Software Modernization Using Machine Learning Techniques

Budapest University of Technology and Economics

Norbert Somogyi (somogyin@edu.bme.hu)

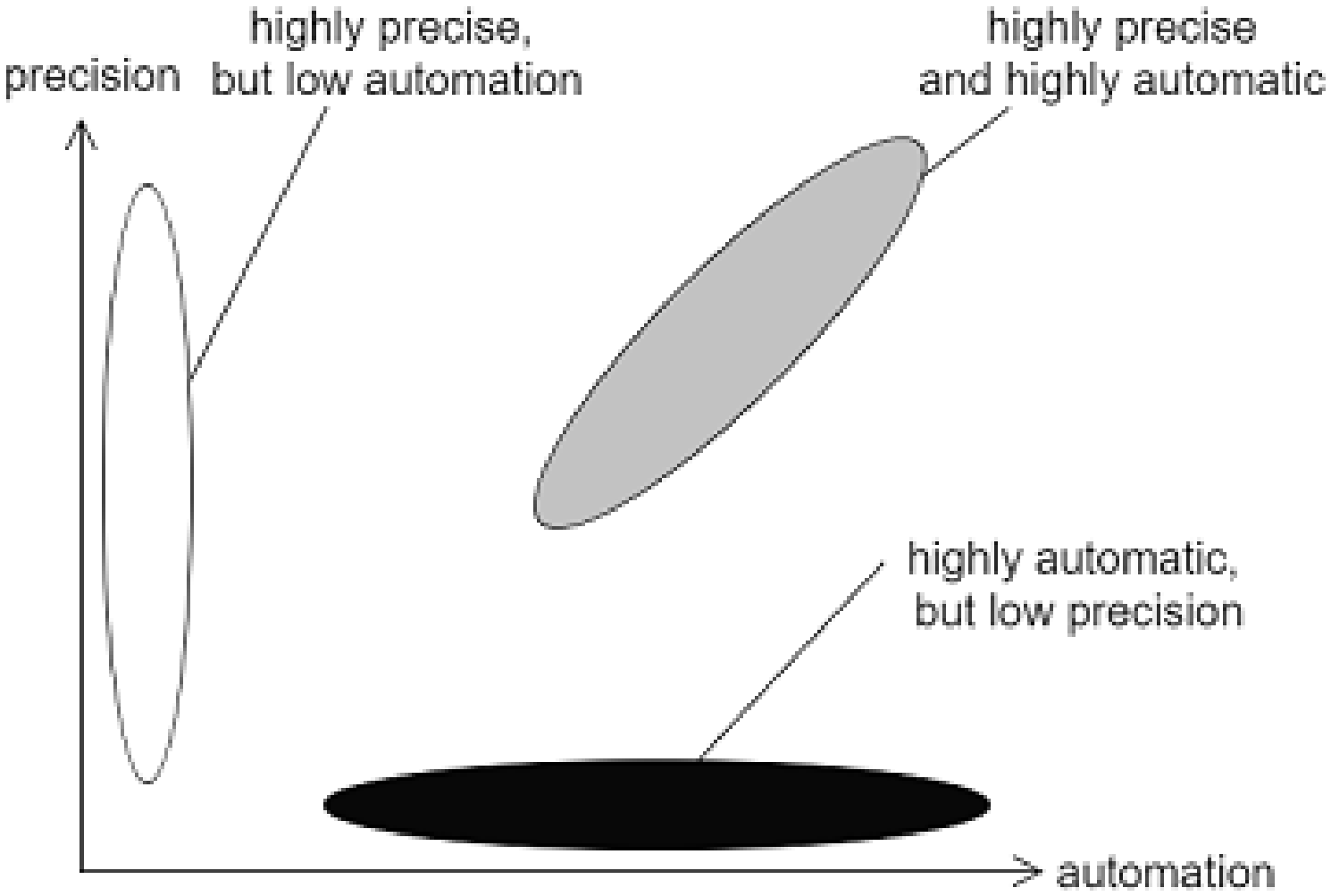
Gábor Kövesdán(kovesdan.gabor@vik.bme.hu)

BME AUT
Automatizálási és
Alkalmazott
Informatikai Tanszék

Software Modernization

- Software engineering evolves, old code becomes "outdated"
  - Legacy code
  - Architecture transform, paradigm-shift, etc.
- OO paradigms useful
  - Extensibility and reusability
- Procedural to OO : C → Java
  - Preserving semantics
  - OO design

Quality vs automation



Case study: pointer conversion

- Converting pointers
  - Binary classification problem
  - C to Java
  - Array or simple Java variable
- Previous solutions
  - User input: not automatic
  - Emulation: questionable quality
  - Static analysis: ?
  - Machine learning: ?

Pointers: Static Analysis

- Assumption: not array
- Find contradiction in code
  - Indexing operator
- Automatic, but not good enough
  - 58% success rate

Pointers: Machine Learning

- Data collection
- Find relevant attributes of pointers
  - > Index operator
  - > Arithmetics
  - > Derefer operator
  - > Parameter
  - > Return value
  - > Correct output

1	0	0	0	0	1
0	1	1	0	0	0
0	1	1	1	0	0
0	1	1	0	1	0

Pointers: Machine Learning

- Neural Network
  - Input nodes: 5
  - Output nodes: 1
  - Hidden nodes: 4
- Learning parameters
  - Learning rate: 0.01
  - Epochs: 30
  - Batch size: 50
  - Seed: 123
- Xavier weight initialization
- Stochastic gradient descent optimization
- ReLU, Sigmoid, Cross-Entropy

Pointers: Machine Learning

- 70% training, 30% validation

Parameter	Value
Accuracy	0.7456
Precision	0.8229
Recall	0.6432
F1 score	0.8343

- Significantly better results

Conclusion

- Code modernization
  - Hard classification problems
  - Static analysis often not good enough
- AI to the rescue!
  - Potentially much better, precise results
  - Fully automatic
- Case study: pointers from C to Java
  - Static analysis: 58%
  - Machine learning: 83% F1 score

Thank you for your attention!