Safety of Artificial Intelligence: A Collaborative Model

Professor John McDermid OBE FREng
Ms Yan Jia
University of York, UK
Overview

Key Topics

• Motivation
• The Collaborative Model
  • System/functional safety
  • AI/ML safety
  • Safety-Critical Software Engineering
• A Case Study – Sepsis treatment
• Refining the Collaborative Model
• Building a Community
Motivation

Different Communities

• Growing understanding of the potential for AI/ML-based systems to produce undesirable results
  • For example, the COMPAS system recommending prison sentences showed systematic bias

• AI/ML community
  • Identified generic issues
  • Solutions by adapting ML methods, i.e. in paradigm

• Safety community
  • Concern with the challenges to established approaches
  • Solutions by adapting approaches, i.e. in paradigm
Motivation

AI Community

• Various “formalisations”
  • The “concrete problems” of AI [Amodei et al 2016]
  • The “reward-result gap” [Leike et al 2018]

• Various approaches to resolution
  • Reward modelling
  • Adversarial resilience
  • Explainability

• In the long-run artificial general intelligence (AGI)
  • Providing context and semantics missing in “narrow” AI
Motivation

Safety Community

• Some attempts to prohibit/limit use
• Some work on standards
  • UL 4600 for autonomous systems includes requirements on ML-elements of systems
• Some work on adapting safety principles
  • Assurance of ML in Autonomous Systems (AMLAS) [Picardi et al 2020] based on ML lifecycle model
    [Ashmore et al 2019] give *desiderata* for lifecycle stages
• Requirements typically for safety/assurance cases
  • Less clarity on how to meet the requirements (evidence)
Assuring Safety

Safety/Assurance Cases

- Concept – argument supported by evidence

Explanation of how work products can be interpreted as indicating acceptable safety
Motivation

Collaboration

• ML produces the deployed system
• If safety processes “merely look on” & document
  • Probably can’t make an adequate judgment of safety
  • Likely to be ignored (as irrelevant)
• To provide value
  • Safety must influence the design
  • Help produce a better system (ensure safety)
  • Provide evidence done so (assure safety)
• Need collaboration – a shared paradigm
The Collaborative Model

Overview

<table>
<thead>
<tr>
<th>System Safety/Functional safety</th>
<th>Hazard Analysis</th>
<th>(Derived) Safety Requirements</th>
<th>Safety/Assurance Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;AI/ML Safety&quot;</td>
<td>Model Alignment</td>
<td>Data collection &amp; ML model development</td>
<td>Satisfaction of ML Desiderata</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Safety Critical Software Engineering</th>
<th>Coding Standards</th>
<th>Static Analysis/Verification</th>
<th>Other practices ...</th>
</tr>
</thead>
</table>
The Collaborative Model

Layer 1: Functional/System Safety

- Hazard analysis
  - Identify hazards and estimate associated risks
- Derived safety requirements (DSRs)
  - On ML and other elements of the system so contribution to hazards is controlled (or role in mitigation is defined)
  - May be on the product and/or on the development process
- Safety/assurance case
  - Arguments for safety of the system supported by ML layer evidence that the desiderata and DSRs are met
The Collaborative Model

Layer 2: AI/ML Safety

- Model alignment
  - Meeting the design intent, including avoidance of hazards
- Data collection and model development
  - The first two stages of the ML lifecycle [Ashmore et al 2019] (third is verification) informed by the DSRs
- Satisfaction of DSRs and desiderata
  - Verification, producing appropriate evidence for the safety/assurance case
    NB doesn’t resolve the ”how much evidence” question
# Safety Standards

**EN50128 (Software, Rail Sector)**

<table>
<thead>
<tr>
<th>TECHNIQUE/MEASURE</th>
<th>Ref</th>
<th>SWS ILO</th>
<th>SWS IL1</th>
<th>SWS IL2</th>
<th>SWS IL3</th>
<th>SWS IL4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Formal Proof</td>
<td>B.31</td>
<td>-</td>
<td>R</td>
<td>R</td>
<td>HR</td>
<td>HR</td>
</tr>
<tr>
<td>2. Probabilistic Testing</td>
<td>B.47</td>
<td>-</td>
<td>R</td>
<td>R</td>
<td>HR</td>
<td>HR</td>
</tr>
<tr>
<td>3. Static Analysis</td>
<td>D.8</td>
<td>-</td>
<td>HR</td>
<td>HR</td>
<td>HR</td>
<td>HR</td>
</tr>
<tr>
<td>4. Dynamic Analysis and Testing</td>
<td>D.2</td>
<td>-</td>
<td>HR</td>
<td>HR</td>
<td>HR</td>
<td>HR</td>
</tr>
<tr>
<td>5. Metrics</td>
<td>B.42</td>
<td>-</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
</tr>
<tr>
<td>6. Traceability Matrix</td>
<td>B.69</td>
<td>-</td>
<td>R</td>
<td>R</td>
<td>HR</td>
<td>HR</td>
</tr>
<tr>
<td>7. Software Error Effects Analysis</td>
<td>B26</td>
<td>-</td>
<td>R</td>
<td>R</td>
<td>HR</td>
<td>HR</td>
</tr>
</tbody>
</table>

**Requirements**

1. For Software Safety Integrity Level 3 or 4, the approved combinations of techniques shall be:-
   a) 1 and 4
   b) 3 and 4
   or  c) 4, 6 and 7

2. For Software Safety Integrity Level 1 or 2, the approved technique shall be 1 or 4.
The Collaborative Model

Layer 3: Software Engineering

• Many relevant software engineering techniques
  • Paper and example focus on coding standards and static analysis
    • Coding standards – rules for programming that avoid common classes of error, e.g. divide by 0, buffer overflow
    • Static analysis – checks on code without executing it

• In practice, ML development very agile
  • Are some good techniques
  • However, safety standards mainly based on V life-cycle
  • So need to draw on principles not specifics of standards
Overview

Key Topics

• Motivation
• The Collaborative Model
  • System/functional safety
  • AI/ML safety
  • Safety-Critical Software Engineering
• A Case Study – Sepsis treatment
• Refining the Collaborative Model
• Building a Community
A Case Study

Sepsis and its Treatment

- Sepsis is a complex life-threatening situation known to be difficult to diagnose and to treat
  - One of the highest causes of deaths in hospital
  - Generally, leads to organ dysfunction
- Treatment includes
  - Delivery of vasopressor
  - Delivery of intravenous (IV) fluids
- Case study uses reinforcement learning (RL) to derive an optimal treatment policy for vasopressor and IV
  - Analysis and refinement of already published RL model
Case Study

Clinical Pathway Incorporating RL

Does the patient have a suspected infection? Yes → Does the patient have EWS > 4? Yes → Is any one sepsis red flag present? Yes → Lab and other tests

Clinician assessment

RL model recommendation

Clinician choice and action

Check serial lactates
Take blood cultures before administering antibiotics
Measure urine output to ensure fluid balance chart commenced & completed hourly
CASE Study

RL Model Basics

• Feature space
  • 48 features, mainly clinical, showing patient state

• Action space
  • 25 discrete actions, codifying a combination of IV and vasopressor doses

<table>
<thead>
<tr>
<th>Dose of IV fluid</th>
<th>Dose of vasopressor (mcg/kg/min)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No.: 0</td>
</tr>
<tr>
<td></td>
<td>Range: 0</td>
</tr>
<tr>
<td></td>
<td>Median: 0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1 (0.002, 0.079)</td>
</tr>
<tr>
<td>2</td>
<td>2 (0.08, 0.2)</td>
</tr>
<tr>
<td>3</td>
<td>3 (0.201, 0.449)</td>
</tr>
<tr>
<td>4</td>
<td>4 (0.45, 1.005)</td>
</tr>
<tr>
<td></td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>0.135</td>
</tr>
<tr>
<td></td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>0.786</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>21</td>
</tr>
</tbody>
</table>
Case Study

Layer 1: Hazard and Risk Analysis

• SHARD analysis method
  • Initially developed for analysing software-intensive systems
  • Applies guidewords to flows in design
    • Omission
    • Commission
    • Incorrect
    • Early
    • Late
  • Adapted to the clinical pathway in the case study
<table>
<thead>
<tr>
<th>Guide word</th>
<th>Deviation (Hazards)</th>
<th>Possible Causes</th>
<th>Effects</th>
<th>Severity</th>
</tr>
</thead>
</table>
| Incorrect  | Sudden change of vasopressor dose is administered (concerns two consecutive doses) | 1 Kink of line  
2 The pump fails, e.g. due to electrical problem or bag/syringe not installed correctly  
3 The delivery line might not be connected to patient’s central line, e.g. due to the patient pulling out the central line  
4 The drug might not be added to the diluent, so the syringe/bag just contains saline (a problem when bags/syringes are being changed over)  
5 Initial recommendation by doctor has a sharp change in dose and doctor carried through the recommendation (not considered in this paper)  
6 RL agent recommends a sharp change in dose and doctor accepts the advice, e.g. due to automation bias  
7 Inappropriate titration of dose by nurse  
8 Doctor fails to check current dose  
9 Features in state space of the RL model are not sufficient to represent the patient conditions for sepsis decision making  
10 Reward function used for RL model is coarse  
11 Cost function used for RL model development is not appropriate  
12 Hyperparameters used for RL model development are not optimised  
13 Training data for RL model development is not appropriate  
14 Nurse prepared wrong dose (e.g. due to calculation error)  
15 Data corruption (e.g. invalid or wrong data produced by over-writing patient’s features)  
16 Features for wrong patient entered  
17 Wrong patient feature values entered (e.g. due to unit difference)  
18 Test results for wrong patient received  
19 Incorrect test results received | Acute Hypotension, Strokes, Renal failure, Heart attack could occur from a sharp drop in the dose  
Hypertension, Cardiac Arrhythmia, Strokes, Raised intracranial pressure, Pulmonary oedema could occur from a sharp rise in the dose | Major/considerable |
# Case Study

## Layer 1: Derived Safety Requirements

<table>
<thead>
<tr>
<th>ID</th>
<th>Description</th>
<th>Type</th>
<th>Allocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>R0</td>
<td>Sudden changes in recommended dose shall be close to clinician practice</td>
<td>Performance &amp; Safety</td>
<td>RL model development</td>
</tr>
<tr>
<td>R1</td>
<td>Feature representation in the state space shall be sufficient to allow the</td>
<td>Performance &amp; Safety</td>
<td>RL model design</td>
</tr>
<tr>
<td></td>
<td>control of sudden changes in recommended dose</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R2</td>
<td>An appropriate reward function shall be defined to allow the recognition of</td>
<td>Performance &amp; Safety</td>
<td>RL model design</td>
</tr>
<tr>
<td></td>
<td>desired clinical outcome</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R3</td>
<td>An appropriate cost function shall be defined to penalise hazardous</td>
<td>Performance &amp; Safety</td>
<td>RL model development</td>
</tr>
<tr>
<td></td>
<td>behaviours</td>
<td></td>
<td></td>
</tr>
<tr>
<td>R4</td>
<td>Hyperparameters shall be optimised based on the validation dataset</td>
<td>Performance &amp; Safety</td>
<td>RL model development</td>
</tr>
<tr>
<td>R5</td>
<td>Patient cohort shall be defined using recognised criteria, i.e. sepsis-3</td>
<td>Performance &amp; Safety</td>
<td>RL model design</td>
</tr>
</tbody>
</table>
Case Study

Layer 2: ML Model Development

• DSRs met by modifying the model state space (R1) and the cost function (R3)
  • NB approach uses double DQNs

<table>
<thead>
<tr>
<th>Table 4. Major changes in the modified RL model</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Features in state space (R1)</strong></td>
</tr>
<tr>
<td>-------------------------------------</td>
</tr>
<tr>
<td><strong>RL model in [32]</strong></td>
</tr>
<tr>
<td>48</td>
</tr>
<tr>
<td><strong>Modified RL model</strong></td>
</tr>
<tr>
<td>48 (Removed one feature – timestep, added an extra one – relative dose change)</td>
</tr>
</tbody>
</table>

Case Study

**Initial Approach**
- High rate of sudden changes

**Modified approach**
- Closer to clinicians
- Meets R0

### Table 4. Summary of max dose change between consecutive doses for the three policies

<table>
<thead>
<tr>
<th>Dose of vasopressor (mcg/kg/min)</th>
<th>Small-Medium Dose Change (0-0.75)</th>
<th>Large Dose Change (&gt;0.75)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clinician Policy</td>
<td>97% (2,100)</td>
<td>3% (60)</td>
</tr>
<tr>
<td>Original Policy</td>
<td>65% (1,404)</td>
<td>35% (756)</td>
</tr>
<tr>
<td>Modified Policy</td>
<td>92% (1,990)</td>
<td>8% (170)</td>
</tr>
</tbody>
</table>

Figure 5. Modified Policy: Comparison of max absolute vasopressor dose change in one step for each patient in the test data set between the clinician and the learnt modified policy.
Case Study

Layer 3: Static Analysis

• Software developed in Python
  • Analysis done using PyLint

• PyLint “tags” issues
  • C: coding convention violation;
  • E: for programming errors, likely a “bug”;  
  • F: for fatal;
  • R: for “refactoring” to improve the score against some quality metric;
  • W: for warnings, e.g. minor programming errors or style
Case Study
Layer 3: Static Analysis

• Fragment above shows errors and style issues
• Code improvement recorded over time (10 is hard)

Your code has been rated at 3.78/10 (previous run: 1.03/10, +2.75)
Refining the Model

Some Necessary Steps

• Approach works directly in the case study
  • Can “implement” DSRs by changing the feature space and the cost function
  • But need to adapt to other ML models – e.g. for unsupervised learning may need to encode DSRs in monitors

• Need to consider wider issues, e.g.
  • A more “ML aware” safety process
  • Role of explainability in assurance
  • Sufficiency of evidence
  • Moving more to a “continuous assurance” model
Refining the Safety Process

Refinement for Autonomous Systems

- Safety processes
  - SOCA: acceptability
  - SACE: whole system, including shared control
  - SAUS: understanding
  - SADA: decision-making
  - AMLAS: assurance of ML

SR – Safety Requirement
Building a Community

ML and Safety and More

- ML and safety communities use different languages
  - Perhaps even mean different things by “AI Safety”!
  - Need to establish better means to communicate and collaborate to achieve safe AI/ML/autonomy

- But the onus is with the safety engineers
  - ML developers produce the systems
    - They will make them safe (or not)
  - Safety engineers must add value, e.g. derived safety requirements to use in ML performance evaluation

- Also involve safety-critical software engineering