CS620: FM in ML
What, Why and How? A High-level Perspective

Supratik Chakraborty

(Week 2 Lecture)
Safety in AI/ML

- **AI/ML based systems**
  - Computational systems that try to mimic (and improve upon?) human reasoning
  - Increasingly pervading our lives

- **Applications span entire spectrum of consequences**
  - Benign: Error causes nothing more than inconvenience
    - Auto completion in chat, game of chess, recommendation of restaurants, …
  - Potentially serious, but recoverable consequences:
    - Approval of bank loans, bail applications, …
  - Serious irrecoverable consequences:
    - Collision avoidance in unmanned drones, self-driving cars, weapons systems, malware detection, …

- **Can we trust decisions by AI/ML based systems in applications where cost of errors is extraordinarily large?**
  - Human lives, breach of privacy, security gaps, loss of critical infrastructure …
A Typical Setup

Application with ML component

ML component under study

Other components in application

Application's environment
A Typical Setup

Application with ML component

ML component under study

Other components in application

Application's environment

Does this do the right thing in all corner cases?
Different perspectives

- **Machine learning perspective**
  - **“Accidents”**
    - Unintended, harmful behaviour stemming from “bad” design of ML components
    - Wrong objective function design?
      - Negative side effects, reward hacking
    - Too expensive to evaluate correct objective function frequently
      - Bad extrapolations
    - Training based on insufficient or poorly curated data
    - Errors due to distributional shift of inputs
  - Core machine learning techniques can be used to reduce “accidents”
    - Scalable, works in a large spectrum of real-world settings
    - Are all corner cases covered? Do we have proofs of correctness?
Different perspectives

Large collection of promising approaches based on ML techniques

See Amodei et al’s paper for details, if interested

Can we depend on training/designing complex networks using to always work as desired in previously unseen corner cases, when the cost of an error is huge?

All ML based techniques to mitigate problem are important and must be used

But are these sufficient?
Different perspectives

- **Formal methods perspective**
  - Required reading: “Towards Verified Artificial Intelligence” by Sanjit Seshia, Dorsa Sadigh, S. Shankar Sastry
  - System: E.g., Neural net in self-driving car
    - Mathematical model of system’s behaviour ($S$)
  - Environment: E.g., Road, weather, traffic, driver interventions, ...
    - Mathematical model of environment’s behaviour ($E$)
  - Property: A precise mathematical formulation ($F$) of acceptable behaviour of $S$ operating in $E$
  - Algorithmic search of proof space
    - Either obtain a proof that system satisfies property
      - $(S || E) \models F$
    - Counterexample (network inputs) that demonstrate violation of property
      - Model of $(S || E) \land \neg F$
  - **Several challenges along the way**
Different perspectives

Ref: Towards Verified Artificial Intelligence, Seshia, Sadigh and Sastry
Different perspectives

Formal methods perspective

- Hugely successful in hardware industry
- Moderately successful in software industry
- Formal methods based verification/analysis routine in several industrial hardware/software design flows
  - Every processor chip from Intel/AMD has parts of the design formally verified
  - Every time you fly an Airbus aircraft, large parts of the auto-pilot software formally verified
  - Every time you insert a USB device into a Windows machine, formal verification of downloaded drivers happens
- Can we make the technology work for AI/ML based systems?
Different perspectives

FM in ML goes beyond proofs/counterexamples of safety properties

Can we use formal methods based reasoning to

- Verify correctness of algorithms used to train complex ML components?
- Do correct-by-construction design of ML components that satisfy formally specified properties?
- Provide explanations based on formal models like decision diagrams, probabilistic programs, Markov Decision Processes?
- Fish out adversarial inputs for well-trained ML components?
- Analyze robustness, fairness, privacy, security, transparency etc.?
Some common problems

System S

High dimn input space, parameter space: scalability of analysis?
Some common problems

Environment E
How do we model ?

Other components in application

Application’s environment
Some common problems

Property F: (Vehicle within 5m on left) $\Rightarrow \neg$ (Steer left)
Some common problems

What is the corresponding property for S?
Modeling environment

● Uncertainty omnipresent: First class entity in reasoning
● Some things are inherently hard to model
  ○ Human behaviour, traffic conditions
● Probabilistic models with uncertainty of parameters built in
● Non-determinism used liberally may result in too many false negatives
● Need to combine probabilistic and non-deterministic modeling intelligently
● Markov Decision Processes (MDPs), probabilistic programs, …
● Abstractions in environment modeling
  ○ Different environment components may need to be abstracted differently
Specification of what is desired behaviour

- Often hard to formalize
  - Significant chunk of time spent on this even in software/hardware verification
  - Even harder in general for AI/ML based systems
- “Data as specification” vs “formal specification”
  - Can this gap be bridged?
  - Specification mining from behaviours, traces?
  - Evolving specification, as system used in different contexts?
- Quantitative vs Boolean specifications
  - Quantitative specs often have an optimization flavour
  - Does a system satisfy/fail a property or get a formal score for property satisfaction?
- End-to-end spec as opposed to compositional spec is often more feasible
  - Can we infer compositional spec from end-to-end spec (needn’t be human readable)
Modeling the system

- Very high dimensional input space
  - Semantic feature spaces can lower dimension
- Very high dimensional parameter space
  - Reasoning over parameter space can quickly turn hopelessly intractable
- Need abstraction mechanisms suitable for scale of ML component complexity
  - Walking a tight rope -- computational efficiency vs precision of analysis
- Use logical formalisms to “explain” ML components
  - Some of these can be used as models
- Model systems in context
  - Perhaps not necessary to model arbitrary behaviours
Efficient computational engines

- Hardware & software verification settings
  - Symbolic model checking, SAT/SMT solvers, numerical simulation techniques..
  - These may not suffice out-of-box for AI/ML contexts.

- AI/ML context
  - Data generation, satisfying soft, hard, distributional constraints (realism)
  - Efficient constraint solving techniques with ReLUs, sigmoids, etc.
  - New abstraction/refinement techniques for ReLUs, sigmoids for sound analysis
  - Compositional reasoning
    - Assume-guarantee reasoning for Boolean models/specifications relatively mature
    - Similar reasoning for probabilistic/quantitative models/specifications?
Holy grail: Correct by construction

- Significant success in hardware, restricted software context
- Can we design ML components that provably satisfy given specs?
  - Inductive synthesis
  - Safe learning-based control
  - Safe reinforcement learning
- Theorem proving for correctness of algorithms used for training ML models
- Resilience and fault tolerance at run time essential given the complexities of AI/ML components
  - Wrap within run-time assurance framework to ensure nothing unsafe happens
Realistic expectations

- Given scale and complexity of today’s AI/ML based systems
  - Challenging, if not impossible, to design correct-by-construction ML system, or formally verify overall correct operation without restrictive/unrealistic assumptions
  - Nascent area, lots of promising ideas in literature

- Therefore,
  - **Core ML techniques, Formal Analysis/Verification AND Run-Time Assurance needed**

Focus of this course: Formal Analysis/Verification
Topics to be covered

- Specifying properties for ML components
- Modeling environments and neural networks
- Abstract interpretation for analyzing deep neural networks
- Customized constraint solvers
- Automata theoretic analysis of recurrent neural networks
- Verified Reinforcement Learning
- Robustness analysis through formal methods lens
- Explainability of ML components: logic based approach