SAT Solver Implementations Analysis

Carlo Kohmann

University of Milan

Mathematical Logic

Goals

- Investigate the extent to which NLP-based models can be employed to solve problems in mathematical logic.
- Focus specifically on the SAT problem for propositional logic formulae.

$$\text{SAT}(F) \iff \exists \tilde{v} : \mathcal{F}_L \to \{0,1\} \text{ s.t. } \tilde{v}(F) = 1$$

What is Needed for the Analysis?

- A dataset of propositional well-formed formulae (WFF), split into training and test sets.
- Machine learning models capable of being evaluated on this task.

Dataset Generation

- All formulae are generated in WFF format.
- Perfectly balanced.
- Logical connectives include $\neg$, $\wedge$, $\vee$, and parentheses.
- The set of propositional letters is $\{a,b,c,d,e,f,g\}$
- Adjustable depth of formula
- The dataset is partitioned into a training and a test set for model evaluation.

BERT-uncased (Pretrained)

- Fine-tuning was performed on `bert-base-uncased`.
- Input format: tokenized propositional formulae.
- Achieved accuracy: **89.5%**.
- Open question: What if we designed a model from scratch for this specific task?

BERT from Scratch

- A domain-specific vocabulary was constructed, including only relevant logical symbols.
- A larger training dataset was generated to facilitate end-to-end training.
- Accuracy after training: **85%**.

Procedural Solvers

- Are there algorithms that are both sound and complete? Yes: the DPLL algorithm.
- SAT solving was performed by first converting all formulae to CNF.
- Achieved accuracy: **100%**.

Conclusions

- Does it make sense to use machine learning models for SAT solving?
- Hybrid approaches are promising:
  - Use machine learning to convert natural language into symbolic representations.
  - Apply symbolic SAT solvers (e.g., DPLL) on the structured input.