Evolution AI Documentation

Evolution AI is an Intelligent Document Processing (IDP) platform that transforms unstructured documents (PDFs, scanned images, mixed-format reports) into validated, normalized data ready for downstream systems. It combines pre-trained foundation models, a zero-shot extraction capability, and a human-in-the-loop self-learning loop to deliver fast time-to-value and continuously improving accuracy. This document explains the platform end-to-end, with particular emphasis on *zero-shot learning*, QA/self-learning mechanics, extraction pipeline, integrations, governance, and a financial-statement extraction example. The content is formatted to be presentation-ready and covers approximately three to four pages when printed.

1. Platform at a glance

Core capabilities

* Extract paragraphs, key-value pairs, and complex tables (nested and multi-page).
* Zero-shot extraction: works on unseen document types without explicit training examples.
* Human-in-the-loop QA with one-click retraining/self-learning.
* Post-processing for normalization (dates, currency, multipliers) and anomaly detection (time-series & summation checks).
* Enterprise integration: REST API, SFTP, Workato connector, webhooks.
* Security & governance: ISO/IEC 27001 compliance, audit trails, role-based permissions.

Typical outputs: CSV, Excel, JSON (API or bulk file), with full audit links back to source pages.

2. System architecture & high-level flow

Major components

* Ingest layer: UI upload, REST API, SFTP, Workato.
* Pre-processing: OCR, deskew, rotation correction, image enhancement.
* Layout & semantic analysis: page segmentation, block detection, table detection.
* Extraction engines: field/key-value detectors, table parsers, paragraph extractors.
* Confidence & QA manager: scoring, QA workflows, annotation interface.
* Self-learning controller: online update and model refresh after QA.
* Post-processing: normalization, currency splitting, computed fields, anomaly detection.
* Output & integration: export formats, API responses, webhooks, connectors.
* Governance & audit: logs, history, role permissions, encryption.

High-level flow: Upload → Preprocess → Layout analysis → Extract → Score → QA (if needed) → Retrain → Post-process → Output → Audit.

3. Zero-shot learning — deep dive

What is zero-shot learning (ZSL)?

Zero-shot learning enables a model to perform a new task (or label unseen during training) by leveraging indirect knowledge learned from related tasks and a broad, diverse training corpus. Instead of training a dedicated model for every document type, ZSL transfers previously learned representations and reasoning to new formats.

How ZSL is applied in Evolution AI

1. Pre-trained representations — Models are trained on very large and varied document corpora (text + layout + table examples). These models learn generic language, layout, and table structure features that generalize beyond specific templates.

2. Schema inference & semantic matching — When a new document arrives, the model:

* identifies candidate content (text blocks, table cells),
* computes embeddings (compact semantic vectors) for each block,
* compares block embeddings to *target field descriptions* (for example: "Total Revenue", "Reporting Period"), and
* selects the best matches without needing example-labeled documents of that exact layout.

3. Flexible field prompts / descriptors — Instead of hard-coded rules, the system uses human-readable field descriptions (prompts or labels) to guide extraction. This makes the system robust to synonyms, layout changes, and multilingual text.

4. Layout-aware vision-language models — For scanned or image-based documents, models combine OCR text with spatial/layout features (bounding boxes, font size) to reason about relationships (e.g., table headers vs. values).

Why ZSL reduces setup time

* Avoids the classical requirement to collect dozens or hundreds of labeled examples for each new document template.
* Allows immediate processing of unseen documents—ideal for POCs and enterprises with many varying suppliers/forms.

Limitations & mitigations

* Ambiguity: Some fields may be ambiguous without domain context (e.g., "Net" could mean net income or net assets). *Mitigation*: confidence thresholds + QA routing.
* Novel visual patterns: Extremely unusual layouts may reduce initial accuracy. *Mitigation*: fast human correction + one-click retrain.
* Regulatory precision: For compliance-critical fields, enforce higher QA and Golden Set checks.

Typical ZSL workflow in Evolution AI (algorithmic steps)

1. OCR and tokenization with layout coordinates.
2. Compute semantic + positional embeddings for blocks/cells.
3. For each requested field, compute similarity between field descriptor embedding and block embeddings.
4. Propose the best candidate(s) with confidence score.
5. Flag low-confidence proposals for human review.

4. Self-learning, QA workflows & quality improvement

Human-in-the-loop (HITL) mechanics

* Configurable QA pipelines: spot-check, full review, double-signoff, or multi-team workflows.
* Confidence thresholds: fields below the threshold auto-route to annotation.
* Annotation interface: point-and-click selection, edit, comments, and metadata tagging.
* One-click retraining: after QA completes, corrections are fed back and the model updates (self-learning).
* Golden Set: curated, high-quality pre-annotated docs used to validate model updates and train new annotators.

Observed learning curve (system example)

|  |  |
| --- | --- |
| Documents processed | Typical field-level accuracy (example) |
| 0–25 | ~70–90% (varies by document complexity) |
| ~25 | ≈90% |
| 200 | ≈98% |

*Note*: These figures are empirical and depend on document heterogeneity and the strictness of QA policies.

5. Document processing pipeline — technical detail

Pre-processing

* OCR (text extraction) with confidence per zone.
* Image cleanup: deskew, binarization, contrast enhancement.

Layout analysis

* Page segmentation into blocks: headers, footers, paragraphs, tables.
* Table detection using bounding-box clustering and visual cues (lines, whitespace).

Extraction

* Key-value extraction: pair detection using proximity and semantic match.
* Table parsing: mapping cells to header columns, handling merged cells and multi-row headers.
* Paragraph extraction: entity recognition inside paragraphs (e.g., dates, currency amounts).

Post-processing & validations

* Date normalization to ISO-8601.
* Currency splitting: currency code + numeric value + multiplier.
* Summation checks (row/column totals).
* Time-series anomaly detection: rolling-delta checks, z-score thresholds, domain rules.

Output

* Structured JSON (sample):

{

"document\_id": "doc-123",

"pages": [

{"page": 1, "extracted\_fields": [{"field":"Total Revenue","value": 12345678, "currency":"USD","confidence":0.97, "source\_page":1}]}

],

"output\_formats": ["csv","excel","json"]

}

6. Complex table extraction — techniques and edge cases

Challenges

* Tables spanning pages, nested tables, multi-line headers, merged cells, footnotes, and rotated tables.

Techniques

* Cell clustering by proximity and text alignment.
* Header propagation: detect header rows and propagate keys down columns using heuristics + model predictions.
* Span resolution: infer cell spans across rows/columns with bounding box overlaps.
* Context-aware header matching: use column header embeddings to match ambiguous numeric columns to field names.

7. Anomaly detection & business rules

Time-series checks

* Rolling-window % change, z-score detection, and rule-based thresholds (e.g., revenue change > X% flags an anomaly).

Summation consistency

* Recompute sums and compare to reported totals. Small tolerance windows or strict equality depending on configuration.

Custom business logic

* Users may define rules (e.g., if multiplier = "millions" then divide values by 1,000,000) and conditional validations.

8. Integration, API examples & export

Integration channels

* REST API for real-time ingestion and retrieval.
* Bulk SFTP for batch processing.
* Workato connector and webhooks for workflow automation.

Sample API endpoints (illustrative)

* POST /api/v1/documents — upload document (multipart or s3 link).
* GET /api/v1/documents/{id}/status — processing status.
* GET /api/v1/documents/{id}/output?format=json — download structured output.

Delivery

* Output attachments or push to downstream systems (ERP, data lake, BI) with mapping templates.

9. Security, governance & deployment

* ISO/IEC 27001 aligned practices and certification.
* Audit trail for every data point (source page link, user who modified it, timestamp).
* Role-based permissions: annotator, reviewer, admin.
* Encryption in transit (TLS) and at rest.
* Deployment models: cloud (SaaS), private cloud, or on-premise (for data residency/compliance).