Xuankang Lin

lin420@purdue.edu
http://xuankanglin.com/About-Me/

Objective

Available on personal blog for better self demonstration.

Education and Work Experience

- **Purdue University** West Lafayette, IN
  M.S. in Computer Science *Expected: 12/2021*
  - Graduate Research Assistant. Applied formal verification to neural network safety reasoning.
  - Graduate Teaching Assistant for "Operating Systems" and "Distributed Systems" courses.
  - GPA 3.93 / 4.0. *A+* in Compiler, OS, Software Engineering, and Programming Languages courses.
- **Intel** Shanghai, China
  Intern at SSG group *12/2012 – 06/2013*
  - Optimized Cocos2d-html5's display on devices with Retina display.
  - Improved Cocos2d-html5's audio engine using Web Audio API (code merged into official repository).
- **Tongji University** Shanghai, China
  B.E. in Software Engineering *06/2013*
  - GPA: 4.69 / 5.0, top 2.4%. Outstanding Graduate of Shanghai in 2013.
  - Exchange student at Rose-Hulman Institute of Technology in fall 2012. GPA 4.0 / 4.0.

Publications

Lin, Xuankang, et al. "ART: Abstraction Refinement-Guided Training for Provably Correct Neural Networks." *2020 Formal Methods in Computer Aided Design (FMCAD)*. IEEE, 2020.

Skills

Advanced: Python, PyTorch, C/C++, Git **Proficient**: Scala, Java, Linux, LaTeX

Selected Projects **Github @XuankangLin**

- **KStepSafe** *01/2020 – Present*
  - Provide k-step safety guarantee of policy networks in cyber-physical systems from Deep Reinforcement Learning, via certification, training, or shielding. Implemented in PyTorch.
- **ART** *02/2019 – 05/2020*
  - Apply *A*bstraction-*R*efinement techniques to *T*raining neural networks. Generated neural networks come with strong safety guarantees due to soundness in DiffAbs' over-approximation, and mild accuracy impacts due to refinement on imprecise input abstractions.
  - Implemented in PyTorch, publicly available on Github. Paper accepted in FMCAD'2020.
- **DiffAbs** *02/2019 – 08/2020*
  - Abstract domain implementations for over-approximating reachable output sets of neural networks, including Interval and DeepPoly domains. Implemented in PyTorch, publicly available on Github.
- **Learning Latent Memory Models from Litmus Tests** *10/2016 – 04/2017*
  - Presented a new approach to learn memory models from litmus tests. Adapt the model simulator tool *herd7* in OCaml, generate *weakest* executions from litmus tests. Learn memory model from executions using Conditional Random Fields or Decision Tree in Scala.
  - More details in technical report, code publicly available on Github.