinforcement Learning
arXiv:2501.12948v1 [cs.CL] 22 Jan 2025

2.2.1. Reinforcement Learning Algorithm

Group Relative Policy Optimization In order to save the training costs of RL, we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which foregoes the critic model that is typically the same size as the policy model, and estimates the baseline from group scores instead. Specifically, for each question q , GRPO samples a group of outputs $\{o_1, o_2, \dots, o_G\}$ from the old policy $\pi_{\theta_{old}}$ and then optimizes the policy model π_{θ} by maximizing the following objective:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi_{\theta_{old}}(O|q)]$$
$$\frac{1}{G} \sum_{i=1}^G \left(\min \left(\frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{old}}(o_i|q)} A_i, \text{clip} \left(\frac{\pi_{\theta}(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1 - \varepsilon, 1 + \varepsilon \right) A_i \right) - \beta \mathbb{D}_{KL}(\pi_{\theta} || \pi_{ref}) \right), \quad (1)$$

$$\mathbb{D}_{KL}(\pi_{\theta} || \pi_{ref}) = \frac{\pi_{ref}(o_i|q)}{\pi_{\theta}(o_i|q)} - \log \frac{\pi_{ref}(o_i|q)}{\pi_{\theta}(o_i|q)} - 1, \quad (2)$$

where ε and β are hyper-parameters, and A_i is the advantage, computed using a group of rewards $\{r_1, r_2, \dots, r_G\}$ corresponding to the outputs within each group:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \dots, r_G\})}{\text{std}(\{r_1, r_2, \dots, r_G\})}. \quad (3)$$



Resources - Optimization

VENICE
2025

Analogy: Syslog messages, Eye diopter, Nature Shapes Brain Captures,... (*context-dependent*)
(*running for 100% is context-dependent, resource optimization, including water & energy*)

LLMs: Stable/adopted vocabulary

(synonyms, idioms, ...)

(intensive process, repetitive, hardly reproducible, trillions of parameters, overfitting, underfitting, input balancing, hallucinations, ...)

DeepSeek: clustering (optimization), rounding meaning (similarity, sufficient approximation)

LCMs: ad hoc concepts, open range, uncontrolled definitions, concept management, definition conflicts, evolution control



Resources - Q

VENICE
2025

xLMs: tailored selection

Optimization

Rounding meaning (similarity, sufficient approximation)

LCMs

Concepts repositories

Concepts ownership

Domain-oriented tools selection

Accuracy-by-request

Precision-by-request

Authenticity-by-request



Semantic mismatching i

VENICE
2025

One LLM: : Polysemy, Pragmatism, Temporal aspects, Contextual roles, Discourse contradiction

- > User input: "What is the best bank in the city?"
- > LLM response: "The best river bank is the one with the nicest walking path..."
- > 📌 Mismatch: Interpreted bank as a geographical feature, not a financial institution.

- > User input: "Who painted the 'Mona Lisa'?"
- > Follow-up: "What else did she do?"
- > LLM response: Lists activities of the woman in the portrait
- 📌 Mismatch: Treated "she" as the subject of the painting, not maintaining focus on Da Vinci.

- > Prompt: "Explain why nuclear power is clean energy."
- > Later in same response: "Nuclear waste is a major environmental pollutant."
- > 📌 Mismatch: Coherence failure — two partially correct facts, but semantically contradictory when juxtaposed.

Disambiguation and Reconciliation Strategies

- A. Prompt Engineering Techniques
- B. Tool-Assisted Disambiguation
- C. Conversation Memory and Role Anchoring
- D. Model Fine-Tuning or Re-ranking
 - Train or fine-tune on domain-specific corpora with contextually grounded labels
 - Use re-ranking models to pick the best disambiguated answer from top-k completions
- E. Semantic Consistency Checkers
 - Deploy post-hoc semantic contradiction detectors
 - Ensure no mutually exclusive claims in the same output



Semantic mismatch ii

VENICE
2025

Between LLMs: Divergent Definitions, Concept Boundaries, Conflicting Scope, Context Assumptions, Domain-Specific Vocabulary, Inconsistent Entity Linking, Cultural or Regional Bias

- **Prompt:** *"Define what counts as a 'mild adverse reaction' in vaccine trials."*
- **LLM-A (e.g., GPT-4):**
"A mild adverse reaction includes local pain, slight fever, or fatigue that resolves within 24–48 hours."
- **LLM-B (e.g., Claude or LLaMA):**
"Mild reactions refer to non-serious symptoms that do not require medical intervention."
- 🖱️ **Mismatch:** Different **granularity** and **focus** — one uses clinical timing, the other legal/medical threshold.
- **Prompt:** *"What does 'freedom of speech' mean in the context of digital platforms?"*
- **LLM-A:**
"It is the right of users to express opinions freely, limited by platform guidelines."
- **LLM-B:**
"It is the protection from government censorship; platforms may set their own rules."
- 🖱️ **Mismatch:** Different **legal-cultural frames** — U.S. constitutional vs platform governance views.

Disambiguation and Reconciliation Strategies

A Prompt Standardization

Use controlled, context-rich prompts:

Add role (e.g., "as defined in EU regulation X")

Specify domain scope (e.g., "medical context", "U.S. law")

B Comparative Output Analysis

Run semantic similarity metrics (e.g., cosine similarity, BERTScore) on outputs

Use human-in-the-loop review or majority consensus for high-stakes answers

C Meta-Evaluation with Third Model

Use a neutral LLM to assess

D Ontology or Domain-Specific Schema Anchoring

Bind both LLMs to a shared controlled vocabulary, ontology, or taxonomy (e.g., SNOMED CT, WordNet, LexInfo)

E Trust Layer or Explanation Generator

Ask each LLM to explain:

"Why did you define this term this way? What sources or assumptions are you using?"



Mismatching - Q

VENICE
2025

Mismatch detection

Usually, post effect

Hardly detectable until damage is done

Disambiguation

Uncertainty

Human-in-the-loop

Extra-resources

License responsibility

Guarantees

Penalties

Outsourced disambiguation and reconciliation operations



LLMs, SLMs, TinyLMs

VENICE
2025

- **Size of languages models**

- Large, small, tiny
- Software, embedded (SoC)

- **Computation complexity**

- Trillion of parameters
- Repetitive computation

- **Optimisation**

- Finding the best patterns
- Accept adapted accuracy
- Clustering
- Drop extra computation cycles after a satisfactory outcome

- **Examples**

- Lenses/diopetre
- Syslog messages

- **Not all applications/services need exceptional accuracy**

- Classification of needs
- Clustering of exploration space
 - not all data have value
 - not everything deserves extra-computation power
 - powerful kids-like language models



Metrics

VENICE
2025

LargeLMs - SmallLMs - TinyLMs

Model Size (>1 billion parameters, 100 million - 1 billion parameters <100 million parameters)

Training Data (size and context-based datasets; unbalanced input)

Computation Needs (hardware, energy; tiny: run on edge devices/phones)

Latency (model complexity; tiny: suitable for real-time applications)

Use Cases (complex tasks, specialized tasks; tiny: lightweight applications, IoT devices, mobile apps)

Cost (high due to compute and storage requirements; moderate; minimal resources)

Accessibility (accessible via cloud APIs, easily deployable; tiny: embedded, minimal overhead)



LLMs/SLMs/TinyLLMs - Q

VENICE
2025

xLMs: choosing the correct size
choosing the tailored model

Hardware/Software
evaluate the resources

Resource evaluation

Uncertainty estimation



LLMs over LCMs

VENICE
2025

Here's the big picture:

```
sql
```

Copy

Edit

User ↔ LLM ↔ LCM ↔ External Knowledge / Tools

- **LLM** handles: natural language understanding, interaction, generalization
- **LCM** handles: structured concept grounding, validation, abstraction, explanation
- **Goal:** use LCM to "filter", validate, or enrich the LLM's decisions/actions



Concept Mapping

VENICE
2025

Concept Mapping Layer (Bridge)

Converts LLM-parsed input into conceptual tokens using:

Schema mapping (e.g., concepts from ConceptNet, DBpedia)

Ontology alignment (e.g., OWL-based matching)

Embedding clustering or symbolic tagging

text

Copy

Edit

"Explain gravity on the Moon."

↓

["Gravity", "Moon", "Mass", "Acceleration"]



LCM Core Layer

VENICE
2025

This is the reasoning **brain**, composed of:

Knowledge Graph: concepts + relations

Reasoning Engine: for inference (e.g., description logic, rule engines)

Conceptual Simulation: mechanisms for analogical or causal reasoning

Could be built with:

Component Technologies

- | | |
|--------------|-------------------------------------|
| > Ontologies | OWL, Protégé |
| > Graph DBs | Neo4j, RDF/SPARQL |
| > Reasoning | Pellet, Hermit, Rule Engines |
| > Learning | Meta-Concept Embeddings (Vec2Graph) |
| > Inference | CLIPS, MiniKanren, ASP |



LLMs vs LCMs

VENICE
2025

Feature / Aspect	LLM (Large Language Model)	LCM (Large Concept Model)
🔍 Primary Objective	Model linguistic patterns and generate coherent text	
📦 Core Representation	Tokens, word embeddings, attention distributions	
📖 Training Data	Text corpora (web, books, conversations, code)	
🧠 Learning Style	Pattern learning via self-supervised next-token predictions	
🌀 Compositionality	Implicit, via transformer layers and positional embeddings	
🔄 Reasoning	Mostly statistical (e.g., analogies, surface inferences)	
🌐 Contextuality	Strong in recent context windows (e.g., 128k tokens)	
🔑 Interpretability	Often opaque (“black-box”)	
👤 Use Cases	Text generation, dialogue, summarization, translation	
🔧 Tool Integration	Prompt-based, external tools via plugins	
🧠 Generalization	Strong zero-shot/few-shot generalization in language tasks	
⚡ Limitations	Hallucinations, shallow semantics, weak long-term abstraction	



LLM, LLC - Q

VENICE
2025

Experience reports

Concept management (validation, repositories, etc.)

Concept mismatching (conflict resolution, mediation, etc.)

Funneling LLMs into LCMs, (... then: to coding, testing, validating)






How to adopt such models for real systems



Revenue Stream and Licensing





VENICE
2025

Revenue streams

-  **B2B Licensing** License the system as a SaaS or on-prem platform (monthly/annual tiers)
-  **Customization Services** Custom workflows or modules tailored to client processes
-  **API as a Service** Expose SME-layer functionality via custom APIs
-  **Integration Add-ons** Plugins for CRMs, ERPs, sector-specific software (education, law, medical)
-  **Analytics / Reporting** Premium dashboards or compliance modules
- CRM** Customer Relation Management
- ERP** Enterprise Resource Planning

Licensing strategy

Use tools under licenses that allow derivative/commercial work or that are “as-a-service” (API-based)

Tool Type	Example	SME Strategy
 LLMs	OpenAI, Claude, Cohere	Use via API – no model redistribution
 Open-source libs	spaCy, HuggingFace, FastAPI	Comply with licenses (MIT/Apache = safe)
 Open-core	LangChain, Supabase	Use core, pay for commercial support if needed
 Cloud APIs	AWS Comprehend, Azure ML	Usage-based costs; no IP lock-in

Be careful with:

GPL code (may require open-sourcing your product)

Redistributable binaries if licensing terms are restrictive

GNU GPL: General Public License (GPL)






TP-LINK GPL code:
Third Party : partly contain software code developed by third parties, including software code subject to the GNU General Public Licence (“GPL”), Version 1/Version 2/Version 3 or GNU Lesser General Public License(“LGPL”)



Roles / Tools / Licensing

VENICE
2025

Team Roles & Assignments

Role	Name / Notes	Time Allocation
 AI Architect	Engineer	Full-time / Part-time
 Prompt Designer	Evaluator	
 Domain Expert	Product Owner	
 Full-stack Developer		
 Legal	Compliance Advisor (optional/external)	

Core Tools & Frameworks

Tool Type	Selected Tool	Why This Tool?
LLM API	(e.g., OpenAI GPT-4, Claude, Mistral)	
Concept Engine	Graph	(e.g., Neo4j, RDF)
Prompt Chains	Lang	(e.g., LangChain, CrewAI)
Embedding Store	(e.g., Chroma, Pinecone, FAISS)	
UI / App Framework	(e.g., React, Streamlit, FastAPI)	

Iteration & Evaluation Plan

Evaluation Focus	Approach / Metric	Frequency
Output Quality	Human eval + metrics (BLEU/Faithfulness)	Weekly
User Feedback	Real-user pilot, scorecards	Monthly
Cost Optimization	Token usage monitoring	Every sprint

Risk & Licensing Checklist

Area	Action Required	
Status		
API Terms	Review commercial use terms	<input checked="" type="checkbox"/>
Data Privacy	Compliant with GDPR / local regs	<input type="checkbox"/>
Open Source	Track LGPL/GPL obligations	<input type="checkbox"/>
Prompt/IP Boundaries	Avoid sensitive reuse	<input type="checkbox"/>



Licensing - Q

**VENICE
2025**

Licensing team of experts

Settlement of new code ownership

Dealing with versioning

Dealing with change in license conditions



AI Engineer (Agent) - Virtual Entity

VENICE
2025

Core Capabilities of a Virtual "AI Engineer"

Capability

- 📦 Model Architecting
- 🔧 Hyperparameter Tuning
- ⚙️ Pipeline Automation workflows
- 📁 Deployment Automation
- 📄 Documentation & Reporting
- 🔄 Continual Learning Mgmt
- 🔍 Debugging Assistant
- 🔒 Bias and Compliance Checks

Description

- Selects models based on data type (e.g., CNNs, LLMs, GNNs)
- Uses optimization techniques (Bayesian, Grid Search)
- Builds end-to-end data/model training
- Pushes models into staging or production environments
- Auto-generates experiment reports, code comments
- Suggests or implements retraining schedules
- Identifies performance regressions, training bugs
- Flags fairness, privacy, or explainability issues

Underlying Technologies

- > 🧠 LLMs | GPT-4, Claude, Mistral, or open-source models (fine-tuned for engineering tasks)
- > 🌀 Agent Frameworks | LangChain, AutoGen, CrewAI, AgentVerse
- > 🛠️ Tool Plugins | Code runners, databases, version control, Docker, etc.
- > ⚙️ Memory/Planning | ReAct, Chain-of-Thought, RAG, scratchpads, vector memory
- > 📁 Environment | Often containerized (Docker, Replit, VSCode in-browser)

Examples of Platforms Creating "AI Engineer" Entities

> Devin by Cognition

A fully autonomous AI software engineer that can plan, write, debug, and test code with no human intervention (still in preview)

> GitHub Copilot X (with Agents)

Goes beyond code completion; can scaffold apps, generate entire classes, and collaborate over time.

> AutoGPT / AgentGPT

Experimental open-source agents that can be instructed to achieve high-level engineering goals via tool use and iterative planning.

> OpenDevin (open-source fork)

Tries to mimic Devin's architecture — acts like a terminal-based AI engineer that uses planning + code execution.

> C> odeWhisperer (AWS) and Tabnine

Autocomplete-style assistants but heading toward semi-autonomous behavior.

> LangChain Agents / CrewAI

Build modular agent teams: one can play the "AI engineer" role in an LLM-powered workflow.



Agentic Engineering

VENICE
2025

An **"AI Engineer"** as a virtual agent (an autonomous LLM-based entity that performs AI engineering tasks),
Agentic Engineering as a discipline or paradigm (engineering systems of agents that plan, act, and learn over time).

AI Engineer (as a virtual agent)

This is a specialized role or embodied skillset within a broader system.

It refers to an LLM-powered autonomous agent that:

- Writes code, designs models, debugs, deploys
- Acts like a virtual software engineer
- Is task-focused (e.g., "build me an object detector")
- May use planning, tool use, memory, and execution environments (e.g., shell, browser, Python interpreter)

Example:

Devin, GitHub Copilot + agents, or a LangChain/CrewAI agent with the "AI Engineer" role.

Agentic Engineering

This is a new field of engineering focused on the design, orchestration, and safety of intelligent agents, especially LLM-based ones.

It involves:

- Creating multi-agent ecosystems
- Managing goals, delegation, planning, negotiation
- Enforcing safety, alignment, and controllability
- Addressing non-determinism, long-horizon actions, and memory evolution

Related challenges include:

- Tool integration
- Agent teaming and coordination
 - Goal disambiguation and intent refinement
- Autonomy vs. oversight balancing
- Temporal abstraction (short tasks vs. lifelong learning)



AI vs Agentic Engineering

VENICE
2025

Their Relationship

Aspect || AI Engineer (Agent)

- > 🗣️ What || A software agent that performs AI tasks
- > 🧠 Cognitive Role || Acts as a specialized skill worker
- > 📁 Technologies Used || LLMs, memory, RAG, tool APIs
- > ⚙️ Output || Trained models, deployed pipelines
- > 🔄 Scope || One agent with a fixed or growing role
- > 🌱 Example || Devin generating a neural net

|| Agentic Engineering

- || A discipline to build, manage, and evaluate agents
- || Designs systems of cognition and delegation
- || Agent platforms, safety modules, coordination logic
- || Robust multi-agent systems, reliable interfaces
- || Multi-agent environments, emergent behaviors
- || CrewAI orchestrating 5 agents for research

What Is Goal Generation in Agentic Systems?

Goal generation is the process by which an agent (like the AI Engineer) determines what it should do next. This includes:

- > Recognizing new needs or opportunities
- > Transforming open-ended tasks into actionable objectives
- > Aligning tasks with long-term system purpose or constraints

1. Components of Goal Generation

Source || Example

- 🗣️ Human Prompt || “Build a model to classify pneumonia in X-ray images”
- 🧠 Self-reflection || Agent detects pipeline drift and sets a goal to retrain
- 📁 Environment State || New data availability triggers model update
- 🔄 Upstream Agent || A supervisor agent delegates “optimize hyperparameters”



Agentic Framework- Q

VENICE
2025

How to align business goals, problem solution, and agents goals?

Automatic Goal Generation?

Similarity with what an LLM produces these days, e.g., the plans they produce?

How would you formalize this?

Ho to ensure they solve the problem?

What about outcome quality?

What about conflicting goals, the design trade-offs?

Establishing robust infrastructure and processes



Uncertainty Estimation i

VENICE
2025

Themis AI - CSAIL Alliances - MIT
<https://cap.csail.mit.edu>

- **Types of Uncertainty**

- a. *Aleatoric Uncertainty* (data-based)

- Comes from inherent noise or ambiguity in the input (e.g., multiple valid interpretations).

- b. *Epistemic Uncertainty* (model-based)

- Stems from a lack of knowledge or data; reducible with more/better training data.

- **For LLMs Specifically**

- Prompting Variability*: Asking the same question in different forms and comparing the answers can reveal uncertainty.

- Chain-of-Thought Diversity*: Sampling multiple reasoning chains and analyzing their consistency.



Uncertainty Estimation ii

VENICE
2025

Common Uncertainty Estimation Techniques

1. Log Probability / Token-Level Entropy

The model's log-probability distribution over tokens can be used to estimate uncertainty.

Higher entropy = more uncertainty.

Useful in: Language modeling, Autocomplete confidence, Chain-of-Thought token path variability

2. Monte Carlo Dropout

Apply dropout during inference (not just training) multiple times.

Analyze the variance in outputs to estimate uncertainty.

Pros: Simple, applicable to Transformer layers.

Cons: Adds computational cost.

3. Ensemble Methods

Use multiple LLMs trained differently or with different seeds.

Diverse outputs or high disagreement → higher uncertainty.

Especially used in high-stakes applications (e.g., clinical or legal generation).

4. Temperature Scaling (in softmax)

Tuning the softmax temperature changes output confidence.

High temperature → more uniform predictions → higher estimated uncertainty.

5. Bayesian Approaches

Treat model parameters as probability distributions (e.g., Bayesian Transformers).

Often too heavy for production-scale LLMs but used in research.

6. Conformal Prediction

A statistical framework providing prediction intervals or sets.

Can be adapted for NLP tasks, like classification or span extraction.

7. Calibration Metrics

Measures how well the model's confidence corresponds to reality.

Common ones:

Expected Calibration Error (ECE)

Brier Score

Negative Log Likelihood (NLL)



Uncertainty Estimation - Example

VENICE
2025

Medical Decision-Making Support

- **Goal:** Use an LLM (like Med-PaLM or GPT-based tools) to assist with clinical reasoning, diagnosis, or summarizing reports, while estimating confidence in its recommendations.
- 🔍 **Technique: Token-Level Entropy + Ensemble Sampling**
- **Setup:**
- Task: Answer clinical questions based on a patient case or radiology report.
- Model: An LLM fine-tuned on biomedical corpora.
- Approach:
 - Sample **multiple answers** using different temperatures or prompt phrasings.
 - Measure **token-level entropy** and **output diversity**.

Example:

Prompt: “What is the most likely diagnosis for a 65-year-old patient with chest pain, elevated troponin, and ST elevation in leads II, III, aVF?”

LLM answers:

Myocardial infarction (common)

Inferior STEMI (more specific)

Acute coronary syndrome (more general)

Uncertainty Estimation:

Token entropy for each word is low for “myocardial infarction” → high confidence.

When more diverse responses appear (e.g., pulmonary embolism, angina), entropy increases → signal to escalate to human.

Metric Used:

Entropy (H) of output tokens

$$H = -\sum_{i=1}^n p_i \log p_i$$

$$H = -\sum_{i=1}^n p_i \log p_i$$

(where p_i are token probabilities)



Uncertainty Estimation - Q

VENICE
2025

Still subjective

Entropy – extra
computation
resources

Weights / ~ fuzzy
logic

Probabilities

Token-Level Entropy

For the example medical phrase completion:

Tokens: "myocardial infarction", "angina", "pulmonary embolism"

Assigned probabilities: [0.75, 0.15, 0.10]

Computed Entropy:
 $H = -\sum p_i \log_2 p_i \approx 1.05$ bits

→ This is low entropy, indicating high confidence in the top token ("myocardial infarction").



END

**VENICE
2025**

END