Starting a PhD in Machine Learning or NLP

A Practical Starter Guide

Venues, deadlines, tools, and the habits that compound

Konstantinos Skianis

Assistant Professor of Natural Language Processing
Department of Computer Science and Engineering
University of Ioannina

https://y3nk0.github.io

Everything here is operational: what to set up in your first week, which venues your work will target, how to find and judge related work, how to keep a library that never loses a reference, how to use an AI assistant without damaging your scholarship, and how to put a durable identity on the web. Treat it as a checklist to work through, not an essay to read once.

October 2026

Deadlines, prices, and tool features change. Verify dates at the source before you plan around them.

Contents

1 How to use this guide **2**
1.1 The first 90 days, condensed . . . 2

2 The publication landscape **2**
2.1 Core machine learning venues . . . 2
2.2 Core NLP venues and the ACL Rolling Review . . . 2
2.3 Adjacent venues and journals . . . 3
2.4 How to judge a venue without asking around . . . 3

3 Deadlines: never be surprised again **4**

4 Writing and sharing: LaTeX and Overleaf **5**
4.1 Overleaf — the default collaborative editor . . . 5
4.2 Prism — closing on 1 January 2027 . . . 5
4.3 The local workflow you should also have . . . 6
4.4 Preprints, anonymity, and sharing . . . 6

5 Your reference library: Zotero **7**
5.1 The setup, in order . . . 7
5.2 Getting a reference into your `.bib` . . . 7
5.3 Habits that keep a library trustworthy . . . 8

6 Your first real move: a focused survey **8**
6.1 A protocol you can run in 8–12 weeks . . . 9

7 Searching the literature properly **10**
7.1 Google Scholar: the operators and features that matter . . . 10
7.2 The rest of the toolkit . . . 11

8 Reading the signals: citations, stars, and attention **12**
8.1 Citations: measure velocity, not volume . . . 12
8.2 GitHub: stars, and the signals that matter more . . . 12

9 Set up an AI assistant — and the rules for using it **13**
9.1 The setup . . . 13
9.2 What to use it for . . . 14
9.3 Non-negotiable guardrails . . . 14

10 Your personal site: `username.github.io` **14**
10.1 How GitHub Pages works . . . 15
10.2 Use the AcademicPages template . . . 15

11 Research infrastructure and habits **16**
11.1 Make experiments reproducible from day one . . . 16
11.2 Habits that compound . . . 16

12 Checklists **16**

A Link directory **17**

B A minimal paper repository **19**

1 How to use this guide

A PhD is won on compounding habits, not on heroic sprints. The first three months decide how much friction every later month carries: whether you can find a paper you read in March when you need it in November, whether your experiments are reproducible, whether your name is findable, whether a deadline surprises you.

Work through this document in order the first time. Section 1.1 is the short version.

1.1 The first 90 days, condensed

Week 1 — infrastructure

- Install **Zotero** + the browser connector + the **Better BibTeX** plugin (§5).
- Create a **Google Scholar** profile and set up three alerts (§7).
- Create accounts: **Overleaf**, **OpenReview**, **GitHub**, **Hugging Face**, **arXiv**, **ORCID** (§4).
- Bookmark the AI deadlines countdown and subscribe the relevant deadlines to your calendar (§3).
- Set up your AI assistant with a project memory file (§9).
- Create one git repository per project, with a `notes/` lab notebook from day one (§11).

Weeks 2–10 — a focused survey

Pick a narrow question with your advisor and run a small systematic survey (§6). Output: a 60–120 paper annotated bibliography, an extraction spreadsheet, a taxonomy figure, 10–15 pages of writing, and a shortlist of three gaps that could become your first paper. This is the single highest-return thing you can do in your first term.

Weeks 10–13 — identity and first experiments

- Publish `username.github.io` with the AcademicPages template (§10).
- Reproduce one baseline from the survey end to end, on your own hardware, with tracked runs (§11).
- Pick a target venue and work backwards from its deadline (§3).

2 The publication landscape

ML and NLP are *conference-first* fields. Unlike most of science, the top-tier results appear at conferences with full peer review and archival proceedings; journals play a secondary (though rising) role. Your CV will be read as a list of venue acronyms, so learn them early.

Three things to internalise:

1. **Cycles, not rolling submissions.** Most venues have one or two hard deadlines per year. Miss by an hour and you wait months.
2. **Reviewing is public-ish.** ICLR and many workshops review on OpenReview, where reviews and rebuttals stay online. Read them — they are the best free writing lesson available.
3. **A paper has a lifecycle:** preprint → submission → reviews → rebuttal → decision → camera-ready → talk/poster → citations. Plan for 6–12 months from idea to published.

2.1 Core machine learning venues

2.2 Core NLP venues and the ACL Rolling Review

ARR: the mechanic that confuses every first-year NLP student

The ACL family separates *reviewing* from *publishing*. You submit to **ACL Rolling Review** (ARR) in one of its cycles — every two to three months in 2026, not monthly; check the current calendar. You receive reviews plus a meta-review. Then, in a separate step, you *commit* that reviewed paper to a specific venue (ACL, EMNLP, NAACL, . . .), whose chairs decide accept / Findings / reject based on those reviews. Consequences worth planning around:

Venue	Scope	Rhythm (verify yearly)
NeurIPS	Broadest ML venue: theory, deep learning, datasets & benchmarks track, many workshops	Abstract ∼May, full ∼May; conference December
ICML	Core ML, strong on methods and theory	Deadline ∼Jan–Feb; conference July
ICLR	Representation learning, deep learning, LLMs; fully open review on OpenReview	Deadline ∼Sept; conference April–May
AISTATS	Statistics × ML, principled methods	Deadline ∼Oct; conference April–May
COLT / UAI	Learning theory / probabilistic and causal reasoning	Winter–spring deadlines
AAAI / IJCAI	Broad AI, including applications and multi-agent	AAAI ∼Aug; IJCAI ∼Jan
COLM	Dedicated to language modelling: data, training, evaluation, science of LMs	Deadline ∼March; conference October
MLSys / LoG	Systems for ML / learning on graphs	Autumn deadlines

Dates are indicative patterns, not promises. Always confirm on the venue site and the deadline tracker (§3).

- There are two deadlines to track: the ARR *submission* deadline and the venue *commitment* deadline.
- Reviews are reusable — a rejected commitment can be revised and committed elsewhere, or resubmitted to ARR with a revision note.
- Mandatory sections: a **Limitations** section (does not count against the page limit) and the **Responsible NLP checklist**. Missing limitations is a desk-reject risk.

2.3 Adjacent venues and journals

- **Data mining / web / IR:** KDD, WSDM, CIKM, SIGIR, The Web Conference, ECML-PKDD. Natural homes for applied and graph work.
- **Vision:** CVPR, ICCV, ECCV — relevant for multimodal work.
- **Journals:** JMLR (open access, strong), **TMLR** (rolling, fast, judged on correctness + interest rather than novelty hype — an excellent fit for careful work that conference reviewers undervalue), TACL, *Computational Linguistics*, TPAMI.
- **Workshops:** co-located, non-archival or lightly archival, 4–8 pages, fast turnaround, real feedback and real audiences (BlackboxNLP, RepL4NLP, WMT, and dozens more each year). The honest advice for a first-year student: *aim your first submission at a good workshop.* You learn the full pipeline — writing, reviewing, rebuttal, poster, conversations — at a fraction of the risk.

2.4 How to judge a venue without asking around

- **Google Scholar Metrics** — per-venue h5-index (https://scholar.google.com/citations?view_op=top_venues).
- **CORE ranking** (https://portal.core.edu.au/conf-ranks/) — A*/A/B/C tiers; used for evaluation in many countries.
- **CSRankings** (https://csrankings.org) — shows which venues count as top-tier per subarea, which is the useful signal.
- **Where does the work you admire get published?** The most reliable test. Open the last three papers you loved and look at the venue line.

Venue	Scope	Notes
ACL	The flagship: all of computational linguistics and NLP	Summer conference; via ARR commitment
EMNLP	Empirical methods — the other flagship	Late-year conference; via ARR
NAACL / EACL / AACL	Regional ACL conferences (Americas, Europe, Asia-Pacific); same standard	Alternate by year
COLING / LREC	Computational linguistics / language resources and evaluation	Historically biennial; check the current cycle
CoNLL / *SEM / SemEval	Learning and language, semantics, shared tasks	Good first-paper targets
Interspeech / ICASSP	Speech and audio	Speech-specific cycles
TACL	Journal of the ACL — rolling submission, presented at *ACL conferences	Slower but prestigious
Findings	Companion archival track of *ACL venues for solid papers below the main-conference bar	A real, citable publication

Predatory venues

If a conference emails you an invitation, promises acceptance in two weeks, charges a large fee, and spans "AI, Big Data, Blockchain and IoT" — it is a scam. Publishing there actively damages your CV. Cross-check every unfamiliar venue against the sources above before submitting.

3 Deadlines: never be surprised again

The one page to bookmark

Conference deadlines are countdowns, not dates: they are stated in **AoE** (Anywhere on Earth = UTC−12), so an "11:59pm" deadline is later than you think — and a venue has far more dates than the one everybody remembers. Two boards cover this, and they complement each other.

- **Papers with Code — AI Deadlines** (start here):
  https://paperswithcode.co/ai-deadlines
  Filter by subfield (Machine Learning, NLP, Computer Vision, Data Mining, Robotics, IR, Speech, RL, ...), or restrict to *top tier only*. Crucially it tracks *every* milestone, not just the submission date: abstract registration, reviews released, author-feedback and rebuttal windows, commitment dates, notifications, camera-ready — each with a live countdown, the conference dates, and the location. The same site carries trending papers, a following feed, and an API and MCP endpoint you can wire into your assistant (§9).
- **AI Deadlines**, pre-filtered for your subfields:
  http://aideadlines.org/?sub=ML,CV,NLP
  The `sub=` parameter carries the filter, so the link itself is your personalised board — bookmark it and edit the tags (`ML`, `CV`, `NLP`, `SP`, `DM`, `RO`, ...) as your interests change. A mirror runs as a Hugging Face Space (https://huggingface.co/spaces/huggingface/ai-deadlines), and the data is an open repository of YAML files (https://github.com/huggingface/ai-deadlines) — so a missing deadline is a deadline you can add by pull request, a trivially easy first open-source contribution.

Cross-check both against the official venue page before you plan around a date. Aggregators lag; call-for-papers pages do not.

A workflow that actually holds:

1. Filter the board to your subfields (NLP, ML, maybe IR/CV) and list the 4–6 venues you could plausibly target in the next 18 months.

2. Put each into your calendar *three times*: the deadline itself, minus 7 days ("results frozen, writing only"), and minus 21 days ("experiments must be running").
3. Add the secondary dates people forget: abstract deadlines (NeurIPS), ARR commitment dates, supplementary-material deadlines, rebuttal windows (often a hard 5–7 day window — do not travel then), camera-ready dates, and early-registration dates.
4. Backward-plan on paper: camera-ready ← decision ← rebuttal ← submission ← freeze ← final experiments ← ablations ← main result. Ablations always take longer than the main result.
5. Alternative trackers to cross-check: https://ccfddl.com and the official venue pages. Trust the venue page over any aggregator.

> **Deadline-driven research is a trap, in moderation**
> Deadlines are useful forcing functions and terrible research strategies. Keep one long-horizon question alive alongside the deadline work — it is where your thesis contribution will come from.

4 Writing and sharing: LaTeX and Overleaf

Every paper you write in this field will be LaTeX. Learn it properly once — a weekend — and it will never cost you again. What differs is *where* you edit and *how* you collaborate. (Typst is pleasant and growing, but no venue template accepts it, so papers stay in LaTeX.)

4.1 Overleaf — the default collaborative editor

https://www.overleaf.com

- Browser-based, nothing to install, compiles server-side. Co-authors type in the same document in real time, which is what you want in the 48 hours before a deadline.
- Template gallery contains the official style files for most venues — search "NeurIPS", "ACL", "ICML" before you start a paper and begin from the official template, never from last year's copy.
- Rich comments and a chat panel; useful for advisor feedback.
- Overleaf now ships its own **AI Assistant** inside the editor: LaTeX-aware, and metered by plan — a handful of uses a day on the free tier, more on paid ones, unmetered at the top. The cautions below apply to it word for word.
- Many universities have institutional licences — check before paying. The free tier is limited on collaborator count, compile time, full version history, Git/Dropbox sync, and track changes.
- Share with a link (view or edit), which is how most advisors want to receive a draft.

4.2 Prism — closing on 1 January 2027

> **Do not start anything in Prism**
> OpenAI's AI-native LaTeX workspace (https://prism.openai.com, launched January 2026) **shuts down on 1 January 2027**. It was free and genuinely good — a frontier model that saw the whole manuscript rather than isolated snippets — and it lasted under a year.
>
> - **If you have projects there, export them now.** The site carries an export link; do not leave it to the last week.
> - **What exports do *not* carry:** comments, version history, collaborators and linked integrations. If a co-author's comments matter, copy them out by hand before the date.
> - **Land somewhere boring:** Overleaf for collaboration, or `git` + `latexmk` locally. Both will outlive your PhD.

The transferable lesson

A free cloud tool can disappear with a quarter's notice, and your thesis cannot. Keep every manuscript in **plain `.tex` files under git** that you can compile offline, and treat any editor — Overleaf included — as a convenient window onto that repository rather than as the place your work lives.

Two cautions before you move your thesis into any AI editor

Policy: most venues now have an explicit policy on AI-assisted writing (disclosure requirements, and a hard ban on using LLMs to write *reviews*). You remain fully responsible for every sentence, citation, and number. Read your venue's policy once, now.

Confidentiality: do not upload data, text, or results covered by an NDA, an industry collaboration, or someone else's unpublished work into a cloud tool without checking. This is a real way for PhD students to get into real trouble.

4.3 The local workflow you should also have

Cloud editors are for collaboration; a local copy is for control. Keep both.

- `git` + `latexmk -pdf -pvc` + VS Code with the LaTeX Workshop extension (or your editor's equivalent).
- **One sentence per line** in your `.tex` files. This single convention turns git diffs from useless to invaluable.
- Split by section (`\input{sections/method}`) so two people can work without conflicts.
- Keep figures as generated artefacts: a script in the repo produces every PDF figure from logged results. No hand-edited plots.
- **Textual vs. parenthetical citations.** With natbib (the ACL and NeurIPS templates): `\citet{}` when the authors are part of the sentence ("Devlin et al. (2019) show"), `\citep{}` when the citation is an aside ("(Devlin et al., 2019)"). With biblatex: `\textcite` and `\parencite`. Mixing them up is the most common formatting error in student drafts.
- `latexdiff` (on CTAN) marks up every change between two versions of the source: ideal for a camera-ready, a journal revision, or showing your advisor what moved since last week.
- For arXiv submission, `arxiv-latex-cleaner` (https://github.com/google-research/arxiv-latex-cleaner) strips comments, removes unused files, and shrinks images — run it every time, not least because *reviewers have found deanonymising information in LaTeX comments.*

4.4 Preprints, anonymity, and sharing

- **arXiv** (https://arxiv.org) — cs.CL for NLP, cs.LG for ML, cs.AI, stat.ML. Your first submission needs endorsement; your advisor or a co-author handles it. Preprints are the norm and accelerate citations.
- **Anonymity periods:** ACL-family venues enforce an anonymity window around the review cycle — posting or publicising a new preprint during it can invalidate your submission. ML venues (NeurIPS/ICML/ICLR) are double-blind but generally permit preprints. *Check the current policy for your target venue before you post.*
- For double-blind submissions, anonymise your code too: https://anonymous.4open.science. Also strip author names from PDF metadata and acknowledgements.
- **Reproducibility artefacts:** release code on GitHub with a licence, checkpoints and datasets on Hugging Face, and an exact environment spec. Papers with runnable code get cited more — and your future self is the main beneficiary.
- **Make the code citable.** A `CITATION.cff` file in the repo (https://citation-file-format.github.io) gives GitHub a "Cite this repository" button, and the Zenodo integration (https://zenodo.org) mints a DOI for every tagged release, so the code can be cited independently of the paper.
- Get an **ORCID** (https://orcid.org) in week one and put it on everything; it disambiguates you from everyone who shares your name.

5 Your reference library: Zotero

You will read a few thousand papers during a PhD. Without a system you will re-find the same paper five times and mis-cite it once in print. Set this up before you read paper number ten.

https://www.zotero.org — free, open source, no lock-in.

5.1 The setup, in order

1. Install **Zotero** (desktop) and the **Zotero Connector** browser extension. The connector is the whole point: one click on an arXiv, ACL Anthology, OpenReview, or Scholar page saves the metadata *and* the PDF.
2. Install **Better BibTeX** (https://retorque.re/zotero-better-bibtex/). This is non-optional for LaTeX users. It gives you:
   - stable, readable citation keys (`devlin2019bert`) that you **pin** so they never change under you;
   - **auto-export**: point it at `paper/refs.bib` in your git repo and it rewrites that file whenever your library changes. Your bibliography is then always current and you never hand-edit BibTeX again.
3. Turn on sync (free account; 300 MB of free file storage, paid tiers or your own WebDAV/cloud folder beyond that). Attachments can also be stored in a linked folder you back up yourself.
4. Create a **group library** with your lab or co-authors — shared reading lists, shared annotations, one source of truth per project.
5. Organise with **collections** per project and **tags** per theme (`method:retrieval`, `status:to-read`, `baseline`). Saved searches then do the filing for you.
6. Read inside Zotero: its built-in PDF reader keeps highlights and notes attached to the item, so your annotations are searchable years later.

5.2 Getting a reference into your `.bib`

The easiest route, and it needs no setup

1. In Google Scholar: *Settings* → *Bibliography manager* → tick *Show links to import citations into BibTeX*. One time only.
2. A **Cite** link now appears under every search result. Click it, then **BibTeX**, and copy the entry.
3. Paste it into your `refs.bib` — or, better, into Zotero with *File* → *Import from clipboard*, so your library and your bibliography stay in step.

Scholar also exports in bulk: save papers to *My library* with the star as you find them, then select them and *Export* → *BibTeX*.

Whatever Scholar gives you, check it

Expect `arXiv preprint arXiv:1810.04805` for a paper that was published at NAACL, missing pages and editors, venue names that differ between entries, and mangled accents in author names. Check `booktitle/journal`, year and pages before you trust an entry — a bibliography full of "arXiv preprint" is the clearest signal that nobody looked.

Cleaner sources, each one click away:

- **ACL Anthology** — every paper page has a BibTeX box, and every volume offers one `.bib` for the entire proceedings.
- **DBLP** — per-paper BibTeX, plus a one-click export of everything an author has published.
- **arXiv** — "Export BibTeX citation" on the abstract page (fine for genuinely unpublished work).
- **Semantic Scholar**, the ACM Digital Library, IEEE Xplore and Springer all have a cite or export button.

- **Any DOI, straight from the terminal:**
  `curl -LH "Accept: application/x-bibtex" https://doi.org/<DOI>`
  returns the publisher's own entry — correct venue, pages and year. Ideal in a script, or when a site buries its export button.

Past twenty or so references, stop copy-pasting entirely: the Zotero connector plus Better BibTeX auto-export does all of the above for you and keeps your keys stable.

5.3 Habits that keep a library trustworthy

- **Cite the published version, not the preprint**, once one exists. Reviewers notice; it is a cheap signal of care. Keep both entries if you must, but cite the proceedings.
- **Get clean BibTeX from the right source.** For CS, the ACL Anthology (https://aclanthology.org) and **DBLP** (https://dblp.org) produce correct proceedings entries; Google Scholar's BibTeX is often sloppy about venue names, pages, and editors. Zotero + Add Item by Identifier (DOI / arXiv ID / ISBN) is the fastest clean path.
- Run *Duplicate Items* monthly. Merge, do not delete.
- **Retraction alerts are already on.** Zotero checks your library against Retraction Watch and marks retracted items with a red banner; when you see one, find out why before you cite it.
- **Tidy the file, not the entries.** `bibtex-tidy` (https://github.com/FlamingTempura/bibtex-tidy; there is also a web version) formats, sorts, deduplicates and strips junk fields such as `abstract` and `file`. One command before every submission — or a Makefile target.
- **Audit before you submit.** A reference that does not resolve is a typo or a fabrication, and reviewers now look for both. Wrap the `curl` line above in a loop over every DOI in `refs.bib` (and the arXiv API for `eprint` fields), flag anything that returns no title, and open whatever is flagged. Ten lines of shell or Python; ask your assistant to write them and keep the script in the repo.
- Add one-sentence notes on save: "what it does / why I care". In month 14 this is the difference between a library and a junkyard.
- Subscribe to feeds inside Zotero (arXiv category RSS, journal feeds) so new work arrives where you already work.

Three BibTeX gotchas that survive every tool

Capitals vanish. Most styles lowercase titles, so protect what must stay upper-case: `title = {{BERT}: Pre-training of ...}`, and the same for `{GPT-4}`, `{Greek}`, `{LLM}`.

Author format. `author = {Devlin, Jacob and Chang, Ming-Wei}` — `Last, First`, joined by `and`; never a comma-separated list, never an "et al." typed by hand.

arXiv-only work gets the arXiv fields, not an invented journal: `@misc{..., eprint={2505.24403}, archivePrefix={arXiv}, primaryClass={cs.LG}}`. And pick *one* source per venue family: DBLP's long booktitles next to the Anthology's short ones look careless.

Zotero × Overleaf

Overleaf's paid tiers can link a Zotero library directly. On a free plan, the Better BibTeX auto-export route is better anyway: your `.bib` lives in git, so it is versioned, diffable, and identical for every co-author. Commit it.

6 Your first real move: a focused survey

Before you can contribute, you need to know what exists — and the fastest honest way to learn a field is to survey it systematically. Not reading "around" the topic: producing a structured artefact.

Why this is the right first project

- It gives you the **vocabulary** to talk to your advisor as a peer within weeks.
- It produces a **taxonomy** — and a taxonomy is the thing that reveals gaps. Your first paper usually lives in a box you drew that has nothing in it.
- It builds your **bibliography**, which you will reuse in every paper and in your thesis introduction.
- It identifies the **baselines** you will have to beat, and the datasets and metrics everyone uses, before you waste a month on a non-standard setup.
- It is **publishable**: as a survey, a position paper, or a workshop paper — and it is the natural core of your thesis' related-work chapter.

6.1 A protocol you can run in 8–12 weeks

1. **Write the question down.** One sentence, narrow enough to be answerable: not "LLM reasoning" but "methods for verifying intermediate reasoning steps in LLMs, 2022–2026, and how they are evaluated". Agree it with your advisor in writing.

2. **Seed set.** Ask your advisor for 5–10 anchor papers. Read them properly. Note their shared vocabulary — those terms become your search queries.

3. **Define inclusion / exclusion criteria.** E.g. include: peer-reviewed or arXiv, 2022+, English, proposes or evaluates a verification method. Exclude: pure position pieces, no empirical evaluation. Write the criteria down *before* screening, or you will rationalise.

4. **Search systematically.** Run the same queries across Google Scholar, Semantic Scholar, the ACL Anthology, OpenReview, and DBLP (§7). **Log every query, the date, and the number of hits** in a spreadsheet tab. This is what makes a survey reproducible rather than anecdotal — and it is the discipline the PRISMA standard formalises in other fields.

5. **Snowball in both directions.** *Backward:* mine the reference lists of your seed papers. *Forward:* for each seed, open "Cited by" and search *within* the citing articles. Forward snowballing is how you find the last six months, which no survey covers yet.

6. **Screen in two passes.** Title + abstract first (fast, ruthless, keep a "maybe" pile), then full text for survivors. Expect to screen 400–600 to keep 60–120. Past a few hundred, **ASReview** (https://asreview.nl; free, open source) learns from your first decisions and orders the rest by likely relevance; Rayyan is the lighter, hosted alternative. Neither replaces the criteria you wrote in step 3.

7. **Extract into a table, not prose.** One row per paper, fixed columns (Table 1). Fill it as you read. The table *is* your survey; the prose comes from reading down its columns.

8. **Build the taxonomy.** Cluster rows until the groups stop changing. Draw it as one figure. If you cannot draw it, you do not understand the field yet — which is useful information.

9. **Write 10–15 pages as you go,** not at the end. Each section summarises one branch of the taxonomy and ends with "what is missing here".

10. **Finish with three candidate gaps,** each stated as a testable question plus the experiment that would answer it. Take those to your advisor. That meeting is where your first paper starts.

Read efficiently: the three-pass method

Keshav's *How to Read a Paper* is two pages and will save you months. Pass 1 (5 min): title, abstract, intro, section headings, conclusions, references — decide whether to continue. Pass 2 (1 h): figures, tables, method at a level where you could summarise it. Pass 3 (several hours, rare): re-derive the work, assumption by assumption. Give pass 3 to about 20 papers; pass 1 to hundreds.

Column	What goes in it
Key	Pinned Better BibTeX key (`author2026short`)
Venue / year	Published venue, not "arXiv" if it has one
Problem	The task, in your words, in one line
Method family	Your taxonomy label (fill this last, revise often)
Data	Datasets and sizes used
Metrics	How success is measured; is it comparable across papers?
Headline result	The number they lead with, plus the baseline it beats
Compute	GPUs, hours, model sizes — can *you* run this?
Code	Link + does it reproduce the paper's numbers? (§8)
Limitations	Stated and unstated
Relevance	Your 1–5 score and a one-line "why I care"

Table 1: Extraction schema. A spreadsheet with these columns, 100 rows deep, is worth more than 100 pages of highlighted PDFs.

> **Scope discipline**
>
> A survey expands without limit if you let it. Timebox it, freeze the search date ("literature up to 15 November 2026"), and ship. You can always write version two — and a frozen, documented cut-off is exactly what a reviewer wants to see.

7 Searching the literature properly

Most people use Google Scholar as a search box. It is closer to a research instrument, and the difference shows up in what you find.

7.1 Google Scholar: the operators and features that matter

https://scholar.google.com

Scholar Labs — ask the question, not the keywords

https://scholar.google.com/scholar_labs/search — a search mode inside Scholar, launched in November 2025 and open to all signed-in users, in any language, since June 2026. Instead of a keyword string you type the actual research question — "which methods verify intermediate reasoning steps in LLMs, and how are they evaluated?" — and it breaks the question into its topics and relationships, searches each, and returns papers that answer the whole question, with a sentence on *how* each one does and follow-up questions to narrow in. Use it in the first week of a survey, when you do not yet know the field's vocabulary; log its queries like any other (§6); and read the papers — the explanation is a lead, not a summary you can cite.

Alerts — set these up today

- **Query alerts** for your 2–3 core topics (use quoted phrases, or you will drown).
- **Citation alerts** on the 5 anchor papers of your area — new work almost always cites them, so this is a near-complete feed of your subfield.
- **An alert on your own profile**, so you learn who is citing you.
- Set a weekly slot (30 minutes, same time) to triage the alert emails into Zotero. Alerts you never read are worse than no alerts.

Your Scholar profile

Create it in week one, even with zero papers. Use your institutional email so it gets verified, add a photo, list your interests, keep the publication list clean (merge duplicates, delete the garbage

Query / feature	What it does
`"contrastive decoding"`	Exact phrase. Always quote multi-word technical terms, or Scholar matches them loosely.
`retrieval OR retrieve`	`OR` must be capitalised. Essential, because fields rename themselves constantly.
`-survey -review`	Excludes a term. Useful once surveys swamp your results.
`author:"K Skianis"`	Author search; try initial + surname, and surname variants.
`source:"Empirical Methods"`	Restricts to a publication whose name contains the phrase — the way to search within a venue.
`intitle:hallucination`	Term must appear in the title. Brutal and effective precision filter; `allintitle:` for all terms.
Custom range (left panel)	Year filtering. Combine with **Sort by date** to see this month's work rather than the most-cited work of 2019.
Cited by N	Forward citations — who built on this paper.
Search within citing articles	The most under-used feature in academia: search *inside* the set of papers citing a key work. "Who cited BERT *and* talks about Greek?" is one query.
Related articles	Scholar's similarity graph. Good for escaping your own vocabulary.
All N **versions**	Finds the free PDF and the published version of the same work.
Cite → BibTeX	Enable it once: Settings → Bibliography manager → Show links to import citations into BibTeX. (Still prefer DBLP/ACL Anthology for final entries.)

Scholar attributes to you). Decide whether to make it public — for a researcher, public is almost always right. It is how programme chairs, reviewers, and future employers check who you are.

> **On h-index and citation counts**
>
> They are age-dependent, field-dependent, and gameable. A first-year student with $h = 0$ is exactly on schedule. Use these numbers as *navigation aids for finding influential work*, never as a measure of your worth or anyone else's.

7.2 The rest of the toolkit

Semantic Scholar (https://www.semanticscholar.org) — better structured than Scholar: TLDR summaries, "highly influential citations", citation *intent* (background vs. method vs. result), and a genuinely good free API for building your own tooling.

OpenAlex (https://openalex.org) — fully open bibliographic graph with a free API. The right tool if you want to compute anything over the literature (growth curves, co-authorship, venue trends).

ACL Anthology (https://aclanthology.org) — every *ACL paper ever, free, full-text searchable, with correct BibTeX. For NLP it beats Scholar for precision.

OpenReview (https://openreview.net) — submissions, *reviews*, rebuttals, and decisions for ICLR, NeurIPS and many workshops. Read the reviews of papers adjacent to yours before you write your own. Nothing else teaches you the standard this quickly.

DBLP (https://dblp.org) — authoritative CS metadata. Use it to enumerate everything a person or venue published, and for clean BibTeX.

Connected Papers (https://www.connectedpapers.com), **ResearchRabbit**, **Litmaps** — visual citation neighbourhoods from one seed paper. Excellent for the first week of a survey, when you do not yet know the right words.

Scite (https://scite.ai) — tags each citation as supporting, mentioning or contrasting the cited claim, which is the question citation *counts* cannot answer. Paid beyond a trial; worth it for a survey.

Unpaywall (https://unpaywall.org/products/extension) — a browser extension that finds the legal open-access copy of a paywalled paper. Install it the day your institutional access fails you.

Hugging Face (https://huggingface.co/papers) — daily papers, plus models and datasets with download counts: a usage signal that citation counts miss entirely.

alphaXiv / arXiv listings — discussion layers and raw daily feeds for cs.CL and cs.LG. Skim titles daily, 10 minutes, no guilt about what you skip.

> **Papers with Code, then and now**
>
> The original `paperswithcode.com` was discontinued in 2025; its domain now redirects to Hugging Face's own Papers hub, so old links still work but land there. A separate successor under the Papers with Code name runs at https://paperswithcode.co: trending papers, a following feed for tasks, authors and organisations, a paper archive, the deadline board of §3, and an API plus MCP endpoint. The site does not say who operates it, so treat it as a convenient mirror and cite original sources.

8 Reading the signals: citations, stars, and attention

Part of research taste is judging, quickly, whether a line of work is gaining momentum or quietly dying. Two cheap signals do most of the work — as long as you know what each actually measures.

8.1 Citations: measure velocity, not volume

- **Normalise by age.** A 2019 paper with 400 citations and a 2026 paper with 60 are not comparable. Compute citations per year since publication; for recent work, citations in the *last* 12 months is the sharper signal.
- **Look at the second derivative.** A paper whose citations are accelerating marks a live area. One with a flat tail marks a closed one. Scholar's per-year bar chart on a paper's page shows this at a glance.
- **Read *who* cites, not how many.** Ten citations from the three strongest groups in the area beat 200 from unrelated surveys. Semantic Scholar's citation-intent labels help: being cited as a *method* others use is worth far more than a background mention.
- **Watch for inflation:** large author lists citing each other, survey-driven citation piles, and benchmark papers that collect citations for existing rather than for being good ideas.
- **Venue trend as a field signal.** Count submissions or accepted papers mentioning a term per year (OpenReview and the ACL Anthology make this scriptable). A term going from 5 to 300 papers in two years tells you both that it is hot and that you are probably late.

8.2 GitHub: stars, and the signals that matter more

Stars are the field's informal applause. They are a real signal of attention — and a poor signal of quality.

- Useful GitHub search syntax: `transformers stars:>500 pushed:>2026-01-01 language:python`, and `path:` / `in:readme` to find code inside repositories.
- **Watch → Custom → Releases** on the 5–10 repositories you depend on. Far better than re-checking manually.
- Cross-check with other attention signals: Hugging Face model downloads and likes, PyPI download counts, and whether the method appears in other people's baselines — the last one is the ground truth.

What to check	What it tells you
Star history (https://star-history.com)	The *shape* matters: a vertical launch-day spike then flat means one viral post; a steady upward slope means real adoption.
Last commit / commit frequency	3k stars and nothing pushed in 14 months is an abandoned repo. Check the default branch, not the releases page.
Contributors	One author vs. thirty. Thirty means a community will keep it alive — and that bugs get found.
Open vs. closed issues, and who answers	Authors replying within days is the strongest health signal there is.
Issues searched for "reproduce"	Do other people get the paper's numbers? Search the issue tracker for `reproduce`, `results`, `mismatch` before you trust a baseline. *Do this every time.*
Forks, "Used by", dependents	Forks are often bookmarks; genuine downstream dependents are the stronger signal.
Releases, tests, CI, pinned deps	Signals that running it will cost you a day, not a fortnight. A lockfile is a gift.
Checkpoints provided	Can you evaluate without retraining? Often the difference between a usable and a theoretical baseline.
LICENSE	Research-only, non-commercial, or custom model licences restrict what you may publish or build on. Check *before* you invest.

> **The trap, stated plainly**
>
> Stars measure how well a project was marketed. Citations measure how well a paper was positioned. Neither measures whether the idea is correct. The only test that counts is running it yourself on a task you understand — which is why reproducing one baseline (§1.1) is in your first 90 days.

9 Set up an AI assistant — and the rules for using it

An AI coding and reading assistant is now part of a competent research setup, in the same way a reference manager is. Used well, it removes weeks of mechanical work per year. Used badly, it fabricates citations into your literature review and hollows out the understanding your thesis is supposed to demonstrate.

9.1 The setup

1. **A terminal/IDE coding agent** in each project repository (Claude Code, Codex CLI, Cursor, or GitHub Copilot — any of them). This is where the leverage is: it reads your code, runs your tests, and edits files.
2. **A project memory file** (e.g. `CLAUDE.md` at the repo root) stating your conventions: Python version and env manager, how to launch training, where data lives, plotting style, "never commit checkpoints", "one sentence per line in .tex". Written once, it stops you re-explaining your project every session.
3. **Connect your sources.** Via connectors/MCP servers, give the assistant access to your filesystem, your Zotero library, and literature APIs (arXiv, Semantic Scholar). An assistant that can read the actual PDFs in your library is a different tool from one that guesses.
4. **Saved prompts for repeated jobs** — paper triage, extraction into your survey schema, "review this draft as a hostile ARR reviewer". Keep them in the repo so they improve over time.
5. **A lab notebook it can read and write:** `notes/` with one dated Markdown file per week. Decisions, failed runs and why, hyperparameters, links. This is also what your assistant needs in order to be useful in month six.

9.2 What to use it for

- Triaging abstracts against your inclusion criteria.
- Pulling tables/numbers out of PDFs into your extraction sheet.
- Explaining unfamiliar notation; walking a derivation step by step.
- Getting an unfamiliar baseline repo to run; writing eval harnesses.
- Data pipelines, plotting code, Slurm scripts, boilerplate.
- Refactoring research code before it becomes unmaintainable.
- LaTeX debugging, table formatting, prose tightening.
- Adversarial pre-review of your own draft against the venue's form.
- Drafting rebuttal *prose* from *your* arguments.
- Turning notes into a first draft you then rewrite.

A first review before any human reads it: Deep Review

Google's **Deep Review** (https://research.google/deepreview/; public since October 2026, after pilots on more than 16,000 manuscripts at STOC, ICML, NeurIPS and ICLR) reads a PDF or plain text the way a careful referee would: it splits the manuscript into sections, checks each in parallel for mathematical errors, logical gaps and unsupported claims, then cross-checks the whole. Two reviews a week are free; more need a Gemini API key. Google states that uploads are not used to train its models.

- Run it on the final PDF before every submission, then *read every flag*: fix what is wrong, and write down why you reject what is not. Treat it as one more hostile reader, never as a verdict.
- It is for *your own* drafts. Papers you are reviewing stay confidential (rule 3 below), and anything under NDA stays off the service whatever its training policy says.

9.3 Non-negotiable guardrails

Five rules

1. **Never cite a paper you have not opened.** Models invent plausible references — correct-looking authors, venue, year, all fabricated. Every entry in your `.bib` must come from Zotero, DBLP, or the Anthology, after you read at least the abstract.
2. **Verify every number and every claim** against the source. Treat assistant output as a competent draft from a stranger, not as a result.
3. **Never paste confidential data** — NDA'd industry data, unpublished third-party manuscripts, papers you are reviewing — into a cloud assistant. Reviewing material is confidential; using an LLM to write reviews is banned at most venues.
4. **Know your venue's disclosure policy** and follow it. You are the author and you are responsible for every word.
5. **Keep the core yours.** Derive your own proofs, design your own experiments, form your own judgements. The point of a PhD is that *you* become the expert; outsource the mechanics, never the thinking.

10 Your personal site: `username.github.io`

You need one web page that you own, that outlives every institution you will work at, and that answers "who is this person and what have they done?" in ten seconds. GitHub Pages makes it free and permanent-ish; a Jekyll template makes it take an afternoon.

Why it matters more than it looks

Programme chairs, reviewers who found your paper, internship recruiters, postdoc hosts, and students looking for a supervisor all search your name. If the first hit is a stale institutional directory page, you lose the introduction. Your site is also where your PDFs, code links, slides, and CV actually stay reachable after you change jobs.

10.1 How GitHub Pages works

Create a repository named **exactly** `username.github.io` (your own GitHub username). Its contents are served at `https://username.github.io`. One such *user site* per account; free; HTTPS included; custom domains supported. Jekyll is built in, so a template of Markdown files becomes a site with no build pipeline of your own.

10.2 Use the AcademicPages template

AcademicPages — https://github.com/academicpages/academicpages.github.io — is the standard academic Jekyll template (built on Minimal Mistakes). It ships exactly the pages an ML/NLP researcher needs: publications, talks, teaching, CV, blog posts, and a portfolio.

1. On the AcademicPages repository, click **Use this template → Create a new repository**; name it `username.github.io`. (Cleaner than forking, which complicates pull requests and search.)
2. Edit `_config.yml`: your name, title, email, bio, location, GitHub, Scholar, ORCID, LinkedIn, and the site URL. This file alone controls most of what visitors see.
3. Replace the content, which is all Markdown with YAML front matter:
   - `_pages/about.md` — the landing page. Three paragraphs maximum: who you are, what you work on, what you are looking for.
   - `_publications/` — one file per paper (title, venue, date, links to PDF / arXiv / code / BibTeX).
   - `_talks/`, `_teaching/`, `_posts/`, `_portfolio/` — same pattern.
   - `files/` — your CV PDF and paper PDFs; `images/` — your profile photo.
4. Generating publications from BibTeX: the scripts in `markdown_generator/` turn a `.bib` or TSV export into the per-publication Markdown files. Export from Zotero, run the script, commit — your site then updates from the same library that feeds your papers.
5. Commit and push. GitHub builds and publishes within a minute or two; check the Actions tab if it does not appear. Optional local preview: `bundle install && bundle exec jekyll serve`.
6. Optional: a custom domain — add a `CNAME` file and point your DNS at GitHub Pages. Worth it if you own `firstname-lastname.com`, but `username.github.io` is perfectly respectable.

> **A worked example: https://y3nk0.github.io**
>
> My own site is built on AcademicPages. Copy its structure rather than its content:
>
> - **About** — short bio with research interests (NLP, ML, robustness, reasoning, interpretability, trustworthy models), plus background and affiliations.
> - **News** — a running list of accepted papers, talks, and grants. The cheapest way to look active, and the first thing returning visitors read.
> - **Research** — themes rather than a paper dump, so a visitor learns what you *work on*, not just what you published.
> - **Publications** — venue, year, and links out to PDF and code.
> - **Talks** and **Teaching** — separate pages, which matter for academic job applications.
> - **CV** — both a downloadable PDF and an HTML version.
>
> Because AcademicPages sites are public repositories, you can inspect the source of any site you like — mine included (`github.com/y3nk0/y3nk0.github.io`) — and borrow its structure directly.

- **Alternatives: al-folio** (Jekyll, very popular in ML, BibTeX-driven publications page), **Minimal Mistakes** (the base theme, if you want full control), **Hugo Blox** (formerly Wowchemy), or a **Quarto** website if you already write in Quarto. Any of them beats having no site.
- **Keep it current.** Update after every acceptance, every talk, and every CV change. A site last touched three years ago reads as worse than no site.

11 Research infrastructure and habits

The tools above handle literature and writing. These handle the part that actually consumes your time: experiments.

11.1 Make experiments reproducible from day one

- **One git repository per project.** Code, configs, notes, the paper. Never "final_v3_real.py".
- **Deterministic environments:** uv or conda with a lockfile, or Docker/Apptainer if your cluster supports it. Record versions of everything, including CUDA.
- **Config files, not command-line archaeology** (Hydra, or plain YAML). Every run must be reconstructible from a committed config plus a commit hash.
- **Seeds and variance.** Report mean ± std over at least 3 seeds. A single-seed improvement of 0.3 points is noise, and a reviewer will say so — correctly.
- **Experiment tracking** from your first run: Weights & Biases, MLflow, or TensorBoard. Log configs, metrics, and the commit hash. "Which run produced Table 3?" must have an answer in month nine.
- **A results file, not screenshots.** Your tables and figures should regenerate from logged results with one script.
- **Compute:** learn your lab's cluster and Slurm early (job arrays, checkpoint-and-resume, time limits); know what free or national resources you can apply for; and treat "can I actually run this?" as a design constraint when choosing problems.

11.2 Habits that compound

Weekly advisor meeting with a written agenda. Send 5 bullets beforehand: what I did, what I found, what is blocked, what I propose, what I need from you. Keep a shared running document. This one habit separates students who progress from students who drift.

A dated lab notebook. Including negative results — especially negative results, since you will otherwise repeat them.

Read daily, 20 minutes. Titles and abstracts from the arXiv feed, then one paper properly. Join or start a reading group; explaining a paper out loud exposes exactly what you did not understand.

Write continuously. Keep a living draft of the current paper from week one; writing is how you discover that your story has a hole.

Say yes to reviewing (workshops first, then ARR) once you have a paper or two. Nothing teaches you what makes a paper acceptable faster than reviewing eight of them.

Be present in the community. Follow researchers in your area, go to your local ML/NLP meetups, apply to summer schools and doctoral consortia, volunteer at a conference you cannot otherwise afford, and talk to people at posters — including, especially, posters you only half understand.

Protect your mind. Impostor feelings are near-universal and peak in year one; rejection is the default outcome of submission, not a verdict on you. Keep interests outside the PhD, keep a list of things that have worked, and talk to people when it gets heavy.

12 Checklists

Week 1

- Zotero + connector + Better BibTeX installed; auto-export configured.
- Google Scholar profile created; 3 alerts live.
- ORCID, arXiv, OpenReview, GitHub, Hugging Face accounts.
- Overleaf account; institutional licence checked.
- Deadline tracker bookmarked; target venues in the calendar.
- AI assistant set up with a project memory file.
- Project repository created with `notes/` and a README.
- Cluster access working; a trivial job submitted successfully.

Month 1

- Survey question agreed in writing with your advisor; criteria written down.
- Seed papers read with the three-pass method; extraction sheet started.
- Search log running (queries, dates, hit counts).
- Weekly advisor meeting slot fixed, with the 5-bullet agenda format.
- Reading group joined.

Months 2–3

- 60–120 papers screened and extracted; taxonomy drawn.
- 10–15 pages of survey written; three candidate gaps written as testable questions.
- One baseline reproduced end to end, tracked, with numbers you trust.
- `username.github.io` live, with About, Publications (even if empty), CV.
- A target venue chosen and backward-planned from its deadline.

Before every submission

- Every DOI and arXiv id in `refs.bib` resolves; titles match; nothing cited that you have not opened.
- Published versions cited where they exist; `bibtex-tidy` run.
- Capitals protected in titles; no "arXiv preprint" for published work.
- `arxiv-latex-cleaner` run; no names in comments or PDF metadata; code anonymised if the venue is double-blind.
- Limitations section and the venue's checklist completed.
- The venue's AI-assistance disclosure made, if required.
- A Deep Review pass on the final PDF; every flag fixed or consciously rejected.
- Every table and figure regenerated by script from logged results; seeds and variance reported.
- Page limit, template and anonymity window checked on the venue page, not the aggregator.

Year 1

- One submission out — workshop or main conference. Submitted beats perfect.
- A rebuttal written (your first one will teach you more than the paper did).
- A talk given somewhere: lab seminar, reading group, local meetup.
- Code and checkpoints released for at least one project.
- A one-page research statement you can hand to anyone who asks what you work on.

A Link directory

What	Where
Deadlines and venues	
PwC AI deadlines	https://paperswithcode.co/ai-deadlines
AI deadlines (ML/CV/NLP)	http://aideadlines.org/?sub=ML,CV,NLP
AI deadlines (mirror)	https://huggingface.co/spaces/huggingface/ai-deadlines
Deadline source repo	https://github.com/huggingface/ai-deadlines
CCF deadlines	https://ccfddl.com
CORE conference ranks	https://portal.core.edu.au/conf-ranks/
CSRankings	https://csrankings.org
Scholar venue metrics	https://scholar.google.com/citations?view_op=top_venues
Writing and sharing	
Overleaf	https://www.overleaf.com
Prism (closes 1 Jan 2027)	https://prism.openai.com
Deep Review (Google)	https://research.google/deepreview/
arXiv	https://arxiv.org
arxiv-latex-cleaner	https://github.com/google-research/arxiv-latex-cleaner
Anonymous code sharing	https://anonymous.4open.science
ORCID	https://orcid.org
latexdiff	https://ctan.org/pkg/latexdiff
CITATION.cff	https://citation-file-format.github.io
Zenodo	https://zenodo.org
References and search	
Zotero	https://www.zotero.org
Better BibTeX	https://retorque.re/zotero-better-bibtex/
bibtex-tidy	https://github.com/FlamingTempura/bibtex-tidy
Google Scholar	https://scholar.google.com
Scholar Labs	https://scholar.google.com/scholar_labs/search
Semantic Scholar	https://www.semanticscholar.org
OpenAlex	https://openalex.org
ACL Anthology	https://aclanthology.org
OpenReview	https://openreview.net
DBLP	https://dblp.org
Connected Papers	https://www.connectedpapers.com
ASReview	https://asreview.nl
Scite	https://scite.ai
Unpaywall	https://unpaywall.org/products/extension
HF Daily Papers	https://huggingface.co/papers
Papers with Code	https://paperswithcode.co
Attention signals	
Star history	https://star-history.com

What	Where
Hugging Face hub	https://huggingface.co/models
Personal site	
AcademicPages template	https://github.com/academicpages/academicpages.github.io
GitHub Pages docs	https://docs.github.com/pages
Example site	https://y3nk0.github.io
al-folio	https://github.com/alshedivat/al-folio

B A minimal paper repository

```
project/
  README.md                                  citable repo; Zenodo DOI per release
  CITATION.cff                               conventions for your AI assistant
  CLAUDE.md                                  pinned
  pyproject.toml / environment.yml           one YAML per experiment
  configs/
  src/                                       train.sh, eval.sh, slurm/
  scripts/                                   2026-10-15.md, dated lab notebook
  notes/                                     logged metrics, committed
  results/                                   results → PDF
  figures/make_figures.py
  paper/
    main.tex, sections/, refs.bib            BBT auto-export target
```

The shortest version of all of this

Keep a clean library, know your deadlines, read forward from the anchor papers, reproduce before you innovate, write every week, publish a page with your name on it, use the assistant for the mechanics and never for the thinking — and ask your advisor the question you are embarrassed to ask. Good luck.