AI Detection Report for Agentic Pipeline for Company Onboarding Built an a...

Aug 17, 2026 | 1,915 Words | Pangram 4.0

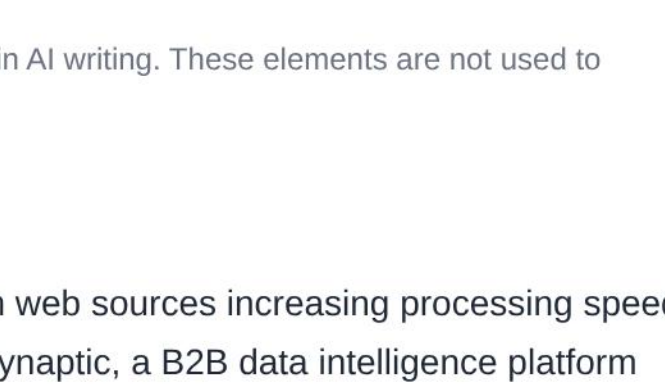
Summary

We believe that this entire text is human-written.

Authorship Breakdown

0 %	of the document	AI Generated
0 %	of the document	Moderately AI Assisted
0 %	of the document	Lightly AI Assisted
100 %	of the document	Human Written

Authorship Progression



Analyzed Text

Words that are underlined indicate AI phrases, while the numbers represent how frequently they are likely to appear in AI writing. These elements are not used to determine the results but are displayed separately as patterns commonly found in AI-generated text.

Human Written | 1491 Words | High Confidence

Agentic Pipeline for Company Onboarding Built an agentic pipeline for company onboarding from web sources increasing processing speed by 30x and saving $400,000 in annual operating costs (eq. to 960 manual hrs/week) Company Synaptic, a B2B data intelligence platform that private investment firms use to discover, evaluate and track promising startups Timeline Dec 2024 – Sep 2025 My Role Product Owner: conceptualized the pipeline, prototyped core functions, set up validation methods, led design and delivery Team 1 Backend engineer, 1 Data engineer, 4 analysts Key Outcome Onboarded 3.6M companies in 30 days: a process that would have taken 3+ years under the previous system. Saved $400K in Annual OpEx. AI Usage Disclosure: I used Claude as an assistant to iteratively critique and suggest edits to drafts of this case study fully authored by me. I have manually edited the final version as you find it below, and I take full responsibility for the entirety of content published here. The Problem Synaptic maintains a database of companies that VCs might invest in. These primarily include startups across stages operating in high-growth spaces like Consumer Tech, Enterprise Tech, AI, Crypto etc. Keeping this database comprehensive and current is foundational to the product's value. The existing onboarding process relied on identifying companies from private market databases (Crunchbase, Pitchbook) and customer requests, then matching them against scraped social media (e.g. LinkedIn, X), marketplaces (e.g. App Store, Play Store) and 3rd-party websites (e.g. Glassdoor) using a rule-based system. A team of analysts manually reviewed the output for errors: fake companies, false

Pangram.

Human Written | 1491 Words | High Confidence

associations, duplicates, mismatches. These errors were generally persistent and frequent. This process had been refined over 5 years and worked reliably well, onboarding 30K–100K companies per month at peak. But after benchmarking against competitors, we estimated Synaptic needed a minimum baseline of 5 million companies. We were at 1.4 million. At 100K per month, closing that gap would've taken us 36 months – by which time we would've been out of the race. Constraints Apart from the sheer technical problem, there were other constraints at play. A lot of people recognised that the existing process was slow and not future-proof, but nobody had actually found a way to use AI to solve this problem. In addition, there was also plenty of skepticism around deploying AI systems since they were felt to not be "well-grounded" and suffered from "frequent hallucinations"; problems that a manual process, though slower, was said to overcome. Since our existing customers valued high quality manually-vetted data from Synaptic, any proposed radical changes to the current approach were met with friction and plain organizational inertia. Add to all of these problems, till December 2024, nobody in the organization had demonstrated any way of using LLMs to solve this problem. The Solution & Key Decisions Realization In December 2024, OpenAI rolled out web search on ChatGPT for all free users, which was when I first tried it out. Now instead of replying from its training corpus, ChatGPT could access real-time information from the web and give an up-to-date accurate answer. I asked it for information on what some very newly founded companies did and it was able to give very accurate answers to those, using information sourced directly from the company homepage. I immediately realized how useful this could be for our use case: we could use ChatGPT web search to gather information on companies much faster. But OpenAI's web search was initially not available via API when we first started. And when it did become available, it was priced around $25/search, unlike token-based pricing for simple APIs without tool calls which was much cheaper. Without web search, this process wouldn't work: the GPT API only returned whatever information was in its pretraining set or more often than not simply hallucinated information. So we figured we'd build a grounded dataset by scraping company websites ourselves. Then we'd use an LLM only to summarize provided information as needed. Our problem became structured into discrete steps: identify a large set of companies for which websites are available scrape these company websites use an LLM to extract information from the scrapes Identifying a source of data The old pipeline pulled from private market databases (like Crunchbase and Pitchbook) and customer uploads, then tried to enrich profiles by matching against scraped third-party profiles. I flipped this around: start with the largest possible universe of companies and work inward. We evaluated several data sources. OpenCorporates had 200M+ companies, but most were too small or not relevant for VCs. Another major provider had 50M+ companies but most of those had very sparse information, were duplicates or had missing websites – and therefore were totally useless for us. Finally, we narrowed down onto LinkedIn. It offered enough coverage of companies across all sizes and stages. Many of these profiles were directly controlled by companies, thus qualifying as a primary source of information. And a lot of these profiles did have multiple points of information, including websites. We discovered a vendor who could provide scraped LinkedIn company profiles at scale, which gave us a starting universe of 40M companies. Scraping company websites The final pipeline wasn't built in one go. It was iteratively refined and validated on progressively larger batches till it was found to work. First, we had to solve the preliminary failure point: invalid websites. This was done using a combination of checks – domain syntax validation, WHOIS lookups, DNS resolution, HTTP response verification, content format checks, and parked website detection. The validation pipeline alone eliminated 25M invalid entries before any expensive LLM processing happened. Then came the actual grunt work. We first scraped 100 websites,

Pangram.

Human Written | 1491 Words | High Confidence

then 1000, then larger and larger randomly sampled batches, treating each round as a study of how things might break. Each batch surfaced a different kind of failure: Cloudflare blocks, untranslated foreign-language content, unpredictable layouts, personal blogs, spammy websites and so on. For each failure mode, we put rules and checks in place, repeatedly tested it, then moved on to another one. And this way, our pipeline became robust to a lot more types of errors than we ever anticipated when starting out. Two interesting decisions came out of this approach: Since the information architecture of a company website could be very unpredictable, our strategy relied on going breadth-first in order to cover all major categories of pages (homepage, products, services, pricing, team, about us, privacy policy, legal, terms and conditions etc.) in a predetermined priority order. We imposed a limit of 25 pages, empirically set — it gathered most of the useful information on a company website while minimizing pointless fluff (blogs, resources, marketing fluff etc.). The information was captured in both the original language and English (in case of foreign language websites), stored as raw HTML and screenshots. Surprisingly, we stumbled upon the usefulness of screenshots via a lucky accident — we found that screenshots consume far fewer input tokens compared to a full-page raw HTML and the output extracted from it is just as useful, sometimes more so. This also had the added benefit of better identifying parked, spammy, gambling or personal websites. Thus the final scraping pipeline was deliberately designed for coverage, cost efficiency and minimising errors and gaps in information. Building a useful summary Instead of extracting information field-wise from the scrape, we inserted an intermediate step: a pair of LLM-generated factsheets that everything downstream could access — a comprehensive version and a compact version. These factsheets became a powerful compressed representation from which firmographic data points were extracted. The compact factsheet (<256 tokens in length) was also used for the company search tool (ColBERT-based retrieval) being built in parallel (described in another case study). These factsheets were used to define new fields outlining the business overview of the company — Industry, Offering Domain, Products/Services, Product Type, Business Model, Customer Segment, Revenue Model, Distribution Channel and Core Technologies. Company names and logos were sourced from the website scrapes directly. And the LinkedIn profile was used to get details like HQ Location, Founding Year, Employee Count etc. Rule-based matching on Crunchbase gave us funding details. With this, complete company profiles were built for a total of 5M companies, using information gathered from a range of primary to tertiary sources. Technical Context In the final implementation: For the full scale image-based content classification, factsheet generation and business overview fields, gemini-2.5-flash-lite was used For tasks requiring LLM judgement or synthetic data generation, gemini-2.5-flash was used The factsheets were generated using a multi-agent system accessing company website screenshots and raw HTMLs in an architecture similar to Magentic-One. The prompts were manually optimized. The business overview fields extracted from factsheets relied on GEPA-optimized prompts and a manually generated dataset of these fields for 12K companies Outcomes Scale: Onboarded 3.6M companies in 30 days (Aug–Sep 2025), compared to the previous rate of 30K–100K per month Speed: 30x increase in processing throughput Cost: $400K in annual operating cost savings, equivalent to eliminating 960 manual hours per week put in by a team of trained analysts

Pangram.

Human Written | 412 Words | High Confidence

Quality: Factsheets achieved ~96%+ data accuracy (human verified on random samples). By extension, all fields downstream of factsheets ensured this level of data quality. The manual process was at a similar level although with a much lower fill rate (it wasn't 100% because slower speeds contributed to frequent issues of stale data). Customer impact: 2/3rd drop in missing company feedback by customers, since most companies likely to be requested were already onboarded. Explicit bulk requests for website-sourced factsheets and new classification fields. Secondary product impact: Improvement in search quality by increasing recall. Team impact: The original analyst team continued operating alongside the new pipeline for several months. However, once the pipeline and downstream benefits were validated at scale, the team size was reduced from 32 to 8. The remaining team moved to higher-value work: data quality, evals, prompts and helping set up new data pipelines. Learnings Ground your data Turns out, if you build your own dataset from primary sources, most of the instances of hallucination go away, even with the smaller LLMs. Every piece of extracted information traced back to some page on the company website. Even today — with larger models, better world knowledge and greater context lengths — the design choice that has compounding effects downstream is controlling the flow of data to the model, not better models. Build a simple system first We didn't plan the entirety of the pipeline starting out. We first had a very simple system (which worked, even if badly), that we empirically tested on progressively larger sets, gathering more and more information, and gradually adding complexity only where it was needed. I'm a big fan of Gall's Law and got to truly apply it in this project. Look for compressed representations In hindsight, the website screenshot decision looks obvious. The full HTML has a lot of information (markup, styling, scripts etc) not directly read by a human but interpreted by a machine; a screenshot is closer to one channel via which humans perceive information — visually. In terms of input tokens for a model, a full webpage might have ~100k tokens, but the same webpage screenshot might have ~5k tokens. A screenshot loses out on a lot of information in the HTML that doesn't matter for firmographic extraction, but preserves what is seen by a human and matters. High signal. Therefore in terms of signal-per-token, the screenshot wins. Since image and text input tokens are similarly priced, the same ratio shows up in cost as well.