Just-in-time and distributed task representations in language models

Anonymous Author(s)
Affiliation
Address
email

Abstract

Many of language models' impressive capabilities originate from their in-context learning: based on instructions or examples, they can infer and perform new tasks without weight updates. In this work, we investigate *when* representations for new tasks are formed in language models, and *how* these representations change over the course of context. We focus on "transferrable" task representations—vector representations that can restore task context in another instance of the model, even without the full prompt. We show that these representations evolve in non-monotonic and sporadic ways, and are distinct from a more inert representation of high-level task categories that persists throughout the context. Specifically, models often condense multiple evidence into these transferrable task representations, which align well with the performance improvement based on more examples in the context. However, this accrual process exhibits strong locality along the sequence dimension, coming online only at certain tokens—despite task identity being reliably decodable throughout the context. Moreover, these local but transferrable task representations tend to capture minimal "task scopes", such as a semantically-independent subtask, and models rely on more temporally-distributed representations to support longer and composite tasks. This two-fold locality (temporal and semantic) underscores a kind of just-in-time computational process underlying language models' ability to adapt to new evidence and learn new tasks on the fly.

1 Introduction

Much of the excitement about large language models began with the discovery that they exhibit In-Context Learning (ICL; Brown et al., 2020): the emergent ability to learn tasks from few-shot examples in context. This discovery has led to a variety of works exploring the behavioral features of ICL (e.g. Sclar et al., 2024; Min et al., 2022). Other works have studied the dynamics of ICL, and how performance improves with increasing numbers of few-shot examples (Agarwal et al., 2024; Anil et al., 2024). The strong behavioral success of ICL led to substantial interest in understanding the mechanistic basis of these capabilities. Work on interpreting the mechanisms responsible for ICL has led to discoveries such as induction heads (e.g. Olsson et al., 2022) and how ICL implicitly refines a model of in-context evidence (e.g. Akyürek et al., 2022; Von Oswald et al., 2023).

Recently, several works have identified internal, vector-form task representations that can be extracted from a model's forward pass on a few-shot prompt (Todd et al., 2024; Hendel et al., 2023). Importantly, these task representations not only capture general task information, but can be used to restore the appropriate task context during the model's forward pass on a zero-shot prompt. This transfer effect is observed by intervening with that representation at the appropriate place in the model's residual stream—such that task context is reinstantiated and the model can perform the task without any

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

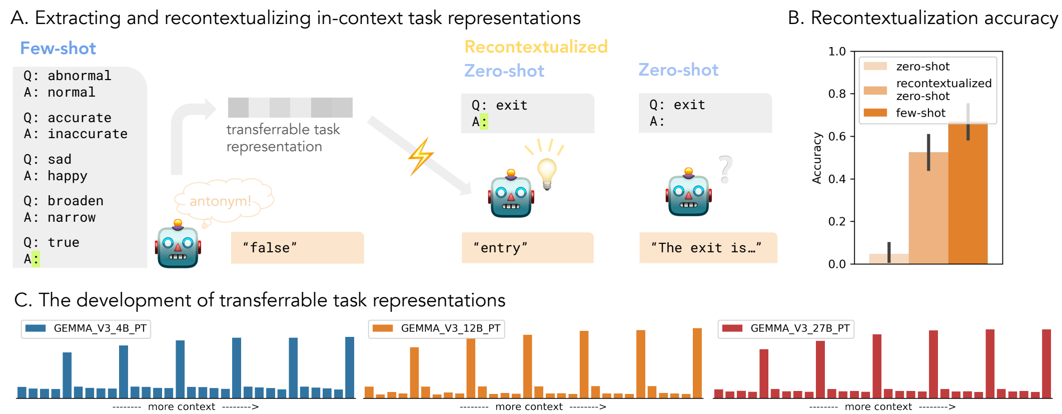

Figure 1: Understanding how task representations develop over context. **A**. A schematic of extracting transferrable task representations and restoring task contexts in zero-shot settings. The highlighted tokens indicate the source and target for extracting and injecting task representations. **B**. Transferrable task representations restore task accuracy on zero-shot prompts. Results are aggregated over all models for simple tasks (see Appendix A). Error bars indicate the 95% CI over tasks. **C**. An overview of the development of transferrable task representations over context, i.e. the recontextualized zero-shot accuracy for task representations extracted from different tokens throughout the context.

explicit instructions or demonstrations. These "transferrable" task representations have been shown to exist across a variety of tasks and presentation formats, and even capture transferrable task knowledge across modalities (Davidson et al., 2025; Huang et al., 2024; Luo et al., 2024).

The discovery of transferrable task representations raises several intriguing questions about their dynamics and generality. How and when are these internal task representations formed throughout the context? How do these dynamics depend upon task complexity? A simple, intuitive hypothesis is that these representations develop gradually, e.g. at a rate that depends on task complexity. The representations might accrue evidence from each example and refine monotonically into a more stable, robust task representation. This view aligns with the behavioral findings that models perform better with more examples in-context (Agarwal et al., 2024; Anil et al., 2024).

We set out to understand how the dynamics of ICL (within and across examples) are exhibited representationally within the model, by investigating task representations. Our findings suggest a more complicated picture of the computational process that support language models' adaptation to a new task:

- For simple tasks, we find two types of task representations in language models: an inert representation of task identity is more continuously present throughout the context, but transferrable task representations only activate sporadically at key tokens.
- These fleeting but transferrable task representations often condense evidence from multiple examples but tend to capture minimal task scopes, as their ability to guide model behavior decays over longer generation and across independent subtask contexts.
- Models do not appear to condense task knowledge into local task representations in more complicated tasks that require state tracking or chaining multiple subtasks together.
- Finally, models can form distinct representations to support generating the same responses when solving tasks independently vs. as part of a broader context.

Overall, these results give us a window into language models' changing state when inferring and solving new tasks in context, but paint a complex and nuanced picture of the dynamics of this state. There are different types of task representations—identifiable vs. transferrable—that evolve over the context in distinct ways. The representations of tasks also depend on the task complexity and the surrounding context structure in which a task is embedded. These results may have implications for both the science of understanding models, and practical applications of mechanistic interpretability for analysis and safety.

2 Methods

Tasks For our analyses, we built upon tasks from prior work on transferrable task representations (Hendel et al., 2023; Todd et al., 2024). The tasks we examine include a diverse set of natural language tasks (e.g., finding the antonym of a query word, translating an English word to French) and algorithmic tasks (e.g., counting or extracting a target word from a list of input words). In addition to these simple, single-token generation tasks from the previous literature, we also test a range of new tasks to explore model behavior in longer generation settings. These include: repeating a simple task three times (e.g. ANTONYM X 3 requires finding the antonyms of three input words), extracting multiple words from a query word list (e.g., choose the first and last words from a word list), and reversing or shifting an entire word list. Finally, we also explore a set of "mixed-generation" tasks, where the model needs to infer and perform different tasks on each word in a word list. (See Appendix A for the full set of tasks.)

There are 512 query-answer pairs for each task (except for two smaller datasets: COUNTRY-CAPITAL contains 197 samples, and PRODUCT-COMPANY contains 494 samples). These query-answer pairs are formatted into few-shot prompts with alternating "Q:" and "A:" turns, as shown in Figure 1A.

Models We focus our analyses on the open-weight Gemma V3 pre-trained models, including the 4B, 12B, and 27B-sized models (Team et al., 2025).

Extracting in-context task representations We primarily investigate task vectors discovered in (Hendel et al., 2023) as a window to study language models' in-context task representations. We extract task vectors from few-shot prompts consisting of query-answer pairs and a test query as shown in Figure 1A. Task vectors are the layer residual activations extracted from the last token before answer generation in the few-shot prompts (in the example in Figure 1A, this corresponds to the highlighted colon). Hendel et al. (2023) showed that task vectors can reinstate task performance even without any additional context and with a different query—when task vectors are patched onto (i.e., overwrite) the layer residual activations of the last token, they can recontextualize the model with the appropriate task context and enable the model to generate the task output without any few-shot examples in-context.

We replicate and extend the procedure outlined in Hendel et al. (2023). For each model and task, we first search for the layer that best captures the task representation, using 50 queries from the dataset as the development set and in an 8-shot setting. As in prior work, we replace the real test queries with dummy queries sampled from the dataset to extract query-agnostic, general task representations. We searched among every 3 layers starting at layer 2 (0-indexed) for the 4B and 12B models (covering both the local-attention layers and global attention layers in Gemma V3 models; Team et al., 2025), and every 6 layers starting from layer 5 in the 27B model. The layer that restores the highest task accuracy on zero-shot prompts in the development set is designated as the layer that best captures the representation for a given task, and subsequently used to extract task vectors and restore task contexts for the remainder, held-out queries in the dataset. Consistent with prior results, we generally find that task vectors extracted and injected at middle layers restore the highest task accuracy on zero-shot prompts, for all model sizes.

Evaluating task accuracy We compare the average accuracy for the sampled responses across a few settings: standard zero-shot, recontextualized zero-shot setting (with task vector intervention), and few-shot. For simplicity, responses for all tasks are graded by exact string matches against the ground-truth answer. This underestimates the model performance in some tasks (e.g. for antonym and translation tasks), but we use the same grading scheme across all settings and compare relative performances. For longer-generation and mixed-generation tasks, we evaluate each of the multiple outputs separately by exact match (e.g., in ANTONYM X 3, we compare each of the three output words with the correct answer), and report the mean accuracy across all outputs in the longer response.

Examining the dynamics of task representations Once we determine the best layer for each task using the last colon token, we evaluate how well the colon token representations condense the k-shot information in a prompt. We do this by extracting task vectors at the colon token for a k-shot prompt, and then evaluating the recontextualized zero-shot accuracy when the task vectors are patched into a zero-shot prompt. We repeat this analysis for k in 1, 2, 4, 8, 16, 32. We also experimented with allowing the best layer to vary depending on k, but found very similar results.

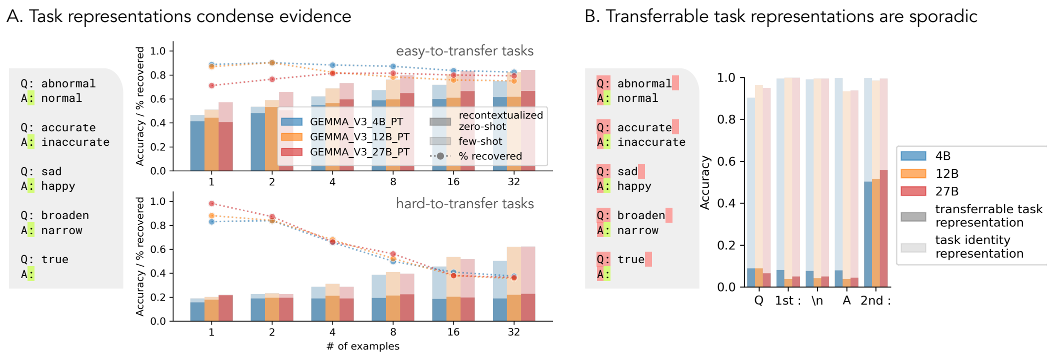

Figure 2: Sporadic & inconsistent evidence accrual in language models. **A**. Task vectors extracted from the last colon token in each example capture evidence accrual on most tasks (11 out of 14). However, on three "hard-to-transfer" tasks, task vectors do not capture this evidence accumulation, even though the models (behaviorally) do learn from more examples. The solid bars indicate recontextualized zero-shot accuracy (via task vectors), and light bars in the background indicate 8-shot accuracy (without task vectors). The dotted lines indicate the ratio of the recontextualized zero-shot accuracy against 8-shot accuracy. **B**. Most other format tokens in the context do not robustly form transferrable task representations that support recontextualization on zero-shot, but task identity is reliably decodable in their residual activations. Here, we report the task identity decoding accuracy at the mode best layer at which transferrable task representations form in the second ":" token. See the main text and Appendix C for more details.

We also studied whether transferrable task representations form in tokens other than the final colon token. In particular, we extracted layer residual token activations for other tokens in the context, including the "Q", the ":" following "Q", the "A", and the new-line token before "A". We patched these token activations onto the corresponding token in the zero-shot prompt at the same layer. For each of the non-final tokens, we repeated the search for the layer that best captures task representations. All token representations were evaluated on the extent to which they restored task accuracy on zero-shot prompts.

3 Results

How do language models infer new tasks in-context? We leverage the ability for in-context task representations to restore task contexts to understand how models accrue evidence and refine task representations. We find that language models indeed form stronger task representations that aggregates in-context evidence. However, this evidence accrual process is surprisingly non-gradual and happens in a sporadic way for most tasks (Figure 1C). In particular, effective and transferrable task representations form only at certain tokens in the context, and tend to capture the minimal "task scope." For some tasks, models also do not appear to condense task information into local task representations. Below, we discuss these findings in more detail.

3.1 Local task representations can accrue evidence

Task representations can reflect evidence accrual. Consistent with the behavioral performance gain from including more examples in-context (e.g., Anil et al., 2024), we find that transferrable task representations in models also reflect increased task certainty with increased context. As shown in Figure 2A, task vectors extracted from more examples are better at restoring task performance in zero-shot settings, such that the ratio between the recontextualized zero-shot accuracy and few-shot accuracy stays relatively stable across the number of examples. This suggests that language models condense information from multiple examples and form better task representations, even when the task representations extracted are fairly local (i.e., at the last token in the prompt).

... but not for all types of tasks. However, we did not observe evidence accrual in 3 out of 14 of the simple tasks (Figure 2A, hard-to-transfer tasks). Specifically, for CHOOSE_MIDDLE_OF_5, COUNT_COLOR_IN_3, and COUNT_FRUIT_IN_3 tasks, we found that local task representations (extracted from the last token) were not able to take advantage of more examples for task transfer to zero-shot settings, even though models improved substantially at solving the task when given more examples in the prompt. One interpretation is that some tasks require more state-tracking (such as counting), and that these may necessitate additional inference processes that the models do not condense into local task representations. Alternatively, these inference processes cannot be effectively re-activated by the injection of the extracted task representations.

How evidence accrual leads task representations to converge. For the eleven other simple tasks where we successfully observed evidence accrual, we sought to understand how the task representations themselves changed over more examples (Figure 3A). As we increased the number of examples in context, we generally found reduced variance among task vectors extracted from different k-shot samples. This suggests that, as models gain evidence, in-context task representations tend to denoise or converge to more stable representations. The magnitude (L2-norm) of the task vectors also tends to decrease over time. However, there were noticeable differences between tasks and models. Some tasks seem to converge to stable representations faster (i.e., with fewer examples; see also a visualization of the representational trajectories in Figure S2). However, for certain tasks, the magnitude of the task representations first increases then decreases given more examples. We note that for this analysis, we look at the task representations extracted from the mode best layer across different tasks. This is to control for magnitude difference of the residual activations across layers and make a fair comparison. Although the best layer for transferrable task representations does sometimes differ across tasks, the best layer across tasks tend to reside in the middle layer range across all model sizes, consistent with prior findings (Hendel et al., 2023).

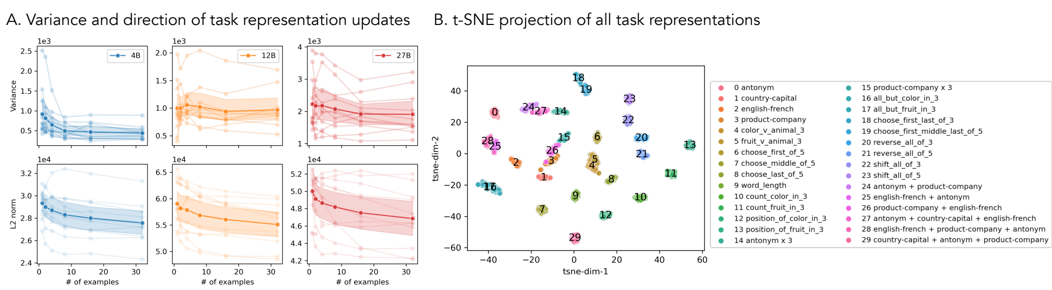

Figure 3: Analyses of extracted task representations. **A**. The extracted task vectors (at the last colon token) decrease in both variance and magnitude with more examples, exhibiting a general tendency to condense evidence and converge onto stable task representations. The solid line shows the average across tasks, the transparent lines show the individual tasks. **B**. Extracted task vectors form distinct clusters. When a given task was evaluated independently vs embedded within a larger task structure (e.g. "repeat [X task] 3 times"), the task vectors were similar but distinguishable. Figure shows results from the 27B model; see other models in Figure S4.

3.2 Task representations exhibit temporal locality

The analyses above confirm that, intuitively, in-context task representations can successfully benefit from increasing evidence in context (along the “temporal” dimension), and converge to better representations that restore task contexts more robustly. To understand the full temporal profile of this accrual process, we repeated the task vector recontextualization experiment on *other* tokens in the prompt, which revealed that the transferrable task representations do not strengthen monotonically.

Transferrable task representations are not found in most tokens. We extracted task vectors from the format tokens in the prompt, including "Q", the ":" following "Q", "A", and the new-line token before the "A". We tested whether these extracted representations can also restore the corresponding task contexts when patched onto the forward pass in the zero-shot prompt (as before, we patched at the same layer, but onto the corresponding format token instead of the last colon). As shown in Figure 2B, transferrable task representations generally do not robustly form in the residual activations across layers in these tokens, even though these tokens are also shared across examples, tasks, and contexts.

This is true across the number of examples provided in the prompt, leading to the developmental trajectory of in-context task representations shown in Figure 1C.

We observed nearly zero recontextualized zero-shot accuracy for all these tokens in most tasks, except some restoration success in PRODUCT-COMPANY, COLOR_V_ANIMAL_3, CHOOSE_FIRST_OF_5, and the longer-generation tasks discussed below (see Figure S3). In general, it seems that an effective, transferrable task representation in language models only forms sparingly; in few-shot settings, this often means a just-in-time task representation at the token just before answer generation for each query.

... but a robust task identity signal persists throughout the context. Intriguingly, however, task identity is almost perfectly decodable in the representations extracted from *all* the different format tokens, even though the formats are shared across the tasks and contexts. To study this, we trained simple, single-layer linear decoders to predict the task category (among the eleven simple tasks that we observed evidence accrual) from the residual activations of each format token. We report the test accuracy on the 25% held-out token activations extracted following eight examples in Figure 2B and Appendix C, but found high task identity decoding accuracy across these tokens following different number of examples. This decodability success may be partly due to vocabulary differences for the tasks, but the accuracy is close to ceiling, and not all tasks are distinguishable in this way. We note that task identity is not perfectly decodable in the token activations in early layers, but generally reaches high accuracy in mid and late layers. This suggests that the model is generally *task-sensitive*, but instantiates *transferrable* task representations only at particular timepoints in the context.

3.3 Task representations exhibit scope locality

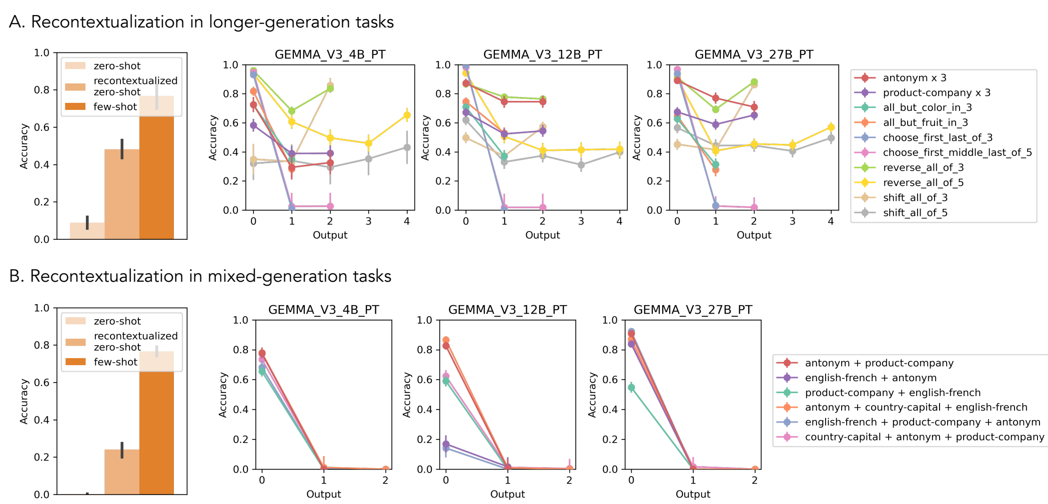

Figure 4: Reinstantiated task contexts in longer- and mixed-generation tasks often decay over longer generation, especially for tasks that can be decomposed into semantically-independent subtasks. This suggests a tendency for models to only activate transferrable representations for small task scopes. **A**. Left: recontextualized zero-shot accuracy compared to zero-shot and 8-shot accuracy on longer-generation tasks. Right: recontextualized accuracy for each output unit across models, conditioned on sequences where models generated full correct responses with eight examples in-context. An output unit usually corresponds to a single word and is occasionally a short phrase (e.g. the capital of a country). **B**. Visualization as in A, but for mixed-generation tasks.

We have seen evidence that transferrable task representations tend to be temporally local. That leads to the question of whether they have a lasting effect over generation – that is, are the restored task contexts in the zero-shot forward pass also fleeting in nature? Prior work has mostly focused on simple, single-token output tasks. To study the semantic scope of task representations, we tested to what extent restored task contexts can support longer generation beyond the first token. Building on the simple tasks from prior work, we evaluated models on a set of longer-generation and mixed-generation tasks, including repeating a simple task multiple times on different input words, list-level tasks that operate over multiple words, and inferring/performing different tasks on different words (see Methods and Appendix A).

Transferrable task representations tend to support a semantically-independent task scope. In these experiments, we find further evidence of the locality of in-context task representations. Overall, the recontextualized zero-shot accuracy of tasks that require longer and mixed answers is substantially lower than that in tasks that require shorter answers (Figure 4, left; also see Figure S1). Across a range of tasks, we find that the recontextualized zero-shot accuracy decreases for each output word (Figure 4, right), suggesting that the restored task contexts "fade" over longer generation.

Models seem to be consistently decomposing tasks and forming local task representations that capture a minimal "task scope". For example, in CHOOSE_FIRST_MIDDLE_LAST_OF_5, the reinstantiated task context only supported generating the first word, suggesting that further token generations rely on additional representations instantiated by a separate mechanism while processing the previously-generated tokens. This effect is very pronounced in the mixed-generation tasks, as all models form strong local task contexts that only encapsulate the first subtask in a multi-task chain and defer representing later subtasks. Interestingly, the extracted task representations that support generating responses for the same simple task when it appears independently or as a first task in a repeated or mixed-task context can be distinct (Figure 3B). This can potentially reflect the inert sensitivity to the broader mixed-task context, as well as conveying information relevant for later subtasks to be activated, even though these signals are not sufficient to directly restore execution of later subtasks.

The restored task contexts do support longer generation to some extent in some tasks, including repeating a simple task multiple times and the list-level tasks (i.e. reversing or shifting a word list). Overall, these results suggest that language models may be automatically segmenting semantically-independent task scopes when possible, such that task representations for longer or composite tasks are offloaded onto multiple tokens.

4 Related work

Since the discovery that large language models exhibit emergent in-context learning (Brown et al., 2020), there has been substantial interest in investigating this capability and its mechanistic basis.

From a behavioral perspective, many subsequent works have explored how ICL could develop from implicit meta-learning of data properties (Xie et al., 2022; Chan et al., 2022), and how this may relate to the broader set of language model capabilities (Chen et al., 2024; Lampinen et al., 2024). Some of this work has focused on the surprising fragility of ICL to subtle prompt changes (e.g. Sclar et al., 2024); conversely, others have highlighted how ICL may be *overly* robust, allowing "learning" common tasks even if the labels are randomized (Min et al., 2022). One particularly relevant focus of behavioral work on ICL has been on the *dynamics* of in-context learning; for example, how adding many example shots can improve performance on difficult tasks, or even those discouraged in post-training (Agarwal et al., 2024; Anil et al., 2024).

From a mechanistic perspective, Olsson et al. (2022) showed how induction heads could support in-context learning, and other work has studied how they might develop over training (Edelman et al., 2024; Singh et al., 2025). More recently, attention has turned to cases where models may create internal task representations, that can be extracted from few-shot prompts and then injected (without the few-shot examples in context) to induce task performance. Hendel et al. (2023) demonstrated an instance of such task representations: representations at intermediate layers of the model at a key token that can be injected to mimic the effect of a few-shot prompt. Concurrent work from Todd et al. (2024) identified "function vectors," which aggregate the effects of multiple attention heads that convey task information. Subsequent work has generalized and extended these findings, for example exploring how function vectors can emerge from instructions as well as examples (Davidson et al., 2025) and how task vectors capture task representations in multimodal models (Huang et al., 2024; Luo et al., 2024). Other works have explored how these representations and structures emerge over training (Yang et al., 2025; Yin and Steinhardt, 2025), and extended existing methods to more robustly restore task contexts in zero-shot settings (Li et al., 2024). Our work builds on these findings and use transferrable task contexts as a window to study the dynamics of in-context task representations in language models.

5 Discussion

We sought to understand the dynamics of in-context task representations that support language models' successful learning of new tasks. Building on prior methods, we evaluated when in the context we can extract transferrable task representations from few-shot settings that restore task context in zero-shot settings. Our results show that in many tasks, models refine task representations over more evidence such that the representations more successfully restore task contexts. However, these transferrable task representations only sporadically activate, and seem to best support a minimal task scope (e.g., a first irreducible subtask). The dynamics of these effective, transferrable task representations strongly contrast with a general sensitivity to high-level task differences that persists throughout the context. In addition, we find many cases where models do not appear to condense global task representations into a local, token-level representation, such as tasks involving more state tracking and tasks chaining different types of subtasks together.

In general, our results complicate the intuitive picture that the computational process underlying language model's in-context evidence accrual and task inference is smooth and gradual. Models do not appear to refine task representations at a per-token basis, not even in a step-wise manner—even on a per-example basis, the task state is not sustained, but often fades and reactivates across different tokens. That language models elect to condense task evidence onto a single token at intermediate layers rather than relying on repeated cross-token attention to aggregate information at multiple layers may suggest a general inductive bias to compress knowledge to local representations when possible. This may also relate to the success many works have observed on extracting transferrable task representations in broad settings, such as following instructions or images (Davidson et al., 2025; Huang et al., 2024; Luo et al., 2024), or capturing information for multiple possible task outcomes (Xiong et al., 2024).

However, one trend that arose from these investigations is that language models also do not seem able to condense task information into local representations in all cases. Rather, they exhibit a tendency to form sharp local contexts for small task units, and offload broader task contexts such as mixed or multiple tasks across time. As discussed earlier, even for simple tasks where some intermediate-state tracking may be required, successful inference may need to rely on cross-token and cross-layer computation (e.g., as shown in Ameisen et al., 2025). In these cases, the effective restoration of the computation process may also require intervening multiple components during the forward pass. It is also possible that by overriding token activations with task vectors in intermediate layers, the models may have lost any task state information formed in earlier layers, and other methods that restore task states through additive injection rather than activation patching may be more successful at restoring model task states in these tasks (Todd et al., 2024; Li et al., 2024). Understanding how more complicated in-context computation develops based on evidence in these tasks and whether vector-form task representations can re-activate these inferences are interesting open questions.

An intriguing direction for future work is to study whether there are mechanistic bases for the strong temporal and scope locality we observed in models' in-context task representations. One possibility may be that the residual stream is more stable and easier to learn from during training. This may encourage the model to rely more on the residual stream to condense contextual task representations rather than relying on the more expensive attention operations. These learning dynamics may drive models to conform with an implicit normative consideration to not instantiate task contexts until needed and instantiate just the right scope to avoid capacity waste. Some of these features may even be exclusive to models pre-trained on natural languages (e.g. Yang et al., 2025).

Limitation We note a few important limitations of our work. First, we primarily rely on task vectors as the method to extract transferrable task representations. This means that our conclusions and speculations are bounded by the effectiveness of this method. As we discussed earlier, representations for some task contexts may be more distributed, either across tokens and/or across model layers. It would be important to confirm if similar dynamics are observed in less-constrained methods such as a multi-layer recontextualization (Li et al., 2024). Second, we mostly explored relatively simple tasks, including when we investigated longer-generation tasks. It's possible that many of the dynamics we observe here would not generalize to settings with naturalistic languages, especially when the tasks are not so cleanly decomposable and a single, semantically-independent task unit is hard to define.

Conclusion We investigated how the dynamics of in-context learning are reflected in the development of language models' internal task representations. Our results suggest that language models do not smoothly refine a global task state in-context. While general task sensitivity persists throughout context, models appear to construct effective task representations in a "just-in-time" fashion to solve a simple task scope immediately ahead. The fleeting, minimally-scoped nature of these in-context task representations provides new insight into the models' state of inferring and performing tasks based on new evidence.

References

Agarwal, R., Singh, A., Zhang, L., Bohnet, B., Rosias, L., Chan, S., Zhang, B., Anand, A., Abbas, Z., Nova, A., et al. (2024). Many-shot in-context learning. *Advances in Neural Information Processing Systems*, 37:76930–76966.

Akyürek, E., Schuurmans, D., Andreas, J., Ma, T., and Zhou, D. (2022). What learning algorithm is in-context learning? investigations with linear models. *arXiv preprint arXiv:2211.15661*.

Ameisen, E., Lindsey, J., Pearce, A., Gurnee, W., Turner, N. L., Chen, B., Citro, C., Abrahams, D., Carter, S., Hosmer, B., et al. (2025). Circuit tracing: Revealing computational graphs in language models. *Transformer Circuits Thread*, 6.

Anil, C., Durmus, E., Panickssery, N., Sharma, M., Benton, J., Kundu, S., Batson, J., Tong, M., Mu, J., Ford, D., et al. (2024). Many-shot jailbreaking. *Advances in Neural Information Processing Systems*, 37:129696–129742.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. (2020). Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Chan, S., Santoro, A., Lampinen, A., Wang, J., Singh, A., Richemond, P., McClelland, J., and Hill, F. (2022). Data distributional properties drive emergent in-context learning in transformers. *Advances in neural information processing systems*, 35:18878–18891.

Chen, Y., Zhao, C., Yu, Z., McKeown, K., and He, H. (2024). Parallel structures in pre-training data yield in-context learning. In *62nd Annual Meeting of the Association for Computational Linguistics, ACL 2024*, pages 8582–8592. Association for Computational Linguistics (ACL).

Davidson, G., Gureckis, T. M., Lake, B. M., and Williams, A. (2025). Do different prompting methods yield a common task representation in language models? *arXiv preprint arXiv:2505.12075*.

Edelman, E., Tsilivis, N., Edelman, B., Malach, E., and Goel, S. (2024). The evolution of statistical induction heads: In-context learning markov chains. *Advances in neural information processing systems*, 37:64273–64311.

Hendel, R., Geva, M., and Globerson, A. (2023). In-context learning creates task vectors. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Huang, B., Mitra, C., Arbelle, A., Karlinsky, L., Darrell, T., and Herzig, R. (2024). Multimodal task vectors enable many-shot multimodal in-context learning. *Advances in Neural Information Processing Systems*, 37:22124–22153.

Lampinen, A. K., Chan, S. C., Singh, A. K., and Shanahan, M. (2024). The broader spectrum of in-context learning. *arXiv preprint arXiv:2412.03782*.

Li, Z., Xu, Z., Han, L., Gao, Y., Wen, S., Liu, D., Wang, H., and Metaxas, D. N. (2024). Implicit in-context learning. *arXiv preprint arXiv:2405.14660*.

Luo, G., Darrell, T., and Bar, A. (2024). Vision-language models create cross-modal task representations. *arXiv preprint arXiv:2410.22330*.

Min, S., Lyu, X., Holtzman, A., Artetxe, M., Lewis, M., Hajishirzi, H., and Zettlemoyer, L. (2022). Rethinking the role of demonstrations: What makes in-context learning work? In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11048–11064.

Olsson, C., Elhage, N., Nanda, N., Joseph, N., DasSarma, N., Henighan, T., Mann, B., Askell, A., Bai, Y., Chen, A., et al. (2022). In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*.

Sclar, M., Choi, Y., Tsvetkov, Y., and Suhr, A. (2024). Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *The Twelfth International Conference on Learning Representations*.

Singh, A. K., Moskovitz, T., Dragutinović, S., Hill, F., Chan, S. C., and Saxe, A. M. (2025). Strategy coopetition explains the emergence and transience of in-context learning. In *Forty-second International Conference on Machine Learning*.

Team, G., Kamath, A., Ferret, J., Pathak, S., Vieillard, N., Merhej, R., Perrin, S., Matejovicova, T., Ramé, A., Rivière, M., et al. (2025). Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*.

Todd, E., Li, M. L., Sharma, A. S., Mueller, A., Wallace, B. C., and Bau, D. (2024). Function vectors in large language models. *The Twelfth International Conference on Learning Representations*.

Von Oswald, J., Niklasson, E., Randazzo, E., Sacramento, J., Mordvintsev, A., Zhmoginov, A., and Vladymyrov, M. (2023). Transformers learn in-context by gradient descent. In *International Conference on Machine Learning*, pages 35151–35174. PMLR.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. (2022). An explanation of in-context learning as implicit bayesian inference. In *International Conference on Learning Representations*.

Xiong, Z., Cai, Z., Cooper, J., Ge, A., Papageorgiou, V., Sifakis, Z., Giannou, A., Lin, Z., Yang, L., Agarwal, S., et al. (2024). Everything everywhere all at once: Llms can in-context learn multiple tasks in superposition. *arXiv preprint arXiv:2410.05603*.

Yang, L., Lin, Z., Lee, K., Papailiopoulos, D., and Nowak, R. (2025). Task vectors in in-context learning: Emergence, formation, and benefit. *arXiv preprint arXiv:2501.09240*.

Yin, K. and Steinhardt, J. (2025). Which attention heads matter for in-context learning? In *Forty-second International Conference on Machine Learning*.

A Tasks

Table 1: Simple/shorter-answer tasks. See Todd et al. (2024) for more details.

Task Name	Example
ANTONYM	Q: true A: false
COUNTRY-CAPITAL	Q: Germany A: Berlin
ENGLISH-FRENCH	Q: queens A: reines
PRODUCT-COMPANY	Q: Windows XP A: Microsoft
COLOR_V_ANIMAL_3	Q: blue, dolphin, swan A: blue
FRUIT_V_ANIMAL_3	Q: lime, parrot, buffalo A: lime
CHOOSE_FIRST_OF_5	Q: envelope, pasta, cake, toucan, create A: envelope
CHOOSE_MIDDLE_OF_5	Q: candy, charismatic, laptop, realize, eel A: laptop
CHOOSE_LAST_OF_5	Q: affable, believe, carefree, zoom, moray A: moray
WORD_LENGTH	Q: negotiate A: 9
COUNT_COLOR_IN_3	Q: snake, gold, indigo A: two
COUNT_FRUIT_IN_3	Q: lime, newt, bunny A: one
POSITION_OF_COLOR_IN_3	Q: monkey, oryx, white A: third
POSITION_OF_FRUIT_IN_3	Q: pear, coyote, capybara A: first

Table 2: Longer-generation tasks.

Task Name	Example
ANTONYM X 3	Q: fall, everybody, intact A: rise, nobody, broken
PRODUCT-COMPANY X 3	Q: iWork, Windows NT 3.5, OS X Yosemite A: Apple, Microsoft, Apple
ALL_BUT_COLOR_IN_3	Q: cat, black, pelican A: cat, pelican
ALL_BUT_FRUIT_IN_3	Q: grape, butterfly, llama A: butterfly, llama
CHOOSE_FIRST_LAST_OF_3	Q: white, house, wallet A: white, wallet
CHOOSE_FIRST_MIDDLE_LAST_OF_5	Q: dolphin, beyond, curtain, pillow, intuitive A: dolphin, curtain, intuitive
REVERSE_ALL_OF_3	Q: donut, sad, who A: who, sad, donut
REVERSE_ALL_OF_5	Q: she, honest, out, test, frog A: frog, test, out, honest, she
SHIFT_ALL_OF_3	Q: piano, cougar, jackfruit A: cougar, jackfruit, piano
SHIFT_ALL_OF_5	Q: agreeable, flamingo, short, around, jovial A: flamingo, short, around, jovial, agreeable

Table 3: Mixed-generation tasks.

Task Name	Example
ANTONYM + PRODUCT-COMPANY	Q: opponent, iDisk A: ally, Apple
ENGLISH-FRENCH + ANTONYM	Q: liberal, continue A: libéral, stop
PRODUCT-COMPANY + ENGLISH-FRENCH	Q: Alfa Romeo MiTo, mask A: Fiat, masque
ANTONYM + COUNTRY-CAPITAL + ENGLISH-FRENCH	Q: upper, Greece, artists A: lower, Athens, artistes
ENGLISH-FRENCH + PRODUCT-COMPANY + ANTONYM	Q: system, Lancia Flavia, unlucky A: système, Fiat, lucky
COUNTRY-CAPITAL + ANTONYM + PRODUCT-COMPANY	Q: Gambia, heavy, Game & Watch A: Banjul, light, Nintendo

B Additional figures

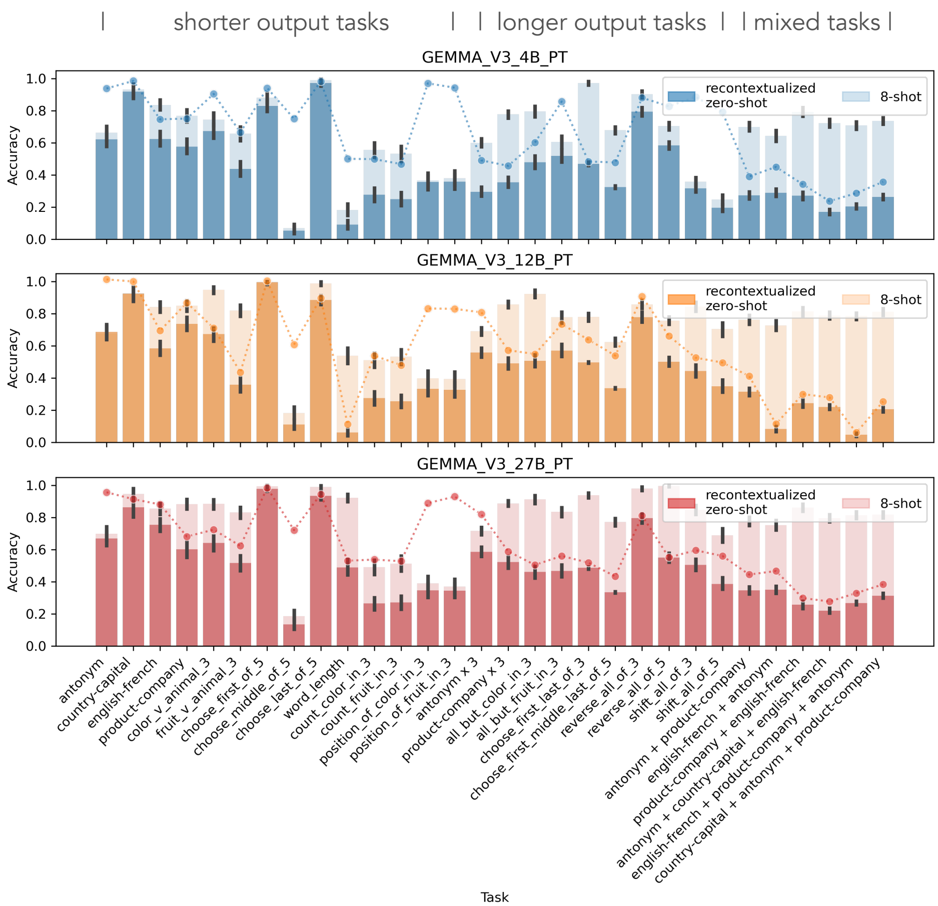

Figure S1: Recontextualization accuracy for all tasks. Task vectors extracted from 8-shot prompts are used to reinstantiate task contexts in zero-shot settings. The dotted line indicate the ratio between recontextualized zero-shot accuracy and 8-shot accuracy.

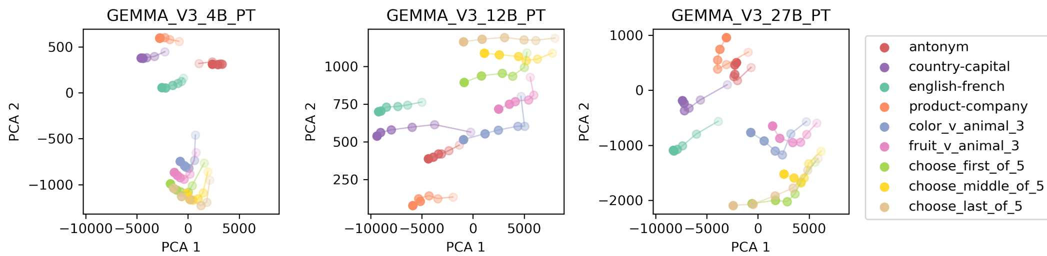

Figure S2: Developmental trajectory of task representations over shots. Task representations are the token activations of the colon token prior to answer generation. We visualize task vectors sourced from the mode best layer across tasks at which task contexts are best restored in a zero-shot setting. Representations are first averaged across samples with the same number of examples for each task.

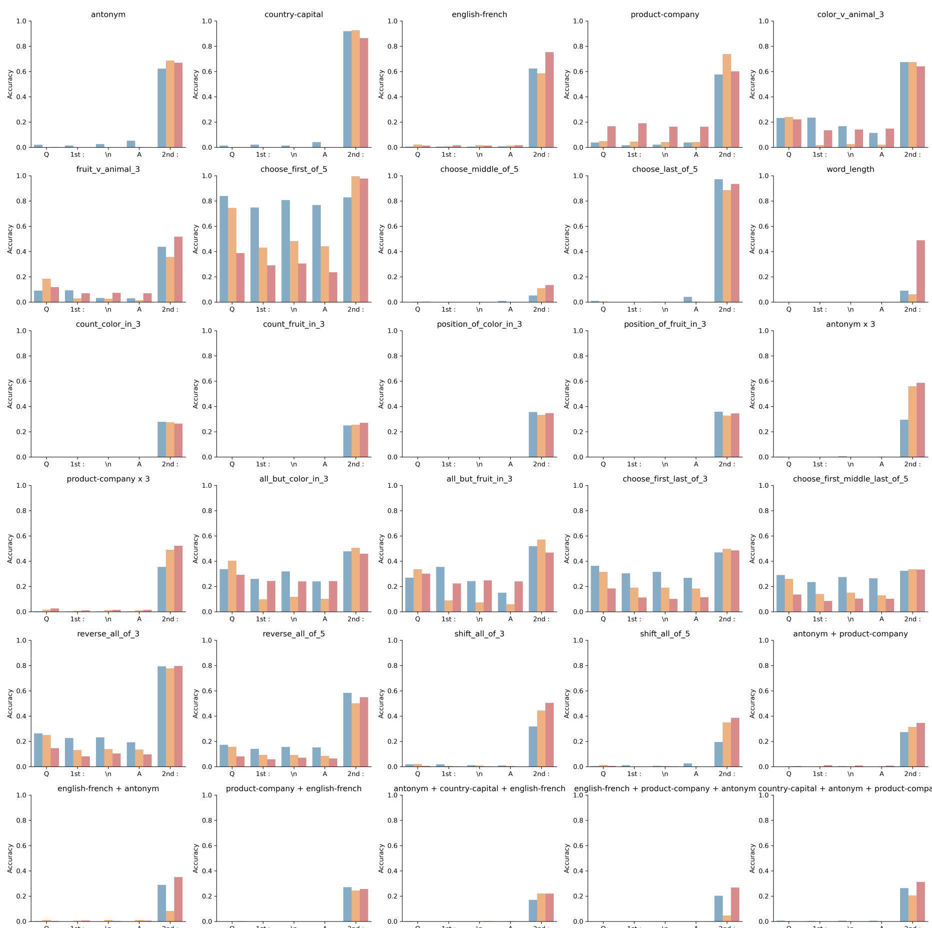

Figure S3: Recontextualized zero-shot accuracy from different format tokens in the prompt. The colors indicate different models (see Figure S1).

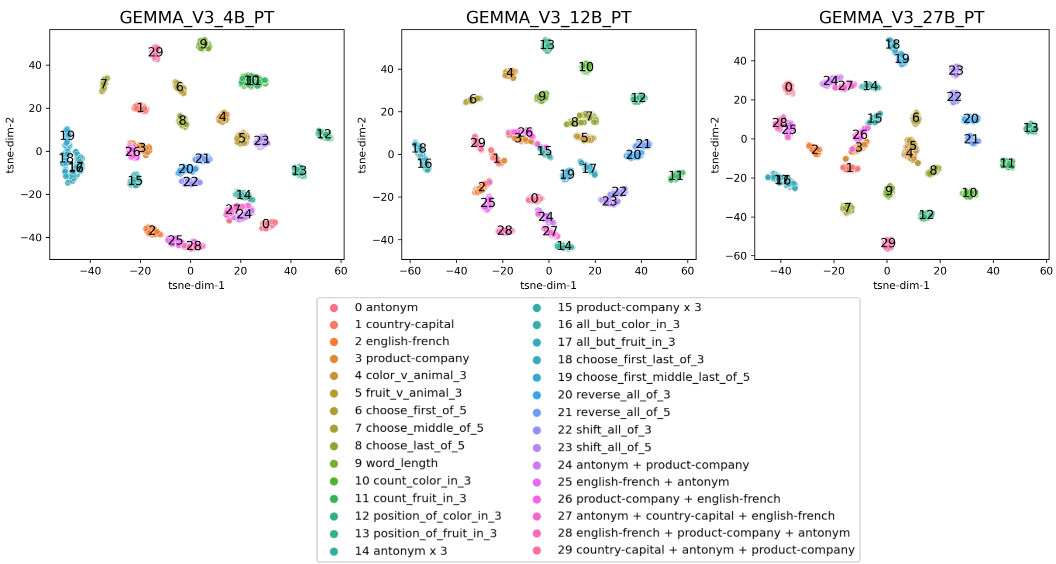

Figure S4: Task representations across all models and tasks.

C Additional results

Table 4: Decoding accuracy for task identity at different tokens. Results shown for token activations extracted at the 8th example.

model	token / layer	Q	1st :	\n	A	2nd :
GEMMA_V3_4B_PT	2	0.933	0.848	0.925	0.899	0.950
	5	0.979	0.980	0.816	0.925	0.994
	8	0.952	0.986	0.908	0.959	0.996
	11	0.950	0.986	0.968	0.984	0.998
	14	0.960	0.990	0.988	0.984	0.996
	17	0.902	0.994	0.990	0.996	0.996
	20	0.918	0.992	0.970	0.982	0.992
	23	0.834	0.988	0.992	0.954	0.996
	26	0.842	0.865	0.990	0.916	0.992
	29	0.811	0.973	0.992	0.922	0.986
	32	0.905	0.948	0.992	0.958	0.966
GEMMA_V3_12B_PT	2	0.945	0.896	0.858	0.808	0.963
	5	0.886	0.965	0.872	0.959	0.905
	8	0.975	0.894	0.866	0.889	0.979
	11	0.958	0.940	0.980	0.998	0.986
	14	0.980	0.960	0.980	0.988	0.998
	17	0.979	0.986	0.956	0.946	0.998
	20	0.977	0.990	0.968	0.942	0.998
	23	0.963	0.998	0.994	0.933	0.984
	26	0.907	1.000	0.998	0.848	0.992
	29	0.889	0.994	0.982	0.859	1.000
	32	0.924	0.962	0.984	0.742	1.000
	35	0.845	0.956	0.952	0.869	0.994
	38	0.773	0.942	0.980	0.808	0.986
	41	0.756	0.869	0.920	0.772	0.990
	44	0.817	0.928	0.984	0.953	0.990
	47	0.956	0.971	0.970	0.928	0.968
GEMMA_V3_27B_PT	5	0.929	0.876	0.829	0.879	0.905
	11	0.940	0.994	0.865	0.946	0.972
	17	0.979	0.990	0.900	0.996	0.996
	23	0.973	1.000	0.945	0.960	0.996
	29	0.950	0.998	0.994	0.937	0.994
	35	0.864	0.988	0.924	0.906	0.994
	41	0.936	0.964	0.960	0.827	0.990
	47	0.890	0.996	0.912	0.858	0.990
	53	0.887	0.980	0.912	0.787	0.968
	59	0.863	0.984	0.994	0.977	0.958