

Graph-Based Attention for Differentiable MaxSAT Solving (Extended Abstract)*

Sota Moriyama^{1,2}, Katsumi Inoue²

¹The Graduate University for Advanced Studies, SOKENDAI

²National Institute of Informatics

{sotam, inoue}@nii.ac.jp

Abstract

The use of deep learning to solve fundamental AI problems such as Boolean Satisfiability (SAT) has been explored recently to develop robust and scalable reasoning systems. This work advances such neural-based reasoning approaches by developing a new Graph Neural Network (GNN) to differentially solve (weighted) Maximum Satisfiability (MaxSAT). To this end, we propose SAT-based Graph Attention Networks (SGATs) as novel GNNs that are built on t-norm based attention and message passing mechanisms, and structurally designed to approximate greedy distributed local search. To demonstrate the effectiveness of our model, we develop a local search solver that uses SGATs to continuously solve any given MaxSAT problem. Experiments on (weighted) MaxSAT benchmark datasets demonstrate that SGATs significantly outperform existing neural-based architectures, and achieve state-of-the-art performance among continuous approaches, highlighting the strength of the proposed model. This paper is an extended abstract of (Moriyama and Inoue 2025).

1 Introduction

Neuro-symbolic AI aims to develop robust and scalable reasoning systems by combining the strengths of both symbolic logic and neural networks. Maximum Satisfiability (MaxSAT), an optimization generalization of Boolean Satisfiability (SAT), has been viewed as a promising approach towards accomplishing various neuro-symbolic tasks. In such cases, continuous optimization based MaxSAT solvers have been used as a key building block for the reasoning system. Existing continuous approaches include the use of Semidefinite Programming (SDP) (Wang and Kolter 2019), Fourier analysis (Kyrillidis et al. 2020), and Graph Neural Networks (GNNs) (Liu et al. 2021). The latter, however, remains largely underexplored.

In this paper, we present SAT-based Graph Attention Networks (SGATs) as novel GNNs crafted for solving MaxSAT, which are the first GNNs to be able to handle weighted MaxSAT problems. SGATs are composed of GNN layers with novel t-norm based attention, where attention mechanisms operate on values computed using t-norms. When a clause is unsatisfied by a candidate assignment, the clause

*This work has been supported by JST CREST Grant Number JPMJCR22D3, JSPS KAKENHI Grant Number JP25K03190 and Grant-in-Aid for JSPS Fellows Grant Number JP26KJ1236, Japan.

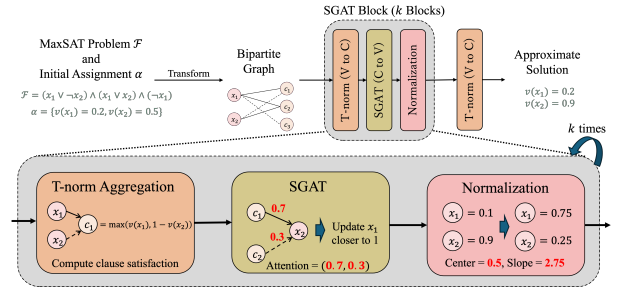


Figure 1: Architectural Diagram of SGATs. Each SGAT block is composed of a t-norm layer, SGAT layer, and a normalization layer.

node sends messages to all its connected variable nodes, requesting them to update their values in parallel towards satisfying the clause. The attention mechanism then computes priorities to decide which variable nodes should change their values to maximally satisfy those requests from clauses, allowing the model to learn which clauses to focus on. Intuitively, this can be regarded as learning a general heuristic that can be applied to a variety of MaxSAT problems.

2 SAT-based Graph Attention Network

In this section, we present key building blocks for constructing SGATs as shown in Figure 1. SGATs are composed of three main components: (i) T-norm layers that compute clause valuations $v(C)$ from the valuations of connected variables $v(x)$, (ii) SGAT layers that update the valuations of variables based on the valuations of connected clauses, and (iii) a normalization layer that ensures the valuations of all variables are sufficiently spread out. A single block of SGATs approximates a distributed local search step, with the attention mechanism learning which clauses to focus on, essentially serving as a heuristic.

Graph Representation of MaxSAT Problems. Using normalized weights $w_j^{\text{norm}} = \frac{w_j}{\max_k w_k}$, we define the edge feature $e_{i,j+n}$ between variable x_i and clause C_j as follows:

$$e_{i,j+n} = \begin{cases} \left(\frac{1}{|\mathcal{N}_j^+|}, \frac{1}{|\mathcal{N}_j^-|}, 0, 0 \right) \cdot w_j^{\text{norm}} & \text{if } i \in \mathcal{N}_j^+ \\ \left(0, 0, \frac{1}{|\mathcal{N}_j^+|}, \frac{1}{|\mathcal{N}_j^-|} \right) \cdot w_j^{\text{norm}} & \text{if } i \in \mathcal{N}_j^- \end{cases},$$

Here, \mathcal{N}_j^+ and \mathcal{N}_j^- correspond to the set of variables with positive and negative polarities included in clause C_j . Note that when the denominator is 0, we set the corresponding value to 0.

T-norm Layer. To evaluate the degree to which each clause is satisfied, we use t-norms T , which generalize the logical conjunction \wedge to the real-valued domain. We specifically focus on three well-known t-norms Gödel, Product and Łukasiewicz. As each clause is a disjunction of literals, we compute valuations by negating the conjunction of each negated literal. Using strong negation defined as $1 - a$ for $a \in [0, 1]$, we set $\tilde{v}_{ir} = 1 - v(x_{ir})$ if $l_{ir} = x_{ir}$ and $\tilde{v}_{ir} = v(x_{ir})$ if $l_{ir} = \neg x_{ir}$ for a clause $C_i = l_{i1} \vee \dots \vee l_{i\ell_i}$, and define

$$v_{\star}(C_i) = 1 - T_{\star}(\tilde{v}_{i1}, \dots, \tilde{v}_{i\ell_i}), \quad \text{where } \star \in \{G, P, L\}.$$

SGAT Layer. To efficiently learn which variables to update, we employ an attention mechanism and a message-passing function that update the valuations of variables based on the valuations of the clauses to which they are connected. Specifically, we define the update message m_{ij}^U and attention message m_{ij}^A between variable x_i and clause C_{j-n} as follows:

$$m_{ij}^U = \begin{cases} v(x_i) + (1 - v(C_{j-n})) & \text{if } i \in \mathcal{N}_{j-n}^+ \\ v(x_i) - (1 - v(C_{j-n})) & \text{otherwise} \end{cases}$$

$$m_{ij}^A = \begin{cases} m_{ij}^U & \text{if } i \in \mathcal{N}_{j-n}^+ \\ 1 - m_{ij}^U & \text{otherwise} \end{cases}$$

The update messages m_{ij}^U represent the valuation the clause requests the variable to move toward. The attention messages m_{ij}^A represent the strength of this request by flipping the valuation for negatively appearing literals, aligning all messages in a common direction regardless of polarity. Using these messages, the variable updates are redefined as:

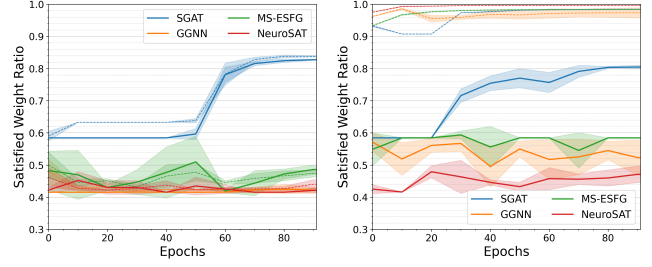
$$\epsilon_{ij} = \text{LeakyReLU}(\mathbf{a}^T [\mathbf{W}m_{ij}^A \parallel \mathbf{W}_e e_{ij}])$$

$$\alpha_{ij} = \frac{\exp(\epsilon_{ij})}{\sum_{k \in \mathcal{N}_i} \exp(\epsilon_{ik})}$$

$$v'(x_i) = \sum_{j \in \mathcal{N}_i} \alpha_{ij} m_{ij}^U$$

From the equation, as long as the input valuations $v(x_i) \in [0, 1]$, $\alpha_{ij} \in [0, 1]$, and $\sum_{j \in \mathcal{N}_i} \alpha_{ij} = 1$, the outputs $v'(x_i) \in [0, 1]$ are guaranteed to hold. This eliminates the need for additional activation functions after each layer, simplifying the computation of both variable assignments and the loss function.

Normalization Layer. To prevent feature values from being overly concentrated or dispersed, we apply a normalization layer based on the sigmoid function. The function has two parameters: γ a parameter for controlling the spread of the values, and $k \in [-1, 1]$ a parameter for controlling the center of the values.



(a) Trained with **MS2018(2MB)**. (b) Trained with **SR(U(40,200))**.

Figure 2: Comparative analysis of model architectures, with different training datasets. Test sets for both were **MS2018(2MB)**. Solid: Test, Dashed: Train.

3 Experiments

We conduct experiments to answer the following questions regarding SGATs: **Q1**) Are SGATs better than existing GNN architectures? and **Q2**) How do SGATs compare with other continuous solvers?

Experiment 1. We evaluate the satisfied weight ratio, which represents the ratio of the total weight of satisfied clauses to the total weight of all clauses, with a higher score indicating a strictly better solution. Experiments on benchmark instances from the 2018 MaxSAT Evaluations (**MS2018**) show that SGATs completely outperform existing models, both when trained on random instances (**SR(U(40,200))**) and MaxSAT evaluation instances, as shown in Figure 2.

Experiment 2. We develop a local search based solver, LS-GNN, that continuously optimizes SGATs, and compare it with existing continuous optimization solvers. Comparisons on MaxSAT Evaluations 2020-2024 show that LS-SGAT performs significantly better than existing approaches, in both weighted and unweighted settings.

Overall, the results show that SGATs are able to learn to solve practical MaxSAT instances better than existing neural architectures, and when integrated into a simple local search framework, they are able to outperform existing continuous optimization solvers by a large margin.

References

- Kyrillidis, A.; Shrivastava, A.; Vardi, M. Y.; and Zhang, Z. 2020. FourierSAT: A Fourier Expansion-Based Algebraic Framework for Solving Hybrid Boolean Constraints. In *AAAI 2020*, 1552–1560. AAAI Press.
- Liu, M.; Jia, F.; Huang, P.; Zhang, F.; Sun, Y.; Cai, S.; Ma, F.; and Zhang, J. 2021. Can Graph Neural Networks Learn to Solve MaxSAT Problem? *CoRR* abs/2111.07568.
- Moriyama, S., and Inoue, K. 2025. Graph-Based Attention for Differentiable MaxSAT Solving. In *NeurIPS 2025*.
- Wang, P., and Kolter, J. Z. 2019. Low-Rank Semidefinite Programming for the MAX2SAT Problem. In *AAAI 2019*, 1641–1649. AAAI Press.