Predicting Unroll Factors Using Supervised Classification

Mark Stephenson, Saman Amarasinghe

Group 12

Anna Li, Ruipu Li, Tianchen Ye, Yuqi Li

Loop Unrolling

- Advantage of applying loop unrolling
  - Improve ILP
  - Reduce the number of executed instructions
  - Provide opportunity for other optimization

- However if an inappropriate loop unrolling factor is selected
  - Increased code size
  - Memory spills
  - Other side effects

Motivation - Why machine learning

- Creating reliable models manually
  - Requires a lot of effort
  - Requires expert knowledge

- Machine learning methods
  - Less effort and expertise
  - Work well with high-dimensional data

Related Works

- This paper (multi-class classification)
  - 65% accuracy score on their own dataset
  - 5% overall improvement on SPEC 2000 benchmark
- Machine Learning Approach for Loop Unrolling Factor Prediction in High Level Synthesis (random forest)
  - Reduced convergence time
- Efficient Loop Unrolling Factor Prediction Algorithm using Machine Learning Models (regression models)
  - 0.9997 accuracy score on LUTED dataset

Multi-class Classification for Compiler Heuristic Design

Two algorithms:

1. Near Neighbor Classification
   a. conceptually simple, but highly effective technique

2. Support Vector Machines
   a. a statistical learning algorithm that is widely used in the machine learning community

Feature
The loop nest level.
The number of ops. in loop body.
The number of floating point ops. in loop body.
The number of branches in loop body.
The number of memory ops. in loop body.
The number of operands in loop body.
The number of implicit instructions in loop body.
The number of unique predicates in loop body.
The estimated latency of the critical path of loop.
The estimated cycle length of loop body.
The language (C or Fortran).
The number of parallel "computations" in loop.
The max. dependence height of computations.
The max. height of memory dependencies of computations.
The max. height of control dependencies of computations.
The average dependence height of computations.
The number of indirect references in loop body.
The min. memory-to-memory loop-carried dependence.
The number of memory-to-memory dependencies.
The tripcount of the loop (-1 if unknown).
The number of uses in the loop.
The number of defs. in the loop.

Experiment with Multi-class Classifier for Loop Unrolling

Nearest Neighbor classification:

Database:

- $<x_i , y_i>$ pairs

Label:

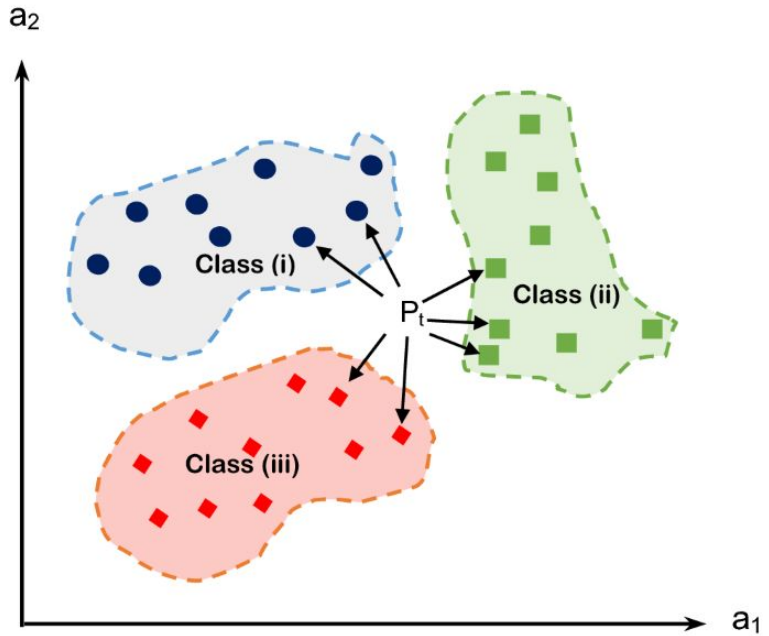
- unrolling factor 1, 2, 3, …,8

Similarity metric:

- $||x_{novel}-x_i||$

Radius:

- R = 0.3



Near Neighbor Classification

Data points:

- All the $<x_i , y_i>$ pairs

Label:

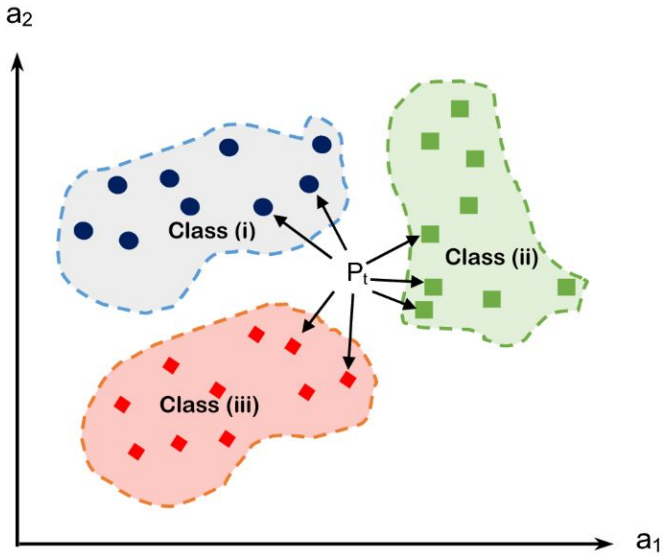
- Unroll factor

Similarity metric:

- Euclidean distance
- $||x_{novel}-x_i||$

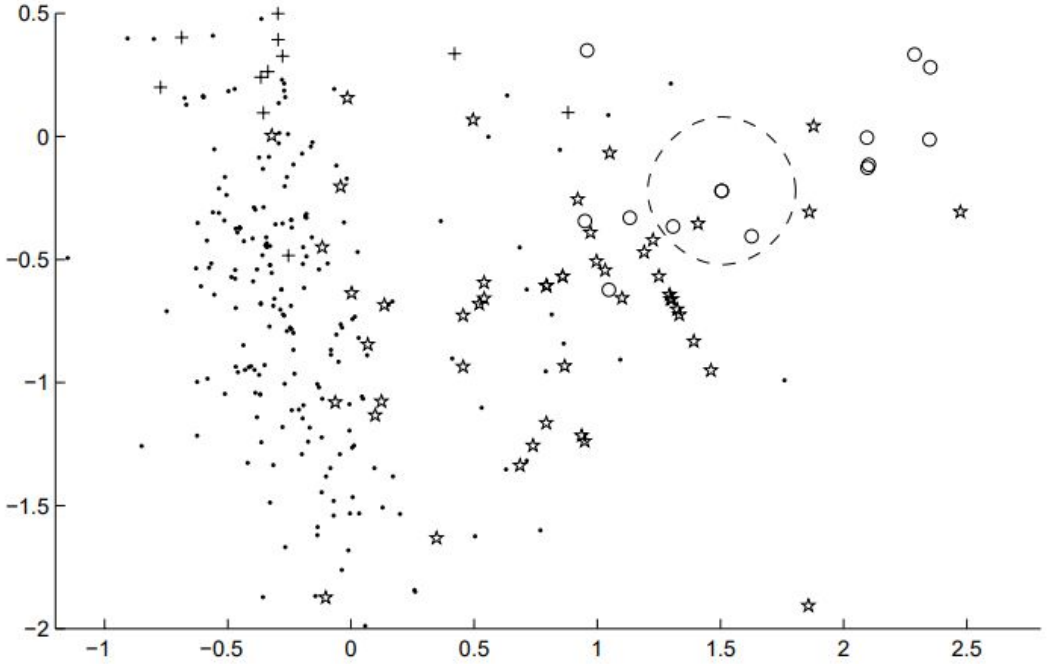
Radius:

- R = 0.3



Near Neighbor Classification

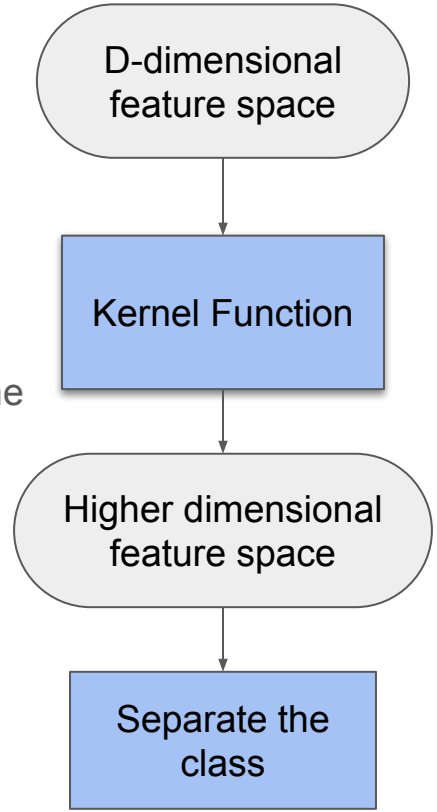
Example:



Support Vector Machine

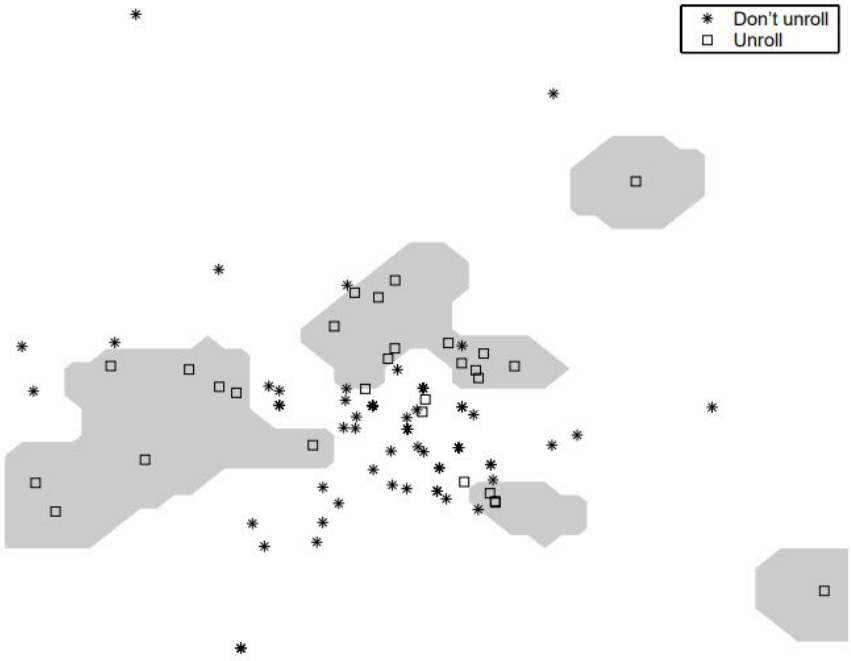
- Binary classifiers (convert to multi-classifiers)
- Output codes:
  - Associate a unique binary code to each label.
  - Train a SVM for each binary bit.
  - Concatenate each binary classifier prediction to get the multi-class prediction.

class	h_1	h_2	h_3
1	1	0	0
2	0	1	0
3	0	0	1

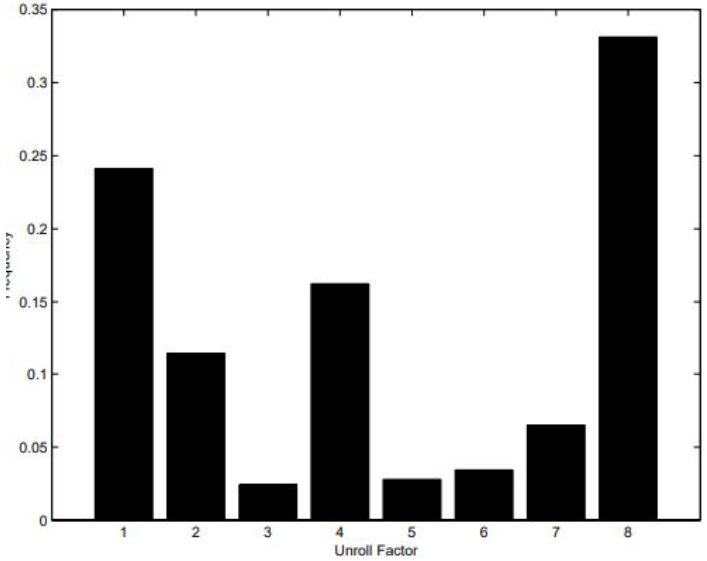


Support Vector Machine

Example:



Results of the Experiments



This figure shows the percentage of loops for which the given unroll factor is optimal.

This table shows the percentage of the predictions that each algorithm made that were optimal. In addition, the table shows the percentage of predictions made by each algorithm that were Nth best. The SVM predicts the optimal or nearly-optimal unroll factor 79% of the time.

Prediction Correctness	NN	SVM	ORC	Cost
Optimal unroll factor	0.62	0.65	0.16	1x
Second-best unroll factor	0.13	0.14	0.21	1.07x
Third-best unroll factor	0.09	0.06	0.21	1.15x
Fourth-best unroll factor	0.06	0.06	0.13	1.20x
Fifth-best unroll factor	0.03	0.02	0.16	1.31x
Sixth-best unroll factor	0.03	0.03	0.04	1.34x
Seventh-best unroll factor	0.02	0.02	0.05	1.65x
Worst unroll factor	0.02	0.02	0.04	1.77x

Comparison: NN vs SVM

- SVMs take longer to train than the NN algorithm.(30s vs 5ms)
- SVMs predicts unroll factors for novel examples more quickly than NN.

Feature Selection

Finding the most informative features for discriminating unroll factors

- Mutual Informative Score

- Greedy Feature Selection

Feature Selection - Mutual Informative Score(MIS)

J: The set of values that f can assume

$$I(f;u) = \sum_{\phi \in J} \sum_{y \in \{1 \ldots 8\}} P(\phi, y) \cdot \log_2 \left(\frac{P(\phi, y)}{P(y) \dot{P}(\phi)} \right),$$

Feature Selection - Mutual Informative Score(MIS)

Rank	Feature	MIS
1	# floating point operations	0.19
2	# operands	0.186
3	instruction fan-in in DAG	0.175
4	live range size	0.16
5	# memory operations	0.148

Table 3. The best five features according to MIS.

Feature Selection - Mutual Informative Score(MIS)

- Can not tell relationships among features
- Cannot guarantee that features will be useful for a particular classifier

Feature Selection - Greedy Feature Selection

Given a feature set $F = \{f_0 \dots f_N\}$,using a particular classifier

here is the algorithm:

1. Choose the best feature $b_0 \in F$
2. Choose second feature $b_1 \in F$, b_1 and b_0 best discriminates the training dataset
3. Stops with a defined number of iteration

Feature Selection - Greedy Feature Selection

Rank	NN	Error	SVM	Error
1	# operands	0.48	# floating point operations	0.59
2	live range size	0.06	loop nest level	0.49
3	critical path length	0.03	# operands	0.34
4	# operations	0.02	# branches	0.20
5	known tripcount	0.02	# memory operations	0.13

Table 4. The top five features chosen by greedy feature selection for two different classifiers on our dataset. The error numbers reported here are for the training set, hence the low error rates for these classifiers.

Feature Selection

1. Finally uses the union of Table 3 and Table 4.

2. Uninformative features would ‘confuse’ a learning algorithm or lead to overfitting of the training data

3. Although the number of instructions in the loop body appears only once, it is the de facto standard when discussing unrolling heuristics

Conclusion

Pros

- Supervised classification can effectively find good heuristics
- Spend little time tuning model parameters

Cons

- Spend a lot time and efforts acquiring training features and labels
- Predictions confined to training labels

Contribution & limitation

- Contributions
  1. Provide a new perspective on the field of compiler optimization
  2. The results got from machine learning could inspire further research in this area
- Limitations
  1. The proposed method heavily relies on the availability of training data
  2. The performance of the method has a lot of room for improvement
  3. It does not address the issue of generalization

Future work

- Exploring other machine learning models
- Incorporating additional loop features
- Optimizing for different evaluation metrics
- Extending to other architectures
- Improving other loop optimizations

Reference

1. M. Stephenson and S. Amarasinghe, "Predicting unroll factors using supervised classification," International Symposium on Code Generation and Optimization, New York, NY, 2005, pp. 123-134, doi: 10.1109/CGO.2005.29.

2. M. Stephenson and S. Amarasinghe, “Predicting Unroll Factors Using Supervised Learning.” *SlidePlayer*, https://slideplayer.com/slide/14574645/.

Q & A