CHRONOS: Time-Aware Zero-Shot Identification of Libraries from Vulnerability Reports

Yunbo Lyu[1], **Thanh Le-Cong[2]**, Hong Jin Kang[1], Ratnadira Widyasari[1],
Zhipeng Zhao[1], Bach Le[2], Ming Li[3], David Lo[1]

[1]*Singapore Management University, Singapore*
[2]*University of Melbourne, Australia*
[3]*Nanjing University, China*

Motivation

- Increasing usage of third-party libraries → user needs to be aware of the library vulnerabilities.
- National Vulnerability Database (NVD) curate vulnerability reports for third party libraries.
- However, vulnerability report fails to include the list of all affected libraries.
- Human efforts required to identify libraries related to the vulnerability:
  -Resource and time incentive
  -Huge number of possible labels

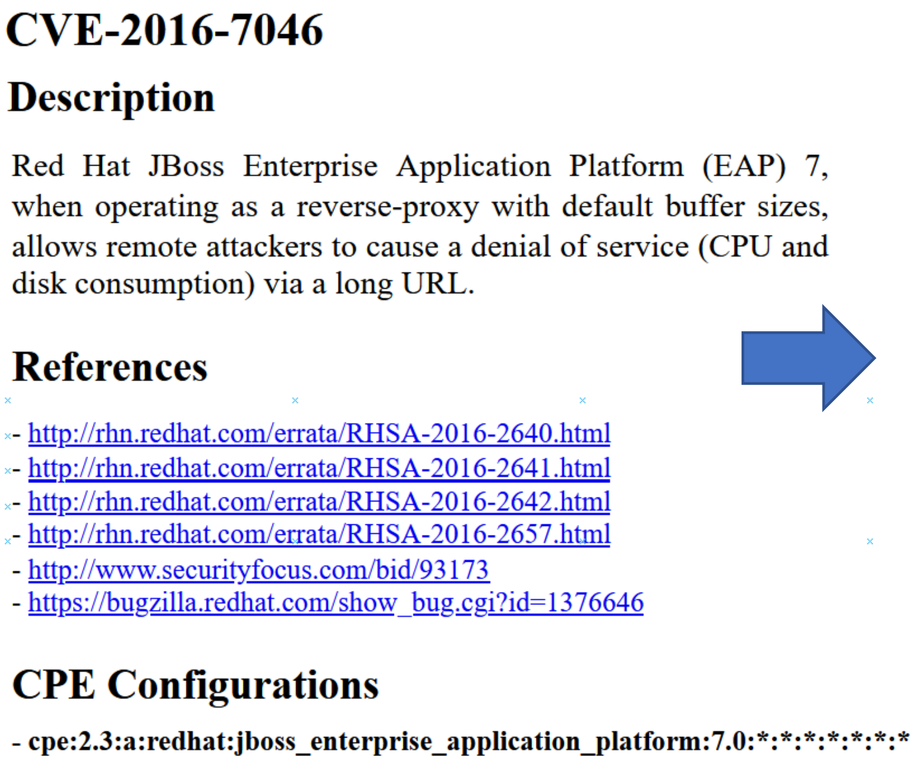
Automated approaches for identifying affected library of a vulnerability report

Problem Statement

<u>Input</u>
Vulnerability Report

<u>Approach</u>
AI Models

<u>Output</u>
Affected Libraries



CVE-2016-7046

Description

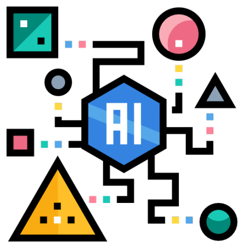
Red Hat JBoss Enterprise Application Platform (EAP) 7, when operating as a reverse-proxy with default buffer sizes, allows remote attackers to cause a denial of service (CPU and disk consumption) via a long URL.

References

- http://rhn.redhat.com/errata/RHSA-2016-2640.html
- http://rhn.redhat.com/errata/RHSA-2016-2641.html
- http://rhn.redhat.com/errata/RHSA-2016-2642.html
- http://rhn.redhat.com/errata/RHSA-2016-2657.html
- http://www.securityfocus.com/bid/93173
- https://bugzilla.redhat.com/show_bug.cgi?id=1376646

CPE Configurations

- **cpe:2.3:a:redhat:jboss_enterprise_application_platform:7.0:*:*:*:*:*:*:***

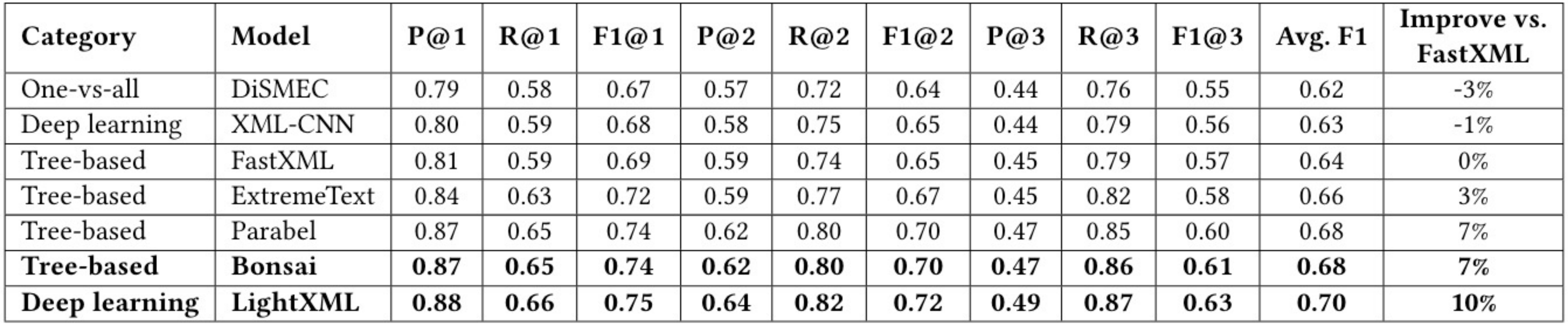


Automated Affected Library Identification

- Undertow
- EAP 7.0 Wildfly
- ActiveMQ Artermis
- Glassfish JSON
- JSoup

Prior Work

- Chen et al.[1] from Veracode have formulated the problem as an extreme multi-label classification (XML) problem and utilized FastXML to address this problem.
- Haryono et al.[2] found that the most effective XML approach for library identification is a deep learning-based approach, LightXML.



Category	Model	P@1	R@1	F1@1	P@2	R@2	F1@2	P@3	R@3	F1@3	Avg. F1	Improve vs. FastXML
One-vs-all	DiSMEC	0.79	0.58	0.67	0.57	0.72	0.64	0.44	0.76	0.55	0.62	-3%
Deep learning	XML-CNN	0.80	0.59	0.68	0.58	0.75	0.65	0.44	0.79	0.56	0.63	-1%
Tree-based	FastXML	0.81	0.59	0.69	0.59	0.74	0.65	0.45	0.79	0.57	0.64	0%
Tree-based	ExtremeText	0.84	0.63	0.72	0.59	0.77	0.67	0.45	0.82	0.58	0.66	3%
Tree-based	Parabel	0.87	0.65	0.74	0.62	0.80	0.70	0.47	0.85	0.60	0.68	7%
Tree-based	**Bonsai**	**0.87**	**0.65**	**0.74**	**0.62**	**0.80**	**0.70**	**0.47**	**0.86**	**0.61**	**0.68**	**7%**
Deep learning	**LightXML**	**0.88**	**0.66**	**0.75**	**0.64**	**0.82**	**0.72**	**0.49**	**0.87**	**0.63**	**0.70**	**10%**

[1] Chen et al. "Automated identification of libraries from vulnerability data." ICSE 2020

[2] Haryono et al. "Automated identification of libraries from vulnerability data: can we do better?" ICPC 2022

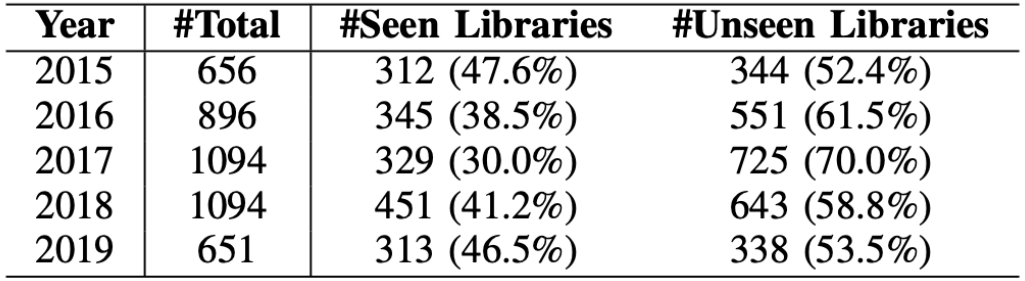
Motivation

+ Assume:

Libraries in Training = Libraries in Inference

+ Random Splitting & No Time-aware

Not suitable as **set of affected libraries could evolve over the time**



Year	#Total	#Seen Libraries	#Unseen Libraries
2015	656	312 (47.6%)	344 (52.4%)
2016	896	345 (38.5%)	551 (61.5%)
2017	1094	329 (30.0%)	725 (70.0%)
2018	1094	451 (41.2%)	643 (58.8%)
2019	651	313 (46.5%)	338 (53.5%)

>50% new affected libraries per years

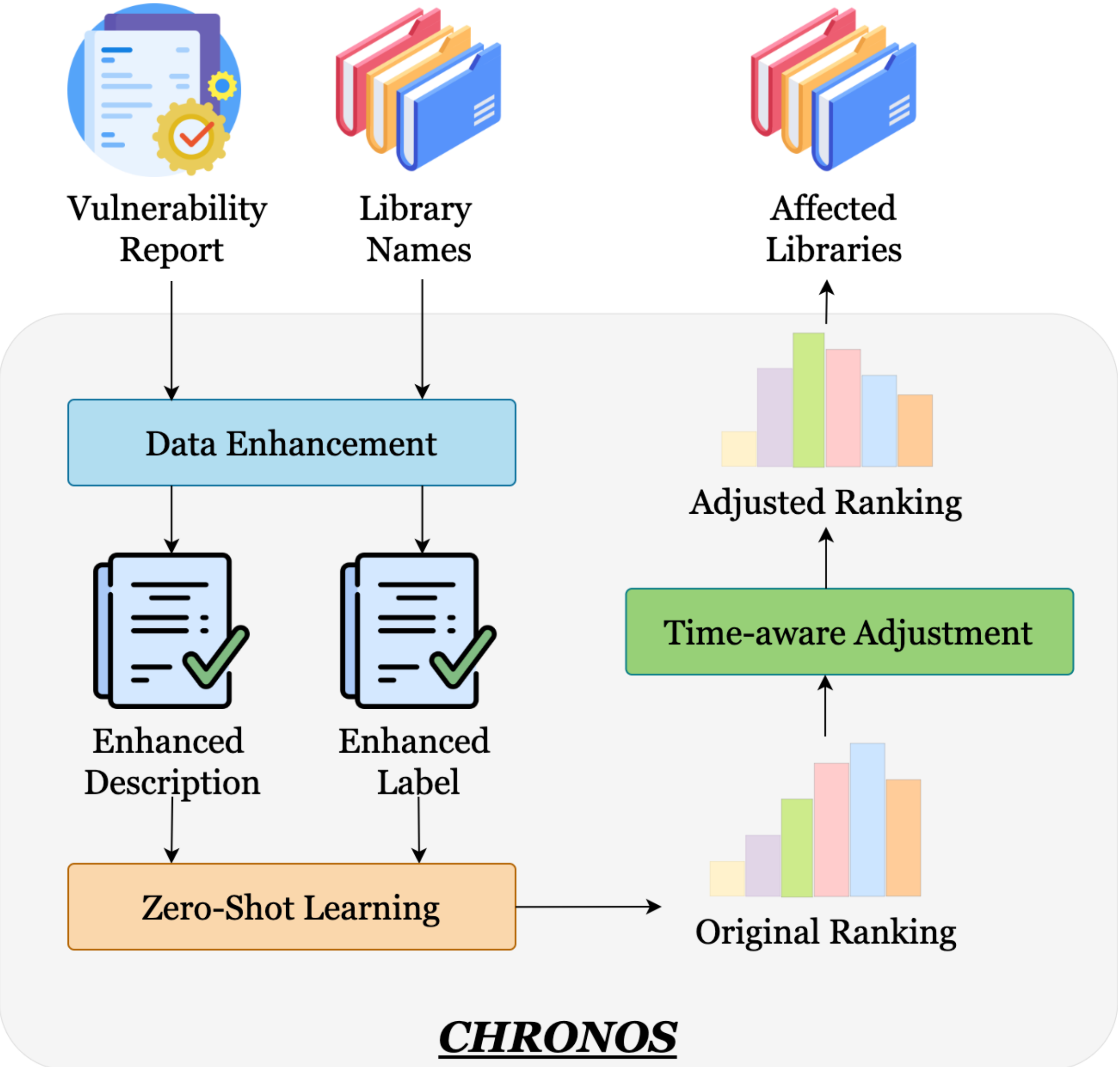
Time-aware Evaluation

- Training data: 2014-2016
- Validation: 2017
- Testing data: 2018-2019

LightXML (SOTA)	Average F1
Current Evaluation	0.7
Time-Aware Evaluation	0.25

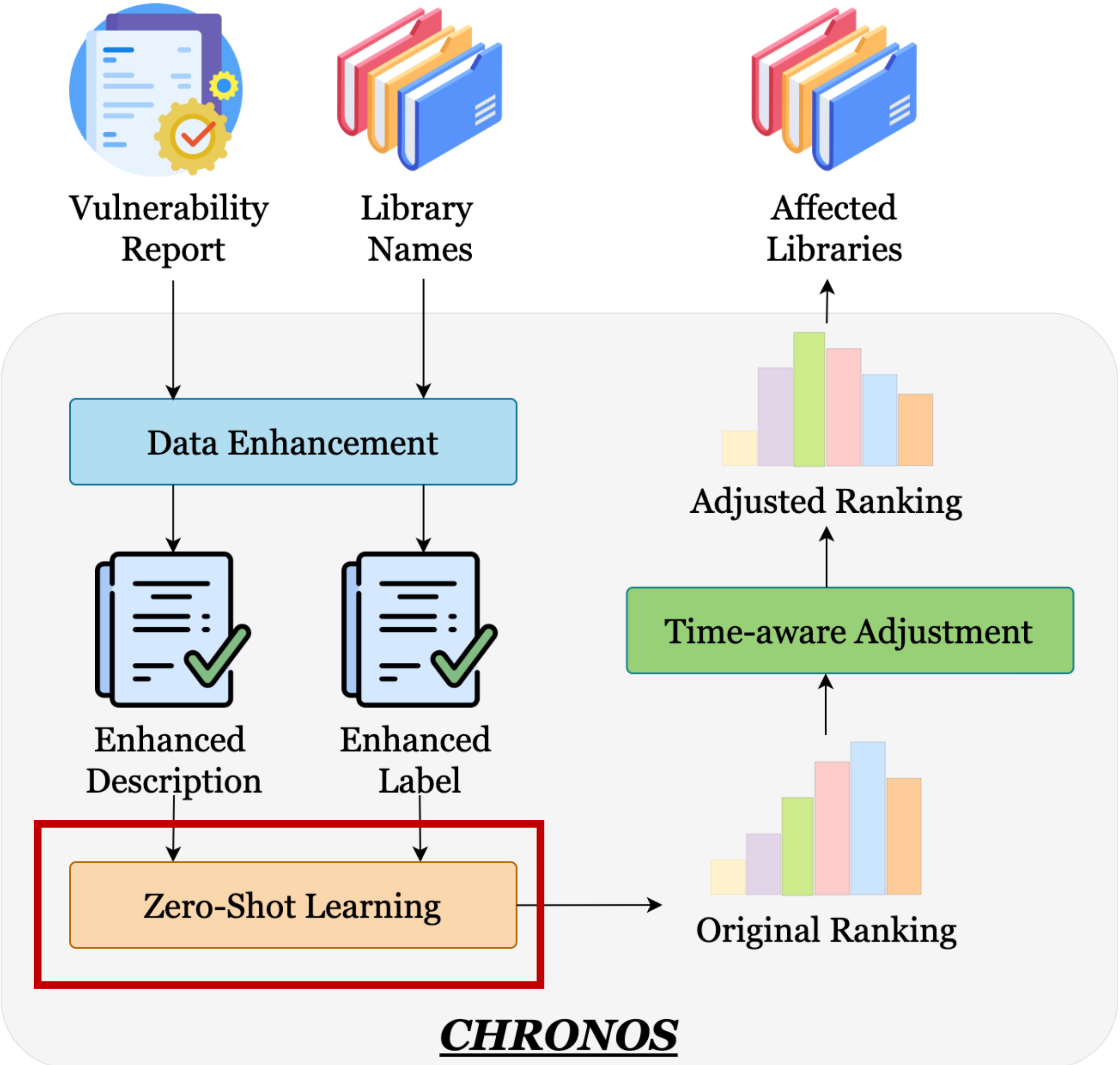
Significant Drop in Performance of SOTA

Proposed Approach: Chronos



Proposed Approach: Chronos

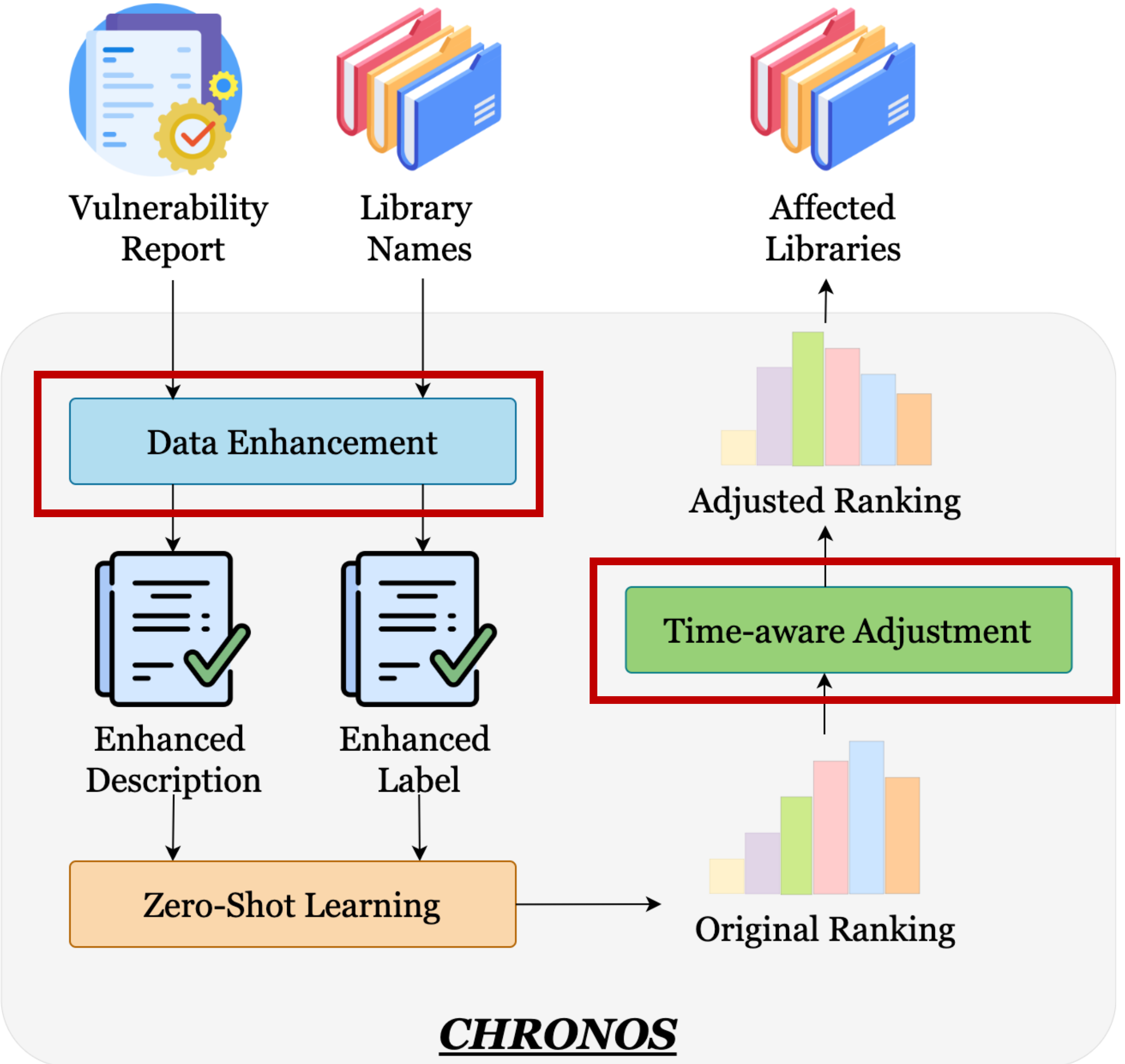
THE UNIVERSITY OF MELBOURNE



- **Model:** ZestXML[1]
- **Feature Extraction:** TF-IDF for both descriptions and labels.
- **ZestXML:** model the relevance between descriptions and labels by analyzing their linear feature interactions.
- Given a description d and a label l, the relevance score R between are calculated as follows:

$$R(d,l) = d^{\intercal} W l$$

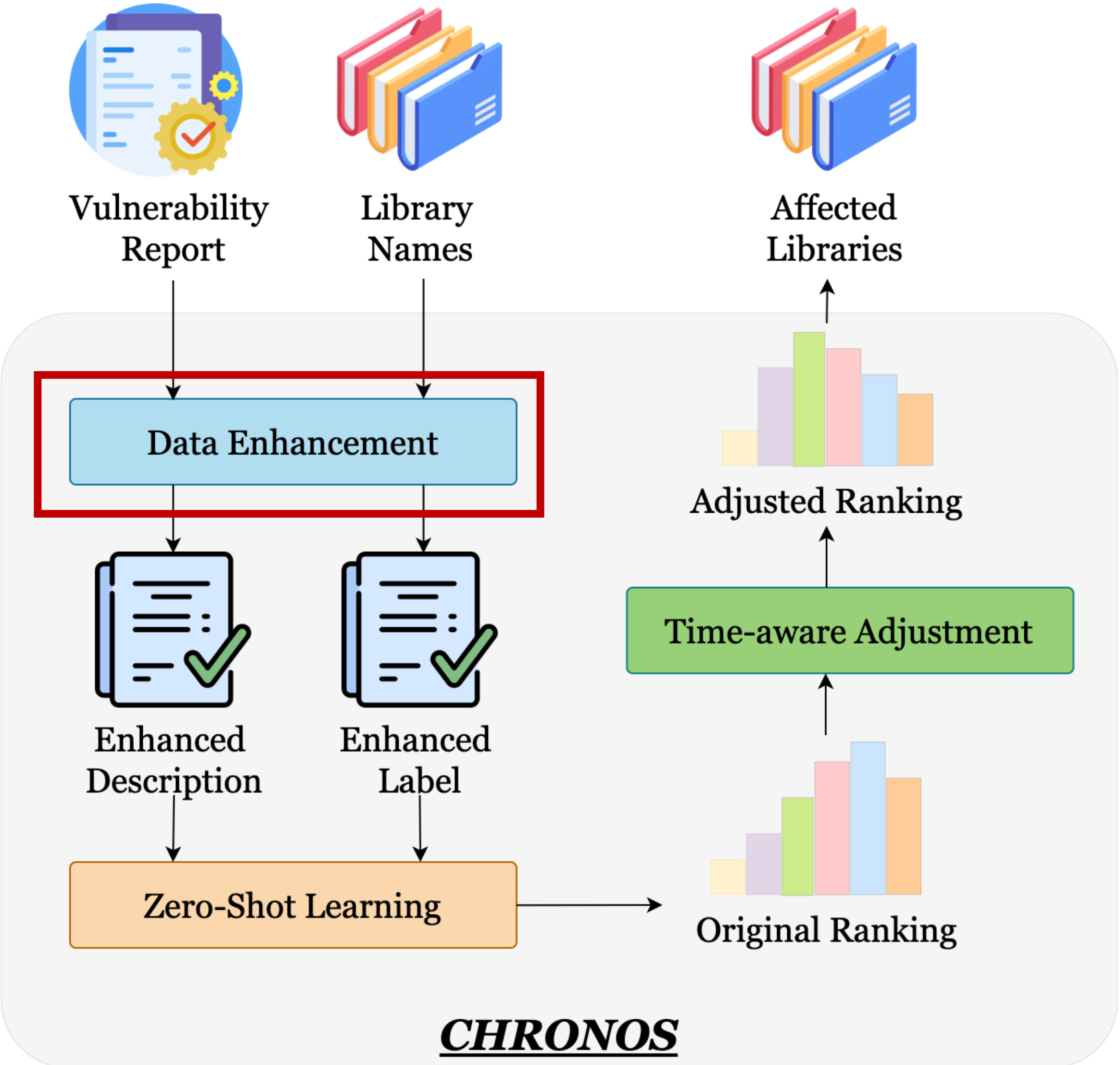
Proposed Approach: Chronos



Observation 1: Additional documents referenced in the NVD entry, e.g., bug reports, mailing lists, can help distinguish multiple previously unseen labels from one another.

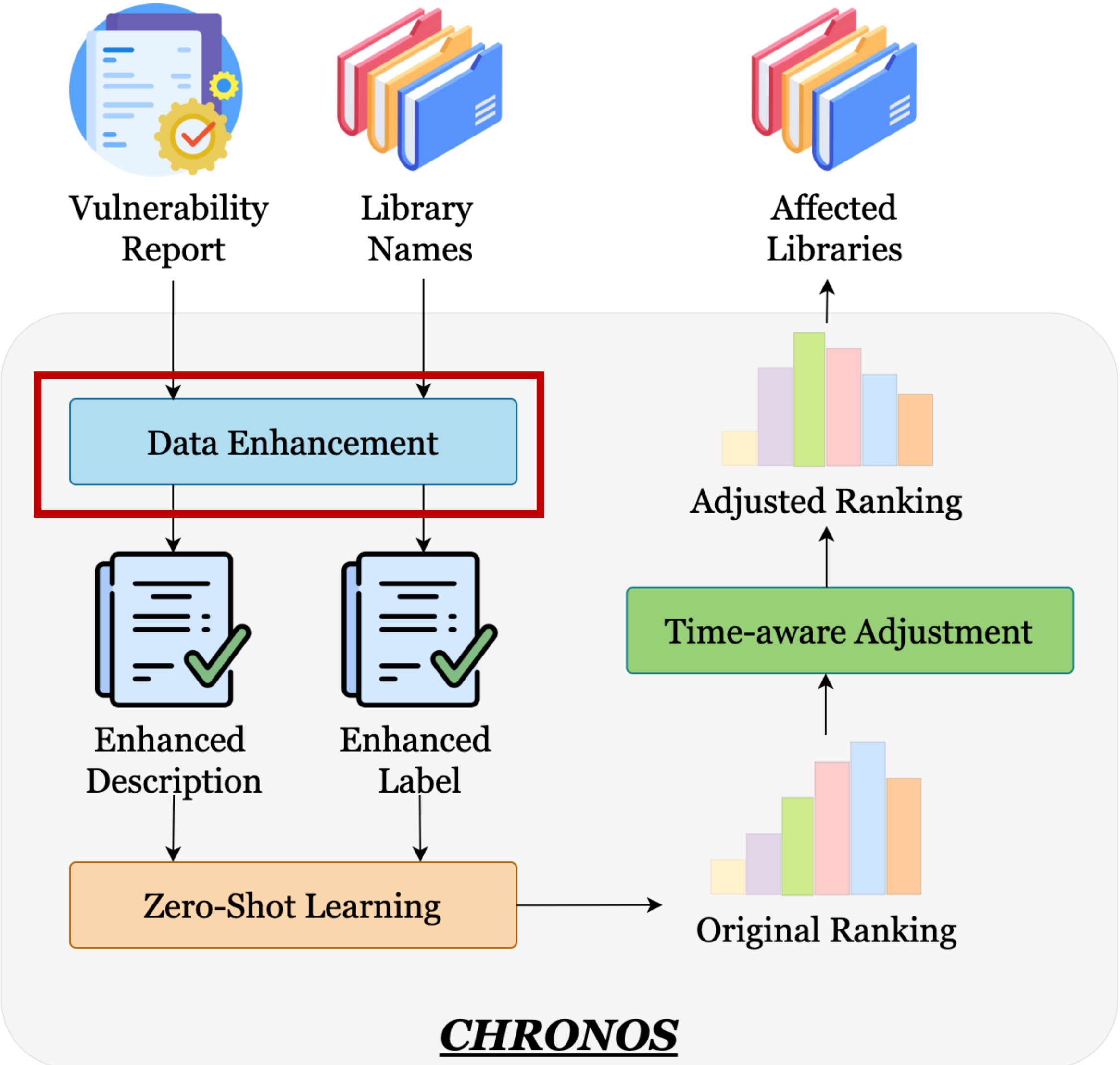
Observation 2: Exploiting temporal connection between vulnerability reports and affected libraries can help boost the prediction accuracy.

Proposed Approach: Chronos



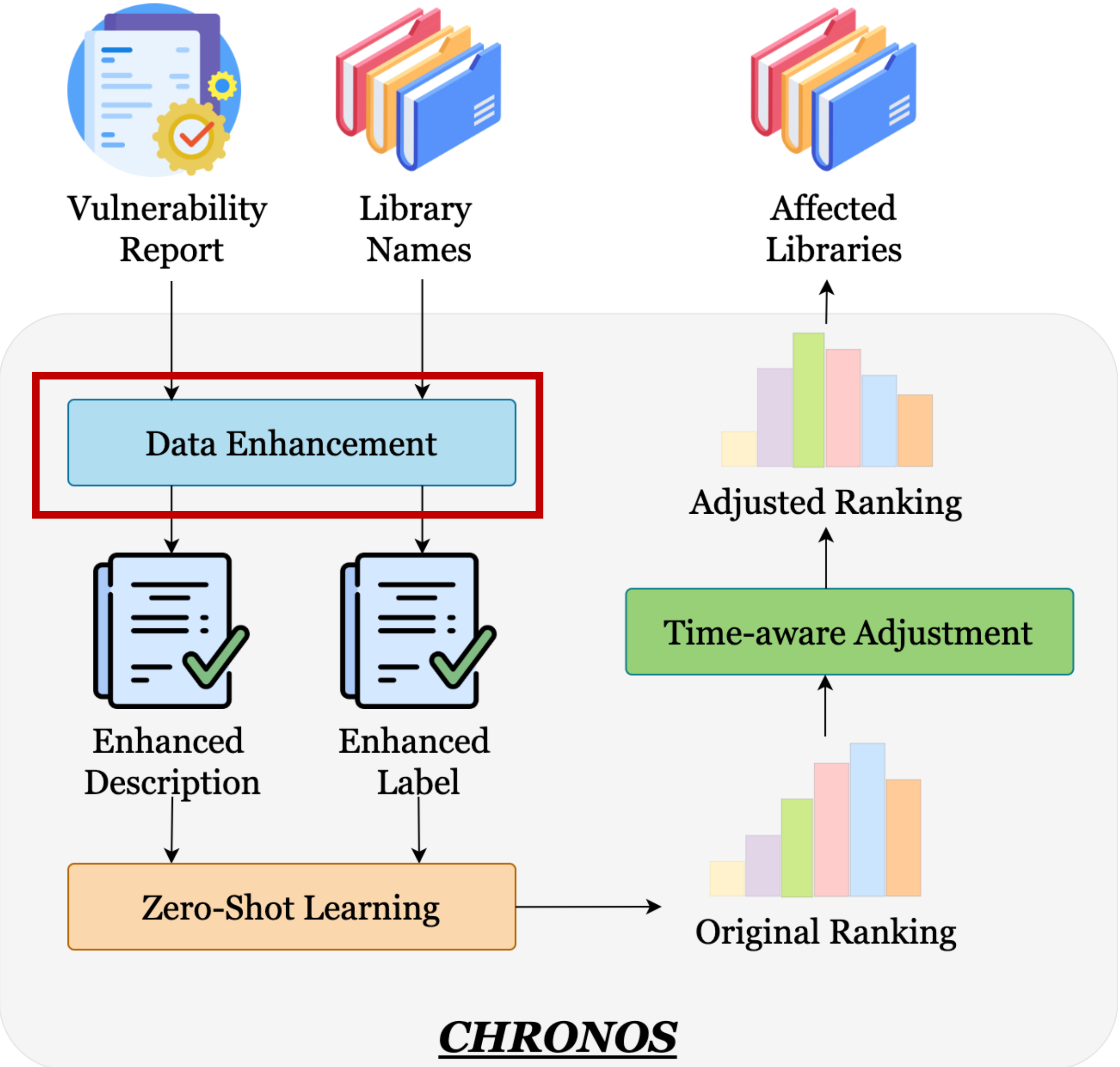
- Collecting Reference Data
- Library Sub-word Splitting

Proposed Approach: Chronos



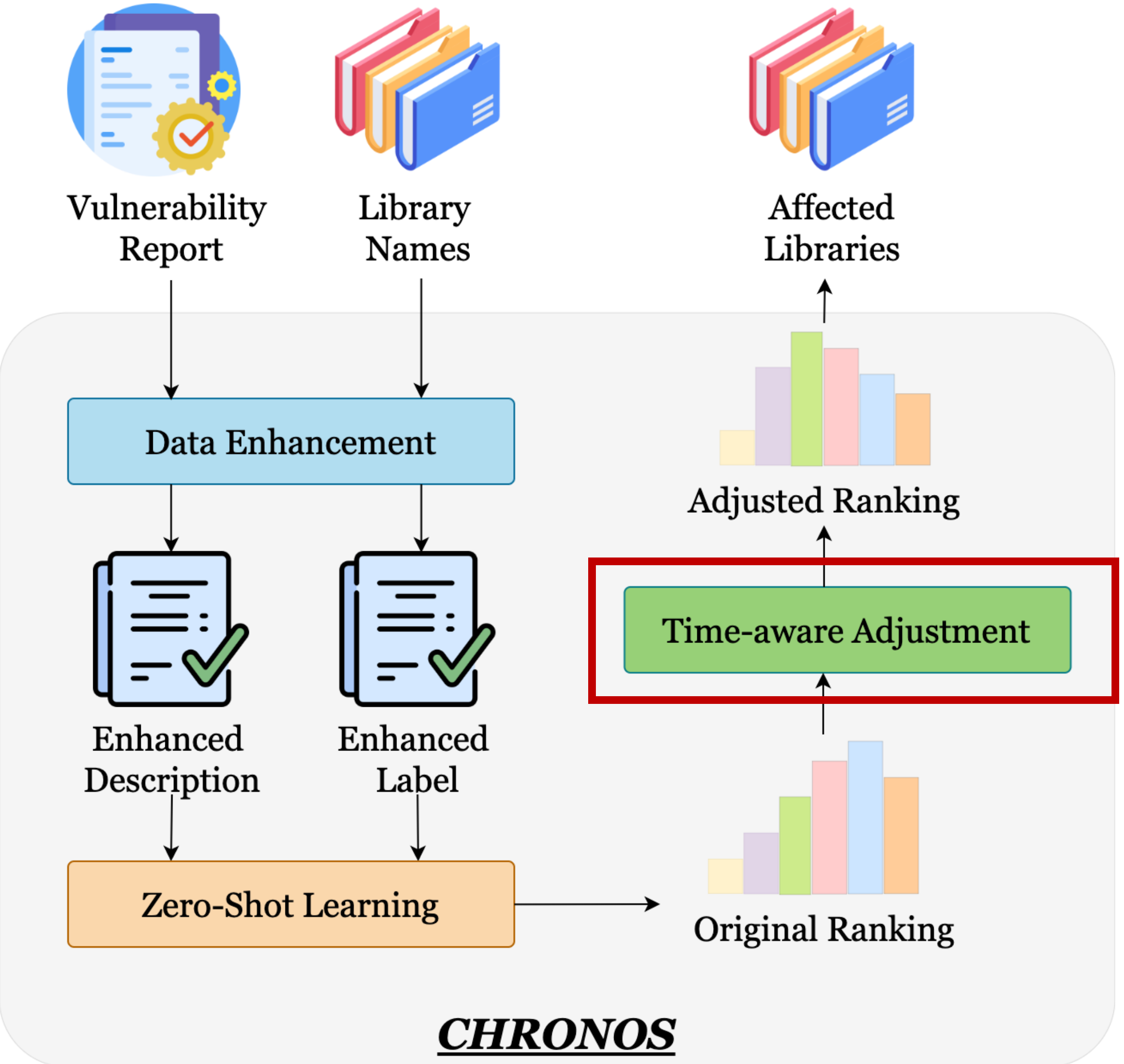
- Collecting Reference Data
    - A vulnerability report can come with a list of website references containing useful information to identify library
    - Crawl the web references to retrieve its textual information
    - Too many reference websites ==> extract data from a list of 12 domain that covers 82.3% vulnerability reports
    - Preprocess: remove non-alphanumeric words or stemming

Proposed Approach: Chronos



- Library Sub-word Splitting
  - Enriches the features that help determine labels associated with vulnerability reports.
  - Apply Spiral [1] token splitter to split tokens into sub-tokens.
    - org.springframework -> org spring framework
    - pyopenssl -> py openssl

Proposed Approach: Chronos



We observe that vulnerabilities in the same time range are more likely to affect the recent versions of the libraries.

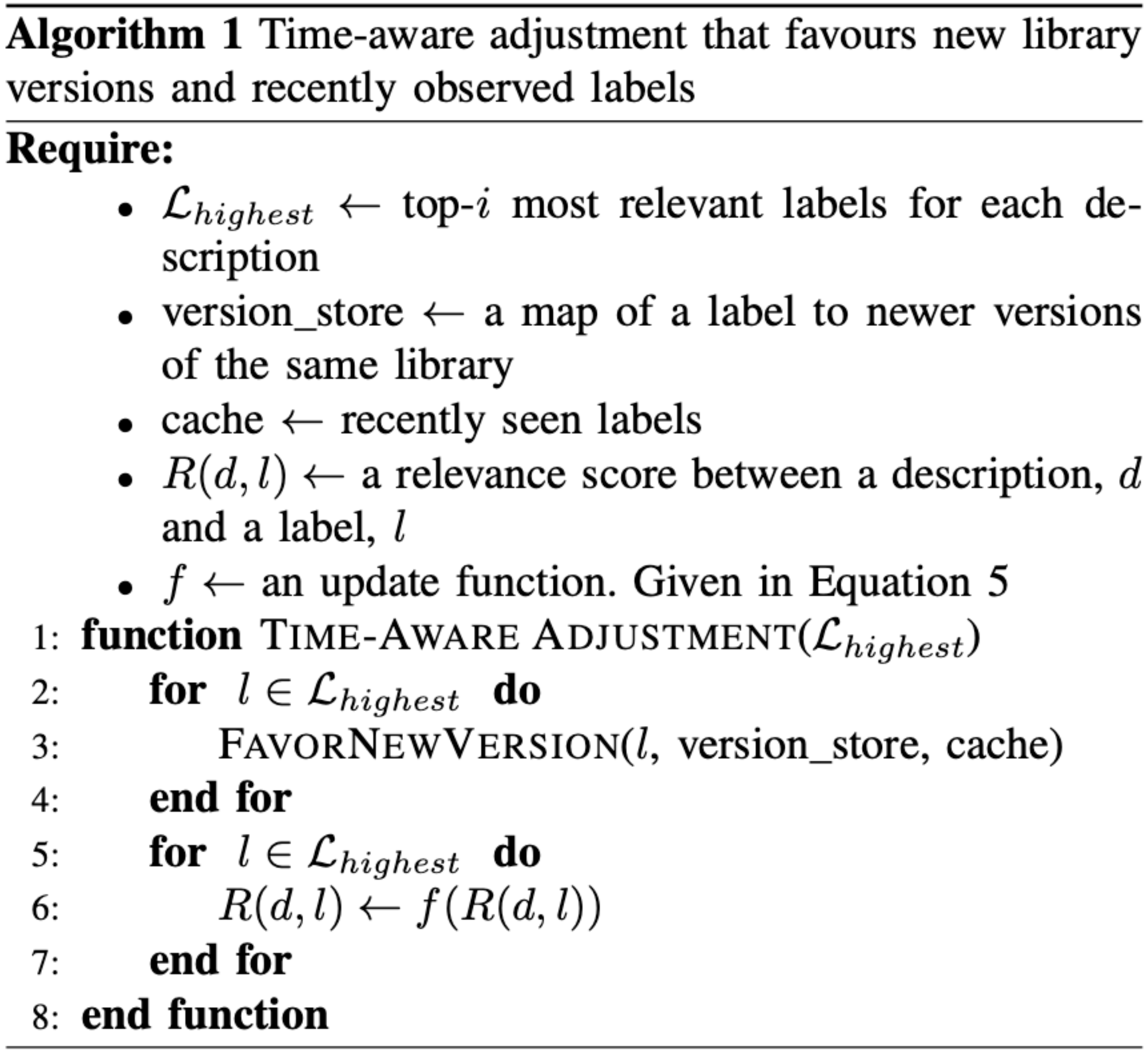
⬇

CHRONOS uses a strategy to prioritize versions of libraries that have been recently affected by vulnerabilities.

⬇

Time-aware Adjustment

Proposed Approach: Chronos



Algorithm 1 Time-aware adjustment that favours new library versions and recently observed labels

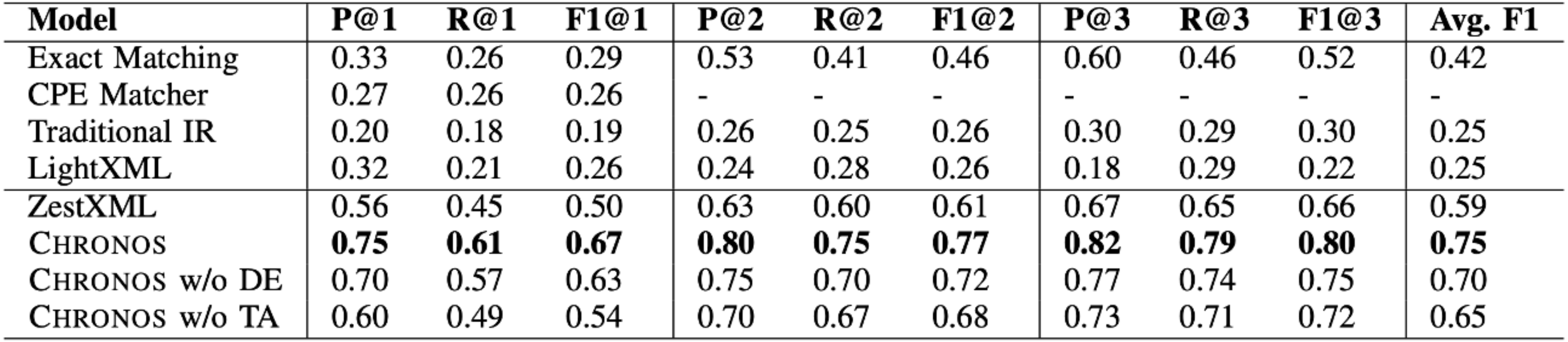
Require:

- $\mathcal{L}_{highest} \leftarrow$ top-i most relevant labels for each description
- version_store $\leftarrow$ a map of a label to newer versions of the same library
- cache $\leftarrow$ recently seen labels
- $R(d, l) \leftarrow$ a relevance score between a description, d and a label, l
- $f \leftarrow$ an update function. Given in Equation 5

```
1: function TIME-AWARE ADJUSTMENT($\mathcal{L}_{highest}$)
2:     for $l \in \mathcal{L}_{highest}$ do
3:         FAVORNEWVERSION($l$, version_store, cache)
4:     end for
5:     for $l \in \mathcal{L}_{highest}$ do
6:         $R(d, l) \leftarrow f(R(d, l))$
7:     end for
8: end function
```

- The **version store** tracks the different versions of each library.
- The **cache** stores the recently affected libraries using a Least Recently Used replacement policy.
- Time-aware adjustment use two steps to modify the relevance scores:
  - **Replacement:** favor newer library versions (line 2-4)
  - **Update:** Add a **recency bias** (line 5-7)

Comparison with Baselines



Model	P@1	R@1	F1@1	P@2	R@2	F1@2	P@3	R@3	F1@3	Avg. F1
Exact Matching	0.33	0.26	0.29	0.53	0.41	0.46	0.60	0.46	0.52	0.42
CPE Matcher	0.27	0.26	0.26	-	-	-	-	-	-	-
Traditional IR	0.20	0.18	0.19	0.26	0.25	0.26	0.30	0.29	0.30	0.25
LightXML	0.32	0.21	0.26	0.24	0.28	0.26	0.18	0.29	0.22	0.25
ZestXML	0.56	0.45	0.50	0.63	0.60	0.61	0.67	0.65	0.66	0.59
CHRONOS	**0.75**	**0.61**	**0.67**	**0.80**	**0.75**	**0.77**	**0.82**	**0.79**	**0.80**	**0.75**
CHRONOS w/o DE	0.70	0.57	0.63	0.75	0.70	0.72	0.77	0.74	0.75	0.70
CHRONOS w/o TA	0.60	0.49	0.54	0.70	0.67	0.68	0.73	0.71	0.72	0.65

CHRONOS outperform baselines by 79-300% in terms of Average F1

Zero-shot learning (ZestXML) perform 2x better than supervised learning (LightXML)

Data Enhancement and Time-aware Adjustment improve ZestXML by 27% over

Conclusion

- We highlight **the evolvement of affected libraries** in the problem of identifying libraries from vulnerability reports
- We empirical investigate SOTAs on a **time-aware evaluation**, showing a **significant drop** on performance of SOTAs
- We propose Chronos, a **practical library identification** approach based on **zero-shot learning** along with **domain-specific mechanisms**: data enhancement and time-aware adjustments
- Our experiments demonstrate that employing both **zero-shot learning**, and **domain-specific mechanisms** yields **substantial improvements**, resulting in Chronos **outperforming the state-of-the-art (SOTA)** approaches significantly.

Thanks for your listening 🚀🚀🚀!

Questions are welcome 😊!