A review of Machine Learning Approaches for Move Method Refactoring Recommendation

Presentation summary

Liviu-Ștefan Neacșu-Miclea – ICA 256/2

Slide 1. Title

Slide 2. Introduction

- Software entropy causes gradual design degradation.
- Refactoring preserves behavior while improving structure.
- Move Method tackles Feature Envy and cohesion issues.
- Automation is necessary for large-scale maintenance.

Slide 3. Motivation

- SBSE treats refactoring as an optimization problem.
- Large search space of method–class pairs.
- Metrics lack semantic awareness.
- ML can learn realistic design patterns from code.

Slide 4. Related work – QMove

- Metric-driven evaluation using QMOOOD attributes.
- Greedy search selects best improvement per step.
- Works well for Feature Envy but limited by metric expressiveness.

Slide 5. Related work – Methodbook

- Uses topic modeling (RTM) for conceptual similarity.
- Treats code text as documents with latent topics.
- Produces fewer but more meaningful recommendations.

Slide 6. Related work – PathMove

- Learns structural embeddings via AST paths.
- SVM classifier trained on real refactorings.
- Better precision than classical tools.

Slide 7. Related work – RMove

- Hybrid structural + semantic features.
- Adds dependency graph embeddings and deep models.
- Achieves state-of-the-art accuracy.

Slide 8. Discussion

- Progression: metrics, semantics, representation learning.
- Modern models capture complex structural/semantic cues.
- Challenges: data needs, explainability, computational cost.

Slide 9. Conclusions

- Hybrid ML approaches give best results today.
- Future work: explainable recommendations, IDE integration, use of LLMs.