# CONTROLLING REPRESENTATION SHARING IN MULTILINGUAL TRANSFORMERS: A MECHANISTIC STUDY

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Understanding the mechanism through which transformers represent shared concepts across languages is a central problem in the study of multilingual large language models. Building on previous work, we find further evidence that multilingual LLMs encode the same concepts with similar representations across languages. To precisely understand shared representations in transformers, we analyze a single-layer transformer trained on a multilingual modular addition task, in which distinct vocabularies express the same underlying numerical concepts. We provide a mechanistic explanation of how these models implement arithmetic using shared Fourier representations across languages. We design a task-specific intervention on the training process of the multilingual modular addition model. This label-based intervention encourages representation sharing across languages and substantially improves cross-lingual generalization.

## 1 INTRODUCTION

Can deep learning models generalize non-locally by sharing circuits that operate on an abstract level across multiple different contexts? For example, there is empirical evidence that the hidden layers of frontier LLMs reuse a significant proportion of their internal representations across multiple different languages, even if these languages barely share any tokens (Wendler et al., 2024; Lindsey et al., 2025; Mantri et al., 2026; Dumas et al., 2025; Brinkmann et al., 2025; Veselovsky et al., 2025).[1] This suggests that LLMs route multiple (superficially very different) prompts through the same shared circuits that process abstract concepts rather than using distinct circuits for each language. Once training establishes shared circuits, all languages can leverage them, allowing for cross-lingual transfer.

However, frontier LLMs are complex with many interacting parts, so it is hard to derive clean scientific conclusions. The goal of this paper is to isolate this phenomenon of shared circuits in a mechanistically-understandable toy model. One of the simplest and most mechanistically well-understood toy settings is the training of a single-layer transformer to perform modular addition (Power et al., 2022; Nanda et al., 2023). A clean example of non-local generalization via shared circuits is "multilingual" transformers trained on disjoint cloned vocabularies. In these multilingual models, perfect token-by-token translations exist between languages, and languages don't share any tokens (Schäfer et al., 2024). Combining these two settings results in *multilingual modular addition* (Dern et al., 2026), which we study in this work. Since languages don't share any tokens in this setting, any transfer of knowledge from one language to another can be cleanly identified as non-local generalization (making this setting even more extreme than natural multilinguality, where typically at least some tokens are shared across languages).

Whenever multilingual modular addition models generalize, we observe the emergence of shared circuits that internally perform the same computations regardless of language, with language-specific translation occurring only in the first and last layers. This structure is a clean, special case of a general multilingual structure hypothesized in Heiss (2024). However, whether our models learn this structure is highly seed-dependent and inconsistent at low training fractions. Furthermore, the models that do generalize often share circuits without having similar intermediate activations, which makes representation-based mechanistic interpretation difficult.

[1] Extended related work can be found in Appendix A



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

To address these shortcomings, we design a task-specific training intervention that encourages models to share both circuits and representations across languages. This intervention substantially improves generalization across seeds and hyperparameter settings, while also allowing for easier mechanistic analysis. The general problem of changing the training data to obtain a specific internal structure in the model is termed patterning by Wang & Murfet (2026). They suggest re-weighting the training data for patterning, while our intervention changes the training labels for a limited number of epochs. After our intervention, shared circuits stay intact even after extended training with the original data.

## 2 MOTIVATION

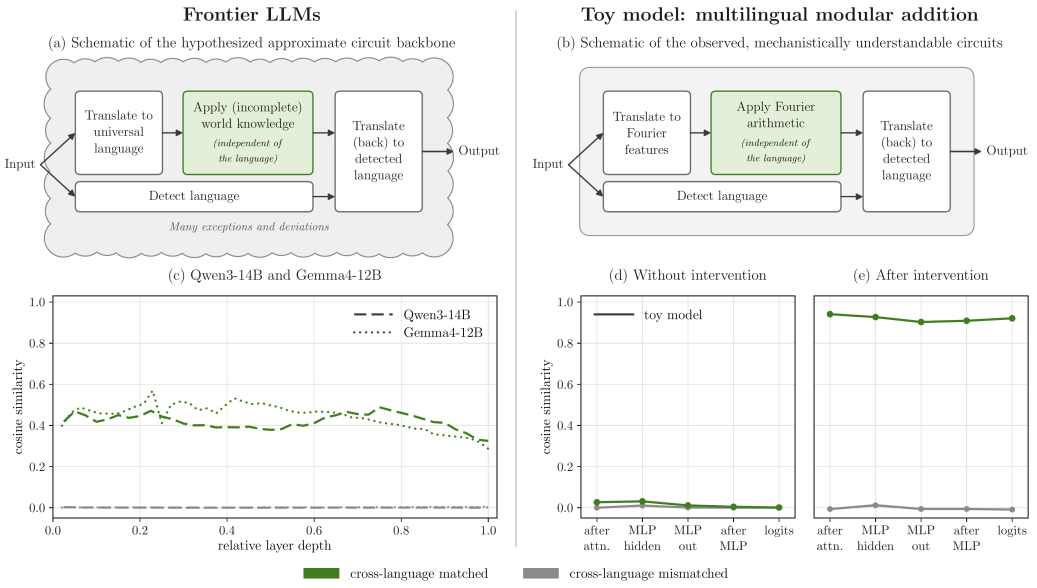


Figure 1: Cross-lingual sharing in LLMs and in a toy model. (a) A hypothesized circuit backbone in LLMs (Heiss, 2024). The hypothesized model uses the same sub-circuit across languages to reason about ideas, then translates back; (b) an observed circuit in our toy models, matching the hypothesized circuit. Panels (c) to (e) show the mean cosine similarity between activations for the same input in two languages (matched) and for randomly permuted inputs (mismatched); (c) residual stream at the final token of Qwen3-14B and Gemma4-12B on 3,000 matched MMMLU prompts in 14 languages; (d, e) The toy model (detailed in Section 3) at successive stages of the forward pass, computed using the same method as used for the LLMs; (d) without the intervention, matched similarity is always less than 0.04; (e) with the intervention (detailed in Section 5), matched similarity exceeds 0.9 at every stage.

Building on prior work (Lindsey et al., 2025; Mantri et al., 2026), we examine the similarity of activations between the same prompts translated into different languages. As a source of matched prompts in different languages that have the same meaning, we use Multilingual Massive Multitask Language Understanding (MMMLU) (OpenAI, 2024), a dataset of prompts from Massive Multitask Language Understanding (MMLU) (Hendrycks et al., 2021), translated into 14 languages by professional human translators. We pass 14 translations of the 3,000 shortest English prompts in MMMLU into Gemma4-12B and Qwen3-14B.

| **English** | **German** |
|---|---|
| What is the meaning of collective security? | Was bedeutet kollektive Sicherheit? |

Figure 2: An example of matched prompts that we compare in the cosine similarity analysis.

We first define two metrics namely matched mean cosine similarity and mismatched mean cosine similarity to measure cross-lingual sharing in LLMs. Denoting the residual stream activation at the last token in layer $t$ when passing in prompt $i$ in language $\ell$ by $\boldsymbol{h}_\ell^t(i)$, we isolate what is specific to the prompt's content by removing the language's mean direction in the residual stream, i.e., we calculate the centered activations $\boldsymbol{h}_{\ell,c}^t(i)$. Then, for each pair of languages $\ell_1, \ell_2$, we also compute the cosine similarities $C_{\ell_1,\ell_2}^{i,t}$ between the activations of the matched prompts in those two languages:

$$\boldsymbol{h}_{\ell,c}^t(i) = \boldsymbol{h}_\ell^t(i) - \frac{1}{3{,}000}\sum_i \boldsymbol{h}_\ell^t(i) \qquad C_{\ell_1,\ell_2}^{i,t} = \text{cosine similarity}(\boldsymbol{h}_{\ell_1,c}^t(i), \boldsymbol{h}_{\ell_2,c}^t(i)).$$

Finally, we take the mean over all these cosine similarities at each layer:

$$\text{Matched mean cosine similarity at layer } t = \frac{1}{3{,}000 \cdot 2 \cdot \binom{14}{2}} \sum_i \sum_{\ell_1 \neq \ell_2} C_{\ell_1,\ell_2}^{i,t}.$$

We also calculate a *mismatched* mean cosine similarity as a baseline. In particular, we randomly permute the prompts in each language separately, so that the pairs of prompts in two languages are random pairs instead of being matched translations, and then recalculate the mean cosine similarity at each layer.

We observe that matched mean cosine similarity ranges from approximately $0.25$ to $0.55$ in both Gemma4-12B and Qwen3-14B, while mismatched mean cosine similarity (the baseline) stays near 0 at every layer (Figure 1). This implies that the same concept expressed in different languages is therefore represented in substantially aligned directions compared to distinct concepts. This raises the questions that we investigate:

*How do models share representations and circuits across languages? How does this sharing affect generalization and interpretability? Under what conditions does sharing emerge during training?*

Lindsey et al. (2025) report that larger models have more similar representations across languages than smaller models do. It could be that part of the reason larger models achieve stronger performance is because they have more similar cross-lingual representations, improving their cross-lingual generalization abilities. If this is the case, could we improve model performance by guiding models to share representations more across languages?

Due to the scale of frontier models and the variability of training data, it is difficult to test these questions directly. We turn instead to a single-layer transformer which is trained to perform modular addition in different "languages." Following Dern et al. (2026), we refer to this setup as "multilingual modular addition," a setting where multilingual sharing can be induced and mechanistically verified. We describe this setting in the next section.

## 3 SETTING

We first review the modular addition task of Power et al. (2022) following which we review the multilingual modular addition task of Dern et al. (2026), which we study in-depth for the rest of the paper.

**Modular addition task.** In the modular addition task, a transformer takes in a three-token input $(a, b, =)$ where $a, b \in \{0, 1, \ldots, P-1\}$ for a prime $P$ and is trained to predict the token representing $a + b \pmod P$.[2] This is the setting in which Power et al. (2022) studies a phenomenon called *grokking*, which shows sudden generalization to high validation accuracy after many epochs of achieving perfect training accuracy, but near zero validation accuracy.

**Multilingual modular addition.** The multilingual modular addition task generalizes the modular addition task by training a single transformer to perform the modular addition task on multiple disjoint token vocabularies of size $P + 1$ (representing $\{0, 1, \ldots, P-1\}$ and "="). We call these disjoint vocabularies "languages," in analogy to multilingual large language models, in which distinct tokens often represent the same concept in different languages. For example, "one" in English

[2]The tokens $a, b$, and $=$ are each input as arbitrary one-hot vectors, *not* as the actual numbers $a, b \in \{0, 1, \ldots, P-1\}$ so the model only learns the numbers they represent through the training data.

means the same thing as "uno" in Spanish, but these concepts are represented by distinct tokens. The total vocabulary of the multilingual modular addition models is of size $L \cdot (P+1)$, where $L$ is the number of languages. We train these models *exclusively on input-output pairs where all tokens are in the same language*. Expressly, the model is trained to map

$$\big(a_{(\ell)}, b_{(\ell)}, (=)_{(\ell)}\big) \mapsto (a + b \pmod P)_{(\ell)},$$

where $n_{(\ell)}$ is a one-hot vector representing the number $n$ in language $\ell \in \{0, 1, \ldots, L-1\}$, and $(=)_{(\ell)}$ is a one-hot vector representing "equals" in language $\ell$. [3]

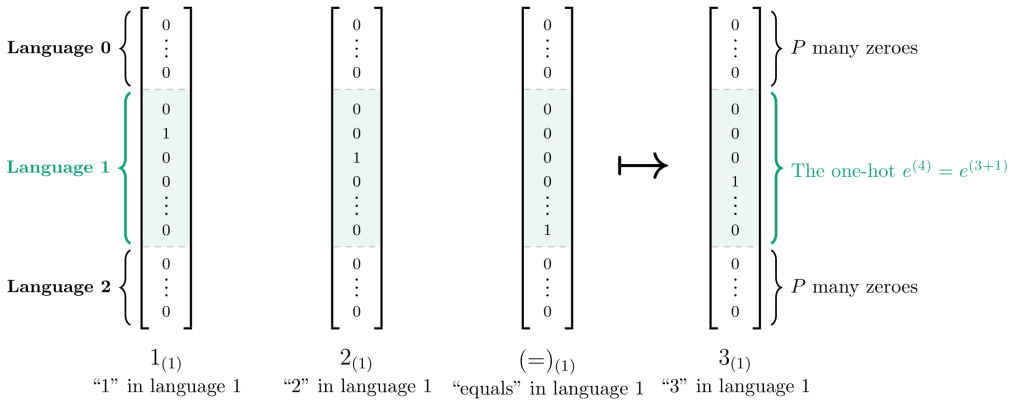


Figure 3: An example of an input-label pair from the multilingual modular addition task. This training example represents the computation $1 + 2 = 3$ in language 1.

The choice of tokenization, that is, which one-hot vector represents which token (i.e., which number and language pair) can be arbitrary; the model is agnostic to this. However, to make visualization of the embedding and unembedding matrices more clear, we pick a one-hot encoding that groups the token by language and puts them in numerical order within language (with the equals token for each language immediately after the numerical tokens in that language). We therefore define

$$n_{(\ell)} \coloneqq e^{((P+1)\ell+n+1)} \quad \text{and} \quad (=)_{(\ell)} \coloneqq e^{((P+1)\ell+P+1)},$$

where $e^{(i)} \in \mathbb{R}^{L\cdot(P+1)}$ is the $i$th standard basis vector. Equivalently, we can write $n_{(\ell)}$ as a block vector (by which we mean a block matrix that is just a column vector):

$$n_{(\ell)} = (\mathbf{0}, \ldots, \mathbf{0}, e^{(n+1)}, \mathbf{0}, \ldots, \mathbf{0}) \in \mathbb{R}^{L\cdot(P+1)},$$

where $\mathbf{0} \in \mathbb{R}^{P+1}$ is the vector of $P+1$ many zeroes, and the $(\ell+1)$th block of $n_{(\ell)}$ is the standard basis vector $e^{(n+1)} \in \mathbb{R}^{P+1}$.

Following Dern et al. (2026), we train single-layer transformers to perform multilingual modular addition with cross-entropy loss using AdamW.[4] We independently sample the training data from the $P^2$ possible input sequences for each language, with different training fractions for each language.[5] A full formalization of our setting can be found in Appendix B.

[3] The equals token in the input is superfluous in the sense that both the language and the numerical solution are identified from $a_{(\ell)}$ and $b_{(\ell)}$ alone. However, adding an additional token helps a single-layer transformer compute the modular sum by giving it a position to mix the information from the $a_{(\ell)}$ and $b_{(\ell)}$ tokens.

[4] By default, we use $P = 113$, $d_{\text{model}} = 128$, 30,000 training epochs (although we have tested up to 150,000 with similar results), learning rate 0.002, weight decay 1.0, and $\beta_1 = 0.9, \beta_2 = 0.98$. We vary these parameters and obtain similar results.

[5] We use different training fractions for each language because (1) frontier language models are usually trained with more data in some languages than others; (2) unbalanced amounts of data across languages has been shown to improve cross-lingual generalization (Schäfer et al., 2024); and (3) we can observe the model as it groks each language, which occurs at distinct periods in training in order of training fraction. Usually, we use 30% of all $P^2$ pairs for language 0, 27% for language 1, 21% for language 2, and 12% for language 3, but we also did stability tests varying the training fractions.

## 4 CROSS-LINGUAL TRANSFER

Since the model has the same modular addition task in each language, just with different tokens, we ask whether learning to do modular addition in the majority languages (the languages with more training data) improves the model's chances to learn the minority language (the language with the least training data). In other words, do multilingual modular addition models exhibit cross-lingual transfer? We examine this question by training a multilingual modular addition model, and then training the model again with the same initialization and hyperparameters, but with only the data in the minority language. Out of 25 seeds, the model with data in all languages achieves greater than 99% validation accuracy on the minority language in 23 seeds, but the model with only minority language data never achieves greater than $1.1\%$ validation accuracy. Therefore, cross-lingual transfer does occur—having already learned the other languages improves the model's chances of learning the minority language.[6]

If models used distinct circuits for each language to compute modular sums, we would expect that these circuits would not be helpful for the model's training on other languages. The fact that we do see transfer therefore suggests that multilingual modular addition models use the same circuits to compute modular sums regardless of the language of the input. However, this evidence is only suggestive. To more conclusively demonstrate the existence of cross-lingual circuits, we employ mechanistic methods (see Section 6). Even if circuits (model internals that drive the modular addition computations) are shared across languages, this doesn't necessitate that representations (such as embeddings, activations, and unembeddings) of the same numbers are similar across languages. We show below that representations are surprisingly dissimilar across languages, but that an intervention can force cross-lingual representation sharing and simultaneously improve model performance.

## 5 INTERVENTION: WARM-UP

The primary result of Nanda et al. (2023) is that single-layer transformers trained to perform modular addition use Fourier transform-based circuits to compute modular sums. We find that multilingual modular addition models also develop Fourier-based representations, but that these representations are nearly orthogonal across languages.

In order to guide the model's training to use shared representations instead of orthogonal representations for the same concepts across languages, we replace the usual "hard" labels (which are one-hot vectors representing the correct numerical answer in the correct language) with a mean of all the one-hots representing the correct numerical answer across every language. In other words, we replace each one-hot label $y_{(\ell_{\text{correct}})}$ with $y_{(\text{agg})} \coloneqq \frac{1}{L}\sum_{\ell=0}^{L-1} y_{(\ell)}$. These "aggregated" labels have $1/L$ in every entry corresponding to the correct numerical modular sum, regardless of language, and zeroes in every other entry. For example, if $L = 2$ and $P = 2$, then we replace the hard label $1_{(0)} = (0, 1, 0, 0)$ with the aggregated label $1_{(\text{agg})} = (0, 1/2, 0, 1/2)$. We can think of the aggregated labels as a probability vector that represents a uniform distribution over the $L$ correct numerical solutions. After training with the aggregated labels for 10,000 epochs (by default), we switch to training with the hard labels.[7] We call this two-phase regime *warmed-up* training.

When training with cross-entropy loss on the aggregated labels, the loss for input $\left(a_{(\ell)}, b_{(\ell)}, (=)_{(\ell)}\right)$ and output distribution $\hat{y}$ is

$$\mathcal{L}\left((a + b \pmod P)_{(\text{agg})}, \hat{y}\right) = -\frac{1}{L}\sum_{\ell=0}^{L-1} \log\left(\hat{y}_{a+b \,(\text{mod } P),\, \ell}\right),$$

where $\hat{y}_{n,\ell}$ denotes the predicted probability of number $n$ in language $\ell$.

When training with the aggregated labels, there is no preferred language for the model to output, so the model has no information about which language is "correct." In particular, the loss above is minimized when the model predicts a uniform distribution over the $L$ tokens representing the correct

[6] At lower training fractions, a model with data in all languages generalizes less often (9 seeds out of 20 with 4 heads, 6 seeds out of 10 with 8 heads), but considerably more than the model with only the minority language, which still never generalizes.

[7] Warmed-up models are always compared to standard-models trained for the same amount of total epochs.

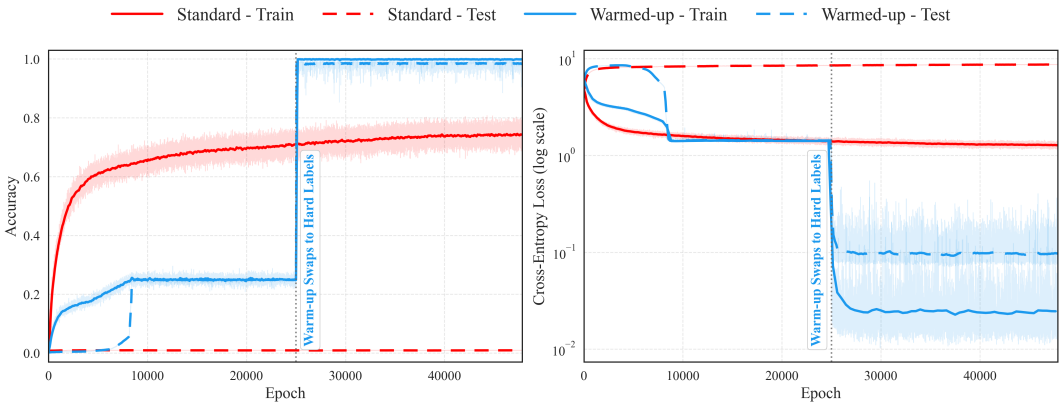


Figure 4: Exact-match accuracy (left) and loss (right) curves of warmed-up (blue) and standard (red) multilingual modular addition training runs. Warmed-up training suddenly generalizes after switching from aggregated labels to hard labels. These runs are trained with $L = 4$ languages, so the warmed-up model can't perform much better than $1/4$ accuracy until we switch to training with hard labels. Transparent curves plot raw data and solid curves plot smoothed data using a moving average. Cross-entropy loss is computed with respect to the hard labels, not the aggregated labels.

number in different languages. The model's exact-match validation accuracy (proportion of outputs matching the correct number *and* language) therefore does not much exceed $1/L$, but its numerical accuracy (proportions of outputs equal to the correct number in *any* language) reaches $\approx 1$ across a wide variety of hyperparameter settings. Since the training labels are invariant to the input language during warm-up, the model is incentivized to use the same representation for the same number in different languages.

### 5.1 Results of Warm-up

Both the validation and training accuracy of warmed-up training have the same consistent trajectory across hyperparameters and seeds: the model quickly reaches an exact-match accuracy of $1/L$, and a numerical accuracy of $\approx 1$, then within 100 epochs of the swap to hard labels, the exact-match accuracy jumps to $\approx 1$ as well. This pattern seems to suggest that warmed-up models learn the modular addition task first, then quickly learn the full multilingual task by appending language detection circuits to their existing language-agnostic modular addition circuits. In contrast, standard training (using only the hard labels) yields unstable behavior at low training fractions, with generalization behavior highly dependent on random seed and hyperparameters.

As a result, warm-up almost always generalizes as well as or better than standard training. Out of 89 combinations of random seed and hyperparameter settings, standard training reached higher validation accuracy than warmed-up training in only 4 instances (excluding cases of $< 0.1\%$ accuracy differences). Warmed-up training grokked (i.e., reached $> 99\%$ validation accuracy) in every setting that standard training grokked. Out of 30 seeds and hyperparameter settings with a low training fraction, standard training grokked in only 15 of them, but warmed-up training with the same hyperparameters groks in all 30.

If a model learns shared representations for each language, we might expect to see this similarity across languages in the embedding and unembedding matrices of the model. We do in fact see this similarity in Figure 5. Note that the embedding matrix $\boldsymbol{E} \in \mathbb{R}^{L(P+1)\times d_{\text{model}}}$ and unembedding matrix $\boldsymbol{U} \in \mathbb{R}^{LP\times d_{\text{model}}}$ can be written as block matrices, splitting the matrices into $L$ "language blocks" along the vocabulary/numerical dimension:

$$\boldsymbol{E} = \begin{bmatrix} \boldsymbol{E}^{(0)} \\ \vdots \\ \boldsymbol{E}^{(L-1)} \end{bmatrix} \qquad \boldsymbol{U} = \begin{bmatrix} \boldsymbol{U}^{(0)} \\ \vdots \\ \boldsymbol{U}^{(L-1)} \end{bmatrix},$$

where $\boldsymbol{E}^{(\ell)} \in \mathbb{R}^{(P+1)\times d_{\text{model}}}$ and $\boldsymbol{U}^{(\ell)} \in \mathbb{R}^{P\times d_{\text{model}}}$.

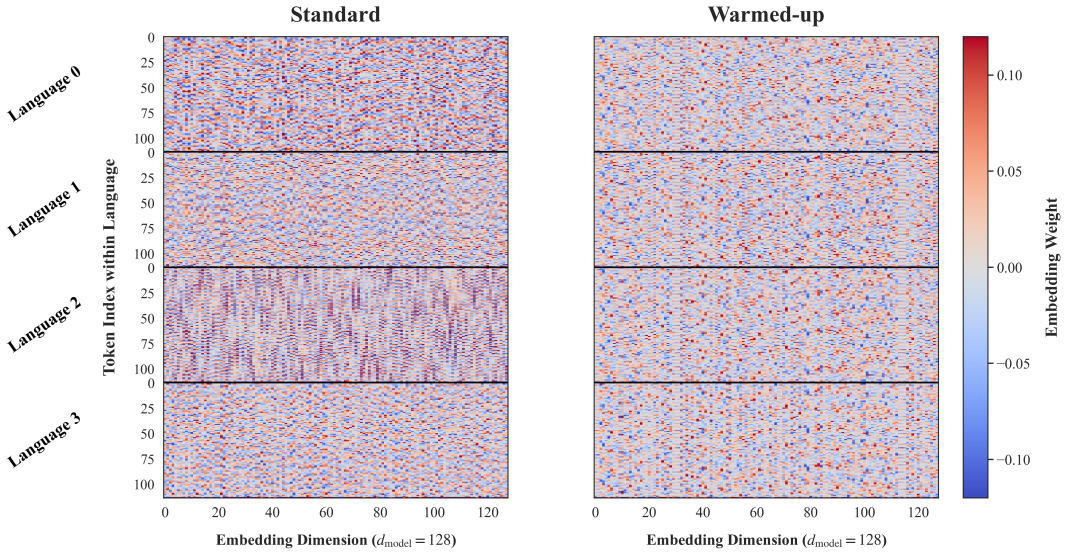


Figure 5: Embedding matrices of two multilingual modular addition models: with standard training (left) vs with warm-up (right). Models trained with warm-up have similar patterns across language blocks, while standard-trained models have different patterns between language blocks. (This is a plot of two embedding matrices, not eight. The black lines between blocks are added only as a visual aid.)

### 5.1.1 From the Fourier Perspective

In the visualization of the embedding matrix (Figure 5), we see a periodic-like pattern in the embeddings. This is not surprising, as Nanda et al. (2023) has shown that small transformers trained on the modular addition task use sinusoidal representations that are sparse in the frequency domain. Nanda et al. (2023) identifies the few *key frequencies* that are used for the modular addition computation by looking at which frequencies have non-negligible Fourier power in the spectra of the embedding and unembedding matrices. This gives us a useful lens to analyze the embedding and unembedding matrices of our multilingual modular addition models. If a model has the same key frequencies when separately applying a discrete Fourier transform of each language block of the embedding matrix (or unembedding matrix), we take this as evidence that the model has shared representations of numerical concepts across languages.

To get the Fourier power $\mathcal{P}_{\boldsymbol{E}}^{(\ell)}(\omega)$ of the embeddings in each language $\ell$ and frequency $\omega$, we perform a discrete Fourier transform on the embeddings within each language block. Concretely, define

$$\mathcal{P}_{\boldsymbol{E}}^{(\ell)}(\omega) = \left\| \sum_{a=0}^{P-1} \boldsymbol{E}_{a,:}^{(\ell)} e^{-i\frac{2\pi\omega a}{P}} \right\|^2 = \left\| \sum_{a=0}^{P-1} \boldsymbol{E}_{a,:}^{(\ell)} \sin\left(\frac{2\pi\omega a}{P}\right) \right\|^2 + \left\| \sum_{a=0}^{P-1} \boldsymbol{E}_{a,:}^{(\ell)} \cos\left(\frac{2\pi\omega a}{P}\right) \right\|^2,$$

where $\boldsymbol{E}_{a,:}^{(\ell)} \in \mathbb{R}^{d_{\text{model}}}$ is the embedding of $a_{(\ell)}$, i.e., of the token $a \in \{0, \ldots, P-1\}$ in language $\ell$. We define $\mathcal{P}_{\boldsymbol{U}}^{(\ell)}$ just as above, but with $\boldsymbol{U}$ in place of $\boldsymbol{E}$.

We see both in the matrix visualization (Figure 5) and in the Fourier spectra (Figure 6) that the embeddings are much more similar across languages when training with warm-up compared to with standard training.[8] When training with warm-up, similar patterns appear in the embedding matrix across languages, while distinct patterns that naturally separate by language block appear in standard training. Key frequencies are shared across languages when training with warm-up, while key frequencies are arbitrary and largely unshared when training with standard training. The cross-lingual similarities in the representations induced by warm-up remain intact even after extended training with the hard labels.

[8]The visualizations of the unembedding matrices look very similar to those of the embedding matrices.

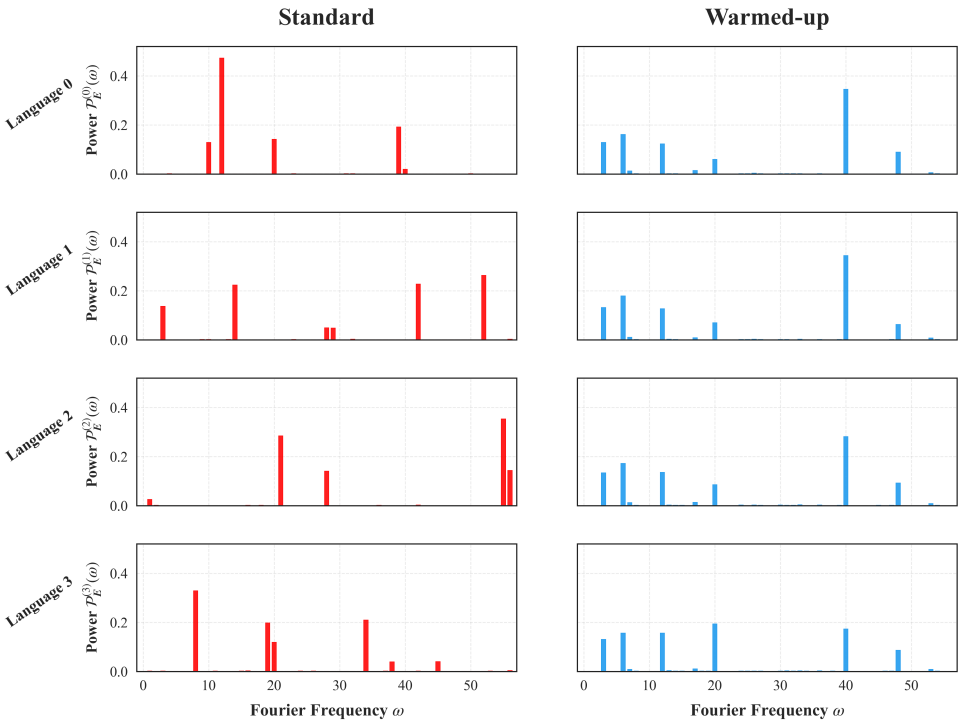


Figure 6: Fourier power spectra of each language block of the embedding matrix. The spectrum is sparse, and key frequencies are shared when training with warm-up, but are largely unshared with standard training.

## 6 Circuit Sharing

Section 5 showed that standard training produces distinct representations for the same concept in different languages. Section 4 suggests what seems to be the opposite conclusion: that standard-trained models use the same circuits for different languages. We use mechanistic methods to reconcile this seeming contradiction, showing that circuits are shared even though representations are not.

**Circuit sharing without representation sharing.** It is possible for languages in a model to do the same computations while having dissimilar representations. Suppose, for example, that one language takes every input number and relabels it by multiplying it by some fixed value $m$ in the embedding layer. This would preserve the internal computation of the model since multiplying both numbers by the same factor before adding them gives the same answer as adding first and multiplying after ($ma + mb \pmod P = m(a + b) \pmod P$). A model exploiting this fact could reuse another language's internal circuit since multiplying by $m \bmod P$ just permutes which tokens correspond to which numbers. Yet the two languages' intermediate activations would not resemble each other, since the relabeling would scramble the patterns that representation-level measures look for. We construct such a model by taking a warmed-up model, whose languages already share key frequencies and mechanically apply this relabeling to one language $\ell$ by permuting language $\ell$'s embedding and unembedding rows with a multiplier $m \notin \{1, -1\}$, leaving every other weight untouched. This model maintains the exact same accuracy and loss as the original warmed-up model over the full $P \times P$ grid.

To directly test whether standard-trained models share circuits across languages, we run the following ablation experiment (detailed in Appendix C). Let $\boldsymbol{h}_\ell(a, b) \in \mathbb{R}^{d_{\text{model}}}$ denote the residual stream immediately after the attention layer at the last token position (that is, at $(=)_{(\ell)}$) on the input $(a_{(\ell)}, b_{(\ell)}, (=)_{(\ell)})$. For a given language $\ell$ and frequency $\omega$, we identify the corresponding 2-dimensional subspace $S_{\ell,\omega} \subseteq \mathbb{R}^{d_{\text{model}}}$ responsible for the computations in frequency $\omega$ in language $\ell$. We then ablate this subspace in the residual stream $\boldsymbol{h}_{\ell'}(a, b)$, by replacing $\boldsymbol{h}_{\ell'}(a, b)$ with

$\boldsymbol{h}_{\ell'}(a,b) - \text{proj}_{S_{\ell,\omega}}(\boldsymbol{h}_{\ell'}(a,b))$. If two languages share circuits, ablating $S_{\ell,\omega}$ should damage the accuracy among inputs in language $\ell'$ about as much as it damages $\ell$ itself.

Ablating $S_{\ell,\omega}$ at a key frequency $\omega$ of language $\ell$ costs 20 to 55 percentage points of accuracy on inputs in language $\ell$, as expected. Inputs in the other languages lose about the same amount, even though $S_{\ell,\omega}$ was identified from language $\ell$ alone.[9] When $\omega$ is not a key frequency of $\ell$, ablating $S_{\ell,\omega}$ leaves accuracy intact in every language, including those for which $\omega$ is a key frequency. Further, ablating a random plane only reduces accuracy by $2.3 \times 10^{-5}$ to $7.6 \times 10^{-4}$. Standard-trained models therefore share circuits across languages even if they do not share representations.

Finally, we confirm that this experiment is capable of detecting sharing when it does actually exist. We take our permuted warmed-up model from above, where circuit sharing between two languages is true by construction but undetectable by representation-level measures — precisely the situation our ablation test is designed to catch. Running the same test on this model, ablating $S_{\ell,\omega}$ damages $\ell'$'s accuracy just as much as it damages $\ell$'s accuracy, confirming the method detects sharing when it is actually present.

## 7 Conclusion

In this work, we analyzed the representations and circuits of multilingual transformers through a modular addition toy model and large open-weight models. We show in our toy model that there is significant transfer between languages that don't share tokens. Motivated by an analysis of shared cross-lingual representations of concepts in LLMs, we investigate the representations used by the toy model, finding that our standard-trained models use different representations for the same concept across languages, even though they benefit from transfer during training and use the same circuits for computation in their hidden layers. To make this model more interpretable, we introduce *warm-up*, an intervention on the training labels that encourages the model to first learn the conceptual meaning of tokens before learning which language they belong to. We show this intervention makes the model share representations and also improves generalization.

## 8 Limitations and Future Work

**What relates circuits across languages in standard-trained models?** Section 6 shows that standard-trained models that grok do share an internal circuit. What we do not know is the precise correspondence between languages that makes this sharing possible. Characterizing this correspondence, and testing whether it generalizes across seeds and hyperparameters, is an open problem.

**Representation-level measures can miss sharing.** The contrast in Section 6 between near-zero representation similarity and near-complete circuit sharing is a general limitation of representation-level diagnostics. Cosine similarity, for example, detects content that is shared *and* aligned in the same directions across languages. A practical consequence is that a null result from the cosine similarity analysis on Qwen or Gemma in Section 2 would not have been sufficient evidence against cross-lingual sharing in those models.

**Supervision assumed by warm-up.** Aggregated labels require prior knowledge of which tokens refer to the same concept in different languages. This is trivial in our toy setting but difficult in actual LLMs. In the toy task, the correspondence between two languages is determined by the data only up to the multiplicative relabelings discussed in Section 6. Whether a correspondence *estimated* during training could replace the given one, without access to ground-truth token alignments, is an open and practically important question.

### AI use statement

We used generative AI tools (primarily various models of Claude and Gemini, especially agentic tools) to generate and edit code, including some aid in creating figures. We used AI for comments and feedback on the phrasing of a few sentences in the paper, but did not use it to generate any

[9] Note that the model still answers in the correct language for $\approx 96.6\%$ of inputs.

sentences appearing in the paper. We also used generative AI to search for related literature and text datasets (but not to generate text datasets) as well as to transcribe internal group meetings. We reviewed all AI-assisted code used to generate data, results, or figures in the main text of the paper. We take responsibility for the final content of this work, including AI artifacts.

### Reproducibility statement

An anonymized link to our code repository, containing implementations of all models, training procedures, and analysis methods described in this paper, is available at https://anonymous.4open.science/r/submission-code-888. Experimental parameters are detailed in footnote 4.

## References


Dzmitry Bahdanau, Shikhar Murty, Michael Noukhovitch, Thien Huu Nguyen, Harm de Vries, and Aaron Courville. Systematic generalization: What is required and can it be learned? In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=HkezXnA9YX.

Lucas Bandarkar, Benjamin Muller, Pritish Yuvraj, Rui Hou, Nayan Singhal, Hongjiang Lv, and Bing Liu. Layer swapping for zero-shot cross-lingual transfer in large language models, 2025. URL https://arxiv.org/abs/2410.01335.

Jannik Brinkmann, Chris Wendler, Christian Bartelt, and Aaron Mueller. Large language models share representations of latent grammatical concepts across typologically diverse languages, 2025. URL https://arxiv.org/abs/2501.06346.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997. URL https://doi.org/10.1023/A:1007379606734.

Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. *Advances in neural information processing systems*, 31, 2018. URL https://arxiv.org/abs/1805.09545.

Bilal Chughtai, Lawrence Chan, and Neel Nanda. A toy model of universality: Reverse engineering how networks learn group operations. In *International Conference on Machine Learning*, pp. 6243–6267. PMLR, 2023.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL https://aclanthology.org/2020.acl-main.747/.

Niclas Dern, Selma Mazioud, Jakob Heiss, Avrajit Ghosh, Curtis James McDonald, Gabriel Clara, and Bin Yu. Tracking training phases in compositional learning with task-agnostic measures. In *ICML 2026 Workshop on Compositional Learning (CompLearn)*, 2026. URL https://openreview.net/forum?id=tNWw6sXmwP.

Philipp Dufter and Hinrich Schütze. Identifying elements essential for BERT's multilinguality. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu (eds.), *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4423–4437, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.358. URL https://aclanthology.org/2020.emnlp-main.358/.

Clément Dumas, Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. Separating tongue from thought: Activation patching reveals language-agnostic concept representations in transformers. In Wanxiang Che, Joyce Nabende, Ekaterina Shutova, and Mohammad Taher Pilehvar (eds.), *Proceedings of the 63rd Annual Meeting of the Association for Computational*

*Linguistics (Volume 1: Long Papers)*, pp. 31822–31841, Vienna, Austria, July 2025. Association for Computational Linguistics. ISBN 979-8-89176-251-0. doi: 10.18653/v1/2025.acl-long.1536. URL https://aclanthology.org/2025.acl-long.1536/.

Nouha Dziri, Ximing Lu, Melanie Sclar, Xiang Lorraine Li, Liwei Jiang, Bill Yuchen Lin, Sean Welleck, Peter West, Chandra Bhagavatula, Ronan Le Bras, et al. Faith and fate: Limits of transformers on compositionality. *Advances in neural information processing systems*, 36:70293–70332, 2023.

Eric Elmoznino, Thomas Jiralerspong, Yoshua Bengio, and Guillaume Lajoie. A complexity-based theory of compositionality. *arXiv preprint arXiv:2410.14817*, 2024.

Jerry A Fodor and Zenon W Pylyshyn. Connectionism and cognitive architecture: A critical analysis. *Cognition*, 28(1-2):3–71, 1988.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018. URL https://arxiv.org/abs/1803.03635.

Jakob Heiss. *Inductive Bias of Neural Networks and Selected Applications*. Doctoral thesis, ETH Zurich, Zurich, 2024. URL https://www.research-collection.ethz.ch/handle/20.500.11850/699241.

Jakob Heiss, Alexis Stockinger, and Josef Teichmann. Reducing the number of neurons of deep ReLU networks based on the current theory of regularization, 2020. URL https://openreview.net/pdf?id=9GUTgHZgKCH.

Jakob Heiss, Josef Teichmann, and Hanna Wutte. How infinitely wide neural networks can benefit from multi-task learning–an exact macroscopic characterization. *arXiv preprint arXiv:2112.15577*, 2021. URL https://arxiv.org/abs/2112.15577.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations (ICLR)*, 2021. URL https://arxiv.org/abs/2009.03300.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. Compositionality decomposed: How do neural networks generalise? *Journal of Artificial Intelligence Research*, 67:757–795, 2020.

Karthikeyan K, Zihan Wang, Stephen Mayhew, and Dan Roth. Cross-lingual ability of multilingual bert: An empirical study. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=HJeT3yrtDr.

Daniel Kunin, Giovanni Luca Marchetti, Feng Chen, Dhruva Karkada, James B Simon, Michael R DeWeese, Surya Ganguli, and Nina Miolane. Alternating gradient flows: A theory of feature learning in two-layer neural networks. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2026. URL https://openreview.net/forum?id=t7LKc0MMW6.

Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International conference on machine learning*, pp. 2873–2882. PMLR, 2018.

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40:e253, 2017. doi: 10.1017/S0140525X16001837.

Jack Lindsey, Wes Gurnee, Emmanuel Ameisen, Brian Chen, Adam Pearce, Nicholas L. Turner, Craig Citro, David Abrahams, Shan Carter, Basil Hosmer, Jonathan Marcus, Michael Sklar, Adly Templeton, Trenton Bricken, Callum McDougall, Hoagy Cunningham, Thomas Henighan, Adam Jermyn, Andy Jones, Andrew Persic, Zhenyi Qi, T. Ben Thompson, Sam Zimmerman, Kelley Rivoire, Thomas Conerly, Chris Olah, and Joshua Batson. On the biology of a large language model, 2025. URL https://transformer-circuits.pub/2025/attribution-graphs/biology.html#dives-multilingual.

Ziming Liu, Eric J Michaud, and Max Tegmark. Omnigrok: Grokking beyond algorithmic data. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=zDiHoIWa0q1.

Raghav Mantri, Saun Chen, Yixuan Wang, and Duygu Ataman. Investigating the structural properties of linguistic biases in multilingual language models. *Information*, 17(5):498, 2026. doi: 10.3390/info17050498. URL https://doi.org/10.3390/info17050498.

Giovanni Luca Marchetti, Daniel Kunin, Adele Myers, Francisco Acosta, and Nina Miolane. Sequential group composition: A window into the mechanics of deep learning. In *International Conference on Machine Learning (ICML)*, 2026.

Sarthak Mittal, Sharath Chandra Raparthy, Irina Rish, Yoshua Bengio, and Guillaume Lajoie. Compositional attention: Disentangling search and retrieval. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=IwJPj2MBcIa.

Depen Morwani, Benjamin L. Edelman, Costin-Andrei Oncescu, Rosie Zhao, and Sham M. Kakade. Feature emergence via margin maximization: case studies in algebraic tasks. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=i9wDX850jR.

Niklas Muennighoff, Thomas Wang, Lintang Sutawika, Adam Roberts, Stella Biderman, Teven Le Scao, M Saiful Bari, Sheng Shen, Zheng Xin Yong, Hailey Schoelkopf, Xiangru Tang, Dragomir Radev, Alham Fikri Aji, Khalid Almubarak, Samuel Albanie, Zaid Alyafeai, Albert Webson, Edward Raff, and Colin Raffel. Crosslingual generalization through multitask finetuning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15991–16111, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.891. URL https://aclanthology.org/2023.acl-long.891/.

Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. In *International Conference on Learning Representations (ICLR)*, 2023.

Chris Olah. Distributed representations: Composition & superposition, 2023. URL https://transformer-circuits.pub/2023/superposition-composition/index.html.

OpenAI. Multilingual massive multitask language understanding (mmmlu). Hugging Face Datasets, 2024. URL https://huggingface.co/datasets/openai/MMMLU.

Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual BERT? In Anna Korhonen, David Traum, and Lluís Màrquez (eds.), *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 4996–5001, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1493. URL https://aclanthology.org/P19-1493/.

Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, October 1986. ISSN 1476-4687. doi: 10.1038/323533a0.

Keitaro Sakamoto and Issei Sato. Explaining grokking and information bottleneck through neural collapse emergence, 2026. URL https://arxiv.org/abs/2509.20829.

Anton Schäfer, Shauli Ravfogel, Thomas Hofmann, Tiago Pimentel, and Imanol Schlag. The role of language imbalance in cross-lingual generalisation: Insights from cloned language experiments. *arXiv preprint arXiv:2404.07982*, 2024.

Simon Schug, Seijin Kobayashi, Yassir Akram, Maciej Wolczyk, Alexandra Maria Proca, Johannes von Oswald, Razvan Pascanu, Joao Sacramento, and Angelika Steger. Discovering modular solutions that generalize compositionally. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=H98CVcX1eh.

Adam Shai, Loren Amdahl-Culleton, Casper L. Christensen, Henry R. Bigelow, Fernando E. Rosas, Alexander B. Boyd, Eric A. Alt, Kyle J. Ray, and Paul M. Riechers. Transformers learn factored representations, 2026. URL https://arxiv.org/abs/2602.02385.

Jaisidh Singh, Diganta Misra, and Antonio Orvieto. Explaining grokking in transformers through the lens of inductive bias. *arXiv preprint arXiv:2602.06702*, 2026.

Veniamin Veselovsky, Berke Argin, Benedikt Stroebl, Chris Wendler, Robert West, James Evans, Thomas L. Griffiths, and Arvind Narayanan. Localized cultural knowledge is conserved and controllable in large language models, 2025. URL https://arxiv.org/abs/2504.10191.

George Wang and Daniel Murfet. Patterning: The dual of interpretability, 2026. URL https://arxiv.org/abs/2601.13548.

Chris Wendler, Veniamin Veselovsky, Giovanni Monea, and Robert West. Do llamas work in English? on the latent language of multilingual transformers. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15366–15394, Bangkok, Thailand, August 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.acl-long.820. URL https://aclanthology.org/2024.acl-long.820/.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=LqOQ1uJmSx.

Thaddäus Wiedemer, Jack Brady, Alexander Panfilov, Attila Juhos, Matthias Bethge, and Wieland Brendel. Provable compositional generalization for object-centric learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7VPTUWkiDQ.

Yongzhong Xu. The geometry of multi-task grokking: Transverse instability, superposition, and weight decay phase structure, 2026. URL https://arxiv.org/abs/2602.18523.

# A Extended Related Literature

## A.1 Multilingual learning

Modern LLMs are trained on texts from multiple languages. Mechanistic interpretability methods provide evidence that real-world LLMs share certain representations of concepts across multiple languages; even two texts that do not share a single token can yield very similar representations if they express identical semantic content (Wendler et al., 2024; Lindsey et al., 2025; Mantri et al., 2026; Dumas et al., 2025; Brinkmann et al., 2025; Veselovsky et al., 2025). Heiss (2024, Section 1.6) argues theoretically that weight decay induces an inductive bias towards learning such shared representations, based on mathematical theory from Heiss et al. (2021). Furthermore, it is commonly observed that adding more training data in one language improves downstream performance in other languages (Schäfer et al., 2024; Conneau et al., 2020; Pires et al., 2019; Muennighoff et al., 2023).

To obtain more controlled experiments, following K et al. (2020); Dufter & Schütze (2020), Schäfer et al. (2024) conducted semi-synthetic experiments using cloned languages. Because real-world multilingual datasets are inherently messy and naturally contain overlapping vocabularies, they explicitly duplicated the original token vocabulary. This allowed them to express each text either in the original language (using the original tokens) or in the cloned language (using the cloned tokens), guaranteeing that absolutely no tokens are shared across the languages. While anchor points (e.g., shared tokens across languages, such as "computer" in English and German) are known to facilitate cross-lingual generalization, Schäfer et al. (2024) systematically demonstrate that cross-lingual generalization can occur even in this extreme case of completely disjoint token vocabularies,

as theoretically suggested by Heiss (2024, Section 1.6). Our goal is to study a minimally complex dataset that preserves these qualitatively similar phenomena. This simplicity aids mechanistic interpretability and significantly reduces the computational costs of training and inference, enabling extensive experimentation across hyperparameter configurations and making the pipeline accessible to a wider range of researchers. Therefore, we study multilingual modular addition, recently introduced by Dern et al. (2026).

Multilingual modular addition allows for compositional representations in the mechanistic sense described by Olah (2023). Under this perspective, a representation is considered compositional if it decomposes a concept like "blue circle" into independent properties ("blue" and "circle"). This enables non-local generalization; for example, seeing "green circles" and "blue squares" allows a model to generalize to "green squares". For multilingual datasets, a sequence of tokens can be decomposed into its abstract content and its language. If the model learns to decouple these, extracting the content and language in early layers, computing the arithmetic $(a + b \bmod P)$ independent of the language, and ultimately composing the result with the target language in later layers (see Heiss, 2024, Example 1.6.2), it achieves non-local generalization. Such a circuit structure implies that any capability learned in one language automatically transfers to another, even if they share no tokens. Recent empirical work provides further evidence for this decoupled structure in real models. For instance, Bandarkar et al. (2025) achieve zero-shot cross-lingual transfer for mathematical reasoning by taking an LLM trained on English math and swapping its top and bottom layers with those of an LLM trained on the target language.

However, models do not inherently have to learn this specific circuit structure. Heiss (2024, Section 1.6) notes that weight decay induces an inductive bias towards shared representations only for sufficiently deep models, and only if the complexity of translation is sufficiently low compared to the complexity of the core task. Even when these conditions are met, gradient descent may still get stuck in suboptimal local minima. Intuitively, wider layers should reduce the likelihood of converging to bad local minima; for two-layer MLPs, it is proven that the probability of getting stuck in a bad local minimum vanishes asymptotically as the number of neurons per layer tends to infinity (Chizat & Bach, 2018). For deeper architectures, larger models are more likely to contain highly performant sub-networks (winning lottery tickets) at initialization (Frankle & Carbin, 2018). Empirically, Lindsey et al. (2025) observe that Anthropic's smaller models share fewer representations across languages, whereas extensive sharing emerges in their larger models. While one could argue that a computational bottleneck (fewer neurons per layer) might force a model to share concepts, this often makes it harder for gradient descent to locate these solutions in practice. Theoretical results (Heiss et al., 2021; Heiss, 2024) confirm that even models with infinite width maintain an inductive bias to share representations when weight decay is applied. Future work should systematically explore which hyperparameter choices strengthen or weaken these inductive biases.

Xu (2026) studies staggered grokking in shared-trunk transformers trained on multiple distinct modular operations (addition, multiplication, squaring). Our setting differs in keeping the underlying operation identical across languages. On a conceptual level, the synthetic datasets suggested by Caruana (1997); Heiss et al. (2020; 2021); Shai et al. (2026) are similar in that they encourage representations to be shared across multiple sub-tasks, though they are not explicitly framed around multilingual data.[10] Ultimately, any synthetic or real dataset can be adapted into a multilingual format to study these representational dynamics.

### A.2 Detailed Related Work on Compositionality

**Formalizations and Generalization.** Compositionality is the property by which a finite set of concepts and rules can combine to produce a vast range of outputs. It underlies both natural language and systematic generalization in learned systems (Fodor & Pylyshyn, 1988; Lake et al., 2017; Lake & Baroni, 2018), and capturing the compositional structure of the data lets models handle novel combinations of previously seen concepts. A key consequence of this property is the ability to generalize to unseen combinations of known concepts. For example, as Elmoznino et al. (2024) note, if a model is compositional with respect to a set of features in its training data, it need not

[10] Historically, Rumelhart et al. (1986) proposed a conceptually similar dataset dealing with two isomorphic family trees, which could be viewed as two cloned languages describing the same structure. However, the inputs to their models did not use completely disjoint coordinates, as the family relations were represented by the same input coordinates, independent of the specific family tree.

observe all possible combinations of those features in order to generalize to novel ones (Bahdanau et al., 2019; Mittal et al., 2022; Schug et al., 2024; Wiedemer et al., 2024; 2023).

To capture these dynamics, compositionality has been formalized in several complementary ways: as a compression ratio grounded in Kolmogorov complexity (Elmoznino et al., 2024), as a property of the computation graph induced by problem decomposition (Dziri et al., 2023), and mechanistically through the lens of representation geometry (Olah, 2023). As discussed in Appendix A.1, Olah (2023) considers a representation compositional if it disentangles a concept into independent, composable properties (e.g., decomposing a "blue circle" into "blue" and "circle"). In our multilingual modular addition setting, this ideal compositional structure would correspond to the model decoupling a token "(L1:27)" into its mathematical meaning "27" and its language "L1". Furthermore, even for classical modular addition, a token's representation can be decomposed into its Fourier frequencies. While the structure of multilingual modular addition allows a model to nest these two types of decompositions in principle, whether standard transformers actually converge to such perfectly decoupled representations depends heavily on their inductive biases and training dynamics (see Appendix A.1).

**Behavioral Properties of Compositionality.** From a behavioral perspective, Hupkes et al. (2020) formalize compositionality through five empirical tests: *systematicity*, *productivity*, *substitutivity*, *localism*, and *overgeneralization*. Our proposed task of multilingual modular addition serves as an excellent testbed precisely because its underlying structure isolates and evaluates several of these properties:

- **Systematicity** tests whether a model can recombine constituents that have not been seen together during training. (Multilingual) modular addition directly evaluates this: the transition from memorization to generalization (grokking) occurs when the model systematically applies the learned modulo arithmetic rule to unseen combinations of tokens from the $P \times P$ grid. In our case, we can directly measure this via the validation accuracy, as the validation set consists entirely of unseen combinations of tokens.
- **Substitutivity**, as formulated by Hupkes et al. (2020), tests whether a model's behavior is robust to synonymous substitutions, which mathematically corresponds to *invariance* under synonym replacement. Our multilingual variant tests a related, arguably more complex property: *equivariance* under language translation. In multilingual modular addition, the true target function is equivariant with respect to the language; translating the input tokens to a different language requires the output to shift into that corresponding language, even though the underlying mathematical operation remains identical. The validation accuracy of multilingual modular addition inherently tests for this equivariance, which is naturally satisfied if the model learns to structurally decouple the mathematical content from the language identity as in Heiss (2024, Example 1.6.2). Future work could easily introduce pure synonyms into our setup (e.g., multiple disjoint input tokens within the same language mapping to the exact same output token) to explicitly test for invariance alongside this equivariance.
- **Localism** (requiring operations to depend only on immediate constituents), **Productivity** (the ability to extrapolate to longer sequences), and **Overgeneralization** (favoring systematic rules over memorized exceptions) are not directly evaluated as learning challenges in our current setup. Localism is satisfied trivially by definition, as modular arithmetic inherently lacks broader sequence context. Meanwhile, productivity and overgeneralization are not tested because the equations have a fixed length and the underlying target function has no true exceptions. Future work could easily extend our setting to allow for these tests, for instance, by evaluating the model on longer compositional sequences or by introducing artificial exceptions to the training data.

### A.3 Further related work on Grokking and modular addition

**Modular Arithmetic as a Testbed.** The phenomenon of delayed generalization, or "grokking", was initially popularized in the context of neural networks learning small algorithmic tasks, most notably modular arithmetic (Power et al., 2022). While grokking has since been shown to be a general phenomenon driven by representation complexity across diverse, non-algorithmic datasets involving images or language (Liu et al., 2023), modular addition remains the standard "fruit fly"

model for studying capability emergence. This is because its underlying simplicity and exact mathematical ground truth make the transition from memorization to generalization both sharp and fully mechanistically tractable. Mechanistic interpretability work has shown that this transition typically corresponds to the network discarding memorized solutions in favor of structured trigonometric algorithms based on Fourier features (Nanda et al., 2023). By extending this foundational setup into a multilingual variant, Dern et al. (2026) introduce just enough compositional complexity to study shared abstract representations.

**Recent Structural and Optimization Insights.** Recent work has increasingly analyzed the structural drivers and representational geometry of grokking. From a structural perspective, Singh et al. (2026) demonstrate that inductive biases from Transformer architecture (e.g., Layer Normalization position) and optimization settings strongly modulate grokking speed. From a geometric perspective, Sakamoto & Sato (2026) theoretically link grokking to Neural Collapse. They prove that generalization depends on the contraction of the population within-class variance.

**Generalization to Broader Group Structures.** While canonical grokking studies focus on cyclic groups via modular addition, recent work extends these insights to broader algebraic settings. Empirically, Chughtai et al. (2023) showed that networks learning operations on finite groups develop features corresponding to irreducible group representations. Morwani et al. (2024) explained this analytically: for modular addition, sparse parities, and finite group operations, margin maximization alone determines the features learned—Fourier features in the modular case and irreducible representations in the group-operation case. Building on the optimization-dynamics side, Kunin et al. (2026) characterize the training dynamics of two-layer networks on modular addition, proving that networks learn Fourier features in decreasing order of coefficient magnitude. Most recently, Marchetti et al. (2026) introduced the Sequential Group Composition task for arbitrary finite groups, showing that two-layer networks acquire one irreducible representation at a time, in an order determined by the Fourier statistics of the encoding.

## B Formalization of Toy Model Setting

Let $e^{(a)} \in \{0,1\}^{P+1}$ denote the one-hot vector with 1 in the $a$th entry and zeroes elsewhere. The vocabulary of the unilingual modular addition transformer of Power et al. (2022); Nanda et al. (2023) is the set of all such one-hots, i.e.,

$$V_{\text{unilingual}} \coloneqq \left\{ e^{(a+1)} \mid a \in \{0,1,\ldots,P\} \right\}.$$

We think of each $e^{(a+1)}$ as a one-hot representation of $a$ itself, except $e^{(P+1)}$, which represents the "=" sign. We train the model to map

$$(e^{(a+1)}, e^{(b+1)}, e^{(P+1)}) \mapsto e^{(a+b \pmod P)+1)}$$

using the dataset

$$\mathcal{D}^{\text{hard}}_{\text{unilingual}} \coloneqq \left\{ \left( (e^{(a+1)}, e^{(b+1)}, e^{(P)}), e^{(a+b \pmod P)+1)} \right) \mid a, b \in \{0,1,\ldots,P-1\} \right\}$$

with cross-entropy loss.[11]

We expand the unilingual setting by cloning the vocabulary to simulate a multilingual language model whose dataset includes $L$ languages. Formally, this means that the vocabulary with $L$ languages and prime $P$ is

$$V_{L,P} \coloneqq \left\{ e^{(k)} \mid k \in \{1,\ldots,L\cdot(P+1)\} \right\} = \{ e^{(\ell+1)} \otimes e^{(a+1)} \mid a \in \{0,1,\ldots,P\}, \ell \in \{0,1,\ldots,L-1\} \},$$

[11] That is, the loss for inputs $e^{(a+1)}, e^{(b+1)}$ and model output vector $\hat{p} \in \Delta_{P-1}$ is

$$\mathcal{L}((e^{(a+1)}, e^{(b+1)}), \hat{p}) = \text{CrossEntropy}(e^{(a+b \pmod P)+1)}, \hat{p}) = -\sum_{i=1}^{P} \mathbb{1}\{a+b \equiv i-1 \pmod P\} \cdot \log(\hat{p}_i) = -\log\left(\hat{p}_{a+b \pmod P)+1}\right).$$

where $\otimes$ denotes the Kronecker product. We think of $a_{(\ell)} \coloneqq e^{(\ell+1)} \otimes e^{(a+1)}$ as a one-hot representation of the number $a$ in language $\ell$ and $(=)_{(\ell)} \coloneqq P_{(\ell)}$ as "=" in language $\ell$. The model is trained to map

$$(a_{(\ell)}, b_{(\ell)}, (=)_{(\ell)}) \mapsto (a + b \pmod P)_{(\ell)},$$

and does not ever see sequences that include numbers in multiple different languages. In other words, we train with the dataset

$$\mathcal{D}^{\text{hard}}_{L,P} \coloneqq \left\{ \left( (a_{(\ell)}, b_{(\ell)}, (=)_{(\ell)}), \quad (a + b \pmod P)_{(\ell)} \right) \mid a, b \in \{0, 1, \ldots, P-1\}, \ell \in \{0, 1, \ldots, L-1\} \right\}$$

using cross-entropy loss.[12]

## C Ablation Experiment

For a given language $\ell$ and frequency $\omega$, we identify and ablate the corresponding 2-dimensional subspace in the following way:

1. For each $s \in \{0, 1, \ldots, P-1\}$, we average $\boldsymbol{h}$ over the inputs for which $a+b \equiv s \pmod P$ to isolate the part of the residual stream that depends only on the sum $s$:

$$\bar{\boldsymbol{h}}_\ell(s) \coloneqq \frac{1}{P} \sum_{a=0}^{P-1} \boldsymbol{h}_\ell(a,\ s - a \pmod P)$$

2. We transform the matrix $\bar{\boldsymbol{H}}_\ell = \left[\bar{\boldsymbol{h}}_\ell(0) \ \cdots \ \bar{\boldsymbol{h}}_\ell(P-1)\right]$ by taking a discrete Fourier transform along its rows, or equivalently along the $s$ axis, yielding cosine and sine directions for each frequency $\omega$:

$$\boldsymbol{u}_{\ell,\omega} \coloneqq \frac{2}{P} \sum_{s=0}^{P-1} \cos\left(\frac{2\pi\omega s}{P}\right) \bar{\boldsymbol{h}}_\ell(s), \qquad \boldsymbol{v}_{\ell,\omega} \coloneqq \frac{2}{P} \sum_{s=0}^{P-1} \sin\left(\frac{2\pi\omega s}{P}\right) \bar{\boldsymbol{h}}_\ell(s)$$

   We call $S_{\ell,\omega} \coloneqq \operatorname{span}\{\boldsymbol{u}_{\ell,\omega}, \boldsymbol{v}_{\ell,\omega}\}$ the *frequency-$\omega$ plane* of language $\ell$

3. For every language $\ell'$ (including both $\ell' = \ell$ and $\ell' \neq \ell$) and every $a, b \in \{0, \ldots, P-1\}$, we input $(a_{(\ell')}, b_{(\ell')}, (=)_{(\ell')})$ and then ablate $S_{\ell,\omega}$ during the forward pass by projecting it out of the residual stream after attention:

$$\boldsymbol{h}^{\text{ablated}}_{\ell'}(a, b) = \boldsymbol{h}_{\ell'}(a, b) - \operatorname{proj}_{S_{\ell,\omega}}\big(\boldsymbol{h}_{\ell'}(a, b)\big).$$

   In other words, an input from any language is passed through the model with the frequency-$\omega$ plane of language $\ell$ removed, whether or not language $\ell$ matches the input language.

## D The role of the "=" token

We ask what role the "=" token plays in the warmed-up models. In principle it could carry no information beyond marking the end of the input, or it could signal which language the answer should be produced in. We evaluate every mismatched combination of operand and "=" languages, ($a$, $b$, and '=' may each be in a different language) $60$ combinations in total, giving $168{,}540$ examples across the full $P \times P$ grid. The model's output lands in the output block of the "=" token's language $99.93\%$ of the time, regardless of the language the operands were in, while the correct number is produced only $12.71\%$ of the time (compared with $100\%$ when all three tokens share a language). The "=" token is therefore what the warmed-up model uses to decide which language to answer in; the operands' language plays almost no role in that decision. The partial-accuracy pattern in mismatched inputs reflects additional symmetries of the model. The "=" token does not play the same role in the standard-trained models.

[12] This generalizes the unilingual setting as $V_{1,P} = V_{\text{unilingual}}$ and $\mathcal{D}^{\text{hard}}_{1,P} = \mathcal{D}^{\text{hard}}_{\text{unilingual}}$.