Herbert Wertheim College of Engineering
UNIVERSITY of FLORIDA

From Groundhog Day to Ground Truth Data:

We Need Curated Datasets to Spur Innovation

Authors

Anurag Swarnim Yadav (anuragswar.yadav@ufl.edu)

Joseph N. Wilson (jnw@ufl.edu)

University of Florida
Gainesville, FL USA

How Did We Get Here?

- Anurag Yadav got interesting in studying methods to automate program security vulnerability repair.
- We know that, even though there are numerous vulnerable programs, the data available for studying this problem is sparse and poorly curated.
- We wanted to understand the state-of-the-art (in 2023) and it looked like the VulRepair model (Fu, et al. 2022) was one such system.
- Anurag wanted to leverage that work and started looking more closely at its behavior.
- The performance seemed just a little too good to be true.
- This observation led us to the work reported in **R+R: Security Vulnerability Dataset Quality Is Critical** to be presented at ACSAC 2024.
- What do we mean by *too good to be true*?

One Sample Repaired by VulRepair*

Replaces call to jhash_3words

With call to call to siphash_3u32

And modifies two of its arguments

Googling “jhash_3words” “siphash_3u32” yields 9 results: all this patch!

```
501  501      void __ip_select_ident(struct net *net, struct iphdr *iph, int segs)
502  502      {
503       -          static u32 ip_idents_hashrnd __read_mostly;
504  503           u32 hash, id;
505  504
506       -          net_get_random_once(&ip_idents_hashrnd, sizeof(ip_idents_hashrnd));
     505  +          /* Note the following code is not safe, but this is okay. */
     506  +          if (unlikely(siphash_key_is_zero(&net->ipv4.ip_id_key)))
     507  +                  get_random_bytes(&net->ipv4.ip_id_key,
     508  +                                   sizeof(net->ipv4.ip_id_key));
507  509
508       -          hash = jhash_3words((__force u32)iph->daddr,
     510  +          hash = siphash_3u32((__force u32)iph->daddr,
509  511                                   (__force u32)iph->saddr,
510       -                                  iph->protocol ^ net_hash_mix(net),
511       -                                  ip_idents_hashrnd);
     512  +                                  iph->protocol,
     513  +                                  &net->ipv4.ip_id_key);
512  514           id = ip_idents_reserve(hash, segs);
513  515           iph->id = htons(id);
514  516      }
```

*Fu et al., VulRepair: A T5-based automated software vulnerability repair. *(ESEC/FSE’22)*.

What are the Problems With This Sample?

VulRepair samples are labeled with CWE tags.

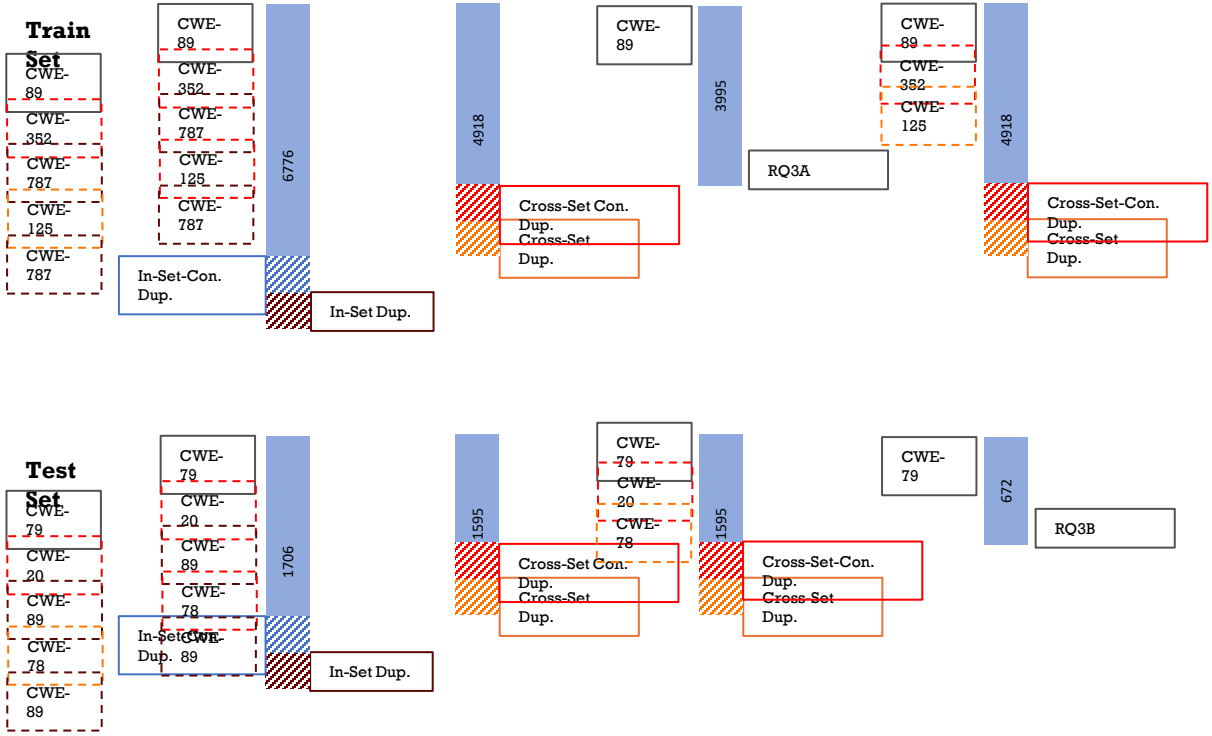
- This sample appeared in the training set with label CWE-200 and in the test set with label CWE-326.
- This inconsistency is a result of one or the other labels being *inaccurate*.
- The information necessary to determine that the change remediates a security vulnerability is missing.
- Upon further inspection there were numerous instances of duplication, both within the training and test sets and across the two sets.
- Four of the five attributes of a high-quality dataset (Croft, et al. 2023*) are not embodied by VulRepair.

*R. Croft et al., Data quality for software vulnerability dataset, 2023 IEEE/ACM ICSE, 2023.

ISO Dataset Quality Requirements

- **Uniqueness** — Each datum is represented exactly once within the dataset.
- **Consistency** — No datum appears in the dataset with conflicting labels.
- **Accuracy** — Each datum embodies semantic label correctness.
- **Completeness** — Each sample contains all information necessary to determine that the identified vulnerability exists and is correctly repaired.
- **Currentness** — The data are temporally homogeneous with the application that will use them—thus, for example, there will be no temporal leakage.

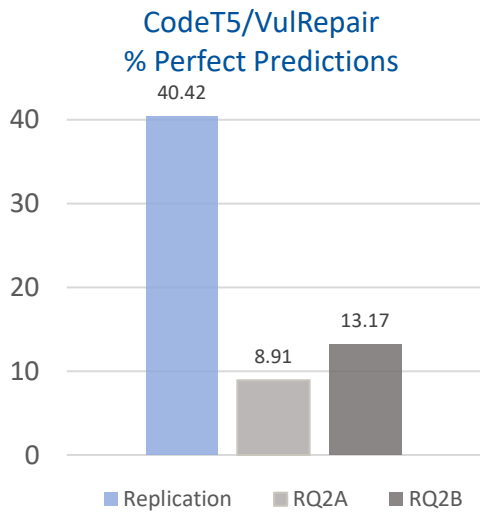
Extent of VulRepair Uniqueness and Consistency Problems

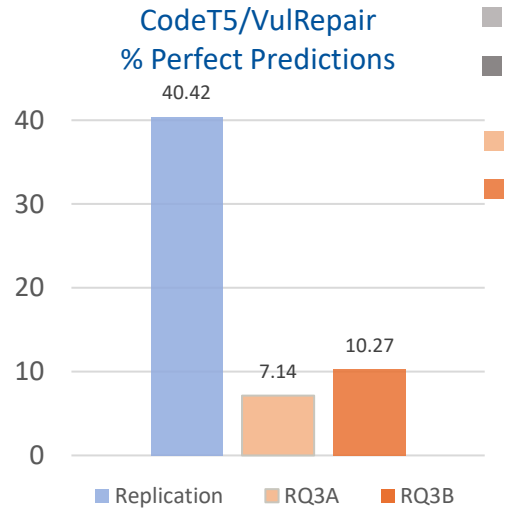


Key Findings:

Samples	Train	Test
Total samples	6,776	1,706
In-Set and In-Set Con. Dup.	1,858	111
Samples Left	4,918	1,595
Cross-Set and Cross-Set Con. Dup.	923	
RQ3A	3995	1595
RQ3B	4918	672

What is the Impact of Duplicates on Repair Performance?





RQ2A: Exact duplicates removed from training set

RQ2B: Exact duplicates removed from test set

RQ3A: Conflicting label duplicates removed from training

RQ3B: Conflicting label duplicates removed from test set

But It's Only One Dataset, Right?





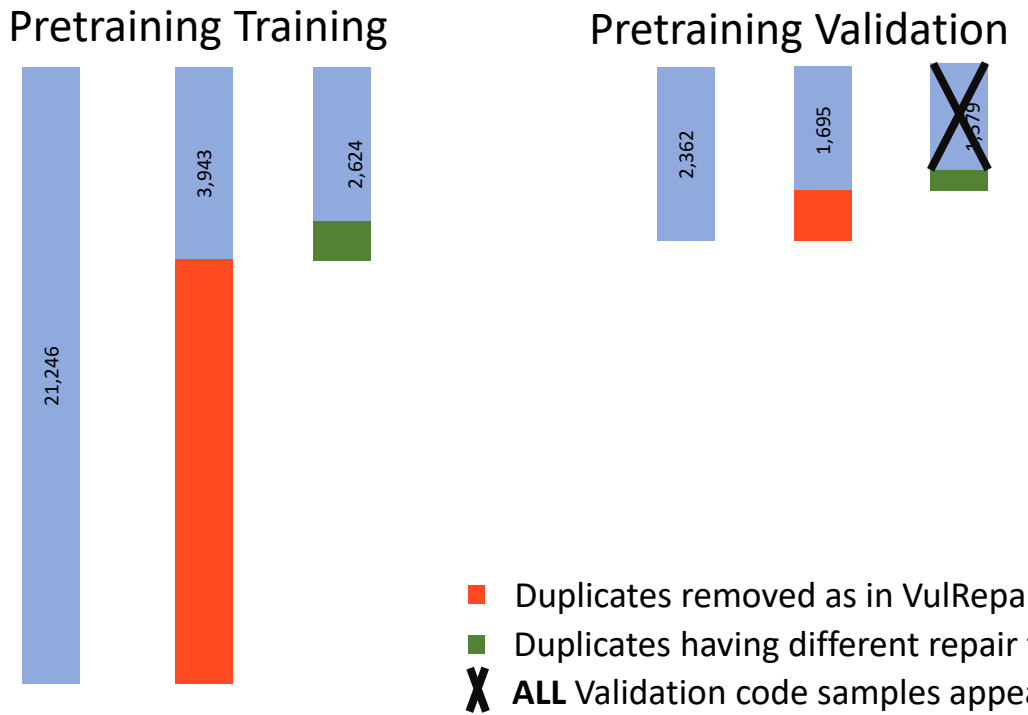
CONFERENCE	YEAR	CITED BY
MSR (BigVul)	2020	341
PROMISE (CVE Fixes)	2021	166
ESEC/FSE (VulRepair)	2022	150
IEEE Transactions on Dependable and Secure Computing	2023	30
ASE	2023	55
ACM Transactions on Software Engineering and Methodology (Modified VulRepair alternate BugFix)	2024	15

VQM Fixed That, Right?

- In **Vision Transformer Inspired Automated Vulnerability Repair,** 2024, Fu et al.* use new pretraining and a deduplicated VulRepair dataset that still exhibit a striking level of duplication.
- The paper identifies the pretraining set as the VRepair Bugfix set, however, it is a different, much smaller dataset.
- No discussion of the impact of duplicates on the earlier reported results of the VulRepair model is provided.
- The pretraining set with 21,246 training and 2,362 validation samples contains only 3,943 unique training and 1,695 unique validation samples with 1,613 shared between the training and validation sets.
- When repair tags are removed from the samples, only 2,624 unique training and 1,579 unique validation samples remain.
- But **all** the validation samples are present in the training set.

* Fu et al., Vision transformer inspired automated vulnerability repair, ACM Transactions on Software Engineering and Methodology, 2024

VQM Pretraining Dataset



But VQM Has Leakage from Pretraining

- The VQM fine-tuning dataset has 3,790 training, 536 validation, and 1,090 test samples.
- There is no duplication within the fine-tuning dataset.
- There **is**, however, duplication of pretraining samples in the finetuning data.
  - 1,747 of the 3,790 fine-tuning training samples (46.09%) are in pretraining.
  - 243 of the 536 fine-tuning validation samples (45.34%) are in pretraining.
  - 511 of the 1,090 fine-tuning test samples (46.88%) are in pretraining.
- Only 103 pretraining samples (< 4%) are not present in the fine-tuning data.
- The percentage of test samples leaked from the pretraining set is larger than the reported accuracy of the VQM model on the fine-tuning test set (45.14 ± 0.86%)!

Didn't the National Academies Deal With This?

- The National Academies published **Reproducibility and Replicability in Science*** in 2019.
- This addresses carrying out activities to demonstrate:
  - Reproducibility: obtaining consistent computational results using the same input data.
  - Replicability: obtaining consistent results across studies answering the same scientific question each with their own data.
- Neither of these gets at the underlying issue.
- Page 44 of the document gets it right: "Do the data and analysis offered in support of the result *in fact* support that result?"

  But page 45 says this question "refers to the act of recomputing the results using the available data, code, and methods of analysis." The semantics of the method may be wrong or may not be embodied properly in the code.
- Why does the VulRepair data not support the reported results?

  Because training with duplicated data leads to information leakage, leading to overestimates of algorithm performance.

*Natl. Acad. Sci. Eng. Med. 2019. Reproducibility and Replicability in Science. Washington, DC: Natl. Acad. Press

ACM Badging Effort

- Like unto the National Academy effort, ACMs badging effort supports ensuring both reproducibility of results (using exact data and methods) and replicability of results (using new artifacts yielding the same finding).
- The badging effort does not address whether the data and methods used to process them support the conclusions drawn by the authors.
- Identifying reproducibility is valuable but, as we have shown, much more needs to be done to ensure that the data, and use of it, support the conclusions that are being drawn using it.

Getting the Dataset Right

- We discussed duplication in VulRepair associated with Uniqueness and Consistency, but we have not yet discussed the broader issues of accuracy and completeness.
- Uniqueness is (relatively) easy to verify and it can be automated.
- Consistency is also readily automated.
- Ensuring accuracy requires us to verify that the CWE label for a particular vulnerable sample is correct.
  - Most CVE patch commits contain modifications that are not associated with the CVE itself—tertiary and stylistic changes may be included in such commits.
  - Rice's theorem tells us that nontrivial program properties are undecidable. Thus, manual verification is required and, indeed, may still yield incorrect results.
- Verification of completeness—making sure the sample includes all the information necessary to identify and repair the bug is also undecidable.
  - In some programs, a single line may be modified to resolve a security vulnerability, however, the code that reveals the appropriate change may be found far away in the program text and possibly in multiple places.

Automated Dataset Creation Can Help

- Ding et al.* describe and improved dataset creation method that resulted in the vulnerability *detection* dataset, PrimeVul.
- The approach they describe involves using only CVE patch commits containing a *single* modified function.
- This effectively addresses the accuracy issue if we assume the commit resolves the vulnerability.
- Consistency could be preserved by labeling the data, however, few PrimeVul samples are CWE-tagged and sometimes the NVE database designation for a given CVE will be modified. Thus, any CWE-label is time-dependent.
- Other than not addressing completeness, this approach is sufficient for a detection dataset at the function level with no CWE labels.
- PrimeVul would require modification to be used for repair because not all the vulnerable samples have corresponding repaired samples and only a few of the samples are CWE tagged.

*Ding et al., Vulnerability Detection with Code Language Models: How Far Are We?, IEEE/ACM ICSE, 2025.

So Now We Know All We Need to Do, Right?

- Not really.
- The current state-of-the-art for automated security vulnerability repair identifies a correct result as a token-by-token match of the expected repair.
- But we know that multiple different sequences of tokens may represent functions with identical semantics.
- Addressing this, however, is beyond the scope of just the data itself—program equivalence is also an undecidable problem.
- But we must demand that datasets support the conclusions they are used to reach as best they can!

None of This is New

- I was involved in a sequence of multi-year studies supported by Ft. Belvoir's NVESD (now C5ISR) to detect landmines using ground-penetrating radar and electromagnetic induction sensors.
- This work followed the typical F@&% Around and Find Out strategy:
  - First you *Fool Around* until you understand what's happening.
  - Then you *Find Out* how to really solve the problem.
- The multiple teams on these efforts had access to the same ever-increasing set of ground truth data samples, but for the first several years they carried out different experiments on different subsets of available data.
- It was not until we agreed to standard, well-curated, datasets appropriate to solve the problem that we really started making significant, robust performance improvements. Robust algorithm development followed.
- In the end, the algorithms we produced in this effort were installed in military vehicles in Afghanistan leading the route clearance personnel to tell us that our systems allowed them, for the first time, to find an IED without it blowing up the vehicle.

What Were the Elements of Our Success?

- We had enough data to actually learn something.

  (The right amount of data is always **more**, but we had enough to do some good.)
- We had a motivated sponsor who wanted to solve the problem!
- We had multiple teams of researchers who were willing to work together to agree on what should be done.
- We had a team willing to collect and curate the data based on the experiments the different groups proposed.
- We maintained a dynamic collection of data that incorporated changes (new data, remediation of earlier errors, addition of new tags and attributes) as time went on and we referenced our results against published versions of those datasets.
- But more than anything else, we had specific funding that was devoted to the data curation effort. Without this, nothing else could have worked.

Just to Emphasize the Point

- A number of papers have reported the F1 scores of applying LineVul and/or CodeBert to detection of function-level vulnerabilities using BigVul.
- Although these report what seem to be the same experiment, here are a few of the results:
- LineVul: 0.91
  Fu et al., LineVul: A transformer-based line-level vulnerability prediction, 2022 Mining Software Repositories Conference, 2022.
- LineVul: 93.23, CodeBert: 21.04
  Steenhoek et al., Dataflow analysis deep-learning for efficient vulnerability detection, IEEE/ACM ICSE, 2024.
- LineVul: 81, CodeBert: 21
  Yang et al., Security vulnerability detection with multitask self-instructed fine-tuning of large language models, arXiv 2024.
- CodeBert: 62.88
  Ding et al., Vulnerability Detection with Code Language Models: How Far Are We?, IEEE/ACM ICSE 2025.
- If we want to be able to compare results, we must start with standardized datasets and processing practices.
  We clearly have not reached that stage.

How Do We Move Forward

- If automated security vulnerability repair is going to become a reality, some entity must take the lead—either government or industry.
- All of us understand the importance of security but we may not agree that automated code repair is the way to achieve it.
- If, however, we don't maintain and use high-quality curated datasets, we will not have any idea whether we are solving the problem or not. We will just be stuck in the *fool-around* phase.

  Although researchers are updating their data to remove duplicates and other errors, they are not reporting revisions of earlier results employing the improved datasets.
- We've addressed the significance of the problem that poor datasets present.
- We are certain the only way to get significant improvement is to bring all the researchers under one umbrella and fund the creation and use of standard datasets needed to support them.
- That's what we're calling for here and now.

Questions and Discussion