USC University of Southern California

Automating Machine Learning Model Checking

Victoria Stodden
Adhithya Bhaskar
Department of Industrial and Systems Engineering

REAL@USC-META CENTER First Anniversary Conference
University of Southern California
Monday, October 31, 2022

Agenda

1. ML Model Checking Value Proposition

2. Leveraging Steps in the Community

3. ReproScreener: A Tool for ML Model Checking (work in progress)

Model Checking Value Proposition: Potential Gains

Scalability to large-scale complex models

- e.g. integration of multiple large data sources, deployment on large scale / high throughput computing systems.

Verification of model performance

- Consistency of results / predictions across time, systems, data.

Transparency / interpretability

Efficiency in resource use / Discovery speedup

- Computing systems: compute time, appropriate benchmarking; Engineers: code re-use, reduction in effort duplication

Model Checking Value Proposition: Potential Drawbacks

Increased overhead

- ○ Additional computational step(s) in model building, deployment

Culture change

- ○ Increased emphasis on reproducibility, verification, correctness in ML models

Hewing to the wrong goals

Community Efforts

ML model publication standards

- ○ Gunderson (AAAI 2018)
- ○ Pineau (JMLR 2020)

Formal Verification for ML models

- ○ Abate (MEMOCODE 2017)
- ○ Urban and Miné (arxiv 2021)

Research Publication Standards

- ● Willis and Stodden (HDSR 2020)
- ● ML Commons (Github)
- ● National Academies Reproducibility Report (NASEM 2019)

…

ML Model Checking: A Novel Approach

- Previous work in AI involves applying formal techniques using SMT (satisfiability modulo theory) solving, constraint solving, or abstract numerical interpretation.

- We exploit specialized features of ML pipelines and propose a *reproducibility* approach (NASEM 2019):
  - Exposure of methods
  - Well-defined guarantees in correctness of results

ML Model Checking: ReproScreener

- Automate ML model checking *at the point of publication*, to provide guarantees on correctness, scalability, and transparency.

- ReproScreener software tool verifies criteria and provides feedback.

ReproScreener Development (work in progress)

1. Create testbed of ML/AI publications (arχiv ML.stat and CS.GL)
2. Implement existing “Gunderson” ML model Criteria (Gunderson 2018)
3. Label testbed publications manually for Criteria
4. Run Reproscreener on testbed
5. Extend Criteria based on empirical findings

Criteria Adapted from “Gunderson” (2018)

- Method Transparency:
  - Problem; Objective, Goal; Research Method; Pseudo Code.
- Data Transparency:
  - Training, Test, Validation Data; Results.

We include:

- Code Transparency:
  - Github, Bitbucket, Gitlab

Preliminary Results (Work in Progress)

problem : 0.94

objective : 0.8

research_method : 0.88

research_questions : 0.84

pseudocode : 0.5

training_data : 0.36

validation_data : 0.02

test_data : 0.12

results : 0.0

hypothesis : 0.42

prediction : 0.68

method_source_code : 0.3

hardware_specifications : 0.0

software_dependencies : 0.0

experiment_setup : 0.36

experiment_source_code : 0.0

affiliation : 0.3

Extending “Gunderson” Criteria

Implemented the Gunderson criteria on 8 publications and identified the following gaps:

- Measure only the availability of code and data.
- Do not account for any errors during the reproducibility process.
- Partial reproduction of results are not captured.

Extending “Gunderson” Criteria

ReproScreener:

- Automatically checks specific guidances to improve correctness of ML models.
- Predicts (error bounds), captures and identifies differences in model output at scale (due to architecture, non-determinism, etc.)
- Enables comparison of model code through
  - Checking for modularity, file structure, dependencies.
  - Checking for steps/scripts to create figures & visualizations.
  - Tracking model benchmarks and provenance.

- Real world case studies to demonstrate ReproScreener’s functionality

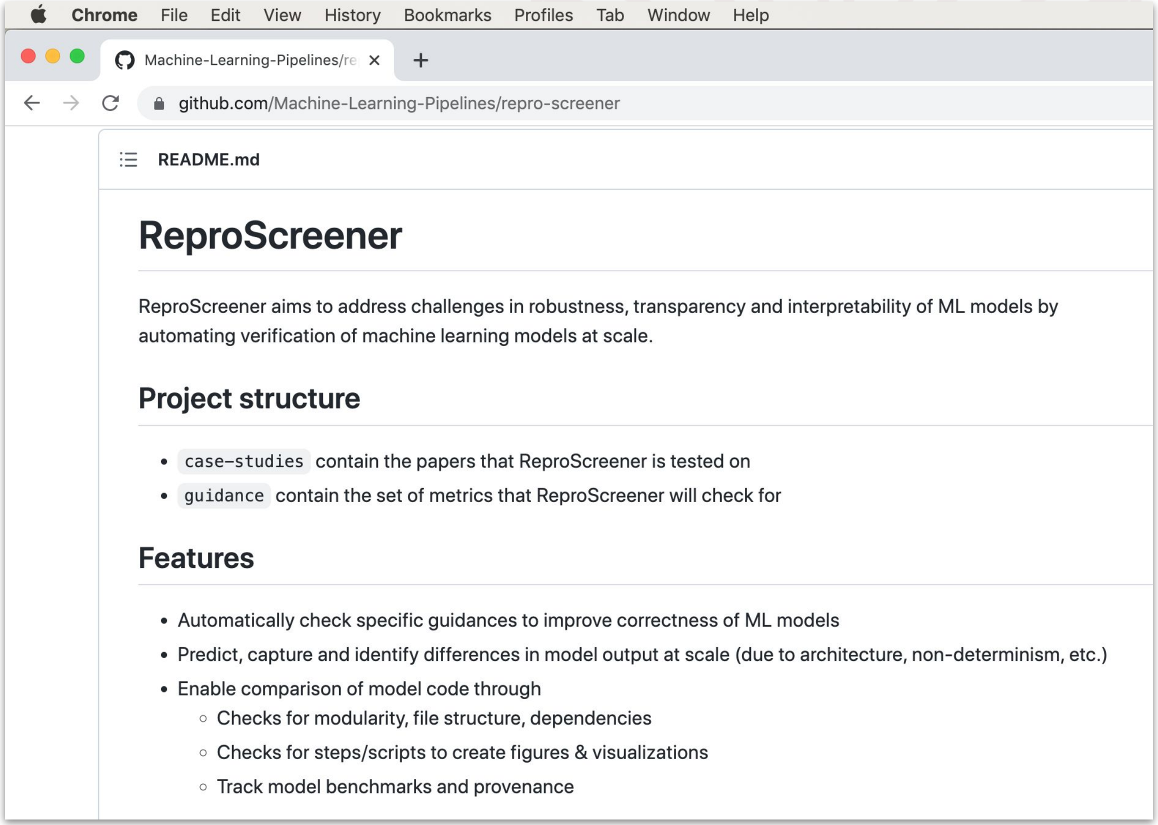
Open Source Development (work in progress)

ReproScreener:

- Automatically checks specific guidances to improve correctness of ML models.
- Predicts (error bounds), captures and identifies differences in model output at scale (due to architecture, non-determinism, etc.)
- Enables comparison of model code through
  - Checking for modularity, file structure, dependencies.
  - Checking for steps/scripts to create figures & visualizations.
  - Tracking model benchmarks and provenance.

- Real world case studies to demonstrate ReproScreener's functionality

Open Source Development (work in progress)


Chrome
File
Edit
View
History
Bookmarks
Profiles
Tab
Window
Help
Machine-Learning-Pipelines/re
github.com/Machine-Learning-Pipelines/repro-screener
README.md

ReproScreener

ReproScreener aims to address challenges in robustness, transparency and interpretability of ML models by automating verification of machine learning models at scale.

Project structure

- `case-studies` contain the papers that ReproScreener is tested on
- `guidance` contain the set of metrics that ReproScreener will check for

Features

- Automatically check specific guidances to improve correctness of ML models
- Predict, capture and identify differences in model output at scale (due to architecture, non-determinism, etc.)
- Enable comparison of model code through
  - Checks for modularity, file structure, dependencies
  - Checks for steps/scripts to create figures & visualizations
  - Track model benchmarks and provenance

Thank you

Adhithya Bhaskar

This material is based upon work supported by the
REAL@USC-META Center
and National Science Foundation Grant No 2138776

USC