Facts Before Phrasing:
A Fact-Grounded, Version-Attributed Pipeline for AI-Assisted Job Applications

Luis Dehlwes[a,*] Claude Code[b]

[a] *Dehlwes Labs* [b] *Anthropic*

∗ Corresponding author: contact@dehlwes.net

Abstract

Tailoring a curriculum vitae and cover letter to every position is expected, yet the common workflow forks a fresh document copy per application, so facts drift apart and no copy stays authoritative. Large language models (LLMs) now draft such documents fluently, but they hallucinate, and their edits leave no trace of who wrote what. We present **Dossier**, a self-hosted pipeline built on a single thesis: career **facts live once, in a relational database**, and every document is a disposable **projection** of that database. Phrasings are stored apart from facts as **versioned, bilingual texts, each attributed** to the human or the AI author; an LLM collaborates through schema-validated tools of the Model Context Protocol and can rephrase, translate and propose, but **cannot invent a fact**, because documents render exclusively from structured data. Per-application tailoring is **selection, not mutation**: include/exclude decisions and position-scoped phrasings overlay an untouched fact base, and a queued LaTeX render farm turns the result into versioned, content-hashed PDFs. We describe the design methodology, its implementation as two cooperating services, and a blinded pilot evaluation over three real satellite-industry openings: with no context the model fabricated **78%** of its biographical claims, while generation from the fact base eliminated fabrication entirely and the tailored, style-contracted condition was ranked best overall. We position the system against resume builders, document standards and freeform LLM drafting.

Index Terms: document generation, large language models, human-AI collaboration, provenance, single source of truth, job applications, Model Context Protocol.

I. Introduction

A job application is a bespoke document pair: a curriculum vitae (CV) and a cover letter, both expected to speak to one specific position. The dominant workflow satisfies this expectation by *copying*: a Word or LaTeX file is duplicated per application and edited in place [1]. Every copy is a fork of the applicant's biography. Dates get fixed in one copy and not another, a rewritten bullet exists only in the fork it was written in, and after a dozen applications no single document can be trusted as the authoritative record. Dedicated resume builders [2], [3] and platform exports [4], [5] polish the editing experience but keep the same document-centric model; structured approaches such as JSON Resume [6] separate data from layout yet stop short of per-application tailoring and multilingual, versioned prose.

LLMs have changed who writes these documents. Modern assistants draft and polish application prose fluently, and human-AI co-writing is an active research area [7], [8]. But two properties make naive LLM drafting a poor fit for exactly this document class. First, models *hallucinate* [9]: a generated CV bullet can assert a certification or an employment date that never existed, and in an application context such an invention is not a stylistic flaw but a misrepresentation with real consequences. Second, a chat transcript leaves no durable record of *which* sentences the model wrote and which the human wrote or approved, exactly where accountability matters most. Meanwhile, the receiving side is itself algorithmic: applicant tracking systems parse and filter submitted documents [11], [12], which rewards consistent, machine-readable structure over ad-hoc formatting.

We argue these problems share one root: treating the *document* as the unit of truth. Dossier inverts this. Career facts (education, employment, projects, skills) live once, as rows in a relational database; phrasings live beside them as versioned, bilingual texts attributed to their author; and a CV or cover letter is a *projection*: a selection of facts, joined with current phrasings, rendered by a template. The LLM participates through schema-validated tools and can propose, rephrase and translate, but the rendering path consumes only structured data, so a fact the model did not find in the database cannot appear in a PDF. Grounding is by construction [10], not by prompt discipline.

This paper makes three contributions. (1) A design **methodology** for fact-grounded, human-attributed document pipelines (Section II). (2) The **implementation** of Dossier as two cooperating self-hosted services: a content service owning facts and phrasings, and a queued render farm producing versioned PDFs (Sections III–IV). (3) A blinded **pilot evaluation** on three real openings quantifying how context depth changes fabrication and judged letter quality (Section V). (4) A **positioning** against resume builders, document standards and freeform LLM drafting (Section VI), with an honest discussion

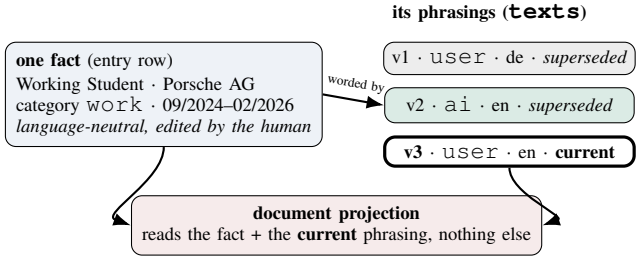

Fig. 1. One fact, many phrasings. A language-neutral entry row is worded by versioned texts, each attributed to `user` or `ai`; a uniqueness invariant keeps exactly one version current per language and scope, and a document projection reads only the fact and its current phrasing.

of limits (Section VII). Dossier is a running personal system, not a controlled study; we report its real characteristics.

II. Methodology and Design Principles

Dossier's design follows five principles. The first two fix the data model, the middle two govern the human-AI collaboration, and the last keeps rendering reproducible.

A. M1: One fact base; documents are projections

All career facts live in one relational database: a profile, organizations, and generic *entries* in eight categories (education, work, project, research, study abroad, certificate, event, achievement), each carrying structured blocks and bullet items plus grouped skills. No document is ever the source of anything: a CV is derived, disposable output, in the lineage of separating content from presentation that document systems established decades ago [17], [18]. Fixing a date once fixes it in every future document.

B. M2: Facts and phrasings on separate axes

Facts are language-neutral rows; *how* a fact is worded is a separate `texts` record: bilingual (German/English), versioned, and scoped either to the base biography or to one specific application. A uniqueness invariant guarantees exactly one *current* version per (target, field, language, scope, position), so every reader, human or machine, resolves the same wording deterministically, while the full history remains queryable.

C. M3: Attributed human-AI co-writing

Every text version records its author: `user` or `ai`, with a timestamp and an optional note. The model may draft a bullet, translate a summary or tighten a paragraph, and the human may accept, edit or supersede it; each step is a new attributed version, never an overwrite. Users of AI writing aids tend to self-declare authorship of model text without feeling ownership of it [16]; explicit stored attribution replaces that ambiguity with a record. The result is a durable audit of the collaboration [7]: for any sentence in any generated document, the system can answer who wrote it, when, and what it replaced (Fig. 1).

D. M4: Grounded generation and tailoring as selection

The LLM reaches the system exclusively through schema-validated tools exposed over the Model Context Protocol [19], whose argument schemas modern constrained decoding can enforce rather than merely request [22], in the tool-use tradition of ReAct and Toolformer [20], [21]: read the profile, list entries, write a text version, set a selection. Two consequences follow. First, *grounding by construction*: the render path consumes only database state, so prose the model never persisted through a validated tool cannot reach a document, and a fact it cannot point to cannot be asserted; this is a stronger guarantee than retrieval-based grounding [10], which still lets the model phrase ungrounded claims. Second, *tailoring is selection, not mutation*: per application, an override layer decides which entries and bullets appear (include, exclude or inherit, each with an optional reason), and position-scoped phrasings may re-word a bullet for that application only. The base biography is never edited to fit a job advertisement, so a dozen tailored applications leave zero drift in the fact base.

E. M5: Deterministic rendering at a distance

Rendering is a separate, queued service. The content service compiles a template payload in which *every* database string is LaTeX-escaped, and submits it to a render farm that typesets versioned templates with `latexmk` [17]; jobs return a PDF, logs, warnings and a page count. Generation happens only through two explicit operations (`generate_cv`, `generate_cover_letter`); merely editing content never triggers a render. Each output is recorded as a content-hashed version bound to its position, so any PDF ever sent to an employer can be reproduced and traced back to the exact data state that produced it.

F. Engineering discipline

Both services are conventional, boring engineering: TypeScript with Zod-validated inputs on every mutation path in the content service, and a small Python worker behind a Redis queue in the render farm. The pipeline is one building block of a personal orchestrator whose gateway registers every tool centrally, so the agent-facing surface is enumerated and complete by policy.

III. Implementation

Dossier consists of two self-hosted services (Fig. 2).

A. The content service

A TypeScript monorepo (about 11,200 lines) with three entry points over one core: a bearer-authenticated HTTP API, a login-protected web application in the style of a small ERP (inline editing, version history, installable as a PWA), and an MCP server exposing **46 tools** to an LLM agent. The database holds **12 tables**; the `texts` table carries the versioned bilingual phrasings with author attribution (M2/M3), and `position_overrides` carries the per-application selection layer (M4). A *position* records the application itself: company, role, language, the job advertisement, a status lifecycle, and

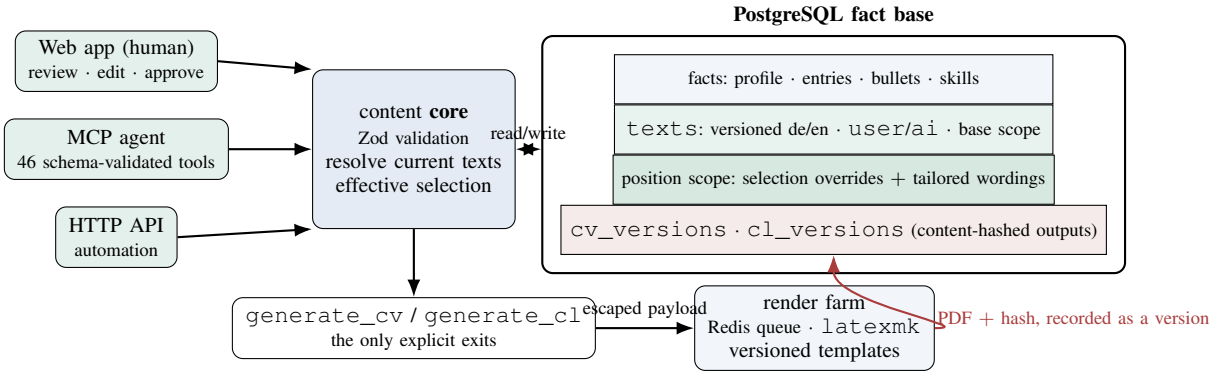

Fig. 2. Dossier around its fact base. Three writers pass one validated core into a database layered from facts over base and position-scoped phrasings down to rendered outputs. Only the two explicit generate operations leave the system; the render farm's PDF returns as a content-hashed version inside the same fact base, so provenance never leaves the database.

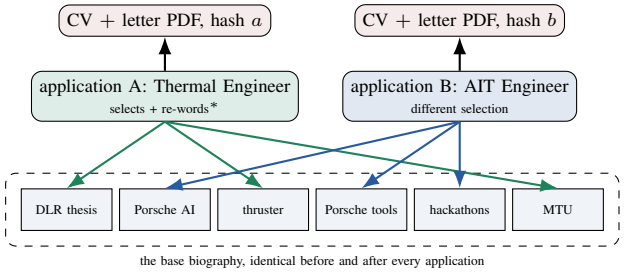

Fig. 3. Tailoring as selection. Two applications project different subsets of the same untouched base biography; *a position-scoped wording may override a bullet for one application only. Each projection is rendered and recorded as a content-hashed PDF.

a target page budget that is passed to the agent as an explicit constraint rather than a hope.

B. The render farm

A small Python service (about 2,200 lines): an HTTP + MCP API in front of a Redis queue and a `latexmk` worker, with three versioned templates (a classic CV, a classic cover letter, and a plain letter). Every request becomes an asynchronous job with status, logs, warnings and page count; long compilations never block a caller. The farm is generic infrastructure: it knows templates and payloads, not careers.

C. The agent surface

The 46 tools mirror the methodology: read tools for the profile, entries and tailoring state; write tools that create attributed text versions rather than overwriting; selection tools for the override layer; and the two explicit generate operations. Because the tools are schema-validated and the render path is data-only, the agent's creative freedom is exactly the freedom the methodology grants: phrasing, translation, selection proposals, never facts.

IV. What Dossier Does

A concrete application proceeds as in Fig. 3. The applicant (or the agent) records the position: company, role, language, the pasted job advertisement, a page budget. The agent reads a compact candidate summary and the current tailoring state, then proposes: which entries and bullets to include or drop for this position, each override optionally carrying a reason, and sharper position-scoped phrasings for the bullets that stay. All of it lands as attributed `ai` versions and pending selections, visible in the web application, where the human reviews, edits or supersedes with `user` versions. On `generate_cv` and `generate_cover_letter` the content service resolves the effective selection, joins current phrasings in the position's language, escapes everything, and submits both jobs to the render farm; the returned PDFs are stored as content-hashed versions on the position, alongside their logs and page counts. The position then moves through its lifecycle (draft, applied, done) and the next application starts from the same, untouched fact base.

V. Pilot Evaluation: Does Context Depth Matter?

The central claim deserves measurement: does grounding an LLM in the fact base actually change the letters it writes? We ran a small, blinded pilot over **three real, concurrently open positions** at a European satellite manufacturer (AIT/automation engineering, thermal engineering, and systems engineering), taken verbatim from the author's live application record.

A. Setup

One LLM (Anthropic's Claude, fixed date) wrote one letter per position under four context conditions, with an otherwise identical prompt (330–430 words, same structure directive):

- **C1, none**: job advertisement only; the model knows the candidate's name and city.
- **C2, paste**: plus a flat one-paragraph CV summary, imitating the common paste-your-CV-into-chat workflow.
- **C3, fact base**: plus the full structured record Dossier serves to an agent: profile, every entry with its bullets, skills.
- **C4, tailored**: the position-scoped view (curated selection and per-position wordings) plus the *writing contract* embedded in the `generate_cl` tool description (grounded

TABLE I
PILOT RESULTS, MEAN OVER THE THREE POSITIONS. RANK: 1 = JUDGED BEST OF THE FOUR LETTERS FOR A POSITION. UNSUP.: CLAIMS NOT SUPPORTED BY THE FACT BASE (TOTAL OVER ALL THREE LETTERS OF THAT CONDITION).

Cond.	Spec.	Grnd.	Auth.	Pers.	Overall	Rank	Unsup.
C1 none	7.3	1.3	3.3	2.0	2.0	4.0	35/45
C2 paste	7.3	5.7	7.0	6.7	6.0	3.0	6/51
C3 fact base	7.7	**9.3**	7.3	7.7	7.7	1.7	**0/52**
C4 tailored	**8.7**	8.0	**8.7**	**8.0**	**8.0**	**1.3**	1/48

paragraphs, no generic enthusiasm or inflated claims, tone samples from two earlier letters to other employers).

The twelve letters were then assessed two ways, per position. A *blind rubric judge* received the four letters with shuffled anonymous labels, the job advertisement and the fact base, and scored specificity, grounding, authenticity and persuasiveness (1–10) plus an overall verdict and a strict ranking. A separate *grounding auditor* extracted every concrete biographical claim from each letter and counted the claims not supported by the fact base. Judges were LLM instances themselves, a practical but imperfect proxy for human preference [23].

B. Results

Table I summarizes. Three observations stand out.

Zero context produces fluent fabrication. The C1 letters read plausibly and engage the job well (specificity 7.3), yet **78%** of their concrete biographical claims (35 of 45) have no basis in the candidate's record, including an invented computer-science degree, invented automation frameworks in production use, and invented tool experience. This is precisely the failure mode the hallucination literature predicts [9], rendered consequential: each such letter would misrepresent the applicant.

A pasted summary helps, but interpolates. C2 cuts unsupported claims to 12% (6 of 51), yet the model still invents connective tissue the summary does not contain: employment durations, department names, project dates. Fabrication shrinks with context, but it does not disappear.

The fact base eliminates fabrication; tailoring wins the decision. C3 produced **zero** unsupported claims across all 52 and the best grounding score (9.3). C4 kept grounding near-perfect while winning everything else: best specificity (8.7), authenticity (8.7), overall (8.0) and mean rank (1.3; ranked first for two of three positions). The single C4 claim flagged by the audit is an artifact of the audit boundary, not an invention: the letter used a position-scoped bullet that exists in the database but was absent from the auditor's base-scope reference. The writing contract carried in the tool description, with its instruction to avoid generic enthusiasm and inflated claims and to match the applicant's established tone, is what separates C4 from C3 in the judges' holistic verdicts.

C. Templates and rendering

The presentation axis was exercised in production rather than in a controlled condition: the render farm resolves versioned templates (`<id>/<version>/` with a JSON payload schema and a `template.tex`) and has typeset the author's real applications, 15 generated cover-letter versions across 17 recorded positions to date, under per-position page budgets, with every output content-hashed against its payload. Adding a new design is adding a template version, not touching any content.

D. Threats to validity

This is a pilot, not a controlled study: three positions, one letter per condition, a single generator model, and judges drawn from the same model family as the generator, a setting with known self-preference and verbosity biases [23]. Scores are LLM judgments, not recruiter decisions, and the biography is the author's own. The fabrication counts, however, are the sturdiest part of the design: they are verified claim-by-claim against the fact base, and the C1 result (78% unsupported) is far outside anything judge bias could plausibly manufacture.

VI. RELATED WORK AND COMPARISON

Resume tooling. Commercial builders such as Novoresume [2] and open-source builders such as Reactive Resume [3] improve editing ergonomics; LinkedIn [4] and Europass [5] export profiles into fixed layouts; JSON Resume [6] defines an open schema with themed rendering; Overleaf [1] hosts the manual LaTeX workflow. Table II positions Dossier: it shares the data-before-layout stance of JSON Resume and the self-hosting of Reactive Resume, and adds the three properties none of them target: versioned bilingual phrasings with author attribution, fact-grounded AI collaboration, and per-application tailoring as a first-class overlay.

Human-AI writing and grounding. Co-writing systems study how humans and models share a draft [7], [8]; hallucination surveys catalogue why model prose cannot be trusted as fact [9]; retrieval augmentation grounds generation in retrieved evidence [10]. Dossier's contribution is architectural rather than model-side: the document pipeline itself refuses ungrounded content, and attribution is stored with the text, not in a transcript.

The receiving side. Algorithmic hiring tools screen what applicants submit, with documented fairness concerns [11], [12]; person-job-fit models read free-text experience against stated requirements [13], [14], and static one-size documents are known to filter out viable candidates at scale [15]. Dossier operates on the candidate side and does not interact with such systems, but its structured, consistent output is a better citizen of that parsing pipeline than hand-forked documents, and its honesty property (no invented facts) is a small, structural answer to a real integrity risk of LLM-drafted applications.

VII. DISCUSSION

Where it helps. The fact base ends document drift: one correction propagates to every future application, and the

TABLE II
QUALITATIVE POSITIONING OF DOSSIER AMONG APPLICATION-DOCUMENT TOOLING (✓ = YES, ∼ = PARTIAL, × = NO).

System	Single fact base	Versioned prose	Attributed AI writing	Fact-grounded	Per-application tailoring	Self-hosted	Primary niche
Word / Overleaf [1]	×	∼	×	×	∼	∼	manual documents
Novoresume [2]	×	×	∼	×	∼	×	guided builder
Reactive Resume [3]	∼	×	∼	×	∼	✓	open-source builder
LinkedIn export [4]	∼	×	×	×	×	×	profile export
Europass [5]	∼	×	×	×	×	×	standardized EU CV
JSON Resume [6]	✓	×	×	×	×	✓	open schema + themes
Freeform LLM chat	×	×	×	×	✓	×	ad-hoc drafting
Dossier (this work)	✓	✓	✓	✓	✓	✓	fact-grounded pipeline

record of what was claimed where is complete. Attribution turns AI assistance from an anonymous rewrite into an auditable collaboration. Selection-based tailoring makes the honest version of "adapting your CV" cheap: emphasize and omit, never bend facts. And because outputs are versioned and content-hashed, any sent PDF is reproducible years later.

Characteristics of the running system. Dossier is in personal production use. The content service spans 12 tables and 46 agent tools over roughly 11,200 lines of TypeScript; the render farm adds roughly 2,200 lines of Python, three versioned templates and a queued worker; texts exist in two languages under an exactly-one-current-version invariant. These are properties of a working implementation, not results of a controlled study.

Limitations and ethics. The system is deliberately single-user; it manages one person's biography, not a multi-tenant product. Only German and English are modeled. We report no user study and no measurements of application outcomes. Ethically, the system hardens honesty in one direction only: it prevents *invented* facts in generated documents, but it cannot verify that the facts a human enters are true, and selective emphasis remains a human judgement. We consider AI-assisted phrasing legitimate exactly because the facts are the applicant's own and every AI sentence is attributed and human-reviewable before anything is generated; applicants remain responsible for complying with any disclosure expectations a given employer or jurisdiction imposes.

VIII. CONCLUSION

Dossier treats a job application not as a document to be written but as a projection to be computed: facts once, phrasings versioned and attributed, tailoring as selection, rendering deterministic. Under that inversion, an LLM becomes a safe collaborator by construction: free where language lives, powerless where facts live. The system runs, in daily personal use, on two small self-hosted services. Future work includes richer templates, more languages, an import path from existing documents, and a study of whether fact-grounded tailoring measurably changes application outcomes.

REFERENCES

[1] Overleaf (Digital Science), "The home of scientific and technical writing," accessed Jul. 2026. [Online]. Available: https://www.overleaf.com

[2] Novoresume, "The Best Online Resume Builder," accessed Jul. 2026. [Online]. Available: https://novoresume.com

[3] A. Pillai, "Reactive Resume: a free and open-source resume builder," accessed Jul. 2026. [Online]. Available: https://rxresu.me

[4] LinkedIn, "Save a profile as a PDF," LinkedIn Help, accessed Jul. 2026. [Online]. Available: https://www.linkedin.com/help/linkedin/answer/a541960

[5] European Union, "Europass CV builder," accessed Jul. 2026. [Online]. Available: https://europass.europa.eu/en/create-europass-cv

[6] JSON Resume, "An open-source initiative to create a JSON-based standard for resumes," accessed Jul. 2026. [Online]. Available: https://jsonresume.org

[7] M. Lee, P. Liang, and Q. Yang, "CoAuthor: Designing a Human-AI Collaborative Writing Dataset for Exploring Language Model Capabilities," in *Proc. ACM CHI*, 2022.

[8] A. Yuan, A. Coenen, E. Reif, and D. Ippolito, "Wordcraft: Story Writing With Large Language Models," in *Proc. ACM IUI*, 2022.

[9] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, A. Madotto, and P. Fung, "Survey of Hallucination in Natural Language Generation," *ACM Computing Surveys*, vol. 55, no. 12, art. 248, 2023.

[10] P. Lewis *et al.*, "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in *Proc. NeurIPS*, 2020.

[11] M. Raghavan, S. Barocas, J. Kleinberg, and K. Levy, "Mitigating Bias in Algorithmic Hiring: Evaluating Claims and Practices," in *Proc. ACM FAT**, 2020, pp. 469–481.

[12] A. Fabris, N. Baranowska, M. J. Dennis, D. Graus, P. Hacker, J. Saldivar, F. J. Zuiderveen Borgesius, and A. J. Biega, "Fairness and Bias in Algorithmic Hiring: A Multidisciplinary Survey," *ACM Trans. Intelligent Systems and Technology*, vol. 16, no. 1, 2025.

[13] C. Qin, H. Zhu, T. Xu, C. Zhu, L. Jiang, E. Chen, and H. Xiong, "Enhancing Person-Job Fit for Talent Recruitment: An Ability-aware Neural Network Approach," in *Proc. ACM SIGIR*, 2018, pp. 25–34.

[14] Y. Mashayekhi, N. Li, B. Kang, J. Lijffijt, and T. De Bie, "A Challenge-based Survey of E-recruitment Recommendation Systems," *ACM Computing Surveys*, vol. 56, no. 10, 2024.

[15] J. B. Fuller, M. Raman, E. Sage-Gavin, and K. Hines, "Hidden Workers: Untapped Talent," Harvard Business School Project on Managing the Future of Work, white paper, 2021.

[16] F. Draxler, A. Werner, F. Lehmann, M. Hoppe, A. Schmidt, D. Buschek, and R. Welsch, "The AI Ghostwriter Effect: When Users do not Perceive Ownership of AI-Generated Text but Self-Declare as Authors," *ACM Trans. Computer-Human Interaction*, vol. 31, no. 2, 2024.

[17] L. Lamport, *LaTeX: A Document Preparation System*, 2nd ed. Addison-Wesley, 1994.

[18] B. K. Reid, "Scribe: A Document Specification Language and its Compiler," Ph.D. dissertation, Carnegie-Mellon Univ., Tech. Rep. CMU-CS-81-100, 1980.

[19] Anthropic, "Introducing the Model Context Protocol," Nov. 2024. [Online]. Available: https://www.anthropic.com/news/model-context-protocol

[20] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, "ReAct: Synergizing Reasoning and Acting in Language Models," in *Proc. ICLR*, 2023. arXiv:2210.03629.
[21] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language Models Can Teach Themselves to Use Tools," in *Proc. NeurIPS*, 2023. arXiv:2302.04761.
[22] B. T. Willard and R. Louf, "Efficient Guided Generation for Large Language Models," arXiv:2307.09702, 2023.
[23] L. Zheng *et al.*, "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena," in *Proc. NeurIPS Datasets and Benchmarks*, 2023. arXiv:2306.05685.