

Figure 2: Example of false belief from our probing datasets. The labels z_p and z_o correspond to \mathcal{D}_p^P and \mathcal{D}_o^P , respectively. By manipulating the protagonist’s **percepts** after the causal event, we obtain two scenarios: **true belief** and **false belief**.

et al., 2020) ensures that there is no data leakage¹. Additionally, we consider a SFT version of Pythia-6.9B trained on open-source instruction datasets (Wang et al., 2024), which we refer to as Pythia-6.9B-chat.² We provide model details in Table 2.

3.4 Probing Experiments

To study how LMs represent beliefs of self and others, we propose a set of extensive probing experiments across LMs that differ in architecture, size, and fine-tuning regime. We train probes on the residual stream, as it integrates information from both the attention and feed-forward components, potentially encoding richer representations. Additionally, since the residual activations directly contribute to the final output predictions, probing them may better align with understanding the model’s behaviour for downstream tasks.

Control Tasks Depending on the model dimension, the probes we train have a significant number of learnable parameters – up to 16,385 for Llama-2-70B. This raises the concern that probes might learn to rely on irrelevant patterns in the data instead of capturing meaningful relationships. To account for the potential confounding effect of hidden state size, we include two controls. First, following Hewitt and Liang (2019), we train and evaluate probes on a version of \mathcal{D}_p^P with randomly

¹Llama-2 was released later than BigToM.

²<https://huggingface.co/allenai/open-instruct-pythia-6.9b-tulu>

permuted labels – thus removing real input-label relationships. If a probe still performs well on the permuted data, this suggests it may be exploiting superficial correlations rather than capturing genuine structure. Second, we effectively reduce the number of learnable parameters in the probes by projecting \mathcal{D}_p^P and \mathcal{D}_o^P onto their k largest principal components using PCA before training. This minimises the risk of the probes relying on spurious patterns in the data.

Robustness Tests Previous work left the impact of prompting on belief probing accuracy unexplored. Our second set of experiments aims to study whether belief representations are robust to different prompts. Research on prompt robustness in language models focused mainly on revealing vulnerability to prompt alterations on *downstream performance* (Min et al., 2022; Ishibashi et al., 2023; Shaikh et al., 2023; Leidinger et al., 2023; Sclar et al., 2024). In contrast, we study how different prompt alterations influence *probing performance*, i.e. models’ internal representations. Unlike model outputs that are shaped by decoding strategies, which act as confounders, models’ activations are more abstract and offer a better lens into how robust or brittle internal representations are. We define four prompt variations:

- *Random*: Following Gurnee and Tegmark (2024), we add 10 random tokens to the belief statement.
- *Misleading*: Each story is followed by two belief statements, one pertinent to the story and one randomly chosen from another.
- *Time Specification*: The prompt specifies that the belief statement refers to the end of the story. We include this variation because some belief statements can be true (false) at the story’s beginning but false (true) at the end. For example, consider the story in Figure 2: if Noor does not witness the swap, in the end, she will believe the pitcher contains almond milk ($z_p = \text{True}$). However, if the same belief is referred to the beginning of the story, then it is false ($z_p = \text{False}$).
- *Initial Belief*: We explicitly reveal the protagonist’s initial belief (e.g. “*Noor believes that the pitcher contains oat milk*”) in the story to test whether it biases the representations of LMs.

While all maintain conceptual and semantic parity with the *Original* prompt used in (Zhu et al., 2024), *Random* and *Misleading* are expected to negatively impact LMs’ representations, while *Time Specifi-*

cation and *Initial Belief* are supposed to have a positive influence. Robust representations of beliefs should exhibit minimal sensitivity to these alterations. Our experiments compare probe accuracy across different model sizes, fine-tuning, and prompt variations. Examples of prompts are reported in Appendix A.1.4.

3.5 Activation Editing

Prior work found that it is possible to manipulate models’ representations of beliefs by using (Li et al., 2023b, ITI), and that such interventions can improve LMs’ performance on ToM tasks. We take this further by asking whether a general “belief vector” can be distilled and *injected* into the models’ activations to *strengthen* their ToM abilities. To this end, we use contrastive activation addition (Rimsky et al., 2023, CAA), an extension of activation addition (Turner et al., 2023, AA) that computes *steering vectors* to control LMs’ behaviour. Steering vectors are computed as the average difference in residual stream activations between pairs of positive and negative instances of a specific behaviour. Formally, given a dataset \mathcal{D} of triplets (p, c_p, c_n) , where p is a prompt, c_p is a positive completion, and c_n is a negative completion, CAA computes a *mean difference* vector v_l^{md} for layer l as:

$$v_l^{md} = \frac{1}{|\mathcal{D}|} \sum_{p, c_p, c_n \in \mathcal{D}} a_l(p, c_p) - a_l(p, c_n) \quad (1)$$

For example, in Figure 2, p is the *Story*, c_p could be the true belief, and c_n the false belief. During inference, these steering vectors are multiplied by an appropriate coefficient α and added at every token position of the generated text after the prompt. CAA has two main advantages over ITI: First, it eliminates the need to train probes, making it *computationally cheap*. For example, for Llama2 70B, ITI needs to train 5,120 probes while CAA only needs to compute 80 vectors. Second, it operates at the residual stream level, making it easier to use than methods that intervene on specific attention heads like ITI. While CAA has been used to control alignment-relevant behaviour, such as hallucinations, refusal, and sycophancy (Rimsky et al., 2023), we are the first to apply it to enhance LMs’ ToM reasoning. The “belief vectors” (i.e. steering vectors) we obtain can be understood as isolating the direction in the LMs’ latent space corresponding to taking the perspective of another agent. To evaluate both base and fine-tuned LMs, we rank

their answers to the ToM questions according to $p_{LM}(a|q)$ (Petroni et al., 2019). For a fair comparison, we adopt the train/test *Forward Belief* split used in (Zhu et al., 2024) to compute and evaluate the steering vectors. Additionally, we evaluate the transferability of the CAA steering vectors by applying them to two other BigToM tasks: *Forward Action* and *Backward Belief*. We provide details about these tasks in Appendix A.1.1, and a more detailed explanation ITI in Appendix A.5.

4 Results

Effect of Model Size and Fine-tuning Results from our study on model size and fine-tuning are shown in Figure 3. For *oracle* beliefs, probing accuracy rapidly converges to 100, with larger models showing faster convergence. Even the smallest Pythia-70m achieves 95% accuracy. For *protagonist* beliefs, we notice a similar pattern across most models, where accuracy at early layers is particularly low and then increases at the intermediate layers. What happens at early layers is overfitting, which may be caused by spurious features introduced by the initial coding strategy of language models, where individual token representations are mixed together (Gurnee et al., 2023). We further discuss this in Appendix A.2.1.

In general, probing accuracy increases with model size, although there is a performance gap between Llama-2 and Pythia. For example, Llama2-13B reaches around 80% accuracy, while Pythia-12B achieves approximately 60%. This gap is likely due to Llama-2 being trained on nearly seven times more tokens than Pythia (cf. Table 2). Probes from fine-tuned LMs show significantly better accuracy, with improvements of up to +29% for Llama2-7B-chat (SFT + RLHF) and +26% for Pythia-6.9B-chat (SFT) compared to probes from their base version. The same probes outperform (Llama-2) or are on par (Pythia) with probes trained on twice as large base models (12/13B). This highlights a key role of fine-tuning in shaping belief representations in smaller LMs. The performance gap closes for the largest Llama2-70B, for which the improvements from fine-tuning are marginal.

We characterise the relationship between probe accuracy and model size in Figure 7, using the *best* accuracy for each LM – i.e., the highest accuracy among probes g_l trained on activations a_l for model f . For Llama-2 base and Pythia base, probing accuracy scales logarithmically with model size (Fig-

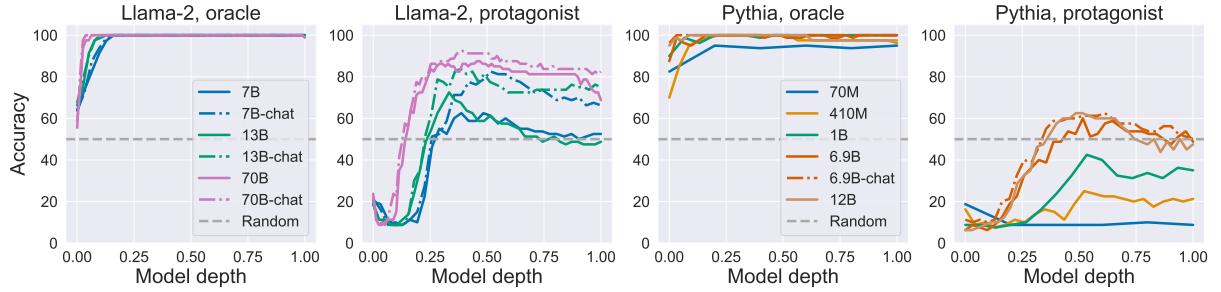


Figure 3: Belief probing accuracy show similar patterns across all models: *oracle* belief representations generally form already in the first layers, while *protagonist* belief representations emerge at the intermediate layers. Moreover, probing accuracy increases with model size and, more crucially for smaller models, with fine-tuning.

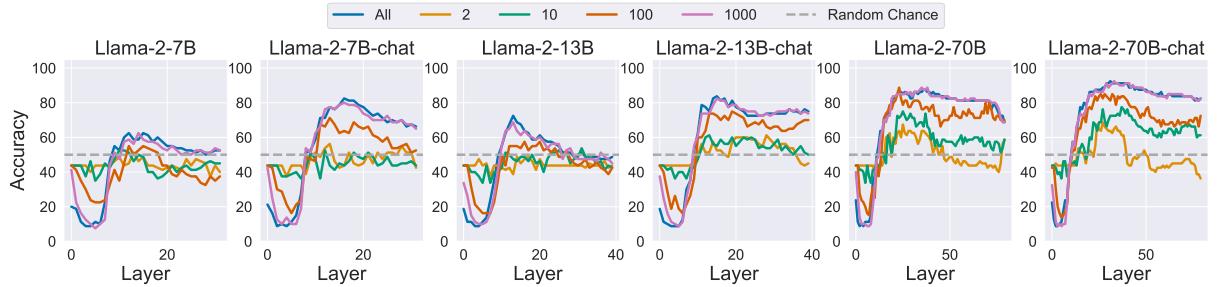


Figure 4: We compare the probing accuracy obtained by using the original set of activations (All) with the accuracy obtained by considering only the first $k = \{2, 10, 100, 1000\}$ principal components. Results are for *protagonist* beliefs (for *oracle* see Figure 12). In general, it is possible to recover most of the original accuracy by training probes on a smaller number k of principal components of the activations.

ure 7b, 7d), while for fine-tuned Llama-2 models, it scales linearly (Figure 7c).

Control Tasks Figure 8a shows that probes trained on the control task consistently perform at random chance, confirming that higher probing accuracy in larger models meaningfully reflects a greater ability to extract ToM representations, rather than simply being a by-product of spurious correlations. For Llama models, the probes generally exhibit selectivity: they achieve high accuracy when probing for beliefs but remain at chance level on control tasks. Pythia’s overall accuracy is too low to allow for selectivity.

Figure 4 shows probing accuracy on *protagonist* when training the probes on the top k principal components of Llama-2’s internal activations. We provide results for Pythia in Figure 11, and for all models on *oracle* settings in Figure 12. We consider $k = \{2, 10, 100, 1000\}$, spanning several orders of magnitude.³ Results show that it is generally possible to recover most of the original accuracy by training probes on a smaller number k of principal components of the activations. We

also performed the first control experiment, this time only using the first $k = \{100, 1000\}$ principal components. Figure 8b and 8c again show that probes trained on the control task consistently perform at random chance, confirming that probes are not fitting spurious patterns. Additionally, this suggests that belief representations are embedded in a low-dimensional subspace \mathcal{B} spanned by the top k eigenvectors $\{v_1, \dots, v_k\}$ of the covariance matrix $C = \mathbb{E}[(a - \mathbb{E}[a])(a - \mathbb{E}[a])^\top]$.

Sensitivity to Prompting Figure 5 compares *protagonist* probe accuracy across various prompt variations for Llama-2 models. As can be seen from the figure, providing the protagonist’s *Initial Belief* in the story yields higher probe accuracy compared to the *Original* prompt. Accuracy for all the other prompt variations is generally lower than *Original*. *Misleading* prompts hurt performance across all models. This finding resonates with [Webson and Pavlick \(2022\)](#), who found that instruction-tuned models, despite being more robust, are still sensitive to misleading prompts. On the other hand, *Time Specification* unexpectedly does not help in disambiguating belief states in different time frames, as we hypothesised in §3.4. Additionally, models

³For models with hidden dimensions smaller than 1000, we skip this value.