Technical Documentation for Processing Pipeline

1 Overview

The objective of this project is to develop a quantitative, data-driven framework for assessing disruption risk in the global lithium supply chain. The core aim is to use high-frequency time-series indicators of disruptive events that can then be used to identify historical trends, estimate disruption likelihoods, and support a broader, empirically grounded criticality assessment.

Lithium supply chains are exposed to a wide range of risks, spanning both physical hazards (such as earthquakes, floods, droughts, and extreme weather) and social, political, and institutional disruptions (such as labour strikes, protests, regulatory interventions, trade restrictions, and geopolitical tensions). Both classes of disruption are economically significant: physical events can damage infrastructure and halt production, while social and political events can close ports, suspend mining licences, or block cross-border trade. A robust criticality framework therefore requires consistent, quantitative information on both types of risk.

For physical disruptions, there is an increasing availability of high-quality data, such as weather observations, satellite-based hazard products, and geological records. Even so, turning these raw hazard measurements into useful indicators that can then be used to predict disruption likelihood (and therefore later real economic disruption effects) is not straightforward. This work can then be used in conjuction with linking physical events to specific mines, processing facilities, transport routes, and ports, which is often non-trivial and context-dependent.

The challenge is even greater for social and political disruptions. Events such as labour strikes, protests, regulatory interventions, or trade restrictions tend to occur irregularly, vary widely across locations, and are rarely captured in structured, high-frequency datasets. Despite this, they often play a decisive role in shaping actual supply-chain outcomes.

The processing pipeline developed in this project is designed to address both challenges by transforming global news reporting into a continuous stream of disruption signals. News media provide near-real-time coverage of a wide range of events. By systematically collecting and analysing this information at scale, the pipeline creates a unified empirical lens through which both physical and social disruptions can be observed and compared.

At a high level, the pipeline operates as follows:

1. **GDELT ingestion.** Daily GDELT event context files provide broad international coverage of news reporting, generating large volumes of candidate URLs that may describe disruption-related events.
2. **URL deduplication and filtering.** Duplicate links and clearly irrelevant material (e.g. sports, entertainment, lifestyle content) are removed using conservative URL-based rules, reducing noise before any costly downstream processing.
3. **Metadata enrichment.** Each remaining URL is fetched and its HTML title and meta description are extracted, producing a compact summary of the article's content.
4. **Machine-learning classification.** A supervised text classifier estimates the probability that each article describes a supply-chain–relevant disruption.
5. **Web scraping and LLM-based extraction.** URLs identified as relevant are scraped in full and passed through a large language model to extract structured information about the disruption event.

6. **Ground-truth validation.** Extracted disruption events are validated against established external datasets (e.g. ACLED, ReliefWeb, DFO, EM-DAT), providing an independent check on event occurrence, timing, and location, and helping to identify systematic gaps in news-derived coverage.
7. **Indicator suitability assessment.** Candidate indicators are evaluated using a composite suitability score that summarises their relevance, coverage, temporal resolution, and alignment with observed disruption events.
8. **Missing-indicator population.** For disruption types and indicators that meet suitability requirements, missing values are populated using appropriate interpolation, aggregation, or data-fusion methods to produce a coherent, temporally continuous indicator set.
9. **Disruption likelihood estimation.** Validated event streams are combined with continuously updated indicator data to compute time-varying likelihood estimates for different classes of disruption.

By converting unstructured global news into structured time-series indicators and integrating them with external datasets, the pipeline provides a common empirical basis for analysing both physical and social sources of supply-chain risk. These indicators constitute the likelihood component of a broader criticality framework, which, when combined with measures of vulnerability, enables the quantification of overall disruption risk.

2 Architectural Design & Decision Rationale

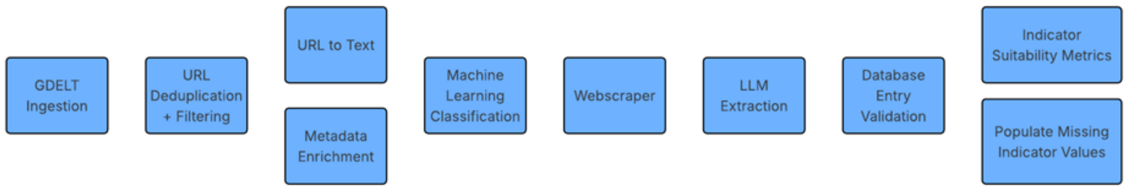
GDELT Ingestion
URL Deduplication + Filtering
URL to Text
Metadata Enrichment
Machine Learning Classification
Webscraper
LLM Extraction
Database Entry Validation
Indicator Suitability Metrics
Populate Missing Indicator Values

Figure 1: Flowchart of processing pipeline.

2.1 GDELT Ingestion

The objective of this project is to estimate the likelihood of disruption events in critical mineral supply chains. Achieving this requires two complementary elements:

1. a record of when and where disruption-related events occurred, and
2. a record of the economic, political, and environmental conditions present at the time those events occurred, so that their relationship to disruption risk can be examined empirically.

Critical mineral supply chains are geographically concentrated and politically exposed. A small number of countries and regions account for a disproportionate share of global extraction, processing, and transport capacity. As a result, relatively localised events (such as mine shutdowns, labour disputes, regulatory interventions, or environmental incidents) can propagate into wider supply disruptions. Many of these events are not systematically captured in structured industrial or trade datasets, particularly when they occur in remote regions or involve smaller operators.

In practice, news reporting provides the earliest and most consistent signal of such disruptions. The Global Database of Events, Language and Tone (GDELT) aggregates news coverage from thousands of sources worldwide and records articles with timestamps, location information, and source URLs. This makes it possible to identify when disruption-related events first enter the public information space and to align them with the economic, political, and environmental conditions that were present when they occurred.

Rather than treating GDELT as a labelled dataset, the pipeline uses it as a global discovery layer. Each day, newly reported events are surfaced as candidate articles that may describe supply-chain disruptions. This high-coverage approach is particularly important for likelihood estimation: rare, local, or politically sensitive incidents must be captured consistently in order to avoid biasing the analysis toward only large or well-documented events.

GDELT publishes event data in high-frequency 15-minute files. While this granularity is useful for real-time monitoring, it is not well suited to large-scale statistical analysis or integration with other data sources. For this reason, the pipeline aggregates 15-minute exports into daily datasets. Working at a daily resolution substantially simplifies downstream processing, filtering, and modelling, and aligns naturally with most economic, political, and environmental indicators, which are typically reported at daily or lower frequencies. This aggregation step therefore improves analytical tractability without materially affecting the ability to identify disruption events.

Overall, the GDELT ingestion stage provides a scalable, global, and empirically grounded foundation for identifying disruption events and linking them to the conditions under which they occur, which is essential for building a defensible criticality assessment.

2.1.1 Component Breakdown

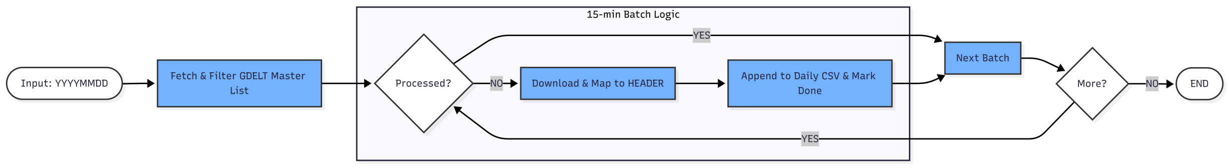

Figure 2: Flowchart of processing pipeline

The GDELT ingestion module begins by taking a user-specified date and querying the GDELT master file list to identify all 15-minute export files published on that day. Each export is checked against a persistent state record to determine whether it has already been processed, ensuring that reruns do not duplicate data. For unprocessed files, a small subset of event and geographic fields, together with the source URL, is extracted and appended to a daily CSV file corresponding to the target date. By aggregating high-frequency exports into a single daily file, the pipeline produces a manageable and analytically convenient representation of global news activity while retaining precise timestamps and source URLs. State tracking and incremental writes allow partially processed days to be resumed safely, providing fault tolerance and reproducibility, while keeping the ingestion stage lightweight so that more expensive filtering and classification steps can be applied downstream.

2.2 URL Deduplication and Filtering

Once the data have been pulled from GDELT, the pipeline is left with a very large list of URLs. Most of these are not actually useful for supply-chain analysis, and many of them are repeated. At the same time, the next stages in the pipeline, such as downloading web pages, extracting metadata, generating embeddings, and running classifiers, are relatively slow and expensive compared to simple string checks.

Because of this, it makes sense to remove as much obvious noise as possible as early as we can. If we can cheaply discard links that are clearly irrelevant or duplicated, we avoid wasting time, bandwidth, and compute later on. This lets the heavier parts of the pipeline focus on the much smaller set of URLs that might actually describe disruption events.

Deduplication is particularly important. GDELT often lists the same article multiple times in a single day, either because the same URL appears in multiple event records or because of how stories are syndicated. If these duplicates are not removed, they can slow down processing and distort counts in later analysis. Removing duplicates early ensures that each article is only processed once per day, while still keeping separate coverage from different news outlets.

It is important to note that deduplication is applied only at the level of identical URLs, not at the level of underlying events. Different news outlets often publish separate articles about the same real-world disruption, and these are intentionally retained. Preserving multiple independent sources allows the pipeline to capture complementary details and perspectives, since key information about a disruption may appear in only a small subset of reports.

2.2.1 Component Breakdown

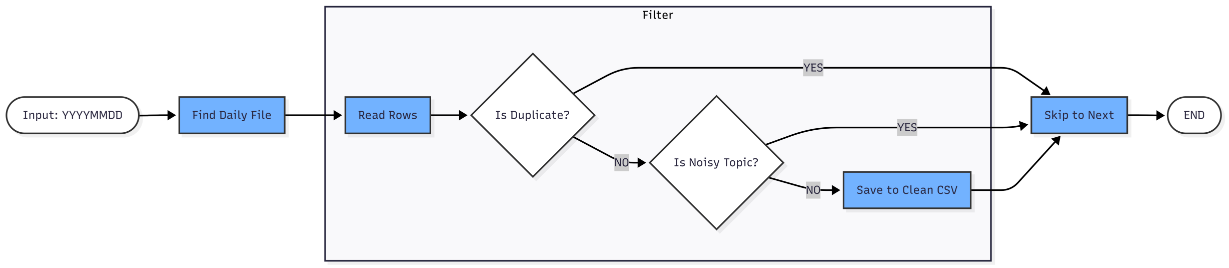

Figure 3: Flowchart of processing pipeline.

The deduplication and filtering module operates at a daily level, taking a user specified date and locating the corresponding GDELT event-context CSV. The file is processed sequentially, first performing URL deduplication by tracking previously seen URLs in memory and ensuring that each unique article is retained only once per day. This prevents duplicated downstream processing while preserving coverage across different news outlets. The deduplicated URLs are then passed through a fast, URL-only relevance filter that removes content which is clearly unrelated to supply-chain disruptions.

This filtering step excludes non-article resources (such as images or PDFs), category or tag pages, and URLs whose structure indicates topics such as sport, entertainment, lifestyle, or other consistently irrelevant content. To make this process robust, URLs are normalised by decoding encoded characters and combining the domain, path, and query string before matching. The list of exclusion keywords is informed by an exploratory analysis that counts the most frequent words appearing in URLs across multiple days of data, allowing common sources of noise to be identified empirically and filtered efficiently. URLs that pass both deduplication and relevance filtering are written to a cleaned daily file, which is then forwarded to metadata enrichment and machine-learning stages where more computationally expensive processing is applied.

2.3 Metadata Enrichment

After URL filtering, the pipeline still has access only to web addresses, rather than the content of the news articles themselves. To decide whether a story describes a real supply-chain disruption, we need text that reflects what the article is actually about.

There are two main ways to obtain this. The first is to convert the URL itself into text by breaking the path into words. This is fast and requires no web requests, but it is also very limited: URLs often contain truncated, ambiguous, or marketing-driven strings that do not reliably describe the underlying article. As a result, URL-based text provides only a weak and noisy signal.

The second approach is to retrieve metadata from the web page itself, in particular the page title and meta description. These fields are designed to summarise the content of the article in a short, human-readable form. They usually contain the key entities, locations, and actions being reported, making them far more informative for classification than raw URLs.

Although fetching this metadata is slower than using URLs alone, it is still much cheaper than downloading and parsing full article bodies. It therefore provides a useful middle ground: significantly richer semantic information, but at a fraction of the cost of full-page scraping. By enriching each URL with its title and description, the pipeline creates a compact but meaningful text representation that can be used for embedding, classification, and ranking, while keeping the system fast enough to run on large daily data volumes.

When we first developed this module, we began by sending only one request at a time. This serial approach meant that retrieving metadata for a single day could take upwards of 8 hours, as the system sat idle while waiting for each individual server to respond. Upon identifying that the longest part of the process was this ”network wait time,” we re-engineered the pipeline to use multiple threads. By increasing iterations and allowing up to 20 concurrent requests, we moved from a sequential bottleneck to a parallel architecture. This shift reduced processing time significantly (shrinking an 8-hour task into less than an hour) and turned the enrichment stage into a high-performance component capable of scaling with the GDELT data stream.

2.3.1 Component Breakdown

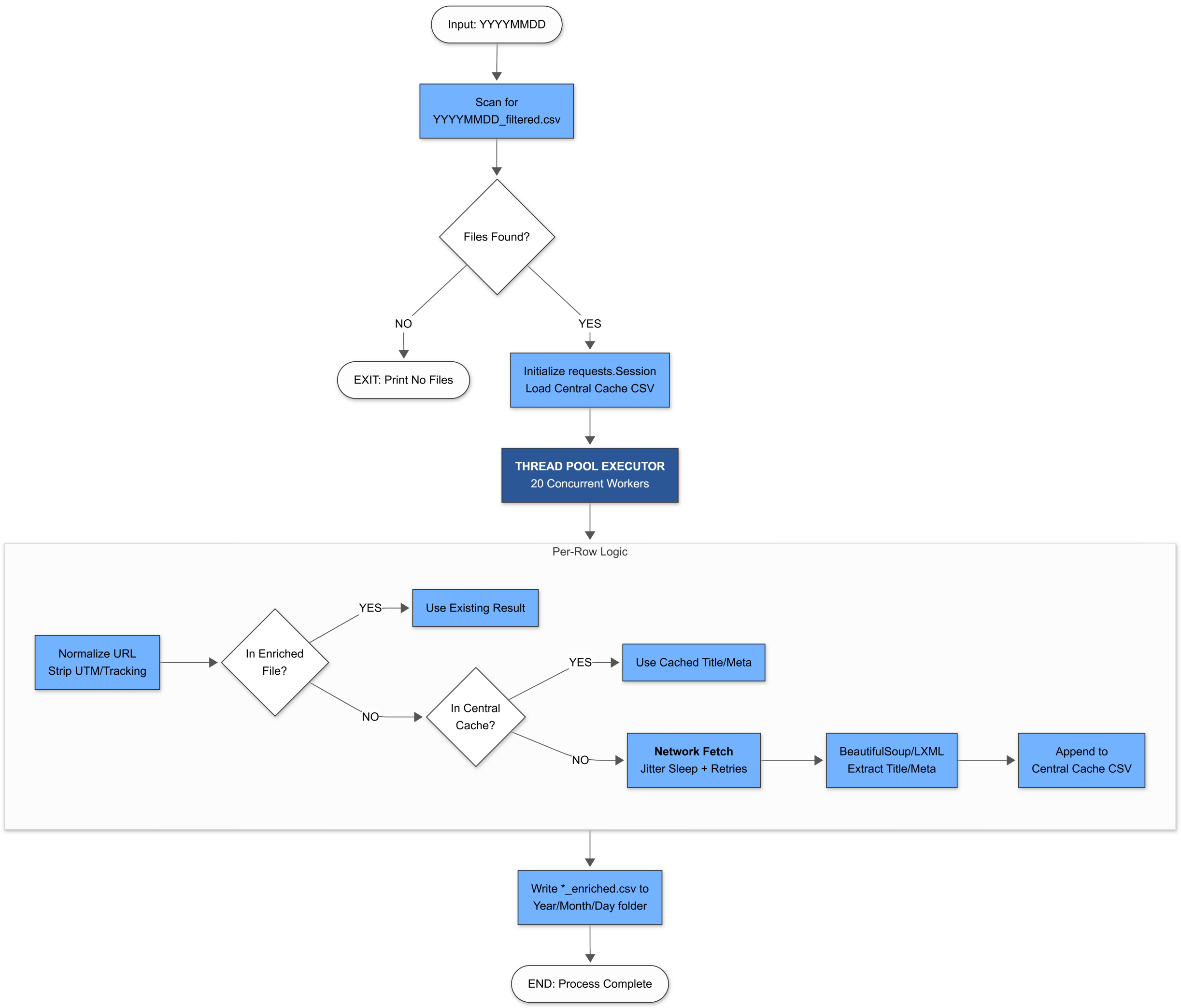
Input: YYYYMMDD
Scan for YYYYMMDD_filtered.csv
Files Found?
NO
YES
EXIT: Print No Files
Initialize requests.Session Load Central Cache CSV
THREAD POOL EXECUTOR 20 Concurrent Workers
Per-Row Logic
Normalize URL Strip UTM/Tracking
In Enriched File?
YES
Use Existing Result
NO
In Central Cache?
YES
Use Cached Title/Meta
NO
Network Fetch Jitter Sleep + Retries
BeautifulSoup/LXML Extract Title/Meta
Append to Central Cache CSV
Write *_enriched.csv to Year/Month/Day folder
END: Process Complete

Figure 4: Flowchart of processing pipeline.

The process begins with data discovery, where the script scans the directory structure to track down the filtered CSV files for the day. Once these files are located, the system sets up a shared network session to keep connections efficient and loads a central state cache from the storage folder. To handle the slow nature of waiting for websites to respond, the engine uses a multithreaded executor that manages 20 workers at once. This allows the system to juggle multiple URLs in parallel rather than getting stuck on a single slow-loading page.

Before reaching out to the web, each worker puts the URL through a three-step efficiency check. First, it cleans the link by stripping out messy tracking parameters like UTM codes to make sure we aren't fetching the same page twice under different names. Second, it performs a resume check by looking at the output file to see if that specific URL was already finished in a previous run. Third, it cross-references the central cache to see if we've ever seen this URL before, even on a different day. These steps ensure that we only spend time and bandwidth on a network fetch if the data truly isn't available locally.

When a fetch is necessary, the worker uses a "jittered" sleep (a tiny, randomized pause) to avoid hitting the same news site with too many requests at the exact same millisecond. It then uses a fast parser to grab the page title and meta description, which provides the context needed for our models. As these details are captured, they are added to the central cache in real-time so other workers can benefit from them immediately. Finally, the new metadata is merged back into the original records and saved to the final enriched file, concluding with a summary of processing throughput and performance gains.

2.4 Labelling disruptions

Once titles and meta-descriptions have been extracted for each article, the next step is to create supervised labels that indicate which disruption type (if any) the article describes. In practice, this is done by manually reviewing the article summary text and assigning labels from a fixed set of disruption categories (e.g. earthquakes, floods, protests, labour strikes, mine accidents, and trade restrictions).

For each article, we read the title and meta description and assign a value of 1 for a disruption category only when the article is genuinely describing that event type in the real world. This semantic check is important: many disruption-related keywords appear in non-disruption contexts (e.g. "a flood of criticism"), and these cases are deliberately labelled as non-disruptive so that the model learns the difference between literal event reporting and figurative or unrelated usage.

In addition to the per-category labels, we maintain an overall binary disruption label that indicates whether any disruption type is present. This allows the pipeline to first train and deploy a general "disruption vs. not" classifier with limited labelled data. As the labelled dataset grows, the same framework can be extended to train specialist expert models for individual disruption types, improving precision and interpretability at the category level.

If the title and meta description are missing or uninformative, the pipeline falls back to using the URL by converting the URL path into readable text and applying the same manual labelling process. This avoids dropping articles purely because metadata could not be extracted and preserves coverage for downstream training.

2.5 Machine Learning Disruption Classification

After URL filtering and metadata enrichment, the pipeline still contains hundreds to thousands of news articles per day. These articles describe events in highly variable language and formats, making simple keyword or rule-based methods unreliable. The classification problem is therefore not about detecting specific words, but about identifying whether a piece of text describes a real-world disruption event.

To address this, the pipeline uses a two-stage learning approach: semantic embeddings followed by a linear classifier.

First, each article is mapped into a dense numerical vector using a pre-trained sentence embedding model. This step transforms short, noisy text (titles, meta descriptions, or URL slugs) into a representation that captures meaning rather than surface wording. Articles describing similar events, for example, different reports of a mine accident or a transport strike, are placed close together in this embedding space even when the phrasing differs. This allows the system to generalise beyond simple keyword matches and recognise disruptions across different countries, outlets, and writing styles.

On top of these embeddings, a linear support vector machine is trained using manually labelled examples of disruption and non-disruption articles. The SVM is a good fit for this setting because it performs well in high-dimensional spaces, requires relatively little training data, and produces a stable decision boundary between the two classes. Unlike deep neural classifiers, it is also easy to retrain and update as new labelled data become available.

The classifier is probability-calibrated so that it outputs a continuous disruption score rather than just a binary decision. This is important operationally: the pipeline is designed to favour high recall, meaning it is preferable to include some irrelevant articles than to miss genuine disruption events. By adjusting the probability threshold, the system can control this trade-off depending on how the downstream analysis is being used.

Overall, this design provides a practical and scalable way to convert a global stream of heterogeneous news text into a structured signal of disruption likelihood, while improving automatically as the labelled dataset grows.

2.5.1 Component breakdown

This stage assigns each enriched news article a probability of being relevant to a supply-chain disruption. It runs on a single day at a time and takes as input the cleaned, day-specific metadata produced by the enrichment stage. Outputs are written to a matching year/month/day folder structure, which makes it easy to rerun, audit, or inspect individual days without affecting the rest of the pipeline.

For each article, a short text representation is constructed. The title and meta description are used as the primary signal, since they are designed to summarise the article's content. If this text is missing or clearly uninformative (for example, cookie notices or access-denied pages), the pipeline falls back to the URL by converting the URL path into readable tokens. This ensures that articles are not dropped simply because metadata extraction failed.

The resulting text is converted into a dense numerical representation using a pre-trained sentence embedding model. Each article is mapped to a fixed-length vector that captures semantic meaning rather than relying on exact keyword matches. Embeddings are normalised to ensure consistent scaling across articles of different lengths and writing styles.

These embeddings are then passed to a calibrated linear support vector machine, which outputs a probability that the article describes a disruption. The classifier is trained with class balancing to account for the fact that disruption-related articles are much rarer than non-disruptive ones. Probability calibration allows the model to return interpretable likelihood scores rather than hard class labels.

A probability threshold is applied to decide which articles are retained for downstream analysis. Alongside a full scored dataset, the pipeline produces ranked outputs containing only the most likely disruption-related URLs, optionally restricted to the top-scoring articles. This keeps later stages focused on a small, high-value subset of news.

The model itself is trained separately using a labelled dataset (as mentioned in the previous section) and saved as a reusable bundle containing the classifier, embedding model identifier, and decision threshold. Daily scoring simply loads this bundle, allowing the classification stage to run quickly and consistently without retraining.

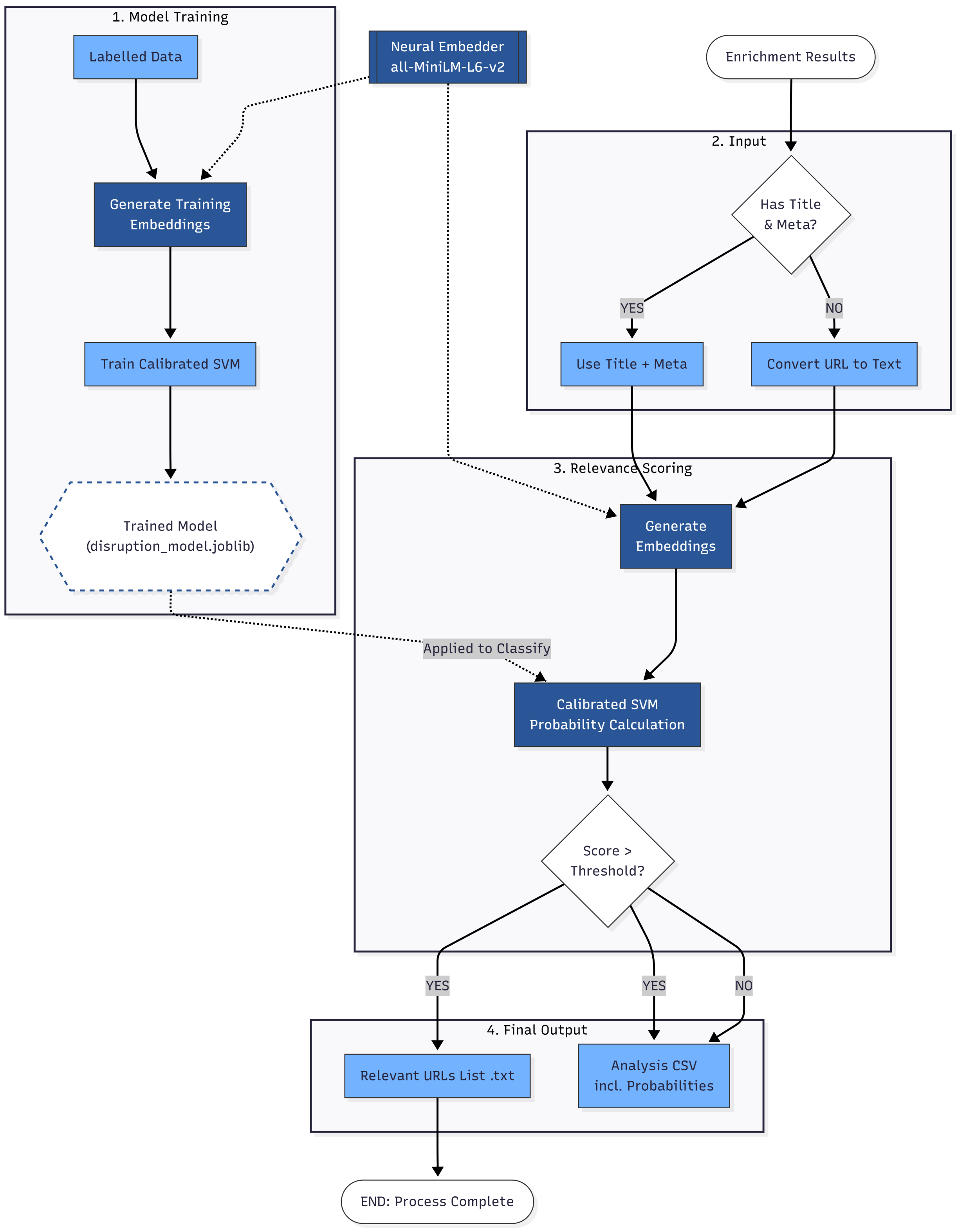

Figure 5: Flowchart of processing pipeline.

2.5.2 Current Classifier Performance

The current disruption classifier was evaluated on a held-out test set of 326 labelled articles (20% of labelled data) using a probability threshold of 0.4. The resulting confusion matrix was

$$\begin{bmatrix} 240 & 16 \\ 13 & 57 \end{bmatrix}$$

This corresponds to 240 true non-disruption articles correctly rejected, 57 true disruption articles correctly identified, 16 false positives, and 13 false negatives.

At this operating point the model achieves:

- Accuracy: 91% (proportion of all articles correctly classified)
- Average precision: 87% (area under the precision–recall curve, measuring ranking quality across thresholds)
- Recall for disruptions: 81% (fraction of true disruption articles correctly identified)
- Precision for disruptions: 78% (fraction of predicted disruption articles that are true disruptions)

The decision threshold was set to 0.4 rather than the default 0.5 in order to prioritise recall over precision. In this application, missing a genuine disruption event is more costly than allowing a small number of irrelevant articles to pass through, since downstream processing and validation can filter noise but missed events permanently degrade the disruption time series. Lowering the threshold shifts the decision boundary to include more borderline cases, reducing false negatives at the cost of a modest increase in false positives.

An error review file is generated automatically, containing all false positives and false negatives. This allows systematic inspection of misclassifications and supports targeted improvements to labelling rules and training data.

As the labelled dataset grows, particularly with more rare and ambiguous cases, both recall and precision are expected to improve. The embedding-plus-SVM architecture is designed to benefit directly from additional training data without requiring structural changes to the model, allowing performance to increase steadily over time.

2.6 Webscraper

After URL-level filtering, the first major content-dependent stage in the pipeline is the web scraper. Its purpose is to reliably fetch and normalise article text from a heterogeneous set of news websites, while minimising downstream noise and failure modes.

News websites vary substantially in HTML structure, rendering strategy, and content layout, meaning that no single extraction method is sufficient to achieve robust coverage. The scraper therefore adopts a defensive, multi-method extraction strategy that prioritises reliability and graceful degradation over raw speed.

For each URL, the page HTML is retrieved and passed to `Trafilatura`, which serves as the primary extraction method. Trafilatura is designed for large-scale news scraping and aggressively removes non-content elements such as navigation menus, advertisements, sidebars, and embedded scripts. It also extracts structured metadata (e.g. the article title), improving consistency across sources and reducing the need for custom parsing logic.

Some sites, particularly those relying on dynamic rendering or non-standard markup, are not reliably handled by Trafilatura. To mitigate systematic extraction failures, a fallback using

`Newspaper3k` is employed. Newspaper3k uses a different parsing strategy and often succeeds where Trafilatura fails, significantly improving overall scrape success rates without introducing site-specific logic.

A lightweight normalisation step is then applied to the extracted text, including removal of non-breaking spaces, collapsing excessive whitespace, and standardisation of title and body formatting. This ensures that all articles conform to a consistent textual representation suitable for downstream machine-learning and event-extraction stages.

The scraper outputs a minimal structured dictionary containing the source URL, cleaned article title, and cleaned article body text. This simple interface decouples scraping from downstream components and allows extracted content to be reused or reprocessed without re-fetching the original pages.

2.6.1 Component Breakdown

Filtered URLs are first fetched via standard HTTP requests. The retrieved HTML is passed to Trafilatura for primary content extraction. If Trafilatura succeeds, the extracted article text is retained; if it fails, a fallback extraction using Newspaper3k is applied. Text from either path is then normalised and cleaned to remove formatting artefacts. The final output is a minimal article dictionary containing the source URL, cleaned title, and cleaned article body text.

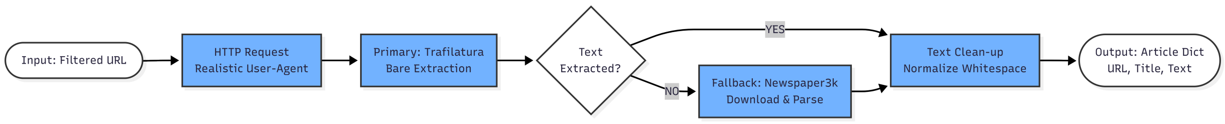

Figure 6: Flowchart of processing pipeline.

2.7 LLM Extraction

Following full-text scraping and cleaning, articles are passed to an event-extraction stage whose purpose is to convert unstructured news text into structured disruption events.

Two design iterations were explored for this component. The initial approach used a hybrid system in which rule-based detectors were applied first to identify candidate signals (e.g. dates, rainfall quantities, strike-related terms, or hazard keywords), after which a local large language model (LLM) was used to fill missing or ambiguous fields. In practice, this approach proved brittle and difficult to scale. Metaphorical language in news reporting frequently triggered false positives (for example, phrases such as “a deluge of criticism” being misclassified as flooding events), and the rule set became increasingly complex and unmaintainable as the number of disruption types expanded. In addition, running both deterministic rules and an LLM on every article introduced significant computational overhead.

The final design therefore adopts a simplified LLM-only extraction approach. Each article is passed directly to the LLM, which performs end-to-end extraction of event attributes in a single step. This design is more robust to linguistic variation, avoids hand-engineered rule maintenance, and scales cleanly as new disruption types and attributes are added.

2.7.1 Operational Characteristics and Scalability

In its current configuration, the extraction pipeline processes on the order of 10^3 news articles per day following upstream filtering. Empirically, approximately 500 articles can be processed for under one US dollar in API costs, implying that daily extraction at current volumes is economically feasible even when operating continuously. This cost profile is dominated by the number of articles passed to the LLM rather than by article length, making early-stage relevance filtering a critical determinant of overall scalability, and is therefore something that can be reduced with better supervised learning early on.

Token usage statistics provide additional insight into computational load. Typical daily extraction runs involve several hundred thousand input tokens and on the order of 10^5 output tokens, corresponding to structured JSON responses containing event attributes and supporting evidence. Importantly, the extraction cost scales approximately linearly with the number of processed articles, enabling predictable cost and time budgeting as the system is extended to longer time horizons or additional commodities. Once the validity and accuracy of the extraction is fully assessed, it would then be possible to include fewer evidence fields, reducing costs further.

This linear scaling behaviour motivates the emphasis placed on upstream relevance classification. Improvements to the supervised classifier applied to GDELT-derived URLs directly reduce the number of articles reaching the LLM stage, yielding proportional reductions in runtime and monetary cost. As a result, investment in classifier refinement, despite requiring manual labelling effort, offers substantial downstream efficiency gains.

2.7.2 Component Breakdown

For each input URL, the scraper is called to obtain cleaned article title and body text. The cleaned content is then sent to the ChatGPT API in a single extraction pass, using a schema-constrained prompt and a strict JSON response format. The returned JSON is parsed defensively, and missing or malformed fields are filled with safe defaults to ensure a consistent output schema even when model responses are imperfect.

Each successfully parsed response is packaged into a standard `ExtractRecord` containing the inferred disruption type, event date, location, duration where available, any explicitly stated indicator values (`extras`), supporting evidence snippets, and a model-derived confidence score. Records are written to both `JSONL` and `CSV` outputs, while failed or malformed extractions are logged separately for inspection and debugging.

Overall, this design allows the extraction stage to operate as a high-throughput, cost-predictable component within the broader pipeline, while remaining flexible enough to accommodate new disruption categories and extraction requirements with minimal architectural change.

2.7.3 Reliability of LLM-Based Event Extraction

To assess the reliability of the LLM-based extraction stage, the outputs were analysed across a representative batch of extracted events spanning multiple disruption types. This analysis focuses on three aspects: type classification behaviour, internal consistency of extracted fields, and the interpretation of confidence scores.

Across the evaluated batch, the extraction stage produced structured events across a diverse set of disruption categories, including floods (69 events), tariffs (39), trade embargoes (38), labour strikes (20), cyclones and hurricanes (28), earthquakes (9), and droughts (7), alongside a residual `unknown` category (276 records). The presence of a large `unknown` class is expected at this stage and reflects a conservative design choice: articles that do not clearly describe a discrete, supply-chain-relevant disruption are retained but explicitly marked as ambiguous rather than being forced into a specific category.

For well-defined physical disruption types, such as floods, earthquakes, and cyclones, the extracted events exhibit a high degree of internal consistency. Event locations are geographically plausible, dates are either explicitly stated or consistently inferred from context, and extracted durations (where present) align with the typical temporal scale of the underlying phenomena. Confidence scores for these categories are generally high, with most events assigned values in the range 0.8–0.95, reflecting strong linguistic cues and unambiguous reporting in the source articles.

Social and policy-related disruptions, such as labour strikes, tariffs, and trade embargoes, display greater variation in extracted detail. This is expected, as these events are often described in more abstract or forward-looking terms, with less precise timing or duration. Nonetheless, the extraction remains consistent in identifying the relevant disruption type and geographic scope, and confidence scores remain informative, decreasing smoothly in cases where reporting is vague, multi-country, or policy-oriented rather than event-specific.

Importantly, the confidence score provided by the extraction stage functions as a useful reliability signal rather than a binary correctness indicator. High-confidence events typically correspond to narrowly defined, clearly reported disruptions, while lower-confidence events tend to involve diffuse locations, extended time horizons, or indirect supply-chain effects. This allows downstream stages, such as validation and modelling, to incorporate extraction uncertainty explicitly rather than treating all events as equally reliable.

Overall, these results indicate that the LLM-based extraction is reliable in producing structured, interpretable event representations across a wide range of disruption types, while remaining conservative in ambiguous cases. Rather than maximising recall at the cost of noise, the system prioritises clarity and traceability, which is essential for subsequent validation against external datasets and for use in quantitative disruption-likelihood modelling.

2.8 Database Entry Validation

To assess the accuracy and coverage of extracted disruption events, the pipeline includes a validation stage that compares them against trusted external reference datasets. This component is currently under active development. At present, smoke tests exist for all reference sources to verify data access, schema stability, and basic retrieval.

The initial validation effort focuses on floods and protests, and the current set of reference datasets reflects this scope: DFO, EM-DAT, GDACS, NASA EONET, ReliefWeb, ACLED, and MMAD. These were selected to provide complementary coverage of physical hazards and social unrest. As additional disruption types are introduced, further specialist datasets will be investigated and integrated.

Validation is designed to operate in two directions. In *forward validation*, each extracted event (starting with floods) is compared against reference datasets to determine whether a corresponding real-world event can be found. In *inverse validation*, the process is reversed: reference events are checked against the extracted dataset to measure how many known events were successfully captured. Together, these two views provide insight into both precision and coverage.

Extracted events are often incomplete, with missing dates, vague locations, or limited contextual detail. To account for this, the validation pipeline includes simple inference and normalisation steps before matching. Dates may be inferred from article text or publication timestamps, and locations may be inferred from textual mentions or URL metadata. Event matching is therefore performed using flexible criteria, combining approximate agreement in time, location, and textual description rather than requiring exact matches.

The validation system is implemented as a modular pipeline. At a high level, it loads and filters extracted events by type, enriches and normalises their metadata, ingests and standardises reference datasets, generates plausible match candidates, and scores these candidates based on temporal, geographic, and textual similarity. The final outputs are summary statistics, lists of matched and unmatched events, and diagnostics highlighting systematic coverage gaps.

2.8.1 Component Breakdown

Figure 7 illustrates the modular structure of the database validation stage and the sequence in which functions are applied. Validation begins by loading the LLM-extracted event records and applying scoped filters (e.g. by disruption type) alongside basic completeness profiling. Extracted records are then passed through metadata enrichment functions, which infer or normalise event dates and locations to produce a consistent internal representation.

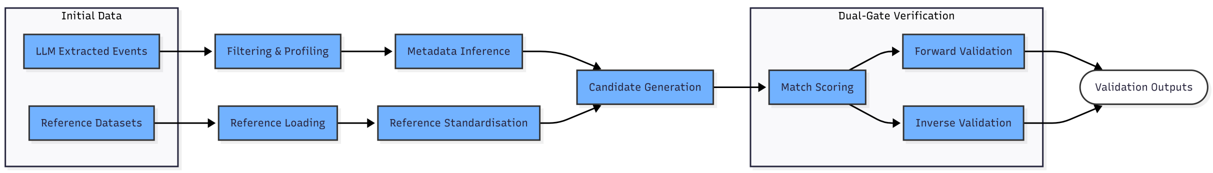

Figure 7: Flowchart of processing pipeline.

In parallel, reference datasets are ingested via dataset-specific loaders and transformed into a unified reference schema through standardisation functions. The two streams are joined during candidate generation, where time-based, geographic, and text-based candidate matches are produced for each extracted event. These candidates are evaluated using a set of scoring functions that compute similarity scores across individual dimensions and combine them into an overall match score.

Based on the resulting scores, best-match selection and classification functions determine whether an extracted event corresponds to a reference event. Forward validation functions operate on extracted events to identify matching references, while inverse validation functions traverse reference events to assess extraction coverage. The final stage aggregates the outputs of both passes into structured validation summaries, unmatched-event reports, and coverage diagnostics for downstream analysis.

2.9 Indicator Suitability Metrics

Once a disruption event dataset has been constructed and validated, the next challenge is deciding which indicators are appropriate for modelling disruption likelihood. In practice, not all indicators are equally useful. Some offer strong spatial coverage but weak temporal resolution, others are updated frequently but lack historical depth, and some are only loosely related to the underlying disruption mechanism. Treating all indicators as equally suitable risks introducing noise, bias, or false confidence into downstream models.

The Indicator Suitability Metrics (ISM) stage is therefore introduced to provide a systematic, transparent way of assessing indicator quality before it is used for modelling. Rather than selecting indicators ad hoc, ISM evaluates each candidate indicator against a common set of criteria reflecting relevance, coverage, reliability, and practical usability. This makes the indicator-selection process explicit and reproducible, and allows trade-offs between indicators to be reasoned about directly.

A further motivation for ISM is scalability. As new disruption types and indicator sources are added, manual judgement becomes increasingly inconsistent and difficult to maintain. By scoring indicators in a structured way, based of the schema in the extracted disruption dataset, the pipeline can be extended to new domains without redesigning the modelling approach or relying on undocumented assumptions.

The output of this stage is a ranked view of indicators and disruption types based on how well they are supported by available data. This ensures that downstream likelihood modelling is restricted to cases where indicators are sufficiently strong, reducing wasted effort on poorly supported disruption types and improving the robustness of subsequent analysis.

2.9.1 Component Breakdown

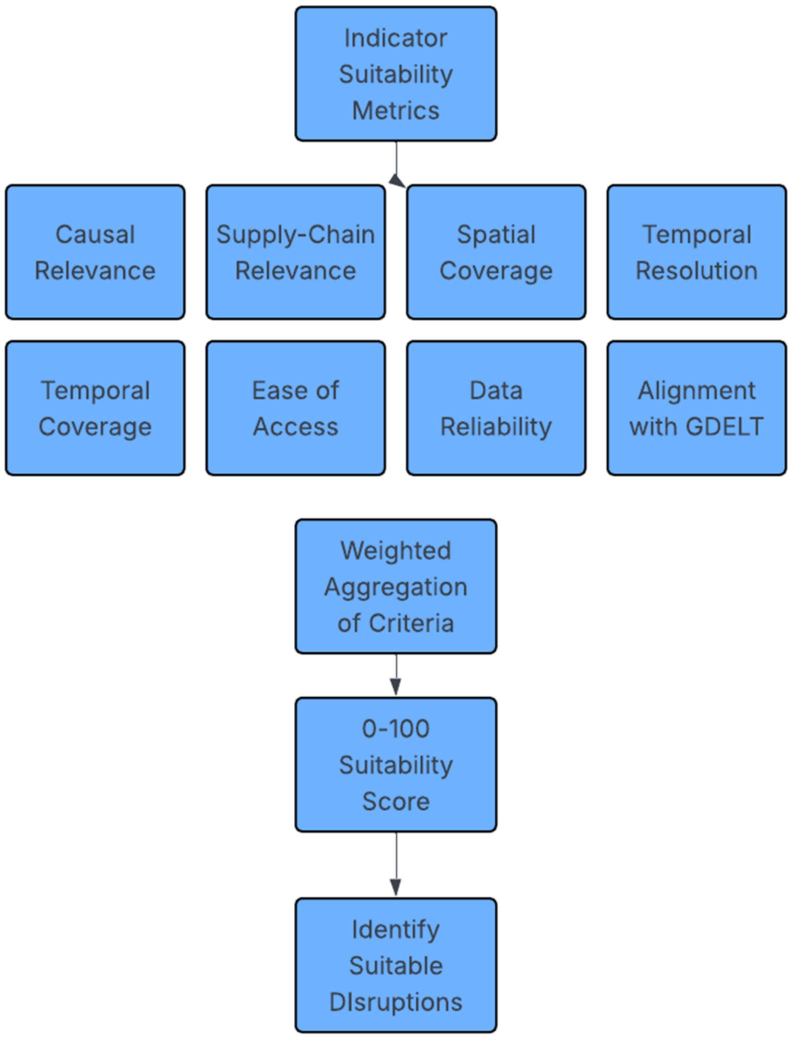

Figure 8: Flowchart of processing pipeline.

2.10 Populate Missing Indicator Values

News-derived disruption events provide high-frequency signals of when and where disruptions occur, but they rarely capture the full set of economic, social, and institutional conditions surrounding those events. In many cases, relevant indicators are either not mentioned in news reporting at all or only become available after more detailed analysis has been carried out by specialist organisations. Populating missing indicator values is therefore essential for constructing complete time series that can be used to model disruption likelihood rather than simply reacting to observed events.

To address this, the pipeline is designed to integrate a range of external datasets that measure background conditions before, during, and after disruption events. These datasets are used to fill gaps in coverage, provide context where news reporting is sparse, and allow indicators to be evaluated consistently across periods with and without observed disruptions.

2.10.1 Social Disruptions

World Bank – World Development Indicators (WDI). The World Bank's World Development Indicators provide globally comparable measures covering unemployment, GDP per capita, inflation, poverty, and trade exposure. These indicators are useful for characterising baseline economic vulnerability and long-run structural conditions across countries. Coverage is global at the national level, with most indicators reported at an annual resolution (and a small subset available quarterly for selected countries). While this limits their ability to capture short-term dynamics around individual disruption events, WDI indicators are well suited for representing background economic conditions that shape exposure and resilience.
Dataset URL: *https://databank.worldbank.org/source/world-development-indicators*

International Labour Organization (ILO). The International Labour Organization publishes labour-market indicators such as employment levels, labour force participation, informality rates, and industrial relations statistics. These indicators are particularly relevant for analysing labour-related disruptions, including strikes, mine shutdowns, and workforce shortages. Data are typically available at the national level and reported at an annual resolution, with limited quarterly coverage for selected economies. While conceptually well aligned with supply-chain risk, temporal resolution and country coverage remain uneven, limiting their use for event-level timing.
Dataset URL: *https://ilostat.ilo.org/data/*

Armed Conflict Location & Event Data Project (ACLED). ACLED provides high-frequency, geolocated records of protests, strikes, riots, and political violence. Events are recorded at a daily temporal resolution and typically resolved to the sub-national level, often at the city or district scale. These data are valuable both for validating news-derived disruption events and for constructing quantitative indicators of social unrest intensity. ACLED's strengths lie in its temporal precision and detailed event categorisation, though reporting density and geographic coverage can vary across regions.
Dataset URL: *https://acleddata.com/*

Worldwide Governance Indicators (WGI). The Worldwide Governance Indicators capture longer-term institutional conditions such as political stability, regulatory quality, and rule of law. Indicators are reported at the national level with an annual temporal resolution. As a result, WGI measures are best interpreted as slow-moving background indicators that influence structural vulnerability and institutional capacity, rather than as predictors of the timing of individual disruption events.
Dataset URL: *https://data360.worldbank.org/en/search*

National statistical offices and sector-specific regulators. Country-level agencies, including labour ministries, national statistical offices, and mining or transport regulators, often publish more granular or timely indicators for key producing regions. Spatial resolution can range from national to sub-national or facility-level, and temporal resolution may be monthly or quarterly in some cases. These sources can provide valuable local detail but are difficult to integrate consistently due to differences in definitions, reporting standards, update frequencies, and long-run availability across jurisdictions.

2.11 Natural Disruptions

2.11.1 Flood Indicators

Flooding is selected as the first natural disruption type examined in this project due to its frequent and well-documented impacts, along with the wealth of data surrounding the indicators and disruption events. Flood-related disruption risk is characterised using a set of hydrological and meteorological indicators capturing both precipitation forcing and catchment response. The datasets described below are globally available, programmatically accessible, and provide sufficient temporal and spatial resolution to align indicators with disruption events by date and location.

CHIRPS (Rainfall and Rainfall Anomalies). The Climate Hazards Group InfraRed Precipitation with Station data (CHIRPS) dataset provides gridded precipitation estimates derived from satellite observations blended with rain gauge data. Owing to its long historical record and relatively high spatial resolution, CHIRPS is used to derive rainfall totals and rainfall anomalies.
Temporal resolution: daily.
Spatial resolution: approximately 0.05° (~5 km).
Dataset URL: *https://www.chc.ucsb.edu/data/chirps*.

GPM IMERG (Rainfall Intensity). The Global Precipitation Measurement (GPM) IMERG product provides high-frequency precipitation estimates designed to capture short-duration, high-intensity rainfall. This dataset supports the construction of rainfall intensity indicators relevant to flash flooding, complementing daily accumulation measures.
Temporal resolution: 30 minutes.
Spatial resolution: approximately 0.1° (~10 km).
Dataset URL: *https://gpm.nasa.gov/data/imerg*.

ERA5 Reanalysis (Soil Saturation). ERA5 is a global atmospheric reanalysis produced by the European Centre for Medium-Range Weather Forecasts (ECMWF), providing internally consistent meteorological and land-surface variables. Volumetric soil moisture estimates strongly conditions flood severity for a given rainfall event.
Temporal resolution: hourly.
Spatial resolution: approximately 0.25° (~31 km).
Dataset URL: *https://www.ecmwf.int/en/forecasts/datasets/reanalysis-datasets/era5*.

GloFAS (River Discharge). The Global Flood Awareness System (GloFAS) provides modelled river discharge estimates across the global river network, together with return-period thresholds for extreme flows. These outputs are used to derive river discharge indicators and proxy measures of flood severity downstream of disruption locations.
Temporal resolution: daily.
Spatial resolution: approximately 0.1° along major river networks.
Dataset URL: *https://www.globalfloods.eu*.

At this stage, indicator population is treated as a data-fusion problem rather than a single-source solution. Different indicators serve different roles depending on disruption type, geography, and time scale. Candidate datasets are therefore evaluated based on relevance, spatial and temporal resolution, coverage, and empirical alignment with observed disruption events. The specific datasets considered in detail are introduced in the following section, *Data Source Deep Dive.*

3 Data Source Deep Dive

3.1 Source Documentation

This project integrates multiple external datasets and services for news ingestion, event extraction, and validation. The sources listed below represent the datasets currently used in the pipeline, primarily to support validation of floods and protest-related disruptions. As additional disruption types are incorporated, further indicator and reference datasets will be evaluated and added using the same documentation standards.

GDELT. The Global Database of Events, Language, and Tone is used as the primary upstream news ingestion source. Daily event context files are accessed via publicly available HTTP endpoints *http://data.gdeltproject.org/gdeltv2/masterfilelist.txt*. GDELT data are released under an open-access model and do not require authentication or API keys.

DFO (Dartmouth Flood Observatory). Flood event data from the Dartmouth Flood Observatory are used as a physical reference catalogue for flood validation. Data are accessed via publicly available datasets and summary tables: *https://floodobservatory-old.colorado.edu/*. Use is likely subject to academic citation standards, however the website is in the process of being updated, and as it stands the privacy policy section is blank. No API key is required.

EM-DAT. The Emergency Events Database, provides historical records of natural disasters. *https://public.emdat.be/* Access requires user registration and acceptance of EM-DAT's data usage terms. Data are used strictly for research and non-commercial purposes in accordance with the licence.(An account has been created linked to my university email but the setup process is instant).

GDACS. The Global Disaster Alert and Coordination System, *https://gdacs.org/*, provides near-real-time operational alerts for major disasters, including floods. Data are accessed via public feeds and APIs. No authentication key is required for basic access, subject to attribution and fair-use guidelines.GDACS data are publicly accessible and provided for informational and research use, subject to explicit disclaimers regarding accuracy, completeness, and liability. The service is "not intended to replace official national alerts or decision-making authorities".

NASA EONET. NASA's Earth Observatory Natural Event Tracker, *https://eonet.gsfc.nasa.gov/*, supplies machine-readable feeds of detected natural hazard events derived from satellite and remote sensing sources. Data are accessed via a public REST API under an open data policy, with no authentication or API key required.

ReliefWeb. ReliefWeb, *https://reliefweb.int/*, is used as a humanitarian disaster reference source and aggregation platform. Data are accessed via the ReliefWeb API, which requires registration and the use of an application identifier. API access is typically approved within one working day and in this case is linked to my university email address.

ReliefWeb acts as a distributor of content provided by external partner organisations (e.g. UN agencies, NGOs, and humanitarian bodies) and is not the copyright holder for most materials. As a result, content retrieved via the API is used for event validation and analysis rather than direct redistribution. Where individual reports or datasets are referenced, attribution is made to the original source organisation in accordance with ReliefWeb's data usage and referencing policies.

ACLED. The Armed Conflict Location & Event Data Project (ACLED) provides geocoded data on conflict, protest, and strike-related events. Access requires account registration, *https://acleddata.com/*, and the use of an API key or authorised data export, which is provided following user registration.

ACLED data are licensed by default for non-commercial academic and research use and are subject to specific use restrictions. These include prohibitions on redistribution of raw data, use of the data to train machine-learning or artificial intelligence models without a separate licence, and creation of competing products or direct substitutes. Reuse is limited to analytical or transformed derivatives, and proper attribution to ACLED is mandatory in any analysis, visualisation, or reporting.

MMAD. The Mass Mobilization in Autocracies Database (MMAD) is used as a supplementary reference source for protest and mass mobilisation events. MMAD is an open-access dataset, *https://mmadatabase.org/*, distributed under a Creative Commons Attribution–NonCommercial–ShareAlike 4.0 International licence (CC BY-NC-SA 4.0). This licence permits reuse, redistribution, and adaptation for non-commercial purposes, provided appropriate attribution is given and any derivative works are shared under the same licence.

ChatGPT API (LLM Extraction). Large language model–based event extraction is performed using the ChatGPT API provided by OpenAI. The API is accessed via authenticated requests using a private API key stored securely outside the codebase (using environment variables). The service is used to transform cleaned article text into structured disruption event records. Usage is subject to OpenAI's API terms of service, rate limits, and cost constraints.

Extensibility. The set of documented sources is expected to expand as additional disruption types are incorporated and as missing indicator values are identified. All new datasets and services will be evaluated with respect to licensing constraints, access stability, update frequency, and relevance before being integrated into the pipeline and documented in this section.

3.2 Data Acquisition Pipeline

This section describes the planned data acquisition and preparation workflow for external datasets used in news ingestion and database validation. At the current stage of the project, these datasets have been integrated only to the extent required for access verification and smoke testing. The steps outlined below describe how data will be systematically retrieved, cleaned, transformed, and stored as the pipeline is extended to additional disruption types and indicators.

External data will be retrieved either via public or authenticated APIs (e.g. GDELT, ReliefWeb, ACLED, GDACS, NASA EONET) or through bulk downloads and flat-file datasets (e.g. EM-DAT, DFO, MMAD). Dataset-specific access requirements, including API keys, application identifiers, pagination, and rate limits, will be handled at this stage. Retrieval will be scoped by time window, event type, and geographic coverage to limit unnecessary data volume.

Where appropriate, raw responses will be persisted prior to processing, together with minimal retrieval metadata such as source name and retrieval timestamp. This provides reproducibility, enables reprocessing if downstream logic changes, and reduces dependence on upstream data availability or schema stability.

Retrieved data will then be transformed from source-specific schemas into a common internal representation. This normalisation step will involve renaming fields, coercing data types, and attaching consistent source identifiers, allowing heterogeneous datasets to be processed uniformly by downstream validation and analysis components.

Following normalisation, records will be filtered to the relevant temporal scope, geographic coverage, and disruption types supported at each stage of development. Cleaning steps will then be applied to remove duplicate records, handle missing or malformed fields, standardise date formats, and normalise country and location names. For reference datasets, additional care will be taken to manage inconsistent spatial granularity and overlapping records across sources.

Lightweight enrichment may be applied where necessary, such as the addition of standardised country codes or internal event identifiers. This enrichment is strictly preparatory and does not involve modelling or inference.

Processed datasets will be stored in structured, versioned formats (e.g. CSV or JSONL), with a clear separation between raw, cleaned, and standardised outputs. As the pipeline matures, dataset-specific integrity checks and smoke tests will be expanded to verify record counts, required fields, and schema consistency before data are passed to downstream extraction, validation, or indicator suitability stages.

Dataset	Primary Use	Access Method	Auth Required	Stored Format
GDELT	News ingestion	Public HTTP (bulk files)	No	CSV / JSONL
DFO	Flood validation	Public dataset download	No	CSV
EM-DAT	Historical disasters	Registered download	Yes (account)	CSV
GDACS	Disaster alerts	Public API / feeds	No	JSON
NASA EONET	Natural hazard events	Public REST API	No	JSON
ReliefWeb	Humanitarian validation	REST API	Yes (app identifier)	JSON
ACLED	Conflict / protests	REST API / export	Yes (API key)	CSV / JSON
MMAD	Protest validation	Flat-file download	No	CSV

Table 1: Summary of external data sources and acquisition methods.

4 Pipeline User Guide & Interface Manual

4.1 Core Function Usage

This section describes how the pipeline is intended to be used in practice, focusing on the primary workflow for progressing from raw news data to extracted disruption events, and ultimately to disruption-likelihood estimates. At present, the system supports event discovery and structured extraction; likelihood estimation is a planned downstream extension that will be implemented once validation and modelling stages are complete.

4.1.1 Current Usage: From News to Structured Disruption Events

The current pipeline is organised as a sequence of executable components, each responsible for a clearly defined stage of processing. Usage is therefore procedural rather than interactive.

Step 1: News Retrieval and Relevance Filtering The entry point to the pipeline is the `pipeline.py` script located in the *Relevant News Retrieval* folder. This script is run with a specified date as input, corresponding to the day of news coverage to be processed. When executed, the script performs the following steps automatically:

- downloads GDELT event context data for the requested day;
- extracts candidate article URLs and removes duplicates;
- filters out clearly irrelevant content using conservative rules;
- fetches article titles and meta descriptions;
- cleans and normalises the retrieved text;
- applies a supervised relevance classifier (SVM) to identify articles likely to describe supply-chain-relevant disruptions.

The output of this stage is a set of high-confidence, disruption-relevant articles saved to disk in a structured intermediate format. This filtering step is critical for scalability, as it determines which articles are passed to more expensive downstream processing.

Step 2: LLM-Based Disruption Extraction Once relevant articles have been identified, the next stage is executed from within the `Database Builder` directory by running `DisruptionExtractor.py`. This script iterates over the filtered articles and applies the LLM-based extraction component to each item.

For every article, the extractor converts unstructured text into a structured disruption record, including the disruption type, event date, location, duration (where available), supporting evidence snippets, and a confidence score. Outputs are written to persistent `JSONL` and `CSV` files, forming the core disruption event database.

For rapid inspection and debugging, a user-friendly summary of extracted events can be generated by running `DisplayExtractionPandas.py`, which loads the extraction outputs into a tabular format for visual inspection and exploratory analysis.

Step 3: Validation (In Progress) Validation is performed within the `DatabaseValidation` module. At this stage, extracted events are compared against external reference datasets to assess authenticity, timing, and location consistency. While this functionality is partially implemented, comprehensive validation across all disruption types is still ongoing and will be completed before modelling is undertaken.

4.2 Planned Usage: From Indicators to Disruption Likelihoods

The intended final usage of the pipeline is to support queries of the form:

> *"Given a set of current or historical indicator values, what is the likelihood of a disruption occurring?"*

In the completed system, validated disruption events will be combined with time-aligned indicator datasets capturing physical, social, political, and economic conditions. Statistical and machine-learning models will then be trained on both event and non-event periods to estimate disruption likelihoods as a function of these indicators.

At present, this modelling interface is not yet available. The required components—namely fully validated event labels, complete indicator time series, and trained likelihood models—are still under development. Once implemented, this stage will provide a unified interface for querying disruption risk based on live or historical indicator values, completing the transition from event discovery to quantitative risk assessment.

4.2.1 Summary

In summary, the current pipeline enables users to progress from raw global news data to structured, interpretable disruption events through a sequence of well-defined scripts. While direct likelihood estimation is not yet supported, the existing architecture is designed to accommodate this functionality once validation and modelling stages are complete.

4.3 Setup and Installation

This section describes how to set up the processing pipeline in a new environment. The full codebase is hosted on GitHub and can be cloned directly from the project repository[1].

The pipeline is implemented in Python (version 3.10 or later). All required dependencies are listed in a `requirements.txt` file and should be installed in a virtual environment prior to execution.

Environment variables and credentials. All credentials and service configuration values are managed via environment variables and are not committed to version control. For local development, these are provided through a `.env` file at the repository root, which is loaded at runtime using `python-dotenv`.

[1]Repository URL: `https://github.com/Ben-Fenocchi/IIB-Project`

The following environment variables are currently required:

- `OPENAI_ADMIN_KEY`: Administrative API key used for accessing OpenAI account-level usage and billing endpoints.
- `OPENAI_PROJECT_KEY`: Project-scoped API key used for ChatGPT-based LLM event extraction.
- `RELIEFWEB_APPNAME`: Application identifier required for requests to the ReliefWeb API.
- `ACLED_EMAIL`: Email address associated with the registered ACLED account.
- `ACLED_PASSWORD`: Password used for authenticated ACLED data access.

Additional environment variables may be introduced as further APIs and services are integrated.

Creating the `.env` file. A local `.env` file should be created with the following structure:

```
OPENAI_ADMIN_KEY=your_admin_key_here
OPENAI_PROJECT_KEY=your_project_key_here
RELIEFWEB_APPNAME=your_reliefweb_app_identifier
ACLED_EMAIL=your_acled_email
ACLED_PASSWORD=your_acled_password
```

This file must not be committed to the repository and should be excluded via `.gitignore`.

Manually downloaded datasets. Several validation datasets are currently accessed via manual download rather than APIs. These datasets must be downloaded separately and placed in the appropriate data directories before running validation scripts. Links to each dataset will be provided in the repository and documentation:

- EM-DAT — historical disaster database (`https://public.emdat.be/`)
- DFO (Dartmouth Flood Observatory) — physical flood catalogues (`https://floodobservatory-old.colorado.edu/temp/`)
- MMAD — Mass Mobilization in Autocracies Database (`https://mmadatabase.org/mmad-repressive-actors/`)

Downloaded datasets are expected to be stored in clearly named subdirectories within the project's data directory, separate from raw API outputs.

Verification and smoke testing. Once dependencies are installed, environment variables are configured, and required datasets are downloaded, dataset-specific smoke tests can be run. These tests verify authentication, connectivity, file availability, and basic schema assumptions before executing the full extraction or validation pipeline.

Extensibility. As the project expands to additional disruption types and data sources, further APIs, credentials, and manually downloaded datasets may be required. All such additions are expected to follow the same environment-based configuration and documented setup process described above.

5 Future Work and Roadmap

The current pipeline provides a scalable method for identifying candidate supply-chain disruption events from global news data. The remaining work in the project focuses on improving the reliability of these events, completing the retrieval of disruption indicators, and using the resulting data to estimate disruption likelihoods.

5.1 Validation and Consolidation of Disruption Events

The most immediate next step is to complete validation of the extracted disruption events. Events identified from news articles must be checked against external reference datasets, such as records of natural hazards, industrial accidents, labour actions, and policy interventions, where such data are available. This validation serves two purposes. First, it confirms that extracted events correspond to real-world disruptions and helps remove false positives. Second, it clarifies where news-derived data provide additional detail, for example by offering more precise timing or location than reference datasets alone.

Validation will be carried out in both directions. Forward validation will assess whether each extracted event can be matched to a known reference event within reasonable time and location bounds. Inverse validation will identify reference events that were not captured by the news-based pipeline, highlighting systematic gaps in coverage. Completing this step is necessary to establish a reliable set of labelled disruption events for subsequent analysis.

5.2 Completion of Indicators and Handling of Missing Data

Although candidate indicators have been defined, many are not yet complete across the full time period of interest. This is partly due to uneven reporting across regions and partly due to gaps in external data sources. The next stage will therefore focus on identifying where indicator values are missing and determining how these gaps should be handled.

A key task is handling the disparity between indicator temporal resolution and disruption occurences, if say a strike initiates in between quarterly unemployment figures. Where appropriate, missing values will be populated using simple and transparent methods such as interpolation, aggregation over longer time windows, or the combination of related indicators. The aim is to produce time series that consistently represent both disruption and non-disruption periods, while avoiding the introduction of artificial signals driven by data availability rather than underlying risk.

5.3 Pipeline Refinements and Outstanding Technical Debt

One significant area of outstanding technical debt concerns the supervised classification stage used to filter GDELT-derived URLs for relevance before downstream processing. At present, this classifier provides a useful first-pass reduction in volume, but its performance can be further improved through additional training and refinement.

Improving this component requires manual labelling of news articles to expand the training dataset, which is time-consuming but necessary to increase classification accuracy. This work is therefore scheduled as a dedicated refinement step rather than being addressed incrementally.

Strengthening the relevance classifier is expected to significantly reduce downstream computational and monetary costs. By more reliably filtering out irrelevant articles at an early stage, only high-confidence disruption-related URLs will be passed to full-text scraping and large language model-based extraction. This reduces unnecessary API calls, shortens processing time, and improves overall pipeline efficiency, making the system more scalable for extended time horizons or additional commodities.

5.4 Disruption Likelihood Modelling

The final stage of the project will use the validated events and completed indicators to estimate disruption likelihoods. Disruption events will be combined with economic, political, environmental, and social indicators in a time-aligned dataset that includes both event and non-event periods.

Using historical data, statistical or machine-learning models will be trained to identify which indicator patterns are associated with an increased probability of disruption. Including non-event periods is essential, as it allows the model to distinguish between conditions that are simply variable and those that are genuinely linked to disruption risk. Backtesting can then be used to gather empirical evidence into the efficacy of the model.

As more validated data become available, the modelling framework can be refined to focus on specific types of disruption or particular regions. The resulting likelihood estimates will then feed into a broader criticality assessment of the lithium supply chain, providing a quantitative and empirically grounded measure of disruption risk.

6 Data Repository

https://github.com/Ben-Fenocchi/IIB-Project