About Me

- Michael Pradel
- At TU Darmstadt since 2014
- Before joining TUDA
  - Master-level studies in Dresden and Paris
  - Master thesis at EPFL, Switzerland
  - PhD at ETH Zurich, Switzerland
  - Postdoctoral researcher at UC Berkeley, USA
About the Software Lab

- My research group since 2014
- Focus: Tools and techniques for building reliable, efficient, and secure software
  - Program analysis
  - Test generation
- Thesis and job opportunities
Plan for Today

1. Organization

2. Topic of this seminar

3. Recent research from the Software Lab
Why Have a Seminar?

- **Learn fundamentals of doing research**
  - Read and digest papers
  - Present complex ideas to others
  - Scientific writing
  - Reviewing

- **Learn about machine learning and program analysis**
  - Maybe your future thesis topic
  - Opportunities for HiWis
Organization

- Today: Kick-off meeting
- January 14–18: One full-day meeting (or two half-day meetings)
  - Talks by students

Your tasks:
- Term paper
- Reviews
- Talk
- Active participation
Organization

- Today: Kick-off meeting
- January 14–18: One full-day meeting (or two half-day meetings)
  - Talks by students

Your tasks:
- Term paper
- Reviews
- Talk
- Active participation

Grading:
- Term paper: 30%
- Reviews: 10%
- Talk: 40%
- Active participation: 20%
Talk

- 15 minutes + questions
- English
- Present a recent research paper

Your mentor will help you prepare the presentation
- Send slides one week before the talk
- Incorporate feedback given by the mentor
Talk: Some Advice

Content:
- No need to explain all technical details
- But: Must contain some "meat"

Presentation:
- Examples are your secret weapon
- Stick to the time limit
- Practice, practice, practice

Pro tip: View video How to give a good research talk by Simon Peyton Jones
Talk: Rules

- Prepare your own slides
  - No copy & paste from existing slides, even if available

- You may use examples from the paper
Term Paper

- 6 pages
- English
- LaTeX template on course web site
- Summarize the paper in your own words
- Must be self-containing
Term Paper: Some Advice

- Don’t waste space on basics
- Examples are your secret weapon (yes, again)
- Bad English distracts from good content
- Revise, revise, revise
Term Paper: Rules

- No verbatim copying of text
  - Exception: Clearly marked, short quotes

- You may copy figures (e.g., result graphs)

- You must use your own example(s)
Reviews

- Imitates peer reviewing process
- Each student reviews three term papers
- Revise your term paper after getting reviews
  - Grade will be for final term paper

- Plain text format
- About 1 page, English
Reviews: Some Advice

- Be constructive
- Be polite
- Your reviews contribute to your grade, not to the reviewee’s grade
Dates

- **Deadlines:**
  - Oct 21, 2018: Pick preferred topics
  - Nov 23: Term paper
  - Dec 7: Reviews
  - Dec 21: Revised term paper
  - 7 days before final meeting: Send slides to mentor

- **Optional (and recommended):**
  - Before Nov 23: Meet mentor to clarify questions about your topic
Plan for Today

1. Organization  

2. Topic of this seminar

3. Recent research from the Software Lab
Topic of This Seminar

Machine Learning for Programming
Topic of This Seminar

Machine Learning for Programming

- Tools for improving software reliability and security
- E.g., program analyses to detect bugs, to complete partial code, or to de-obfuscate code
Topic of This Seminar

Machine Learning for Programming

- Source code as data
- Large code corpora to learn from
- Train models that predict program properties
What is Program Analysis?

Automated analysis of program behavior, e.g., to
- find programming errors
- optimize performance
- find security vulnerabilities

Input → Program → Output
What is Program Analysis?

- Automated analysis of program behavior, e.g., to
  - find programming errors
  - optimize performance
  - find security vulnerabilities
What is Program Analysis?

- Automated analysis of program behavior, e.g., to
  - find programming errors
  - optimize performance
  - find security vulnerabilities
Why Do We Need It?

Basis for various tools that make developers productive

- Compilers
- Bug finding tools
- Performance profilers
- Code completion
- Automated testing
- Code summarization/documentation
Traditional Approaches

- **Analysis has built-in knowledge about the problem to solve**
  - Significant human effort to create a program analysis
    - Conceptual challenges
    - Implementation effort
  - Analyze a **single program** at a time
Learning from Existing Data

■ Huge amount of existing code ("big code")
■ Programs are regular and repetitive
■ Machine learning: Extract knowledge and apply in new contexts
■ Learn how to ..
  □ .. complete partial code
  □ .. use an API
  □ .. fix programming errors
  □ .. create inputs for testing
Deep Learning

Class of machine learning algorithms

- Neural network architectures
- "Deep" = multiple layers
- Features and representation of inputs are extracted automatically

Revolutionizes entire areas

AlphaGo
Topics To Choose From

- 19 recently published research papers
- Submit your preferences by end of week
  - You pick three topics, we assign one
Plan for Today

1. Organization ✓

2. Topic of this seminar ✓

3. Recent research from the Software Lab