Tran Ma

tranma | tranma | ma.ngoc.tran@gmail.com | +61425896023

About

I have experience building data-centric applications, machine learning foundations, experience with modern computer vision models, strong research skills, and a passion for machine learning.

Work Experience

Ghost Autonomy Mountain View, CA, USA and Sydney, AU
Machine Learning System Engineer April 2019 - March 2024

I worked on Ghost Train, the internal *training and deployment* system, targeting Qualcomm mobile hardware in the car.

– I wrote a *library for building models* as Scala programs, analogous to PyTorch models in Python. Most of Ghost software was in the Scala/Java ecosystem, and this enabled engineers to seamlessly integrate with other parts of the platform such as sensors.

– I implemented various *array operations* and their derivatives for CPU and GPU (with OpenCL). These operators ran in both training and inference in the car. Having the same implementation for both cases improved confidence in the model and reduced maintenance overhead.

– I tested the evaluation and back-propagation engine. I used random generation techniques to find errors where shared model structures broke the algorithm.

– I wrote the *compiler to produce inference code* for models trained in our system. It stripped out any part of the model not required for inference, and defined static components such as buffer layout in advance. The compiled model is a C-like Scala program that plugged into the rest of our driving platform.

– I *accelerated the inference code* with rewrite rules and operator fusion, reducing memory IO to/from the GPU. These techniques got our perception models to run in the order of hundred-milliseconds.

– I *specialized operator implementations* to take full advantage of the compute and memory model on our target GPU, the Qualcomm Adreno 630. I used empirical experiments and benchmarks to discover optimal input partitions and memory locations, producing several convolutions specialized to input shapes. These techniques got our perception models to run in tens-of-milliseconds.

– I *optimized array layouts* in the compiled model to reduce the number of array transposes. I identified the trade-offs between transposes and better convolution performance when the arrays were in a different order. This was necessary to get our bigger models to run in real-time (30fps).

– I *quantized* our models to use low-precision for model weights. This took better advantage of the memory model on our GPU while keeping most of the model accuracy.

– I wrote an exporter to produce ONNX files from our models. This served as a package for trained model weights, and allowed us to interact with other tools.

– As tools matured, we stopped development on Ghost Train in favour of PyTorch and vendor deployment options (QNN), but maintained support for production models that had difficulties with the vendor option. I wrote tests for the new deployment system and found problems where low-precision operators produced the wrong results.

I worked on *lane detection* in the model engineering team, building a lane model in birds-eye view (BEV).

- I implemented known methods in BEV lane detection and evaluated them on our data, finding poor performance due to differences in data distribution with regard to camera pitch and road geometry.
- I evaluated our lane model on open-source datasets and researched ways to represent complex lane geometries in BEV.
- I built lane identification on top of a probabilistic ground-truth algorithm for finding BEV lanes. This produced a small set of high-quality ground-truth we could use to bootstrap our BEV model.

I also contributed to program verification in other parts of the platform. I introduced the synchronous model for control systems, later used by other teams to verify the vehicle interface.

Standard Chartered Bank London, UK
Quantitative Developer Oct 2018 - Mar 2019

I was part of the risk analytics team and wrote checks for the bank's value-at-risk estimate.

Cog Systems Sydney, Australia
Senior Software Engineer (Contract) Jun 2018 - Sep 2018

I built a toolchain for trusted control systems and protocols.

- I designed and built the front-end specification language, a simplified version of the modelling tool Promela/SPIN. I wrote examples of simple protocols and algorithms in this language to check its expressivity for Cog applications.
- I wrote a compiler to produce verified binaries for programs specified in the front-end language. I targeted a subset of C supported by the certifying C compiler CompCert.
- I wrote a tool to extract proof obligations in Isabelle/HOL from the same front-end program. A user could complete the proofs and gain confidence that the compiled binaries behave as the proof specified.

Ambiata Sydney, Australia
Software Engineer Jun 2015 - Mar 2018

I worked on the Extract-Transform-Load pipeline, which processed terabytes of unstructured data for customers in insurance and advertising.

- I built the compiler for icicle, a *streaming query language*. I wrote the code generation to C, and the type-checker to ensure missing and invalid data is handled, and that input streams are processed once per query. The system eliminated costly ingest mistakes, ensuring that properly formed queries did not fail and produced valid results.
- I tested and fixed bugs in a number of related projects, mainly zebra, a columnar binary data store, and ivory, an immutable feature store.

Anchor Sydney, Australia
Software Engineer Aug 2014 - Apr 2015

I worked on cloud and web services on top of OpenStack.

- I wrote a tool to aggregate customer billing information from counters and metrics across the cluster.

– I wrote telemetry tools for an internal distributed data store, producing profiling information to identify performance bottlenecks in the store.

– I wrote a tool to propagate non-conflicting changes across stores of structured data. This eliminated the need for ad-hoc scripts to reconcile large amounts of JSON/XML produced by other tools.

Open Kernel Labs Sydney, Australia

Software Engineer Mar 2013 - Jun 2014

I worked hypervisor-based virtualization for secure embedded devices.

– I implemented support for virtual hardware interrupts using extensions from ARMv7 to achieve faster virtualization over existing methods.

– I worked on the internal build tool to propagate changes across different Linux kernel versions.

National ICT Australia Sydney, Australia

Research Assistant 2011 - 2012

I was an undergraduate student in the seL4 verified microkernel project. I worked on information flow proofs and proof automation tactics.

Education

2009 - 2013 BSc. Computer Science at **University of New South Wales, Australia** (Honours)

Skills

Machine Learning	Computer Vision, Implementation
Compilers	EDSLs, Optimizations, Type Theory
ML Libraries	PyTorch, TensorFlow, scikit-learn
Languages	Python, C, OpenCL C, Scala, Java, Haskell