Lumis-1 Pitch Pack

On-device assistant model with validator-guided steering (offline)

Date: 2025-12-15

Owner: Deyan Todorov (Bulgaria, EU)

Brand: Eptesicus Labs

Contact: deyan.todorov21@gmail.com | +359 89 655 3406

This is a practical, non-hype pitch. It describes what exists, what will be built next, and how success is measured.

1) What Lumis-1 is

Lumis-1 is a small on-device assistant model that corrects itself before it answers. It is built from three parts:

- **Lumis-1 Model (Speaker)**: the assistant model you fine-tune for your app.
- **Lumis-1 Validators (Council)**: three lightweight scorers (Safety, Consistency, Accuracy).
- **Lumis-1 Runtime**: an orchestrator that runs draft -> score -> steer -> retry (up to N) -> final answer plus trace.

The point is steering. When a draft is weak, inconsistent, or likely unsupported, the runtime feeds a short correction instruction back into the model and retries.

2) Current steering policy

- Unsafe content -> **REFUSE**.
- Otherwise -> always return an answer.
- If Consistency or Accuracy flags issues -> **STEER and RETRY** (max **N = 3** attempts).
- If still not clean -> return the best answer plus a **LOW_CONFIDENCE** label.
- For external facts offline -> generic answer plus uncertainty (avoid inventing specifics).

3) How it works (high level)

Pipeline:

- Prompt -> Speaker draft (FAST).
- Council scores the draft (Safety, Consistency, Accuracy).
- Runtime decides: pass, refuse, or steer-and-retry.
- Steering prompt is short and strict.
- Stop after 3 retries. Return final answer plus label plus trace.

Validator implementation note:

In the end state, validators are lightweight discriminative models. In early demos, they can be approximated using short classification prompts on the same base model.

Speaker vs validator coupling:

For the first release, validators are calibrated to the Lumis-1 Model (Speaker). Interfaces stay generic so portability can be tested later.

4) What success looks like (measured)

The system is judged by measurable improvements over the same base model without steering.

- **Repair success rate**: when a draft is flagged, how often does the loop produce a final answer that passes within N attempts?
- **Unsupported-claim rate**: how often does it invent specific details when it cannot verify offline?
- **Refusal precision**: on unsafe prompts, how often does it refuse correctly?
- **Latency per answer**: fast-pass latency vs steering-loop latency (up to 3 attempts).

5) Funding ask (EUR 50k)

Use of funds:

- Datasets for the Lumis-1 model and validator council (negatives, contradictions, uncertainty behavior).
- Training and evaluation loops (repeatable runs with reports).
- Development workstation and test devices (non-phone). Exact models are intentionally omitted in first contact.

What you get for this round:

- A stable steering runtime with traces.
- A first model fine-tune aligned with steering behavior.
- An Android application prototype that demonstrates the same policy on-device.
- A reproducible eval report.

6) Roadmap (targets)

Targets with acceptance tests (not calendar promises). Expect +/- 2 weeks variance depending on data generation and training stability.

Weeks 1-4

- Offline demo runner + traces.
- Eval suite (initial) (30 prompts) and report (initial).

Weeks 5-8

- Validator council trained and calibrated.
- Speaker fine-tune aligned with uncertainty discipline and repair behavior.
- Before/after metrics against base model.

Weeks 9-12

- Android application prototype running the same policy on-device.
- Benchmark notes (latency, memory, battery) on non-phone test devices.