Scalr Technical

Scalr

Real-Time Crypto Trend Intelligence from X (Twitter)

Version: 0.1 | Date: 23 July 2025 | Authors: Avex | Contact: @wencex

Abstract

SCALR is an AI-augmented intelligence platform that ingests live X (Twitter) data, filters for crypto-market relevance direct/indirect clusters emerging narratives using unsupervised learning, and streams insights to an immersive 3D interface. A hybrid LLM + statistical pipeline (LLMs, SentenceTransformers, TF-IDF, HDBSCAN) powers keyword mining, persona extraction, engagement scoring, and trend labeling. The system exposes a backend with SSE streams for low-latency delivery, while a React/Three.js frontend renders a "cosmos" of tweets and clusters for both public and premium users.

SCALR aims to become a signal engine for traders, builders, personal brands and analysts, surfacing narratives before they are obvious, while reducing noise, spam, and scams. This paper details the architecture, algorithms, and operational considerations. Optional crypto-economic mechanisms (access tokens, staking for signal quality, DAO governance) are proposed in *9.

1. Motivation & Problem Statement

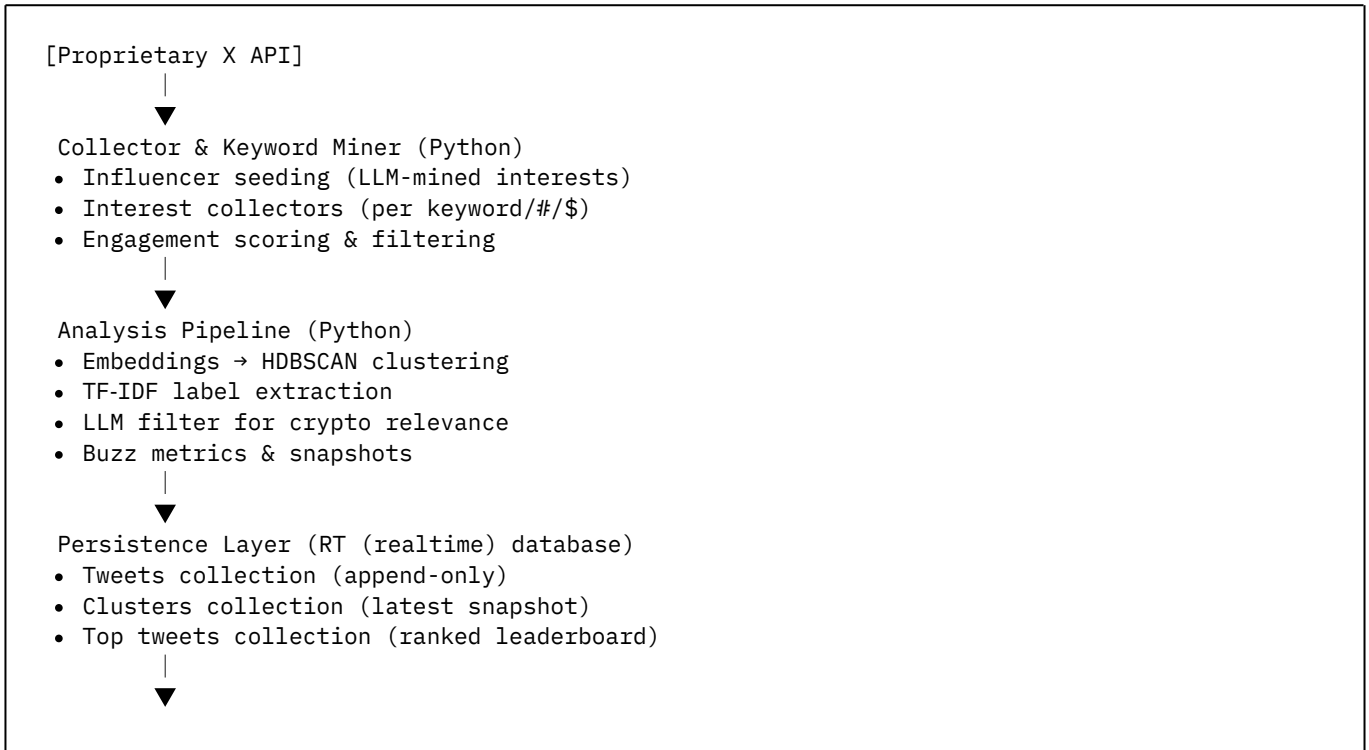
Crypto markets are narrative-driven and extremely fast-moving. Alpha often originates on social platforms long before it reaches price charts. Manual monitoring is noisy, fragmented, and biased. Existing tools focus on price/volume metrics or raw social counts, lacking:

- **Contextual relevance** (what matters to crypto vs. general chatter)
- **Semantic clustering** of narratives vs. hashtag spam
- **Real-time UX** for exploring relationships, not dashboards of tables
- **Programmable access** to structured trend data

SCALR addresses this by:

- Continuously harvesting high-engagement tweets and interest-specific streams
- Ranking and clustering content into labeled “trend clusters” with buzz scores
- Providing developer and UI endpoints to consume and visualize insights

2. High-Level System Overview



API Backend (REST + SSE)
- /tweets, /stream, /top, /top_stream
- /interest, /interest/watch (per-keyword threads)
- /persona, /rewrite (LLM utilities)

React + Three.js Frontend
- Public dashboard (global feed)
- Premium dashboard (user-selected interests)
- 3D "Cosmos" & "Trend Radar" visualizations

3. Architecture

3.1. Data Collection Layer

- **Sources:** X/Twitter using scalr endpoints (search-v) and a custom serverless endpoint for influencer timelines.
- **Seed Interests:** LLM extracts up to 5 high-impact crypto interests from influencer tweets (kw_from_llm).
- **Interest Watchers:** /interest endpoint spawns a thread (collect_for_interest) to poll searches for a keyword/hashtag/cashtag at configurable intervals, persisting to store/interests/<interest> collection.
- **Engagement Scoring:** Custom heuristic formula not-disclosed:
  - Follower bonuses at ≥10k and ≥100k
  - Freshness bonus for < 120 minutes old
- **Deduplication:** seen_ids collection and memory sets prevent re-processing.

- ***TBD:*** *Rate limits & backoff strategies; legal/ToS compliance for X data + 100% compliance to their robots.txt.*

3.2. Motivation & Problem Statement

- **Pre-Filter:** Tweets scoring ≥ FILTER_SCORE are retained.
- **Text Cleaning:** URLs stripped via regex.
- **Embeddings:** SentenceTransformer("all-MiniLM-L6-v2") encodes tweet text.
- **Clustering:** hdbscan.HDBSCAN(with a custom cluster_size) groups semantically similar tweets. Noise cluster = X.
- **Cluster Labeling:** TF-IDF over (1–3)-grams picks top n terms (label() helper) per cluster.
- **Buzz Metric:** buzz = Σ(score × freshness_factor) for cluster members.
- **LLM Relevance Filter:** Opensource model (meta-llama) picks which top tweets and clusters are crypto-relevant.
- **Keyword Expansion:** Returned cluster labels explode into new search terms for the next loop.

3.3 Persistence & State

- **Tweets:** Append-only log of enriched tweets (append_tweets).
- **Clusters:** Latest cluster snapshot incl. tweets.
- **Top_full** Time-stamped leaderboard of top N tweets each analysis cycle.
- **Categories / categories_full:** for bucket categorizations.

Storage is currently using a realtime DB (firebase GCP) with indexing.

3.4 API & Streaming Layer (Scalr API)

Endpoint	Method	Purpose
/tweets	GET	Tail of tweets log (initial load)
/stream	GET (SSE)	Live tweet stream increments
/top	GET	Last N top-ranked rows from top_full
/top_stream	GET (SSE)	Live stream of top tweets
/clusters	GET	Cluster map + metadata (labels, buzz)
/clusters/tweets	GET	Full clusters with tweets
/clusters/{id}/tweets	GET	Tweets in a single cluster
/interest	GET	Schedule/read/reset collector for interest
/interest/watch	GET (SSE)	Live tweets for an interest
/persona	POST	Build persona from tweet array
/rewrite	POST	Rewrite a tweet (with/without persona)
/search	GET	Proxy username search
/health	GET	Status & heartbeat

- SSE provides minimal overhead real-time streaming vs. WebSockets.
- Background threads (daemon) for collectors; startup hook starts main collector.
- CORS open to * for dev; custom JWT access will be added for prod.

3.5 Frontend (React/Three.js)

- **Framework:** React + Zustand state, React Router, Tailwind UI; Three.js.
- **Auth/Wallet:** @dynamic-labs/sdk-react-core + Ethereum connectors.
- Stores:
  - userStore collection (name, interests, X username, onboarding state)
  - statusStore collection (collecting/analyzing/live flags, premium mode)
  - cryptoStore collection (tweets, clusters, 3D star particles, selections)
- **Public vs. Premium:**
  - Public: global stream /stream, clusters polling.
  - Premium: user interests spawn dedicated collectors & SSE watchers.
- **3D Views:**
  - Cosmos (public) & PremiumCosmosView (premium): Avatar spheres, cluster spokes, animated star field, orbit controls. Top tweets haloed.

- **Panels:** Panel, TopTweets, AllTweets - interactive UI for details, rewrite wizard.
- **Rewrite Wizard:** Persona mode fetches user timeline → persona → rewrite tweet.

4. Algorithms & Models

4.1 Engagement Score

score(tweet, followers) = Not-disclosed.

- Bonused for follower tiers (≥10k, ≥100k) and freshness (< 2h).
- Tunable thresholds: ENG_THRESH, FILTER_SCORE.

4.2 Semantic Clustering

- **Embeddings:** Dense 384-dim vectors (MiniLM-L6-v2).
- **Clustering:** HDBSCAN finds variable-density clusters without k.
- **Noise handling:** cluster -X ignored in buzz.

4.3 Cluster Labeling

- TF-IDF across cluster texts → top 3 n-grams concatenated.
- Perspective: KeyBERT, YAKE, or LLM summarization for human-readable titles.

4.4 Crypto Relevance Filter (LLM)

- Prompt instructs removal of spam & non-crypto lines.
- Outputs JSON: { "tweets": [indices], "clusters": [labels] }.
- Model: Opensource (Llama-4) for structured output.

4.4 Crypto Relevance Filter (LLM)

- Personas built from timeline tweets: User Tweets exctraction + tone style summary.
- Rewrite function enforces output-only policy to avoid hallucinated boilerplate.
- Stop sequences to cut explanations; sanitizes quotes.

5. Data Schemas

5.1 Tweet Object (enriched)


```
{
  "tweet_id": "string",
  "created_at": "ISO8601",
  "lang": "en",
  "hashtags": ["BTC", "ETF"],
  "symbols": ["BTC"],
  "favorite_count": 123,
  "retweet_count": 45,
  "reply_count": 12,
  "quote_count": 3,
```

```
 "possibly_sensitive": false,
 "text": "raw tweet text…",
 "author_id": "123456",
 "author": { …see Author schema… },
 "score": 456,
 "cluster_label": "bitcoin etf approval, blackrock, sec",
 "cluster_buzz": 789.2
}
```

5.2 Cluster Snapshot

```
{
 "ts": "ISO8601",
 "cluster_id": 3,
 "label": "bitcoin etf approval, blackrock, sec",
 "buzz": 1234.5,
 "tweets": [<Tweet objects>]
}
```

6. Security, Privacy & Compliance

- **API Keys & Secrets:** Secret managers (GCP Secret Manager, Vault). **TBD:** rotation policy.
- **LLM Safety:** We ensure no PII leaks; persona builder only uses public tweets.
- **X/Twitter ToS:** Our scraping/search API usage complies with licensing.
- **User Data:** Only minimal profile info is persisted (localStorage for onboarding). For premium users, wallet addresses are handled by Dynamic Labs SDK - review their security model.
- **DoS & Abuse:** Our endpoints are rate-limited and protected from abuse, auth for premium mode endpoints.
- **Content Moderation:** LLM relevance filter removes scams; added heuristic blacklists and verification.

7. Performance & Scalability

- **Current:** Single-process Scalr API with thread-based collectors; local distributed I/O.
- **Bottlenecks:**
  - Collection based .db tailing; remote contention
  - Embedding & clustering on a hosted GPU.
  - Thread proliferation for many interests
- **Scaling Plan:**
  - Collectors moved to async tasks or separate microservice (Celery/Arq/Temporal).
  - Kafka / Redis Streams for tweet events, ClickHouse for analytics.
  - Vector DB (Qdrant/Weaviate) for embeddings.
  - Batch clustering or streaming incremental clustering (e.g., incremental HDBSCAN).
  - Horizontal scale Scalr API; SSE over Nginx/GCP Cloud Run + perspective to switch to WebSockets/Socket.IO.

8. Frontend UX & Accessibility

- 3D views are optimized without being GPU-heavy
- Keyboard navigation & reduced motion modes
- Mobile responsive layout for panels, not 3D canvas
- Internationalization: i18n hooks, time formatting

8.1 Personal Brand Growth on X with SCALR Premium

SCALR isn't only market intel—it's a content engine for creators who want to scale a personal brand on X.

Workflow

1. **Input niche interests** in the Premium Dashboard (keyword, #hashtag, $cashtag, phrase).
2. SCALR spawns a dedicated collector thread per interest and polls X roughly every 5 minutes (default) to grab fresh high-engagement posts.
3. Tweets are scored, filtered, clustered, and labeled—surfacing only the juiciest, crypto-relevant content.
4. Use the Rewrite Wizard on any tweet:
    - **Scalr way** – default high-engagement rewrite style.
    - **Persona (your @handle)** – SCALR builds a persona from your own timeline and rewrites in your tone.
    - **Persona (any @handle)** – mirror any public account's style using public data.
5. **Publish & iterate:** copy to X, schedule, or A/B test versions.

Why it Scales Your Brand

- **Rapid ideation → output:** New angles every few minutes kill writer's block.
- **Consistent voice:** Persona extraction keeps tone cohesive across posts.
- **Relevance moat:** Interest targeting + LLM realtime filtering strips spam/off-topic noise.
- **Timing intelligence:** Cluster buzz + engagement scores indicate when to post.

Optional KPIs to Track

- Posts/day (baseline vs. with SCALR).
- Engagement delta (likes/RTs/replies) on rewritten vs. original tweets.
- Time from narrative emergence → first post.
- Follower growth correlated with cluster participation.

Future Extensions

- Auto-queue rewrites into schedulers (Buffer, Scalr APIs).
- Multi-variant rewrites with automatic winner selection.
- Persona "similarity score" to ensure tone fidelity.
- Real-time alerts when a watched interest spikes in buzz.

9. Crypto-Economic Design (Optional / TBD)

Possible models:

1. **Access Token (Utility):** Stake/hold to unlock premium collectors, higher rate limits, advanced analytics APIs.
2. **Signal Mining:** Users contribute curated interests or label quality; earn tokens proportional to downstream usage.
3. **DAO Governance:** Token holders vote on model params (FILTER_SCORE, ENG_THRESH), feature roadmap, dataset releases.
4. **Revenue Share:** Subscription fees flow to treasury; stakers get a share.

TBD:

- Token supply, distribution, vesting
- Utility & burn mechanisms
- Compliance (jurisdictions, KYC/AML)

10. Future Work

- Multi-source ingestion (Reddit, Farcaster, Discord)
- On-chain data correlation (DEX volumes, whale moves)
- Real-time anomaly detection & alerting (CUSUM, Twitter bot nets)
- Interactive backtesting of narrative impact vs. price
- Open API SDK (Python/JS) & webhooks

11. Appendix

11.1 Mermaid Diagrams

flowchart TD A[Influencer Tweets API] -->|hot tweets| B(LLM keyword miner) B -->|interests| C[Tweet Collectors] C --> D{Filter & Score} D -->|keep| E[Embeddings] E --> F[HDBSCAN] F --> G[TF-IDF Labels] G --> H[LLM Relevance Filter] H--> I[Snapshots JSON/JSONL] I --> J[Scalr API REST + SSE] J --> K[React/Three.js Frontend]

11.2 Key Parameters to be disclosed to the community

Parameter	Value (default)	Purpose
ENG_THRESH	X	Influencer tweet engagement threshold
FILTER_SCORE	X	Minimum score to pass filter
WINDOW_HR	X	Analysis window for clustering
MIN_CLUSTER	X	HDBSCAN min_cluster_size
TOP_TWEETS	X	Leaderboard size per cycle
SLEEP_SEC	X	Main loop sleep (dev)

11.3 Environment & Deployment

- **Backend:** Python 3.11+, Scalr API
- **LLM Provider:** Opensource hosted on a local GPU + nginx access
- **Vectorizer/Cluster:** Sentence-transformers, scikit, hdbscan
- **Frontend:** Vite/React, Zustand, three.js
- **Infra (prod):** GCP Cloud Run (API), Cloud Functions for collectors, GCS/S3 for logs

11.3 Environment & Deployment

- **Buzz:** Weighted sum capturing engagement × freshness for a cluster
- **SSE:** Server-Sent Events, unidirectional push over HTTP
- **HDBSCAN:** Hierarchical Density-Based Spatial Clustering of Applications with Noise

12. Contacts & Contributions

- **GitHub:** To be shared with the community if DAO agreed
- **Telegram:** @wencex
- **Contributions:** PRs welcome; feature requests via DMs