On the Relationship Between Code Verifiability and Understandability

Kobi Feldman[b], **Martin Kellogg**[a], Oscar Chaparro[b]
[a]New Jersey Institute of Technology [b]College of William & Mary

Common Wisdom

easier to verify -> **easier to understand**

Common Wisdom

"**rewrite your code to be simpler** for the checker to analyze; this is likely to make it easier for people to understand, too."

- Checker Framework manual

easier to understand

Common Wisdom

"**rewrite your code to be simpler** for the checker to analyze; this is likely to make it **easier for people to understand**, too"

- Checker Framework manual

easier to understand

Common Wisdom

"**rewrite your code to be simpler** for the checker to analyze; this is likely to make it **easier for people to understand**, too"

- Checker Framework manual

easier to understand

"**success in checking the consistency of the specifications and the code** will depend on... the **complexity and style** in which the code and specifications are written"

- OpenJML manual

Common Wisdom

easier to verify -> **easier to understand**

Common Wisdom

easier to verify -> **easier to understand**

But **how do we know** that this is true?

Common Wisdom

easier to verify -> **easier to understand**

But **how do we know** that this is true?

Our goal: **fill this gap** in the literature with an empirical study

Does it matter?

- An empirical study's results must be **actionable**

Does it matter?

- An empirical study's results must be **actionable**
- So, what are the **implications** if our hypothesis is correct?

Does it matter?

- An empirical study’s results must be **actionable**
- So, what are the **implications** if our hypothesis is correct?
- Our hypothesis:
  - ***“There is a correlation between code that is hard to verify and code that is hard for humans to understand.”***

Implications

> ***"There is a correlation between code that is hard to verify and code that is hard for humans to understand."***

Implications

> ***"There is a correlation between code that is hard to verify and code that is hard for humans to understand."***

- For the **builders** of verification tools:

Implications

> *"There is a correlation between code that is hard to verify and code that is hard for humans to understand."*

- For the **builders** of verification tools:
  - we are giving good advice to our users (yay!)

Implications

"There is a correlation between code that is hard to verify and code that is hard for humans to understand."

- For the builders of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)

Implications

> *"There is a correlation between code that is hard to verify and code that is hard for humans to understand."*

- For the builders of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)
- For the users of verification tools:

Implications

> *"There is a correlation between code that is hard to verify and code that is hard for humans to understand."*

- For the builders of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)
- For the users of verification tools:
  - refactor to avoid warnings

Implications

> *"There is a correlation between code that is hard to verify and code that is hard for humans to understand."*

- For the **builders** of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)
- For the **users** of verification tools:
  - refactor to avoid warnings

Auxiliary benefit of verification:
points to hard-to-understand code

Implications

> *"There is a correlation between code that is hard to verify and code that is hard for humans to understand."*

- For the builders of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)
- For the users of verification tools:
  - refactor to avoid warnings
- For code understanding researchers:

Implications

"There is a correlation between code that is hard to verify and code that is hard for humans to understand."

- For the builders of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)
- For the users of verification tools:
  - refactor to avoid warnings
- For code understanding researchers:
  - there is a semantic component to human code understanding

Implications

> *"There is a correlation between code that is hard to verify and code that is hard for humans to understand."*

- For the **builders** of verification tools:
  - we are giving good advice to our users (yay!)
  - error messages should suggest semantically-equivalent code that would verify (new research direction!)
- For the **users** of verification tools:
  - refactor to avoid warnings
- For **code understanding researchers**:
  - there is a semantic component to human code understanding
  - explains ineffectiveness of traditional, syntactic metrics like cyclomatic complexity

Empirical study design

> ***"There is a correlation between code that is hard to verify and code that is hard for humans to understand."***

- **Problem**: neither of these are easy to measure directly

Empirical study design

> ***"There is a correlation between code that is hard to verify and code that is hard for humans to understand."***

- **Problem**: neither of these are easy to measure directly
  - must use **proxies**

Proxy for verifiability

Proxy for verifiability

Warnings on **unannotated, correct** code snippets

Proxy for verifiability

Warnings on **unannotated, correct** code snippets

- “unannotated” = “no specifications”

Proxy for verifiability

Warnings on **unannotated, correct** code snippets

- “unannotated” = “no specifications”
    - but still trying to prove e.g., absence of buffer overflows

Proxy for verifiability

Warnings on **unannotated, correct** code snippets

- "unannotated" = "no specifications"
  - but still trying to prove e.g., absence of buffer overflows
- "correct" so that no warnings correspond to real bugs

Proxy for verifiability

Warnings on **unannotated, correct** code snippets

- “unannotated” = “no specifications”
  - but still trying to prove e.g., absence of buffer overflows
- “correct” so that no warnings correspond to real bugs
  - that is, all warnings are **false positives**

Choosing verifiers

- We selected **four** “verifiers”:

CHECKER framework

Infer

JML

OpenJML

Java Typestate Checker

Choosing verifiers

- We selected **four** “verifiers”:

All tools have **sound cores**: internally, they try to construct a proof (= “do verification”).

Proxy for understandability

Proxy for understandability

Metrics for understandability from **prior work**

Proxy for understandability

Metrics for understandability from **prior work**

- this is a **pragmatic** decision: don’t run another human study!

Proxy for understandability

Metrics for understandability from **prior work**

- this is a **pragmatic** decision: don’t run another human study!
  - but studies in the literature don’t use the same set of metrics

Prior studies

- we used **6** prior studies

Prior studies: descriptive stats

Table 1: Datasets (DSs) of code snippets and understandability measurements/metrics used in our study. The metrics types are "C" for correctness, "R" for ratings, "T" for time, and "P" for physiological.

DS	Snippets	NCLOC	Participants	Understandability Task	Understandability Metrics	Meas.
1 [81]	23 CS algorithms	6 - 20	41 students	Determine prog. output	**C:** *correct_output_rating* (3-level correctness score for program output) **R:** *output_difficulty* (5-level difficulty score for determining program output) **T:** *time_to_give_output* (seconds to read program and answer a question)	2,829
2 [70]	12 CS algorithms	7 - 15	16 students	Determine prog. output	**P:** *brain_deact_31ant* (deactivation of brain area BA31ant) **P:** *brain_deact_31post* (deactivation of brain area BA31post) **P:** *brain_deact_32* (deactivation of brain area BA32) **T:** *time_to_understand* (seconds to understand program within 60 secs.)	228
3 [16]	100 OSS methods	5 - 13	121 students	Rate prog. readability	**R:** *readability_level* (5-level score for readability/ease to understand)	12,100
6 [77]	50 OSS methods	18 - 75	50 students and 13 developers	Rate underst./answer Qs	**R:** *binary_understandability* (0/1 program understandability score) **C:** *correct_verif_questions* (% of correct answers to verification questions) **T:** *time_to_understand* (seconds to understand program)	1,197
9 [14]	10 OSS methods	10 - 34	104 students	Rate read./complete prog.	**C:** *gap_accuracy* (0/1 accuracy score for filling in program blanks) **R:** *readability_level_ba* (5-level avg. score for readability b/a code completion) **R:** *readability_level_before* (5-level score for readability before code completion) **T:** *time_to_read_complete* (avg. seconds to rate readability and complete code)	2,600
F [68]	16 CS algorithms	7 - 19	19 students	Determine prog. output	**P:** *brain_deact_31*(deactivation of brain area BA31) **P:** *brain_deact_32* (deactivation of brain area BA32) **R:** *complexity_level* (score for program complexity) **C:** *perc_correct_output* (% of subjects who correctly gave program output) **T:** *time_to_understand* (seconds to understand program within 60 seconds)	631

Prior studies: descriptive stats

Table 1: Datasets (DSs) of code snippets and understandability measurements/metrics used in our study. The metrics types are "C" for correctness, "R" for ratings, "T" for time, and "P" for physiological.

snippets/study has a wide range

DS	Snipp			ity Task	Understandability Metrics	Meas.
1 [81]	23 CS alg			output	**C:** *correct_output_rating* (3-level correctness score for program output) **R:** *output_difficulty* (5-level difficulty score for determining program output) **T:** *time_to_give_output* (seconds to read program and answer a question)	2,829
2 [70]	12 CS algorithms	7 - 15	16 students	Determine prog. output	**P:** *brain_deact_31ant* (deactivation of brain area BA31ant) **P:** *brain_deact_31post* (deactivation of brain area BA31post) **P:** *brain_deact_32* (deactivation of brain area BA32) **T:** *time_to_understand* (seconds to understand program within 60 secs.)	228
3 [16]	100 OSS methods	5 - 13	121 students	Rate prog. readability	**R:** *readability_level* (5-level score for readability/ease to understand)	12,100
6 [77]	50 OSS methods	18 - 75	50 students and 13 developers	Rate underst./answer Qs	**R:** *binary_understandability* (0/1 program understandability score) **C:** *correct_verif_questions* (% of correct answers to verification questions) **T:** *time_to_understand* (seconds to understand program)	1,197
9 [14]	10 OSS methods	10 - 34	104 students	Rate read./complete prog.	**C:** *gap_accuracy* (0/1 accuracy score for filling in program blanks) **R:** *readability_level_ba* (5-level avg. score for readability b/a code completion) **R:** *readability_level_before* (5-level score for readability before code completion) **T:** *time_to_read_complete* (avg. seconds to rate readability and complete code)	2,600
F [68]	16 CS algorithms	7 - 19	19 students	Determine prog. output	**P:** *brain_deact_31*(deactivation of brain area BA31) **P:** *brain_deact_32* (deactivation of brain area BA32) **R:** *complexity_level* (score for program complexity) **C:** *perc_correct_output* (% of subjects who correctly gave program output) **T:** *time_to_understand* (seconds to understand program within 60 seconds)	631

Prior studies: descriptive stats

Table 1: Datasets (DSs) of code snippets and understandability measurements/metrics used in our study. The metrics types are "C" for correctness, "R" for ratings, "T" for time, and "P" for physiological.

DS	Snippets	NCLOC	Participants	Understandability Task	Understandability Metrics	Meas.
1 [81]	2[illegible] CS algorithms	6[illegible]	[illegible]	[illegible]	[illegible]*ating* (3-level correctness score for program output) [illegible] (5-level difficulty score for determining program output) [illegible]*put* (seconds to read program and answer a question)	2,829
2 [70]	1[illegible] CS algorithms	7[illegible]	[illegible]	[illegible]	[illegible]*t* (deactivation of brain area BA31ant) [illegible]*st* (deactivation of brain area BA31post) [illegible]deactivation of brain area BA32) **T:** *time_to_understand* (seconds to understand program within 60 secs.)	228
3 [16]	1[illegible]0 OSS methods	5 - 13	121 students	Rate prog. readability	**R:** *readability_level* (5-level score for readability/ease to understand)	12,100
6 [77]	5[illegible] OSS methods	18 - 75	50 students and 13 developers	Rate underst./answer Qs	**R:** *binary_understandability* (0/1 program understandability score) **C:** *correct_verif_questions* (% of correct answers to verification questions) **T:** *time_to_understand* (seconds to understand program)	1,197
9 [14]	1[illegible] OSS methods	10 - 34	104 students	Rate read./complete prog.	**C:** *gap_accuracy* (0/1 accuracy score for filling in program blanks) **R:** *readability_level_ba* (5-level avg. score for readability b/a code completion) **R:** *readability_level_before* (5-level score for readability before code completion) **T:** *time_to_read_complete* (avg. seconds to rate readability and complete code)	2,600
F [68]	1[illegible] CS algorithms	7 - 19	19 students	Determine prog. output	**P:** *brain_deact_31*(deactivation of brain area BA31) **P:** *brain_deact_32* (deactivation of brain area BA32) **R:** *complexity_level* (score for program complexity) **C:** *perc_correct_output* (% of subjects who correctly gave program output) **T:** *time_to_understand* (seconds to understand program within 60 seconds)	631

mix of classic algorithms and open source

Prior studies: descriptive stats

Table 1: Datasets (DSs) of code snippets and understandability measurements/metrics used in our study. The metrics types are "C" for correctness, "R" for ratings, "T" for time, and "P" for physiological.

small snippets

DS	Snippets	NCLOC	Pa		Understandability Metrics	Meas.
1 [81]	23 CS algorithms	6 - 20	41		*correct_output_rating* (3-level correctness score for program output) *output_difficulty* (5-level difficulty score for determining program output) *time_to_give_output* (seconds to read program and answer a question)	2,829
2 [70]	12 CS algorithms	7 - 15	16 students	Determine prog. output	**P:** *brain_deact_31ant* (deactivation of brain area BA31ant) **P:** *brain_deact_31post* (deactivation of brain area BA31post) **P:** *brain_deact_32* (deactivation of brain area BA32) **T:** *time_to_understand* (seconds to understand program within 60 secs.)	228
3 [16]	100 OSS methods	5 - 13	121 students	Rate prog. readability	**R:** *readability_level* (5-level score for readability/ease to understand)	12,100
6 [77]	50 OSS methods	18 - 75	50 students and 13 developers	Rate underst./answer Qs	**R:** *binary_understandability* (0/1 program understandability score) **C:** *correct_verif_questions* (% of correct answers to verification questions) **T:** *time_to_understand* (seconds to understand program)	1,197
9 [14]	10 OSS methods	10 - 34	104 students	Rate read./complete prog.	**C:** *gap_accuracy* (0/1 accuracy score for filling in program blanks) **R:** *readability_level_ba* (5-level avg. score for readability b/a code completion) **R:** *readability_level_before* (5-level score for readability before code completion) **T:** *time_to_read_complete* (avg. seconds to rate readability and complete code)	2,600
F [68]	16 CS algorithms	7 - 19	19 students	Determine prog. output	**P:** *brain_deact_31*(deactivation of brain area BA31) **P:** *brain_deact_32* (deactivation of brain area BA32) **R:** *complexity_level* (score for program complexity) **C:** *perc_correct_output* (% of subjects who correctly gave program output) **T:** *time_to_understand* (seconds to understand program within 60 seconds)	631

Prior studies: descriptive stats

Table 1: Datasets (DSs) of code snippets and understandability measurements/metrics used in our study. The metrics types are "C" for correctness, "R" for ratings, "T" for time, and "P" for physiological.

DS	Snippets	NCLOC	Participants	Understandability Task	Understandability Metrics	Meas.
1 [81]	23 CS algorithms	6 - 20	41 students	Determ	**C:** *correct_output_rating* (3-level correctness score for program output) etermining program output) and answer a question)	2,829
2 [70]	12 CS algorithms	7 - 15	16 students	Determ	BA31ant) BA31post) 32) ogram within 60 secs.)	228
3 [16]	100 OSS methods	5 - 13	121 students	Rate prog. readability	**R:** *readability_level* (5-level score for readability/ease to understand)	12,100
6 [77]	50 OSS methods	18 - 75	50 students and 13 developers	Rate underst./answer Qs	**R:** *binary_understandability* (0/1 program understandability score) **C:** *correct_verif_questions* (% of correct answers to verification questions) **T:** *time_to_understand* (seconds to understand program)	1,197
9 [14]	10 OSS methods	10 - 34	104 students	Rate read./complete prog.	**C:** *gap_accuracy* (0/1 accuracy score for filling in program blanks) **R:** *readability_level_ba* (5-level avg. score for readability b/a code completion) **R:** *readability_level_before* (5-level score for readability before code completion) **T:** *time_to_read_complete* (avg. seconds to rate readability and complete code)	2,600
F [68]	16 CS algorithms	7 - 19	19 students	Determine prog. output	**P:** *brain_deact_31*(deactivation of brain area BA31) **P:** *brain_deact_32* (deactivation of brain area BA32) **R:** *complexity_level* (score for program complexity) **C:** *perc_correct_output* (% of subjects who correctly gave program output) **T:** *time_to_understand* (seconds to understand program within 60 seconds)	631

almost all students;
# of participants varies

Prior studies: metrics

- we used 6 prior studies
- 20 metrics:

Prior studies: metrics

- we used 6 prior studies
- 20 metrics:
  - 4 **correctness** (e.g., “% answering a question correctly”)

Prior studies: metrics

- we used 6 prior studies
- 20 metrics:
  - 4 **correctness** (e.g., “% answering a question correctly”)
  - 6 **rating** (e.g., “readability level”)

Prior studies: metrics

- we used 6 prior studies
- 20 metrics:
  - 4 **correctness** (e.g., “% answering a question correctly”)
  - 6 **rating** (e.g., “readability level”)
  - 5 **time** (e.g., “time to read program and answer a question”)

Prior studies: metrics

- we used 6 prior studies
- 20 metrics:
  - 4 **correctness** (e.g., “% answering a question correctly”)
  - 6 **rating** (e.g., “readability level”)
  - 5 **time** (e.g., “time to read program and answer a question”)
  - 5 **physiological** (e.g., brain area deactivation via fMRI)

Meta-analysis

- it is **not obvious** how to combine these metrics

Meta-analysis

- it is **not obvious** how to combine these metrics
- **tempting but wrong** idea: measure correlation for each metric independently, then count correlations

Meta-analysis

- it is **not obvious** how to combine these metrics
- **tempting but wrong** idea: measure correlation for each metric independently, then count correlations
  - a statistical error! ("**vote counting**"):

Meta-analysis

- it is **not obvious** how to combine these metrics
- **tempting but wrong** idea: measure correlation for each metric independently, then count correlations
  - a statistical error! ("**vote counting**"):
    - overweights studies with more metrics
    - doesn't take into account effect sizes

Meta-analysis

- it is **not obvious** how to combine these metrics
- **tempting but wrong** idea: measure correlation for each metric independently, then count correlations
    - a statistical error! (“**vote counting**”):
        - overweights studies with more metrics
        - doesn’t take into account effect sizes
- instead, use **random-effects meta-analysis**

Meta-analysis

- it is **not obvious** how to combine these metrics
- **tempting but wrong** idea: measure correlation for each metric independently, then count correlations
    - a statistical error! ("**vote counting**"):
        - overweights studies with more metrics
        - doesn't take into account effect sizes
- instead, use **random-effects meta-analysis**
    - technique for combining medical studies on different populations and proxies

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation
    - however, our correlations are **not independent**!

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation
  - however, our correlations are **not independent**!
    - each study has **same** subjects, **same** snippets

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation
    - however, our correlations are **not independent**!
        - each study has **same** subjects, **same** snippets
- in meta-analysis, this is the "**unit-of-analysis problem**"

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation
  - however, our correlations are **not independent**!
    - each study has **same** subjects, **same** snippets
- in meta-analysis, this is the "**unit-of-analysis problem**"
  - an **open problem** (!) in statistical methods research

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation
  - however, our correlations are **not independent**!
    - each study has **same** subjects, **same** snippets
- in meta-analysis, this is the "**unit-of-analysis problem**"
  - an **open problem** (!) in statistical methods research
    - we tried some cutting-edge statistical techniques, but their (strong) assumptions weren't satisfied

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate correlation
  - however, our correlations are **not independent**!
    - each study has **same** subjects, **same** snippets
- in meta-analysis, this is the "**unit-of-analysis problem**"
  - an **open problem** (!) in statistical methods research
    - we tried some cutting-edge statistical techniques, but their (strong) assumptions weren't satisfied
    - instead, use **brute force**: combine all metrics for each study into one correlation

Unit-of-analysis problem

- meta-analysis **combines** independent correlations into a single, aggregate c
  - however
    - each

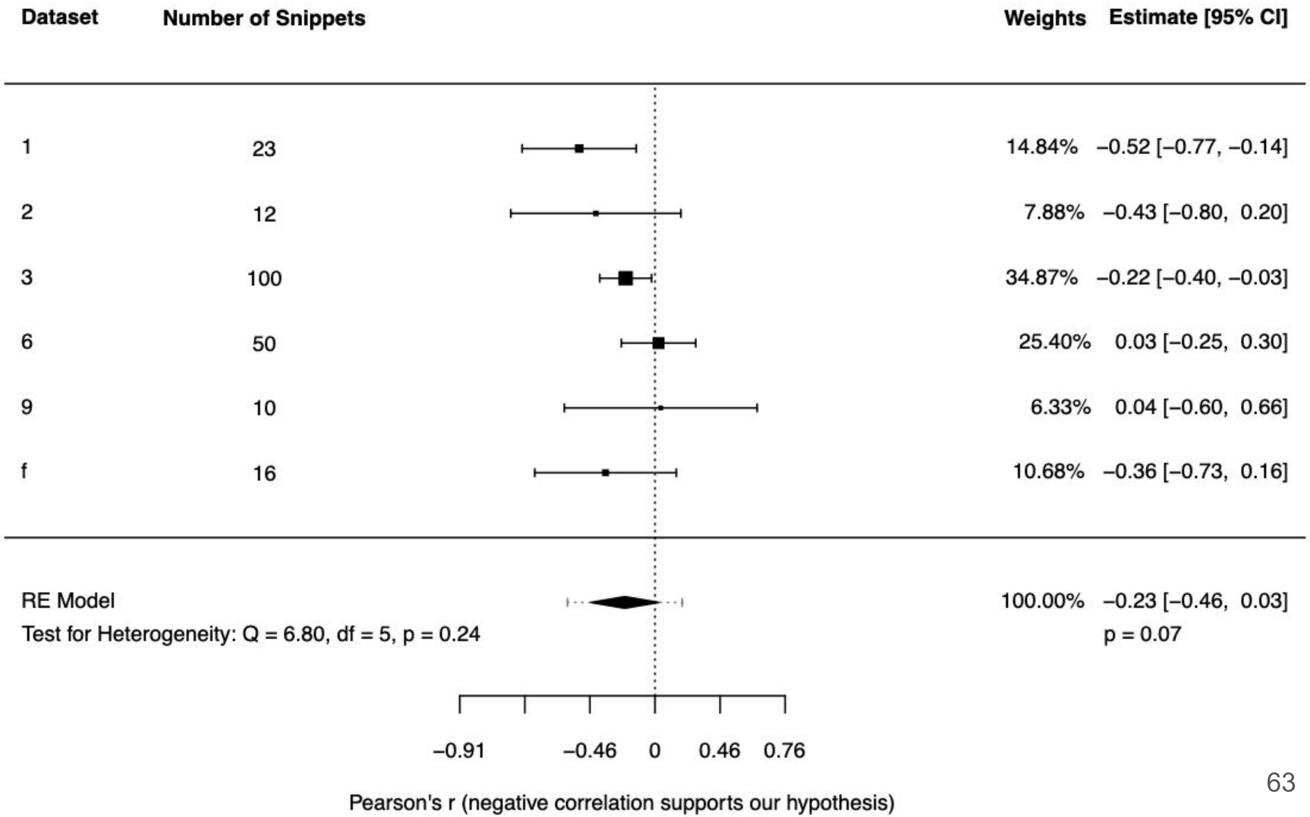
Brute force is **safe**, but **throws away** the benefit of multiple metrics per study

- in meta-analysis, this is the "**unit-of-analysis problem**"
  - an **open problem** (!) in statistical methods research
    - we tried some cutting-edge statistical techniques, but their (strong) assumptions weren't satisfied
    - instead, use **brute force**: combine all metrics for each study into one correlation

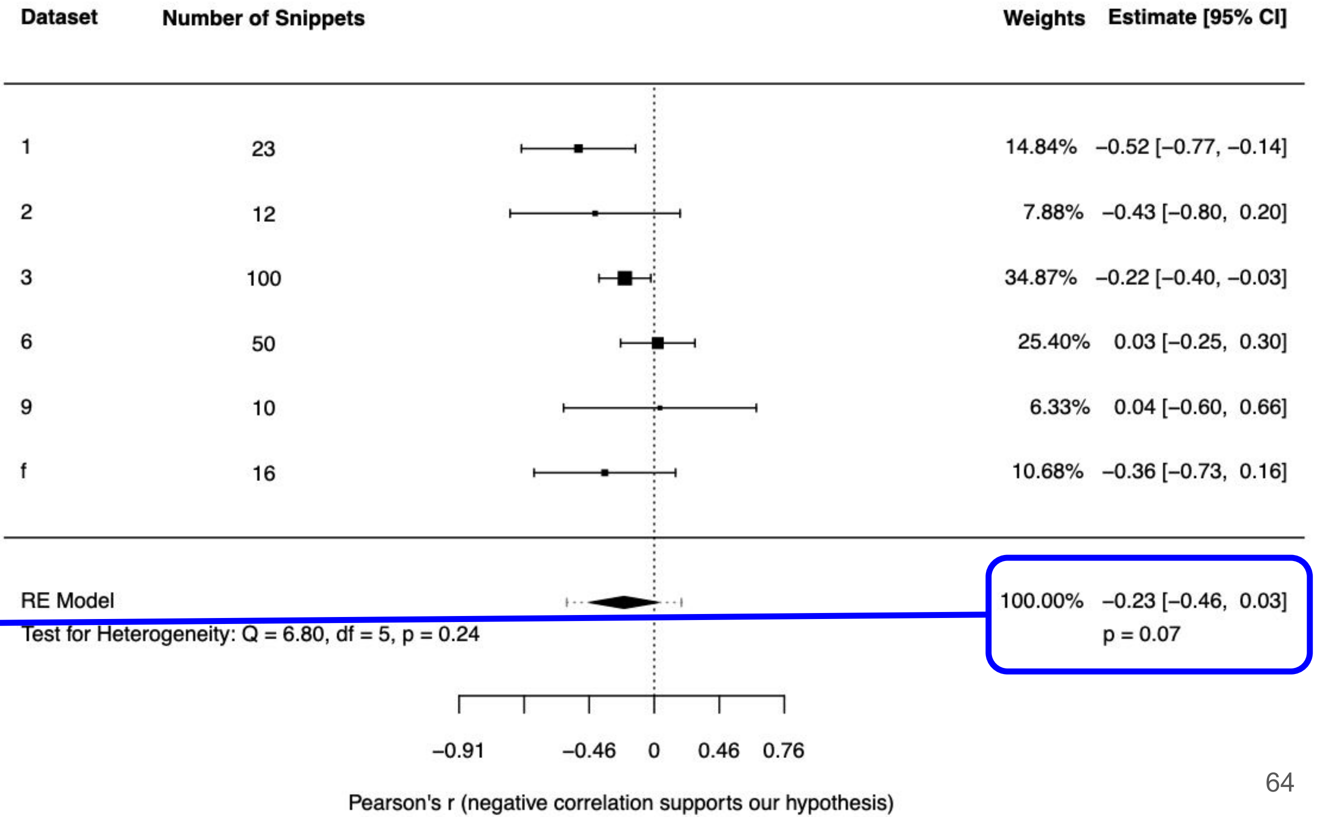
Results

Results: overall

Results: overall



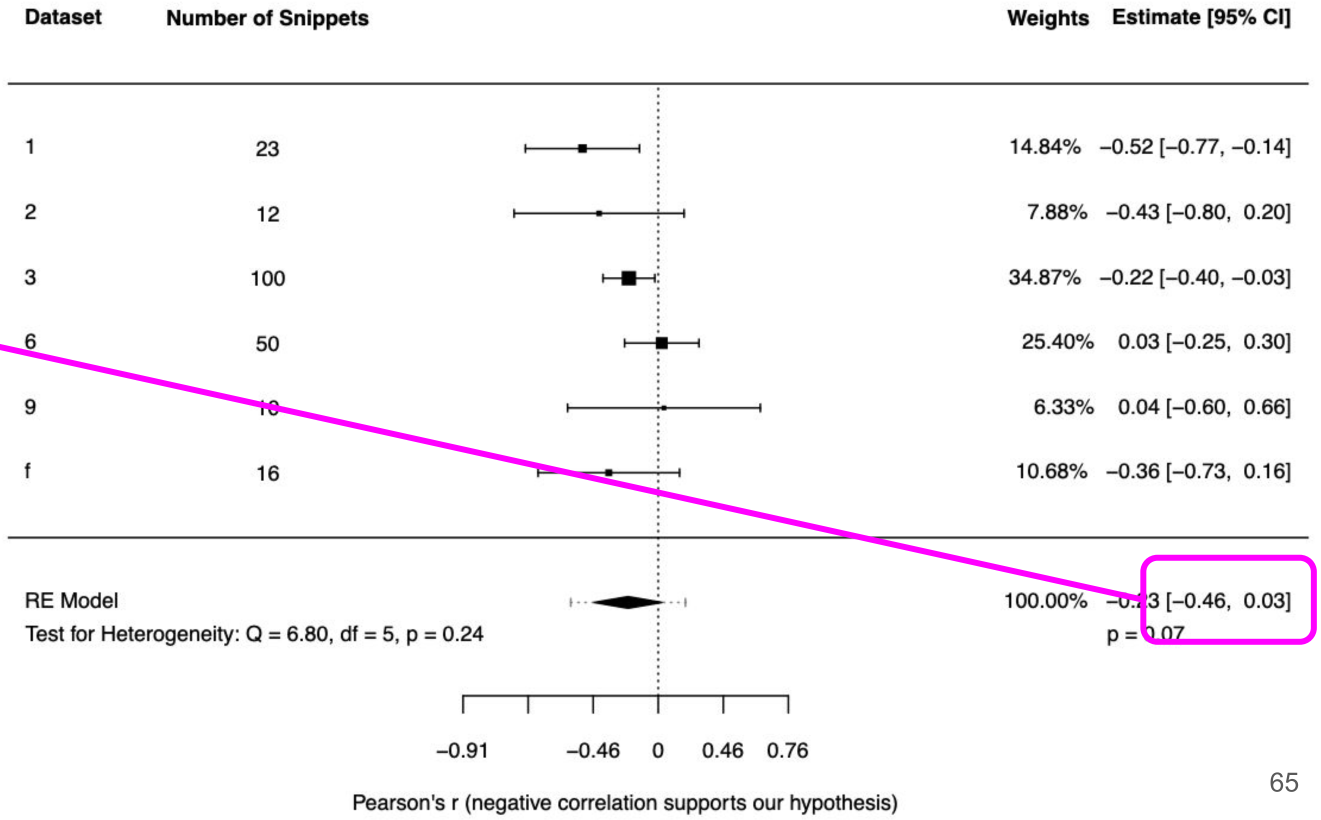
Results: overall



overall correlation of *r*=0.23 (small effect size)

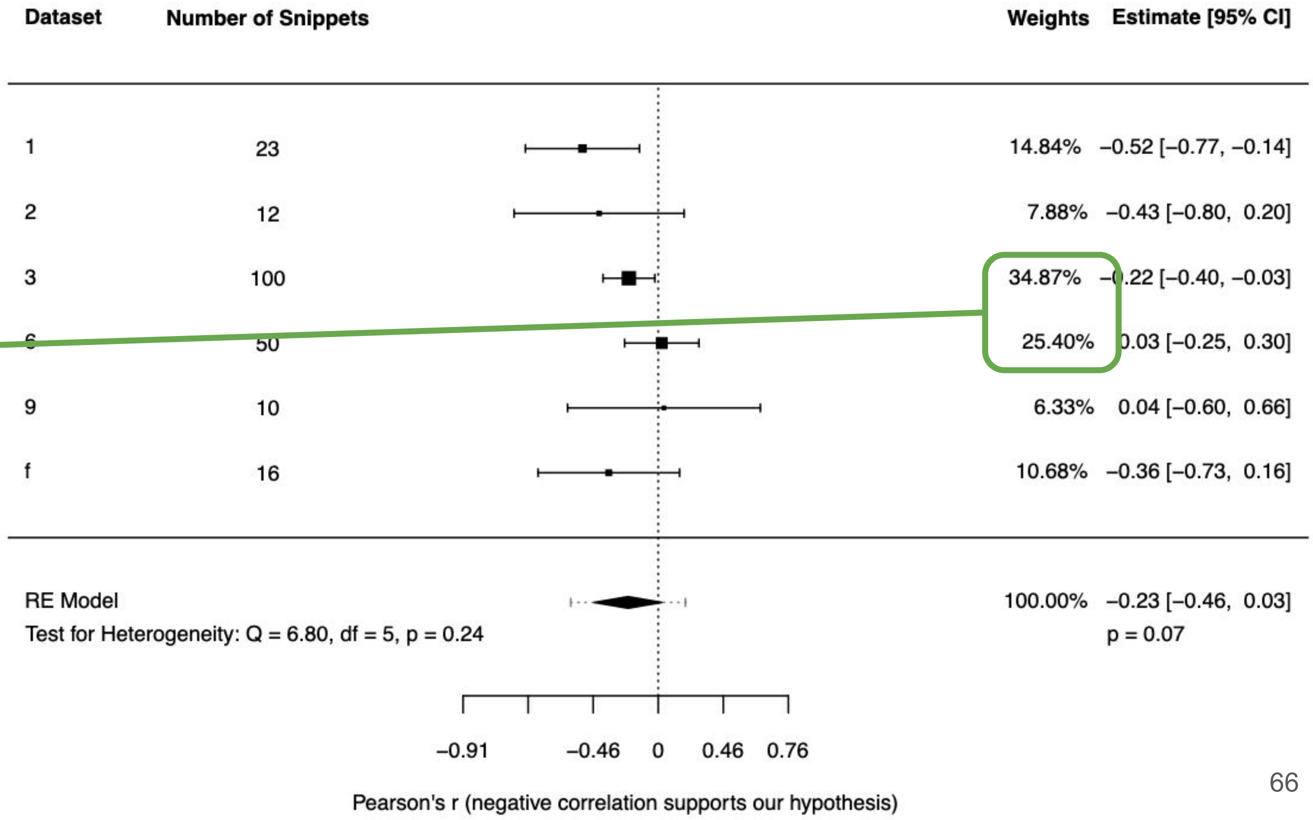
Results: overall

95% confidence interval is wide [-0.46, 0.03], but most of it supports our hypothesis



Results: overall

meta-analysis weights these two datasets (with 50 and 100 snippets) much higher than the others



Results: interpretation

- Our results give **mild but suggestive** support for our hypothesis

Results: interpretation

- Our results give **mild but suggestive** support for our hypothesis
  - especially given our **relatively conservative** statistical methods

Results: interpretation

- Our results give **mild but suggestive** support for our hypothesis
  - especially given our **relatively conservative** statistical methods
- The main limitation preventing us from making stronger conclusions is **the small number of snippets** in prior work

Results: interpretation

- Our results give **mild but suggestive** support for our hypothesis
  - especially given our **relatively conservative** statistical methods
- The main limitation preventing us from making stronger conclusions is **the small number of snippets** in prior work
  - future work: new study with a **larger number of snippets**

Results: secondary analyses

Results: secondary analyses: per-tool

- **per-tool** analysis:
  - same meta-analysis using one tool’s warnings

Results: secondary analyses: per-tool

- **per-tool** analysis:
  - same meta-analysis using one tool's warnings
  - results were **similar**:
    - all tools have same pattern of correlations
    - gives us **a bit more confidence**

Results: secondary analyses: ablation

- **leave-one-out ablation** analysis:
  - same meta-analysis without the warnings from each tool

Results: secondary analyses: ablation

- **leave-one-out ablation** analysis:
  - same meta-analysis without the warnings from each tool
  - results **nearly identical**, implying no one tool dominates

Results: secondary analyses: categories

- **per-metric-category** analysis:
    - same meta-analysis, but with only metrics from one category
    - correctness, rating, time, and physiological categories

Results: secondary analyses: categories

- **per-metric-category** analysis:
  - same meta-analysis, but with only metrics from one category
  - correctness, rating, time, and physiological categories
  - similar results; **too-wide** confidence intervals (except rating)

Contributions

Contributions

- The first **empirical evidence** of a correlation between verifiability and understandability
    - supports the **common wisdom** of verification experts

Contributions

- The first **empirical evidence** of a correlation between verifiability and understandability
  - supports the **common wisdom** of verification experts
- **Implications** for verification tool builders, verification tool users, and comprehensibility researchers

Contributions

- The first **empirical evidence** of a correlation between verifiability and understandability
  - supports the **common wisdom** of verification experts
- **Implications** for verification tool builders, verification tool users, and comprehensibility researchers
- A **replication package** with our scripts and data, so that others can repeat or extend our experiments
  - https://tinyurl.com/34hv45bm

Contributions

- The first **empirical evidence** of a correlation between verifiability and understandability
  - supports the **common wisdom** of verification experts
- **Implications** for verification tool builders, verification tool users, and comprehensibility researchers
- A **replication package** with our scripts and data, so that others can repeat or extend our experiments
  - https://tinyurl.com/34hv45bm

Thanks to my fabulous collaborators!