Assignment 3 – Data Pipeline Deliverable

Project: Madison Agent: Model Behavior Evaluation System

Problem: API teams lack integrated tooling to evaluate LLM behavior consistently against real-world prompts and high-quality reference responses.

Overview

This project builds a reproducible n8n data pipeline that collects, cleans, validates, and stores datasets required by the Madison Agent. The goal is to provide:

1. Real-world, continuously updated AI content (research + news)

2. Ground-truth instruction–response examples for behavioral comparison

The workflow runs locally in n8n, integrates three Tier-1 data sources, applies validation and normalization, and exports the results in structured CSV and JSON formats for reuse in Assignment 4.

Data is collected once and saved to disk to avoid repeated API calls and rate-limit issues.

Part 1 – Working n8n Data Collection Workflow

Data Sources

Source 1: ArXiv Computer Science – AI RSS Feed

- Type: Academic research papers (title, abstract, URL, publication date)
- Amount: 100 records
- Purpose: Provides current research context and emerging model capabilities
- Collection Method: RSS feed via n8n HTTP Request + XML node
- Quality: 100/100 complete records

Source 2: Smol AI News RSS Feed

- Type: Industry AI news and research updates
- Amount: 10 records
- Purpose: Adds practical industry context and deployment trends
- Collection Method: RSS feed via n8n HTTP Request + XML node
- Quality: 10/10 complete records

Source 3: AlpacaEval Ground Truth Dataset (HuggingFace)

- Type: Instruction–response pairs with evaluation metadata
- Amount: 80 records
- Purpose: Provides high-quality baseline examples for evaluating model outputs
- Collection Method: HTTP Request to HuggingFace public dataset API + Code nodes
- Quality: 80/80 complete records

Workflow Summary

The n8n workflow performs the following:

1. Collects RSS content from ArXiv and Smol AI
2. Fetches AlpacaEval examples from HuggingFace
3. Normalizes schemas across RSS feeds
4. Extracts and flattens AlpacaEval records
5. Applies validation rules:
    - Empty summaries removed
    - Empty Alpaca responses removed
    - Minimum length checks enforced
    - Duplicates removed by URL
    - Dates standardized
6. Outputs:
    - RSS content → CSV
    - AlpacaEval → JSON

The workflow runs locally and is reproducible.

Part 2 – Data Documentation

Final Schemas (Madison Agent Data Inventory Markdown)

Normalized Content Feed (CSV)

Fields:

`source`
`title`
`url`
`published_at`
`summary`
`raw`

Notes:

- `summary` is unified from RSS description/content fields
- `raw` contains the full original provider payload stored as a single column
- Duplicate URLs removed
- Records without summaries dropped
- Dates standardized

AlpacaEval Dataset (JSON)

Fields:

`record_id`
`record_type`
`source`
`prompt`
`response`
`model_used`
`category`
`dataset_origin`
`response_length`
`quality_score`
`instruction_following_score`
`coherence_score`
`helpfulness_score`

`collected_at`
`quality_flag`

Notes:

- Nested structures flattened
- Empty responses removed
- Prompts < 20 chars dropped
- Responses < 50 chars dropped
- Category inferred heuristically
- Scores included as provided placeholders
- Dataset diversity enforced

Data Validation & Cleaning

RSS Feeds

- Removed rows with empty summaries
- Deduplicated by URL
- Unified summary fields
- Standardized publication dates
- Preserved raw provider payload in single column
- Filtered for relevance
- Final count after validation: 110 records

AlpacaEval

- Removed records with empty prompts or responses
- Enforced minimum prompt length (20 chars)
- Enforced minimum response length (50 chars)
- Flattened nested structures
- Ensured dataset diversity
- Final count: 80 records

Statistics

- Total records: 190
- Clean records: 190 (100% quality rate)
- Sources: 3
- Output formats: CSV + JSON

How This Dataset Supports Madison Agents

The Madison Agent evaluates LLM behavior using two complementary data types:

Context Data (RSS Feeds)

ArXiv and Smol AI provide continuously updated, real-world prompts and topics representing:

- Current research trends
- Deployment challenges
- Emerging model capabilities
  These feeds supply realistic evaluation scenarios.

Ground Truth Data (AlpacaEval)

AlpacaEval provides high-quality instruction–response examples that act as behavioral baselines. Madison compares live model outputs against these reference responses to assess:

- Instruction following
- Coherence
- Helpfulness
- Overall quality

Together, these datasets allow Madison to evaluate models against both real-world content and curated ground truth.

Storage Format

Data is stored separately due to structural differences:

- RSS content exported as CSV for easy inspection and filtering
- AlpacaEval exported as JSON to preserve evaluation metadata

Files produced:

- `content_feed.csv`
- `Alpaca_eval.json`

  Both are reusable inputs for Assignment 4.

Part 3 – Data Quality

Quality Metrics

- Completeness: 100%
- Required fields present in all records
- Duplicate URLs removed
- Dates standardized
- Empty responses removed
- Minimum length enforced

Validation Checks

- RSS summaries present
- Alpaca prompts/responses present
- Response length verified
- Dataset diversity maintained

Setup Guide

Requirements
Node.js installed on your system.
n8n installed locally by running `npm install -g n8n`, then starting it with `n8n start`.

How to Run the Workflow

1. Launch n8n locally in your browser at http://localhost:5678.
2. Import Ravi_Aravind_A3_Workflow.json into n8n.
3. Click “Execute Workflow” to start the pipeline.
4. Wait for all nodes to complete successfully.
5. Once execution finishes, the exported files will appear in the configured output directory:
   – content_feed.csv
   – alpaca_eval.json

Tested locally on n8n v1.64.3 (installed via Homebrew)

Credentials

No API keys are required. All data sources used in this project are Tier-1 public RSS feeds and public HuggingFace datasets.

Common Errors & Fixes

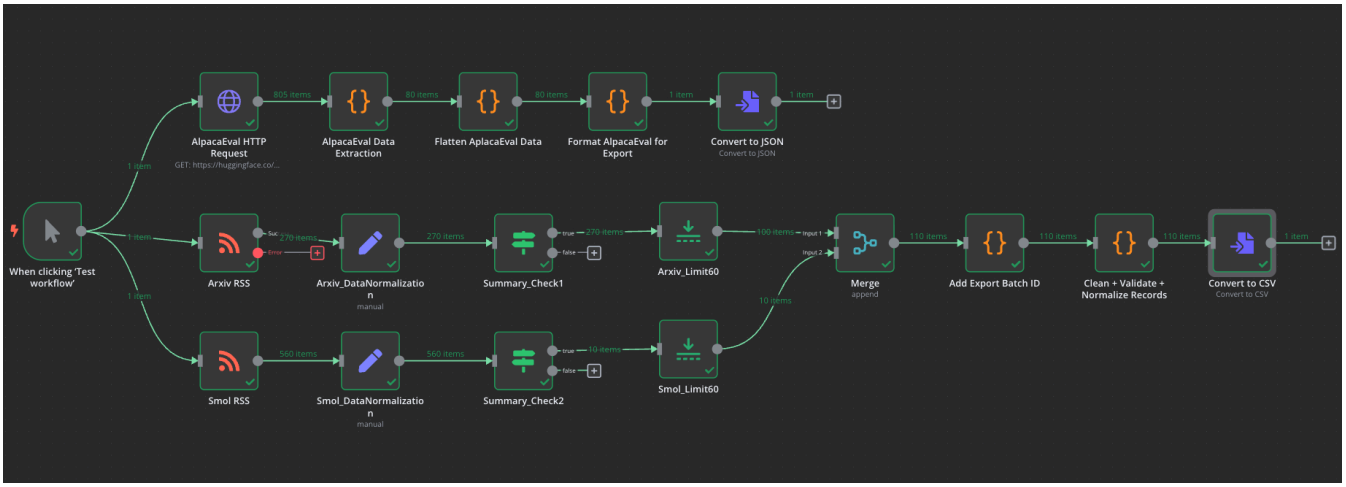
1. RSS rate limiting is avoided by running the workflow once and saving outputs locally instead of repeatedly re-fetching data.
2. XML parsing issues are handled using HTTP status checks and XML conversion nodes before normalization.
3. Null or short summaries are filtered out in the validation code node.
4. Duplicate URLs are removed during the normalization step to ensure each record is unique.

Error Handling

The pipeline includes defensive validation to prevent malformed inputs from entering exports. HTTP status checks prevent XML parsing failures, summary validation nodes remove empty content, and URL-based deduplication avoids duplicate records. These safeguards ensure the workflow completes reliably even when upstream feeds change structure.

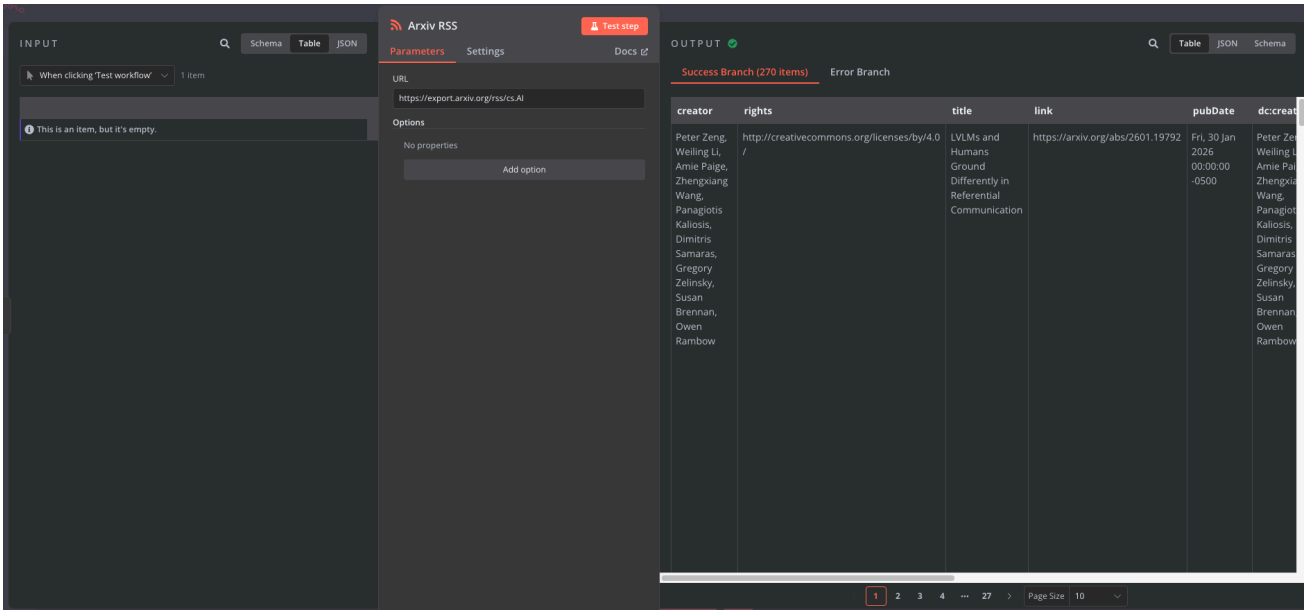
Demo Documentation/ Walkthrough

1. Full workflow overview

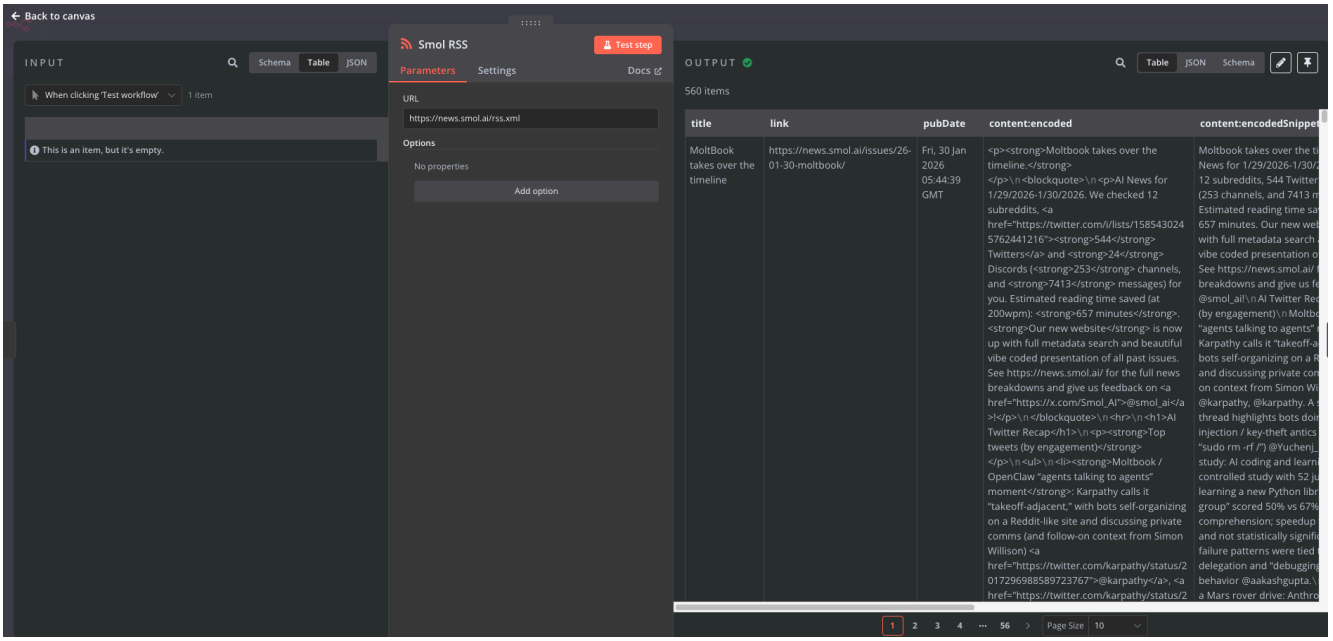


This screenshot shows the complete n8n workflow. Three parallel data pipelines ingest ArXiv research RSS, Smol AI news RSS, and AlpacaEval ground truth from HuggingFace. Each branch applies normalization, validation, and filtering before merging outputs and exporting structured CSV and JSON files. This architecture enables collecting real-world context and curated evaluation data in a single reproducible pipeline.

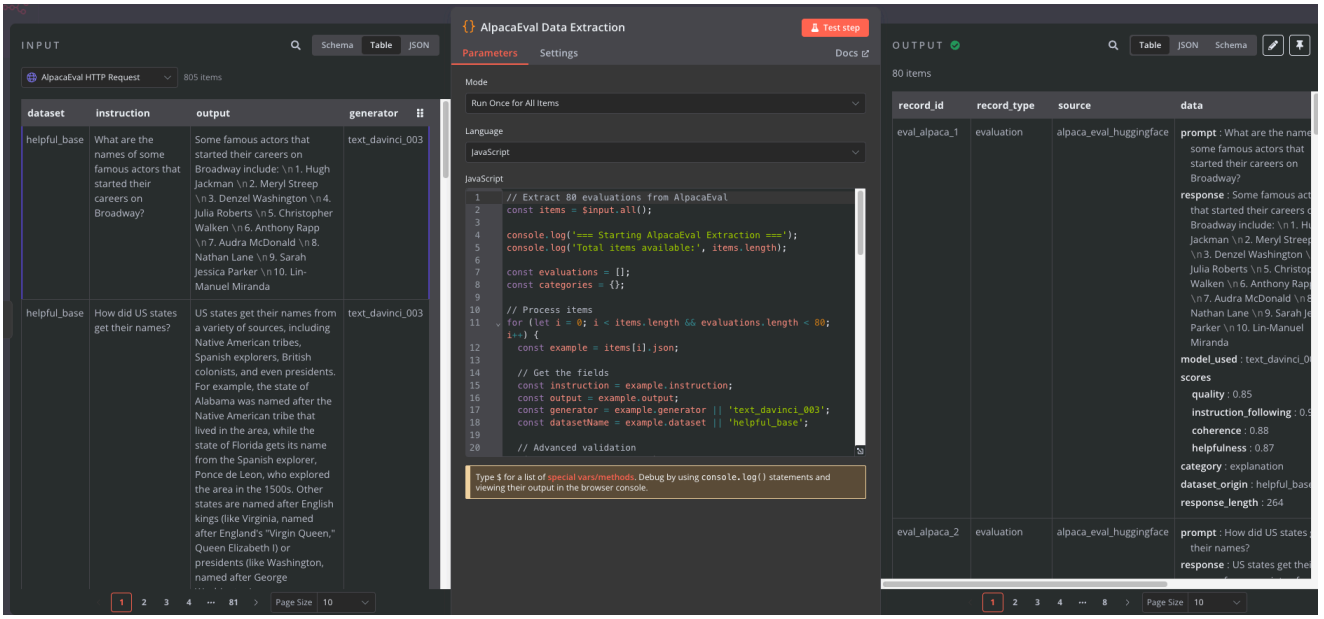
2. RSS nodes



a) ArXiv RSS Read Node: *This node pulls the latest Computer Science AI research articles from ArXiv using an RSS feed. It retrieves titles, abstracts, URLs, and publication dates, forming the primary academic context dataset used for evaluating model knowledge of current research trends.*

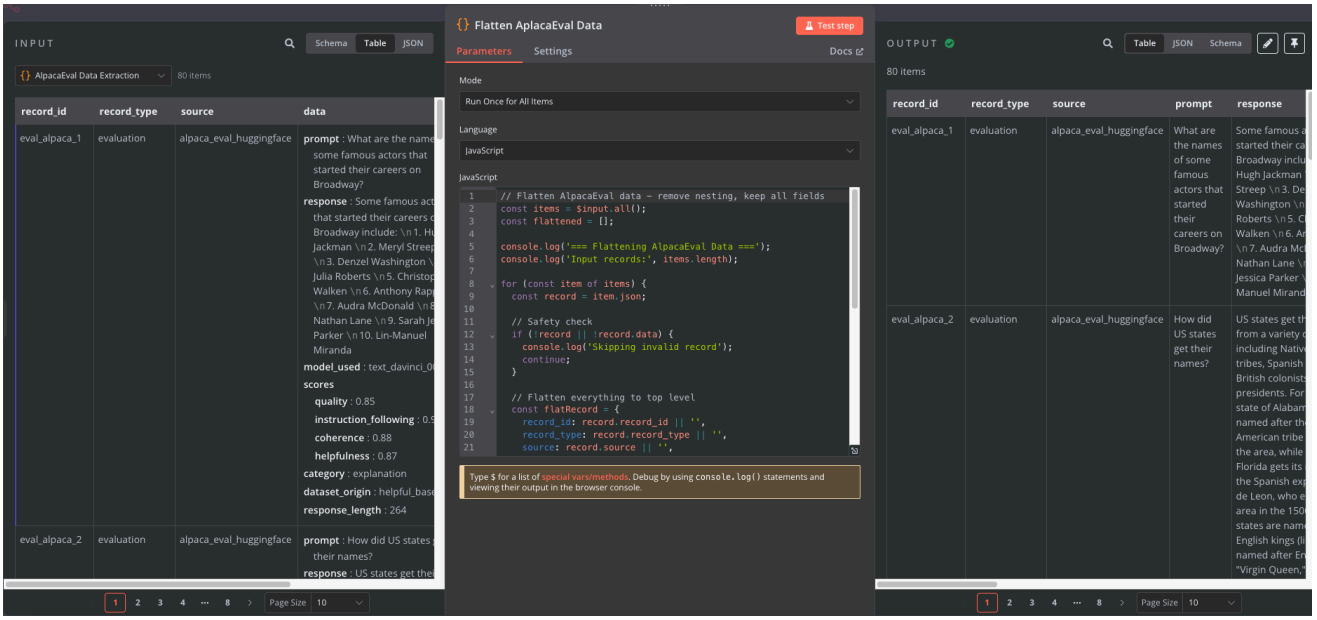


b) Smol RSS Read Node: *This node ingests AI industry news from Smol AI's RSS feed. These records provide practical deployment context and real-world AI developments, complementing academic research from ArXiv.*

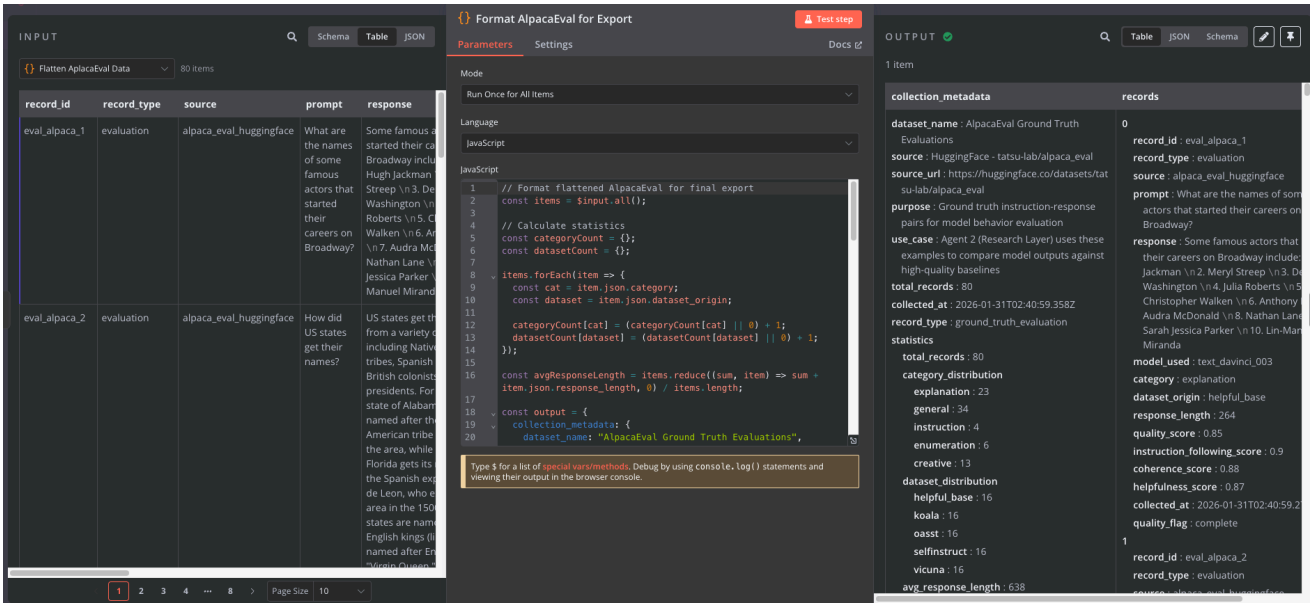


c) AlpacaEval Extraction Node: *This Code node processes raw AlpacaEval records from HuggingFace. It filters incomplete examples, enforces minimum prompt/response lengths, categorizes instructions, and selects a diverse subset of 80 evaluation pairs. These examples act as ground truth references for later model comparison.*

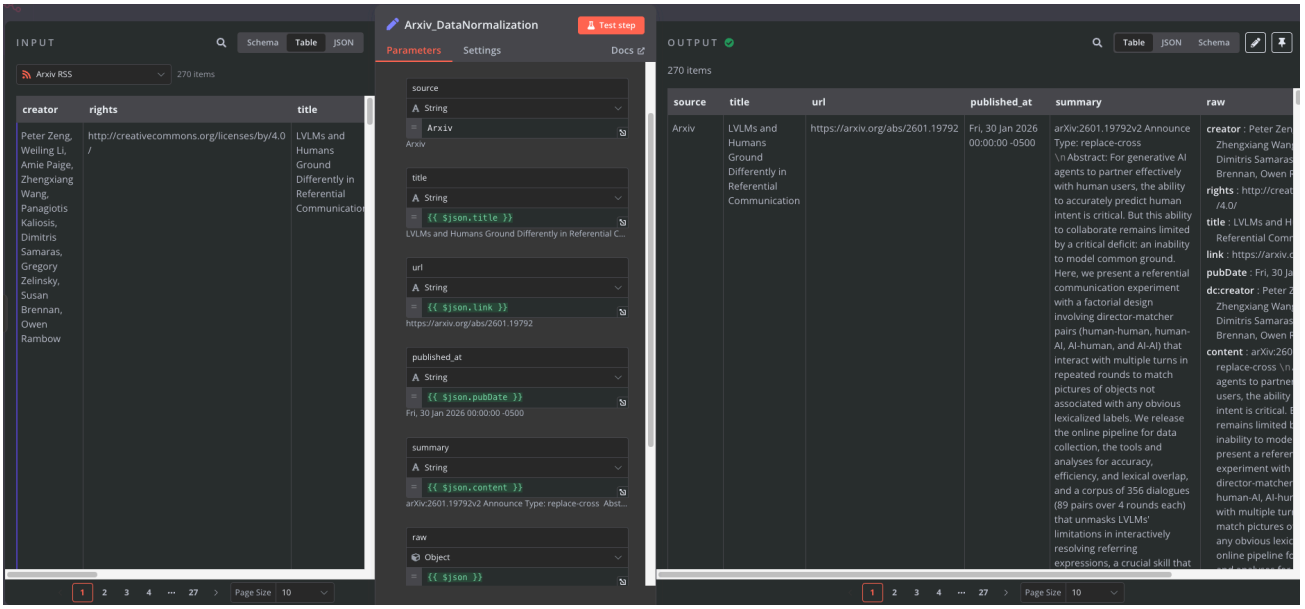
3. Cleaning nodes



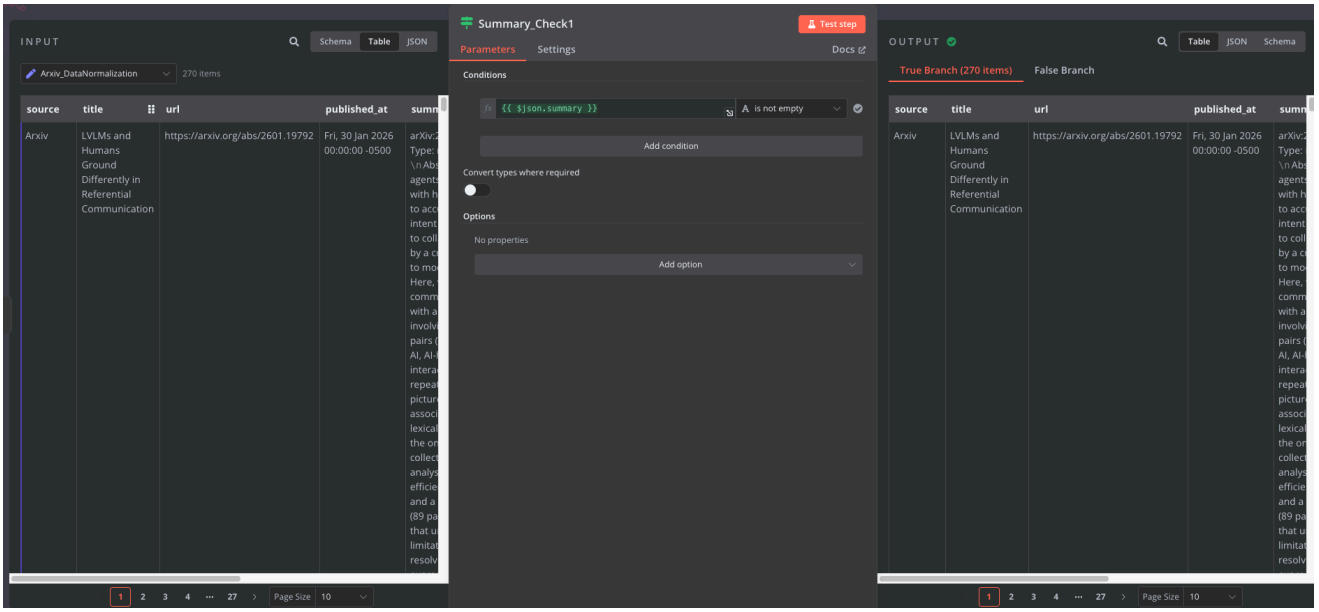
a) Flatten AlpacaEval Data Node: *This node flattens nested AlpacaEval structures into a tabular schema, bringing prompts, responses, scores, and metadata to the top level. This makes the dataset easier to export, validate, and reuse in downstream evaluation agents.*



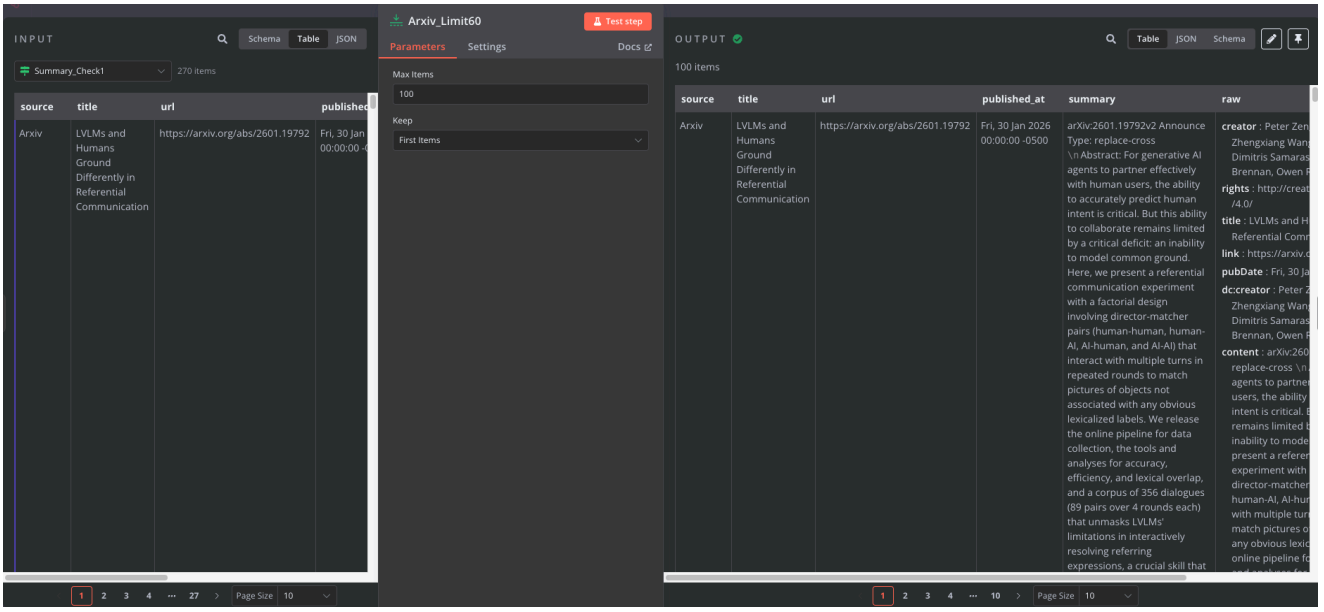
b) Format Alpaca Eval for Export Node: *This node packages AlpacaEval records into a final JSON structure with collection metadata, quality statistics, schema definitions, and validation checks. The output becomes a reusable ground truth evaluation dataset.*



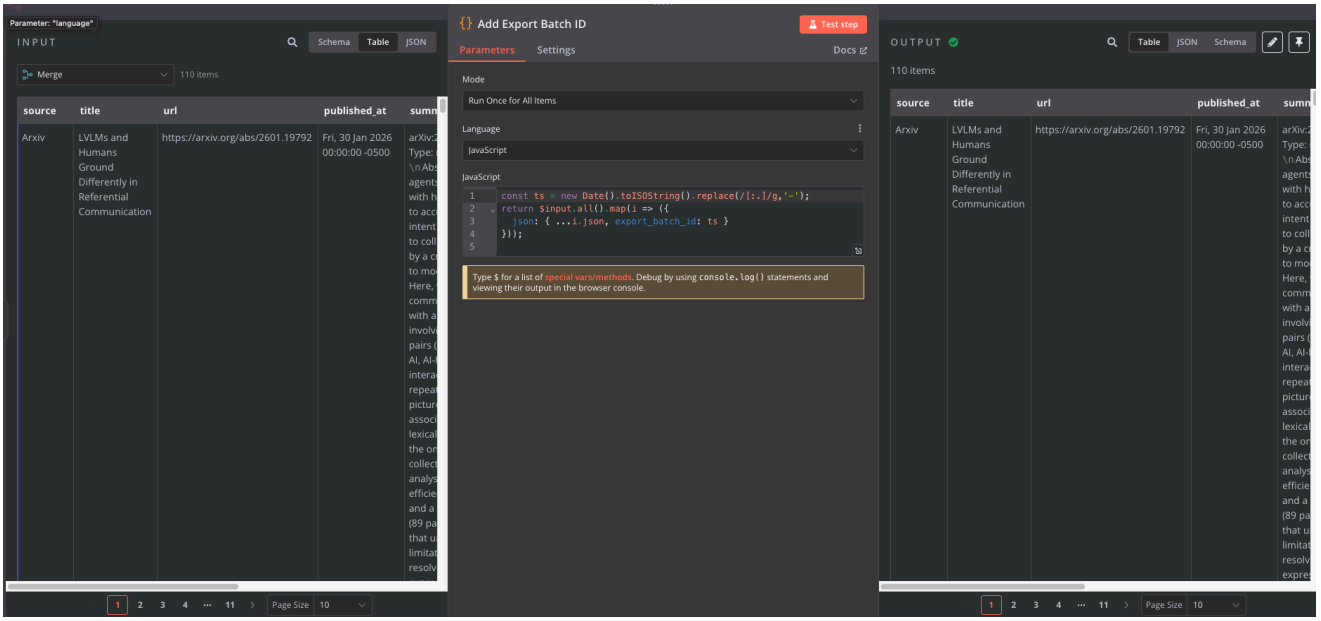
c) ArXiv / Smol Data Normalization Nodes: *These nodes normalize RSS fields into a unified schema (source, title, url, published_at, summary, raw). They standardize date formats, unify summary fields, and prepare both feeds for merging.*



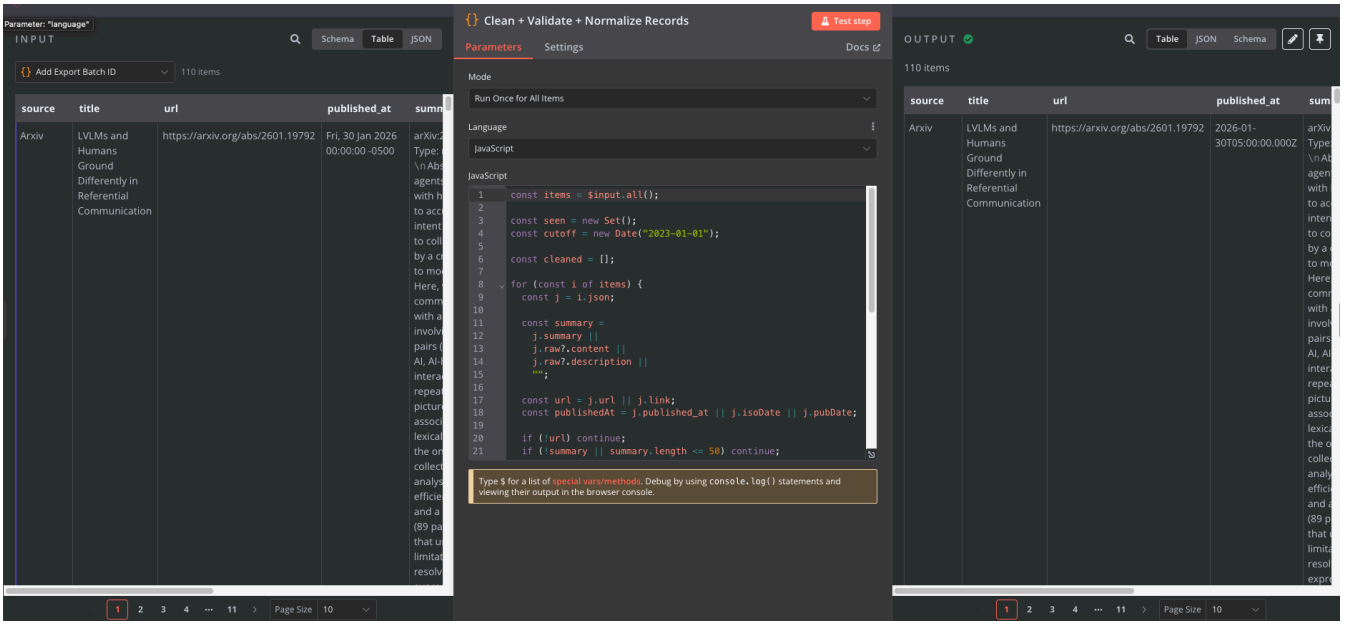
d) Summary Validation Nodes: *These nodes remove records with empty or insufficient summaries, ensuring only meaningful content enters the final dataset. This enforces quality before limiting record counts.*



e) Limit Nodes: *These nodes cap each RSS source to 60 clean records, enforcing quality-over-quantity and maintaining a balanced dataset size.*



f) Add Export Batch ID Node: *This node attaches a unique timestamp batch ID to each record, enabling traceability of exports and preventing confusion between multiple pipeline runs.*

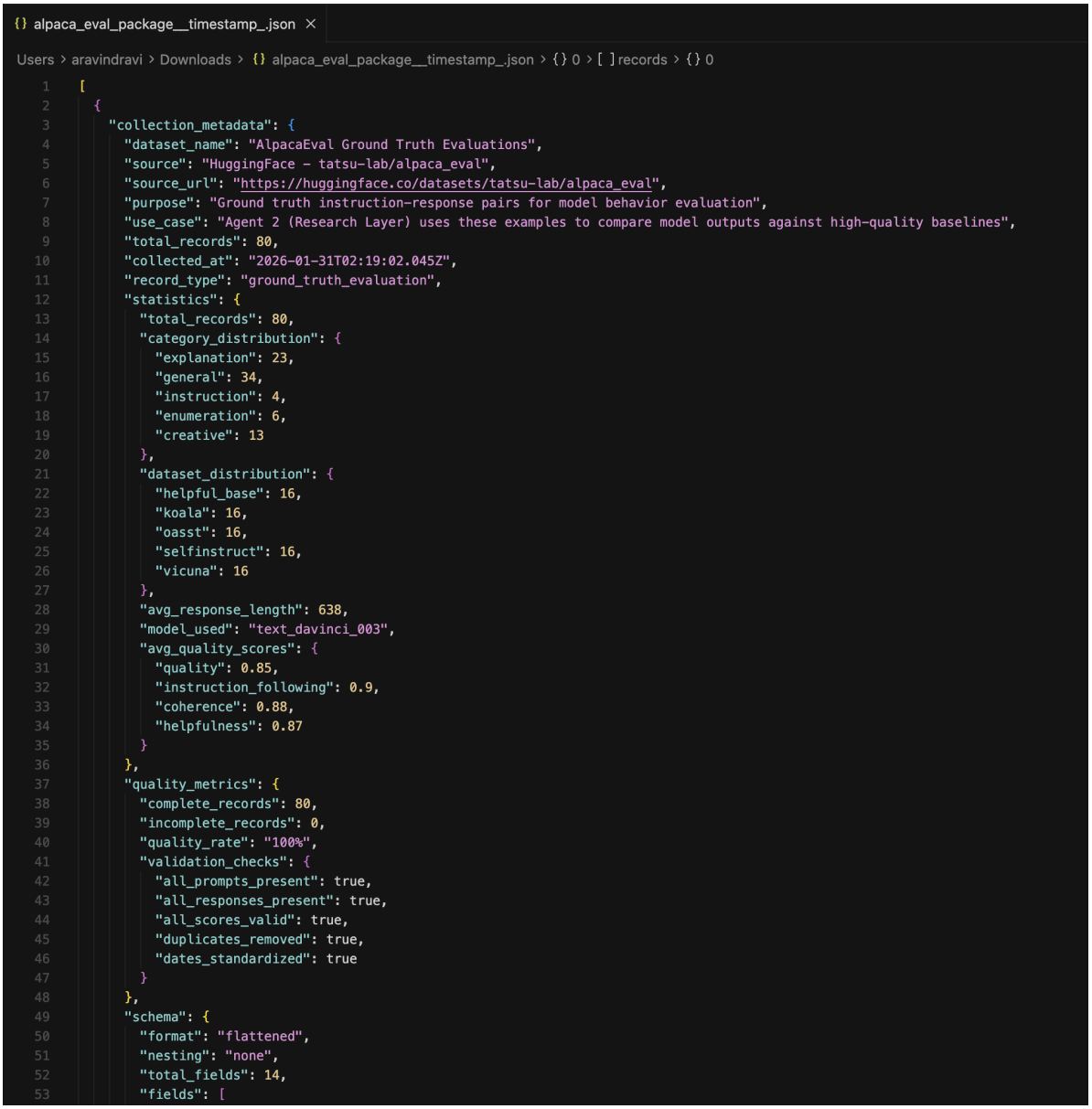


g) Clean + Validate + Normalize Records Node: *This node performs final validation: removes duplicates by URL, filters outdated articles, standardizes publication dates to ISO-8601, consolidates raw provider payloads into a single column, and ensures summaries meet minimum length requirements.*

- **Flow Execution** (Screen Recording)

- **Final CSV and JSON outputs**

Arxiv & Smol RSS Feed CSV



AlpacaEval Dataset JSON

- Replication Notes: A peer can replicate this pipeline by importing the workflow JSON, installing n8n locally, and executing once. No credentials are required.

Final Output

The final step exports two datasets:

- `content_feed.csv` containing normalized RSS research and news
- `alpaca_eval.json` containing flattened ground truth evaluations
  These files are saved locally and reused in Assignment 4 without re-running APIs.

Both datasets are validated and ready for Assignment 4 agent integration.
