Entropy-Guided LLM Decoding via Probabilistic Circuits

Zhizhen Chen[1] **Daniel Israel**[2] **Guy Van den Broeck**[2] **Zhe Zeng**[1]

[1]University of Virginia
[2]University of California, Los Angeles

Abstract

Existing decoding algorithms for large language models (LLMs) always rely on the information in the current step. We argue that future entropy is also essential for LLM decoding. However, this quantity is intractable for autoregressive LLMs, since it requires marginalizing over exponentially many continuations. We estimate it by computing the lower and upper bounds of it on a hidden Markov model (HMM), which is distilled as a surrogate of the LLM. By representing the HMM as probabilistic circuits, we make its latent-variable structure explicit and use it to derive an upper bound and a novel lower bound on the circuit entropy. On the HMM, these bounds yield efficient estimates of future entropy, which we leverage to implement entropy-guided decoding. On GSM8K and CSQA, this guidance improves math and logical reasoning over the decoding baselines in the Qwen2.5 and Mistral models, showing that estimated future entropy is effective for decoding.

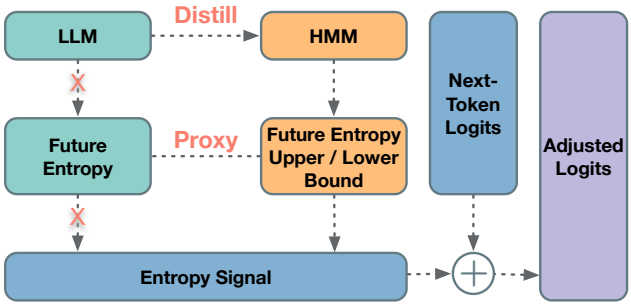


Figure 1: The pipeline of our proposed entropy-guided decoding method. Since the future entropy of an LLM is intractable, we establish a tractable surrogate for it, and leverage the surrogate to guide the LLM decoding.

1 INTRODUCTION

Recent works such as EDT [Zhang et al., 2024b] and Top-H [Baghaei Potraghloo et al., 2026] use the entropy of the model to dynamically adjust sampling, achieving a better performance than the conventional decoding algorithms (e.g., Top-K [Fan et al., 2018] sampling). However, both methods condition only on the current-step entropy, which is *myopic*. We argue that future entropy is a more important signal for guiding generation. Intuitively, for creative writing that requires more diverse generations, we prefer that the generation of the model has higher future entropy; for mathematical reasoning that requires a deterministic answer, a lower future entropy would be helpful to this objective.

However, future entropy is intractable to compute for existing LLM architectures. In autoregressive models, it requires marginalizing over an exponentially large space of future continuations. To establish an efficient estimation of this intractable signal, we turn to probabilistic circuits (PCs) [Vergari et al., 2021]. PCs are a family of generative models whose computation graphs are structured to guarantee efficient, exact inference for a rich class of queries that are otherwise intractable in unconstrained neural models [Liu et al., 2024]. A hidden Markov model (HMM), which can be represented as a PC, is particularly well-suited to sequential data: its latent Markov structure enables efficient forward and backward inference over all possible future continuations via dynamic programming, without explicitly enumerating them. Building on Ctrl-G [Zhang et al., 2024a], which distills an HMM as a tractable surrogate of an LLM by training it to approximate the next token of the LLM distribution, we use the future entropy of an HMM as a tractable proxy for that of the underlying LLM.

Computing the entropy of a PC is still a coNP-hard problem [Vergari et al., 2021]. Our key observation is that the latent-variable structure of a PC makes it possible to bound this entropy tractably. Concretely, our contributions are twofold. First, by making the latent-variable view of a PC

explicit, we derive an **upper bound** and a **lower bound** on its entropy, both of which are computable in polynomial time. We estimate the upper and lower bounds of HMMs by this method. Second, we wrap these estimates into the entropy-guided decoding that steers the decoding by incorporating the entropy signal into the next-token logits. The experimental results confirm that the entropy signal helps the LLM improve problem solving in GSM8K and CSQA datasets Cobbe et al. [2021], Talmor et al. [2019].

2 PRELIMINARY

In this section, we state the concepts of probabilistic circuits and hidden Markov models. We also introduce the notations we used in this paper.

2.1 PROBABILISTIC CIRCUITS

Probabilistic circuits [Choi et al., 2020] can be used to describe a large family of tractable probabilistic models (TPMs), including hidden Markov models (HMMs), Chow-Liu trees, sum-product networks, and others. Specifically, PCs are actually a type of computation graph defined by sum and product operations:

Definition 2.1 (Probabilistic circuit)**.** A *probabilistic circuit* $P(\mathbf{X})$ over variables $\mathbf{X}$ is a rooted directed acyclic computation graph with a single root node n_r; we define $\mathrm{ch}(n)$ as the set of children of node n. Each *input* node (a leaf without children) encodes a univariate distribution over a subset of random variables $\mathbf{X}_n \subseteq \mathbf{X}$, called its *scope*. Each inner node is either a *product* node, encoding a factorized distribution over its children, or a *sum* node, encoding a weighted mixture of its children's distributions. The scope of an inner node equals the union of its children's scopes: $\mathbf{X}_n = \bigcup_{c\in\mathrm{ch}(n)} \mathbf{X}_c$. Formally, the distribution at each node n is defined recursively by

$$P_n(\mathbf{X}_n) := \begin{cases} f_n(\mathbf{X}_n) & n \text{ is an input node,} \\ \sum_{c\in\mathrm{ch}(n)} \theta_{n,c} \cdot P_c(\mathbf{X}_c) & n \text{ is a sum node,} \\ \prod_{c\in\mathrm{ch}(n)} P_c(\mathbf{X}_c) & n \text{ is a product node,} \end{cases}$$

where $\theta_{n,c} \geq 0$ is the weight on edge (n, c). Without loss of generality, we assume every root-to-input path alternates between sum and product nodes (since consecutive nodes of the same type can be collapsed), all weights are positive, and the PC is locally normalized (i.e., for every sum node n, $\sum_{c\in\mathrm{ch}(n)} \theta_{n,c} = 1$; for every input node n, $\sum_{x_n\in\mathbf{X}_n} f_n(x_n) = 1$). We also assume that the parent node of any input node is a sum node, since when we need to set an input node's parent to a product node (e.g., implement a Gaussian mixture model), we can add a dummy sum node as that node's parent.

All the PCs we discuss in this paper satisfy *smoothness* and *decomposability*. These two properties enable them to support linear-time marginalization [Vergari et al., 2021].

Definition 2.2 (Smoothness and Decomposability)**.** A PC is smooth if for every sum node n all children share the same scope, i.e. $\mathbf{X}_{c_1} = \mathbf{X}_{c_2}$ for all $c_1, c_2 \in \mathrm{ch}(n)$; it is decomposable if for every product node n the children have pairwise disjoint scopes, i.e. $\mathbf{X}_{c_1} \cap \mathbf{X}_{c_2} = \emptyset$ for all distinct $c_1, c_2 \in \mathrm{ch}(n)$.

The top-down probabilities quantify how much the node contributes to the circuit's output.

Definition 2.3 (Top-down probabilities)**.** Given a PC p, the *top-down probability* $\mathrm{TD}(n)$ of a node n is defined recursively from the root toward the inputs,

$$\mathrm{TD}(n) := \begin{cases} 1 & n \text{ is the root node,} \\ \sum_{m\in\mathrm{pa}(n)} \mathrm{TD}(m) & n \text{ is a sum node,} \\ \sum_{m\in\mathrm{pa}(n)} \theta_{m,n} \cdot \mathrm{TD}(m) & n \text{ is a product / input node,} \end{cases}$$

where $\mathrm{pa}(n)$ denotes the set of parents of n. The top-down probability of an edge (n, c) is $\mathrm{TD}(\theta_{n,c}) = \theta_{n,c} \cdot \mathrm{TD}(n)$.

2.2 HIDDEN MARKOV MODELS

A hidden Markov model is a latent variable model for sequences, where the observed sequence $\mathbf{X} = \mathbf{X}_1\mathbf{X}_2 \cdots \mathbf{X}_T$ depends on the hidden state sequence $\mathbf{Y} = \mathbf{Y}_1\mathbf{Y}_2 \cdots \mathbf{Y}_T$, and $\mathbf{Y}$ satisfies the Markov property. For language modeling, x_t is the token at position t and y_t is the corresponding hidden state. x_t takes value in $\{1, 2, \cdots, V\}$, where V is the vocabulary size; y_t takes value in $\{1, 2, \cdots, C\}$, where C is the number of hidden states.

The computational graph of an HMM is a dynamically expanding PC, as we show in Figure 3. Once we have established the upper and lower bounds for PCs, we can leverage them to estimate the future entropy for HMMs.

3 FROM CIRCUIT PARAMETERS TO AN EXPLICIT LATENT-VARIABLE MODEL

We first make the explicit latent-variable view of a PC, following the construction in Liu et al. [2025] (see their Appendix B.1) and Peharz et al. [2016].

Definition 3.1 (Latent augmentation)**.** Let $P(\mathbf{X})$ be a smooth, decomposable, and locally normalized PC with root n_r and sum-node set $\mathcal{S}$. With every sum node $n \in \mathcal{S}$, once n is reached, it selects exactly one of its children $c \in \mathrm{ch}(n)$, an event we record as the tuple (n, c). For a node n, let $\mathcal{S}_n \subseteq \mathcal{S}$ collect the sum nodes in the sub-circuit rooted at

n, and let $\mathbf{Z}_n = \{(m, c_m)\}_{m \in \mathcal{S}_n}$, where $c_m \in \mathrm{ch}(m)$ denotes the child selected at m, be their joint latent. We define the set of reached nodes in $\mathbf{Z}$ as $\mathcal{N}(z) = \cup_{(u,v)\in z}\{u, v\}$. Because a sum node descends only into its selected child, the reached sum nodes are determined by the assignment itself; in the recursions below, $\mathbf{Z}_c = \{(m, c_m)\}_{m \in \mathcal{S}_c}$ is simply the subset of $\mathbf{Z}_n$ indexed by the sum nodes of child c. We define the support $\mathsf{supp}(n)$ of $\mathbf{Z}_n$, and on the sum and product nodes the augmented distribution $P_n(\mathbf{X}_n, \mathbf{Z}_n)$, by structural recursion:

$$\mathsf{supp}(n) := \begin{cases} \{\{\}\} & n \text{ is an input node,} \\ \bigcup_{c\in\mathrm{ch}(n)}\{(n, c)\} \times \mathsf{supp}(c) & n \text{ is a sum node,} \\ \prod_{c\in\mathrm{ch}(n)} \mathsf{supp}(c) & n \text{ is a product node,} \end{cases}$$

$$P_n(\mathbf{X}_n, \mathbf{Z}_n) := \begin{cases} P_n(\mathbf{X}_n) & n \text{ is an input node,} \\ \sum_{c\in\mathrm{ch}(n)} \theta_{n,c} \cdot P_c(\mathbf{X}_c, \mathbf{Z}_c) & \\ \qquad \cdot \mathbb{1}[(n, c) \in \mathbf{Z}_n] & n \text{ is a sum node,} \\ \prod_{c\in\mathrm{ch}(n)} P_c(\mathbf{X}_c, \mathbf{Z}_c) & n \text{ is a product node.} \end{cases}$$

Taking $n = n_r$ gives the global latent $\mathbf{Z} = \mathbf{Z}_{n_r}$ with support $\mathsf{supp}(n_r)$, and we call $P(\mathbf{X}, \mathbf{Z}) = P_{n_r}(\mathbf{X}, \mathbf{Z})$ the *latent augmentation* of P.

4 APPROXIMATE THE ENTROPY OF HIDDEN MARKOV MODELS

Computing the entropy $H(\mathbf{X})$ of an HMM $P_{\mathrm{HMM}}(\mathbf{X})$ is a coNP-hard problem [Vergari et al., 2021]. However, leveraging the latent-variable view of PCs, we can obtain some tractable entropies in PC for latent variables, and exploit them to obtain the upper and lower bounds of $H(\mathbf{X})$. The proofs can be found in Appendix C.

4.1 THE UPPER BOUND OF ENTROPY

The entropy $H(\mathbf{X})$ can be decomposed into $H(\mathbf{X}, \mathbf{Z}) - H(\mathbf{Z}|\mathbf{X})$. By dropping $H(\mathbf{Z}|\mathbf{X})$, we can obtain the upper bound of $H(\mathbf{X})$.

Corollary 4.1 (Upper bound)**.** *For a smooth, decomposable, locally normalized PC P, we have*

$$H(\mathbf{X}) \le H(\mathbf{X}, \mathbf{Z}) = H(\mathbf{X}|\mathbf{Z}) + H(\mathbf{Z}). \tag{1}$$

The right-hand side is given by Theorems 4.1 and 4.2, which can be computed with the time complexity $O(|P|)$.

We then demonstrate that $H(\mathbf{X}|\mathbf{Z})$ and $H(\mathbf{Z})$ can be computed by the top-down probability.

4.2 TWO TRACTABLE ENTROPIES

Theorem 4.1 (Latent entropy)**.** *For a smooth, decomposable, locally normalized PC,*

$$H(\mathbf{Z}) = -\sum_{n\in\mathcal{S}} \sum_{c\in\mathrm{ch}(n)} \mathrm{TD}(\theta_{n,c}) \cdot \log \theta_{n,c}. \tag{2}$$

Theorem 4.2 (Conditional entropy)**.** *For a smooth, decomposable, and locally normalized PC,*

$$H(\mathbf{X}|\mathbf{Z}) = \sum_{n\in\mathcal{I}} \mathrm{TD}(n) \cdot H(f_n(\mathbf{X}_n)), \tag{3}$$

where $\mathcal{I}$ is the set of all input nodes, and $H(f_n(\mathbf{X}_n))$ is the entropy of the univariate input distribution $f_n(\mathbf{X}_n)$.

Remark. The time complexity for computing the above two entropies is $O(|P|)$, where $|P|$ is the size of the circuit. The time complexity of computing all TD is $O(|P|)$, because each $\theta_{n,c}$ will only be accessed once. The computation of latent entropy will access each $\theta_{n,c}$ once; therefore, the final time complexity is $O(|P|)$. The computation of the conditional entropy only accesses each input node, and the number of input nodes is smaller than $|P|$. Therefore, the computation in PC will cost $O(|P|)$ time.

4.3 THE LOWER BOUND OF ENTROPY

The term $H(\mathbf{Z}|\mathbf{X})$ is intractable for PCs, because the range of values for $\mathbf{X}$ is exponential. However, if for each sum node n, we let the condition $\mathbf{X}$ only include $\tilde{X}_n$, the union of the scope of input nodes that can be reached directly from n without passing through other non-dummy sum nodes (we formally define it in Definition 4.1), then in the usual case (such as HMM, HCLT [Liu and Van den Broeck, 2021]) where the size of the support of $\tilde{X}_n$ is polynomial, we can tractably obtain the upper bound of $H(\mathbf{Z}|\mathbf{X})$ (because the number of conditions is reduced).

Definition 4.1 (Local selection and local scope)**.** For a sum node n, its *local selection* is

$$\mathbf{Z}^{(n)} := \begin{cases} c & (n, c) \in \mathbf{Z}, \\ \varnothing & n \notin \mathcal{N}(\mathbf{Z}), \end{cases}$$

and $\mathbf{Z}$ together with $(\mathbf{Z}^{(n)})_{n\in\mathcal{S}}$ determine each other. Let $\tilde{\mathcal{I}}_n \subseteq \mathcal{I}$ collect the input nodes reachable from n without crossing another non-dummy sum node (equivalently, those whose nearest non-dummy sum-node ancestor is n), and let $\tilde{\mathbf{X}}_n := \bigcup_{m\in\tilde{\mathcal{I}}_n} \mathbf{X}_m$ be the *local scope* of n. The *local selection entropy* of n is

$$\tilde{H}_n := H\big(\mathbf{Z}^{(n)}|\tilde{\mathbf{X}}_n,\ n \in \mathcal{N}(\mathbf{Z})\big),$$

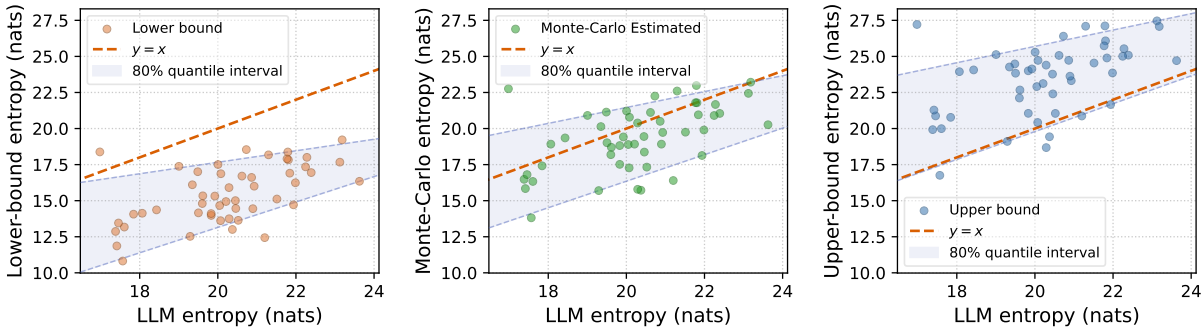
i.e. the entropy of $\mathbf{Z}^{(n)}$ given the variables it directly emits, conditioned on n being reached.

Proposition 4.1 (Lower bound)**.** *For a smooth, decomposable, locally normalized PC P,*

$$H(\mathbf{X}) \ \ge\ H(\mathbf{X}|\mathbf{Z}) + H(\mathbf{Z}) - \tilde{H}(\mathbf{Z}|\mathbf{X}),$$

where $\tilde{H}(\mathbf{Z}|\mathbf{X}) := \sum_{n\in\mathcal{S}} \mathrm{TD}(n) \cdot \tilde{H}_n$.

Each term on the right is given by Theorems 4.1, 4.2, and Definition 4.1, which can be computed within time complexity $O(C \cdot V \cdot |P|)$. Here, $C = \max(|\mathrm{ch}(n)| : \forall n \in \mathcal{S})$, and $V = \max(|\mathsf{supp}(\tilde{\mathbf{X}}_n)| : \forall n \in \mathcal{S})$.



(a) Lower bound (b) Monte-Carlo Estimation (c) Upper bound

Figure 2: The relationship between HMM's entropy and LLM's entropy. For a given sequence, we selected 50 distinct tokens to append to the end of the sentence, and these 50 tokens have the corresponding top 50 logits in the LLM distribution. We then used both LLM and HMM to estimate the future 8-step entropy corresponding to different next token selections.

5 ENTROPY-GUIDED DECODING

In this section, we leverage the entropy bounds to estimate the future entropy of the HMM, and use this to guide the decoding process of the LLM.

We first demonstrate how to estimate the future entropy of the HMM. First, we compute the current $P(\mathbf{Y}_t|\mathbf{X}_{<t})$ by the forward algorithm. Then, we replace the edge values of the root of the HMM shown in Figure 3 with $P(\mathbf{Y}_t|\mathbf{X}_{<t})$. To predict the future k-step entropy, we extend the PC to cover $X_{t:t+k-1}$. By applying Corollary 4.1 and Proposition 4.1, we can obtain the upper and lower bounds for future entropy of the HMM. The time complexity for the upper bound is $O(V \cdot |P|)$, since we need $O(V)$ time to compute $H(f_n(\mathbf{X}_n))$. The time complexity for the lower bound is $O(C \cdot V \cdot |P|)$, where C and V exactly match the definition of the HMM in Section 2.2.

In Figure 2, we demonstrate that the entropy of the HMM is positively correlated with the entropy of the LLM in the decoding process. Therefore, it can be regarded as a signal to reflect the entropy of the LLM. To guide the generation, we need to incorporate the entropy signal into the logits score, where $\texttt{score} = \texttt{llm\_score} + \texttt{mix\_ratio} \cdot \texttt{H}(\mathbf{X}_{t:t+k-1})$, where `llm_score` is the original logits score. A similar incorporation can be found in Ahmed and Singh [2026].

We evaluate the entropy-guided decoding performance on GSM8K and CSQA datasets [Cobbe et al., 2021, Talmor et al., 2019]. The `Empirical` is the average of the upper and lower bounds, which can be regarded as the approximation to the real entropy. The HMM is distilled by sampling the LLM's distribution in the training split, using the pipeline proposed in Ctrl-G [Zhang et al., 2024a]. To reduce computation, we first use Top-K filtering to select 50 next tokens, and then predict the future entropies corresponding to these tokens. The baseline we compared includes Top-H [Baghaei Potraghloo et al., 2026], P-Less [Tan et al., 2026], Min-P [Nguyen et al., 2025], Top-K [Fan et al., 2018], and Top-P [Holtzman et al., 2019]. We reported the hyperparameters of these methods in Appendix D. Table 1 demonstrates the experimental results, where our proposed method consistently achieves higher accuracy than Top-K. In the experiment on the Qwen2.5-7B and Mistral-7B models, our method achieves the best accuracy in both GSM8K and CSQA tasks.

Table 1: Accuracy (%) of Llama3.1-8B, Qwen2.5-7B, and Mistral-7B in the math and logical reasoning tasks with different decoding strategies. The configurations of different strategies are reported in Appendix D.

	Llama3.1-8B		Qwen2.5-7B		Mistral-7B-v0.3	
	GSM8K	CSQA	GSM8K	CSQA	GSM8K	CSQA
None	85.6	75.4	88.9	82.2	66.0	70.1
Top-K	85.1	75.4	88.5	81.6	67.2	71.1
Top-P	86.7	77.0	89.3	81.7	66.7	70.1
Min-P	**87.8**	75.9	89.2	80.7	68.6	70.4
P-Less	87.6	**77.6**	89.5	81.4	**70.6**	70.4
Top-H	87.7	77.3	89.5	81.5	69.7	70.4
Upper	85.4	76.3	**89.7**	79.8	**70.6**	70.1
Lower	85.4	74.6	88.5	81.1	70.0	70.6
Empirical	85.6	75.3	89.3	**82.3**	68.5	**71.4**

6 CONCLUSION

We introduced an efficient approach to estimating the future entropy of an LLM, a signal that is intractable to compute directly yet valuable for steering generation. By making the latent-variable view of a probabilistic circuit explicit, we derived an upper bound and a lower bound on its entropy, and applied them on an HMM surrogate to obtain future-entropy estimates. Wrapping these estimates into the entropy-guided decoding improved math and logical reasoning accuracy on GSM8K and CSQA, demonstrating that prospective, future-aware guidance is useful in practice.

Acknowledgements

We would like to thank Zilei Shao for helpful discussions. The authors acknowledge Research Computing at The University of Virginia for providing computational resources.

References

Kareem Ahmed and Sameer Singh. Entropy-aligned decoding of lms for better writing and reasoning. *arXiv preprint arXiv:2601.01714*, 2026.

Erfan Baghaei Potraghloo, Seyedarmin Azizi, Souvik Kundu, and Massoud Pedram. Top-h decoding: Adapting the creativity and coherence with bounded entropy in text generation. *Advances in Neural Information Processing Systems*, 38:28482–28513, 2026.

Y Choi, Antonio Vergari, and Guy Van den Broeck. Probabilistic circuits: A unifying framework for tractable probabilistic models. *UCLA. URL: http://starai. cs. ucla. edu/papers/ProbCirc20. pdf*, 6, 2020.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Angela Fan, Mike Lewis, and Yann Dauphin. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 889–898, 2018.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv:1904.09751*, 2019.

Anji Liu and Guy Van den Broeck. Tractable regularization of probabilistic circuits. *Advances in Neural Information Processing Systems*, 34:3558–3570, 2021.

Anji Liu, Mathias Niepert, and Guy Van den Broeck. Image inpainting via tractable steering of diffusion models. In *International Conference on Learning Representations*, volume 2024, pages 968–989, 2024.

Anji Liu, Zilei Shao, and Guy Van den Broeck. Rethinking probabilistic circuit parameter learning. *arXiv preprint arXiv:2505.19982*, 2025.

Minh Nguyen, Andrew Baker, Clement Neo, Allen Roush, Andreas Kirsch, and Ravid Shwartz-Ziv. Turning up the heat: Min-p sampling for creative and coherent llm outputs. In *International Conference on Learning Representations*, volume 2025, pages 70333–70366, 2025.

Robert Peharz, Robert Gens, Franz Pernkopf, and Pedro Domingos. On the latent variable interpretation in sum-product networks. *IEEE transactions on pattern analysis and machine intelligence*, 39(10):2030–2044, 2016.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1421. URL https://aclanthology.org/N19-1421.

Runyan Tan, Shuang Wu, and Phillip Howard. P-less sampling: A robust hyperparameter-free approach for llm decoding. In *International Conference on Learning Representations*, volume 2026, pages 119506–119549, 2026.

Antonio Vergari, YooJung Choi, Anji Liu, Stefano Teso, and Guy Van den Broeck. A compositional atlas of tractable circuit operations for probabilistic inference. *Advances in Neural Information Processing Systems*, 34:13189–13201, 2021.

Honghua Zhang, Po-Nien Kung, Masahiro Yoshida, Guy Van den Broeck, and Nanyun Peng. Adaptable logical control for large language models. *Advances in Neural Information Processing Systems*, 37:115563–115587, 2024a.

Shimao Zhang, Yu Bao, and Shujian Huang. Edt: Improving large language models' generation by entropy-based dynamic temperature sampling. *arXiv preprint arXiv:2403.14541*, 2024b.

Entropy-Guided LLM Decoding via Probabilistic Circuits (Supplementary Material)

Zhizhen Chen[1] **Daniel Israel**[2] **Guy Van den Broeck**[2] **Zhe Zeng**[1]

[1]University of Virginia
[2]University of California, Los Angeles

A THE PC REPRESENTATION OF HMM

A hidden Markov model has three parameters: initial probability γ, where $p(y_1) = \gamma_{y_1}$; transition matrix α, where $p(y_t|y_{t-1}) = \alpha_{y_{t-1},y_t}$ holds for every t; and emission matrix β, where $p(x_t|y_t) = \beta_{y_t,x_t}$ holds for every t. We demonstrate the PC representation of an HMM in Figure 3, where the parameters of PC are derived from the parameters of HMM.

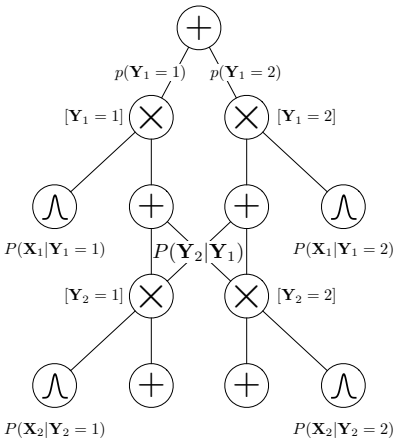


Figure 3: The dynamic extending PC to a HMM, ignoring the dummy sum nodes. As the sequence length increases, the PC will grow using the parameters of the HMM. Each sum node denotes the initial probability or a transition process; each product symbolizes an event that a hidden variable $\mathbf{Y}$ takes a certain value; each input node represents a conditional distribution encoded by the emission matrix β.

B AUXILIARY RESULTS FOR THE LATENT-VARIABLE VIEW

The entropy $H(\mathbf{X})$ of a PC is coNP-hard to compute directly, yet the latent-variable view of Definition 3.1 breaks it into pieces that *are* tractable. Turning that idea into formulas, however, requires knowing the augmented model explicitly: what the joint $P(\mathbf{X}, \mathbf{Z})$ looks like, what the law $P(\mathbf{Z})$ of the latent code is on its own, how $P(\mathbf{X} \mid \mathbf{Z})$ factorizes once the code is fixed, and how often each node and edge is visited. This appendix develops exactly these four facts, in the order in which they depend on one another, and the next appendix feeds them into the entropy expressions and bounds of the main text. We begin with the proof of Proposition B.1, whose statement (given in the main text) guarantees that the augmentation is exact: marginalizing $\mathbf{Z}$ recovers $P(\mathbf{X})$, so every subsequent statement about $P(\mathbf{X}, \mathbf{Z})$ is also a statement about the circuit we care about.

B.1 EXACT LATENT AUGMENTATION)

We demonstrate that the latent augmentation we proposed in Definition 3.1 is an exact latent augmentation to the original PC.

Proposition B.1 (Exact Latent augmentation)**.** *For a smooth, decomposable, locally normalized PC, the latent augmentation of Definition 3.1 marginalizes to the circuit distribution: for every x,*

$$\sum_{z \in \mathsf{supp}(n_r)} P_{n_r}(\mathbf{X}, z) = P_{n_r}(\mathbf{X}) = P(\mathbf{X}). \tag{4}$$

Consequently $P(\mathbf{X}, \mathbf{Z})$ is a normalized distribution such that $\sum_{z \in \mathsf{supp}(n_r)} \sum_x p(x, z) = 1$, and $\mathbf{Z}$ is an exact latent augmentation of $P(\mathbf{X})$.

Proof. For every node n and $z_n \in \mathsf{supp}(n)$, we let

$$P_n(\mathbf{X}_n) = \sum_{z_n \in \mathsf{supp}(n)} P_n(\mathbf{X}_n, \mathbf{Z}_n)$$

be the latent-marginalized value of the sub-circuit P_n; at the root $P_{n_r}(\mathbf{X}) = \sum_{z \in \mathsf{supp}(n_r)} P(\mathbf{X}, z)$. We prove Equation 4 by induction over n.

In the base case where n is an input node, according to the Definition 3.1, we have that

$$\sum_{z \in \mathsf{supp}(n)} P_n(\mathbf{X}_n, z_n) = P_n(\mathbf{X}_n, \{\}) = P_n(\mathbf{X}_n).$$

Next, assume n is a sum node, and Equation 4 holds for all its children, we have

$$\begin{aligned}
&\sum_{z_n \in \mathsf{supp}(n)} P_n(\mathbf{X}_n, z_n) \\
&= \sum_{c \in \mathrm{ch}(n)} \theta_{n,c} \cdot \sum_{z_c \in \mathsf{supp}(c)} P_c(\mathbf{X}_c, z_c) \\
&= \sum_{c \in \mathrm{ch}(n)} \theta_{n,c} \cdot P_c(\mathbf{X}_c) = P_n(\mathbf{X}_n).
\end{aligned}$$

Finally, if n is a product node, and Equation 4 holds for all its children, we have

$$\begin{aligned}
&\sum_{z_n \in \mathsf{supp}(n)} P_n(\mathbf{X}_n, z_n) \\
&= \sum_{z_n \in \mathsf{supp}(n)} \prod_{c \in \mathrm{ch}(n)} P_c(\mathbf{X}_c, z_c) \\
&= \prod_{c \in \mathrm{ch}(n)} \sum_{z_c \in \mathsf{supp}(c)} P_c(\mathbf{X}_c, z_c) \\
&= \prod_{c \in \mathrm{ch}(n)} P_c(\mathbf{X}_c) = P_n(\mathbf{X}_n).
\end{aligned}$$

By applying the induction over all nodes in PC, we have Equation 4. □

Proposition B.1 tells us the augmentation is exact, but not yet what $P(\mathbf{X}, \mathbf{Z})$ actually is. The next result makes the joint explicit: along any latent assignment z, the probability is simply the product of the selected sum-edge weights times the emissions of the active input nodes. This proposition is the workhorse of everything that follows: the latent marginal, the conditional, and both entropies are all read off from it.

B.2 JOINT DISTRIBUTION

Proposition B.2 (Joint distribution)**.** *For a smooth, decomposable, locally normalized PC, for every $z \in \mathsf{supp}(n_r)$, the joint distribution of $P(\mathbf{X}, \mathbf{Z})$ is defined by*

$$p(x, z) = \prod_{(n,c)\in z} \theta_{n,c} \prod_{m\in I_{n_r}(z)} f_m(x_m), \tag{5}$$

where $I_n(z) = I(z) \cup \{n : n \text{ is an input node}\}$, and $I(z) = \{c : \exists(n, c) \in z \text{ and } c \text{ is an input node}\}$.

Proof. We prove Equation 5 by induction over n.

In the base case where n is an input node, according to the Definition 3.1, we have

$$p_n(x_n, z_n) = f_n(x_n).$$

When n is a sum node, and Equation 5 holds for all its children, we have

$$\begin{aligned}
&p_n(x_n, z_n) \\
&= \theta_{n,c} \cdot p_c(x_c, z_c) \\
&= \theta_{n,c} \cdot \prod_{(c,m)\in z_c} \theta_{c,m} \prod_{m\in I_c(z_c)} f_m(x_m) \\
&= \prod_{(n',c')\in z_n} \theta_{n',c'} \prod_{m\in I_n(z_n)} f_m(x_m).
\end{aligned}$$

When n is a product node, and Equation 5 holds for all its children, we have

$$\begin{aligned}
&p_n(x_n, z_n) \\
&= \prod_{c\in \mathrm{ch}(n)} p_c(x_c, z_c) \\
&= \prod_{c\in \mathrm{ch}(n)} \prod_{(c,m)\in z_c} \theta_{c,m} \prod_{m\in I_c(z_c)} f_m(x_m) \\
&= \prod_{(n',c')\in z_n} \theta_{n',c'} \prod_{m\in I_n(z_n)} f_m(x_m).
\end{aligned}$$

By applying the induction over all nodes in PC, we have Equation 5. □

Summing the joint over $\mathbf{X}$ erases the emissions and leaves the law of the latent code by itself. We isolate this marginal because it is the only ingredient the latent entropy $H(\mathbf{Z})$ depends on.

B.3 DISTRIBUTION OF THE LATENT VARIABLES

Corollary B.1 (Distribution of latent variables)**.** *For a smooth, decomposable, and locally normalized PC, every $z \in \mathsf{supp}(n_r)$ satisfies*

$$p(z) = \prod_{(n,c)\in z} \theta_{n,c}, \tag{6}$$

and these weights sum to one, $\sum_{z\in\mathsf{supp}(n)} p(z) = 1$.

Proof. Marginalizing $\mathbf{X}$ out of Proposition B.2 and using $\sum_{x_m} f_m(x_m) = 1$ at every input node leaves $p(z) = \prod_{(n,c)\in z} \theta_{n,c}$. Normalization $\sum_{z\in\mathsf{supp}(n_r)} p(z) = 1$ then follows from Proposition B.1. □

Dividing the joint by this marginal gives the law of the observations once the code is fixed. The key point is that conditioning collapses the circuit to a single product over the active input nodes, so $P(\mathbf{X} \mid \mathbf{Z} = z)$ factorizes; this is precisely what later lets the conditional entropy $H(\mathbf{X} \mid \mathbf{Z})$ split into a sum of independent per-input terms.

B.4 CONDITIONAL DISTRIBUTION

Corollary B.2 (Conditional distribution)**.** *For a smooth, decomposable, locally normalized PC and every $z \in \mathsf{supp}(n)$, the conditional distribution given $\mathbf{Z}$ can be represented as below,*

$$P(\mathbf{X}|\mathbf{Z}=z) = \prod_{n \in I_{n_r}(z)} f_n(x_n). \tag{7}$$

Proof. By applying Proposition B.2 and Corollary B.1, we have

$$\begin{aligned} P(\mathbf{X}|\mathbf{Z}=z) &= \frac{P(\mathbf{X}, z)}{p(z)} \\ &= \frac{\prod_{(n,c)\in z} \theta_{n,c} \cdot \prod_{m \in I_{n_r}(z_n)} f_m(x_m)}{\prod_{(n,c)\in z} \theta_{n,c}} \\ &= \prod_{n \in I_{n_r}(z)} f_n(x_n). \end{aligned}$$

□

The three results above all describe the model for a *single* fixed assignment z. To assemble an entropy, we must instead average over z, which means weighting each node and edge by how likely the latent variable is to pass through it. The final lemma identifies these visiting probabilities with the top-down probabilities of Definition 2.3; this is what converts every otherwise-exponential sum over z into a single-pass computation over the circuit.

B.5 TOP-DOWN PROBABILITIES ARE SELECTION PROBABILITIES

Lemma B.1 (Top-down probabilities are selection probabilities)**.** *For a locally normalized PC and a random latent variable $\mathbf{Z} \sim P(\mathbf{Z})$, for every node n and sum edge (n, c),*

$$\begin{aligned} &\Pr[n \in \mathcal{N}(\mathbf{Z})] = \mathrm{TD}(n), \\ &\Pr[(n,c) \in \mathbf{Z}] = \theta_{n,c} \cdot \mathrm{TD}(n) = \mathrm{TD}(\theta_{n,c}). \end{aligned}$$

Proof. We induct in a topological order from the root. For the base case, $\Pr[n_r \in \mathcal{N}(\mathbf{Z})] = 1 = \mathrm{TD}(n_r)$. For $n \neq n_r$, the definition of $\mathsf{supp}(n)$ (refer to Definition 3.1) gives that every node n appears at most once in z. Therefore, the events $\big[m \in \mathcal{N}(\mathbf{Z})\big]$ over $m \in \mathrm{pa}(n)$ are disjoint (i.e., there is no $m_1 \neq m_2$ such that $m_1, m_2 \in \mathrm{pa}(n)$, $m_1 \in \mathcal{N}(z)$ and $m_2 \in \mathcal{N}(z)$). When n is a sum node, we have

$$\begin{aligned} \Pr[n \in \mathcal{N}(\mathbf{Z})] &= \sum_{m \in \mathrm{pa}(n)} \Pr[m \in \mathcal{N}(\mathbf{Z})] \\ &= \sum_{m \in \mathrm{pa}(n)} \mathrm{TD}(m) = \mathrm{TD}(n). \end{aligned}$$

When n is a product or input node, we have

$$\Pr[n \in \mathcal{N}(\mathbf{Z})] = \sum_{m \in \mathrm{pa}(n)} \Pr[m \in \mathcal{N}(\mathbf{Z})] \times \Pr\big[(m,n) \in \mathbf{Z} | m \in \mathcal{N}(\mathbf{Z})\big].$$

If n is a product node, then every parent m is a sum node, which selects n with probability $\theta_{m,n}$, so $\Pr[n \in \mathcal{N}(\mathbf{Z})] = \sum_m \theta_{m,n} \cdot \mathrm{TD}(m) = \mathrm{TD}(n)$.

The edge statement follows since, given $n \in \mathcal{N}(\mathbf{Z}) \cap \mathcal{S}$, child c is selected with probability $\theta_{n,c}$: $\Pr[(n,c) \in \mathbf{Z}] = \theta_{n,c} \cdot \mathrm{TD}(n) = \mathrm{TD}(\theta_{n,c})$. □

C PROOFS OF THE ENTROPY BOUNDS

With the latent model in hand, the entropy results of the main text follow by combining the facts of Appendix B. The two tractable entropies come first: $H(\mathbf{Z})$ is built from the latent marginal (Corollary B.1) and $H(\mathbf{X} \mid \mathbf{Z})$ from the conditional factorization (Corollary B.2), each collapsed into a single circuit pass by the selection probabilities of Lemma B.1. These two quantities then bracket the intractable $H(\mathbf{X})$ from both sides: their sum is the joint entropy $H(\mathbf{X}, \mathbf{Z})$, which over-counts by exactly $H(\mathbf{Z} \mid \mathbf{X}) \geq 0$ and so gives the upper bound; subtracting instead a *tractable* over-estimate of $H(\mathbf{Z} \mid \mathbf{X})$ gives the lower bound. We prove the three results in this order.

C.1 PROOF OF THEOREM 4.1 (LATENT ENTROPY)

Since $p(z)$ factorizes over the selected edges (Corollary B.1), $\log p(z)$ is a sum over those edges, and the expectation of each term is supplied by Lemma B.1.

Proof. By Corollary B.1, for all $z \in \mathsf{supp}(n)$, we have $\log p(z) = \sum_{(n,c)\in z} \log \theta_{n,c}$. Hence

$$\begin{aligned} H(\mathbf{Z}) &= -\sum_{z\in\mathsf{supp}(n)} p(z) \log p(z) \\ &= -\sum_{z\in\mathsf{supp}(n)} p(z) \sum_{(n,c)\in z} \log \theta_{n,c} \\ &= -\sum_{(n,c)\in z} \log \theta_{n,c} \sum_{z\in\mathsf{supp}(n)} p(z) \cdot \mathbb{1}[(n,c) \in z], \end{aligned}$$

where $\sum_{z\in\mathsf{supp}(n)} p(z) \cdot \mathbb{1}[(n,c) \in z] = \Pr[(n,c) \in \mathbf{Z}] = \mathrm{TD}(\theta_{n,c})$, according to Lemma B.1. □

C.2 PROOF OF THEOREM 4.2 (CONDITIONAL ENTROPY)

The conditional factorization (Corollary B.2) makes $H(\mathbf{X} \mid \mathbf{Z} = z)$ a sum of per-input entropies; averaging over z and applying Lemma B.1 reweights each input node by its top-down probability.

Proof. According to the Corollary B.2, conditioned on $\mathbf{Z} = z$, the entropy of $H(\mathbf{X}|\mathbf{Z} = z)$ can be decomposed as $\sum_{n\in I_{n_r}(z)} H(f_n(\mathbf{X}_n))$. Averaging over $z \in \mathsf{supp}(n)$ and exchanging summation,

$$\begin{aligned} H(\mathbf{X}|\mathbf{Z}) &= \sum_{z\in\mathsf{supp}(n_r)} p(z) \sum_{n\in I_{n_r}(z)} H(f_n(\mathbf{X}_n)) \\ &= \sum_{n\in\mathcal{I}} H(f_n(\mathbf{X}_n)) \sum_{z\in\mathsf{supp}(n_r)} p(z) \cdot \mathbb{1}[n \in \mathcal{N}(z)] \\ &= \sum_{n\in\mathcal{I}} \mathrm{TD}(n) \cdot H(f_n(\mathbf{X}_n)), \end{aligned}$$

where $\sum_z p(z) \cdot \mathbb{1}[n \in \mathcal{N}(z)] = \Pr[n \in \mathcal{N}(\mathbf{Z})] = \mathrm{TD}(n)$, according to Lemma B.1. □

C.3 PROOF OF PROPOSITION 4.1 (LOWER BOUND)

The upper bound merely drops $H(\mathbf{Z} \mid \mathbf{X}) \geq 0$; the lower bound keeps that term and overestimates it by the local selection entropies of Definition 4.1, which condition each sum node's choice only on the variables it directly emits.

Proof. We decompose the entropy of $\mathbf{X}$ into $H(\mathbf{X}) = H(\mathbf{X}|\mathbf{Z}) + H(\mathbf{Z}) - H(\mathbf{Z}|\mathbf{X})$, and the first two terms are exact, so it suffices to bound $H(\mathbf{Z}|\mathbf{X})$ from above by $\tilde{H}(\mathbf{Z}|\mathbf{X})$.

Bound		Llama3.1-8B		Qwen2.5-7B		Mistral-7B	
		GSM8K	CSQA	GSM8K	CSQA	GSM8K	CSQA
Upper	Mix Ratio	-0.05	0.1	0.1	-0.2	-0.3	0.1
	k	4	8	4	8	8	4
Lower	Mix Ratio	-0.1	-0.1	-0.05	-0.2	-0.2	0.3
	k	8	4	4	8	8	4
Empirical	Mix Ratio	-0.1	0.05	-0.3	-0.05	-0.3	0.2
	k	8	8	4	4	8	4

Table 2: Hyperparameter settings (mix ratio and k) for each strategy across models and datasets.

According to Definition 4.1, each $\mathbf{Z}^{(n)}$ takes finite values in $\mathrm{ch}(n) \cup \{\varnothing\}$, hence $(\mathbf{Z}^{(n)})_{n\in\mathcal{S}}$ is a finite family of discrete variables and its entropy chain rule holds under any enumeration. We adopt a topological enumeration $\prec$ of $\mathcal{S}$ (root first); since $\mathbf{Z}$ and $(\mathbf{Z}^{(n)})_{n\in\mathcal{S}}$ determine each other, we have

$$H(\mathbf{Z}|\mathbf{X}) = H((\mathbf{Z}^{(n)})_{n\in\mathcal{S}}|\mathbf{X}) = \sum_{n\in\mathcal{S}} H\big(\mathbf{Z}^{(n)}|(\mathbf{Z}^{(m)})_{m\prec n},\, \mathbf{X}\big).$$

Under this enumeration, whether n is reached and through which parent are already fixed by $(\mathbf{Z}^{(m)})_{m\prec n}$. Let U_n be the reached parent of n, i.e. the $m \in \mathrm{pa}(n) \cap \mathcal{N}(\mathbf{Z})$ when it exists; $\varnothing$ otherwise. U_n is unique because the events $[m \in \mathcal{N}(\mathbf{Z})]$ over $m \in \mathrm{pa}(n)$ are pairwise disjoint (Lemma B.1). Thus U_n is a function of $(\mathbf{Z}^{(m)})_{m\prec n}$ and $\tilde{\mathbf{X}}_n \subseteq \mathbf{X}$, so conditioning on less cannot decrease the entropy:

$$H\big(\mathbf{Z}^{(n)}|(\mathbf{Z}^{(m)})_{m\prec n},\, \mathbf{X}\big) \;\le\; H\big(\mathbf{Z}^{(n)}|U_n,\, \tilde{\mathbf{X}}_n\big).$$

By Corollary B.1, the latent variable's probability factorizes as $p(z) = \prod_{(n,c)\in z} \theta_{n,c}$, so once $n \in \mathcal{N}(\mathbf{Z})$,the selection satisfies that $\Pr[\mathbf{Z}^{(n)} = c \mid n \in \mathcal{N}(\mathbf{Z})] = \theta_{n,c}$, no matter how n was reached; with the emission factorization of Corollary B.2, the conditional distribution of $(\mathbf{Z}^{(n)}, \tilde{\mathbf{X}}_n)$ given $U_n = m \in \mathrm{pa}(n)$ is the same for every such m and equals its conditional distribution given $n \in \mathcal{N}(\mathbf{Z})$. Hence $H(\mathbf{Z}^{(n)} \mid U_n = m, \tilde{\mathbf{X}}_n) = \tilde{H}_n$ for every $m \in \mathrm{pa}(n)$, while $U_n = \varnothing$ forces $\mathbf{Z}^{(n)} = \varnothing$ and contributes 0. With $\Pr[U_n = m] = \Pr[m \in \mathcal{N}(\mathbf{Z})] = \mathrm{TD}(m)$ (Lemma B.1) and $\sum_{m\in\mathrm{pa}(n)} \mathrm{TD}(m) = \mathrm{TD}(n)$, we have

$$H\big(\mathbf{Z}^{(n)}|U_n,\, \tilde{\mathbf{X}}_n\big) = \sum_{m\in\mathrm{pa}(n)} \mathrm{TD}(m)\cdot\tilde{H}_n = \mathrm{TD}(n)\cdot\tilde{H}_n.$$

Summing over $n \in \mathcal{S}$ gives $H(\mathbf{Z}|\mathbf{X}) \le \sum_{n\in\mathcal{S}} \mathrm{TD}(n)\cdot\tilde{H}_n = \tilde{H}(\mathbf{Z}|\mathbf{X})$, and the bound follows.

Computing $\tilde{H}_n$ needs the value of $p(z^{(n)}, \tilde{x}_n | n \in \mathcal{N}(\mathbf{Z}))$ for every $z^{(n)}$ and $\tilde{x}_n$. There are at most C choices for $z^{(n)}$, and at most V choices for $\tilde{x}_n$. The time complexity of computing $\tilde{H}_n$ is $O(C \cdot V)$. Since we need to compute $\tilde{H}_n$ for every $n \in \mathcal{S}$, the final complexity is $O(C \cdot V \cdot |P|)$. □

D CONFIGURATIONS OF DIFFERENT DECODING STRATEGIES

We follow the configurations in Baghaei Potraghloo et al. [2026] that `min_p` $= 0.1$, `top_p` $= 0.9$, and the α in Top-H is 0.4. P-Less is a hyperparameter-free method. To align with the Top-K filter before the entropy estimation, we set `top_k` $= 50$. The hyperparameters in our method are presented in Table 2. These hyperparameters were determined via a grid search over the `mix_ratio` $\in \{-0.3, -0.2, -0.1, -0.05, 0.05, 0.1, 0.2, 0.3\}$ and $k \in \{4, 8\}$.