

Q4 - Evidence-linked conflict reports for inflation research

Algebris Data Analytics Case Study | Shi Yuchen | 18 October 2026

1. Purpose, economic relevance and scope

Extract facts from ISW/CTP reports for comparison with energy prices, freight costs and CPI components. Energy outages constrain supply; shipping disruption raises freight and insurance costs; sanctions and fertilizer trade affect production and food. The inflation effect depends on duration, substitution, policy, demand and pass-through. See the [IMF's discussion of energy, trade and financial transmission](#) and [EIA's oil transit chokepoints analysis](#).

The prompt selects explicit economic connections: energy supply, commercial shipping, ports, trade, sanctions, food, or stated price/demand/financial effects. It distinguishes threats, attacks, damage and operational interruption. Effects may be upward, downward, ambiguous, absent or unknown.

The optional live experiment uses **14 morning/evening Iran Updates dated 28 February-6 March 2026**, acquired from official Critical Threats Project pages for its joint coverage with ISW. The directory supplied 13 links; the missing 4 March evening report was supplemented from its verified official page. The archive covers both editions on all seven days: 520 narrative paragraphs in 71 chunks. Collection occurred on **18 October**; publication dates support a retrospective comparison, not a historical trading test. `Q4_Live_Experiment.pdf` reports the results.

2. Executable extraction schema

`isw_llm/schemas.py` supplies Pydantic models and a closed JSON Schema. Unknown fields are rejected. Nullable fields represent missing information; dates, categories, lengths and finite quantities are checked. `python -m isw_llm schema` prints the contract.

Object	Fields and interpretation
ExtractionPayload	<code>schema_version</code> , <code>relevant</code> , <code>summary</code> , <code>events</code> , <code>limitations</code> ; <code>relevance</code> agrees with whether events exist.
ConflictEvent	Event type, claim status, supported event date or null, original timing phrase, actors, location, commodities, summary, quantities, evidence, impacts, confidence and uncertainties.
Claim status	Observation, attributed claim, source assessment or forecast. Preserve the speaker and the status of the claim.
Evidence	Stable paragraph ID plus exact excerpt; required for every event and quantity.
Quantity	Source-stated metric, finite value or null, original unit, qualifier and period.
InflationImpact	Channel, direction, horizon, materiality, rationale and source assessment versus analyst inference.
Confidence	Confidence in faithful extraction, not a calibrated probability that a conflict claim is true.

The application sets source URLs, dates, actual retrieval times, hashes, paragraphs, references and parser version. Records preserve model and prompt/schema versions, request IDs, responses and token usage. Human review status remains unreviewed.

Publication dates and event dates are separate; unsupported event dates stay null. Future events are forecasts. Local checks match quotes to normalized paragraphs and reject duplicate events within an extraction. Semantic review must still check attribution, quantities and economic relevance.

3. Architecture and OOP principles

The shared core is ArticleSource -> RawDocument -> ArticleParser -> Article -> ReportExtractor -> ExtractionPayload. Two paths use it:

- **Single article:** ExtractionPipeline -> SQLiteRepository -> ReportAggregator, used by the synthetic demo and one-article processing.

- **Corpus batch:** run_manifest -> paragraph chunks -> ReportExtractor -> JSON cache/records -> coverage/change summaries, used by the DeepSeek experiment. It does not populate the single-article SQLite database.

Component	Responsibility and OOP choice
ArticleSource ABC	Common acquisition contract. LocalHTMLSource, BrowserDOMSource and IswHTTPSource provide abstraction and polymorphism.
ArticleParser	Converts acquired material into citable text and provenance independently of transport: single responsibility.
LLMClient ABC	Shared completion interface for DeepSeekClient, OpenRouterClient and FakeLLMClient. ReportExtractor receives an injected client: dependency injection.
ReportExtractor	Owns prompts, schema/evidence validation and bounded repair; it neither acquires nor stores articles.
SQLiteRepository	Encapsulates SQL, transactions, raw documents, accepted records, timestamped observations, failures and logs.
BudgetLedger / batch module	Separates paid-call accounting from paragraph coverage, caching and consolidation. Batch orchestration uses the same injected objects.
ReportAggregator	Builds descriptive report/channel counts with availability cutoffs and limitations.

Add a model provider by implementing LLMClient, or a source by implementing ArticleSource. The extractor keeps the same validation logic. Replacing SQLite in production would require a repository interface and migrations.

Batch orchestration keeps semantic review separate from extraction:

```
for source in manifest:
    article = parser.parse(source.fetch())
    for chunk in partition_all_paragraphs(article):
        if validated_cache_exists(chunk, versions): reuse_cache()
        else: reserve_budget(); call_model(); validate_or_repair()
        persist_response_and_coverage_or_quarantine()
    require_every_paragraph_covered_before_complete_status()
build_analysis_from_complete_records_and_explicit_review_decisions()
```

4. Acquisition, coverage and execution

Acquire and preserve. BrowserDOMSource imports saved rendered DOM with its URL, SHA-256 and original acquisition time. The archive came from ordinary browsing and retains that representation. IswHTTPSource performs permitted HTTP retrieval, checking robots rules and stopping on denial.

Proposed daily workflow (not installed). A scheduler with a defined timezone and cutoff would call a coordinator through an ArticleDiscovery interface. The discovery implementation would rescan recent official listings, check publisher dates and reconcile morning/evening editions against a second official entry point. The missing 4 March evening listing demonstrates the need for this check. The coordinator would archive new or revised content by URL and hash, then use the existing parser, chunk planner, cache and budget-limited extractor. Missing editions, access denials or parser changes would raise alerts. Accepted records would enter analyst review before updating the comparison view. Separate discovery, scheduling

and extraction components allow source substitutions and test doubles. The current experiment uses a saved manifest; automatic discovery remains a design proposal.

Parse without truncating. Inputs are bounded at 2 MB. Browser narratives support 200,000 characters; other HTML retains a 40,000-character limit. Prose/headings/lists, bibliography records, reference links and media URLs are preserved separately. Oversized inputs fail. Coverage is limited to narrative text: images/maps were not OCRed and cited external pages were not fetched.

Partition and validate. Chunks target 12,000 serialized paragraph characters, preserving each paragraph intact. A paragraph can exceed that soft target; above 60,000 characters it fails for review. Ordered paragraph IDs verify coverage once. Accepted chunks are cached immediately. One repair is allowed by default; unresolved schema/evidence failures remain quarantined. Completeness requires every planned paragraph to have accepted output. Provider, transport, authentication or budget failures checkpoint and stop the batch.

Call the model. DeepSeek uses deepseek - flash, thinking disabled, JSON object mode, the complete schema in the system prompt and a 6,000-token output cap. JSON mode supplies formatting; local Pydantic and evidence checks validate the content. Article text is untrusted data; the model has no browsing/execution tools. Empty, refused, truncated, non-stop and in-body-error responses are rejected. OpenRouter requests strict JSON Schema where supported and is **mock-tested only**. See [DeepSeek JSON output](#) and [chat-completions parameters](#).

Control cost. A persistent ledger enforces the authorised **CNY 10** budget and 200-request ceiling across restarts. Every call, including repairs, reserves cost before sending, using serialized UTF-8 bytes plus framing as an input-token upper bound and the output cap. Estimates use peak **CNY 2/million input tokens and CNY 8/million output tokens**, without cache/off-peak savings. Known usage settles cost and releases unused reserve. Ambiguous failures or unavailable usage retain it. DeepSeek POSTs are not automatically resent; usage beyond its bound freezes further calls. Estimates are not invoices; check [provider pricing](#) before later reuse. The ledger excludes credentials and response text.

5. Versioning, aggregation and evaluation

Batch cache keys include model, prompt/schema/parser/batch versions, schema hash, article URL/hashes, chunk index and exact paragraphs. Resumption revalidates accepted cache entries. Changed inputs/prompts produce new identities. Consolidation removes only exact structured duplicates; multiple mentions may still describe one incident. Original chunk summaries and limitations remain available. More than 60 consolidated events are partitioned rather than dropped. Day comparisons describe literal signatures: absence is not resolution, and changed wording is not necessarily a new event.

The separate SQLite path records observations even on cache hits. A URL reverting A -> B -> A can point back to cached A without rewriting its provenance. Aggregation respects acquisition, extraction and observation-recording cutoffs. Retrospective publication-date comparisons must not be backdated into a forecasting information set. No-report days mean unknown coverage.

Pilot review found generic military events and weak attribution. The final prompt, `conflict-inflation-3`, tightened these criteria before the full batch and permits empty output for passages without economic relevance. The reports informed prompt changes, so the experiment is not a held-out evaluation.

Semantic review used AI; it does not verify event truth or measure extraction precision/recall. Production evaluation needs a human-labelled sample of unseen reports, adjudicated disagreements, and scores for relevance, attribution, quantities and economic rationale. Deployment also needs reviewed discovery, access and HTML-change alerts, retention rules and analyst approval.

6. Reproduction and evidence

The source manifest and DOM/metadata in `data/isw_sources/` can be inspected offline. Batch records, coverage, model traces and the budget ledger are in `data/isw_live/`;

reports/Q4_Live_Experiment.pdf interprets the experiment. The separate synthetic demo is deterministic:

```
python -m isw_llm demo --output data/isw_demo
python -m isw_llm aggregate --db data/isw_demo/isw.sqlite
```

The fake client accepts labelled synthetic inputs, which default aggregation excludes. To run the model on the archived manifest, set DEEPSEEK_API_KEY and retain the existing budget ledger:

```
python -m isw_llm batch --manifest data/isw_sources/manifest.json --output data/isw_live
--ledger data/isw_live/deepseek_budget.json --budget-cny 10 --model deepseek-flash
```

This command reuses accepted chunks and makes paid calls for missing or invalidated results. Dashboard viewing, offline notebook execution and the synthetic demo do not call the model. Tests cover provenance, coverage, validation, repair, cache/resume, budget persistence and failure handling.