How to engineer machines that learn?
Software Engineering practices revisited in the age of ML
Two perspectives

**ML4SE** – how to use ML as a technique to enhance software engineering?

**SE4ML** – how to apply SE methods when building software with ML components?

Software Engineering  Machine Learning

**ML4SE** – how to use ML as a technique to enhance software engineering?

**SE4ML** – how to apply SE methods when building software with ML components?
Between extremes

AI will solve all our problems!!

The robots will replace us!!

Let’s build and apply AI systems in a robust and responsible manner.

Software Engineering
Questions

Is there anything special about software that contains ML?

How does that impact software engineering?

What is currently known about engineering practices for ML?

What challenges still await us?

For now, focus on ML, rather than broader, less delineated area of AI.
What is so special about software that contains ML?

from an engineering perspective

data intensive
inherent uncertainty
empirical iteration
What is so special about software that contains ML?

from a social and organizational perspective

- sky-high expectations
- wide talent gap
- potential for harm

"AI and ML to solve complex challenges"
"Europe has an AI skills shortage"
"Anti-fraud system SyRI violates privacy"
Benefits of AI

The average flight of a Boeing plane involves only seven minutes of human-steered flight, which is typically reserved only for takeoff and landing.

The vast majority of major banks rely on technology [...] which uses AI and ML to decipher and convert handwriting on checks into text via OCR.

Less than 0.1% of email in the average Gmail inbox is spam, and the amount of wanted mail landing in the spam folder is even lower, at under 0.05%.

A standard feature on smartphones today is voice-to-text. [Voice-to-text] has become the control interface for [...] smart personal assistants [e.g. Alexa]

Risks of AI

**COMPAS** = Correctional Offender Management Profiling for Alternative Sanctions
Predict recidivism – will a person become a repeat offender?
Used to decide who can be released from jail on bail pending trial

---

**Prediction Fails Differently for Black Defendants**

<table>
<thead>
<tr>
<th></th>
<th>WHITE</th>
<th>AFRICAN AMERICAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Labeled Higher Risk, But Didn't Re-Offend</td>
<td>23.5%</td>
<td>44.9%</td>
</tr>
<tr>
<td>Labeled Lower Risk, Yet Did Re-Offend</td>
<td>47.7%</td>
<td>28.0%</td>
</tr>
</tbody>
</table>

Overall, Northpointe’s assessment tool correctly predicts recidivism 61 percent of the time. But blacks are almost twice as likely as whites to be labeled a higher risk but not actually re-offend. It makes the opposite mistake among whites: They are much more likely than blacks to be labeled lower risk but go on to commit other crimes. *(Source: ProPublica analysis of data from Broward County, Fla.)*

Regulation is on its way

On 8 April 2019, the High-Level Expert Group on AI presented the **Ethics Guidelines for Trustworthy Artificial Intelligence**.

Trustworthy means:

- Lawful
- Ethical
- Robust

“[T]he views expressed in this document reflect the opinion of the AI HLEG and may not in any circumstances be regarded as reflecting an official position of the European Commission.”

Seven key requirements

Evaluate and address these continuously throughout the AI system’s lifecycle, via:

- **Technical methods**
  e.g., Constraints in the software architecture, embedded in design and implementation. Explanation functionality. Deliberate testing and validation. Measure algorithm quality indicators.

- **Non-technical methods**
  e.g., Regulations, code of conduct, standardization, certification, governance, education, awareness, stakeholder participation, diversity in design teams.
Demand *for* and demands *on* Machine Learning software

A **challenge** for Software Engineering

An **opportunity** for Software Engineering **Research**
Software Engineering practices in the age of ML

How are software engineering practices **impacted** by incorporation of ML components in software systems?

What new practices are being **proposed** by researchers and practitioners?

To what extent are practices **adopted** by engineering teams?

What are the **effects** of practices adoption on the quality of systems that incorporate ML components?
Investigating ML engineering practices

- **Review literature**
- **Create practice catalog**
- **Survey adoption and effects**
- **Interview practitioners**
- **Add trustworthiness**
- **Add AutoML architecture**
- **Add tactics linked to quality aspects**

Academic and grey literature

- **400+ practitioners**

Serban et al, “Adoption and effects of software engineering best practices in machine learning”, ESEM 2020

- **Awesome reading list**
  - 29 practices in fixed format
  - **ranking** of practices and links to effects
  - “State of ML engineering practices” report
  - +14 practices and link to seven requirements
  - “State of AutoML” report
Online catalog of engineering practices for ML

Originally, 29 practices. Now grown to 45.

Grouped into 6 categories.

- Intent
- Motivation
- Applicability
- Description
- Adoption
- Related practices
- References

Ranked on difficulty

basic  medium  advanced
Use Sanity Checks for All External Data Sources

January, 2021 • Alex Serban, Koen van der Blom, Joost Visser

Intent
Avoid invalid or incomplete data being processed.

Motivation
Data is at the heart of any machine learning model. Therefore, avoiding data errors is crucial for model quality.

Applicability
Data quality control should be applied to any machine learning application.

Description
Whenever external data sources are used, or data is collected that may be incomplete or ill formatted, it is important to verify the data quality. Invalid or incomplete data may cause outages in production or lead to inaccurate models.

Start by checking simple data attributes, such as:
- data types,
- missing values,
- data min. or max. values,
- histograms of continuous values,

and gradually include more complex data statistics, such as the ones recommended here.

Missing data can also be substituted using data imputation; such as imputation by zero, mean, median, random values, etc.

Also, make sure the data verification scripts are reusable and can be later integrated in any processing pipeline.
Measuring practice adoption

Survey among teams building software that incorporates ML components.

Questions:

- **General**
  ex. Team size, team experience, country, kind of organization, type of data, tools used.

- **Practices**
  ex. “Our process for deploying our ML model is fully automated.”

- **Effects**
  ex. “We are able to easily and precisely reproduce past behavior of our models and applications.”

- Not at all
- Partially
- Mostly
- Completely
Tech companies lead practice adoption

The adoption of best practices by tech companies is higher than by non-tech companies, governmental organizations, and research labs.
Practice adoption increases with team size and experience.
ML-specific practices are adopted slightly more than general Software Engineering practices

Among ML teams, the adoption of ML-specific practices is highest, followed by general Software Engineering (SE) practices and SE practices adapted to ML.
Use Sanity Checks for All External Data Sources

January, 2021 • Alex Serban, Koen van der Blom, Joost Visser

Intent
Avoid invalid or incomplete data being processed.

Motivation
Data is at the heart of any machine learning model. Therefore, avoiding data errors is crucial for model quality.

Applicability
Data quality control should be applied to any machine learning application.

Description
Whenever external data sources are used, or data is collected that may be incomplete or ill formatted, it is important to verify the data quality. Invalid or incomplete data may cause outages in production or lead to inaccurate models.

Start by checking simple data attributes, such as:
- data types,
- missing values,
- data min. or max. values,
- histograms of continuous values,

and gradually include more complex data statistics, such as the ones recommended [here](#).

Missing data can also be substituted using data imputation; such as imputation by zero, mean, median, random values, etc.

Also, make sure the data verification scripts are reusable and can be later integrated in any processing pipeline.
Example practice

Title

Nr • Category • Difficulty

- Intent
- Motivation
- Applicability
- Description
- Adoption
- Related practices
- References

- Not at all
- Partially
- Mostly
- Completely
Example practice

Title

Nr • Category • Difficulty

- Intent
- Motivation
- Applicability
- Description
- Adoption
- Related practices
- References

Related
- Check that Input Data is Complete, Balanced and Well Distributed
- Write Reusable Scripts for Data Cleaning and Merging

Read more
- Data management challenges in production machine learning
- ML Ops: Machine Learning as an engineered disciplined
### 29 practices, ranked

Capture the training objective in a metric that is easy to measure and understand
Share a clearly defined training objective within the team
Use versioning for data, models, configurations and training scripts
Continuously measure model quality and performance
Write reusable scripts for data cleaning and merging
Enable parallel training experiments
Share status and outcomes of experiments within the team
Use a collaborative development platform
Work against a shared backlog
Communicate, align and collaborate with multidisciplinary team members
Ensure data labeling is performed in a strictly controlled process
Continuously monitor the behaviour of deployed models
Enable automatic roll backs for production models
Make data sets available on shared infrastructure
Automate model deployment
Use continuous integration
Perform checks to detect skews between models
Check that input data is complete, balanced and well distributed
Log production predictions with the model’s version and input data
Peer review training scripts
Enforce fairness and privacy
Use sanity checks for all external data sources
Test all feature extraction code
Use static analysis to check code quality
Enable shadow deployment
Automate hyper-parameter optimisation and model selection
Run automated regression tests
Actively remove or archive features that are not used
Assign an owner to each feature and document its rationale

Practices are ranked by the average of: their rank on Completely, their rank on Completely+Mostly, and their rank on Completely+Mostly+Partially.
Most adopted practices

Practices related to measurement and versioning are widely adopted.

The top 4 adopted practices are all related to model training.

Top 5

1. Capture the training objective in a metric that is easy to measure and understand
2. Share a clearly defined training objective within the team
3. Use versioning for data, model, configurations and training scripts
4. Continuously measure model quality and performance
5. Write reusable scripts for data cleaning and merging
The two most neglected practices are related to feature management.

Outside research, Automated ML through automated optimisation of hyper-parameters and model selection, is not (yet) widely applied.

Bottom 5

1. Assign an owner to each feature and document its rationale
2. Actively remove or archive features that are not used
3. Run automated regression tests
4. Automate hyper-parameter optimisation and Model Selection
5. Enable shadow deployment
Measuring effects of practice adoption

For four effects, we hypothesized a relation with a specific selection of practices.

- **Linear regression**
  Confirmed hypotheses.

- **Non-linear regression – Random Forest**
  Demonstrated non-linear influence.

- **Importance of each practice – Shapley**
  Some very important practices have low adoption.

<table>
<thead>
<tr>
<th>Effects</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agility</td>
<td>The team can quickly experiment with new data and algorithms, and quickly assess and deploy new models</td>
</tr>
<tr>
<td>Software Quality</td>
<td>The software produced is of high quality (technical and functional)</td>
</tr>
<tr>
<td>Team Effectiveness</td>
<td>Experts with different skill sets (e.g., data science, software development, operations) collaborate efficiently</td>
</tr>
<tr>
<td>Traceability</td>
<td>Outcomes of production models can easily be traced back to model configuration and input data</td>
</tr>
</tbody>
</table>
Different practices, different outcomes

Analysis of survey responses shows that desired outcomes such as traceability, agility, team effectiveness, and software quality are each related to specific sets of practices.

Per desired outcome, we list the three practices with the largest influence.

### Agility
1. Automate model deployment
2. Communicate, align, and collaborate with multidisciplinary team members
3. Enable parallel training experiments

### Traceability
1. Log production predictions with the model’s version and input data
2. Continuously monitor the behaviour of deployed models
3. Use versioning for data, model, configurations and training scripts

### Team Effectiveness
1. Work against a shared backlog
2. Use a collaborative development platform
3. Share a clearly defined training objective within the team

### Software Quality
1. Use continuous integration
2. Run automated regression tests
3. Use static analysis to check code quality
Key findings

Tech companies are leading in adoption of ML engineering best practices.

Larger and more experienced teams tend to adopt more practices.

General software engineering practices enjoy slightly lower adoption than specific machine learning practices.

Best practices for feature management are the least well adopted.

Desired outcomes such as traceability, agility, effectiveness, and quality are each related to specific sets of practices.

From 2020 global survey on adoption of 29 practices, among 350 teams.
Software Engineering practices in the age of ML

How are software engineering practices impacted by incorporation of ML components in software systems?

What new practices are being proposed by researchers and practitioners?

To what extent are practices adopted by engineering teams?

What are the effects of practices adoption on the quality of systems that incorporate ML components?

Answers lead to new questions ...

- **Trustworthiness**
  More practices? Link to requirements?

- **Architecture**
  Practices as tactics to reach architectural goals.

- **AutoML**
  Transfer from research to broad adoption?
Back to the

Seven key requirements

Evaluate and address these continuously throughout the AI system’s lifecycle, via:

- **Technical methods**
  e.g., Constraints in the software architecture, embedded in design and implementation. Explanation functionality. Deliberate testing and validation. Measure algorithm quality indicators.

- **Non-technical methods**
  e.g., Regulations, code of conduct, standardization, certification, governance, education, awareness, stakeholder participation, diversity in design teams.
New practices, mapped to trustworthiness requirements

- **Lawful**
  - Human agency and oversight
  - Technical robustness and safety
  - Privacy and data governance
  - Transparency
  - Diversity, non-discrimination, fairness
  - Societal, environmental well-being

- **Ethical**
  - Accountability

- **Robust**

   **T1** - Test for social bias in training data
   **T2** - Prevent discriminatory data attributes as model features
   **T3** - Use privacy-preserving ML techniques
   **T4** - Employ interpretable models whenever possible
   **T5** - Assess and manage subgroup bias
   **T6** - Assure application security
   **T7** - Provide audit trails
   **T8** - Decide trade-offs through an established team process
   **T9** - Establish responsible AI values
   **T10** - Perform risk assessments
   **T11** - Inform users on ML usage
   **T12** - Explain results and decision to users
   **T13** - Provide safe channels to raise concerns
   **T14** - Have your application audited
Back to

Demand *for* and demands *on* Machine Learning software

A **challenge** for Software Engineering

An **opportunity** mission for Software Engineering **Research**

- Tools, (automated) processes
- Quality instruments
- Body of knowledge
- Evidence for best practices
- Input for regulators
- Education
Take away

Software that incorporates Machine Learning (or other AI) challenges traditional software engineering practices, due to data intensity, inherent uncertainty, and iterative empirical design.

Demand for robust and responsible development and use are not unique to ML, but become more acute.

Engineering practices are being modified and developed at a quick pace. Adoption varies and effects are not well-understood.

Software Engineering researchers should embrace the challenge of ML, investigate and enhance practice development.
You can help

Take the Survey

If you have not done so yet, please take our 10-min survey!

We will use your answers for our next report on the State of Engineering Practices for Machine Learning.

https://se-ml.github.io/survey
We reviewed scientific and popular literature to identify recommended practices. Check out this Awesome List with relevant literature.

The best practices that we identified are described in more detail in this Catalogue of ML Engineering Best Practices.

Full details of the methodology behind our survey are described in a scientific article. Read the preprint here.

Visit our project website for more details, to take the survey yourself, and to stay up-to-date with our latest results.
Team

https://se-ml.github.io/members/

LIACS, Leiden University, The Netherlands
ICIS, Radboud University, The Netherlands
University of British Columbia, Canada