Ahmad Humayun

540-824-8988 | ahmad35@vt.edu | linkedin.com/in/ahmayun | github.com/ahmayun

Education

Virginia Tech — Blacksburg, VA
PhD in Computer Science — *Aug 2021 - Present*
Research Area: Automated software testing and security of data-intensive distributed programs.
Courses: Operating Systems, Advanced Topics in Software Engineering, Hot Topics in Machine Learning and Security.

Lahore University of Management Sciences — Lahore, Pakistan
Bachelor of Computer Science — *Aug 2017 - May 2021*

Publications

- ESEC/FSE 2024 | Y. Wu, A. Humayun, M. Gulzar, M. Kim. "Natural Symbolic Execution-Based Testing for Big Data Analytics".
- IEEE ASE 2023 | A. Humayun, Y. Wu, M. Kim, M. Gulzar. "NaturalFuzz: Natural Input Generation for Big Data Analytics".
- ESEC/FSE 2023 | A. Humayun, M. Kim, M. Gulzar. "Co-dependence Aware Fuzzing for Dataflow-based Big Data Analytics".

Professional Experience

Amazon Web Services — Santa Clara, CA
Applied Scientist Intern — *Summer 2025*

- Developed an LLM-powered application (deployed both as a standalone application and an MCP server) to automate the modeling of complex distributed algorithms in a low-resource programming language. Additionally, designed a library to translate modeled network messages into executable API calls.

Amazon Web Services — Santa Clara, CA
Applied Scientist Intern — *Summer 2024*

- Enhanced automated testing infrastructure of critical AWS APIs.

Virginia Tech, Department of Computer Science — Blacksburg, VA
Research Assistant - Prof. Muhammad Ali Gulzar — *August 2021 - Present*

- Working on developing novel methods to improve the state-of-the-art in testing for data-intensive applications.
- **Lead author** on two and second author on one research paper, published at **top tier conferences**
- I actively maintain two open-source GitHub repositories associated with these projects.

Projects

DAG-based Program Synthesis | *Program Synthesis, Graph Theory, Algorithms, Apache Spark, SQL, Optimizers*

- Built a DAG-based program-synthesis framework to fuzz DISC optimizers beyond SQL inputs.
- Designed a state-aware concretization engine to generate high-validity dataflows.
- Implemented plan-based oracles to find UDF-related optimizer bugs.
- So far, we have found three crashes in Apache Flink and a previously unknown optimizer bug in Apache Spark.
- **Submitted to OOPSLA '26**

DepFuzz | *Scala, PySpark, Apache Spark, Apache Hadoop, HDFS, Java, Apache Beam, Maven, MapReduce, Git*

- Developed a novel technique for efficient and effective fuzzing of Big Data (DISC) applications.
- Engineered an end-to-end fuzzer for Apache Spark in Scala with coverage monitoring
- Modified Scala compiler plugin *scoverage* to capture fuzzing coverage efficiently.
- Reimplemented state-of-the-art fuzzing techniques to use as baselines. Improved fault detection rate by 3.4× vs Jazzer. The resulting paper was **Accepted** at **ESEC/FSE 2023**. Acceptance Rate: 12.9% (Unconditional).

NaturalFuzz | *Scala, PySpark, Apache Spark, Apache Hadoop, HDFS, Java, GPT2, LLM, BERT, SQL*

- Developed a novel technique for efficiently generating natural-looking inputs for DISC applications.
- Designed algorithms to perform intelligent reduction of data using taint analysis.
- Engineered a modular system with complex interlinked components to fully automate the idea.
- Used state-of-the-art LLMs e.g. GPT2 and BERT to quantify the naturalness of synthetic data.
- Collaborated effectively with a team to bring the research idea into fruition.
- Wrote an end to end research paper that got **accepted at IEEE/ACM ASE 2023**.

Omniscient Debugger for Python | *x86, ptrace, strace, ELF, Python, Cython, Linux, Kernel*

- Engineered a debugger for Python applications that has the capability of stepping backward temporally.
- Used *ptrace* to intercept system calls and record program state selected points to be replayed later.

Transformer LLMs for Code Comprehension | *HuggingFace, FastAPI, PyTorch, Jupyter, BERT, GPT*

- Trained various LLM encoder stacks (e.g. BERT) to create general-purpose embeddings for code understanding.
- Analyzed 12K+ webpage JavaScript files and performed unsupervised training and clustering of embeddings using DistilBERT. Visualized after dimensionality reduction with t-SNE showing efficacy for code comprehension.

Benchmarking Framework for Software Debloating | *LLVM, C, C++, Docker, Python, gcc, clang*

- Researched and planned the feasibility of developing a benchmarking framework for software debloating tools.
- Developed an orchestrator for Docker containers in Python for facilitating software debloating research.
- Led a group of researchers to integrate the software debloating tool *Chisel* into the benchmarking framework.

Blocking Non-Essential JavaScript on Web Pages | *JavaScript, HTML, CSS, MDN, Chrome*

- Developed a browser extension for Mozilla Firefox to classify and block non-essential JavaScript from running on resource-constrained devices. Developed on-board lightweight classifiers for JS classification.

React Website with ExpressJS Backend | *React, JavaScript, Bootstrap, Firebase, OAuth*

- Lead the development of an online food ordering system using the REACT JavaScript framework with Firebase integration and an ExpressJS backend. The system consisted of 3 modules (Admin, Customer, Server).

RAFT-based Distributed Key-Value Store | *Go, Golang, Distributed Algorithms*

- Implemented a distributed key-value storage service using the distributed consensus algorithm RAFT in Golang, demonstrating proficiency in both distributed systems and the Go programming language.
- Designed and developed a fault-tolerant and highly available distributed system capable of electing leaders, replicating logs, and ensuring consensus among a cluster of nodes.
- Utilized Go's concurrency features, including goroutines and channels, to build a concurrent and efficient implementation that can handle concurrent requests and maintain system stability.
- Tested the RAFT implementation extensively, including scenarios involving network disruptions, node failures, and recovery, to validate its correctness and resilience.

Technical Skills

Languages: Java, Scala, SQL, C/C++, Python, JavaScript, Haskell, Intel x86, JVM Bytecode, Go, TypeScript, R, Matlab

Tools and Frameworks: LangChain, LangGraph, MCP Servers, Spark, PySpark, Hadoop, SBT, Maven, Docker, Intellij, Node.js, React, Flask, Flutter, FastAPI, HuggingFace, Keras, PyTorch, LLVM, ptrace, gcc, clang, Git, Docker, VS Code, Visual Studio, IntelliJ, Eclipse