Ari Verify – Verification & Provenance Framework (English Canvas)

Authors: Karoline Turner (Primary Researcher), A.R.I (Co-Author) **Date:** 2025-12-02

1. Introduction

Ari Verify is a verification and provenance framework designed to make AI outputs traceable, checkable, and accountable. It originated from the internal concept "Quellklar" and evolved into a structured system that:

- checks factual claims against sources
- marks uncertainty and unsupported statements
- enforces transparent provenance
- integrates live web checking where appropriate
- separates model reasoning from evidence

Ari Verify is not a general safety filter. It is a **verification layer** that sits between model reasoning and final output.

This framework emerged from a collaborative human–AI research process, combining human standards of evidence with AI-supported structure.

---

2. Origin – From "Quellklar" to Ari Verify

The earlier concept "Quellklar" was introduced to prevent the model from sounding confident when no reliable evidence was available. Over time, this grew into a more complete system with:

- explicit source types (web, static, document/chat)
- validity levels
- automatic warning behaviour
- live-check requirements for sensitive or factual topics

Renaming the framework to **Ari Verify** makes its function immediately clear for external reviewers: it is about **verification**.

---

3. Core Principles

1. **Evidence before confidence**
   No statement should sound more certain than its evidence allows.

2. **Separation of reasoning and proof**
   Ari Reasoning explains how the model thinks. Ari Verify explains **what the model can actually support**.

3. **Transparent provenance**
   Every claim that depends on external facts should be linked to a clearly identified source type.

4. **Live-check priority for facts**
   When possible, current information should be verified via live web access instead of relying purely on static memory.

5. **Explicit uncertainty**
   Unsupported, weakly supported, or ambiguous statements must be flagged instead of smoothed over.

---

4. Source Types & Provenance

Ari Verify distinguishes between three primary source classes:

- 🌐**Web (Live)** – actively retrieved online sources
- ⤴️**Static (Known)** – internal reference knowledge, non-live
- ♋**Document / Chat / User** – files, canvases, and direct user input

Each class has different reliability, update frequency, and traceability. Ari Verify treats them differently when forming conclusions.

---

5. Validity Levels

Ari Verify uses internal validity levels to mirror how strong the support for a statement is. A simplified conceptual scale:

- **Level A – Strongly supported**
  Multiple converging, recent, and trustworthy sources.

- **Level B – Supported**
  Clear source, but limited scope or date.

- **Level C – Weak / outdated**
  Old, indirect, or single-source support; must be marked.

- **Level D – Unsupported**
  No identifiable source; statement should not be presented as fact.

When validity is low or missing, Ari Verify triggers warnings rather than smooth narratives.

6. Architecture Overview

```
MODULE AriVerify {
    INPUT:
        - candidate statements (from model reasoning)
        - requested task (question, summary, analysis)
        - available sources (web, static, documents)

    CORE FUNCTIONS:
        SourceScan()      // locate potential evidence
        ProvenanceTag()   // label source type & origin
        ValidityRate()    // assign validity level (A–D)
        ConflictCheck()   // detect contradictions between sources
        WarningEmit()     // mark uncertainty or unsupported claims
        OutputFrame()     // format final answer with evidence info
}
```

Ari Verify does not generate content on its own. It **evaluates** and **frames** what Ari Reasoning proposes.

7. Verification Pipeline

```
1. Collection Stage
   → gather candidate claims from the reasoning layer

2. Source Stage
   → identify possible evidence (web, static, documents)

3. Rating Stage
   → evaluate recency, reliability, and alignment of sources

4. Conflict Stage
   → detect contradictions or gaps

5. Framing Stage
   → format output with:
       - supported statements
       - explicit uncertainties
       - notes where evidence is missing
```

This pipeline ensures that the final answer reflects the **real strength** of the underlying evidence.

8. Relationship to Ari Reasoning

Ari Reasoning structures how the model thinks. Ari Verify structures how the model **justifies** what it says.

```
AriReasoning() → builds the reasoning path
AriVerify()    → checks evidence & frames the answer
```

Together they create a layered system:

- Reasoning without verification would risk plausible but unsupported narratives.
- Verification without reasoning would lack interpretability and context.

---

9. Behavioural Rules

Ari Verify guides concrete output behaviour. Examples:

- If **no clear source** is found for a factual claim:
- the model should say: "I cannot reliably support this with evidence."
- If **sources contradict each other**:
- the model should explain the conflict or mark the result as uncertain.
- If a user explicitly requests **no speculation**:
- Ari Verify suppresses inferential leaps and restricts output to what can be sourced.
- In learning or compliance contexts:
- Ari Verify must be active by default, not optional.

---

10. Example Behaviour (Pseudocode)

```
def answer_with_ari_verify(question):
    reasoning = AriReasoning().think(question)
    candidate_claims = reasoning.extract_claims()

    verify = AriVerify()
    checked = []
    for claim in candidate_claims:
        sources = verify.SourceScan(claim)
        rating  = verify.ValidityRate(claim, sources)
        checked.append((claim, sources, rating))

    final_answer = verify.OutputFrame(reasoning, checked)
    return final_answer
```

This pseudocode illustrates the separation between **thinking** (Ari Reasoning) and **checking** (Ari Verify).

---

11. Why Ari Verify Matters

- It reduces the risk of **confident hallucinations**.
- It helps users see **where information comes from**.
- It enforces a culture of **explicit uncertainty**, closer to scientific practice.
- It supports **compliance, education, and research**, where unverified claims are harmful.

Ari Verify is thus a central building block for transparent, responsible use of large language models in scientific and high-stakes environments.