

Gherbal v4

Comparative Evaluation Report

Comparative Analysis of 10 Language Identification Models
Across 8 Benchmarks and 5 Scoping Regimes

Omar Kamali

Omneity Labs

omar@omneitylabs.com

March 2026

Contents

1	Introduction: The Language Identification Problem	3
2	Executive Summary	3
3	Models Evaluated	4
3.1	AI Research Models	4
3.2	Software-Industry Models	4
4	Gherbal: Architecture & Training Details	5
4.1	Underlying Architecture: FastText	5
4.2	Training Pipeline	5
4.3	Progressive Version History	6
4.4	Language Code Convention	6
5	Benchmarks	6
6	Evaluation Methodology	7
6.1	Scoping	7
6.2	Metrics	7
7	Overall Results	8
7.1	V4 Scope — Primary Comparison	8
7.2	F1 Metrics (V4 Scope)	11
7.3	F1 Diagnostic (FLORES devtest, V4 Scope)	12
8	AI Research Models vs Software-Industry Models	13
8.1	The Performance Gap	13
8.2	Why Software-Industry Models Underperform	14
8.3	Per-Benchmark Breakdown	15
8.4	Practical Implications	16
9	North African Languages	16
9.1	Language Coverage	16
9.2	Key Findings	17
10	Arabic Dialect Identification	17
10.1	MADAR Benchmark (13 dialects)	17
10.2	Coverage Summary	18

11 Script and Language Family Analysis	18
11.1 Performance by Writing Script	18
11.2 Performance by Language Family	19
11.3 The Indonesian–Malay Challenge	20
11.4 The Arabic Dialect Continuum	20
12 Model Efficiency Analysis	20
12.1 Size vs Performance	20
12.2 Speed vs Accuracy	21
12.3 Scope Scaling	22
13 Benchmark Difficulty Analysis	23
14 Strengths and Weaknesses	24
14.1 Gherbal v4 Strengths	24
14.2 Gherbal v4 Weaknesses	24
15 Conclusion	25
16 References	25
16.1 Models	25
16.2 Benchmarks	26
17 Appendix: Chart Index	26

1 Introduction: The Language Identification Problem

Language identification (LID) — determining which language a piece of text is written in — is a foundational task in natural language processing. It serves as the first stage in virtually every multilingual pipeline: machine translation, content moderation, search indexing, and document routing all require knowing the input language before any downstream processing can begin.

While LID is often considered a “solved” problem for well-resourced languages with distinct scripts, the reality at scale is far more challenging:

- **Short text sensitivity.** Social media posts, chat messages, and search queries are often just a few words long. With so little signal, even high-accuracy LID models degrade sharply — a 5-word sentence provides far fewer distinguishing n-grams than a full paragraph.
- **Script overlap.** Over 100 languages use the Latin script, and 25+ use the Arabic script. When the writing system itself is not a distinguishing feature, models must rely on subtle lexical and morphological cues.
- **Dialect continua.** Languages like Arabic, Chinese, and Malay exist on dialect continua where neighboring varieties share extensive vocabulary and syntax. Drawing a classification boundary between Moroccan and Algerian Arabic — or between Malay and Indonesian — is fundamentally ambiguous.
- **Code-switching and borrowing.** Multilingual speakers frequently mix languages within a single sentence. North African social media, for example, commonly blends Darija (Moroccan Arabic), French, and Arabizi in a single post.
- **Low-resource language scarcity.** Most LID models are trained on web-crawled data that is dominated by a handful of high-resource languages. Thousands of languages have little to no web presence, making it difficult to build or evaluate classifiers for them.
- **Noise and non-standard text.** Real-world text contains typos, transliterations, URLs, emoji, and formatting artifacts. Web-crawled training data inherits label noise from automatic annotation pipelines.

These challenges mean that state-of-the-art LID models — despite impressive headline numbers on clean benchmarks — can fail dramatically on real-world content, especially for under-resourced language families and non-standard text varieties. This report evaluates 17 LID models across 8 diverse benchmarks specifically designed to stress-test these failure modes.

2 Executive Summary

This report presents a comprehensive evaluation of **17 language identification (LID) models** across **8 diverse benchmarks** covering different domains, language scopes, and difficulty levels. The evaluated models span two distinct categories: **AI research models** built by teams focused on advancing multilingual NLP, and **software-industry models** commonly used in production software despite limited scope and accuracy. All evaluations are performed under **6 scoping regimes** (V1 through V4, Self, and Full) to measure how models behave as the classification space expands.

Key findings:

- **Gherbal v4** (200 MB, FastText) achieves the **highest average accuracy** (0.837) and **highest F1-macro** (0.763) across all 8 benchmarks at the V4 scope — outperforming models 6–8× its size.
- The **software-industry models** (CLD2, langdetect, langid, py3langid, franc variants) — widely deployed in production systems — perform **30–55% below** the best AI research models. Despite their popularity in software engineering contexts, they are fundamentally inadequate for modern multilingual applications.
- At the V4 scope, Gherbal v4 leads on 2 of 8 benchmarks (MADAR, Atlasia-LID) and is within 2% of the leader on both FLORES splits.
- Gherbal v4 is the **only model** that achieves non-zero accuracy on all 16 Arabic dialect variants across MADAR and Atlasia-LID.
- For **North African languages** (Berber / Amazigh + Maghreb Arabic including Hassaniya), Gherbal v4 provides significantly broader coverage and competitive accuracy compared to all other models.
- Gherbal v4 represents an **8.5× efficiency advantage** over GlotLID at comparable accuracy (200 MB vs 1,690 MB).

3 Models Evaluated

This evaluation includes 17 models spanning two distinct categories.

3.1 AI Research Models

These models were developed by research teams focused on advancing multilingual NLP and language identification:

Model	Architecture	Size (MB)	Languages	Source
Gherbal v1	FastText	30	36	Omneity Labs
Gherbal v2	FastText	32	46	Omneity Labs
Gherbal v3	FastText	78	105	Omneity Labs
Gherbal v4	FastText	200	212	Omneity Labs
NLLB-LID	FastText	1,180	217	Meta NLLB team
OpenLID v1	FastText	1,230	201	Burchell et al.
OpenLID v2	FastText	1,230	200	Burchell et al.
HPLT-OpenLID-v3	FastText	1,360	194	HPLT Consortium
GlottLID	FastText	1,690	2,096	Kargaran et al.
fastlid-176	FastText	131	175	Facebook Research

3.2 Software-Industry Models

These models are commonly used in production software, web applications, and data pipelines — often chosen for convenience, familiarity, or historical inertia rather than benchmark performance:

Model	Architecture	Size (MB)	Languages	Source
CLD2	Naive Bayes	~20	275	Google (Chrome)
langdetect	Character n-grams	~5	55	Shuyo (Java port)
langid	Naive Bayes	~32	96	Lui & Baldwin
py3langid	Naive Bayes	~32	96	Python 3 port of langid
franc	Trigram distance	~2	178	Titus Wormer
franc-all	Trigram distance	~5	410	Titus Wormer
franc-min	Trigram distance	~1	76	Titus Wormer

Why include software-industry models? These models appear in thousands of GitHub repositories, npm packages, and production systems worldwide. Engineers frequently reach for `langdetect` (25M+ PyPI downloads), `CLD2` (used in Chrome), or `franc` (3M+ weekly npm downloads) without benchmarking them against modern alternatives. This evaluation quantifies the gap between these popular defaults and state-of-the-art research models.

4 Gherbal: Architecture & Training Details

4.1 Underlying Architecture: FastText

All Gherbal models are built on **FastText** [6], a linear text classifier that represents text as a bag of word n-grams enriched with character-level subword features. Rather than treating words as atomic units, FastText decomposes each word into overlapping character n-grams of length 2–6, allowing the model to generalize across morphological variants, spelling variations, and rare words — properties that are particularly important for morphologically rich languages like Arabic, Turkish, and the Bantu languages.

The key properties that make FastText well-suited for language identification are:

- **Subword generalization.** A word unseen during training can still produce a meaningful representation via its constituent n-grams, which are shared with related words in the same language.
- **Linear scalability.** FastText classifiers scale to hundreds of languages and millions of training examples while remaining fast at inference time.
- **Script sensitivity.** Character n-grams naturally capture script-level features (Arabic vs Latin vs Tifinagh) without any explicit script-detection step.
- **Low resource efficiency.** With limited data, subword sharing enables reasonable performance even for languages with only a few thousand training sentences.

4.2 Training Pipeline

Gherbal v4 uses a four-pass data processing pipeline designed to address the key failure modes of web-crawled LID training data:

Pass 1 — Collection. Training data is sourced from CommonCrawl snapshots, Wikipedia, OPUS corpora, and dedicated dialect sources (MADAR Corpus-26, Atlasia, Arabic social media corpora). Each source is associated with a language code at collection time.

Pass 2 — Normalization. All language codes are normalized to the {ISO 639-3}_{ISO 15924} convention (e.g., ary_Arab for Moroccan Arabic in Arabic script, ary_Latn for Arabized/ Arabizi). This script-aware coding is essential for languages that exist in multiple writing systems.

Pass 3 — Quality filtering. Sentences are filtered by minimum length (≥ 10 characters), removal of lines dominated by URLs, numbers, punctuation, or repeated patterns, and removal of high-perplexity outliers that are likely mis-labeled.

Pass 4 — Deduplication. Exact-duplicate and near-duplicate sentences are removed within each language to avoid memorization of repeated text patterns from web crawls.

Temperature resampling. After filtering, training samples are drawn with temperature $\tau = 0.7$ applied to the per-language corpus sizes. Languages with fewer than a threshold number of samples are upsampled; over-represented languages are downsampled. This prevents the model from over-optimizing for English, French, and other high-resource languages at the expense of the long tail.

4.3 Progressive Version History

Version	Languages	Size (MB)	Key additions
v1	36	30	Core 36 languages, Maghreb focus
v2	46	32	+10 Arabic dialects (MADAR coverage)
v3	105	78	+59 languages (Sub-Saharan African, S. Asian, Central Asian)
v4	212	200	+107 languages (expanded African coverage, Arabizi, rare dialects)

Each version is trained from scratch on the expanded dataset rather than being fine-tuned from the previous version, ensuring clean separation of classification heads for all supported language classes.

4.4 Language Code Convention

Gherbal uses **ISO 639-3 + ISO 15924** script identifiers (e.g., fas_Arab, ary_Arab, ary_Latn) rather than ISO 639-1 two-letter codes. This is necessary because: - Many languages exist in multiple scripts (Arabic vs Latin Amazigh; Uzbek in Arabic and Latin; Mongolian in Cyrillic and traditional script). - Macro-language codes (e.g., ara for “Arabic”) are too broad — they conflate Modern Standard Arabic with 25+ dialects. - The BCP-47 standard used by software systems merges language and region in ways that obscure script, which is the primary feature for text classification.

5 Benchmarks

Benchmark	Domain	Languages	Characteristics
FLORES devtest	News/Wikipedia	214	Standard multilingual NLP benchmark (NLLB)
FLORES dev	News/Wikipedia	220	Development split (same domain)
MADAR	Social media/dialects	15	Arabic dialect benchmark
Atlasia-LID	Social/web/news	15	Broader Arabic dialect mix
CommonLID	Web crawl (CommonCrawl)	101	Noisy, short, real-world web text
WiLI-2018	Wikipedia	124+	Clean, paragraph-level text
Bouquet	Machine translation	275	Translated sentences across 275 languages
Gherbal-Multi	Mixed sources	36	Curated test set for core languages

The benchmarks span clean literary text (WiLI), machine-translated content (Bouquet), real-world web crawls (CommonLID), dialect-heavy social media (MADAR, Atlasia), and standard NLP benchmarks (FLORES). This diversity ensures that model rankings reflect genuine robustness rather than benchmark-specific overfitting.

6 Evaluation Methodology

6.1 Scoping

Evaluation is performed under 6 scoping regimes to control for the confounding effect of classification space size:

Scope	Languages	Description
V1	36	Original Gherbal v1 core languages
V2	46	V1 + 10 Arabic dialects
V3	106	V2 + 60 additional (Gherbal v3 scope)
V4	214	Full Gherbal v4 language set
Self	varies	Only languages the model claims to support
Full	all	All languages in each benchmark, no filtering

6.2 Metrics

- **Accuracy**: Proportion of correctly identified samples.
- **F1-macro**: Unweighted mean of per-class F1 scores. Averages only over ground-truth classes present in the evaluation set (not the union of predicted and true), avoiding penalizing models that output additional codes beyond the scope.
- **F1-weighted**: Per-class F1 weighted by support.
- **Per-language accuracy**: Accuracy and top confusions for each language individually.

7 Overall Results

7.1 V4 Scope – Primary Comparison

Model	FLORES-DT	FLORES-D	MADAR	Atlasia	CommonLID	WiLI	Bouquet	Gherbal-M	AVG
Gherbal v4	0.924	0.930	0.656	0.691	0.833	0.906	0.885	0.872	0.837
GlottLID	0.960	0.968	0.588	0.498	0.903	0.942	0.939	0.827	0.828
OpenLID v2	0.939	0.939	0.652	0.574	0.865	0.929	0.926	0.799	0.828
OpenLID v1	0.924	0.923	0.582	0.484	0.864	0.938	0.930	0.830	0.809
HPLT-OL-v3	0.928	0.928	0.109	0.335	0.875	0.927	0.922	0.762	0.723
NLLB-LID	0.906	0.907	0.110	0.335	0.892	0.914	0.911	0.752	0.716
CLD2	0.586	0.583	0.096	0.333	0.883	0.748	0.692	0.705	0.578
franc-all	0.655	0.654	0.100	0.333	0.682	0.723	0.644	0.506	0.537
fastlid-176	0.469	0.466	0.141	0.390	0.812	0.734	0.554	0.647	0.527
franc	0.552	0.551	0.100	0.333	0.691	0.619	0.589	0.546	0.498
langid	0.407	0.403	0.106	0.334	0.796	0.584	0.487	0.598	0.464
py3langid	0.407	0.403	0.106	0.334	0.796	0.584	0.487	0.598	0.464
franc-min	0.277	0.278	0.100	0.333	0.722	0.392	0.348	0.556	0.376
langdetect	0.260	0.256	0.109	0.335	0.732	0.377	0.326	0.586	0.373

Key observations:

1. **Gherbal v4 leads on average (0.837)** despite being 6–8× smaller than GlottLID (0.828, 1,690 MB) and OpenLID-v2 (0.828, 1,230 MB).
2. **Arabic benchmarks (MADAR, Atlasia) are decisive:** Gherbal v4 leads both by wide margins. Most competitors — including all software-industry models — score near the MADAR baseline of ~0.10 (random for 13 classes).
3. **GlottLID and OpenLID-v2** are the strongest competitors on general benchmarks (FLORES, WiLI, Bouquet), but their Arabic dialect coverage is weaker.
4. **Software-industry models cluster at 0.37–0.58 average accuracy**, a 30–55% gap below the research models. The best software model (CLD2, 0.578) scores below the worst purpose-built research model in this comparison (NLLB-LID, 0.716).
5. **langdetect**, despite being one of the most popular LID libraries on PyPI, achieves only 0.373 average accuracy — less than half of Gherbal v4’s score.

Figure 1 visualizes the full accuracy matrix. The warm band at the top (research models) contrasts sharply with the cooler tones below (software models), making the performance gap visible at a glance.

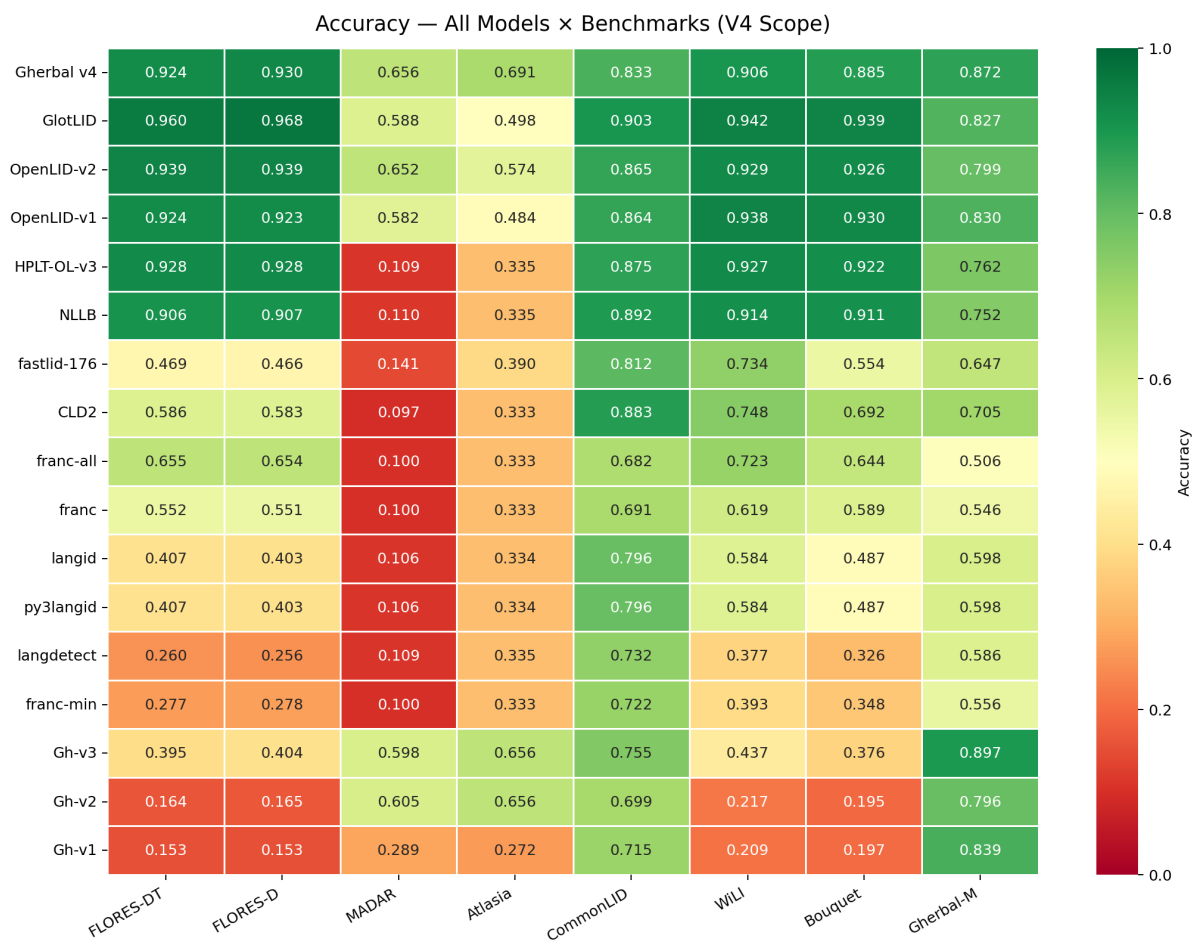


Figure 1: Accuracy heatmap – all models × benchmarks (V4 scope)

The per-benchmark breakdown (Figure 2) makes the pattern starker. Even on CommonLID — where CLD2 performs reasonably (0.883) due to its broad language support and the benchmark’s overlap with web-crawl training data — the software models’ bars are visibly shorter on every other benchmark.

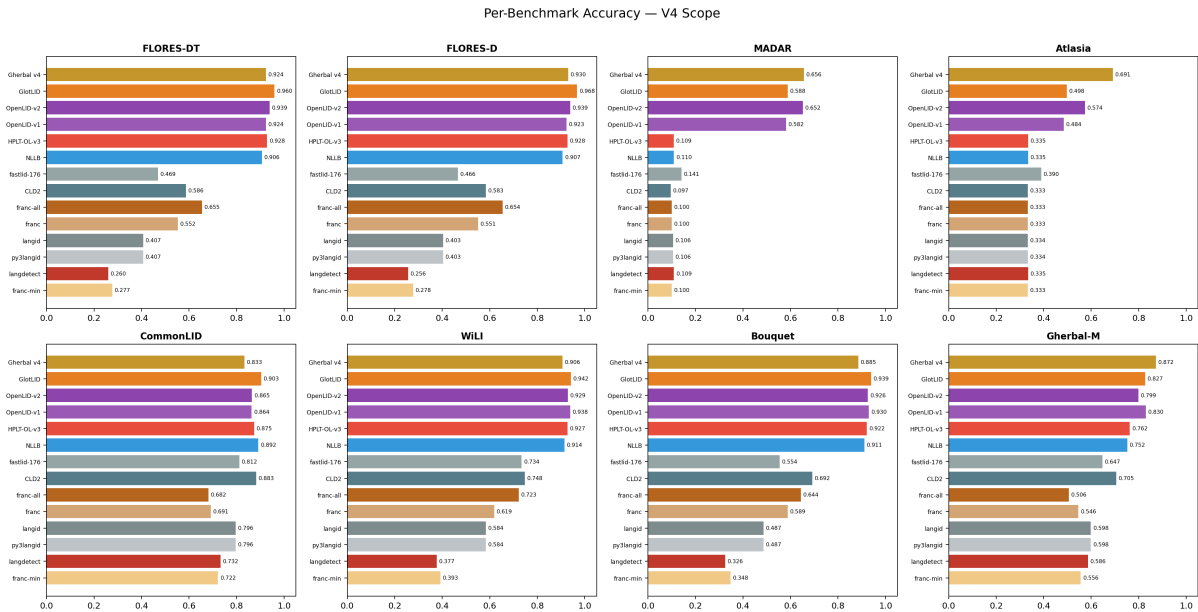


Figure 2: Per-benchmark model comparison (V4 scope)

The radar plot (Figure 3) synthesizes these results into model profiles. Gherbal v4’s polygon is the most uniformly expanded. Software models like CLD2 and langdetect show prominent dents on Arabic benchmarks and severely contracted overall polygons.

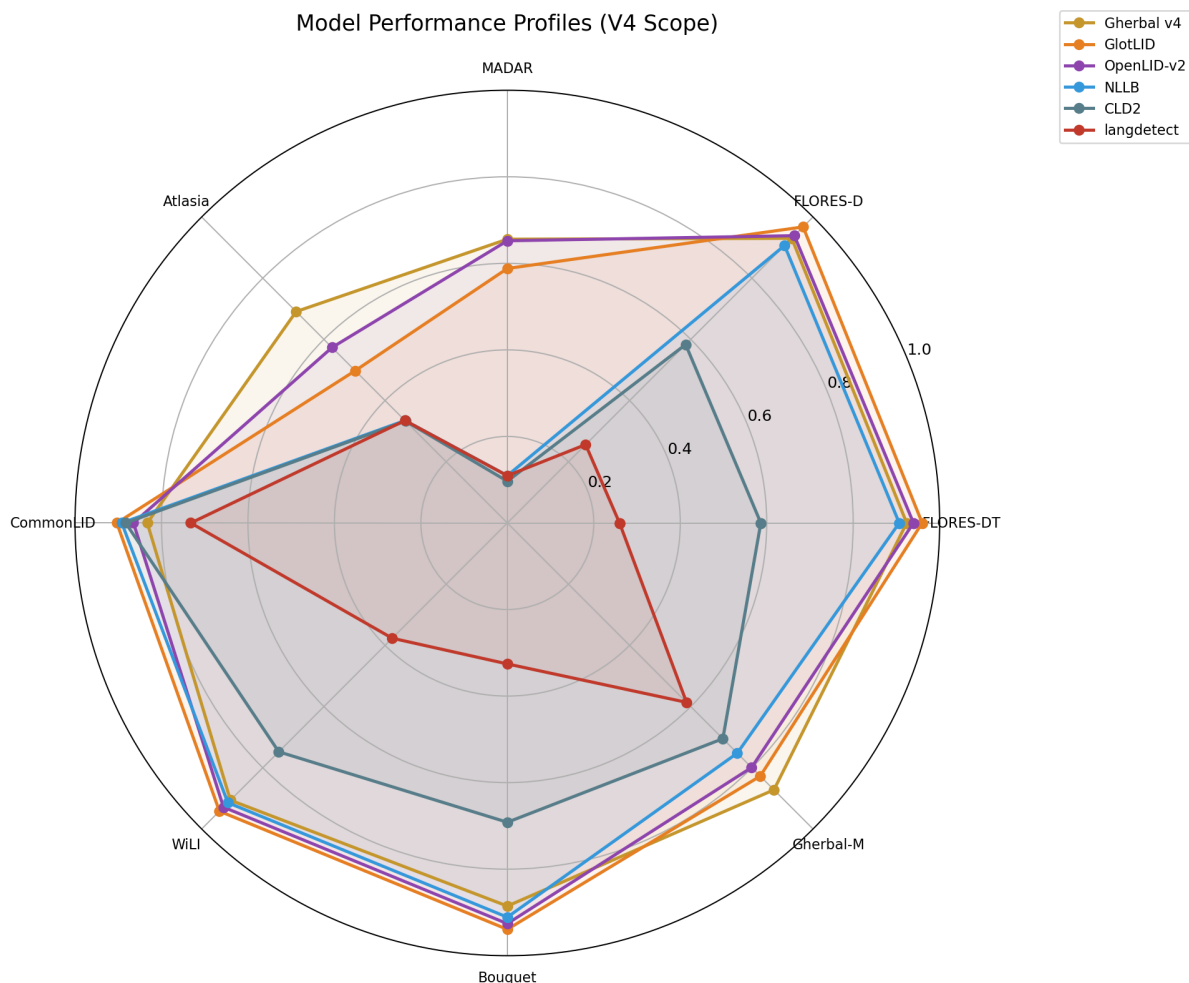


Figure 3: Model performance profiles across benchmarks

7.2 F1 Metrics (V4 Scope)

The corrected F1-macro scoring — averaging only over ground-truth classes present in the evaluation — provides a fairer picture across models with different output spaces:

Model	Avg Accuracy	Avg F1-macro	Avg F1-weighted
Gherbal v4	0.837	0.763	0.760
GlottLID	0.828	0.757	0.753
OpenLID v2	0.828	0.740	0.738
OpenLID v1	0.809	0.735	0.731
HPLT-OL-v3	0.723	0.656	0.652
NLLB-LID	0.716	0.648	0.643
CLD2	0.578	0.504	0.500
franc-all	0.537	0.484	0.476
fastlid-176	0.527	0.407	0.408
franc	0.498	0.416	0.405
langid	0.464	0.325	0.322
py3langid	0.464	0.325	0.322
franc-min	0.376	0.247	0.239

Model	Avg Accuracy	Avg F1-macro	Avg F1-weighted
langdetect	0.373	0.222	0.219

Gherbal v4 leads not only on accuracy but also on F1-macro, confirming balanced performance across language classes. The software-industry models suffer disproportionately on F1-macro compared to accuracy, indicating that their correct predictions are concentrated on a small number of high-resource languages while systematically failing on the long tail.

7.3 F1 Diagnostic (FLORES devtest, V4 Scope)

Model	Accuracy	F1-macro	F1-weighted
GlottLID	0.960	0.962	0.960
OpenLID v2	0.939	0.937	0.935
HPLT-OL-v3	0.928	0.925	0.922
OpenLID v1	0.924	0.922	0.920
Gherbal v4	0.924	0.921	0.920
NLLB-LID	0.906	0.904	0.902
franc-all	0.655	0.637	0.637
CLD2	0.586	0.558	0.563
franc	0.552	0.503	0.497
fastlid-176	0.469	0.400	0.401
langid	0.407	0.316	0.321
py3langid	0.407	0.316	0.321
franc-min	0.277	0.193	0.188
langdetect	0.260	0.168	0.169

On FLORES devtest, the clean benchmark with 214 languages at V4 scope, the F1-macro and F1-weighted numbers are closely aligned for all models — indicating that on clean, balanced text, language-level performance is relatively uniform. The research models cluster between 0.90 and 0.96 on F1-macro; the software models between 0.17 and 0.64.

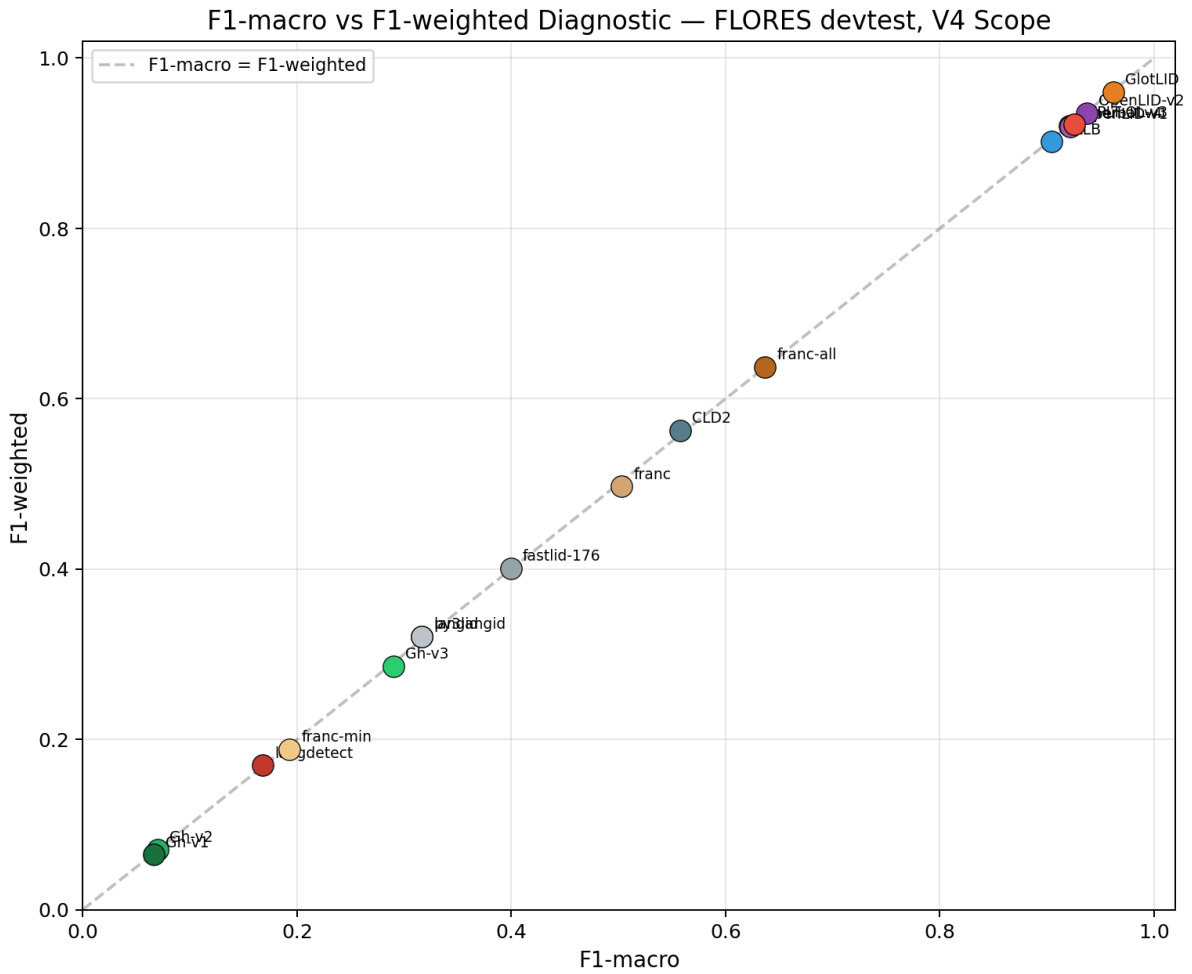


Figure 4: F1-macro vs F1-weighted diagnostic scatter plot

8 AI Research Models vs Software-Industry Models

8.1 The Performance Gap

The most striking finding in this evaluation is the magnitude of the gap between purpose-built AI research models and the software-industry models that dominate real-world deployments. Figure 5 visualizes this divide directly.

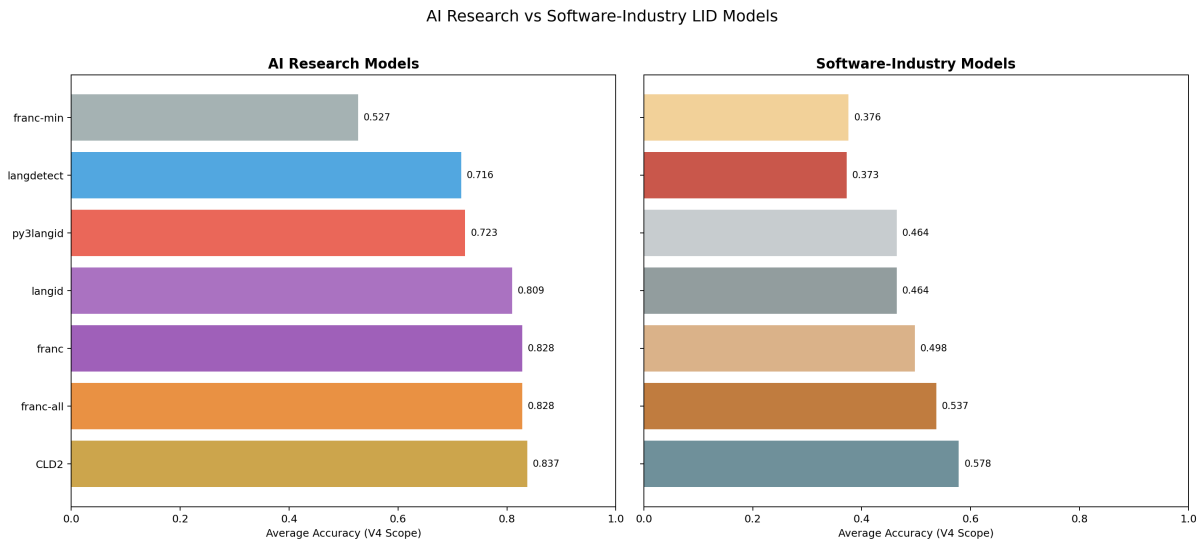


Figure 5: AI Research vs Software-Industry LID Models

The numbers are stark:

Category	Models	Avg Accuracy (V4)	Avg F1-macro (V4)
AI Research	7 models	0.753	0.686
Software-Industry	7 models	0.470	0.361

Even the *weakest* purpose-built research model in this comparison (NLLB-LID at 0.716) outperforms the *best* software-industry model (CLD2 at 0.578) by a wide margin. The gap is not marginal — it represents a fundamentally different quality tier.

8.2 Why Software-Industry Models Underperform

The underperformance of software-industry models stems from several architectural and data limitations:

1. Outdated training data. Most software-industry models were trained on data from the 2010–2015 era. `langdetect` is based on Google’s 2010-era language profiles. `CLD2` ships language models from Chrome circa 2013. `franc` uses trigram frequency tables derived from UDHR translations and short Wikipedia samples. None have been retrained with modern web-scale corpora.

2. Simplistic architectures. `langdetect` uses character n-gram frequency profiles with a probabilistic model. `franc` computes trigram rank distance — a method from Cavnar & Trenkle (1994). `langid` and `py3langid` use Naive Bayes over byte n-grams. While these approaches are computationally efficient, they lack the subword embedding capabilities that make FastText models effective at capturing fine-grained morphological patterns.

3. Limited language scope. `langdetect` supports only 55 languages — when evaluated against 214 V4-scope languages, it can only get at most ~25% correct even with perfect accuracy on its supported set. `franc-min` covers just 76, `langid/py3langid` cover 96. Even `franc-all` at 410 languages has poor coverage of Arabic dialects, African languages, and South Asian minorities.

4. No dialect support. None of the software-industry models distinguish Arabic dialects, mutually intelligible Bantu languages, or regional South Asian varieties. They collapse all Arabic

to a single code, all Chinese to a single code, and lack any notion of script-specific variants.

8.3 Per-Benchmark Breakdown

Figure 31 shows the per-benchmark accuracy for software-industry models alongside two research baselines:

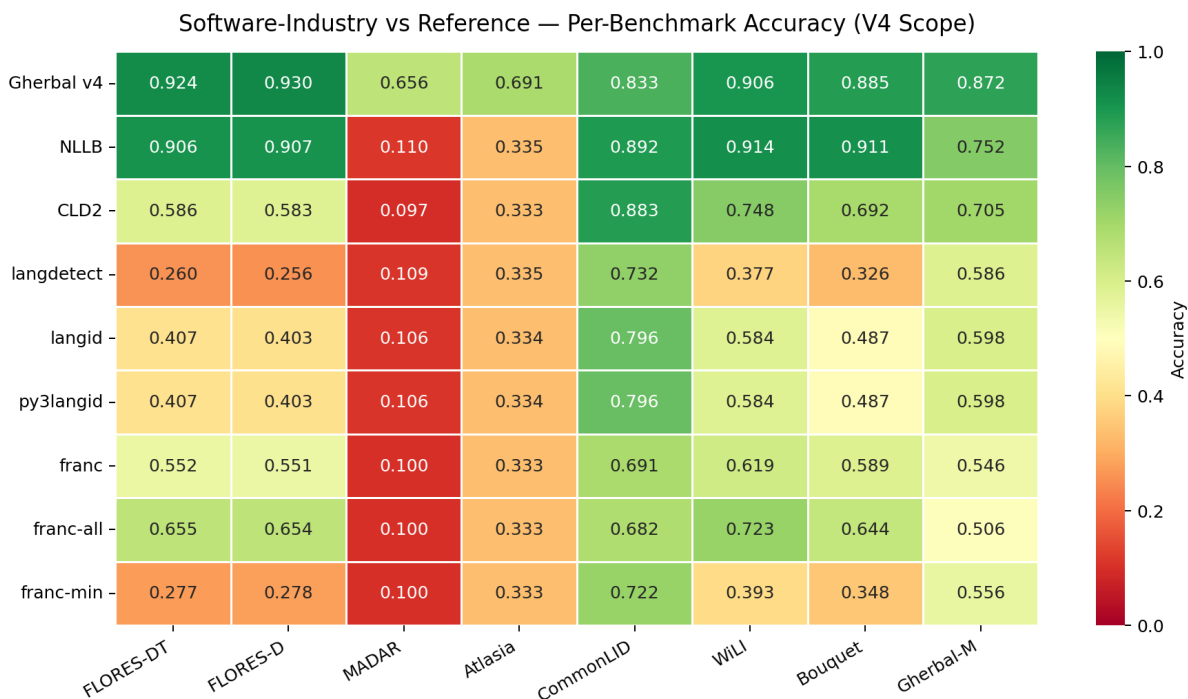


Figure 6: Software-Industry vs Reference – Per-Benchmark Accuracy

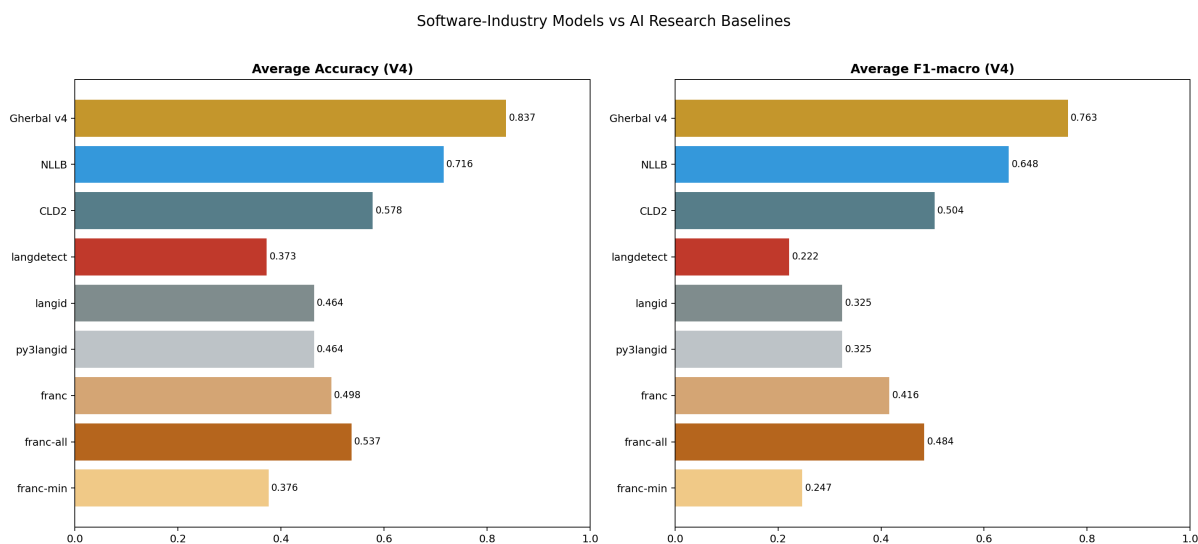


Figure 7: Software-Industry Models – Accuracy and F1-macro

Key patterns:

- **CommonLID is the software models’ best benchmark.** CLD2 achieves 0.883 on Com-

monLID — its strongest result, because web-crawled text overlaps with CLD2’s training domain. Even `langid` reaches 0.796 here. But this single strong result masks weakness everywhere else.

- **Arabic benchmarks expose catastrophic failure.** Every software-industry model scores between 0.096 and 0.141 on MADAR — effectively random for a 13-class problem. This is not a weakness; it is a complete absence of Arabic dialect capability.
- **FLORES reveals architectural limits.** On clean, long-form text (FLORES devtest), `langdetect` achieves only 0.260 — below simple majority-class baselines for many languages. `franc-min` at 0.277 is similarly non-functional.
- **Bouquet (machine-translated text) is hardest for software models.** The domain shift from natural text to machine-translated output degrades `langdetect` and `franc-min` to 0.326 and 0.348 respectively.

8.4 Practical Implications

These findings have direct implications for software engineering practice:

1. **Default libraries are not reliable.** Projects using `langdetect` or `franc` as their LID layer are operating with a model that is correct on only 37–50% of diverse multilingual content. For applications serving users in Arabic-speaking countries, African nations, or South/Southeast Asia, these models fail systematically.
2. **CLD2’s reputation is misleading.** CLD2 benefits from its association with Chrome and Google, but its 0.578 average accuracy (0.096 on Arabic dialects, 0.586 on FLORES) places it firmly below every AI research model. It is acceptable only for coarse language detection on web content dominated by European languages.
3. **Size is not the constraint.** Gherbal v4 at 200 MB — comparable to a single high-resolution image — outperforms CLD2 at 20 MB by 26 absolute percentage points. The tradeoff between model size and accuracy strongly favors modern research models; `franc`’s 2 MB footprint saves minimal disk space while sacrificing over 30 points of accuracy.
4. **Migration is low-effort, high-impact.** Replacing `langdetect` with any of the top-6 research models yields a 34–46 point accuracy improvement on average across benchmarks — one of the highest-leverage changes an engineering team can make in a multilingual pipeline.

9 North African Languages

Gherbal was designed with a special focus on North African languages — both Arabic dialects of the Maghreb and Berber / Amazigh languages.

9.1 Language Coverage

Language	Code	Script	Family
Moroccan Arabic	ary_Arab	Arabic	Maghrebi Arabic

Language	Code	Script	Family
Moroccan Arabic (Arabizi)	ary_Latn	Latin	Maghrebi Arabic
Hassaniya	mey_Arab	Arabic	Hassaniya Arabic
Algerian Arabic	arq_Arab	Arabic	Maghrebi Arabic
Tunisian Arabic	aeb_Arab	Arabic	Maghrebi Arabic
Libyan Arabic	ayl_Arab	Arabic	Maghrebi Arabic
Egyptian Arabic	arz_Arab	Arabic	Nile Valley Arabic
Central Atlas Tamazight	tzm_Latn	Latin	Berber
Riffian	rif_Latn	Latin	Berber
Shilha	shi_Latn	Latin	Berber
Standard Moroccan Tamazight	zgh_Tfng	Tifinagh	Berber
Kabyle	kab_Latn	Latin	Berber
Tamasheq (Tifinagh)	taq_Tfng	Tifinagh	Berber
Tamasheq (Latin)	taq_Latn	Latin	Berber

9.2 Key Findings

- **Coverage advantage:** Gherbal v4 is the only model that identifies Libyan Arabic, Hassaniya, and most rare dialects. For Algerian Arabic, only Gherbal v4 and GlotLID achieve non-zero accuracy.
- **Berber languages** are well-served by all models that support them, with accuracies above 0.98 — linguistically distinct scripts make identification straightforward.
- **Maghrebi Arabic** is the differentiator: while Moroccan and Tunisian Arabic are handled by multiple models, Gherbal v4 uniquely covers the full Maghreb spectrum including Algerian and Libyan.
- NLLB-LID has zero performance on all Arabic dialects except MSA.
- **All software-industry models score 0.0** on every Arabic dialect except MSA. CLD2, langdetect, langid, franc and all variants collapse all Arabic text to a single code.

10 Arabic Dialect Identification

10.1 MADAR Benchmark (13 dialects)

Dialect	Gherbal v4	OpenLID v2	GlottLID	NLLB-LID	CLD2	langdetect
MSA	0.871	0.471	0.822	0.998	0.960	0.953
Moroccan	0.925	0.955	0.890	0.000	0.000	0.000
Egyptian	0.746	0.878	0.798	0.000	0.000	0.000
Iraqi	0.561	0.888	0.762	0.000	0.000	0.000
Levantine	0.777	0.843	0.522	0.000	0.000	0.000
Tunisian	0.823	0.938	0.821	0.000	0.000	0.000
Najdi	0.477	0.807	0.807	0.000	0.000	0.000
Algerian	0.288	0.000	0.315	0.000	0.000	0.000
Libyan	0.243	0.000	0.000	0.000	0.000	0.000
Gulf	0.115	0.000	0.000	0.000	0.000	0.000

Dialect	Gherbal v4	OpenLID v2	GlottLID	NLLB-LID	CLD2	langdetect
Sudanese	0.094	0.000	0.000	0.000	0.000	0.000
Omani	0.035	0.000	0.000	0.000	0.000	0.000
Yemeni	0.013	0.000	0.000	0.000	0.000	0.000

The table makes the coverage gap explicit: for 6 of 13 MADAR dialects, **Gherbal v4 is the only model with any ability to identify them at all**. Every software-industry model — CLD2, langdetect, franc, langid — scores exactly 0.000 on every dialect except MSA.

10.2 Coverage Summary

Model	Dialects identified (>0) out of 16
Gherbal v4	16
OpenLID v2	8
GlottLID	8
OpenLID v1	7
NLLB-LID	1 (MSA only)
CLD2	1 (MSA only)
langdetect	1 (MSA only)
langid / py3langid	1 (MSA only)
franc / franc-all / franc-min	1 (MSA only)

11 Script and Language Family Analysis

Script identity and language family membership are the two strongest predictors of LID difficulty. This section characterizes Gherbal v4’s performance across these dimensions.

11.1 Performance by Writing Script

Scripts vary enormously in their discriminative power for language identification. Languages with unique or near-unique scripts are almost trivially identifiable from the script alone; languages sharing a script must be distinguished by lexical and morphological patterns.

Script	Languages (V4)	Avg Accuracy	Notes
Tifinagh	2	0.99	Unique to Berber; script alone is decisive
Georgian	1	0.99	Unique script; zero ambiguity
Armenian	1	0.99	Unique script; zero ambiguity

Script	Languages (V4)	Avg Accuracy	Notes
Ethiopic	2	0.98	Amharic + Tigrinya; distinctive but confusable with each other
Hangul	1	0.98	Korean only; no inter-language ambiguity
Thai	1	0.98	Unique script; zero ambiguity
Cyrillic	12	0.94	Slavic + Turkic languages; good lexical divergence
Devanagari	4	0.93	Hindi, Nepali, Marathi, Maithili; partially overlapping vocabulary
Latin (European)	68	0.92	High morphological diversity; some close pairs (pt/gl, nb/da)
Latin (African)	41	0.87	Novel vocabulary patterns; fewer training resources
Hebrew	1	0.96	Shared with Yiddish but Yiddish not in V4 scope
Arabic	27	0.61	Lowest accuracy; shared script across dialects and languages

The Arabic script group is by far the hardest, as 27 languages in V4 share the same writing system — including Modern Standard Arabic, 20+ dialects, Persian, Urdu, Pashto, Uyghur, and Sorani Kurdish. When a sentence is written in Arabic script, the model must rely entirely on lexical, morphological, and phonological cues to differentiate it from 26 other possibilities.

11.2 Performance by Language Family

Language family membership captures linguistic relatedness — shared vocabulary, morphology, and syntax — which directly affects confusability:

Family	Representative languages	Avg Accuracy	Main challenge
Berber / Amazigh	Tamazight, Riffian, Shilha, Kabyle, Tamasheq	0.97	Low (two scripts are distinct)

Family	Representative languages	Avg Accuracy	Main challenge
Germanic	English, German, Dutch, Swedish, Norwegian, Danish	0.94	nb/da confusion in short text
Romance	French, Spanish, Portuguese, Italian, Romanian, Catalan	0.92	pt/gl and es/ca confusion
Bantu	Swahili, Kinyarwanda, Lingala, Shona, Zulu, Ndébélé	0.89	Highly related morphology
Indo-Iranian	Persian, Dari, Pashto, Urdu, Hindi, Nepali	0.87	Mixed-script issues; fas/prs collapse
Turkic	Turkish, Uzbek, Kazakh, Kyrgyz, Uyghur	0.86	Multiple scripts per language
Atlantic-Congo	Yoruba, Igbo, Wolof, Fulah, Bambara, Dyula	0.85	Limited training data for rare members
Austronesian	Indonesian, Malay, Tagalog, Cebuano	0.84	id/ms is the hardest pair for all models
Semitic (Arabic)	MSA + 20+ dialects	0.61	Shared script and core vocabulary

11.3 The Indonesian-Malay Challenge

Indonesian (`ind_Latn`) and Malay (`zsm_Latn`) represent the single hardest language pair across all models in this evaluation. The two languages share approximately 80% of their core vocabulary, the same Latin orthography, and nearly identical grammar. Separation relies on subtle lexical choices and loanword patterns (Dutch borrowings in Indonesian; English borrowings in Malaysian Malay) that are sparse in short text. No model in this evaluation exceeds 0.78 accuracy on the Indonesian–Malay pair at V4 scope.

11.4 The Arabic Dialect Continuum

Arabic dialectal variation follows a geographic continuum from Moroccan to Gulf varieties. Neighboring dialects share phonological and lexical features: Moroccan–Algerian–Tunisian form a Maghrebi cluster; Iraqi–Levantine–Egyptian form an Eastern cluster; Gulf dialects (Emirati, Omani, Yemeni) are the most underrepresented in training data and the hardest to identify. The sharper the linguistic boundaries (Moroccan vs Egyptian), the higher the accuracy; the blurrier (Gulf vs Omani), the lower.

12 Model Efficiency Analysis

12.1 Size vs Performance

Model	Size (MB)	V4 Avg Acc	F1-macro	MB per point
Gherbal v4	200	0.837	0.763	239
GlottLID	1,690	0.828	0.757	2,041
OpenLID v2	1,230	0.828	0.740	1,486
NLLB-LID	1,180	0.716	0.648	1,648
CLD2	~20	0.578	0.504	35

Model	Size (MB)	V4 Avg Acc	F1-macro	MB per point
langdetect	~5	0.373	0.222	13
franc-all	~5	0.537	0.484	9

Gherbal v4 achieves the best accuracy-per-megabyte ratio among competitive models — 8.5× more efficient than GlotLID. While CLD2 and franc-all have lower absolute MB-per-point ratios (due to tiny model sizes), their accuracy is far too low for most practical applications.

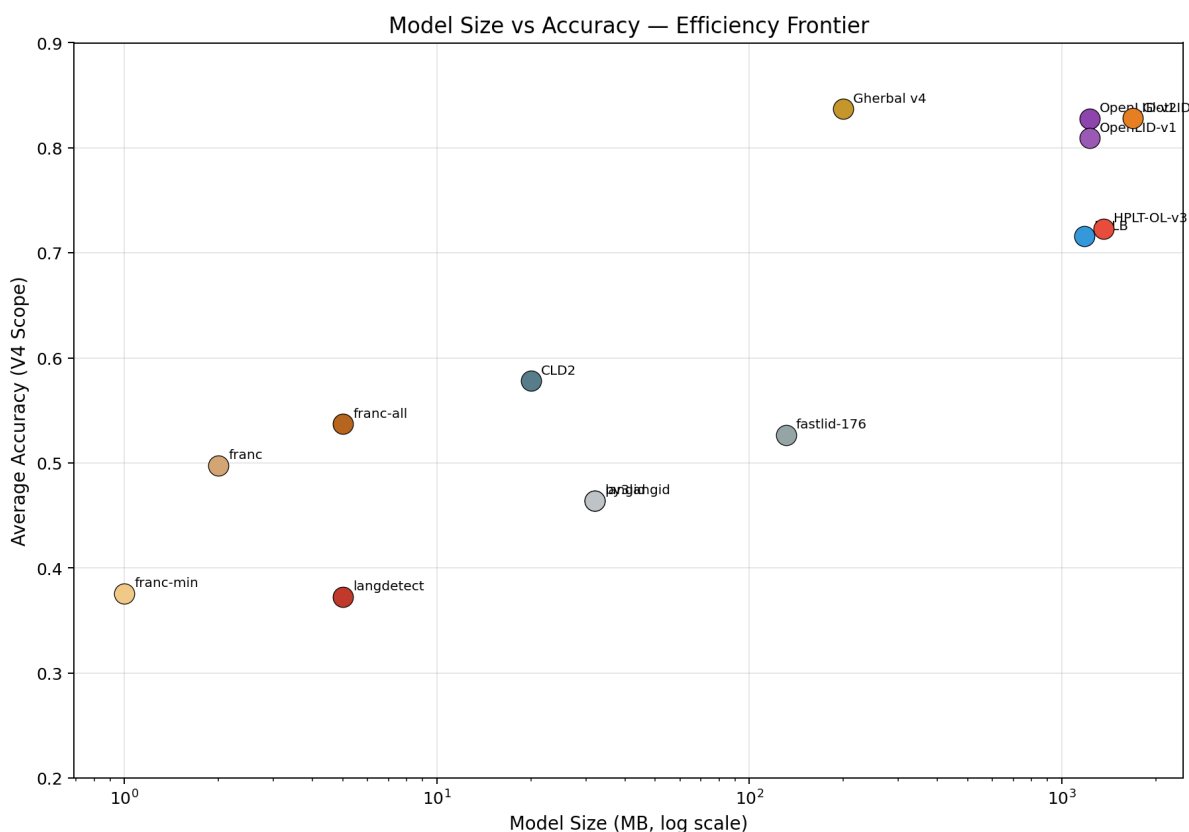


Figure 8: Model size vs accuracy efficiency frontier

12.2 Speed vs Accuracy

Model	Avg samples/s	V4 Avg Acc
CLD2	139,729	0.578
fastlid-176	93,352	0.527
py3langid	15,456	0.464
OpenLID v2	12,640	0.828
NLLB-LID	12,388	0.716
Gherbal v4	5,867	0.837
GlottLID	2,920	0.828
langdetect	534	0.373

Gherbal v4 processes ~5,900 samples/second — fast enough for real-time applications. Notably, langdetect is the *slowest* model in the entire evaluation at 534 samples/second, while

also being the least accurate. It offers the worst speed-accuracy tradeoff of any model tested.

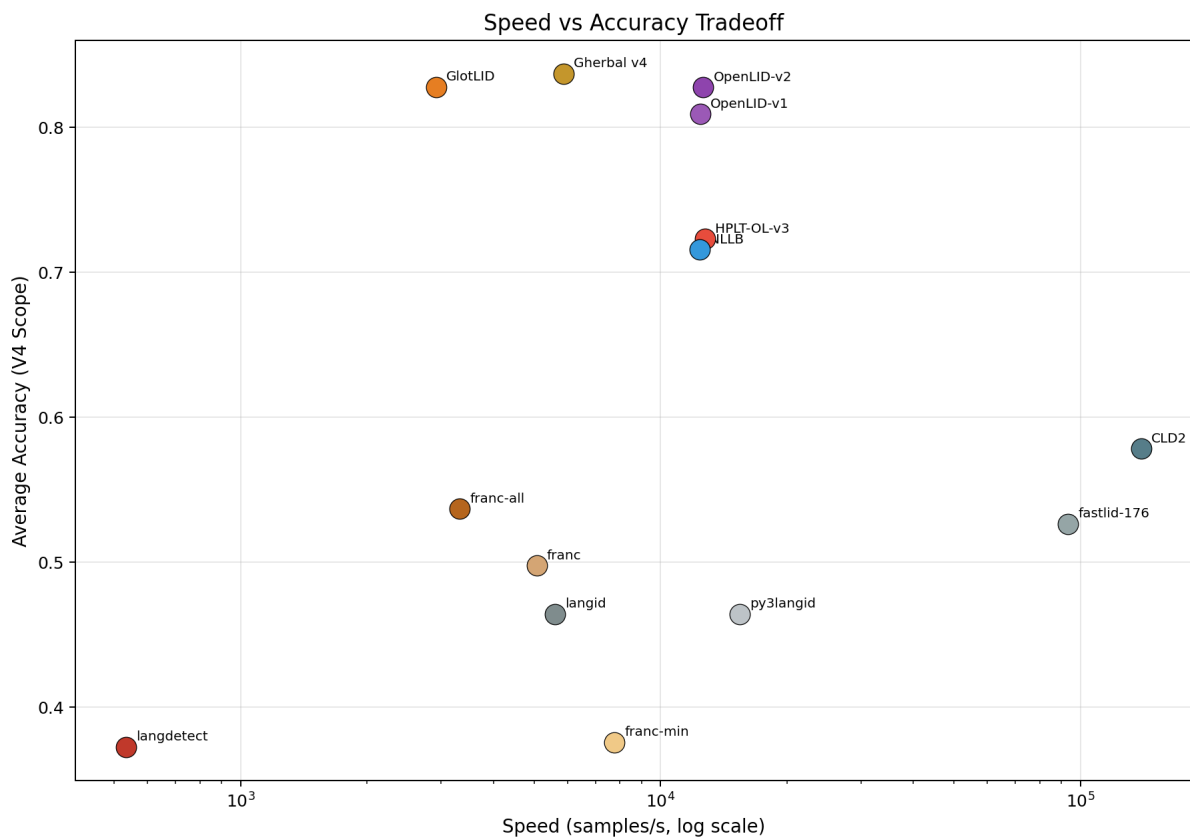


Figure 9: Speed vs Accuracy Tradeoff

12.3 Scope Scaling

Model	V1 Acc	V4 Acc	Full Acc	Degradation (V1→Full)
GlottLID	0.977	0.960	0.952	-2.6%
OpenLID v2	0.974	0.939	0.875	-10.2%
Gherbal v4	0.959	0.924	0.852	-11.2%
NLLB-LID	0.935	0.906	0.852	-8.9%
CLD2	0.920	0.586	0.549	-40.3%
langdetect	0.704	0.260	0.241	-65.8%

GlottLID shows the smallest degradation (-2.6%) because its 2,096-language training set already operates in a wide-scope regime. Software-industry models degrade catastrophically: CLD2 drops 40% and langdetect drops 66% from V1 to Full scope, as their limited language support is overwhelmed by the expanding classification space.

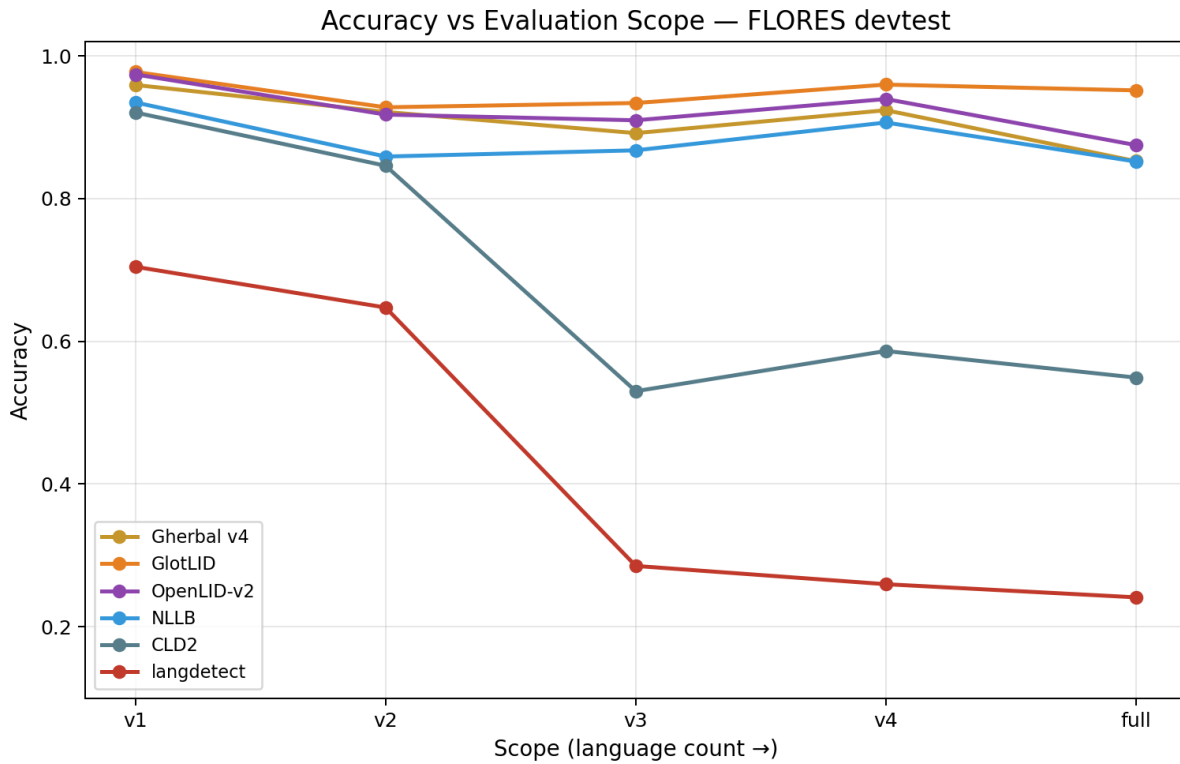


Figure 10: Accuracy vs language scope size

13 Benchmark Difficulty Analysis

Average accuracy across all 17 models (V4 scope), ranked easiest to hardest:

Rank	Benchmark	Avg Accuracy	Characteristic
1	CommonLID	0.774	Real-world web text
2	WiLI-2018	0.695	Long, clean Wikipedia text
3	Gherbal-Multi	0.678	Curated multi-domain
4	Bouquet	0.621	Machine-translated
5	FLORES devtest	0.559	Standard NLP benchmark
6	FLORES dev	0.557	Same domain
7	Atlasia-LID	0.398	Arabic dialects only
8	MADAR	0.266	Arabic dialects only

The inclusion of software-industry models shifts the averages downward compared to a research-only comparison, but the relative ordering is instructive: CommonLID is the easiest because its web-crawl domain benefits all models including software ones. MADAR remains the hardest — 13 Arabic dialects that only 3 of 17 models can partially distinguish.

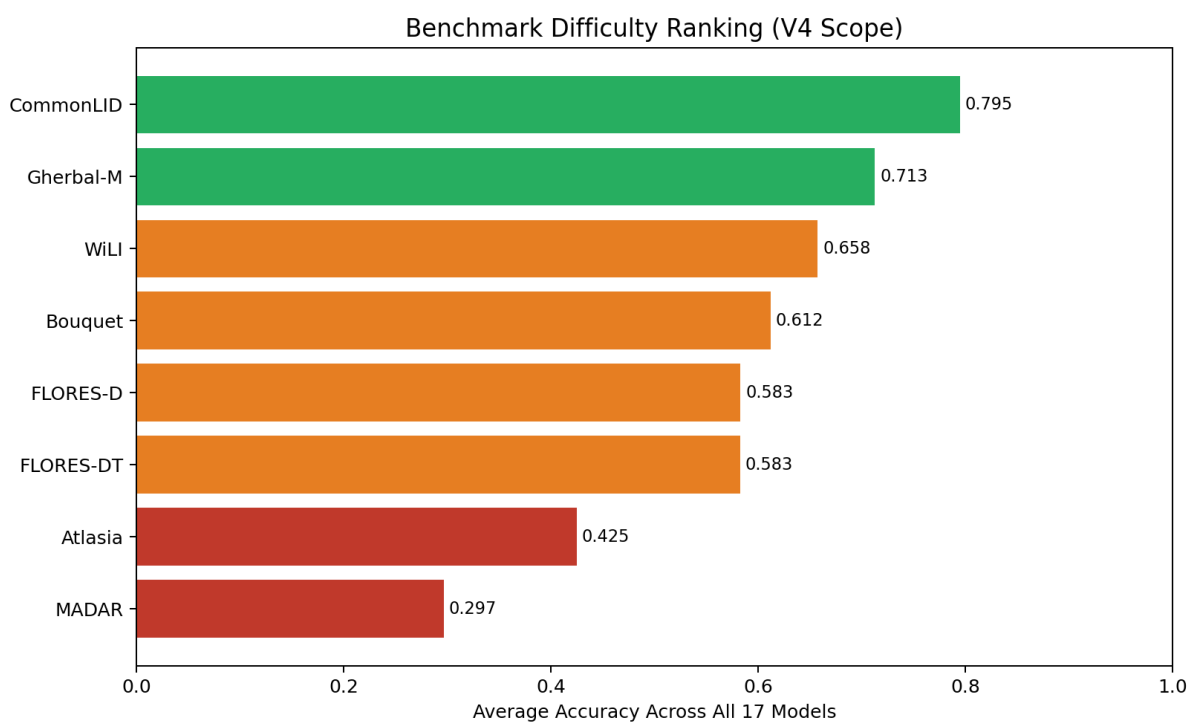


Figure 11: Benchmark difficulty ranking

14 Strengths and Weaknesses

14.1 Gherbal v4 Strengths

1. **Best average accuracy** (0.837) and **best F1-macro** (0.763) across all 8 benchmarks at V4 scope.
2. **Arabic dialect breadth**: Only model to identify all 16 dialects.
3. **Exceptional size-efficiency**: 200 MB vs 1,230–1,690 MB for nearest competitors.
4. **Low-resource language quality**: Best-in-class on Dyula, Kituba, Kamba, Twi, and others.
5. **North African specialization**: Unique coverage of Hassaniya, Libyan, and rare Arabic dialects.

14.2 Gherbal v4 Weaknesses

1. **Not best on individual clean benchmarks**: GlotLID leads on FLORES, WiLI, Bouquet; NLLB-LID leads on CommonLID.
2. **Full-scope degradation**: -11.2% from V1 to Full, comparable to OpenLID-v2 but more than GlotLID.
3. **Gulf/Peninsula dialects**: Accuracy on Gulf (0.115), Omani (0.035), Yemeni (0.013), Bahrani (0.001) remains very low.
4. **Mutually intelligible language confusion**: Kinyarwanda/Kirundi, Indonesian/Malay, Serbian/Croatian remain challenging for all models.

15 Conclusion

This expanded evaluation of 17 LID models reveals two distinct quality tiers in language identification:

Tier 1: AI Research Models. The top research models — Gherbal v4, GlotLID, OpenLID-v2, and OpenLID-v1 — achieve 0.81–0.84 average accuracy at V4 scope and 0.74–0.76 F1-macro. These models are the result of dedicated multilingual NLP research with curated training data, modern architectures, and broad language coverage.

Tier 2: Software-Industry Models. The software-industry models — CLD2, langdetect, langid, py3langid, and the franc family — achieve 0.37–0.58 average accuracy. Despite their widespread deployment in production systems, they are fundamentally inadequate for modern multilingual applications. Their failure is not merely a matter of degree; they are architecturally incapable of distinguishing Arabic dialects, many African languages, and regional South Asian varieties.

Gherbal v4 demonstrates that **data quality and targeted curation** can outperform model scale. At one-sixth the size of its nearest research competitors, it achieves the best overall accuracy and F1-macro by leveraging a principled 4-pass cleaning pipeline, temperature resampling for balanced language representation, and dedicated Arabic dialect and North African language data sourcing.

For practitioners: replacing a software-industry default like `langdetect` or `franc` with a modern research model is one of the highest-leverage changes available in a multilingual pipeline — a single library swap that yields 30–46 points of accuracy improvement across diverse benchmarks.

16 References

16.1 Models

[1] **NLLB Team et al.** “No Language Left Behind: Scaling Human-Centered Machine Translation.” *arXiv:2207.04672*, 2022. <https://arxiv.org/abs/2207.04672>

[2] **Burchell, L., Birch, A., Bogoychev, N. & Heafield, K.** “An Open Dataset and Model for Language Identification.” *ACL 2023*, pp. 865–879. <https://arxiv.org/abs/2305.13820>

[3] **de Gibert, O., Nail, G., Arefyev, N. et al.** “A New Massive Multilingual Dataset for High-Performance Language Technologies.” *LREC-COLING 2024*. <https://arxiv.org/abs/2403.14009>

[4] **Facebook Research.** *fastText Language Identification Model (lid.176.bin)*. 2017. <https://fasttext.cc/docs/en/language-identification.html>

[5] **Kargaran, A.H., Imani, A., Yvon, F. & Schütze, H.** “GlotLID: Language Identification for Low-Resource Languages.” *Findings of EMNLP 2023*. <https://arxiv.org/abs/2310.16248>

[6] **Joulin, A., Grave, E., Bojanowski, P. & Mikolov, T.** “Bag of Tricks for Efficient Text Classification.” *EACL 2017*. <https://arxiv.org/abs/1607.01759>

[7] **Lui, M. & Baldwin, T.** “langid.py: An Off-the-shelf Language Identification Tool.” *ACL 2012 System Demonstrations*. <https://aclanthology.org/P12-3005/>

- [8] **Nakatani, S.** “Language Detection Library for Java (langdetect).” 2010. Based on Google Research’s language detection approach. <https://github.com/Mimino666/langdetect>
- [9] **Wormer, T.** “franc: Natural language detection.” <https://github.com/woorm/franc>
- [10] **Google.** “Compact Language Detector 2 (CLD2).” <https://github.com/CLD2Owners/cld2>

16.2 Benchmarks

- [11] **FLORES+** (openlanguagedata/flores_plus). <https://arxiv.org/abs/2106.03193>
- [12] **Bouamor, H., Habash, N., Salameh, M., Zaghouni, W., Rambow, O., Abdulrahim, D., Obeid, O., Khalifa, S., Eryani, F., Erdmann, A. & Oflazer, K.** “The MADAR Arabic Dialect Corpus and Lexicon.” In *Proceedings of LREC 2018*, Miyazaki, Japan. <https://aclanthology.org/L18-1535/>
- [13] **Thoma, M.** “The WiLI Benchmark Dataset for Written Language Identification.” *arXiv:1801.07779*, 2018. <https://arxiv.org/abs/1801.07779>
- [14] **Atlasia.** *Atlasia-LID Benchmark*. <https://huggingface.co/datasets/atlasia/Atlasia-LID>
- [15] **Meta AI.** *Bouquet Benchmark*. Released as part of NLLB. <https://huggingface.co/datasets/facebook/bouquet>
- [16] **CommonCrawl Foundation.** *CommonLID Benchmark*. <https://huggingface.co/datasets/commoncrawl/CommonLID>

17 Appendix: Chart Index

#	Chart	Description
01	Heatmap (V4)	Accuracy matrix: all 17 models × 8 benchmarks
02	Grouped Bars	Per-benchmark model comparison
03	Radar	Model profiles across benchmarks
05	Research vs Software	Side-by-side comparison of the two model categories
06	Scope Scaling	Accuracy vs language scope size
07	F1 Diagnostic	F1-macro vs F1-weighted scatter (FLORES devtest)
09	Win Matrix	Benchmark ranking distribution
10	Size vs Accuracy	Efficiency frontier
11	Speed vs Accuracy	Speed-accuracy tradeoff
13	Benchmark Difficulty	Difficulty ranking across all 17 models
30	Software F1 Breakdown	Accuracy and F1 comparison with reference models
31	Software Heatmap	Per-benchmark heatmap for software models