Starting a PhD in Machine Learning or NLP

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

y3nk0.github.io

October 2026 ▪ companion to the full starter guide and the one-page cheat sheet

What this deck is

A PhD is won on **compounding habits**, not heroic sprints.

The first 90 days decide how much friction every later month carries:

- Can you find, in November, the paper you read in March?
- Can you rerun the experiment behind Table 3?
- Can anyone find *you*?
- Does a deadline ever surprise you?

Roadmap

1. Venues: where your work goes
2. Deadlines
3. Writing & sharing
4. Zotero
5. Your first project: a survey
6. Searching the literature
7. Reading the signals
8. AI assistant
9. Your personal site
10. Infrastructure & habits

The first 90 days, condensed

Week 1 — infrastructure

Zotero + Better BibTeX
Scholar profile + 3 alerts
Accounts: Overleaf, OpenReview, GitHub, HF, arXiv, ORCID
Deadlines in the calendar
AI assistant + memory file
One repo per project, with `notes/`

Weeks 2–10 — a survey

A narrow question, agreed in writing.

Output: 60–120 papers screened, an extraction sheet, a taxonomy figure, 10–15 pages written, **three gaps** stated as testable questions.

The highest-return thing you can do in term one.

Weeks 10–13 — identity

Publish `username.github.io`

Reproduce *one* baseline end to end, tracked, on your own hardware

Pick a target venue and backward-plan from its deadline

Part 1

The publication landscape

ML and NLP are conference-first

1. **Cycles, not rolling submissions.** One or two hard deadlines per year. Miss by an hour, wait months.
2. **Reviewing is public-ish.** ICLR and many workshops review on OpenReview — reviews and rebuttals stay online.
3. **A paper has a lifecycle.** 6–12 months from idea to published.

The lifecycle

preprint → submission → reviews → **rebuttal** → decision → camera-ready → talk / poster → citations

Read other people's reviews

OpenReview is the best free writing lesson in the field. Read the reviews and rebuttals of papers near your topic *before* you write yours.

Core ML venues

Dates are indicative patterns — verify every year

Venue	Scope	Rhythm
NeurIPS	Broadest ML: theory, deep learning, datasets & benchmarks	abstract ∼May; December
ICML	Core ML, methods and theory	∼Jan–Feb; July
ICLR	Representation learning, LLMs; fully open review	∼Sept; April–May
AISTATS	Statistics × ML	∼Oct; April–May
COLT / UAI	Learning theory / probabilistic & causal	winter–spring
AAAI / IJCAI	Broad AI and applications	∼Aug / ∼Jan
COLM	Language modelling: data, training, evaluation	∼March; October
MLSys / LoG	Systems for ML / learning on graphs	autumn

Core NLP venues

Venue	Scope	Notes
ACL	The flagship: all of CL and NLP	summer; via ARR
EMNLP	Empirical methods — the other flagship	late year; via ARR
NAACL / EACL / AACL	Regional ACL conferences, same standard	alternate by year
COLING / LREC	CL / language resources and evaluation	check the current cycle
CoNLL / *SEM / SemEval	Learning and language, semantics, shared tasks	good first-paper targets
Interspeech / ICASSP	Speech and audio	own cycles
TACL	Journal of the ACL, rolling; presented at *ACL	slower, prestigious
Findings	Companion archival track for solid papers	a real, citable publication

ARR: the mechanic that confuses every first-year



Plan around this

Two deadlines: ARR submission *and* venue commitment.

Reviews are reusable: revise and commit elsewhere, or resubmit with a revision note.

Desk-reject risks

A **Limitations** section is mandatory (and free of page limit).

So is the Responsible NLP checklist.

Where a first paper realistically lands

Adjacent venues
KDD, WSDM, CIKM, SIGIR, The Web Conference, ECML-PKDD · CVPR / ICCV / ECCV for multimodal work.

Journals
JMLR · **TMLR** — rolling, fast, judged on correctness and interest rather than novelty hype; an excellent fit for careful work conference reviewers undervalue · TACL · *Computational Linguistics* · TPAMI.

Aim your first submission at a good workshop

4–8 pages, fast turnaround, real feedback, real audiences.

You learn the entire pipeline — writing, reviewing, rebuttal, poster, conversations — at a fraction of the risk.

Judging a venue without asking around

- **Google Scholar Metrics** — per-venue h5-index.
- **CORE ranking** — A*/A/B/C tiers, used for evaluation in many countries.
- **CSRankings** — which venues count as top-tier *per subarea*: the useful signal.
- **Where does the work you admire appear?** The most reliable test.

Predatory venues

Invitation email + acceptance in two weeks + a large fee + a scope spanning "AI, Big Data, Blockchain and IoT" = a scam.

Publishing there actively damages your CV.

Part 2

Deadlines

Two boards to bookmark

1. Papers with Code — AI Deadlines paperswithcode.co/ai-deadlines

Filter by subfield (ML, NLP, CV, DM, Robotics, IR, Speech, RL, . . .) or *top tier only*. Tracks **every** milestone — abstract registration, reviews released, author feedback, rebuttal, commitment, notification, camera-ready — with live countdowns, conference dates and locations. API + MCP endpoint for your assistant.

2. AI Deadlines, pre-filtered aideadlines.org/?sub=ML,CV,NLP

The `sub=` parameter carries the filter, so the link *is* your personalised board — edit the tags as your interests change. Mirrored as a Hugging Face Space; the data is open YAML, so a missing deadline is a pull request away: an easy first open-source contribution.

AoE = UTC−12. Cross-check both against the official venue page — aggregators lag, calls for papers do not.

A workflow that actually holds

1. Filter to your subfields; list the 4–6 venues you could target in 18 months.
2. Calendar each one **three times**: the deadline, -7 days (results frozen, writing only), -21 days (experiments running).
3. Add the dates people forget: abstracts, ARR commitment, supplementary, **rebuttal window** (do not travel), camera-ready, early registration.

Backward-plan on paper

camera-ready ← decision ← rebuttal ← submission ← freeze ← final experiments ← **ablations** ← main result

Ablations always take longer than the main result.

In moderation

Deadlines are useful forcing functions and terrible research strategies. Keep one long-horizon question alive alongside them — that is where your thesis contribution comes from.

Part 3

Writing and sharing

Where you write

Overleaf overleaf.com

- Real-time co-editing — what you want in the 48 hours before a deadline.
- Official venue templates in the gallery: *start from them*, not from last year's copy.
- Comments and chat for advisor feedback.
- Now has its own metered AI assistant — same rules as any AI editor.
- Check for an institutional licence before paying; the free tier limits collaborators, compile time, history, Git sync.

Prism closes on 1 January 2027

OpenAI's AI-native LaTeX workspace launched in January 2026, was free and genuinely good, and lasted under a year.

- **Do not start anything there.**
- **Export existing projects now** — exports drop comments, version history, collaborators and integrations.
- Land on Overleaf, or `git` + `latexmk` locally.

The transferable lesson

A free cloud tool can disappear with a quarter's notice; your thesis cannot. Keep manuscripts as plain `.tex` under git that you can compile offline, and treat any editor as a window onto that

Two cautions, and the local workflow

Before moving your thesis into any AI editor

Policy: most venues have an explicit AI-assistance policy — disclosure requirements, and a hard ban on LLM-written *reviews*. You remain responsible for every sentence, citation and number.

Confidentiality: never upload NDA'd data, industry collaborations, or someone else's unpublished work.

Keep a local copy too

- `git` + `latexmk -pdf -pvc` + VS Code.
- **One sentence per line** — turns git diffs from useless to invaluable.
- `\input` per section, so two people do not collide.
- Figures generated by a script from logged results. Never hand-edited.
- `\citet` in the sentence, `\citep` as an aside; `latexdiff` for revisions.

Preprints, anonymity, artefacts

- **arXiv** — cs.CL, cs.LG, cs.AI, stat.ML. First submission needs endorsement; your advisor handles it.
- **Anonymity:** ACL venues enforce a window around the review cycle; ML venues are double-blind but usually allow preprints. *Check before you post.*
- Anonymise code too (anonymous.4open.science), and strip names from PDF metadata.

Run this every time

arxiv-latex-cleaner strips comments, unused files and bloat.

Reviewers have found deanonymising information in LaTeX comments.

Artefacts

Code on GitHub with a licence, checkpoints and data on Hugging Face, an exact environment spec — and an **ORCID** on everything.

A **CITATION.cff** in the repo and a Zenodo DOI per release make the code citable on its own.

Part 4

Zotero

Set this up before paper number ten

The setup, in order

1. **Zotero** desktop + the browser **connector** — one click saves metadata *and* the PDF.
2. **Better BibTeX**: pinned citation keys (`devlin2019bert`) and **auto-export** into your repo's `paper/refs.bib`. Never hand-edit BibTeX again.
3. Sync (300 MB free, or your own WebDAV / linked folder).
4. A **group library** with your lab and co-authors.
5. Collections per project, tags per theme, saved searches.
6. Read and annotate *inside* Zotero — highlights stay searchable for years.

Habits that keep it trustworthy

- **Cite the published version**, not the preprint.
- Clean BibTeX from **DBLP** and the **ACL Anthology** — Scholar's is sloppy about venues, pages, editors.
- Add items by DOI / arXiv ID.
- Merge duplicates monthly.
- One-sentence note on save: what it does, why I care.

Overleaf's paid tiers link Zotero directly — but auto-export into git is better anyway: versioned, diffable, identical for every co-author.

Getting a reference into your `.bib`

The easiest route — no setup

1. Google Scholar: *Settings* → *Bibliography manager* → tick *Show links to import citations into BibTeX*. Once, ever.
2. A **Cite** link appears under every result → **BibTeX** → copy.
3. Paste into `refs.bib`, or into Zotero with *File* → *Import from clipboard*.

Bulk: star papers into *My library*, select, *Export* → *BibTeX*.

Check what it gives you

"arXiv preprint arXiv:1810.04805" for a paper published at NAACL, missing pages, inconsistent venue names, mangled accents. A bibliography full of "arXiv preprint" says nobody looked.

Cleaner, one click each: ACL Anthology (per paper, and a `.bib` per volume) · **DBLP** · **arXiv** "Export BibTeX" · Semantic Scholar, ACM, IEEE, Springer.

Any DOI, from the terminal:

```
curl -LH "Accept:
application/x-bibtex"
https://doi.org/<DOI>
```

Past ∼20 refs: connector + Better BibTeX auto-export does it all.

Three BibTeX gotchas, three habits

- **Capitals vanish.** Styles lowercase titles — protect them: `title = {{BERT}: Pre-training ...}`.
- **Authors:** `Last, First and Last, First`. Never a comma list, never a hand-typed "et al."
- **arXiv-only work** gets `eprint`, `archivePrefix`, `primaryClass` — not an invented journal. One source per venue family.

Three habits

bibtex-tidy formats, sorts, deduplicates and strips junk fields (github.com/FlamingTempura/bibtex-tidy).

Audit: loop the `curl` line over every DOI in `refs.bib`. A reference that returns no title is a typo or a fabrication — open whatever the script flags.

Retractions: Zotero checks Retraction Watch and marks retracted items with a red banner — look before you cite.

Part 5

Your first project: a focused survey

Why a survey is the right first project

- **Vocabulary** — you can talk to your advisor as a peer within weeks.
- **A taxonomy** — and your first paper usually lives in a box you drew that has nothing in it.
- **A bibliography** you reuse in every paper and in your thesis.
- **Baselines, datasets, metrics** — learned before you waste a month on a non-standard setup.
- **Publishable**: a survey, a position paper, a workshop paper, and the core of your related-work chapter.

The deliverables

- 60–120 papers screened and extracted
- An annotated bibliography in Zotero
- An extraction spreadsheet
- One taxonomy figure
- 10–15 pages of writing
- **Three gaps**, each a testable question plus the experiment that would answer it

8–12 weeks. Timeboxed.

A protocol you can run

1. **Write the question down.** Not "LLM reasoning" but "methods for verifying intermediate reasoning steps, 2022–2026, and how they are evaluated". Agree it in writing.
2. **Seed set:** 5–10 anchor papers from your advisor. Their shared vocabulary becomes your queries.
3. **Criteria** for inclusion and exclusion, written *before* screening — or you will rationalise.
4. **Search systematically** across Scholar, Semantic Scholar, the Anthology, OpenReview, DBLP — and **log every query, date and hit count**.
5. **Snowball both ways:** backward through references, forward through "cited by". Forward finds the last six months.
6. **Screen in two passes:** title + abstract (ruthless, keep a "maybe" pile), then full text. Expect 400–600 screened, 60–120 kept. Past a few hundred, **ASReview** orders the pile by learned relevance.
7. **Extract into a table**, not prose. One row per paper.
8. **Build the taxonomy.** Cluster until the groups stop changing. If you cannot draw it, you do not understand the field yet.
9. **Write as you go** — each section ends with "what is missing here".
10. **Finish with three gaps.** Take them to your advisor; that meeting is where your first paper starts.

The extraction schema

A spreadsheet 100 rows deep beats 100 highlighted PDFs

Key	pinned BibTeX key
Venue / year	published venue, not "arXiv"
Problem	the task, one line, your words
Method family	your taxonomy label
Data	datasets and sizes
Metrics	comparable across papers?

Headline result	the number they lead with + the baseline
Compute	GPUs, hours, model size — can *you* run it?
Code	link + does it reproduce?
Limitations	stated and unstated
Relevance	1–5 and "why I care"

Read in three passes (Keshav)

5 min: skim and decide → 1 h: figures, tables, method → hours: re-derive. Pass 3 for ∼20 papers; pass 1 for hundreds.

Scope discipline

Freeze the search date ("literature up to 15 November 2026") and ship. A documented cut-off is exactly what a reviewer wants to see.

Part 6

Searching the literature

Google Scholar is an instrument, not a search box

Query / feature	What it does
`"contrastive decoding"`	exact phrase — always quote technical terms
`retrieval OR retrieve`	`OR` must be capitalised; fields rename themselves
`-survey -review`	exclude terms once surveys swamp results
`author:"K Skianis"`	author search; try initial + surname variants
`source:"Empirical Methods"`	search within a venue
`intitle:hallucination`	brutal, effective precision filter
Custom range + Sort by date	this month's work, not 2019's most-cited
Cited by → search within citing articles	the most under-used feature in academia: "who cited BERT *and* mentions Greek?"
Related articles	escape your own vocabulary
All N versions	the free PDF and the published version
Cite → BibTeX	enable once in Settings (still prefer DBLP)

Alerts, profile, and the rest of the toolkit

Alerts — today

Query alerts for 2–3 core topics · **citation alerts on your 5 anchor papers** ≈ a near-complete feed of your subfield · an alert on your own profile · a fixed 30-minute weekly triage into Zotero.

On h-index and citation counts

Age-dependent, field-dependent, gameable. A first-year with $h = 0$ is exactly on schedule. Navigation aids, never a measure of worth.

- **Semantic Scholar** — TLDRs, "highly influential" citations, citation *intent*, a good free API.
- **OpenAlex** — open graph; compute over the literature.
- **ACL Anthology** — every *ACL paper, correct BibTeX.
- **OpenReview** — submissions, reviews, rebuttals, decisions.
- **DBLP** — authoritative CS metadata.
- **Connected Papers**, ResearchRabbit, Litmaps — citation neighbourhoods from one seed.
- **Scite** — supporting vs contrasting citations (paid).
- **Hugging Face papers** + **paperswithcode.co** (a successor site: trending, feed, API, MCP; `.com` now redirects to HF Papers).

Two Google tools: ask a question, get a first review

Scholar Labs

scholar.google.com/scholar_labs

- A search mode inside Scholar: type the *research question*, not keywords. Open to all signed-in users since June 2026.
- Breaks it into topics and relationships, searches each, and returns papers that answer the whole question — with a sentence on how.
- Follow-up questions to narrow in. Best in the first week of a survey.
- Log its queries like any other; read the papers — the explanation is a lead, not a citation.

Deep Review

research.google/deepreview

- Google Research, public since October 2026 after pilots at STOC, ICML, NeurIPS and ICLR. Upload a PDF; agents check each section for maths errors, logical gaps and unsupported claims, then the whole.
- Two free reviews a week; more with a Gemini API key. Uploads are not used for training.
- Run it on the final PDF before every submission. Read every flag: fix what is wrong, note why you reject the rest.
- **Your own drafts only.** Never a paper you are reviewing, never NDA'd work.

Part 7

Reading the signals

Citations: measure velocity, not volume

- **Normalise by age.** Citations per year since publication; for recent work, citations in the *last* 12 months.
- **Look at the second derivative.** Scholar's per-year bar chart: accelerating = a live area, flat tail = a closed one.
- **Read who cites, not how many.** Ten citations from the three strongest groups beat 200 from unrelated surveys.
- **Cited as a method** > cited as background.
- **Field signal:** papers per year using a term — 5 to 300 in two years means hot, and that you are late.

Inflation

Large author groups citing each other, survey-driven piles, and benchmark papers that collect citations for *existing* rather than for being good ideas.

GitHub: stars, and the signals that matter more

What to check	What it tells you
Star history (star-history.com)	launch-day spike then flat = one viral post; steady slope = real adoption
Last commit, frequency	3k stars and nothing in 14 months = abandoned
Contributors	one author vs. thirty: thirty means bugs get found
Who answers issues	authors replying within days is the strongest health signal
Search issues for "reproduce"	**do this every time** before trusting a baseline
Releases, CI, pinned deps	whether running it costs a day or a fortnight
Checkpoints	evaluate without retraining
LICENSE	research-only / non-commercial terms restrict what you may publish

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 — which is why reproducing one baseline is in your first 90 days.

Part 8

Your AI assistant

Setup, and what to use it for

The setup

1. A coding agent in each project repository (Claude Code, Codex CLI, Cursor, Copilot — any of them).
2. A **project memory file** with your conventions: env manager, how to launch training, where data lives, "never commit checkpoints".
3. **Connect your sources** via MCP: filesystem, Zotero, arXiv / Semantic Scholar, the deadlines API.
4. Saved prompts for repeated jobs.
5. A dated `notes/` lab notebook it can read and write.

What it is genuinely good at

- Triaging abstracts against your criteria.
- Pulling tables out of PDFs into your extraction sheet.
- Explaining notation; walking a derivation.
- Getting an unfamiliar baseline repo to run.
- Eval harnesses, data pipelines, plots, Slurm scripts.
- LaTeX debugging and prose tightening.
- An adversarial pre-review against the venue's own form.
- Rebuttal prose from *your* arguments.

Five non-negotiable guardrails

1. **Never cite a paper you have not opened.** Models invent plausible references — right-looking authors, venue, year, all fabricated. Every `.bib` entry comes from Zotero, DBLP or the Anthology.
2. **Verify every number and claim** against the source. Assistant output is a competent draft from a stranger, not a result.
3. **Never paste confidential material** — NDA'd data, unpublished third-party manuscripts, papers you are reviewing. LLM-written reviews are banned at most venues.
4. **Know your venue's disclosure policy** and follow it. You are the author; you are responsible.
5. **Keep the core yours.** Outsource the mechanics, never the thinking — the point of a PhD is that *you* become the expert.

Part 9

Your personal site

`username.github.io` — an afternoon, then permanent

Why it matters more than it looks

Programme chairs, reviewers who found your paper, recruiters, postdoc hosts and prospective students all search your name. If the first hit is a stale institutional directory page, you lose the introduction.

It also keeps your PDFs, code, slides and CV reachable *after* you change institution.

A repository named **exactly** `username.github.io` is served at `https://username.github.io`: free, HTTPS, one user site per account, Jekyll built in.

AcademicPages, step by step

1. *Use this template* → new repository named `username.github.io`.
2. Edit `_config.yml`: name, bio, email, GitHub, Scholar, ORCID, site URL.
3. Fill `_pages/`, `_publications/`, `_talks/`, `_teaching/`, `_posts/`, `files/`, `images/` — all Markdown with YAML front matter.
4. Generate publications from your `.bib` with `markdown_generator/`.
5. Push; GitHub Actions builds it in a minute.
6. Optional `CNAME` for a custom domain.

A worked example: y3nk0.github.io

My own site, on AcademicPages. Copy the *structure*:

- **About** — short bio and research interests.
- **News** — a running list of accepted papers, talks, grants. The cheapest way to look active, and the first thing returning visitors read.
- **Research** — themes, not a paper dump: what you *work on*.
- **Publications** — venue, year, links to PDF and code.
- **Talks** and **Teaching** — separate pages; they matter for academic applications.
- **CV** — downloadable PDF *and* an HTML version.

Steal from the source

AcademicPages sites are public repositories — inspect any site you like (`github.com/y3nk0/y3nk0.github.io`) and borrow its structure directly.

Alternatives

al-folio · Minimal Mistakes · Hugo Blox · Quarto. Any of them beats having no site.

Update after every acceptance.
A site last touched three years ago reads worse than no site.

Part 10

Infrastructure and habits

Make experiments reproducible from day one

- One git repository per project. Never `final_v3_real.py`.
- Deterministic environments (uv / conda / Docker), CUDA version recorded.
- Config files, not command-line archaeology: every run reconstructible from a committed config + commit hash.
- **Seeds and variance:** mean $\pm$ std over 3+ seeds. A single-seed $+0.3$ is noise, and a reviewer will say so.
- **Experiment tracking** from your first run (W&B / MLflow / TensorBoard): configs, metrics, commit hash. "Which run produced Table 3?" must have an answer in month nine.
- Tables and figures regenerate from logged results with one script.
- Learn Slurm early: job arrays, checkpoint-and-resume, time limits.
- "Can I actually run this?" is a design constraint when choosing problems.

Habits that compound

- **Weekly advisor meeting with a written agenda:** what I did, what I found, what is blocked, what I propose, what I need from you. This one habit separates students who progress from students who drift.
- **A dated lab notebook** — including negative results, or you will repeat them.
- **Read daily, 20 minutes.** Titles and abstracts, then one paper properly. Join a reading group.
- **Write continuously.** Writing is how you discover the hole in your story.
- **Review papers** once you have published one. Nothing teaches the standard faster.
- **Be present:** local meetups, summer schools, doctoral consortia, volunteering — and talk at 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 a list of what has worked, and talk to people when it gets heavy.

Part 11

Checklists

Week 1 and Month 1

Week 1

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

Month 1

- Survey question agreed in writing; criteria written down
- Seed papers read in three passes; extraction sheet started
- Search log running: queries, dates, hit counts
- Weekly advisor meeting fixed, 5-bullet agenda
- Reading group joined

Months 2–3, and Year 1

Months 2–3

- 60–120 papers screened and extracted; taxonomy drawn
- 10–15 pages written; three gaps as testable questions
- One baseline reproduced end to end, tracked, numbers you trust
- `username.github.io` live — About, Publications, CV
- A target venue chosen and backward-planned

Year 1

- One submission out — workshop or conference. **Submitted beats perfect.**
- A rebuttal written (it teaches more than the paper did)
- A talk given somewhere
- Code and checkpoints released for one project
- A one-page research statement you can hand to anyone

Before every submission

Bibliography

- Every DOI and arXiv id resolves; titles match; nothing cited unopened
- Published versions cited; `bibtex-tidy` run
- Capitals protected; no "arXiv preprint" for published work

Manuscript

- `arxiv-latex-cleaner` run; no names in comments or metadata; code anonymised
- Limitations section and the venue checklist done
- AI-assistance disclosure made, if required
- Tables and figures regenerated by script; seeds and variance reported
- Page limit, template and anonymity window checked on the venue page
- A Deep Review pass on the final PDF; flags fixed or consciously rejected

The shortest version

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.

Konstantinos Skianis ▪ University of Ioannina ▪ y3nk0.github.io