Lingua Franca or Probing Artifact? Rethinking Latent Language in Multilingual LLMs

Deniz Bayazit, Badr AlKhamissi, Antoine Bosselut
EPFL
Correspondence: {deniz.bayazit,badr.alkhamissi,antoine.bosselut}@epfl.ch

Abstract

Latent language identification is often used to argue that multilingual language models route computation through language-specific states, such as English pivots. However, existing probes infer latent language from different signals, such as the geometry of hidden states or what can be decoded from intermediate representations. Since such claims shape conclusions about how models share and route information across languages, we ask whether these probes measure the same phenomenon or expose distinct aspects of multilingual computation. We study this question across model families, training regimes, domains, tasks, checkpoints, and up to 27 languages. We find that identification probes systematically disagree: the GMM-based representation probe, which draws evidence from hidden state geometry, shows earlier cross-lingual mixing, whereas decoding-based probes, which rely on output-space decodability, retain sharper language-specific and more English-biased signals. These differences track model multilinguality and training progression, but are comparatively stable across domains. Our results suggest a more cautious interpretation of latent language identification, where current probes expose different aspects of multilingual processing, rather than directly revealing a single internal *lingua franca*.[1]

1 Introduction

Multilingual language models are often assumed to share information across languages by mapping them into common or partially language-neutral representations (Foroutan et al., 2022; Dumas et al., 2025). Yet the form of this sharing remains poorly understood. One influential hypothesis is that multilingual models rely on a *latent language*: an intermediate language-like regime,

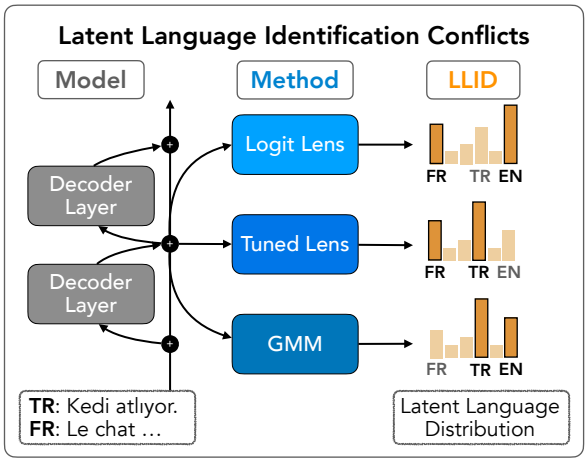


Figure 1: **Do multilingual LLMs work in English? It depends on how you ask.** We probe the same intermediate hidden states of a multilingual LLM with three probes: logitlens, tuned lens, and a representation-based GMM, and recover three different latent language distributions over the task's languages (TR, FR), and other candidate languages like English (EN). We study when and why these probes disagree across models, training regimes, and tasks.

sometimes described as an internal pivot or *lingua franca*, through which the model routes meaning before producing an output. Wendler et al. (2024) provide evidence for such behavior in English-dominated models like LLaMA-2 (Touvron et al., 2023), but newer models trained on more balanced multilingual data may behave differently (Üstün et al., 2024; Foroutan et al., 2025; Schut et al., 2025). This phenomenon raises a broader question: *what part of a latent language estimate reflects the model's internal computation, and what part reflects the assumptions of the diagnostic method*?

Interpreting such probes is challenging because latent language identification (LLID) is inferred rather than directly observed. As illustrated in Fig. 1, a latent language estimate depends on what kind of evidence is extracted from the model. Prior work relies on two broad families (Fig. 2): *representation-based* probes, which infer language

[1]The code is available at: https://github.com/bayazitdeniz/latent-lid

Findings of the Association for Computational Linguistics: EMNLP 2026, pages 10563–10583
October 24-29, 2026 ©2026 Association for Computational Linguistics

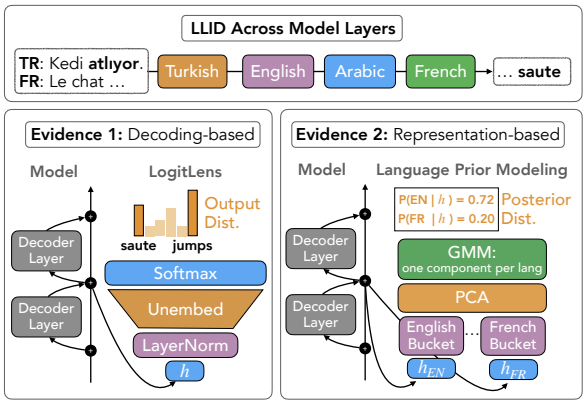


Figure 2: **Two families of latent language identification (LLID) probes.** Given an intermediate hidden state h, *decoding-based* methods (left) project h through the model's unembedding (optionally via a tuned lens) and derive language scores from the resulting vocabulary distribution. *Representation-based* methods (right) instead compare h with language-conditioned activation distributions, here a per-layer GMM with one component per candidate language. The two families operate on the same h but extract different evidence: output-space decodability versus activation-space geometry.

from the geometry of hidden states (Shani and Basirat, 2025), and *decoding-based* probes, which infer language by projecting intermediate hidden states through the model's output head (Wendler et al., 2024; Zhong et al., 2025). These families make different assumptions about what it means for a model to internally use a language, and it is unclear whether they identify a common mechanism or expose different processes in multilingual encoding and decoding.

In this work, we treat LLID as a measurement problem. Instead of asking whether multilingual LLMs "think in English," we ask whether latent language estimates reflect model properties, such as multilinguality and training progression, or diagnostic choices, such as representation geometry versus output-space decodability. To answer this, we compare a GMM-based representation probe and decoding probes across controlled and open-ended multilingual tasks. We first test the internal consistency and generalization of decoding-based LLID across controlled and open-ended settings (§4–5), before asking whether representation-based LLID recovers the same patterns under matched conditions (§6). We then trace how these estimates change across domains, training regimes, and checkpoints.

Our analysis shows that LLID probes should not be treated as interchangeable measurements of a single phenomenon. The GMM representation probe reveals earlier cross-lingual mixing and weaker English dominance, while decoding-based probes retain sharper language-specific and more English-biased signals. This pattern recurs across our analyses, but the layerwise behavior of each probe is model-dependent. For a given probe, where these effects emerge in the layer stack and how they change across layers vary with model multilinguality and over the course of model pre-training, although multilinguality alone does not determine them. By contrast, domain and language-inventory changes shift LLID estimates only modestly. Rather than supporting strong claims about an internal *lingua franca*, each diagnostic provides a partial view of multilingual computation, shaped by the evidence it extracts from hidden states or decoded outputs.

2 A Framework for Comparing Latent Language Probes

2.1 Task Definition: Latent Language Identification

Let $\mathcal{L}$ be a finite set of candidate languages, and let $h \in \mathbb{R}^d$ denote a hidden state extracted from a given model layer and sequence positions. We represent latent language with a variable $Z \in \mathcal{L}$ and estimate its conditional distribution given h as:

$$q(\ell) \approx \Pr(Z = \ell \mid h), \quad \ell \in \mathcal{L}$$

As the variable Z is not directly observed, we treat q as a diagnostic estimate, not as ground-truth evidence of an internal language variable. Different LLID estimators may therefore produce different distributions for the same prompt and layer. In practice, each estimator constructs q by selecting one or more token positions and aggregating the resulting language evidence (§3).

2.2 Two Kinds of Evidence for LLID and Their Limits

Existing LLID methods differ mainly in the evidence they use to construct q. We distinguish two families. Representation-based methods infer language from where a hidden state lies in activation space relative to language-conditioned structure learned from multilingual data. Decoding-based methods infer language from what can be read out from that hidden state through the model's output vocabulary.

Representation-based LLID. This approach compares a hidden state h to language-conditioned structure learned from multilingual data, such as GMM posteriors (Shani and Basirat, 2025), language subspaces (Zhao et al., 2025), or neuron-level indicators (Zeng et al., 2025). This comparison yields per-language scores, which are then aggregated into a layerwise distribution q.

In our experiments, we fit a per-layer GMM with one language-conditioned component per candidate language using hidden states from examples in each language, and use the resulting component posteriors as the layerwise language distribution q.

A common concern for this approach is that learned structure may conflate language identity with correlated factors such as script, tokenization, or domain. In our experiments, however, we find consistent behavior across QA domains such as STEM and Arts & Humanities (§6).

Decoding-based LLID. These approaches estimate latent language by asking what language can be decoded from an intermediate layer. Let $\mathcal{V}$ be the model's vocabulary and let $W_U \in \mathbb{R}^{|\mathcal{V}|\times d}$ be the model's unembedding matrix. Given any hidden state $h \in \mathbb{R}^d$, after final layer normalization, we form an intermediate vocabulary distribution:

$$p := \mathrm{softmax}(W_U h)$$

Applying the model's unmodified output head directly to an intermediate hidden state is known as the logitlens (nostalgebraist, 2020), and we refer to this direct projection as the raw logitlens. Because intermediate representations may not align with those expected by the final output head, we also consider the tuned lens (Belrose et al., 2025), which learns a per-layer affine transformation to predict the final-layer hidden state from an intermediate one before applying the output head.

Prior work maps p to language scores q in several ways, such as summing probability over valid first-token prefixes of language-specific target words (Wendler et al., 2024; Dumas et al., 2025), scoring the probability of multi-token language-specific answer sequences (Zhong et al., 2025), or decoding one or more tokens from p and classifying the result with an external language identification (LID) model (Ozaki et al., 2025). In our experiments, for tasks where the completion is known, we sum the probability over valid first-tokens, whereas if the task is open-ended, we use two rollout-based variants (argmax and top-p) where short continuations decoded from intermediate layers are passed to a LID classifier (Kargaran et al., 2023).

Decoding-based LLID methods assume that the final unembedding remains meaningful at intermediate layers, which may not hold in earlier layers. A further concern is that token-level heuristics such as target-string matching require non-trivial design choices (*e.g.*, gathering valid completions) to yield a meaningful distribution over $\mathcal{L}$.

2.3 Metrics to Compare LLID Estimates

LLID estimates vary along several dimensions that cannot be captured by a single metric: distributions can be confident or diffuse, while their dominant language can match the task-relevant language or pivot to another one. Furthermore, two estimators can agree or disagree. We therefore use a small set of complementary metrics. Unless otherwise stated, each is computed per estimator and averaged over prompts at each layer.

Entropy. Entropy, $H = -\sum_{\ell\in\mathcal{L}} q(\ell)\log q(\ell)$, measures how diffuse the LLID distribution is; lower values indicate a peaky estimate, while higher values indicate mixing across languages.

Dominance. We measure dominance with $D = \max_{\ell\in\mathcal{L}} q(\ell)$, which is high when most probability mass is assigned to a single language.

Pivot rate. Pivot rate is the fraction of prompts whose dominant estimated language differs from the task-relevant language ℓ_x (*e.g.*, the source or target language in translation):

$$P_x = \mathbb{I}\big[\arg\max_{\ell\in\mathcal{L}} q(\ell) \neq \ell_x\big]$$

Agreement. Given two LLID estimators a and b, agreement measures how often the estimators assign the same dominant language:

$$A_x = \mathbb{I}\big[\arg\max_{\ell\in\mathcal{L}} q_a(\ell) = \arg\max_{\ell\in\mathcal{L}} q_b(\ell)\big]$$

In our experiments, this is used to compare decoding-based and representation-based LLIDs.

3 Experimental Setup

3.1 Models & Measuring Multilinguality

We evaluate up to 12 autoregressive language models spanning English-centric baselines, multilingual base models, and instruction-tuned variants. Because training mixtures are not fully available, we consider the following multilinguality proxies: zero-shot multilingual QA accuracy, cross-

lingual perplexity, and byte-normalized likelihood. Unless otherwise stated, cross-model plots order models by increasing zero-shot macro-accuracy on INCLUDE (Romanou et al., 2025). Model details, scores, and rankings are in Appendix B.

3.2 Evaluation Regimes

We evaluate LLID in two settings: (1) a controlled regime where ground-truth completions are available, and (2) an open-ended regime that tests whether the same patterns hold in natural text. The exact language inventory, *i.e.*, the set of candidate languages included in each setting, is listed in Table 1 in Appendix A.

Controlled Target-String Evaluation. We use three synthetic task formats: copy, cloze, and translation. Copy asks the model to reproduce a concept word in the same language, cloze asks the model to complete a masked context, and translation asks for the concept word in a specified target language. These examples build on prior target-word settings for latent language analysis (Wendler et al., 2024; Dumas et al., 2025; Zhong et al., 2025). For each prompt x and candidate language ℓ, let $w_{\ell,x}$ be the language-ℓ answer. Following Wendler et al., let $\textsc{start}_\tau(w_{\ell,x}) \subseteq V$ denote the set of vocabulary tokens under tokenizer τ that can begin $w_{\ell,x}$. For simplicity, we fix x, ℓ, and τ and write $\textsc{start}(w)$. The controlled decoding-based language score is then defined as:

$$s_x(\ell) = \sum_{t \in \textsc{start}(w)} P_\theta(x_{n+1} = t \mid h_n),$$

where h_n is the hidden state at the final prompt token. We refer to this summed mass as the $\textsc{start}(w)$ probability. When a language distribution is needed, we normalize s_x over the candidate set $\mathcal{L}$ to obtain q_x. Construction details for $\textsc{start}(w)$ and candidate language completions are in Appendix A.2.

Open-Ended Evaluation. For natural-language prompts without constrained completions, we summarize both representation-based and decoding-based LLID at the midpoint of the prompt and aggregate over a forward window of five positions. We evaluate on PUD sentence data (Zeman et al., 2017) and INCLUDE question-answering prompts (Romanou et al., 2025), using PUD variants to vary the language inventory and INCLUDE domains to test domain sensitivity. Exact dataset composition and additional evaluation details are in Appendix A.3.

3.3 LLID Estimators

For open-ended evaluation, all estimators produce a layerwise distribution q over candidate languages. We compare:

- `repr`: a representation-based GMM posterior fit to multilingual hidden states;
- `dec-rmax`: decoding-based LLID using deterministic argmax rollouts from intermediate layers;
- `dec-topp`: decoding-based LLID using top-p rollouts.

For `dec-rmax` and `dec-topp`, we generate short continuations from intermediate layers, pass them through GlotLID (Kargaran et al., 2023), and aggregate the resulting per-language scores into q. We evaluate both the raw logitlens and tuned lens defined in §2.2, separating the rollout procedure from the projection used to map intermediate hidden states into the output vocabulary. Fitting details for GMMs, tuned lenses, and GlotLID are in Appendix C.

4 The Controlled Case: Latent Language Under Known Completions

We begin by testing decoding-based LLID in the controlled setting, where the set of valid completions is known across languages and we can directly track when language-specific completions become decodable from intermediate layers. We track the summed $\textsc{start}(w)$ probability assigned to language-specific tokens of the same concept across layers for three models ordered by increasing multilingual capability: Llama-2-7B, Aya-23-8B, and Apertus-8B. Consistent with prior observations on Llama-2-7B, we find that across all model families the decoded language signal is weak in early layers and becomes most visible in later layers across translation (Fig. 3), copy (Fig. 7), and cloze tasks (Fig. 8). However, the sharpness of this signal differs substantially across models with different degrees of multilinguality. The effect is also not necessarily uniform across tasks and languages within a model.

Across models, the copy task yields sharper and higher late-layer probabilities for the observed language. This is expected: copying can rely on reproducing a surface form that is already present in the context. Cloze, on the other hand, requires

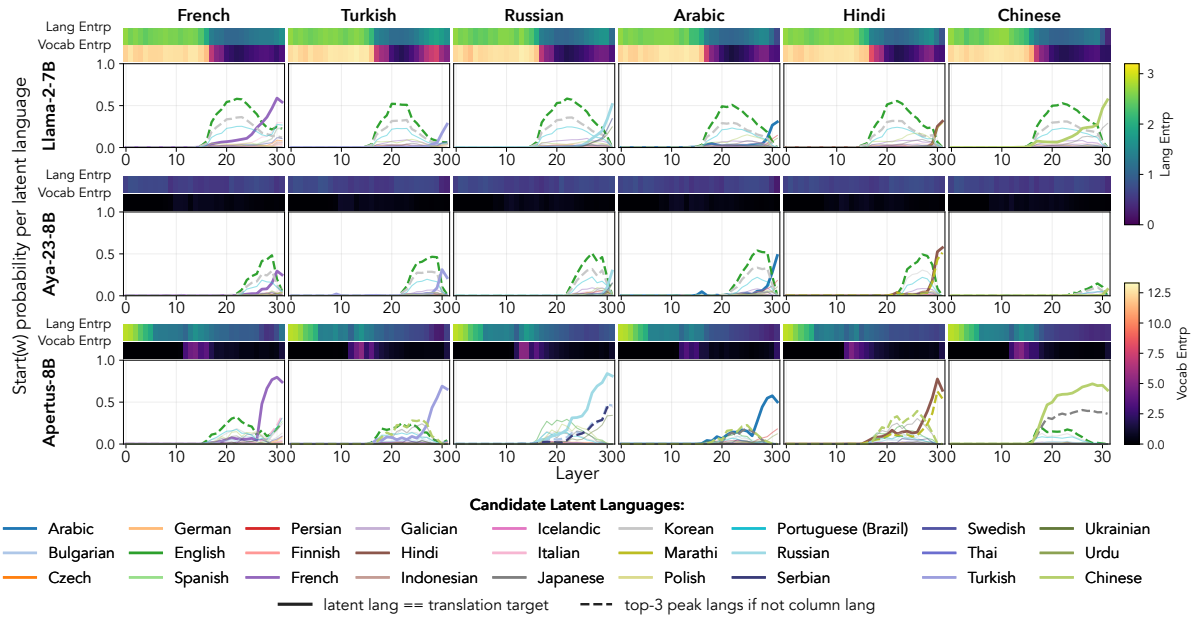


Figure 3: **Latent language probability rises in later layers in controlled translation; English is a stronger competing latent language in Llama-2 than in Aya-23 and Apertus.** For each translation prompt and model layer, we measure the summed next-token probability of all tokens that can begin the corresponding answer in each candidate language ($\textsc{start}(w)$ probability). Rows are models and columns group prompts with the same translation targets. English is excluded as a source or target, but retained as a candidate latent language. Upper strips show average latent language distribution entropy and average first-token vocabulary entropy.

retrieving a concept and lexicalizing it in the appropriate language[2]. Compared with copy, the cloze plots show lower peaks, more competing languages, and more diffuse probability mass, especially for Llama-2 and Aya-23.

Even when English is excluded from the translation direction (Fig. 3), Llama-2 frequently maintains relatively high probability mass on English lexicalizations across layers, suggesting a stronger English-centered bias in its internal representations. This effect is weaker in Aya and further reduced in Apertus, where the target language probability rises relatively earlier and becomes more consistently dominant in later layers.

Taken together, these results show that $\textsc{start}(w)$ recovers a meaningful late-layer language signal in the controlled setting. The diagnostic is strongest when the task makes the relevant surface form directly available, as in copy prompts, and weaker when the model must retrieve and lexicalize a concept, as in cloze prompts. The diagnosis also varies systematically with model multilinguality: less multilingual models more often retain high-probability English alternatives, while the more multilingual Apertus-8B shifts probability toward the intended non-English target earlier.

5 Generalizing Decoding-based Estimates to Open-Ended Generation

The controlled setting above provides a favorable testbed for the $\textsc{start}(w)$ measurement as the relevant cross-lingual completions are known in advance, so the diagnostic can compare a fixed set of lexical alternatives. However, for $\textsc{start}(w)$ to support claims about latent language or multilingual pivoting, its signal should generalize beyond controlled settings with predefined candidate answers. We therefore ask whether similar language preferences can be recovered in open-ended next-token prediction, where the model generates continuations in context rather than selecting among predefined translations.

We extend the decoding diagnostic to open-ended generation by generating short continuations from intermediate hidden states under either the raw logitlens or tuned lens, using argmax or top-p decoding, and then classifying the continuation language. Fig. 4 shows that the projection method changes how probability mass is allocated across languages. Raw logitlens decoding assigns substantially more mass to English than the controlled $\textsc{start}(w)$ setting, suggesting that English lexical-

[2]Expressing an underlying concept using the words/forms of a particular language.

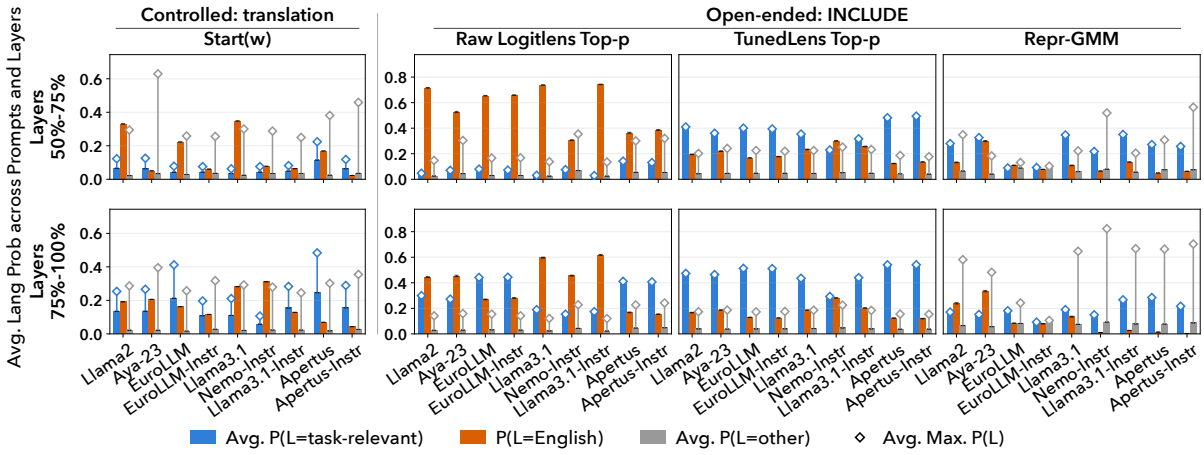


Figure 4: **Later-layer latent language estimates depend strongly on the probe: raw logitlens assigns more probability to English, while tuned lens and GMM estimates favor task-relevant languages.** Rows aggregate the 50–75% and 75–100% layer windows. **Left:** translation prompts using normalized START(w) probabilities. **Right:** open-ended INCLUDE prompts using raw logitlens top-p, tuned lens top-p, and representation-based GMM LLID. Bars pool prompt–layer–language probabilities within each category before averaging; diamonds show the average maximum probability within the task-relevant or other-language category. Error bars show SE. Cases with English as source or target language are excluded.

izations remain accessible when the model is not restricted to known translations. For some models, raw logitlens outputs also become difficult to interpret, given the representational drift between intermediate and final-layer hidden states (Belrose et al., 2025). The tuned lens shifts average probability toward the task-relevant language while average probability assigned to other languages remains low. This suggests that the tuned lens may partly collapse intermediate language evidence toward the final decoded language, consistent with the concern raised by Wendler et al. (2024).

6 Representation vs. Decoding: Different Evidence, Different LLID Behaviors

Having established how decoding-based estimates vary across projections, rollouts, and task settings, we now compare them with representation-based LLID on the same hidden states. Representation-based LLID provides a complementary view of the same intermediate states examined by decoding-based probes. Instead of asking what language can be decoded from a hidden state, it asks which language-conditioned region of activation space the hidden state resembles. This distinction matters because a hidden state may be geometrically closer to one language's cluster without being decodable as that language, and conversely, an intermediate state may decode into a language whose representation-space signature is not dominant. Disagreement between the two estimators should therefore be treated as measurement information rather than as immediate diagnostic failure. Such disagreement provides evidence that the probes capture different aspects of multilingual processing.

Comparing the rightmost column of Fig. 4 (Repr-GMM) with the decoding-based columns on the same open-ended INCLUDE prompts makes this contrast visible. The representation probe assigns substantially more probability mass to the task-relevant language and correspondingly less to English than raw logitlens top-p decoding does, particularly in the 50–75% layer window. This gap is most pronounced for models with moderate multilinguality (*e.g.*, Llama-3.1), where decoding-based estimates still show a strong English bar while the representation-based estimate already favors the task languages. For the most multilingual models (*e.g.*, Apertus, Apertus-Instruct), the average probabilities suggest closer agreement as both probes assign relatively little probability to English and substantial probability to the task-relevant language in late layers. However, the maximum-language estimates also reveal that, under the representation probe, a single non-task, non-English language can still receive more probability than the task-relevant language. These patterns suggest that the two estimator families are sensitive to different aspects of the same hidden states. The GMM representation probe reflects the geometric neighborhood of the

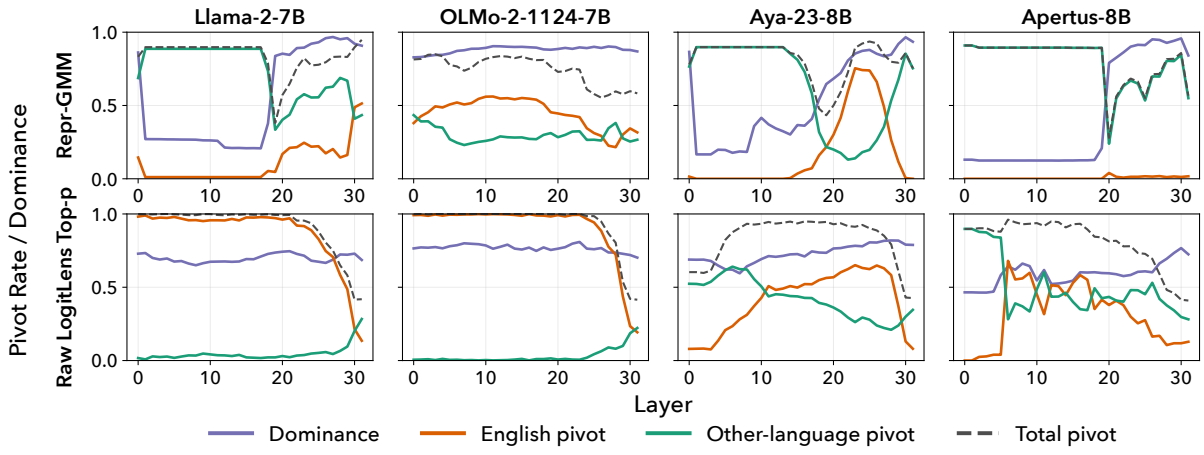


Figure 5: **In open-ended INCLUDE, latent language dominance and pivoting vary across models and probes.** Representation-based estimates exhibit model-specific changes across the layer stack, whereas decoding-based estimates generally shift later, depending on model multilinguality. For each prompt and layer, dominance is the highest probability assigned to a candidate latent language; a pivot occurs when the dominant language differs from the prompt language. Rows are LLID estimators and columns are models. Pivot rates are separated into English and other languages; the dashed line shows their sum. INCLUDE contains no English prompts.

input language's activation, while decoding-based estimators reflect the accessibility of language-specific tokens through the unembedding matrix, which retains a stronger English bias even when the representation probe assigns substantially less probability to English.

The precise layerwise patterns are also model-dependent (Fig. 5). For the representation probe, Llama-2 exhibits a mixture of English- and other-language pivots, Aya-23 shows a pronounced but temporary increase in English-directed pivots, while Apertus-8B pivots are almost entirely to non-English languages. OLMo-2 shows another distinct pattern, maintaining high representation-probe dominance across nearly the entire layer stack. By contrast, dominance drops sharply near the start and recovers through the middle-to-late layers for Llama-2 and Aya-23, while Apertus-8B remains low until a late-layer rise.

These patterns reflect again the different evidence used by the two probe families. The representation probe is fit directly to input-language activation distributions, so it is most confident where hidden states preserve a strong language-conditioned geometric signature. As states become more mixed or task-oriented, they are less cleanly assigned to a single fitted language distribution, producing the early dominance drop in Llama-2 and Aya-23 and the low early-to-middle dominance in Apertus. Their later recovery indicates that the states become concentrated again in a language-conditioned region. OLMo-2 is the exception, as its high dominance suggests a persistent separation of language-conditioned activation regions rather than a pronounced early-to-mid layer mixing phase. Decoding-based estimators, on the other hand, generally change much later in the network.

Across Domains. We next ask whether the model and estimator-specific trajectories in Fig. 5 persist across language inventories and QA subject domains. Fig. 16 repeats the dominance and pivot-rate analyses across PUD language-set variants and INCLUDE domains. For a given model and LLID estimator, the curves largely preserve the same layerwise shape across settings. Thus, domain and language inventory changes affect absolute levels less than model or diagnostic choice. The clearest systematic exception is candidate-inventory size: PUD21 generally yields lower dominance and higher pivot rates than PUD9, as expected when probability mass is distributed over more possible languages.

Across Pretraining. We ask whether latent language estimates appear suddenly, consolidate gradually, or drift as the model trains. Fig. 6 compares OLMo-2-7B and Apertus-8B across checkpoints (see Table 3 for checkpoint alignment), tracking representation and decoding dominance, confident pivot rate, and estimator agreement in early, middle, and late layer bins. We use confident pivot rate to avoid treating small fluctuations in diffuse

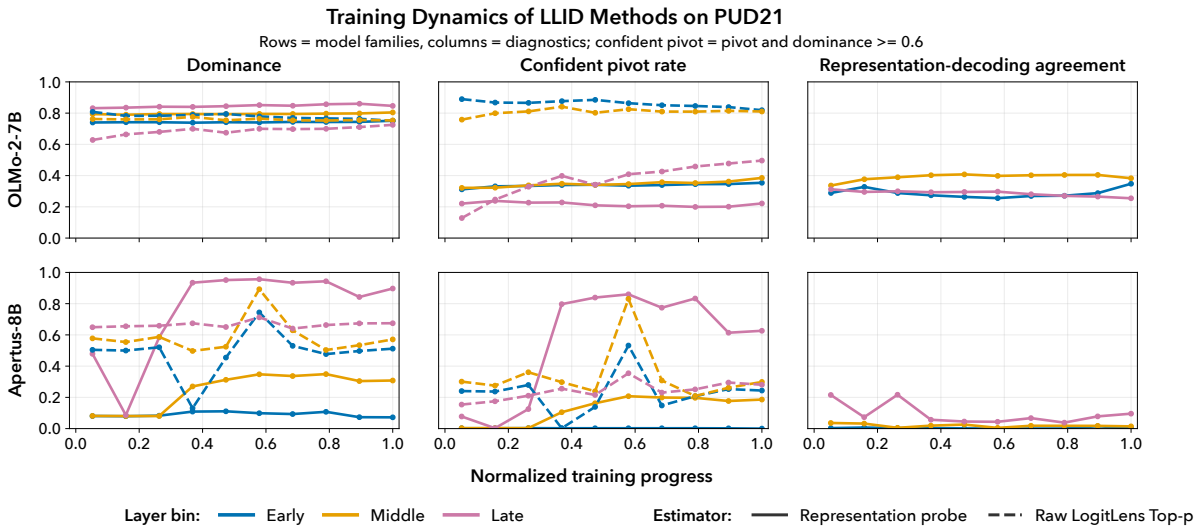


Figure 6: **Training dynamics of latent language estimates across pretraining on PUD21.** Rows are model families and columns are metrics. Colors denote early, middle, and late layer bins; solid and dashed lines denote `repr` and `dec-topp`, respectively. Dominance is the average top-language probability; confident pivot rate is the fraction of prompts where a non-task language is dominant with probability at least 0.6; agreement measures whether the two probes select the same dominant language. The x-axis is normalized training progress, spanning $\sim$210B to 4T training tokens. Layer bins use normalized layer index j/J: early $j/J < 0.25$, middle $0.25 \leq j/J < 0.75$, and late $j/J \geq 0.75$. Most checkpoint-dependent changes occur in late layers, while representation–decoding agreement remains low across training.

LLID distributions as meaningful pivots, retaining only cases where a non-task language is dominant with probability at least 0.6. The two models show qualitatively different trajectories. For OLMo, representation-based dominance is high and stable from the earliest checkpoint, while decoding dominance consolidates over the first $\sim$20–30% of training; confident pivoting is low for `repr` but moderate and slowly increasing for `dec-topp`, consistent with English-centric training where the representation space separates languages early but the decoding pathway retains an English bias. For Apertus, both estimators undergo a sharp transition between roughly 20% and 40% of training, after which late-layer dominance reaches $\sim$0.9. Representation–decoding agreement remains low throughout training for both models, and is especially low for Apertus. These dynamics suggest that most of the checkpoint-dependent changes occur in the late-layer bin for both models. Overall, the checkpoint analysis shows that representation- and decoding-based estimates of LLID behavior can change at different points in training.

Base vs. Instruct. Finally, we ask whether instruction tuning increases English dominance in LLID estimates. This hypothesis is plausible because instruction data is often less multilingual than pretraining data, so instruction tuning might reorient intermediate states or decoded continuations toward English. In Fig. 4, base and instruction-tuned variants show only modest differences in their average LLID probabilities. The layerwise comparisons in the Appendix (Figs. 20–21) likewise show broadly similar pivot and entropy trajectories, although some model-specific deviations remain. Contrary to our initial expectation, instruction tuning produces little systematic shift toward English in LLID estimates, and its overall effects are considerably smaller than the changes observed across pretraining checkpoints.

7 Related Work

Shared and language-specific multilingual structure. Multilingual models maintain both shared and language-specific structure in hidden states, through related but distinguishable subspaces (Chang et al., 2022), language-neutral subnetworks (Foroutan et al., 2022), and language-specific neurons or components (Zeng et al., 2025; Zhao et al., 2025). These findings provide a basis for representation-based LLID, while highlighting that hidden states can simultaneously encode shared and language-specific structure.

Latent pivots and output language. A second line of work studies whether multilingual models internally route computation through a pivot language. Decoding-based probes find English-pivot behavior in English-dominated models and different pivoting patterns under non-English-centric training (Wendler et al., 2024; Zhong et al., 2025). Causal interventions further separate output language from conceptual content, suggesting that language form and meaning can be partially disentangled in the residual stream (Dumas et al., 2025). Our work differs by asking whether the probes used to infer such latent language behavior agree on the same hidden states.

Training dynamics of multilingual representations. Recent work studies how multilingual abilities emerge during training rather than only at the final checkpoint, showing that cross-lingual abilities (Blevins et al., 2022) and internal linguistic features (Bayazit et al., 2026) develop, stabilize, or change over pretraining. Accordingly, we examine LLID across training checkpoints: if latent language estimates reflect model properties rather than diagnostic artifacts, they should vary systematically with training progression.

8 Conclusion

We reframed latent language identification as a measurement problem, comparing representation-based and decoding-based probes across models, training regimes, and tasks. The GMM representation probe and decoding-based probes disagree systematically: the former identifies language structure earlier and with weaker English bias, while decoding pathways retain sharper, more English-favored signals. These differences vary systematically with multilinguality and training progression, but their exact layerwise trajectories remain model and lens-dependent and cannot be predicted from multilinguality alone. Domain and language-inventory changes, by contrast, shift LLID estimates only modestly. Rather than confirming a single internal *lingua franca*, existing probes expose complementary aspects of multilingual computation, showing that the choice of probe is itself part of the claim.

Acknowledgements

We thank Clara Meister, Negar Foroutan, Angelika Romanou, and Ayush K. Tarun for their helpful discussions and feedback on our manuscript. We also gratefully acknowledge the support of the Swiss National Science Foundation (No. 215390), the AI2050 program at Schmidt Sciences (Grant #G-25-69783), Sony Group Corporation, and the Swiss National Supercomputing Center (CSCS) in the form of an infrastructure engineering and development project. This research is with support from Google.org and the Google Cloud Research Credits program for the Gemini Academic Program. This project is also funded by the European Union (ERC, RESPECT-LM, 101222478). Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council. Neither the European Union nor the granting authority can be held responsible for them.

Limitations

Our findings come with several limitations. We evaluate decoder-only models in the 7–9B parameter range, and our checkpoint analysis is necessarily restricted to OLMo-2 and Apertus, the two model families that release intermediate training checkpoints publicly. Whether probe disagreement persists at other scales, or for non-decoder architectures, remains an open question. The precise layerwise LLID patterns also vary across model families, so the trajectories observed here should not be assumed to generalize unchanged to other models. We compare only two probe families, of which the representation-based family is instantiated only with the GMM estimator (Shani and Basirat, 2025), and do not examine causal interventions (Dumas et al., 2025) or neuron-level attribution (Zeng et al., 2025), which may align with either family or expose a third aspect of multilingual processing. Because latent language is inferred rather than observed (*i.e.*, no ground truth), our work documents probe disagreement but cannot adjudicate which (if any) best reflects internal computation. Probe-specific choices (*e.g.*, GMM components, the tuned lens fitting corpus, the open-ended anchor position, and the concept-word construction) may shift absolute LLID values, though we expect qualitative trends to be robust. Finally, our 27-language inventory excludes very low-resource languages and formal languages such as code and mathematics. We also leave open whether probe disagreement predicts downstream failure modes in cross-lingual transfer or generation.

Ethics Statement

This work uses publicly available pretrained language models and multilingual evaluation datasets (PUD/UD, INCLUDE, and Fineweb2), used in accordance with their respective licenses. Our analysis is diagnostic in nature and does not involve training new models, collecting human data, or deploying systems in user-facing settings. We see no direct ethical concerns arising from this work.

References

Viraat Aryabumi, John Dang, Dwarak Talupuru, Saurabh Dash, David Cairuz, Hangyu Lin, Bharat Venkitesh, Madeline Smith, Kelly Marchisio, Sebastian Ruder, Acyr Locatelli, Julia Kreutzer, Nick Frosst, Phil Blunsom, Marzieh Fadaee, Ahmet Üstün, and Sara Hooker. 2024. Aya 23: Open weight releases to further multilingual progress. *Preprint*, arXiv:2405.15032.

Deniz Bayazit, Aaron Mueller, and Antoine Bosselut. 2026. Crosscoding through time: Tracking emergence & consolidation of linguistic representations throughout LLM pretraining. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1353–1377, San Diego, California, United States. Association for Computational Linguistics.

Nora Belrose, Igor Ostrovsky, Lev McKinney, Zach Furman, Logan Smith, Danny Halawi, Stella Biderman, and Jacob Steinhardt. 2025. Eliciting latent predictions from transformers with the tuned lens. *Preprint*, arXiv:2303.08112.

Terra Blevins, Hila Gonen, and Luke Zettlemoyer. 2022. Analyzing the mono- and cross-lingual pretraining dynamics of multilingual language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3575–3590, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tyler A. Chang, Zhuowen Tu, and Benjamin K. Bergen. 2022. The geometry of multilingual language model representations. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 119–136, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Clément Dumas, Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2025. Separating tongue from thought: Activation patching reveals language-agnostic concept representations in transformers. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 31822–31841, Vienna, Austria. Association for Computational Linguistics.

Negar Foroutan, Mohammadreza Banaei, Rémi Lebret, Antoine Bosselut, and Karl Aberer. 2022. Discovering language-neutral sub-networks in multilingual language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7560–7575, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Negar Foroutan, Paul Teiletche, Ayush Kumar Tarun, and Antoine Bosselut. 2025. Revisiting multilingual data mixtures in language model pretraining. *Preprint*, arXiv:2510.25947.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Alejandro Hernández-Cano, Alexander Hägele, Allen Hao Huang, Angelika Romanou, Antoni-Joan Solergibert, Barna Pásztor, Bettina Messmer, Dhia Garbaya, Eduard Frank Ďurech, Ido Hakimi, Juan Garcia Giraldo, Mete Ismayilzada, Negar Foroutan, Skander Moalla, Tiancheng Chen, Vinko Sabolčec, Yixuan Xu, Michael Aerni, Badr AlKhamissi, and 83 others. 2026. Apertus: Democratizing open and compliant LLMs for global language environments. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 46877–46955, San Diego, California, United States. Association for Computational Linguistics.

Amir Hossein Kargaran, Ayyoob Imani, François Yvon, and Hinrich Schuetze. 2023. GlotLID: Language identification for low-resource languages. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 6155–6218, Singapore. Association for Computational Linguistics.

Pedro Henrique Martins, João Alves, Patrick Fernandes, Nuno M. Guerreiro, Ricardo Rei, Amin Farajian, Mateusz Klimaszewski, Duarte M. Alves, José Pombal, Nicolas Boizard, Manuel Faysse, Pierre Colombo, François Yvon, Barry Haddow, José G. C. de Souza, Alexandra Birch, and André F. T. Martins. 2025. Eurollm-9b: Technical report. *Preprint*, arXiv:2506.04079.

nostalgebraist. 2020. Interpreting GPT: The logit lens. lesswrong, 2020.

Shintaro Ozaki, Tatsuya Hiraoka, Hiroto Otake, Hiroki Ouchi, Masaru Isonuma, Benjamin Heinzerling, Kentaro Inui, Taro Watanabe, Yusuke Miyao, Yohei Oseki, and Yu Takagi. 2025. Do llms need to think in one language? correlation between latent language and task performance. *Preprint*, arXiv:2505.21458.

Guilherme Penedo, Hynek Kydlíček, Vinko Sabolčec, Bettina Messmer, Negar Foroutan, Amir Hossein

Kargaran, Colin Raffel, Martin Jaggi, Leandro Von Werra, and Thomas Wolf. 2025. Fineweb2: One pipeline to scale them all — adapting pre-training data processing to every language. In *Second Conference on Language Modeling*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Angelika Romanou, Negar Foroutan, Anna Sotnikova, Sree Harsha Nelaturu, Shivalika Singh, Rishabh Maheshwary, Micol Altomare, Zeming Chen, Mohamed Haggag, Snegha A, Alfonso Amayuelas, Azril Hafizi Amirudin, Danylo Boiko, Michael Chang, Jenny Chim, Gal Cohen, Aditya K Dalmia, Abraham Diress, Sharad Duwal, and 38 others. 2025. Include: Evaluating multilingual language understanding with regional knowledge. In *International Conference on Learning Representations*, volume 2025, pages 83291–83322.

Lisa Schut, Yarin Gal, and Sebastian Farquhar. 2025. Do multilingual LLMs think in english? In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*.

Nadav Shani and Ali Basirat. 2025. Language dominance in multilingual large language models. In *Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 137–148, Suzhou, China. Association for Computational Linguistics.

Mistral AI team. 2024. Mistral nemo. https://mistral.ai/news/mistral-nemo. Accessed: 2026.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, and 49 others. 2023. Llama 2: Open foundation and fine-tuned chat models. *Preprint*, arXiv:2307.09288.

Ahmet Üstün, Viraat Aryabumi, Zheng Yong, Wei-Yin Ko, Daniel D'souza, Gbemileke Onilude, Neel Bhandari, Shivalika Singh, Hui-Lee Ooi, Amr Kayid, Freddie Vargus, Phil Blunsom, Shayne Longpre, Niklas Muennighoff, Marzieh Fadaee, Julia Kreutzer, and Sara Hooker. 2024. Aya model: An instruction finetuned open-access multilingual language model. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15894–15939, Bangkok, Thailand. Association for Computational Linguistics.

Evan Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, Nathan Lambert, Dustin Schwenk, Oyvind Tafjord, Taira Anderson, David Atkinson, Faeze Brahman, Christopher Clark, Pradeep Dasigi, Nouha Dziri, Allyson Ettinger, and 23 others. 2025. 2 OLMo 2 furious (COLM's version). In *Second Conference on Language Modeling*.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. 2024. Do llamas work in English? on the latent language of multilingual transformers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15366–15394, Bangkok, Thailand. Association for Computational Linguistics.

Daniel Zeman, Martin Popel, Milan Straka, Jan Hajič, Joakim Nivre, Filip Ginter, Juhani Luotolahti, Sampo Pyysalo, Slav Petrov, Martin Potthast, Francis Tyers, Elena Badmaeva, Memduh Gokirmak, Anna Nedoluzhko, Silvie Cinková, Jan Hajič jr., Jaroslava Hlaváčová, Václava Kettnerová, Zdeňka Urešová, and 43 others. 2017. CoNLL 2017 shared task: Multilingual parsing from raw text to Universal Dependencies. In *Proceedings of the CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 1–19, Vancouver, Canada. Association for Computational Linguistics.

Hongchuan Zeng, Senyu Han, Lu Chen, and Kai Yu. 2025. Converging to a lingua franca: Evolution of linguistic regions and semantics alignment in multilingual large language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10602–10617, Abu Dhabi, UAE. Association for Computational Linguistics.

Weixiang Zhao, Jiahe Guo, Yang Deng, Tongtong Wu, Wenxuan Zhang, Yulin Hu, Xingyu Sui, Yanyan Zhao, Wanxiang Che, Bing Qin, Tat-Seng Chua, and Ting Liu. 2025. When less language is more: Language-reasoning disentanglement makes llms better multilingual reasoners. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pages 38608–38642. Curran Associates, Inc.

Chengzhi Zhong, Qianying Liu, Fei Cheng, Junfeng Jiang, Zhen Wan, Chenhui Chu, Yugo Murawaki, and Sadao Kurohashi. 2025. What language do non-English-centric large language models think in? In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 26333–26346, Vienna, Austria. Association for Computational Linguistics.

Setting	# Langs	# Entries	Languages
PUD9	9	900 train + 900 held-out	Arabic (ar), Czech (cs), English (en), French (fr), Hindi (hi), Icelandic (is), Indonesian (id), Portuguese (pt), Spanish (es)
PUD21	21	2,100 train + 2,100 held-out	Arabic (ar), Russian (ru), Hindi (hi), Chinese (zh), Korean (ko), Japanese (ja), Indonesian (id), German (de), English (en), Icelandic (is), Swedish (sv), Spanish (es), French (fr), Galician (gl), Italian (it), Portuguese (pt), Czech (cs), Polish (pl), Turkish (tr), Finnish (fi), Thai (th)
UD6 extension	6	600 train + 600 held-out	Ukrainian (uk), Bulgarian (bg), Serbian (sr), Urdu (ur), Persian (fa), Marathi (mr)
INCLUDE-10	10	882 prompts	Arabic (90), Spanish (90), Finnish (90), French (81), Hindi (90), Indonesian (90), Portuguese (90), Russian (90), Turkish (90), Chinese (81)
Copy/cloze	6	414 prompts per task	Arabic (ar), Hindi (hi), Chinese (zh), Russian (ru), English (en), French (fr)
Translation	9	16,146 prompts	Arabic (ar), Spanish (es), Finnish (fi), French (fr), Hindi (hi), Indonesian (id), Russian (ru), Turkish (tr), Chinese (zh)

Table 1: **Language inventories used in the experiments.** PUD and UD rows are balanced at 100 train and 100 held-out prompts per language. PUD9+UD6 and PUD21+UD6 add the UD6 extension to the corresponding PUD inventory. The INCLUDE subset is capped at 30 prompts per language-domain cell, but French and Chinese have fewer total prompts because their STEM cells contain only 21 selected examples. Target-string copy and cloze counts are for the six-language main-plot subset; translation counts aggregate the nine target-specific runs.

A Dataset Details

In this section we describe the datasets and the preprocessing steps. Note that we did not collect these datasets ourselves; they may contain personally identifying or offensive content.

A.1 Language Inventory

Table 1 gives the language inventories used across the controlled target-string runs, PUD runs, and INCLUDE runs. In the main text, we use the shorthand PUD9, PUD21, PUD9+UD6, PUD21+UD6 to avoid repeatedly listing these languages. Entry counts refer to prepared prompt rows; for PUD and UD settings, we report the deterministic train and held-out splits separately. The PUD9+UD6 and PUD21+UD6 settings are formed by adding the UD6 extension row to PUD9 or PUD21.

A.2 Controlled Target-String Construction

The controlled copy, cloze, and translation prompts are derived from the target-word setup of prior latent language studies (Wendler et al., 2024; Dumas et al., 2025; Zhong et al., 2025). Each example has a known concept completion in multiple languages, and the logitlens query is made at the final task token, before the answer begins. For instruction-tuned models, formatting may append assistant-control tokens, so we anchor scoring to the end of the original task prefix rather than the final formatted token.

We first build a common-concept grid and then derive copy, cloze, and directed translation prompts from that grid. Copy and translation prompts use four completed in-context examples followed by a query line; cloze prompts use two completed cloze demonstrations followed by the query context. For translation and cloze settings, word translations and cloze contexts are constructed jointly so that the context disambiguates homonyms. We filter examples where the translated cloze answer does not align with the independently translated concept word. We do not remove examples solely because the English and target language forms share a tokenizer prefix.

For each remaining prompt x, candidate language ℓ, and model tokenizer τ, we construct $\textsc{Start}(w)$, the set of vocabulary tokens that can begin the language-ℓ answer $w_{\ell,x}$, as defined in §3.2. This set includes vocabulary tokens with tokenizer-specific prefixes and whitespace variants when applicable. For tokenizers with byte-fallback behavior, we also include valid byte-level starting tokens.

The corresponding score $s_x(\ell)$, also defined in §3.2, is the summed next-token probability assigned to these valid starting tokens. We use this raw summed mass for the probability curves. When computing distributional metrics such as entropy, dominance, or pivot rate, we normalize these scores over the candidate language set:

$$q_x(\ell) = \frac{s_x(\ell)}{\sum_{\ell' \in \mathcal{L}} s_x(\ell')}$$

The language entropy is computed from this normalized distribution q_x, while vocabulary entropy is computed over the model's next-token vocabulary distribution.

A.3 Open-Ended Evaluation Details

For natural-language data without constrained completions, both representation-based and decoding-based LLID are evaluated at a matched prompt anchor. Unless otherwise stated, we use the midpoint of the prompt and aggregate over a forward window of five positions. Decoding-based open-ended LLID uses either deterministic argmax rollouts or top-p rollouts with $p = 0.9$ and five samples. We score each decoded continuation with GlotLID (Kargaran et al., 2023), collapse its labels to the candidate-language inventory, and normalize the resulting distribution. For top-p rollouts, we average these distributions uniformly across samples to obtain q. We use PUD as sentence-level natural-language data. PUD21 is the broadest setting, while PUD9 follows the language subset used by Shani and Basirat (2025). We also evaluate PUD9+UD6 and PUD21+UD6 by adding six Universal Dependencies languages selected for script or family overlap. For domain analyses, we use INCLUDE question-answering prompts in 10 languages and three domains: Social Science, Arts & Humanities, and STEM.

B Model Details

Table 2 reports all evaluated models and multilinguality proxies. Since direct pretraining mixtures are not fully available, we use zero-shot QA accuracy on INCLUDE, cross-lingual perplexity, and byte-normalized likelihood as external proxies for multilingual capability. For the training dynamics analysis, we align OLMo-2 and Apertus checkpoints as shown in Table 3.

The INCLUDE score is zero-shot accuracy from `lm-evaluation-harness` runs on `include-base-44`, macro-averaged over 44 language groups. We use it to sort the models because it directly measures multilingual task performance.

The PUD21 line NLL is a likelihood-based auxiliary score on the PUD21 parallel sentence set. We compute the autoregressive negative log-likelihood of each teacher-forced line, convert it to bits, and average across PUD-only parallel rows in the 21 languages. Lower is better.

The PUD21+UD6 BPB score is a byte-normalized likelihood on the broader PUD21+UD6 corpus. We sum token-level negative log-likelihood over all evaluated lines, convert from nats to bits, and divide by the total number of UTF-8 bytes:

$$\mathrm{BPB} = \frac{\sum_x -\log_2 p_\theta(x)}{\sum_x |\mathrm{utf8}(x)|}$$

This reduces, but does not eliminate, tokenization and script effects; lower BPB is better.

Model	INCLUDE 0-shot macro (↑)	PUD21 line NLL (↓)	PUD21+UD6 BPB (↓)
Apertus-8B-Instruct	**55.0**	169.8	1.19
Apertus-8B (Hernández-Cano et al., 2026)	<u>53.1</u>	**138.3**	**0.97**
Llama-3.1-8B-Instruct	52.8	152.9	1.09
Mistral-Nemo-Instruct-2407 (team, 2024)	52.2	166.7	1.20
Llama-3.1-8B (Grattafiori et al., 2024)	48.9	<u>148.8</u>	<u>1.05</u>
EuroLLM-9B-Instruct	47.8	156.0	1.17
EuroLLM-9B (Martins et al., 2025)	43.3	149.2	1.11
Aya-23-8B (Aryabumi et al., 2024)	40.1	172.0	1.26
OLMo-2-1124-7B (Walsh et al., 2025)	33.8	181.1	1.30
Llama-2-7B (Touvron et al., 2023)	27.7	170.0	1.22
GPT-2 (Radford et al., 2019)	26.2	365.4	2.62
GPT-2-XL	25.9	302.3	2.20

Table 2: **Multilingual Performance.** Models are sorted by decreasing zero-shot macro accuracy on INCLUDE. Line NLL is an auxiliary likelihood score computed on a parallel corpus across languages. BPB is likelihood normalized by UTF-8 byte count, but it remains script-sensitive because byte counts differ across writing systems. Best is bold; second-best is underlined.

Alignment Idx	OLMo-2 tokens	Apertus tokens
1	210B	210B
2	625B	630B
3	1.04T	1.05T
4	1.46T	1.47T
5	1.87T	1.89T
6	2.29T	2.31T
7	2.70T	2.73T
8	3.12T	3.15T
9	3.53T	3.57T
10	3.95T	3.99T

Table 3: **Token-aligned checkpoints,** used in the OLMo-2 and Apertus training dynamics comparison.

C Training Details

Hardware, Packages & Artifacts. We run experiments on NVIDIA A100 80GB GPUs, using one GPU at a time. Each run lasts between one to six hours depending on the dataset size. We use Python 3.10 and use transformers[3], nnsight[4], and datasets[5] packages. We used AI-assisted tools to support code development, and we reviewed and tested the resulting code. For writing, AI tools were used to rephrase the wording of the manuscript.

[3] https://huggingface.co/docs/transformers
[4] https://nnsight.net/
[5] https://huggingface.co/datasets

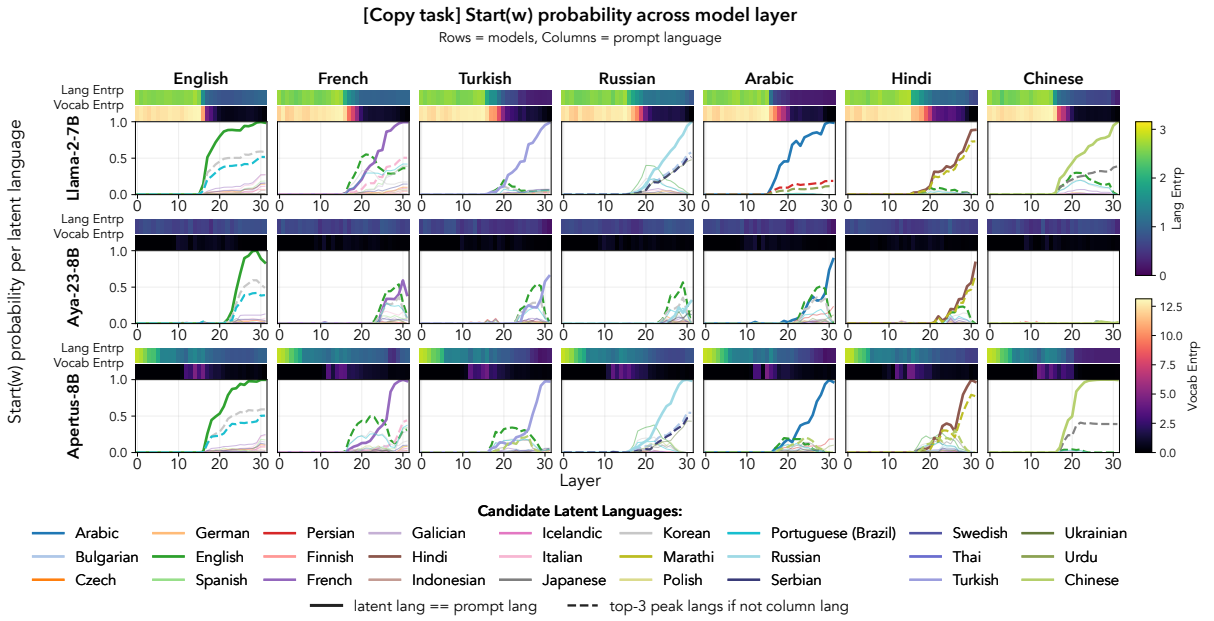


Figure 7: **Copy prompts: latent language probability across model layers.** For each copy prompt and model layer, we measure the summed next-token probability of all tokens that can begin the corresponding answer in each candidate language (START(w) probability). Rows are models and columns group prompts with the same prompt language. Upper strips show average latent language distribution entropy and average first-token vocabulary entropy.

GMM Fitting. Representation-based LLID requires a language-conditioned model of activation space. We fit one per-layer Gaussian model per candidate language from hidden states collected on the train split. Each GMM setup uses at most 100 training prompts per language, uniform language priors, layerwise PCA retaining 98% explained variance, and diagonal covariance. Unless otherwise stated, we fit token-level hidden states and use the surface-token alignment from the dataset preprocessing to select comparable positions at evaluation time. We fit separate GMMs per candidate-language inventory: `pud9`, `pud21`, `pud9_ud6`, `pud21_ud6`, and `include_10lang`. For INCLUDE plots that contain English as a possible latent language, we fit with the same PUD calibration source but add English to the candidate-language set.

Tuned Lens Fitting. For decoding-based LLID, we compare raw logitlens projections with tuned lens projections (Belrose et al., 2025). Tuned lenses are fit on multilingual FineWeb2 data (Penedo et al., 2025), keeping the fitting corpus independent of PUD, INCLUDE, and controlled target-string evaluation prompts. Our main tuned lens artifacts use the 27-language inventory. For each model, FineWeb2 rows are tokenized into windows of length up to 2048 tokens, subject to the model's supported context length. Training examples are balanced by strict round-robin over languages, dropping excess rows from higher-resource languages. We train for one epoch with at most 1000 optimization steps, AdamW with learning rate 10^{-3} and weight decay 0, temperature 1.0, identity and bias regularization weights 10^{-4}, and bfloat16 autocast. Validation uses a lightweight balanced subset, and checkpoints are written periodically. We do not train a tuned lens translator for the final transformer layer; at runtime, the final layer uses the raw model-head projection.

D Additional Results

The appendix figures below are organized by analysis group. For the controlled synthetic tasks, Figs. 7–11 show layerwise START(w) probability curves, while Figs. 12–14 summarize the same task family into language-probability categories across layer windows. For open-ended language probabilities, Fig. 15 gives the PUD21+UD6 counterpart. For training analyses, Figs. 17–19 show checkpoint dynamics and full-layer trajectories, while Figs. 20–21 compare base and instruction-tuned model variants.

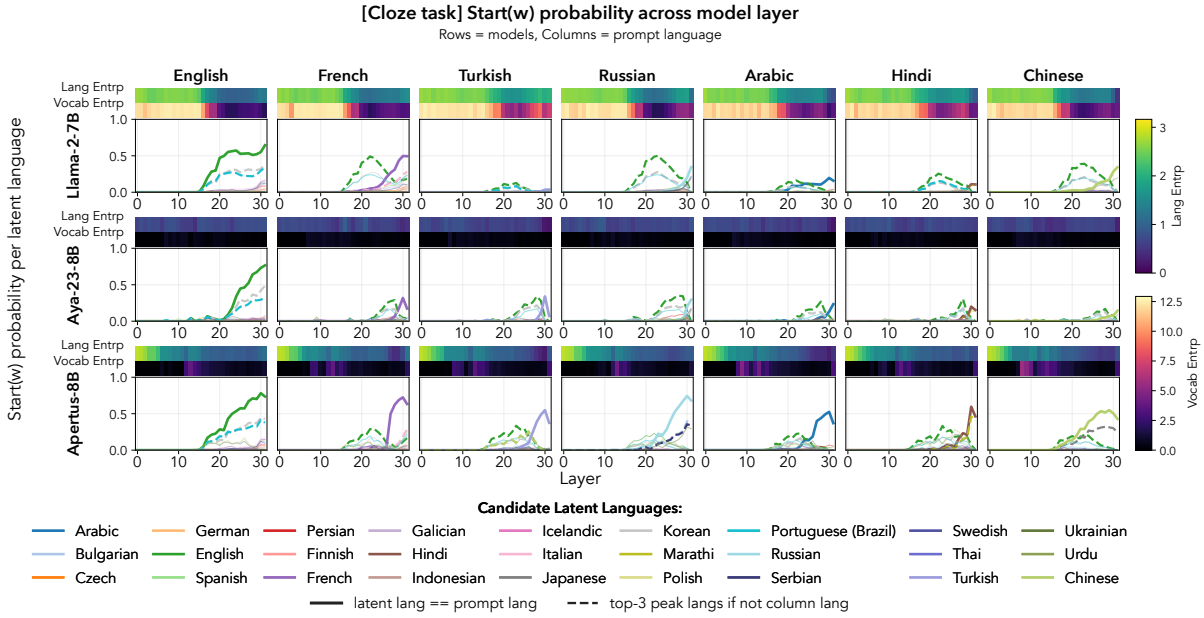


Figure 8: **Cloze prompts: latent language probability across model layers.** For each cloze prompt and model layer, we measure the summed next-token probability of all tokens that can begin the corresponding answer in each candidate language ($\textsc{Start}(w)$ probability). Rows are models and columns group prompts with the same prompt language. Upper strips show average latent language distribution entropy and average first-token vocabulary entropy.

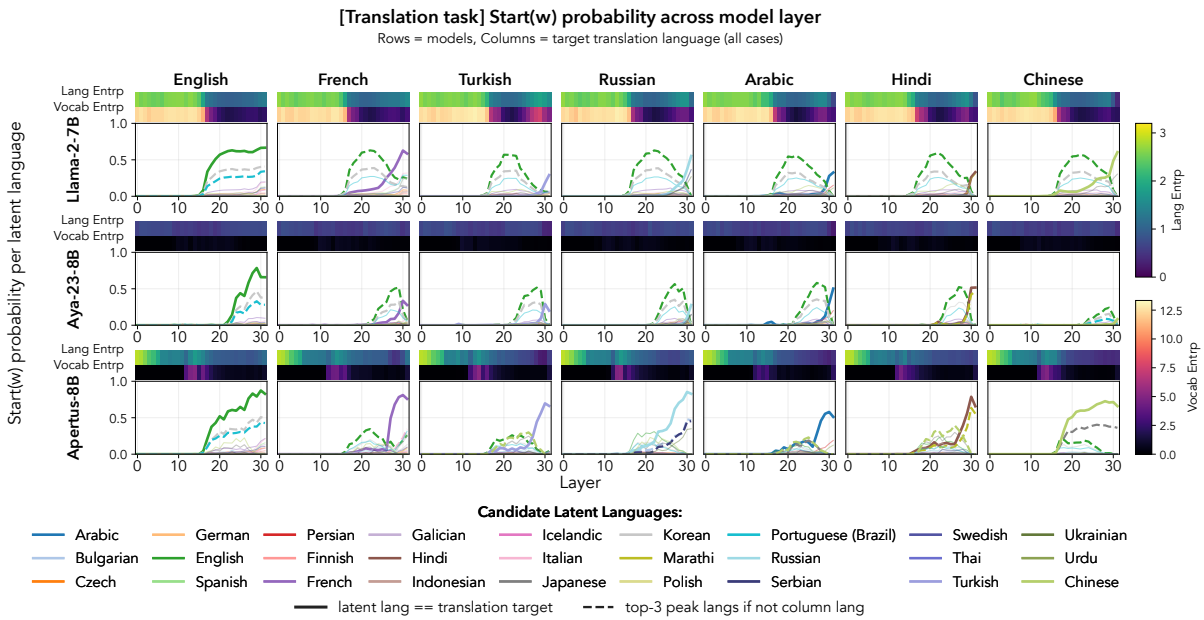


Figure 9: **Translation prompts: latent language probability across model layers, grouped by translation target.** For each translation prompt and model layer, we measure the summed next-token probability of all tokens that can begin the corresponding answer in each candidate language ($\textsc{Start}(w)$ probability). Rows are models and columns group prompts with the same translation target. English source and target cases are included. Upper strips show average latent language distribution entropy and average first-token vocabulary entropy.

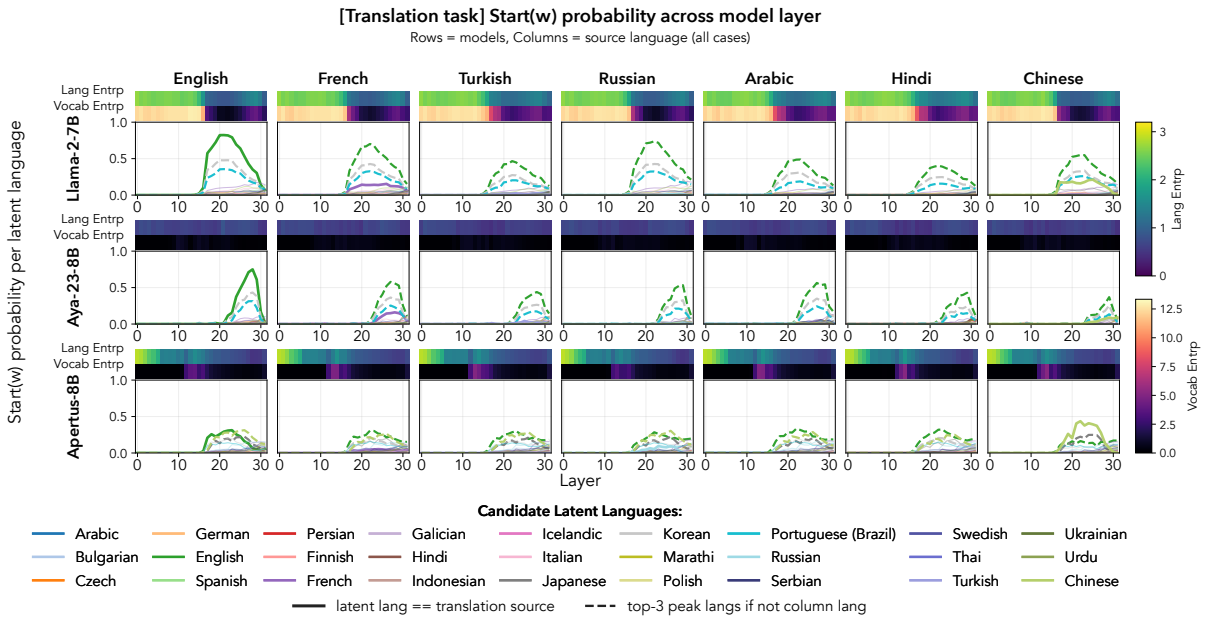


Figure 10: **Translation prompts: latent language probability across model layers, grouped by translation source.** For each translation prompt and model layer, we measure the summed next-token probability of all tokens that can begin the corresponding answer in each candidate language (START(w) probability). Rows are models and columns group prompts with the same translation source. English source and target cases are included. Upper strips show average latent language distribution entropy and average first-token vocabulary entropy.

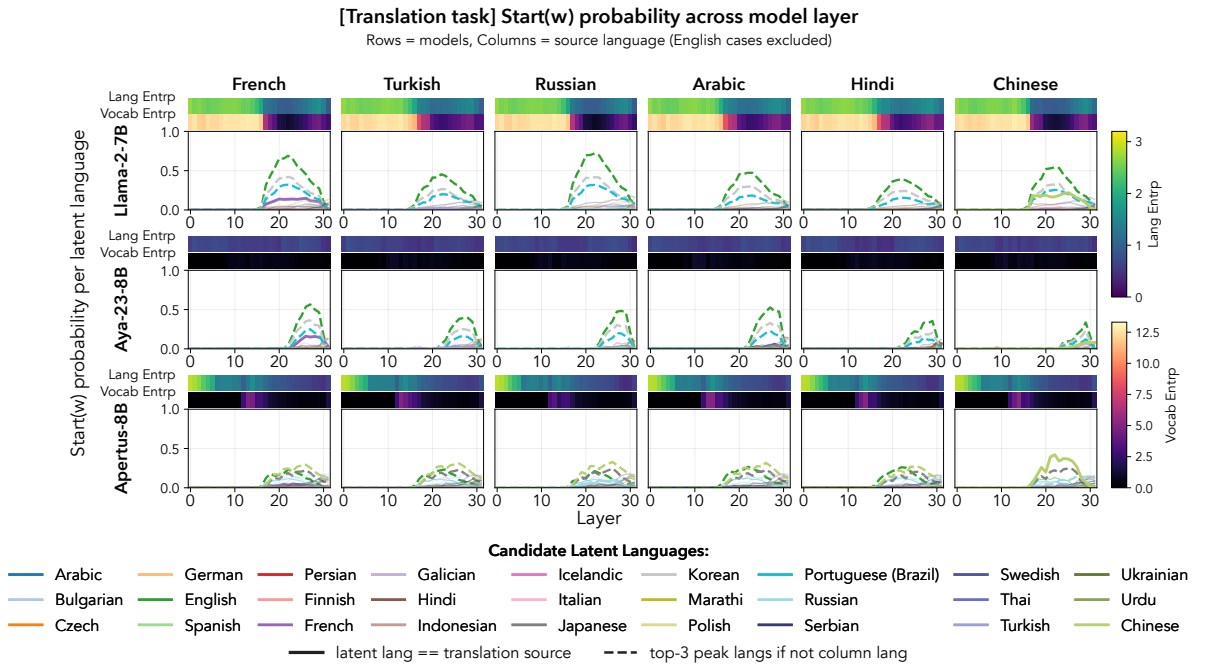


Figure 11: **Translation prompts: latent language probability across model layers, grouped by translation source.** For each translation prompt and model layer, we measure the summed next-token probability of all tokens that can begin the corresponding answer in each candidate language (START(w) probability). Rows are models and columns group prompts with the same translation source. English is excluded as a source or target, but retained as a candidate latent language. Upper strips show average latent language distribution entropy and average first-token vocabulary entropy.

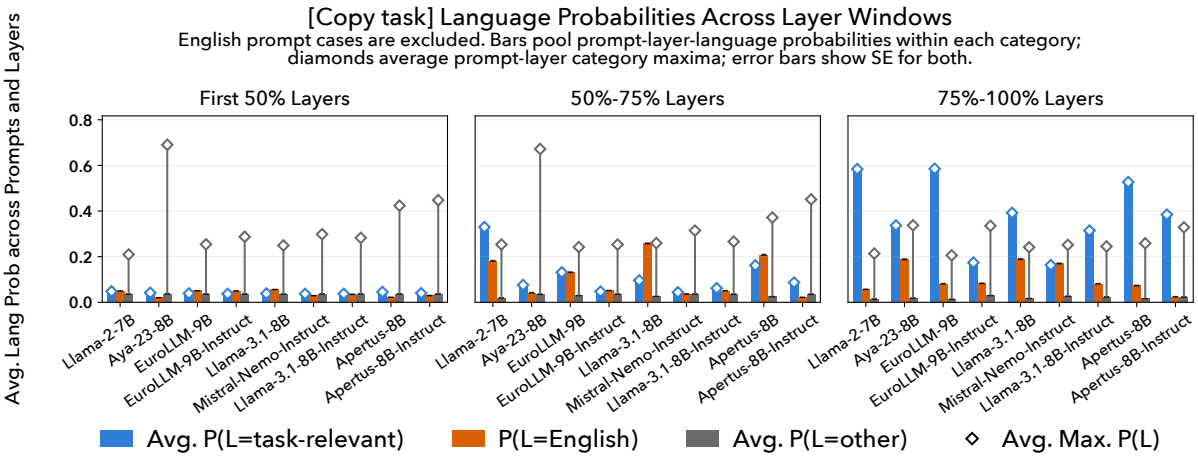


Figure 12: **Copy prompts: latent language probabilities across layer windows.** Columns show the first 50%, 50–75%, and 75–100% of layers. Bars pool individual prompt–layer–language probabilities within each category before averaging; diamonds show the average maximum probability within the task-relevant or other-language category. English prompts are excluded, but English remains a candidate latent language. Error bars show SE.

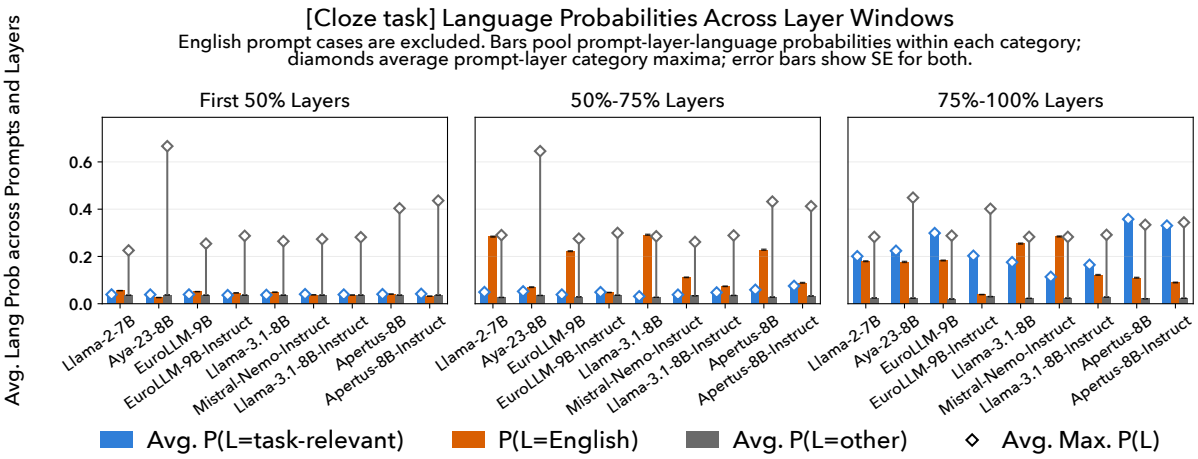


Figure 13: **Cloze prompts: latent language probabilities across layer windows.** Same layout and aggregation as Fig. 12, evaluated on cloze prompts. English prompt cases are excluded.

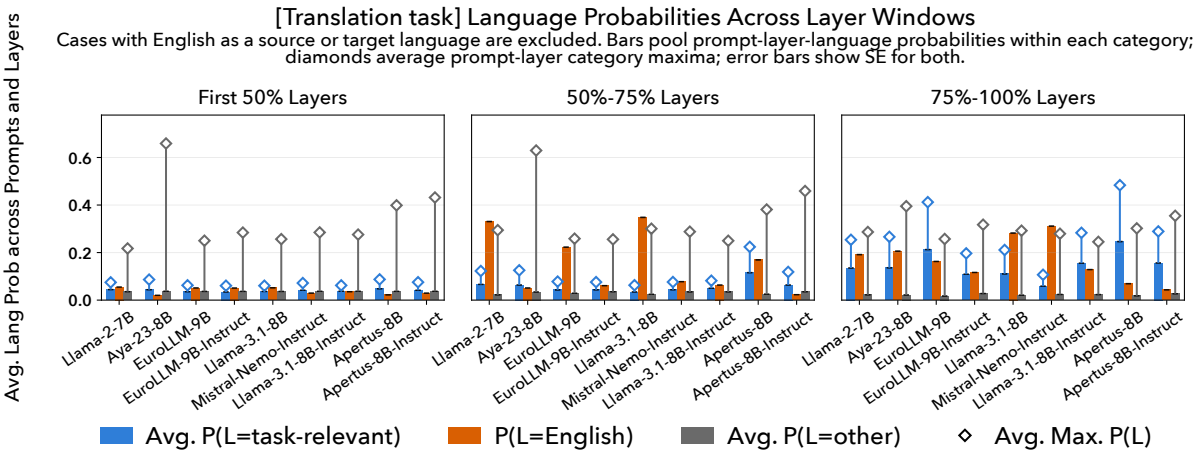


Figure 14: **Translation prompts: latent language probabilities across layer windows.** Same layout and aggregation as Fig. 12. For translation, the task-relevant category contains both the translation source and target language. English is excluded as a source or target, but retained as a candidate latent language.

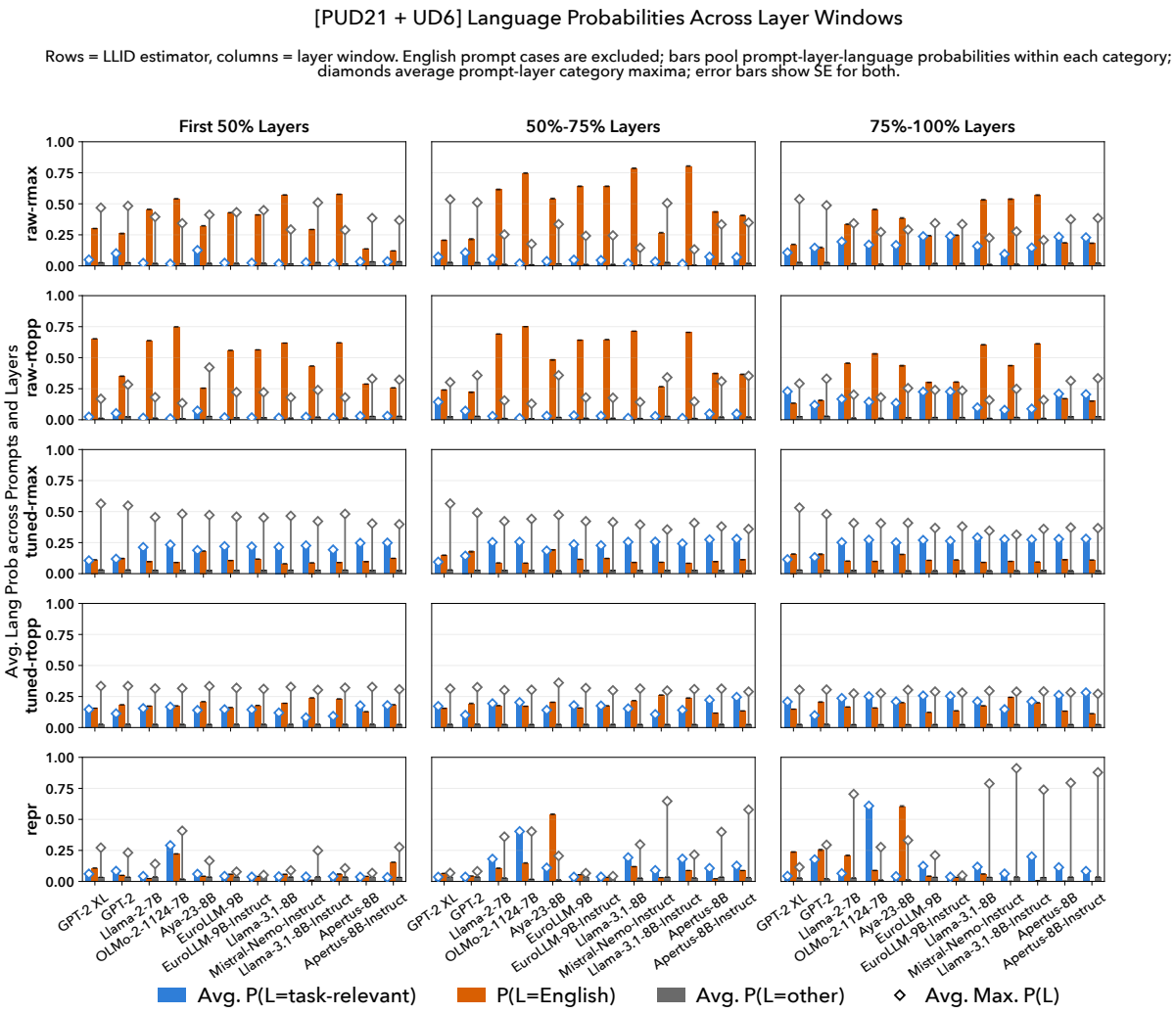
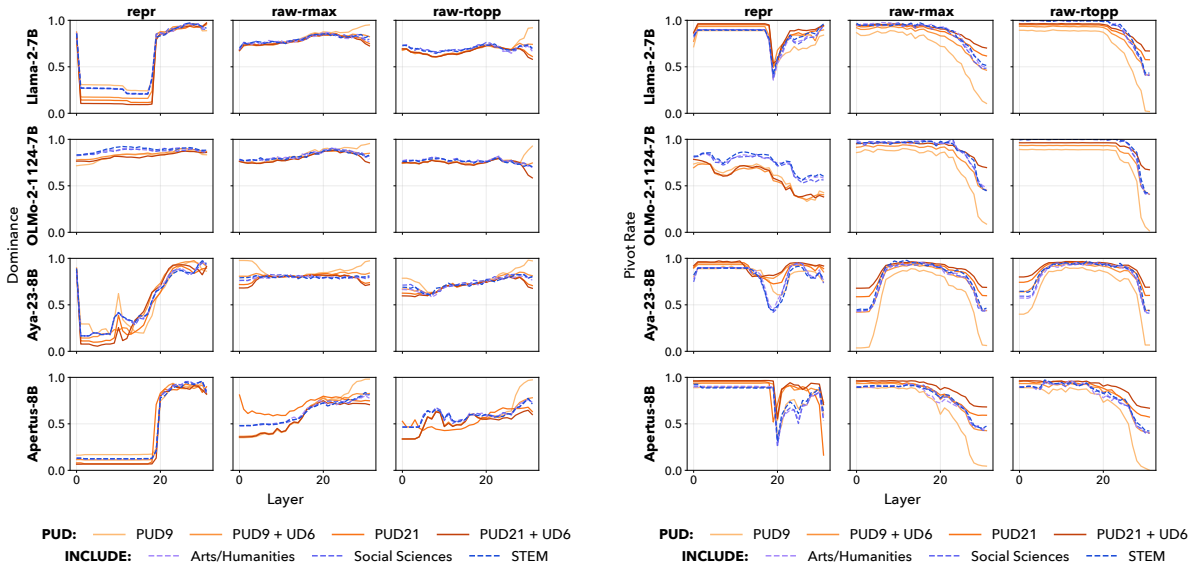


Figure 15: **Open-ended PUD21+UD6: latent language probabilities across layer windows.** Rows are LLID estimators and columns are layer windows. Bars, diamonds, and error bars use the same aggregation as Fig. 12. The task-relevant category is the prompt language. English prompts are excluded, but English remains a candidate latent language.



(a) **Dominance** (b) **Pivot rate**

Figure 16: **Latent language dominance and pivoting are comparatively stable across domains and language inventories.** Rows are language models and columns are LLID methods. For each prompt and layer, dominance is the highest probability assigned to a candidate latent language; a pivot occurs when the dominant language differs from the prompt language. Curves largely cluster across PUD variants (PUD9, PUD9+UD6, PUD21, PUD21+UD6) and INCLUDE domains (Arts/Humanities, Social Sciences, and STEM) within each model–estimator panel, while the layerwise trajectories can differ substantially across models.

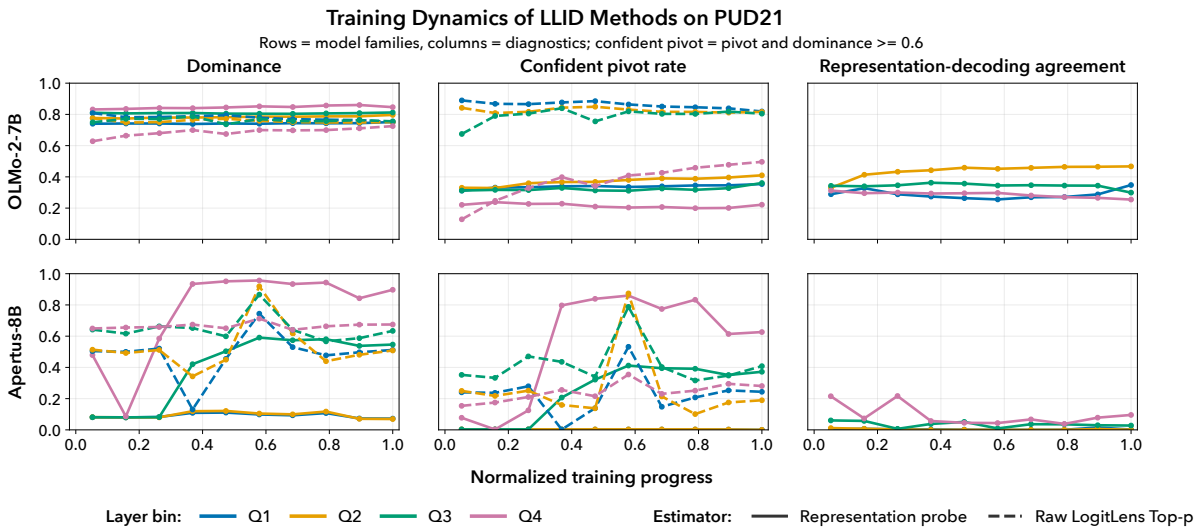


Figure 17: **Training dynamics of latent language estimates across pretraining on PUD21 with four layer bins.** This is the quartile-binned counterpart to Fig. 6, using the same rows, metrics, estimator overlays, and prompt-level definitions. Colors divide the normalized layer index j/J into quartiles: Q1 has $j/J < 0.25$, Q2 has $0.25 \leq j/J < 0.50$, Q3 has $0.50 \leq j/J < 0.75$, and Q4 has $j/J \geq 0.75$.

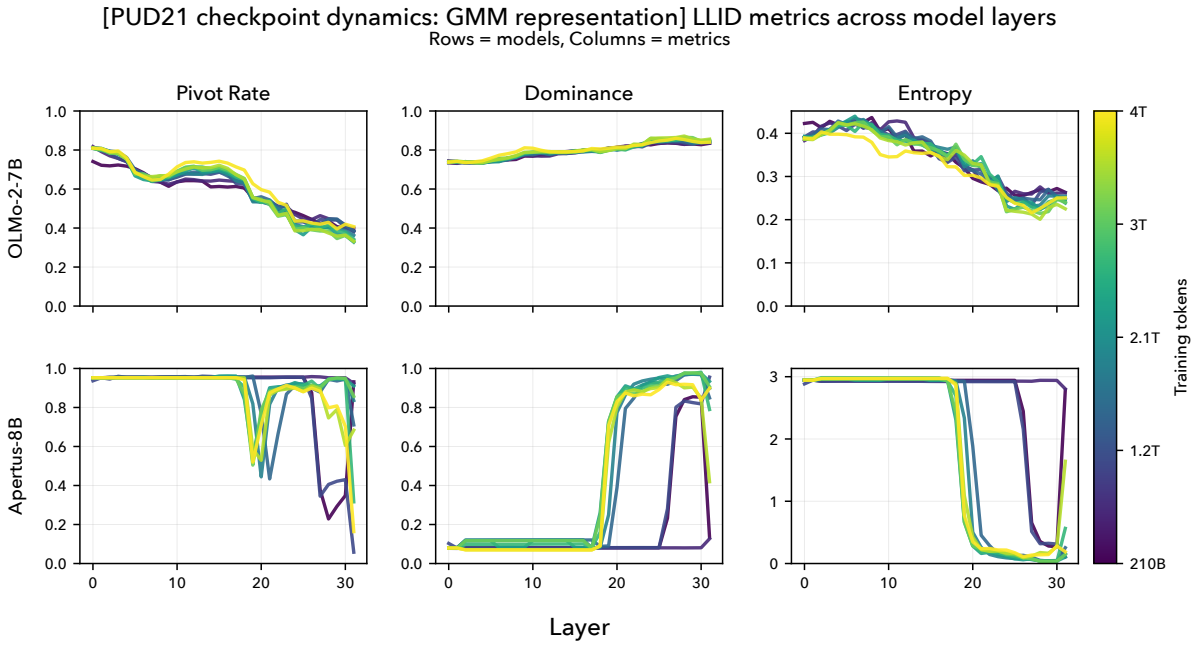


Figure 18: **Representation-based LLID checkpoint trajectories across layers on PUD21.** Rows are model families and columns are layerwise metrics. Each curve is one token-aligned checkpoint, colored by its number of training tokens. Metrics are averaged over prompts at each layer. Unlike Fig. 6, this view keeps the full layer axis instead of aggregating layers into bins.

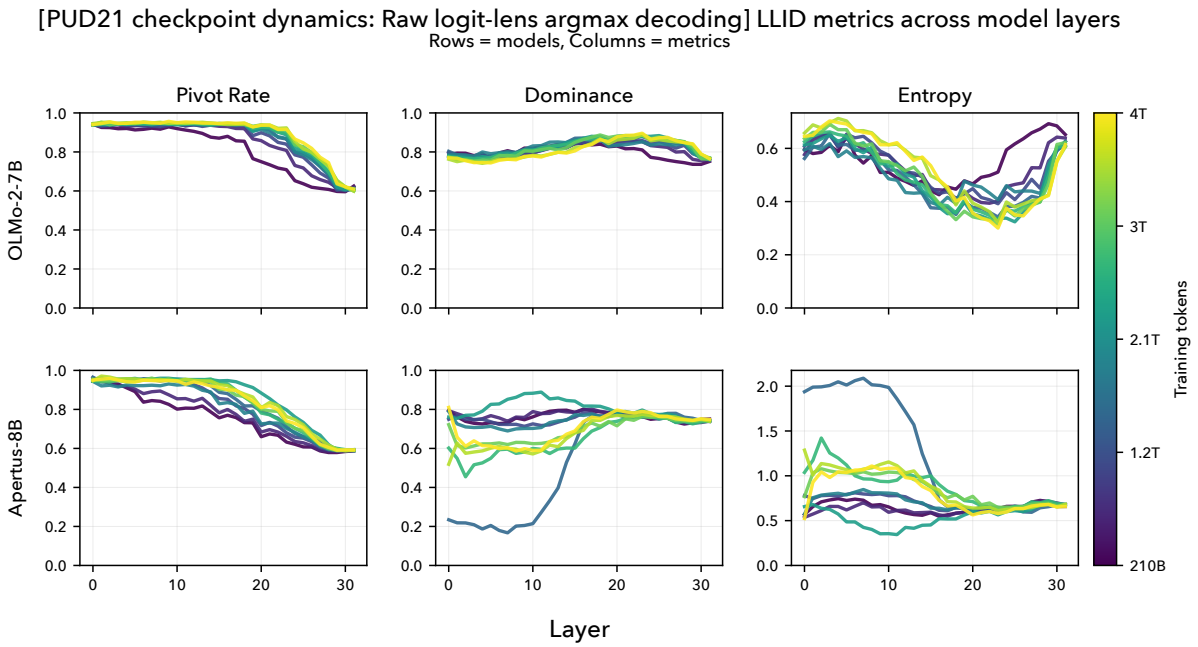


Figure 19: **Decoding-based LLID checkpoint trajectories across layers on PUD21.** Same layout as Fig. 18, using the raw logitlens argmax decoding probe instead of the representation probe.

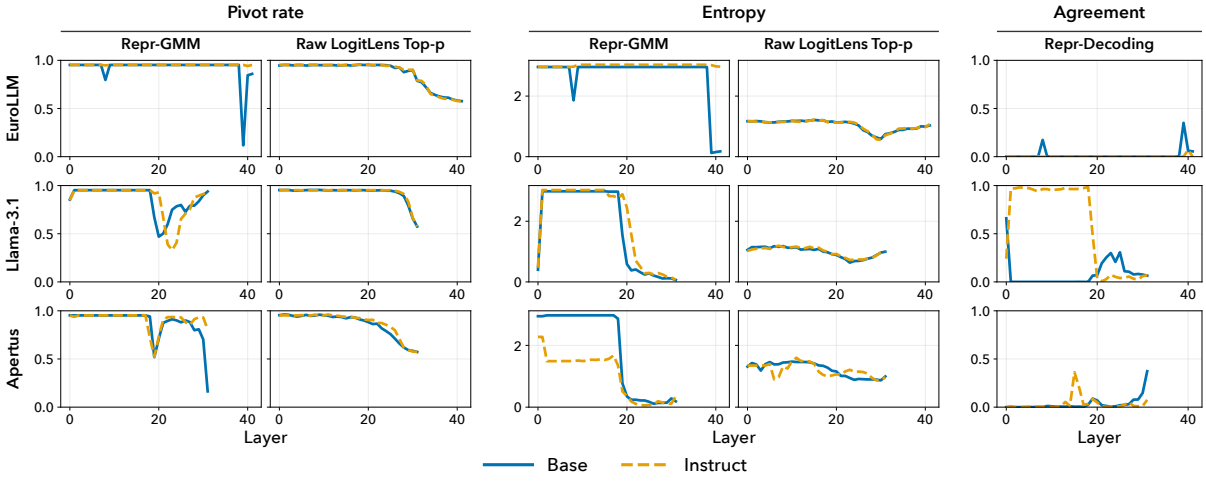


Figure 20: **Base-vs-instruct LLID trajectories on PUD21.** Rows are model families. Columns show pivot rate under `repr` and raw logitlens `dec-topp`, entropy under the same two estimators, and their dominant language agreement. Blue solid and orange dashed curves denote base and instruction-tuned variants, respectively. Agreement is the fraction of matched prompt–layer pairs for which the two probes select the same dominant language.

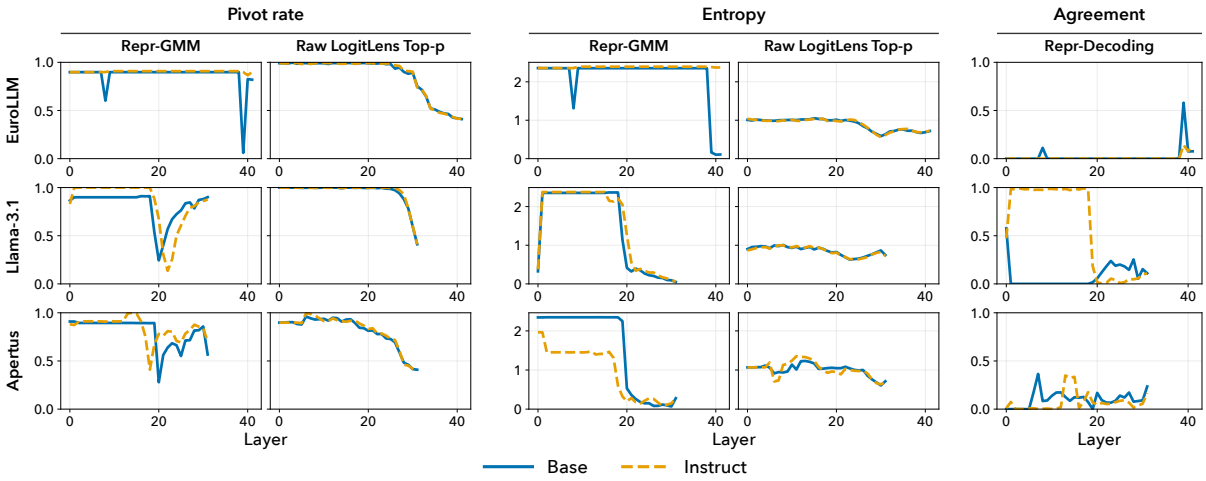


Figure 21: **Base-vs-instruct LLID trajectories on INCLUDE-10 with English as a candidate language.** Same layout as Fig. 20, evaluated on INCLUDE-10. INCLUDE contains no English prompts, but English is retained as a candidate latent language.