$l_{ij} \in \{0,1\}$ there is a corresponding input to the neural net $x_{ij} \in \{-1,1\}$ such that $l_{ij} \Leftrightarrow x_{ij} = 1 \wedge \overline{l_{ij}} \Leftrightarrow x_{ij} = -1$. For each input variable x_{ij} the weight of the neuron connection is 1 if the propositional variable l_{ij} appears as a positive literal in the C_i clause and -1 if it appears as a negative literal $\overline{l_{ij}}$ in C_i.

For every clause C_i we construct a *disjunction gadget*, a perceptron equivalent function to the OR gate (Figure 5). Given m inputs $x_{i1}, x_{i2}, \ldots x_{im} \in \{-1,1\}$, the disjunction gadget determines the output of neuron q_i. The output is the linear layer is $t_i = \sum_{j=1}^{m} w_j \cdot x_{ij} + m$. The output neuron q_i is 1 if the activation function $\texttt{sign}(t_i)$ returns 1. Namely, the output is 1 only if at least one literal is true, i.e., not all $w_j \cdot x_{ij}$ terms evaluate to -1. Notice that we only need $m+2$ neurons (m for the inputs and 2 for the intermediate outputs) for each clause C_i with m literals. Next, we introduce the *conjunction gadget* which, given n inputs $q_1, \ldots, q_n \in \{-1,1\}$ outputs $y = 1$ only if $q_1 + q_2 + \ldots + q_n \geq n$ (Figure 6). The linear layer's output is $t' = \sum_{i=1}^{n} w_i \cdot q_i - n$ over which we apply the sign activation function. The output of this gadget, $y = \sum_{i=1}^{n} w_i \cdot q_i \geq n$, is 1 if all of the variables q_i are 1, i.e., if all the clauses are satisfied.Notice that if we consider the batch normalization a transformation over t_i that returns t_i, we can obtain a binarized neural network f.

If the output of f on inputs $\mathbf{x}$ is 1 the formula F is SAT, otherwise it is UNSAT. Moreover, the binarized neural network constructed for binary input vectors of size $m \times n$ outputs $y = 1$ for every satisfying assignment τ of the formula F, i.e., $f(\tau(\mathbf{x})) = 1$. Given a procedure #SAT(F) that accepts formula F and outputs a number r which is the number of satisfying assignments, it will also compute the number of inputs for which the output of the BNN is 1. Specifically, we can construct a quantitative verifier for the neural net f and a specification $\varphi(\mathbf{x}, y) = (y = N(\mathbf{x})) \wedge y = 1$ using #SAT(F).

Reduction is polynomial. The size of the formula F is the size of the input $\mathbf{x}$ to the neural net, i.e., $m \cdot n$. The neural net has $n+1$ perceptrons (n for each disjunction gadget and one for the conjunction gadget).

□