Probabilistic Loss Functions for Self-Supervised SAT Solvers

Anonymous Author(s)
Affiliation
Address
email

Abstract

In this study, we design novel probabilistic loss functions for training neural networks in an unsupervised way to tackle the CNF-SAT problem. In particular, we leverage the power of the *Lovász Local Lemma* (LLL) in obtaining satisfiability certificates to train models that achieve this in a differentiable manner. Given that the LLL provides provable discretization procedures, such as the Moser-Tardos algorithm, our approach leads to an end-to-end hybrid SAT solver.

1 Introduction

There is a recent line of work in the intersection of machine learning and combinatorial optimization [Karalias and Loukas, 2020, Yau et al., 2024, Karalias et al., 2022, Wang et al., 2019, Selsam et al., 2019]. Karalias and Loukas [2020] propose a principled unsupervised learning framework for tackling a large class of NP-hard problems defined on graphs. More specifically, the authors construct a loss function that is inspired by the Probabilistic Method and that guarantees the existence of high-quality discrete solutions. Such solutions can also be constructed deterministically in an efficient way. We apply this framework in the context of the CNF-SAT problem and present the loss function that arises naturally, which we refer to as the Union Bound loss. The CNF-SAT problem – a typical NP-complete problem with a wide variety of applications – asks for a boolean assignment x to a set of variables that appear in a CNF boolean formula ϕ (a conjunction of disjunctions) such that x satisfies all clauses of ϕ [Karp, 1972]. Using the symmetric and asymmetric versions of the Lovász Local Lemma (LLL) we propose new loss functions that are exploiting the instance structure in a more direct way [Erdős and Lovász, 1974, Alon and Spencer, 2016]. The LLL exhibits two appealing features: it provides nontrivial certificates of satisfiability and efficient algorithms for finding a satisfying assignment. The Moser-Tardos result showed that the simplest local search algorithm is efficient under the conditions of the LLL [Moser and Tardos, 2010]. We obtain the certificate in a differentiable manner that when combined with the efficient discretization step gives rise to an end-to-end hybrid SAT solver. The proposed framework can thus be thought of as a *learnable stochastic local search* algorithm and can be easily extended to tackle more general problems in the class of Constraint Satisfaction Problems (CSPs).

In the context of SAT, one well-known supervised model is NeuroSAT, where a message passing neural network (MPNN) is trained to predict satisfiability in a supervised way [Selsam et al., 2019]. SATNet introduces a differentiable maximum satisfiability (MAXSAT) solver that can be directly embedded in deep learning models, enabling them to incorporate logical reasoning [Wang et al., 2019]. Another way of incorporating neural methods in SAT solvers is to inject a neural network in a well-known heuristic. The authors in Yolcu and Poczos [2019] propose learning SAT solver heuristics from scratch using deep reinforcement learning. FourierSAT considers continuous extensions for SAT problems via Fourier analysis of Boolean functions and continuous optimization on the resulting multilinear polynomials [Kyrillidis et al., 2020].

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

We provide a conceptual motivation for the use of the LLL via the notion of entropy rate, and we hypothesize that satisfiability certificates that are effective for high-entropy distributions are amenable to continuous optimization algorithms. We experimented with different loss functions on a wide class of randomly generated CNF formulae. The results showed that in the low dependency degree regime, the symmetric LLL can outperform the Union Bound and there are other cases for which the asymmetric LLL works better than both of the other two losses. Furthermore, we propose a Graph Neural Network (GNN) parameterization (we refer to our model as GINGAT) on which we conducted experiments. The results obtained showed that the neural parameterization can typically lead to improved results (compared to the pure Adam-based approach) on the training set and in some cases on an unseen test set. We finally present interesting directions for future research.

2 Conceptual motivation

We want to design an unsupervised learning framework for solving Clause CSPs (where each constraint forbids a single assignment to some subset of the variables). More specifically, we want to define loss functions that are derived from probabilistic statements implying the satisfiability of a given CSP instance. Moreover, we want the probabilistic statements to also provide us with an efficient procedure for actually finding such a satisfying assignment. To simplify our discussion, we will focus on the search version of the CNF-SAT problem, in which we are given a CNF boolean formula ϕ on n variables $x_1, \ldots, x_n$ and m clauses $C_1, \ldots, C_m$ (where each clause is the disjunction of some number of literals) and the goal is to find an assignment $x \in \{0,1\}^n$ to the variables such that every clause evaluates to True. We work with product measures on $[n]$, i.e. we consider only marginal vectors $p = (p_1, \ldots, p_n)$ such that $\mathbb{P}\left(X_i = \text{True}\right) = p_i$, and the random variables $X_i \sim \text{Ber}(p_i)$ are mutually independent.

We write $\boldsymbol{x} \sim p$ to mean that we sample an assignment $\boldsymbol{x}$ according to distribution p. For every $j \in [m]$, define the event $B_j =$ "clause j is false (under x)". Given some ϕ, we want to find conditions on $p \in [0,1]^n$ that imply $\mathbb{P}_{\boldsymbol{x}\sim p}\left(\cap_{j\in[m]}\overline{B_j}\right) > 0$, in which case a satisfying assignment exists (by the probabilistic method). We will write $\phi \in \text{SAT}$ to mean that ϕ is satisfiable. When looking for sufficient conditions for satisfiability, probably the simplest condition is the one that follows from the Union Bound:

$$\sum_{j\in[m]} \mathbb{P}_{\boldsymbol{x}\sim p}\left(C_j(\boldsymbol{x}) = 0\right) < 1 \implies \phi \in \text{SAT}. \tag{1}$$

Denote the set of vectors p that satisfy the antecedent of 1 with $\mathcal{P}_{\text{UB}}(\phi)$. To simplify notation, we define $f_j(p) \coloneqq \mathbb{P}_{\boldsymbol{x}\sim p}\left(C_j(\boldsymbol{x}) = 0\right), \forall j \in [m]$. Since the variables X_i's are independent (from our setup), we have that $f_j(p) = \prod_{i\in C_j^-} p_i \cdot \prod_{i\in C_j^+}(1 - p_i)$, where C_j^- is the set of variables appearing in clause C_j negatively and C_j^+ is the set of variables appearing in clause C_j positively. Now, it is easy to see that given some $p \in \mathcal{P}_{\text{UB}}(\phi)$, a satisfying assignment can be found efficiently using the method of conditionals expectations. Thus, a natural choice for the loss function to be used is $\mathcal{L}_{\text{UB}}(p, \phi) \coloneqq \sum_{j\in[m]} f_j(p)$. We will refer to $\mathcal{L}_{\text{UB}}$ as the Union Bound loss function.

The Union Bound loss function presented above uses a worst case bound in which all events are mutually independent. In practice, clauses will share variables which introduces dependencies between them. Hence, we will investigate loss functions that take into account this dependency structure. To this end, we will consider the Lovász Local Lemma (LLL), which leverages the dependency structure of the clauses to certify satisfiability. Specifically, the symmetric version of the (variable) LLL, yields the following:

$$\forall j \in [m], f_j(p) \leq \frac{1}{e(d+1)} \implies \phi \in \text{SAT}, \tag{2}$$

where d is the maximum degree across the vertices in the dependency graph of ϕ (vertices correspond to clauses and we draw an edge between two vertices if the corresponding clauses share a variable). The asymmetric version of the LLL provides the following

$$\exists \mu : [m] \to [0,1) : \forall j \in [m], f_j(p) \leq \mu(j) \cdot \prod_{j'\in N(j)} (1 - \mu(j')) \implies \phi \in \text{SAT}, \tag{3}$$

where $N(j)$ denotes the set of clauses that share some variable with clause C_j. Denote the set of vectors p that satisfy the antecedent of 2 with $\mathcal{P}_{\text{LLL}}(\phi)$ and the set of vectors p and μ that meet 3 with $\mathcal{P}_{\text{GLLL}}(\phi)$. Note that for the LLL, we have point-wise inequalities that need to hold for every clause (unlike the global sum of the Union Bound).

We define the loss functions as follows. For the symmetric LLL, we have

$$\mathcal{L}_{\text{LLL}}(p, \phi) \coloneqq \sum_{j \in [m]} \text{ReLU}\left(f_j(p) - 1/(e(d+1))\right).$$

For the asymmetric LLL, the loss is given by

$$\mathcal{L}_{\text{GLLL}}(p, \mu, \phi) \coloneqq \sum_{j \in [m]} \text{ReLU}\left(f_j(p) - \mu(j) \cdot \prod_{j' \in N(j)} (1 - \mu(j'))\right),$$

where $\text{ReLU}(x) = \max(x, 0)$.

Altogether, we have three different loss functions and a condition on each that implies the satisfiability of the given CNF: $\mathcal{L}_{\text{UB}}(p, \phi) < 1, \mathcal{L}_{\text{LLL}}(p, \phi) = 0$ and $\mathcal{L}_{\text{GLLL}}(p, \mu, \phi) = 0$.

The premise of this work is that the integrality of predicted solutions can play an important role in the success of the model. Consider the loss in (1). If our algorithm produces p which leads to a constant probability of violation for each clause, then naturally as the number of clauses increases then the union bound will be harder to satisfy. Hence, the probability of clause violation will have to decrease as the number of clauses grows $\mathbf{p}$. A natural way for this to occur is if $\mathbf{p}$ tends to an integral assignment as the formula grows. Our goal is to investigate this and propose losses that could overcome this challenge. To clarify this idea, we perform the following simple test which gives us an indication of the "effective entropy" of the Union Bound and the symmetric LLL. We construct random k-CNFs $\phi_{n,i}$ for some given density. For a CNF ϕ, a satisfying assignment x of ϕ, a function $\mathcal{L}$ and a small constant δ, we denote with T^δ the smallest factor of δ perturbation we need to add to the all 1/2 vector in the direction of x so that we get an element $p \in \mathcal{P}_{\mathcal{L},\phi}$ (this measures some notion of "satisfiability threshold" for the function under consideration). In Figure 3a, we show the average T^δ value for each value of n, where n is the number of variables in random 3-CNFs of density 2.5. similarly, Figure 3b shows the same phenomenon for density 3.5. For both cases, we can see that the symmetric LLL kicks in with distributions that are away from binary while the union bound threshold converges to 0.5.

In what follows, we formalize this phenomenon by looking at the sequence of maximum average entropy values of marginal vectors that satisfy the LLL condition as we consider larger and larger CNF formulas coming from an infinite family. We define the analogous quantity for the case of the Union Bound, we then compare the two and show that there are CNF families for which the LLL is effective with joint distributions of higher per-variable entropy. We note that we will only consider satisfiable SAT formulas so that the sets $\mathcal{P}_{\text{UB}}, \mathcal{P}_{\text{LLL}}$ and $\mathcal{P}_{\text{GLLL}}$ are non-empty.

Definition 1 *For $p \in [0,1]^n$, the entropy of p is defined as $H(p) \coloneqq \sum_{i \in [n]} H(p_i) \coloneqq -\sum_{i \in [n]} (p_i \log(p_i) + (1 - p_i) \log(1 - p_i))$. This is the joint entropy of n independent Ber(p_i) variables. Furthermore, define the average entropy of p as $\overline{H}(p) \coloneqq \frac{1}{n} H(p)$.*

Definition 2 *Let $\Phi = \{\phi_z\}_{z \in \mathbb{N}}$ be an infinite satisfiable CNF family and let $\mathcal{L} \in \{UB, LLL, GLLL\}$. We define the $\mathcal{L}$-entropy-rate of Φ as: $H^\infty_{\mathcal{L}}(\Phi) \coloneqq \limsup_{z \to \infty} \sup_{p \in \mathcal{P}_{\mathcal{L}}(\phi_z)} \overline{H}(p)$.*

We start our discussion by showing that there exists an infinite family Φ_1 with $d = O(1)$, $H^\infty_{\text{UB}}(\Phi_1) = 0$ and $H^\infty_{\text{LLL}}(\Phi_1) \geq \frac{1}{3}$, where d is the maximum degree in the clause dependency graph. We start by considering an easy-to-construct family, every CNF of which admits a unique satisfying assignment.

Claim 1 (proof in Appendix A) *Let ϕ be a k-CNF that admits a unique satisfying assignment $p^* \in \{0,1\}^n$. For simplicity, assume that $n = kz$, for some $z \in \mathbb{N}$. Then ϕ can be equivalently written as a k-CNF on n variables and $(2^k - 1)z$ clauses.*

We consider the constant-degree infinite family of 3-CNFs described in the proof of Claim 1 (i.e. we fix $k = 3$) with unique satisfying assignment $p^* = \mathbf{1}^n$. Denote the CNF family with Φ_1.

Claim 2 (proof in Appendix A) *Let Φ_1 be the infinite family constructed above. Then, $H^{\infty}_{LLL}(\Phi_1) = \frac{1}{3}$ and $H^{\infty}_{UB}(\Phi_1) = 0$.*

We now show how to get a constant entropy rate for the symmetric LLL with a CNF family having a linear maximum degree.

Claim 3 (proof in Appendix A) *There exists an infinite family Φ_2 such that $H^{\infty}_{LLL}(\Phi_2) = 1$.*

We now separate the asymmetric LLL from the Union Bound function and the symmetric LLL. More specifically, we construct an infinite family Φ_3 with $d = O(m)$, $H^{\infty}_{\mathrm{UB}}(\Phi_3) = H^{\infty}_{\mathrm{LLL}}(\Phi_3) = 0$ and $H^{\infty}_{\mathrm{GLLL}}(\Phi_3) \geq \frac{1}{4}$.

Claim 4 (proof in Appendix A) *There exists an infinite family Φ_3 such that: $H^{\infty}_{UB}(\Phi_3) = 0, H^{\infty}_{LLL}(\Phi_3) = 0, H^{\infty}_{GLLL}(\Phi_3) \geq \frac{1}{4}$.*

The families of synthetic instances discussed above serve as motivating examples for our discussion. We conjecture that more powerful theoretical tools will be required to establish the benefits of the LLL for realistic data distributions.

3 Methodology

We first present the different CNF generators used and the different loss functions considered. We then explain the implementation that we used for optimizing the loss functions. In this work, we primarily focus on minimizing the loss by directly optimizing the assignment in the hypercube using first order methods (i.e., Adam). By removing the model, we can make consistent observations about the loss that do not depend on a specific neural net architecture.

CNF generation We consider different classes (types) of CNF formulas, each encoding a different combinatorial problem. We use an established CNF generation library for the following CNF types: Coloring, EvenColoring, GraphOrdering and PerfectMatching [Massimo Lauria]. The underlying structures are generated from a variety of random graph models: Barabasi–Albert, expected-degree model, Erdős–Rényi, power-law cluster model, random regular graphs and Watts-Strogatz [Hagberg et al., 2008]. We note that we use the MiniSAT solver to filter out unsatisfiable instances in the generation process. The generated CNFs have between 20 and 400 variables and between 50 and 8,000 clauses. The average ratio (clauses over variables) is 6.75. For testing Adam, we generated a sample of about 40 instances for each CNF type/graph generator combination.

Loss functions We implemented and experimented with different loss functions: the Union Bound loss function and loss functions that are inspired by the LLL. We used smooth approximations of the ReLU function. Smooth activations like GELU or Softplus provide non-zero gradients that improve Adam's stability, while ReLU is faster but can suffer from dead neurons due to zero gradients for negative inputs. In the case of the asymmetric LLL (implication 3), recall that there are free parameters for each clause. In the experiments, we set $\mu(j) = 1/(d_j + 1), \forall j \in [m]$, where d_j is the degree of clause j in the dependency graph of ϕ.

Stochastic Gradient Descent implementation The optimization is performed with five restarts, where each restart initializes the logits using a standard normal distribution (`randn`) to avoid poor local minima. Each batch contains multiple CNF instances, with literal and clause padding handled via masks, and instances that reach a threshold are masked out to prevent unnecessary updates. For each CNF, the lowest left-hand side (LHS) value seen across steps is tracked to monitor progress. This LHS value corresponds to the loss we are optimizing[1] and that we try to lower below a certain threshold (1 in the case of the union bound and 0 in the case of the LLL[2]). Gradients are updated using the Adam optimizer, which benefits from adaptive learning rates. Furthermore, we applied early stopping. Logits are converted to probabilities via a sigmoid, and probabilities are clamped between 10^{-12} and $1 - 10^{-12}$ to prevent numerical issues. Finally, instances are considered solved and removed from further updates once their LHS drops below a function-dependent threshold.

[1] In the case of the LLL, this is not the proxy loss but the actual sum of ReLU values.

[2] We used a threshold of 10^{-5} for the LLL loss functions.

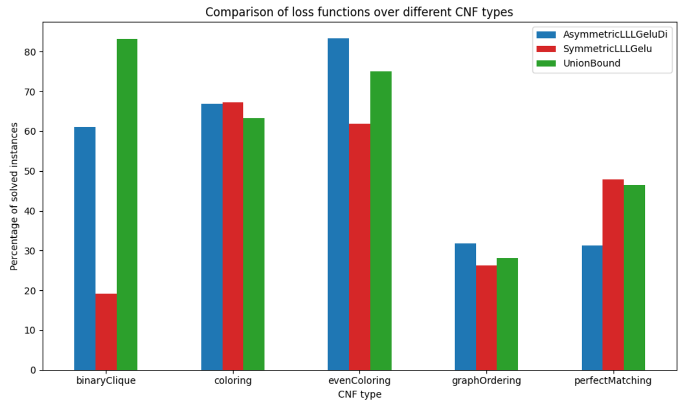


Figure 1: Percentage of solved instances per CNF type

Graph Neural Network parameterization We propose a neural architecture that we tested against the Adam-based implementation. The architecture combines two well-known Graph Neural Networks (GNNs): the *Graph Isomorphism Network* (GIN) and the *Graph Attention Network* (GAT) [Xu et al., 2019, Velickovic et al., 2018]. We refer to the proposed neural network as the GINGAT model. This model is a hybrid GNN designed to operate on bipartite graphs connecting variables and clauses. It integrates GIN layers that capture graph structure with GAT layers that incorporate attention-based relational information. The model processes node features and edge attributes through a sequence of these layers, optionally using residual connections, dropout, and graph normalization. This combination of GIN and GAT layers allows the model to capture both structural patterns and the relative importance of neighboring nodes, making it effective for reasoning over the variable-clause bipartite graphs coming from SAT instances. The key hyperparameters of the model are summarized in Table 2. Finally, the instances that were used for training the GINGAT model are shown in Table 3. We used a 70/15/15 training/validation/test split.

The experiments were run on a machine with an RTX 3090 GPU, 12 CPU cores (Xeon E5-2650 v4) and 32GB RAM.

4 Results

In Figure 1 and Table 5, we show the percentage of CNFs that were solved[3] using the pure Adam-based approach by each loss function. Figure 4 and Table 6 contain a more in-depth comparison of the different loss functions on the basis of a CNF type and graph generator combination, where in the Figure we show the difference in performance (difference in percentage of CNFs solved) for each pair of functions in increasing order. The effect of the normalized maximum and average degree of the clause dependency graph on the performance difference between the functions is shown in Figures 2, 5, 6 and 7. Finally, in Figure 8, we observe the runtime of the different loss functions. We separately plot the runtime of Adam for processing an individual CNF (that was solved or not) and the runtime for solving a CNF.

Concerning the results obtained on the GINGAT training, the best hyperparameters found for training the GINGAT model were determined through extensive tuning. These parameters are summarized in Table 4. The results obtained by GINGAT in training, validation and testing for some selected datasets on which we observed a performance that was comparable or higher than the pure Adam-based approach are shown in Table 1. We hypothesize that a future investigation into suitable GNN architectures for each CNF type can give an improvement on the Adam-based approach across the board.

[3]By "solving", we mean that the algorithm found a certificate of satisfiability. Recall that we are guaranteed to then be able to find a solution, but for simplicity we omit this discretization step.

Table 1: Results of GINGAT for different CNF types, generators, and loss functions

CNF type	Graph generator	Parameters	Loss function	Test %	Train %	Val %	Adam %
graphOrdering	watts strogatz graph	k: 4, p: 0.3, n: 10	SymmetricLLLGelu	62.09	100	65.36	50
			UnionBound	62.09	100	68.63	53
perfectMatching	powerlaw cluster graph	m: 4, p: 0.05, n: 10	AsymmetricLLLGeluDi	53.52	100	91.67	53
graphOrdering	expected degree graph	a: 5, min: 2, max: 4, n: 10	SymmetricLLLGelu	28.76	100	47.71	33

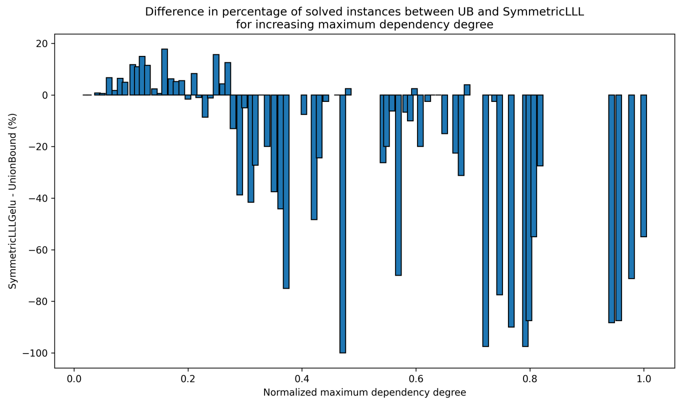


Figure 2: Symmetric LLL vs Union Bound for increasing maximum dependency degree

5 Conclusions

In this work, we proposed an unsupervised learning approach for solving the CNF-SAT problem using probabilistic loss functions. We presented LLL-inspired losses that explicitly use instance-specific structure unlike the Union Bound (UB) loss. We implemented both a GNN model and a pure Adam-based method for finding satisfiability certificates on different types of CNFs. Empirically, the symmetric-LLL objective performs best in the low-degree regime where it outperforms the UB (as we observe in Figure 2); as the dependency degree grows, the symmetric condition expectedly becomes hard to satisfy; the asymmetric-LLL objective is the most robust in high normalized average-degree settings, outperforming the symmetric LLL and in many cases the UB (as shown in Figures 7a and 6a). As for the runtime comparison, the UB loss tends to converge more slowly in the optimization dynamics (as Figure 8 clearly depicts), suggesting that LLL-inspired losses have more favorable gradient descent dynamics. Concerning GINGAT, Table 1 shows that while its training performance was high, it did not quite generalize to unseen test instances, which can be seen as a limitation of this study.

6 Conclusion

We have proposed a self-supervised approach to SAT solving the makes use of specialized loss functions based on classic tools from probabilistic combinatorics. While our losses rely on simpler versions of the LLL, this direction opens up the possibility of employing more powerful tools from the literature like Shearer's bound and the cluster expansion lemma [Bissacot et al., 2011, Shearer, 1985]. Another important direction to explore is the role of the model and its interplay with the loss function. Even though a specialized loss might improve direct optimization results, it is unclear how a model will interact with the loss.

References

Noga Alon and Joel H. Spencer. *The Probabilistic Method*. Wiley Publishing, 4th edition, 2016. ISBN 1119061954.

Rodrigo Bissacot, Roberto FernÁndez, Aldo Procacci, and Benedetto Scoppola. An improvement of the lovász local lemma via cluster expansion. *Comb. Probab. Comput.*, 20(5):709–719, September

2011. ISSN 0963-5483. doi: 10.1017/S0963548311000253. URL `https://doi.org/10.1017/S0963548311000253`.

Paul Erdős and László Lovász. Problems and results on 3-chromatic hypergraphs and some related questions. *Coll Math Soc J Bolyai*, 10, 01 1974.

Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. In Gaël Varoquaux, Travis Vaught, and Jarrod Millman, editors, *Proceedings of the 7th Python in Science Conference*, pages 11 – 15, Pasadena, CA USA, 2008.

Nikolaos Karalias and Andreas Loukas. Erdos goes neural: an unsupervised learning framework for combinatorial optimization on graphs. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020. URL `https://proceedings.neurips.cc/paper/2020/hash/49f85a9ed090b20c8bed85a5923c669f-Abstract.html`.

Nikolaos Karalias, Joshua Robinson, Andreas Loukas, and Stefanie Jegelka. Neural set function extensions: Learning with discrete functions in high dimensions. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL `http://papers.nips.cc/paper_files/paper/2022/hash/6294a235c0b80f0a2b224375c546c750-Abstract-Conference.html`.

Richard M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972. ISBN 978-1-4684-2001-2. doi: 10.1007/978-1-4684-2001-2_9. URL `https://doi.org/10.1007/978-1-4684-2001-2_9`.

Anastasios Kyrillidis, Anshumali Shrivastava, Moshe Y. Vardi, and Zhiwei Zhang. Fouriersat: A fourier expansion-based algebraic framework for solving hybrid boolean constraints. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 1552–1560. AAAI Press, 2020. doi: 10.1609/AAAI.V34I02.5515. URL `https://doi.org/10.1609/aaai.v34i02.5515`.

Massimo Lauria. Cnfgen formula generator. URL `https://github.com/MassimoLauria/cnfgen`.

Robin A. Moser and Gábor Tardos. A constructive proof of the general lovász local lemma. *J. ACM*, 57(2), February 2010. ISSN 0004-5411. doi: 10.1145/1667053.1667060. URL `https://doi.org/10.1145/1667053.1667060`.

Daniel Selsam, Matthew Lamm, Benedikt Bünz, Percy Liang, Leonardo de Moura, and David L. Dill. Learning a SAT solver from single-bit supervision. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL `https://openreview.net/forum?id=HJMC_iA5tm`.

James B. Shearer. On a problem of spencer. *Combinatorica*, 5(3):241–245, 1985. doi: 10.1007/BF02579368.

Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL `https://openreview.net/forum?id=rJXMpikCZ`.

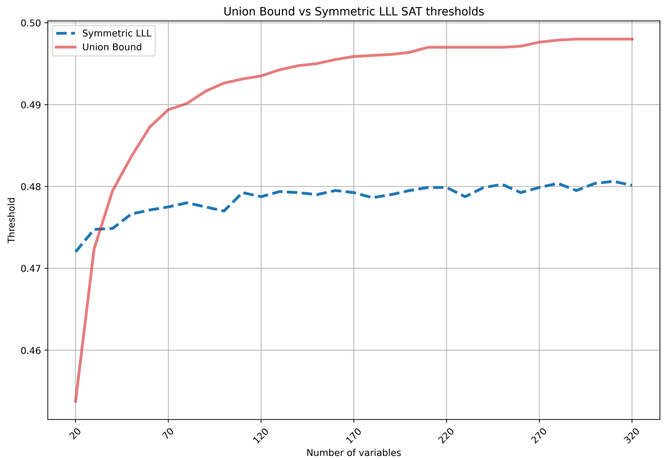
Po-Wei Wang, Priya L. Donti, Bryan Wilder, and J. Zico Kolter. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6545–6554. PMLR, 2019. URL `http://proceedings.mlr.press/v97/wang19e.html`.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=ryGs6iA5Km.

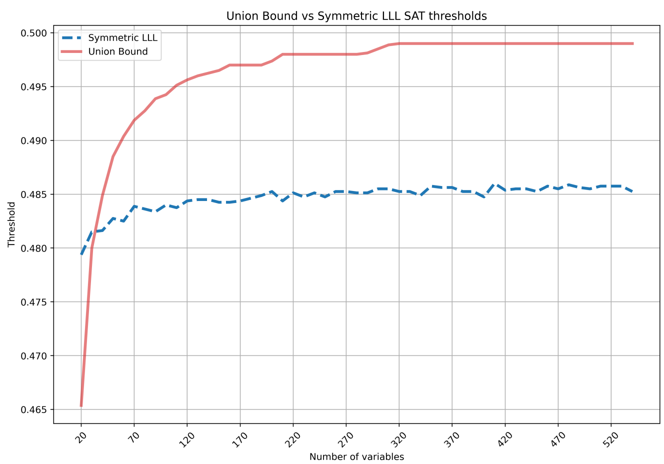
Morris Yau, Nikolaos Karalias, Eric Lu, Jessica Xu, and Stefanie Jegelka. Are graph neural networks optimal approximation algorithms? In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang, editors, *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/85db52cc08c5e00cfb1d216b1c85ba35-Abstract-Conference.html.

Emre Yolcu and Barnabas Poczos. Learning local search heuristics for boolean satisfiability. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/12e59a33dea1bf0630f46edfe13d6ea2-Paper.pdf.

A Effective entropy



(a) Average T^δ values for increasingly large random 3-CNFs of density 2.5



(b) Average T^δ values for increasingly large random 3-CNFs of density 3.5

Figure 3: Comparison of average T^δ values for random 3-CNFs of different densities.

Proof 1 (proof of Claim 1) *We design a k-CNF $\phi' := u(\phi)$ that meets the condition of the Claim. More generally, ϕ' is defined on n variables $x_1, \ldots, x_n$ (same as ϕ) and has $m = (2^k - 1)z$ clauses. We show that ϕ' has z groups $\mathcal{C}^1, \ldots, \mathcal{C}^z$ of clauses of size $2^k - 1$. Group $\mathcal{C}^r$ ($r \in [z]$) contains k variables $X^r := \{x_{(r-1)k+1}, \ldots, x_{(r-1)k+k}\}$ and $2^k - 1$ clauses involving X^r.*

Definition 3 *Let $S \in \{-1, 1\}^k$. Define:*

$$C(S, X^r) := \bigvee_{(i,s)\in \text{zip}(((r-1)k+1,\ldots,(r-1)k+k),S)} x_i^x,$$

where

$$x_i^s = \begin{cases} x_i, \text{ if } s = 1, \\ \overline{x_i}, \text{ if } s = -1. \end{cases}$$

Define each group of clauses $\mathcal{C}^r$ as

$$\mathcal{C}^r := \bigcup_{S\in\{-1,1\}^k : S \neq (-sign(p_x))_{x\in X^r}} C(S, X^r).$$

Overall, $\phi' = \left([n], \bigcup_{r\in[z]} \mathcal{C}^r\right)$*. We now show that* $p^* = (p_1, \ldots, p_n)$ *is the unique satisfying assignment of* ϕ'*.*

1. $\phi'(p^*) = 1$*: Consider an arbitrary* $r \in [z]$*. For every* $S \in \{-1,1\}^k \setminus \{(-sign(p^*_x))_{x\in X^r}\}$*, there is some* $i \in [k]$ *such that* $sign(s_i) = sign(p^*_{s_i})$*. Thus, every clause* $C \in \mathcal{C}^r$ *is satisfied.*

2. *Let* $p \in \{0,1\}^n \setminus p^*$*. Then, we show that* $\phi'(p) = 0$*. We show that* $\exists r \in [z]$ *such that* $\mathcal{C}^r$ *contains a clause violated by* p*. Let* $D = \{i \in [n] : p_i = 1 - p^*_i\}$*. Without loss of generality,* $1 \in D$*. Let* X' *be a maximal subset of* D *such that* $X' \subseteq Var(\mathcal{C}^1)$*. Without loss of generality,* $X' = [k'], k' < k$*. Now the constraint*

$$C\left((\overbrace{1,1,\ldots,1}^{k'}, \overbrace{-1,-1,\ldots,-1}^{k-k'}, X^1)\right)$$

is violated by p*.*

Observation 1 *Let* ϕ *be a* k*-CNF formula with* m *clauses. Let* p^* *be any satisfying assignment of* ϕ*. For any* $\varepsilon \in (0,1)$*, define* $p^* \oplus \varepsilon$ *as:*

$$(p^* \oplus \varepsilon)_i = \begin{cases} \varepsilon, \text{ if } p^*_i = 0 \\ \\ 1-\varepsilon, \text{ otherwise.} \end{cases}$$

Then, for every $j \in [m]$*, we have that:*

$$f_j(p^* \oplus \varepsilon) := \mathbb{P}_{x\sim(p^*\oplus\varepsilon)}\left(C_j(x) = 0\right) = \varepsilon^{c_j^{True}(p^*)} \cdot (1-\varepsilon)^{k - c_j^{True}(p^*)},$$

where $c_j^{True}(p^*) := |\{l \in C_j : l(p^*) = 1\}|$ *(number of literals that evaluate to True in clause* j *under assignment* p^**).*

Proof 2 (proof of Claim 2) *We show that for every* $z \geq 1$*,* $\sup_{p\in\mathcal{P}_{LLL}(\phi_z)} \overline{H}(p) \geq \frac{1}{3}$*. Fix an arbitrary* $z \geq 1$*. It suffices to find* $p \in \mathcal{P}_{LLL}(\phi_z)$ *with* $\overline{H}(p) \geq \frac{1}{3}$*. Let* $\varepsilon = 0.071$ *and* $p := (1-\varepsilon)\mathbf{1}^n$ *(recall that* $n = 3z$*). To show that* $p \in \mathcal{P}_{LLL}(\phi_z)$*, we need that*

$$\max_{j\in[m]} \mathbb{P}_{x\sim p}\left(C_j(x) = 0\right) \leq \frac{1}{6e},$$

since $d = 6$*. From Observation 1 and since* $\varepsilon < 0.5$*, the maximum is equal to* $\varepsilon(1-\varepsilon)^2 = 0.06127\dots \leq \frac{1}{6e} = 0.061313\dots$ *Now, as for the average entropy, we have that:*

$$\overline{H}(p) = \frac{1}{n}\sum_{i\in[n]} H(1-\varepsilon) = H(1-\varepsilon) \approx 0.369 \geq \frac{1}{3}.$$

We now consider the Union Bound. Fix an arbitrary $z \geq 1$ *and a* $p \in \mathcal{P}_{UB}(\phi_z)$*. We upper bound* $\overline{H}(p)$*. Let*

$$f_j(p) = \mathbb{P}_{x\sim p}\left(C_j(x) = 0\right), j \in [m].$$

Fix any $\delta \in \left(0, \frac{1}{2}\right)$*. Let*

$$B_{p,\delta} := \{i \in [n] : p_i \leq 1 - \delta\}$$

and

$$W_{p,\delta} := \{r \in [z] : V(\mathcal{C}^r) \cap B_{p,\delta} \neq \emptyset\}.$$

Now observe that

$$f(p) := \sum_{j\in[m]} f_j(p) \geq \delta \cdot |W_{p,\delta}|,$$

since the sum in each group of clauses is equal to $1 - p_i p_q p_l$*. Since* $f(p) < 1$ *(by assumption), we get* $|W_{p,\delta}| < \frac{1}{\delta}$*. Each group of clauses contains exactly three variables and thus* $|B_{p,\delta}| < \frac{3}{\delta}$*. Thus, we have that:*

$$\overline{H}(p) \leq \frac{1}{3z} \cdot \left(\frac{3}{\delta} + H(\delta)\cdot 3z\right) = \frac{1}{z\delta} + H(\delta).$$

Recall that the inequality above holds for any $\delta \in \left(0, \frac{1}{2}\right)$. Set $\delta \coloneqq \frac{1}{\sqrt{z}}$ (define the family only for $z > 4$ to make sure that δ is in the right range). Since p was arbitrarily chosen in $\mathcal{P}_{UB}(\phi_z)$, we have

$$\sup_{p \in \mathcal{P}_{UB}(\phi_z)} \overline{H}(p) \leq \frac{\sqrt{z}}{z} + H\left(\frac{1}{\sqrt{z}}\right).$$

Thus,

$$H^{\infty}_{UB}(\Phi) \leq \limsup_{z \to \infty} \left\{ \frac{\sqrt{z}}{z} + H\left(\frac{1}{\sqrt{z}}\right) \right\} = 0$$

and we conclude.

Proof 3 (proof of Claim 3) *We now construct an infinite family Φ_2 with $d = O(m)$ and $H^{\infty}_{LLL}(\Phi_2) = 1$. We define $\Phi_2 = (\phi_z)_{z \in \mathbb{N}}$ in the following way. Fix a constant $q \in \mathbb{N}$. The CNF formula ϕ_z has $q \cdot (z+1)$ variables and $q \cdot (z+1)$ clauses. We set*

$$\mathit{Var}(\phi_z) \coloneqq \bigcup_{i \in [q]} x_i \cup \bigcup_{(i,r) \in [q] \times [z]} y_{i,r}$$

and

$$C(\phi_z) = \bigcup_{i \in [q]} (\overline{x_i} \vee y_{i,1} \vee y_{i,2}) \cup \bigcup_{(i,r) \in [q] \times [z]} (\overline{x_i} \vee \overline{y_{i,r}} \vee y_{i,r+1}),$$

where $y_{i,z+1} \coloneqq y_{i,1}$.

First, note that the dependency graph of any ϕ_z is a disjoint union of q copies of K_{z+1} and thus every clause has degree z. Thus, $d = z = O(m)$.

Fix any $z \in \mathbb{N}$ and consider the corresponding formula ϕ_z. Define a joint distribution $p_z \in [0,1]^{q \cdot (z+1)}$ as follows:

$$\begin{cases} (p_z)_{x_i} = \frac{1}{z+1}, \forall i \in [q], \\ (p_z)_{y_{i,r}} = \frac{1}{2}, \forall (i,r) \in [q] \times [z]. \end{cases}$$

We now see that

$$f_j(p_z) = \frac{1}{4 \cdot (z+1)}, \forall j \in [m],$$

where we recall that $f_j(p_z)$ is the probability that clause j is falsified when sampling from p_z.

Thus, we have that

$$f_j(p_z) \leq \frac{1}{ed} = \frac{1}{ez}$$

and thus $p_z \in \mathcal{P}_{LLL}(\phi_z)$. It now suffices to show that

$$\lim_{z \to \infty} \frac{1}{q \cdot (z+1)} H(p_z) = 1.$$

The average joint entropy is computed as

$$\begin{aligned} \frac{1}{q \cdot (z+1)} H(p_z) &= \frac{1}{q \cdot (z+1)} \left(\sum_{i \in [q]} H((p_z)_{x_i}) + \sum_{(i,r) \in [q] \times [z]} H((p_z)_{y_{i,r}}) \right) \\ &= \frac{1}{q \cdot (z+1)} \left(q \cdot H\left(\frac{1}{z+1}\right) + qz \cdot H\left(\frac{1}{2}\right) \right) \\ &= \frac{1}{z+1} H\left(\frac{1}{z+1}\right) + \frac{qz}{qz+q} \xrightarrow{z \to \infty} 1. \end{aligned}$$

Proof 4 (proof of Claim 4) *Fix some $q = O(1)$. We define $\Phi_3 = (\phi_z)_{z \in \mathbb{N}_{\geq 5}}$ in the following way:*

$$\mathit{Var}(\phi_z) \coloneqq \bigcup_{(i,r) \in [q] \times [z]} x_{i,r}$$

and
$$C(\phi_z) = \bigcup_{i\in[q]} \bigvee_{r\in[z]} \overline{x_{i,r}} \cup \bigcup_{(i,r)\in[q]\times[z]} (\overline{x_{i,r}})\,.$$
We clearly have that $n = qz$, $m = q\cdot(z+1)$ and $d = z = O(m)$.

We start with the Union Bound. Fix any $z \in \mathbb{N}_{\geq 5}$ and any sequence $(p_z)_z$ such that $p_z \in \mathcal{P}_{UB}(\phi_z), \forall z$. We show that
$$\lim_{z\to\infty} \frac{1}{qz} H(p_z) = 0.$$
Since, $p_z \in \mathcal{P}_{UB}(\phi_z)$, we know that
$$\sum_{j\in[m]} f_j(p_z) < 1.$$
Set $\delta := \frac{1}{\sqrt{z}}$ and define the set of "middle" variables
$$B_{p_z,\delta} := \{(i,r)\in[q]\times[z] : p_{i,r}\in[\delta, 1-\delta]\}\,.$$
We compute
$$\sum_{j\in[m]} f_j(p_z) = \sum_{(i,r)\in[z]\times[z]} p_{i,r} + \sum_{i\in[q]}\prod_{r\in[z]} p_{i,r} \geq |B_{p_z,\delta}|\cdot\delta.$$
Thus, we get that $|B_{p_z,\delta}| < \frac{1}{\delta}$ since by assumption we have that $p_z \in \mathcal{P}_{UB}(\phi_z)$.

To conclude, we write
$$\frac{1}{qz}H(p_z) \leq \frac{1}{qz}\left(\frac{1}{\delta} + qz\cdot H(\delta)\right) = \frac{1}{qz\delta} + H(\delta) \overset{z\to\infty}{\longrightarrow} 0.$$
We now consider the symmetric LLL. Fix any $z \in \mathbb{N}_{\geq 5}$ and any sequence $(p_z)_z$ such that $p_z \in \mathcal{P}_{LLL}(\phi_z), \forall z$. We show that
$$\lim_{z\to\infty} \frac{1}{qz} H(p_z) = 0.$$
The LLL conditions require:
$$\begin{cases} p_{i,r} \leq \frac{1}{ez}, \quad \forall (i,r)\in[q]\times[z], \\ \prod_{r\in[z]} p_{i,r} \leq \frac{1}{ez}, \quad \forall i\in[q]. \end{cases}$$
Thus,
$$\frac{1}{qz}H(p_z) \leq \frac{1}{qz}\left(qz\cdot H\left(\frac{1}{ez}\right)\right) = H\left(\frac{1}{ez}\right) \overset{z\to\infty}{\longrightarrow} 0.$$
Finally, we consider the asymmetric LLL. It suffices to show the following: let $\varepsilon \in (0, 1/4]$. Then, for every $\phi_z \in \Phi_3$, there exists some $p \in [0,1]^{qz}$ such that:

1. $\frac{1}{qz}H(p) = \varepsilon$
2. $\exists \mu \in (0,1)^{q\cdot(z+1)} : (p,\mu) \in \mathcal{P}_{GLLL}(\phi_z)$.

Fix any $\varepsilon \in (0, 1/4]$. Let $\delta \in (0,1)$ be such that $\delta(1-\delta) = \varepsilon$. Fix any $\phi_z \in \Phi_3$. Now define
$$\begin{cases} p_z = \delta(1-\delta)\cdot\mathbf{1}^{qz}, \\ \mu_z = \delta\cdot\mathbf{1}^{q\cdot(z+1)}. \end{cases}$$
We now have that:
$$\begin{cases} \delta(1-\delta) \leq \delta(1-\delta), \\ (\delta(1-\delta))^z \leq \delta(1-\delta)^z. \end{cases}$$
and thus $(p,\mu) \in \mathcal{P}_{GLLL}(\phi_z)$. We also get that $\frac{1}{qz}H(p) = \delta(1-\delta) = \varepsilon$.

B GINGAT and Adam results

Table 2: Key parameters of the GINGAT model

Parameter	**Description**
Hidden dimensions	h_{vc} and h_{cv} denote message dimensions from variables to clauses and clauses to variables, respectively
Input feature dimensions	d_v and d_c for variable and clause nodes
Number of layers	L_{gin} GIN layers and L_{gat} GATv2 layers
Activation function	$\sigma(\cdot)$, typically ReLU
Aggregation method	aggr, e.g., sum or mean
Attention heads	H in GATv2 layers
Dropout	probability p
Residual connections, concatenation, graph normalization	optional design choices
Epsilon parameter	ϵ, trainable in GIN layers to control self-loop contribution

Table 3: GNN training instance generators

CNF types	Graph Generators	Parameters
binaryClique	gnm random graph	m: 5, k: 3
coloring	barabasi albert graph	m: 2, col: 3
	expected degree graph	a: 5.5, min: 4, max: 10, col: 3
	gnp random graph	p: 0.08, col: 3
	random regular graph	d: 6, col: 4
	watts strogatz graph	k: 4, p: 0.3, col: 3
evenColoring	random regular graph	d: 4
	watts strogatz graph	k: 3, p: 0.3
graphOrdering	expected degree graph	a: 5, min: 2, max: 4
	powerlaw cluster graph	m: 2, p: 0.05
	watts strogatz graph	k: 4, p: 0.3
perfectMatching	expected degree graph	a: 4.5, min: 3, max: 4
	expected degree graph	a: 5.5, min: 3, max: 10
	gnp random graph	p: 0.1
	powerlaw cluster graph	m: 4, p: 0.05
	watts strogatz graph	k: 4, p: 0.2

Table 4: Best hyperparameters for training the GINGAT model

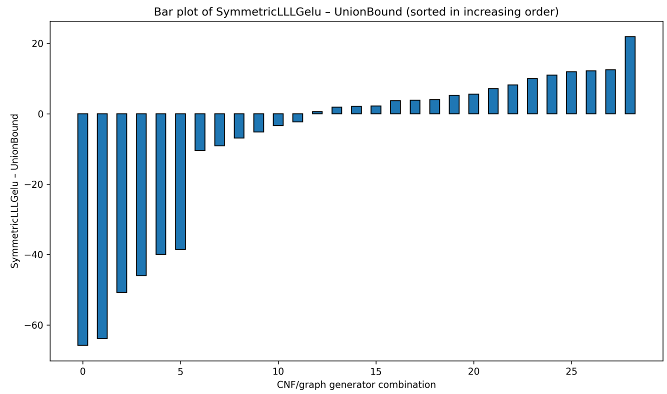
Parameter	**Value**
Number of epochs	200
Learning rate	0.001
Number of samples	1024
Optimizer	Adam or AdamW
Weight decay	0.0003
Variable initialization	Kaiming
Clause initialization	Kaiming
Hidden dimension (variable $\rightarrow$ clause)	128
Hidden dimension (clause $\rightarrow$ variable)	128
Variable feature dimension	128
Clause feature dimension	128
Output dimension	1
Number of GIN layers	10
Number of GAT layers	1
Activation function	ReLU
Aggregation method	Sum
Number of attention heads	8
Dropout rate	0.1
Add self-loops	False
Residual connections	True
Concatenate heads	True
Graph normalization	True
Epsilon in GIN	0
Trainable epsilon	True

Table 5: Performance of pure Adam across CNF types and loss functions

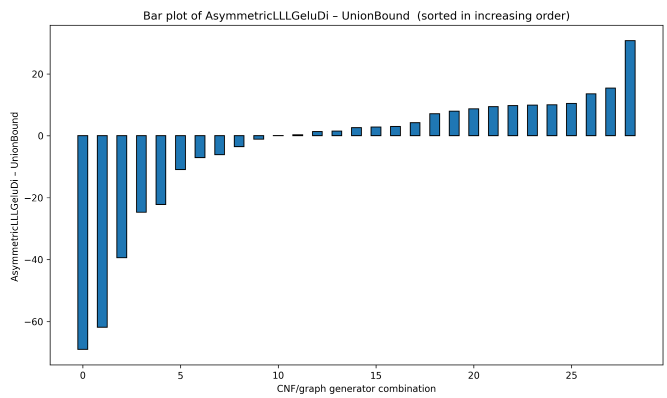
CNF type	**AsymmetricLLLGeluDi**	**SymmetricLLLGelu**	**UnionBound**	**Total**
binaryClique	61%	19%	83%	53%
coloring	67%	67%	63%	66%
evenColoring	83%	62%	75%	72%
graphOrdering	32%	26%	28%	29%
perfectMatching	31%	48%	47%	44%
Total	**57%**	**52%**	**57%**	**55%**

Table 6: Detailed Adam performance comparison across CNF types and graph generators

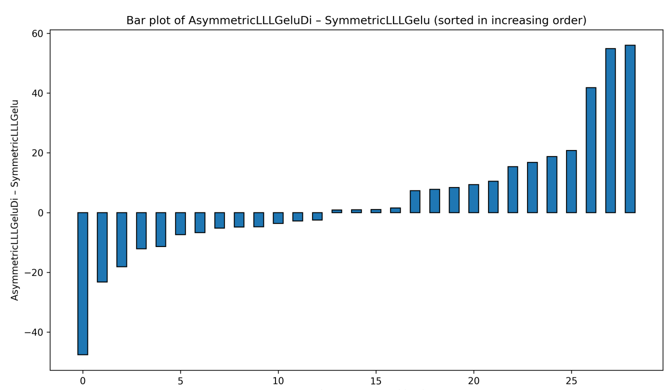
CNF type	Graph generator	Asymmetric LLL GeluDi	Symmetric LLL Gelu	Union Bound	Asym LLL GeluDi vs. UB	Sym LLL Gelu vs. UB	Asym vs. Sym
binary Clique	gnm random graph	61.03%	19.24%	83.18%	-22.15%	-63.94%	41.79%
coloring	barabasi albert graph	39.75%	44.53%	46.83%	-7.08%	-2.30%	-4.78%
	expected degree graph	89.10%	87.58%	80.40%	8.70%	7.18%	1.52%
	gnp random graph	61.64%	53.25%	51.13%	10.51%	2.13%	8.39%
	powerlaw cluster graph	40.29%	51.67%	46.43%	-6.14%	5.24%	-11.38%
	random degree sequence graph	70.86%	76.11%	70.56%	0.31%	5.56%	-5.25%
	random regular graph	62.59%	70.00%	66.11%	-3.52%	3.89%	-7.41%
	watts strogatz graph	68.07%	71.70%	67.95%	0.12%	3.75%	-3.63%
even Coloring	barabasi albert graph	100.00%	44.00%	90.00%	10.00%	-46.00%	56.00%
	expected degree graph	99.75%	89.23%	98.33%	1.42%	-9.10%	10.52%
	gnp random graph	74.38%	59.00%	99.00%	-24.63%	-40.00%	15.38%
	powerlaw cluster graph	80.74%	25.83%	91.67%	-10.93%	-65.83%	54.91%
	random degree sequence graph	81.25%	62.50%	65.83%	15.42%	-3.33%	18.75%
	random regular graph	67.10%	46.33%	36.33%	30.76%	10.00%	20.76%
	watts strogatz graph	82.07%	84.86%	72.64%	9.43%	12.22%	-2.79%
graph Ordering	barabasi albert graph	32.27%	22.92%	33.33%	-1.06%	-10.42%	9.36%
	expected degree graph	16.79%	15.94%	13.75%	3.04%	2.19%	0.85%
	gnp random graph	61.29%	53.49%	58.67%	2.62%	-5.18%	7.80%
	powerlaw cluster graph	45.35%	28.54%	35.42%	9.93%	-6.88%	16.81%
	random degree sequence graph	17.50%	20.00%	15.94%	1.56%	4.06%	-2.50%
	random regular graph	22.00%	21.04%	19.17%	2.83%	1.88%	0.96%
	watts strogatz graph	24.00%	16.67%	16.04%	7.96%	0.62%	7.33%
perfect Matching	barabasi albert graph	1.00%	19.17%	70.00%	-69.00%	-50.83%	-18.17%
	expected degree graph	25.00%	29.81%	17.88%	7.12%	11.92%	-4.81%
	gnp random graph	20.63%	68.21%	60.00%	-39.38%	8.21%	-47.59%
	powerlaw cluster graph	23.97%	47.23%	85.80%	-61.84%	-38.57%	-23.27%
	random degree sequence graph	84.38%	83.33%	70.83%	13.54%	12.50%	1.04%
	random regular graph	43.75%	50.50%	39.50%	4.25%	11.00%	-6.75%
	watts strogatz graph	32.31%	44.44%	22.51%	9.80%	21.93%	-12.14%



(a) Symmetric LLL vs Union Bound



(b) Asymmetric LLL vs Union Bound



(c) Asymmetric LLL vs Symmetric LLL

Figure 4: Pair-wise comparisons between loss functions across different CNF/graph generator combinations.

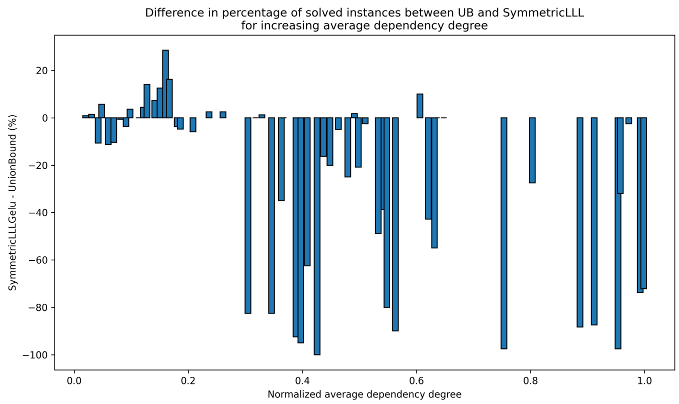
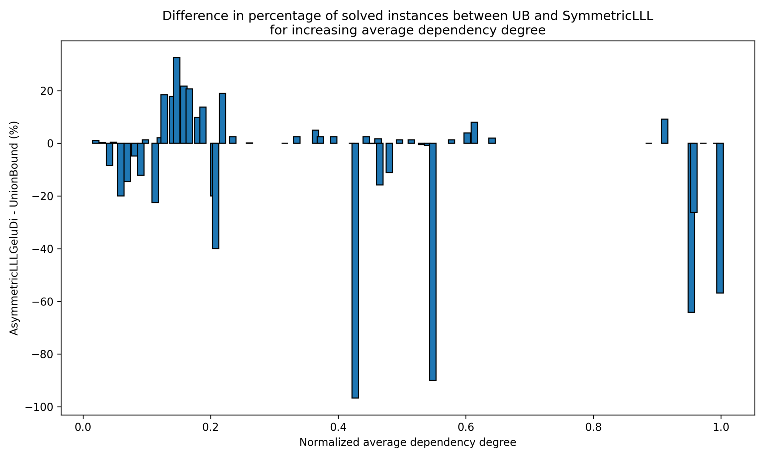
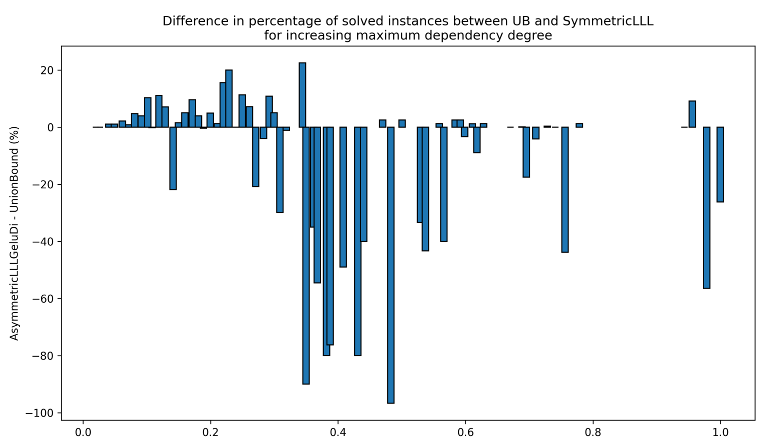


Figure 5: Symmetric LLL vs Union Bound for increasing average dependency degree

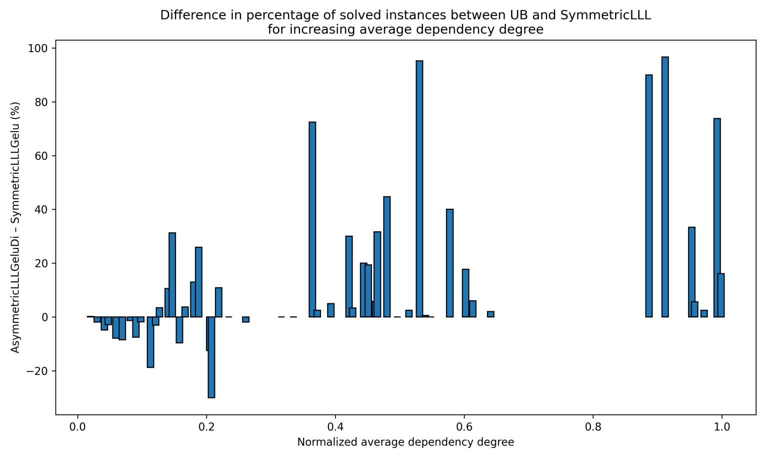


(a) Asymmetric LLL vs Union Bound for increasing average dependency degree

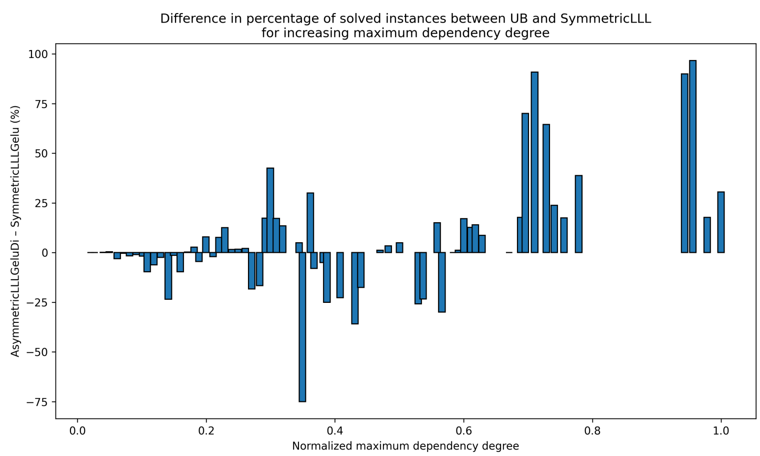


(b) Asymmetric LLL vs Union Bound for increasing maximum dependency degree

Figure 6: Asymmetric LLL vs Union Bound

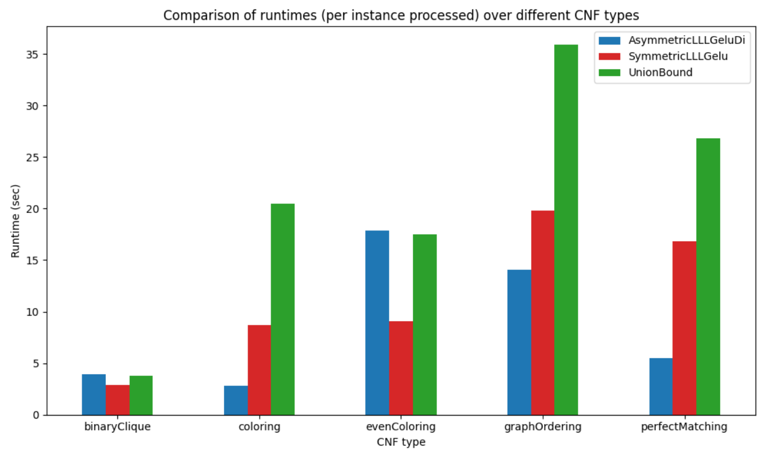


(a) Asymmetric LLL vs Symmetric LLL for increasing average dependency degree

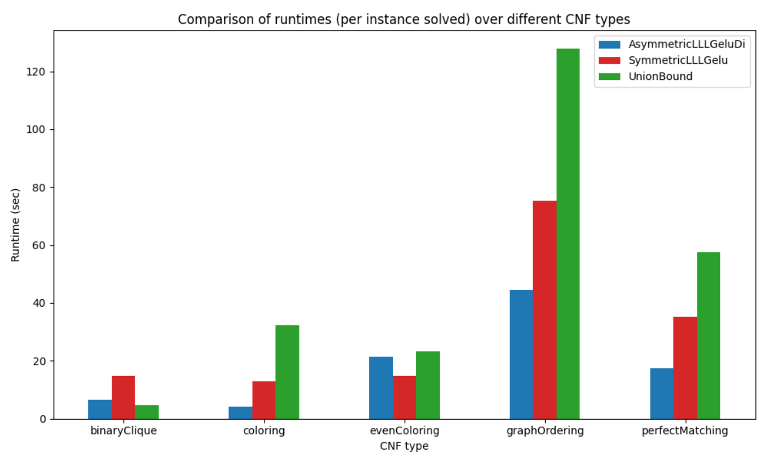


(b) Asymmetric LLL vs Symmetric LLL for increasing maximum dependency degree

Figure 7: Asymmetric LLL vs Symmetric LLL



(a) Processing runtime



(b) Solving runtime

Figure 8: Runtime comparison between different loss functions

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.

- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [No]

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.

- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).

- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: There is no direct impact.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.

- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.