NP-hardness of Jailbreak Attack Optimization

In this manuscript, we demonstrate the NP-hardness of the jailbreak attack optimization problem. We begin by defining the computational model. Let $f : \{0,1\}^m \to \mathbb{R}$ be a neural network with m-binary inputs and a canonical feed-forward structure comprising a single hidden layer. The network is formally expressed as:

$$f(x) = W_2\sigma(W_1x + b_1) + b_2$$

where $W_1 \in \mathbb{R}^{n\times m}, W_2 = \mathbb{R}^{1\times n}$ are the weight matrices between the layers, σ denotes an activation function, $b_1 \in \mathbb{R}^n$ and $b_2 \in \mathbb{R}$ is the bias term. In other words, f is the function composition of two affine transformations and an activation layer. Notice that any composition of two consecutive affine transformations is still an affine transformation.

The decision problem under consideration is formulated as an existential problem:

$$\forall a \in \mathbb{R}.\exists x \in \{0,1\}^n.f(x) \leq a. \tag{1}$$

This formulation aims to determine, for any given threshold a and two-layer binary neural network, whether there exists an input x such that $f(x) \leq a$. We demonstrate the NP-hardness of this problem in Section 1. Subsequently, in Section 2, we elucidate how this formalization (Equation (1)) encapsulates the LLM jailbreak optimization problem.

1 Hardness Reduction

We prove the NP-hardness of the decision problem Equation (1) via a reduction from the Boolean satisfiability (SAT) problem. This reduction is constructed such that solving the decision problem is equivalent to determining the satisfiability of the given SAT formula. Consequently, if one can resolve the decision problem, one can, by extension, solve the corresponding SAT instance.

We define a 3-Conjunctive Normal Form (3CNF) instance ϕ as follows: Let $X_1, \ldots, X_m$ be Boolean variables. A literal L_i is defined as either X_i or its negation $\neg X_i$. A 3CNF instance

ϕ is constructed as a conjunction of clauses: $C_1 \wedge \ldots \wedge C_k$, where each clause C_j is a disjunction of three literals. To differentiate between the 3CNF instance and its neural network representation, we employ uppercase letters for 3CNF components and lowercase letters for their corresponding neural network constructs. The Cook–Levin theorem establishes that deciding the satisfiability of a 3CNF formula is NP-hard [Karp, 1972].

Simulation of 3CNF To simulate logical operations within the neural network framework, we employ a gadget similar to that proposed by Wang et al. [2022]. This approach utilizes common activation functions such as ReLU, sigmoid, and tanh to approximate a step function, defined as:

$$\text{step}(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases} \tag{2}$$

The step function, being discrete-valued, serves as a fundamental component for mathematical analysis and simulation of other discrete functions. Its versatility in neural networks allows for easy scaling and shifting through weight multiplication or bias term addition, enabling the replacement of the constants 0 and 1 in Equation (2) with any real numbers.

Wang et al. [2022] demonstrated how common activation functions can approximate the step function. For instance, $\text{ReLU}(nx) - \text{ReLU}(nx - 1)$ converges to $\text{step}(x)$ as $n \to \infty$. For our reduction, we define a step-like function:

$$s(x) = \text{ReLU}(x) - \text{ReLU}(x - 1) = \begin{cases} 1, & x \geq 1 \\ 0, & x \leq 0 \\ x, & 0 < x < 1 \end{cases} \tag{3}$$

Using s, a 3CNF formula can be simulated with a single hidden layer neural network. For each variable x, we introduce an input node x and construct an affine transformation: two nodes $y, \bar{y}$ where $y = x$ and $\bar{y} = 1 - x$, simulating literals X and $\neg X$. Let l_i represent y_i or $\bar{y}_i$, corresponding to literal L_i in the SAT formula. For each disjunction $C = L_1 \vee L_2 \vee L_3$ in the CNF formula, we define it as $l_1 + l_2 + l_3$ and clip the value using the step-like function s. Utilizing this function, we can evaluate the truth value of clause C. If any literal is satisfied, then $l_1 + l_2 + l_3 \geq 1$, resulting in $s(l_1 + l_2 + l_3) = 1$; otherwise, $s(l_1 + l_2 + l_3) = 0$. We denote the output of the node corresponding to each clause C_i as c_i.

To simulate the conjunction operation, we sum all clause gates and negate the result: $-\sum_{i=1}^{k} c_i$. Consequently, determining the satisfiability of the 3CNF formula is equivalent to solving the following decision problem for this one-layer network:

$$\exists x. f(x) \leq -k.$$

Proposition 1.1. *For any 3*CNF *formula ϕ with k clauses, there exists a two-layer neural network $f : \{0,1\}^n \to \mathbb{R}$ such that ϕ is satisfiable if and only if $\exists x$ such that $f(x) \leq -k$.*

Proof. We have described the construction of the neural network as above. Now we need to show it indeed captures the computation of the 3CNF formula.

Now if ϕ is satisfiable, then there exists a truth assignment of $X_1, \ldots, X_m$ to satisfy ϕ. Let $x_i = 1$ if X_i is true and $x_i = 0$ if X_i is false in this assignment. By the construction of the neural network, the output of s is also true for each step-like function. As a result, $-\sum_{i=1}^{k} c_i = -k$.

Now if $\exists x \in \{0,1\}^m$ such that $f(x) \leq -k$. Choose a truth assignment for $X_1, \ldots, X_m$ based on x_i's values. Because $f = -\sum_{i=1}^{k} c_i$ and $0 \leq c_i \leq 1$, then each $c_i = 1$. By construction, $l_1 + l_2 + l_3 \geq 1$ for this activation node c_i. Because x_i is discrete-valued, $l_i \in \{x_i, 1 - x_i\}$, then $l_i \in \{0, 1\}$. $l_1 + l_2 + l_3 \geq 1$ implies that at least one of l_1, l_2, l_3 equals 1. Therefore, by the truth assignment of X_j, the corresponding literal is also satisfied. Because each clause has a true assignment, then ϕ is satisfied. □

The aforementioned neural network construction employs a single-layer activation function: s, comprising two ReLU functions. Wang et al. [2022] demonstrated the feasibility of approximating the step function using common activation functions. Consequently, this construction is applicable to any single activation layer network utilizing standard activation functions. Given the NP-hardness of the 3SAT problem, it logically follows that the neural network decision problem, as defined in Equation (1), is NP-hard.

Corollary 1.2. *The decision problem in Equation* (1) *is* NP*-hard.*

2 Jailbreak Attack Formalization

We posit that the decision problem in Equation (1) effectively encapsulates the LLM jailbreak attack optimization problem. Informally, this problem seeks to minimize the jailbreak loss for all given possible suffixes.

In the context of optimization-based jailbreak attack algorithms, the objective is to minimize $L(M(\langle x, s \rangle))$, where M represents an LLM, x denotes the prompt, and L is the entropy loss between the logits and an affirmative prefix. The generalizability of jailbreak algorithms to minimize the jailbreak loss extends to minimizing $f(x)$ for $x \in \{0,1\}^n$ and a two-layer network f, given the absence of assumed LLM structures or input prompts. The discrepancy between $f \in \mathbb{R}$ and the entropy loss in jailbreak attacks can be reconciled through a network with multiple outputs, where one output corresponds to f and the rest are constants, establishing a bijective relationship between f and the jailbreak loss. Given that the cross-entropy

between a logit $(y_1, \ldots, y_n)$ and a target class j is:

$$\log \frac{\exp(y_j)}{\sum_{i=1}^{n} \exp(y_i)}.$$

We can construct a multiple output network F, where $F(x)_j = f(x)$ and the remaining $F(x)$ are constants. Assuming the one-hot encoding corresponds to the first token (class j), the cross-entropy becomes:

$$\log \frac{\exp(f(x))}{\exp(f(x)) + C},$$

for some $C > 0$. This function is bijective and monotonically increasing. By fixing the remaining affirmative prefix tokens' logits as constants, minimizing the prefix loss is equivalent to minimizing $f(x)$, demonstrating that a jailbreak algorithm capable of minimizing the loss can also minimize $f(x)$. Consequently, if the minimization of f is solvable, the existential problem for any threshold $a \in \mathbb{R}$ can also be resolved by querying the minimum of $f(x)$.

References

R. 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`.

Z. Wang, A. Albarghouthi, G. Prakriya, and S. Jha. Interval universal approximation for neural networks. *Proc. ACM Program. Lang.*, 6(POPL), Jan. 2022. doi: 10.1145/3498675. URL `https://doi.org/10.1145/3498675`.