

VulSim: Leveraging Similarity of Multi-Dimensional Neighbor Embeddings for Vulnerability Detection



Samiha Shimmi, Northern Illinois University
Ashiqur Rahman, Northern Illinois University
Mohan Gadde, Northern Illinois University
Hamed Okhravi, MIT Lincoln Laboratory
Mona Rahimi, Northern Illinois University

Presented By: Samiha Shimmi
sshimmi@niu.edu

Unhappy Deep Learning Model!!



Happy Deep Learning Model!!



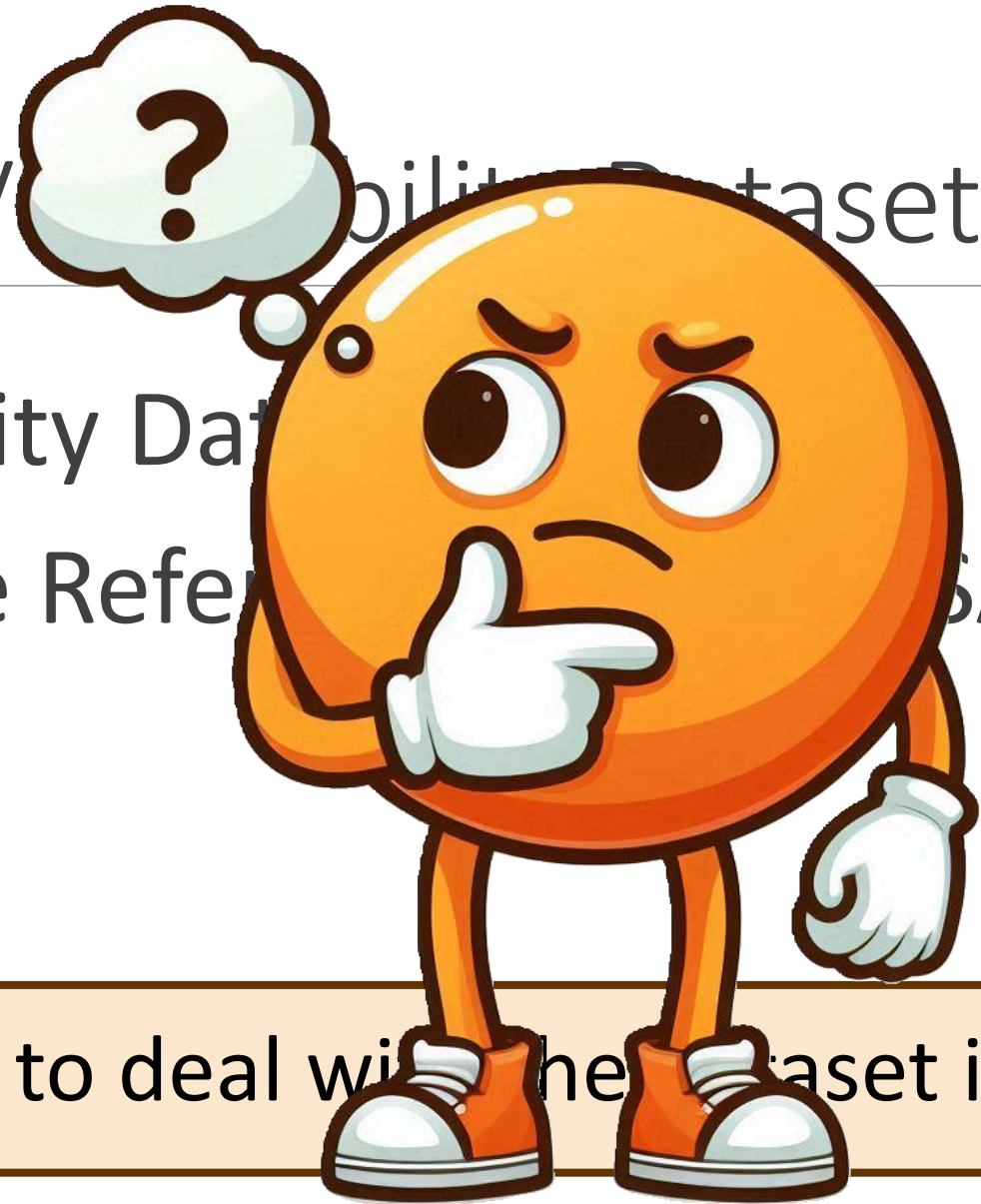
What is the challenge in Software Vulnerability Detection?

We do not have enough data!!



Existing Software Vulnerability Datasets

- ❌ National Vulnerability Database (NVD)
- ❌ Software Assurance Reference (SARD)
- ❌ DiverseVul



A robust solution is required to deal with the dataset issue!

Key Idea

You are the average of the five people you spend the most time with!!

~Jim Rohn



Key Idea (contd.)



Semantic Dimension

Function 1

```
void streamFreeCG(streamCG cg)
{
    raxFreeWithCallback(cg->pel,
        (void*)(void*) streamFreeNACK);
    raxFreeWithCallback(cg->consumers,
        (void*)(void*) streamFreeConsumers);
    zfree(cg);
}
```

Function 2

```
static void *StreamTcpSessionPoolAlloc(void)
{
    void *ptr = NULL;
    if (StreamTcpCheckMemcap((uint32_t)
        sizeof(TcpSession)) == 0)
        return NULL;
    ptr = SCMalloc(sizeof(TcpSession));
    if (unlikely(ptr == NULL))
        return NULL;
    StreamTcpSessionClear();
    return ptr;
}
```

Stream
related
vulnerability!

Contextual Dimension

Function 1

```
void host_lookup(char *user_supplied_addr)
{
    struct hostent *hp;
    in_addr_t *addr;
    char hostname[64];
    in_addr_t inet_addr(const char *cp)

    /* routine that ensures user_supplied_addr is in
    for conversion */
    validate_addr_form(user_supplied_addr);
    addr = inet_addr(user_supplied_addr);
    hp = gethostbyaddr(addr, sizeof(struct in_addr), AF_INET);
    strcpy(hostname, hp->h_name);
}
```

Memory
buffer
vulnerability!

Function 2

```
char *copy_input(char *user_supplied_string){
    int i, dst_index;
    char *dst_buf = (char*)malloc(4 * sizeof(char) *
    MAX_SIZE);
    if (MAX_SIZE <= strlen(user_supplied_string)) {
        die("user string too long, die evil hacker!");
    }
    dst_index = 0;
    for (i = 0; i < strlen(user_supplied_string); i++) {
        if ('&' == user_supplied_string[i]) {
            dst_buf[dst_index++] = '&';
            dst_buf[dst_index++] = 'a';
            dst_buf[dst_index++] = 'm';
            dst_buf[dst_index++] = 'p';
            dst_buf[dst_index++] = ';';
        }
        else if ('<' == user_supplied_string[i]) {
            /* encode to &lt; */
            dst_buf[dst_index++] = '&';
            dst_buf[dst_index++] = 'l';
            dst_buf[dst_index++] = 't';
            dst_buf[dst_index++] = ';';
        }
        else {
            dst_buf[dst_index++] = user_supplied_string[i]
        }
    }
    return dst_buf;
}
```


Syntactic Dimension

Function 1

Function 2

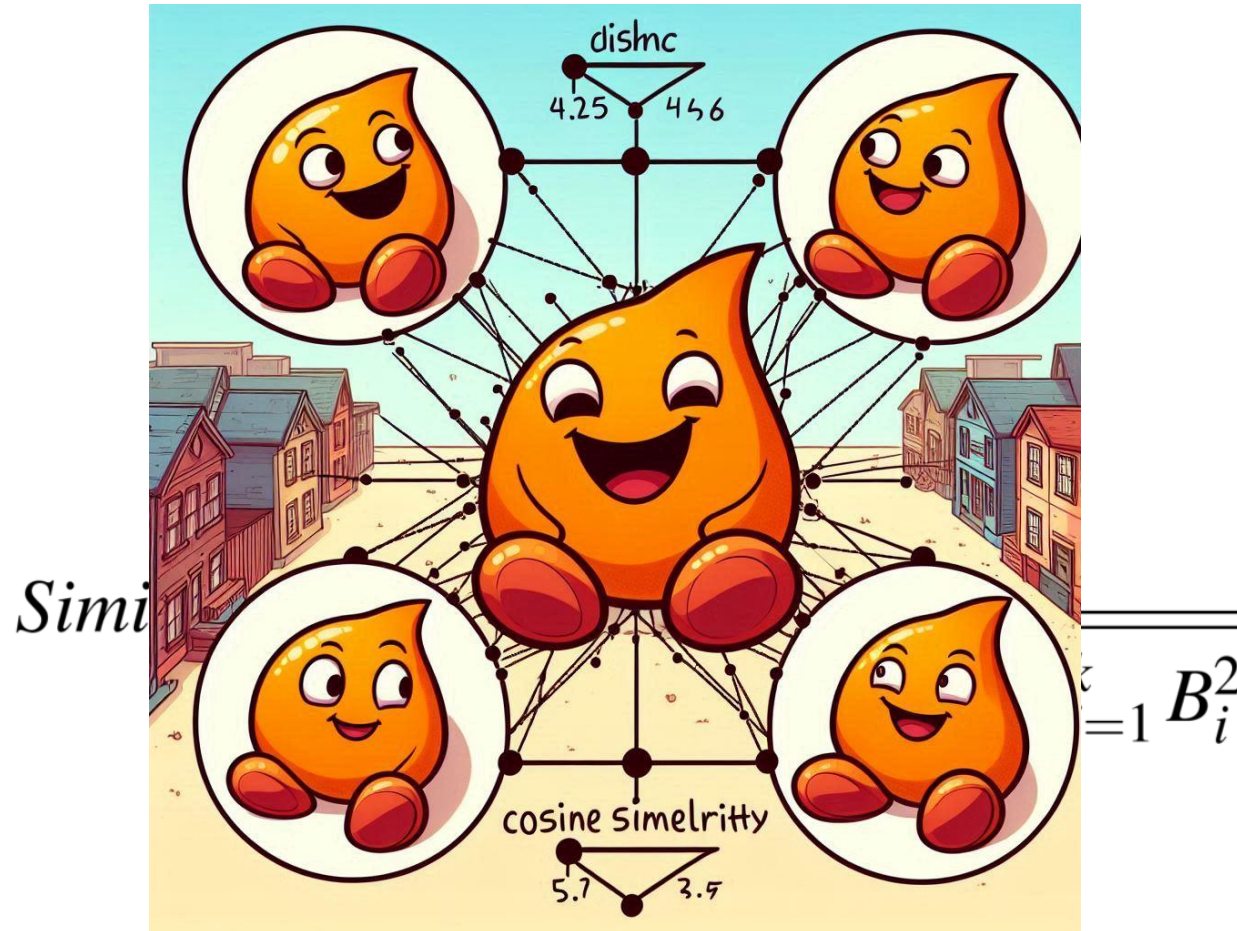
```
void PaymentRequest::NoUpdatedPaymentDetails()  
{  
    spec_>RecomputeSpecForDetails();  
}
```

```
void InitPrefMembers()  
{  
    settings_>InitPrefMembers();  
}
```



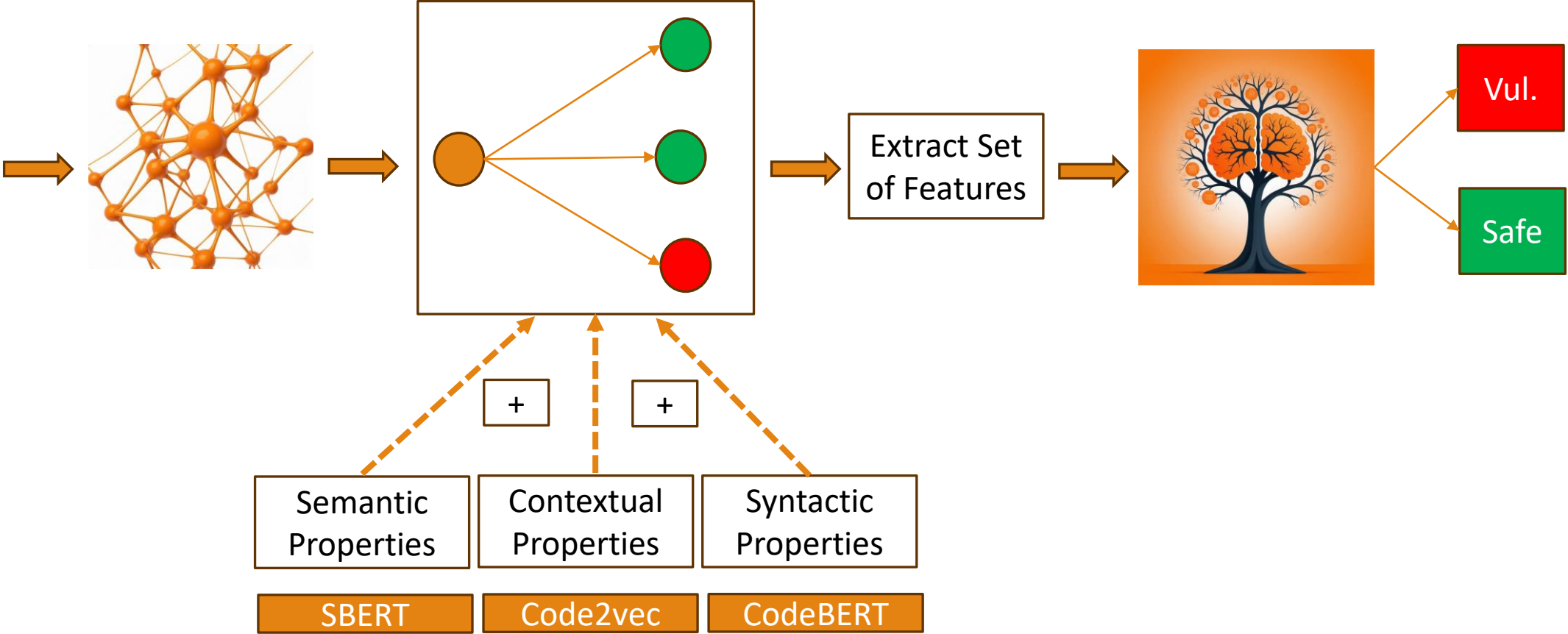
Memory
buffer
vulnerability!

Similarity Metric



The Approach

Data
Function 1
Function 2
.....
.....
Function N



Benefits

- ❑ Utilized multiple dimensions from same data
- ❑ Utilizing neighborhood information

Thus address the dataset limitation!



Evaluation: Devign Dataset

	SBERT	CodeBERT	Code2vec
Accuracy (%)	54.46	62.12	64.6

	Semantic	Syntactic	Contextual	Semantic- Contextual	Semantic- Syntactic	Syntactic- Contextual	VulSim (ours)
Accuracy (%)	60.92	55.01	74.31	75.41	60.92	74.31	75.41

Evaluation: Devign Dataset

	SBERT	CodeBERT	Code2vec
Accuracy (%)	54.46	62.12	64.6

	Semantic	Syntactic	Contextual	Semantic- Contextual	Semantic- Syntactic	Syntactic- Contextual	VulSim (ours)
Accuracy (%)	60.92	55.01	74.31	75.41	60.92	74.31	75.41

Evaluation: Devign Dataset

	SBERT	CodeBERT	Code2vec
Accuracy (%)	54.46	62.12	64.6

	Semantic	Syntactic	Contextual	Semantic-Contextual	Semantic-Syntactic	Syntactic-Contextual	VulSim (ours)
Accuracy (%)	60.92	55.01	74.31	75.41	60.92	74.31	75.41

Evaluation: Devign Dataset

	SBERT	CodeBERT	Code2vec
Accuracy (%)	54.46	62.12	64.6

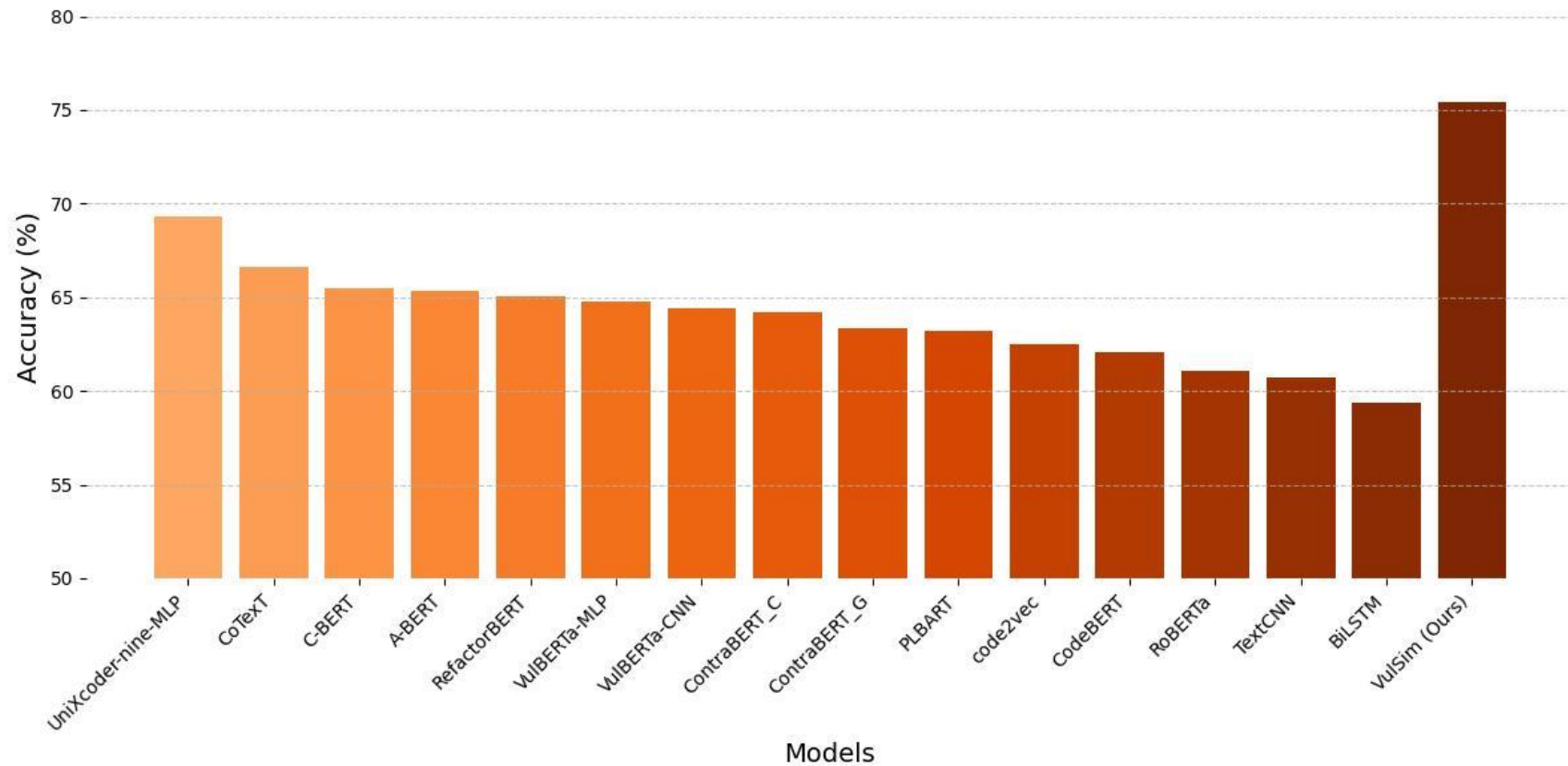
	Semantic	Syntactic	Contextual	Semantic-Contextual	Semantic-Syntactic	Syntactic-Contextual	VulSim (ours)
Accuracy (%)	60.92	55.01	74.31	75.41	60.92	74.31	75.41

Evaluation: Devign Dataset

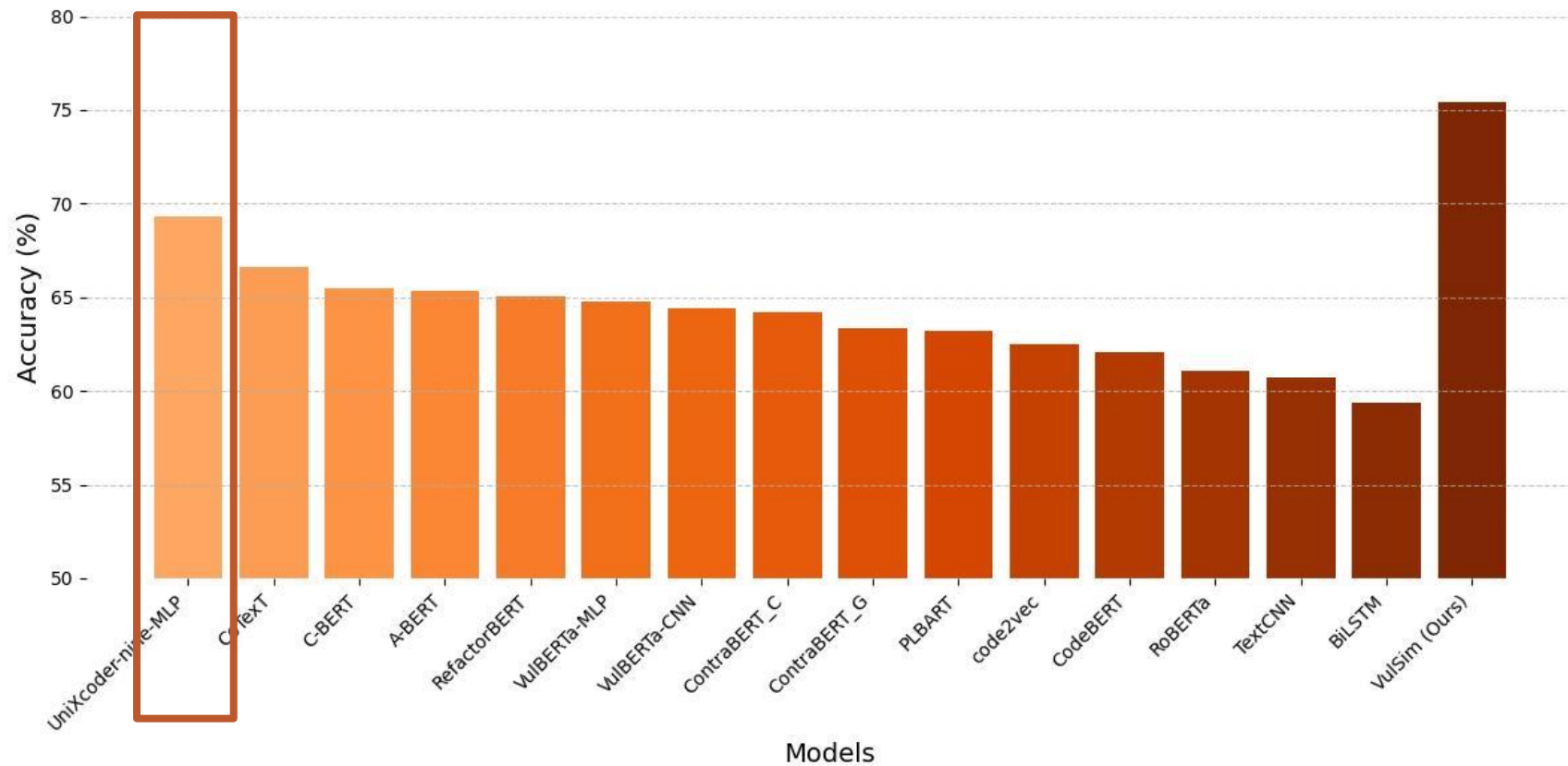
	SBERT	CodeBERT	Code2vec
Accuracy (%)	54.46	62.12	64.6

	Semantic	Syntactic	Contextual	Semantic- Contextual	Semantic- Syntactic	Syntactic- Contextual	VulSim (ours)
Accuracy (%)	60.92	55.01	74.31	75.41	60.92	74.31	75.41

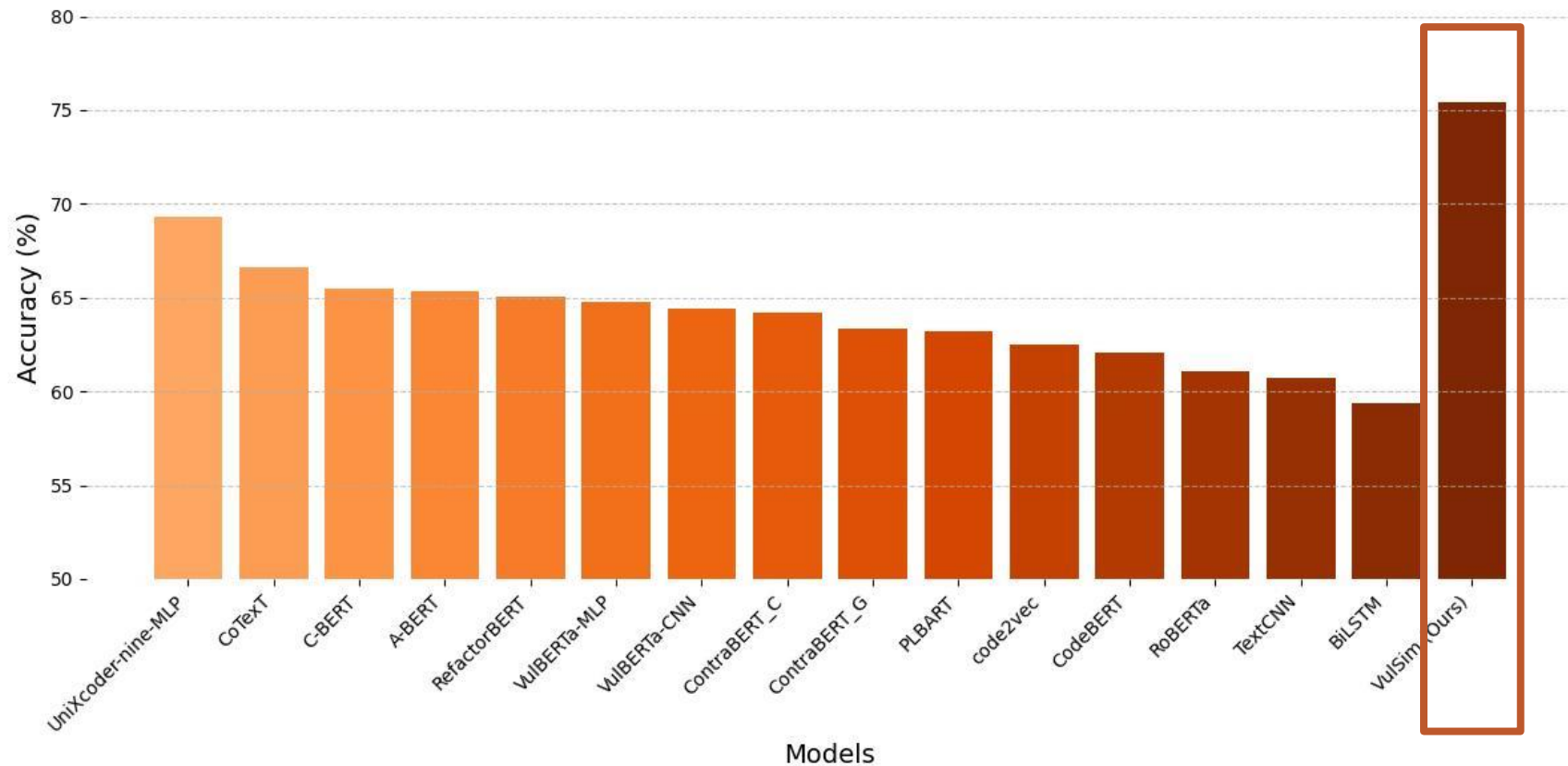
Evaluation: Comparison with CodexGLUE



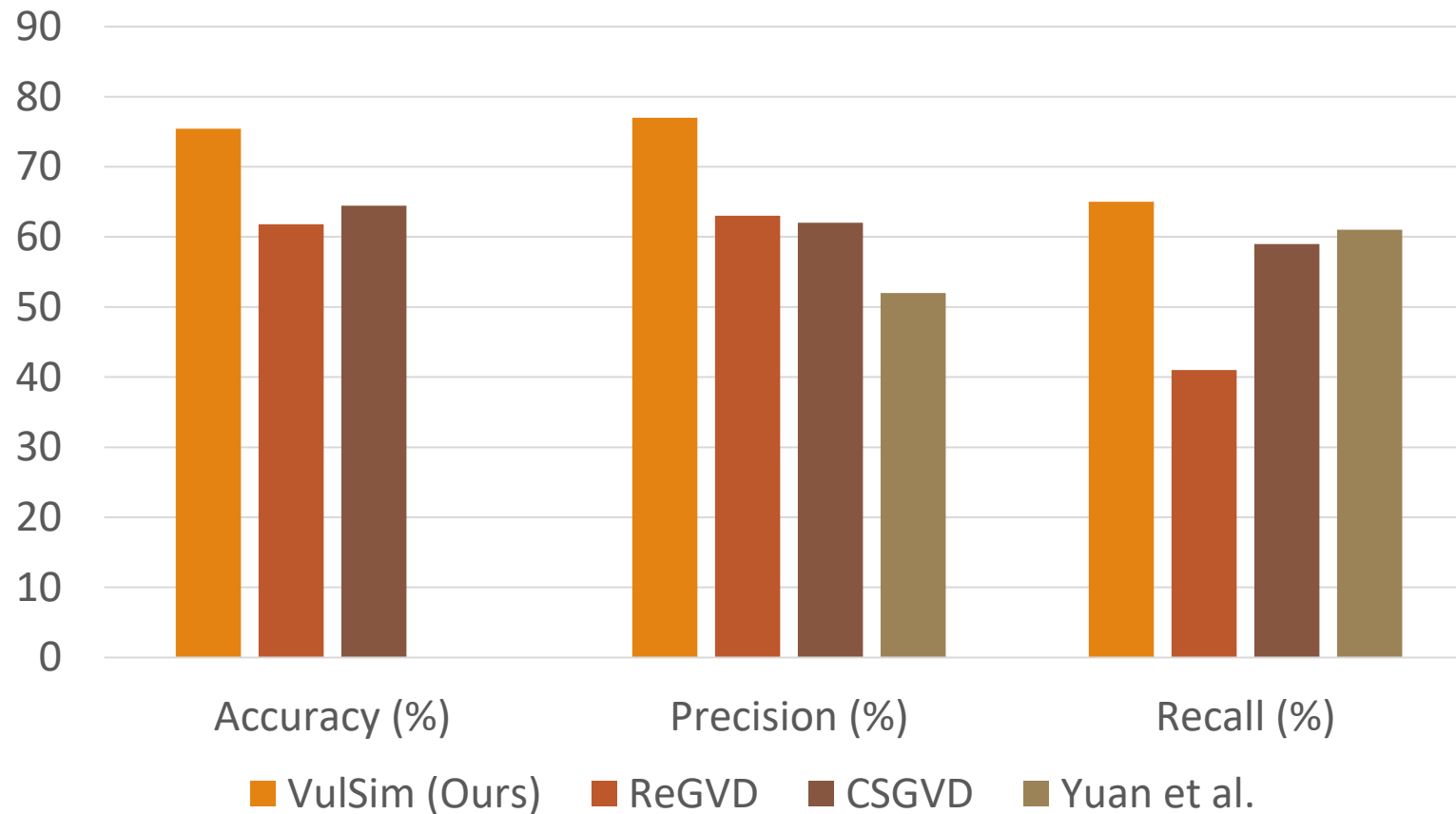
Evaluation: Comparison with CodexGLUE



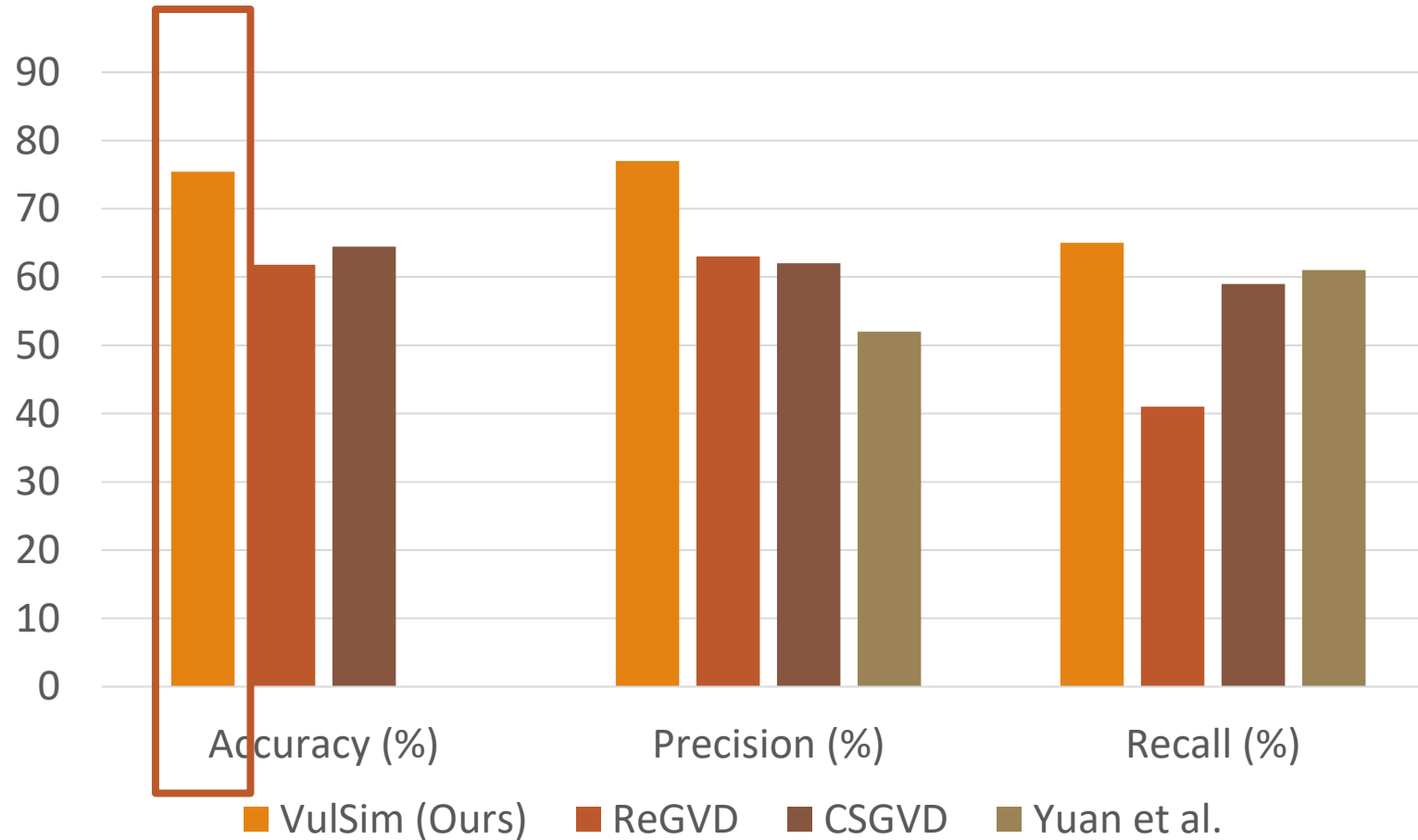
Evaluation: Comparison with CodexGLUE



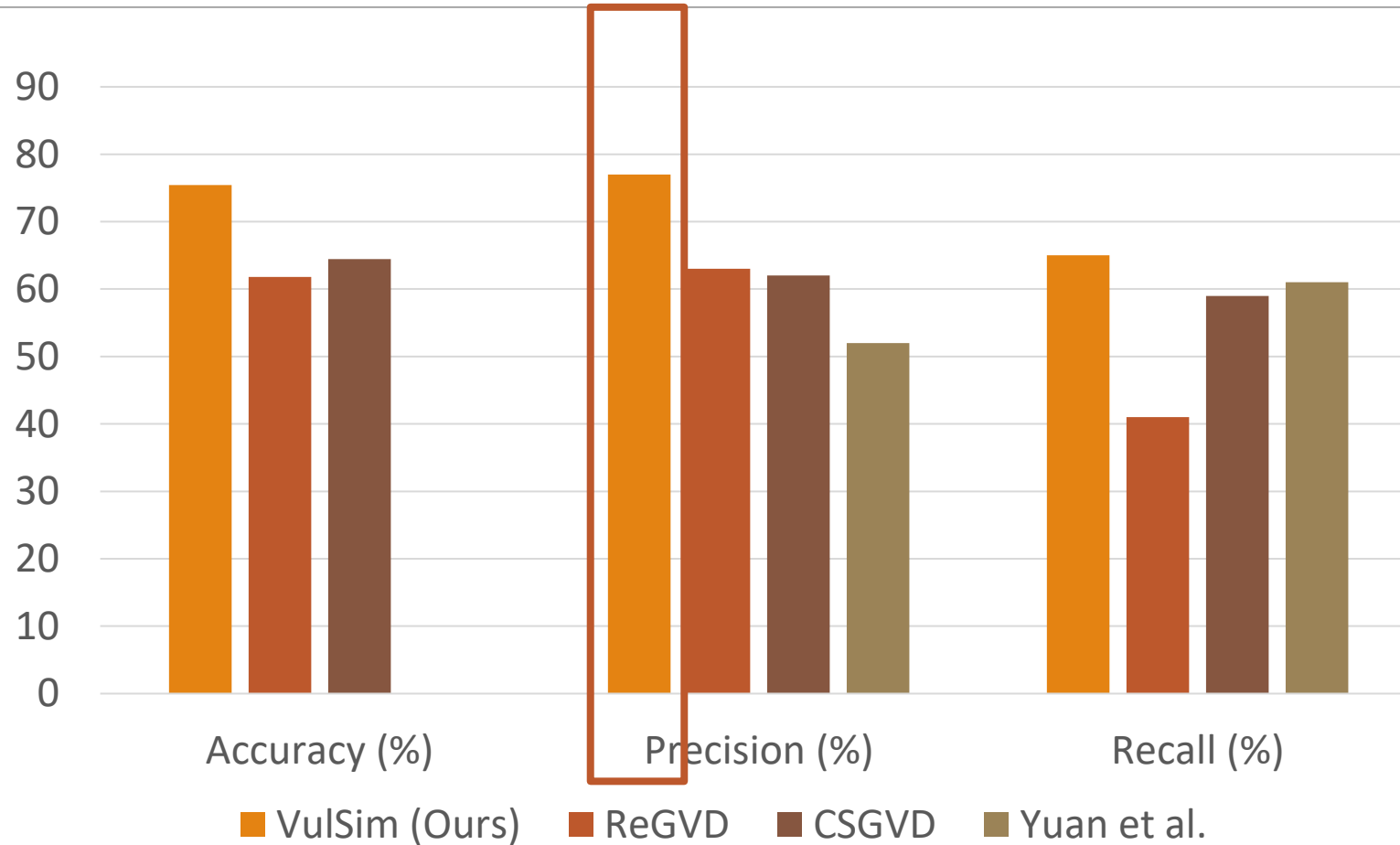
Evaluation: Comparison with SOTA Vulnerability Detection Tools



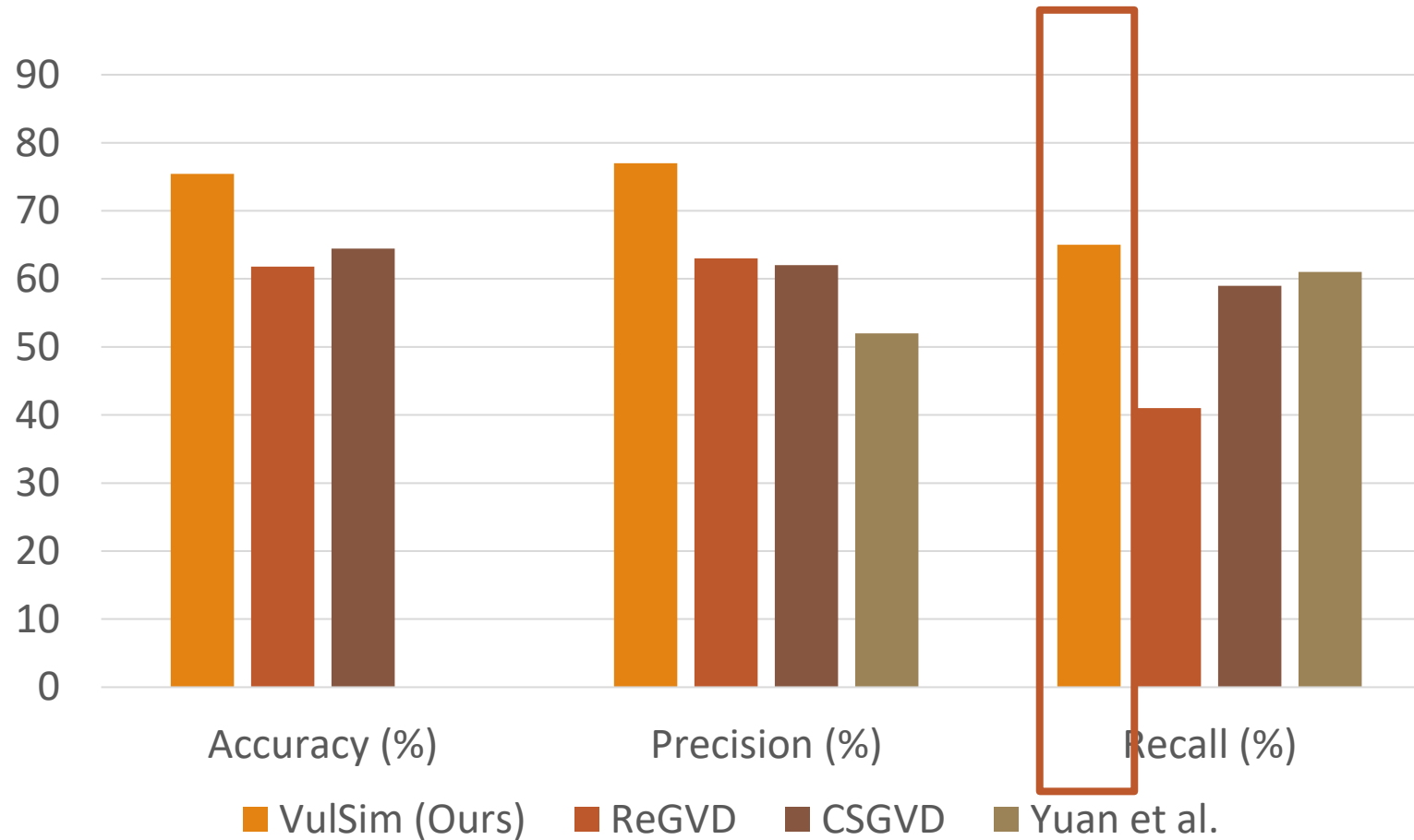
Evaluation: Comparison with SOTA Vulnerability Detection Tools



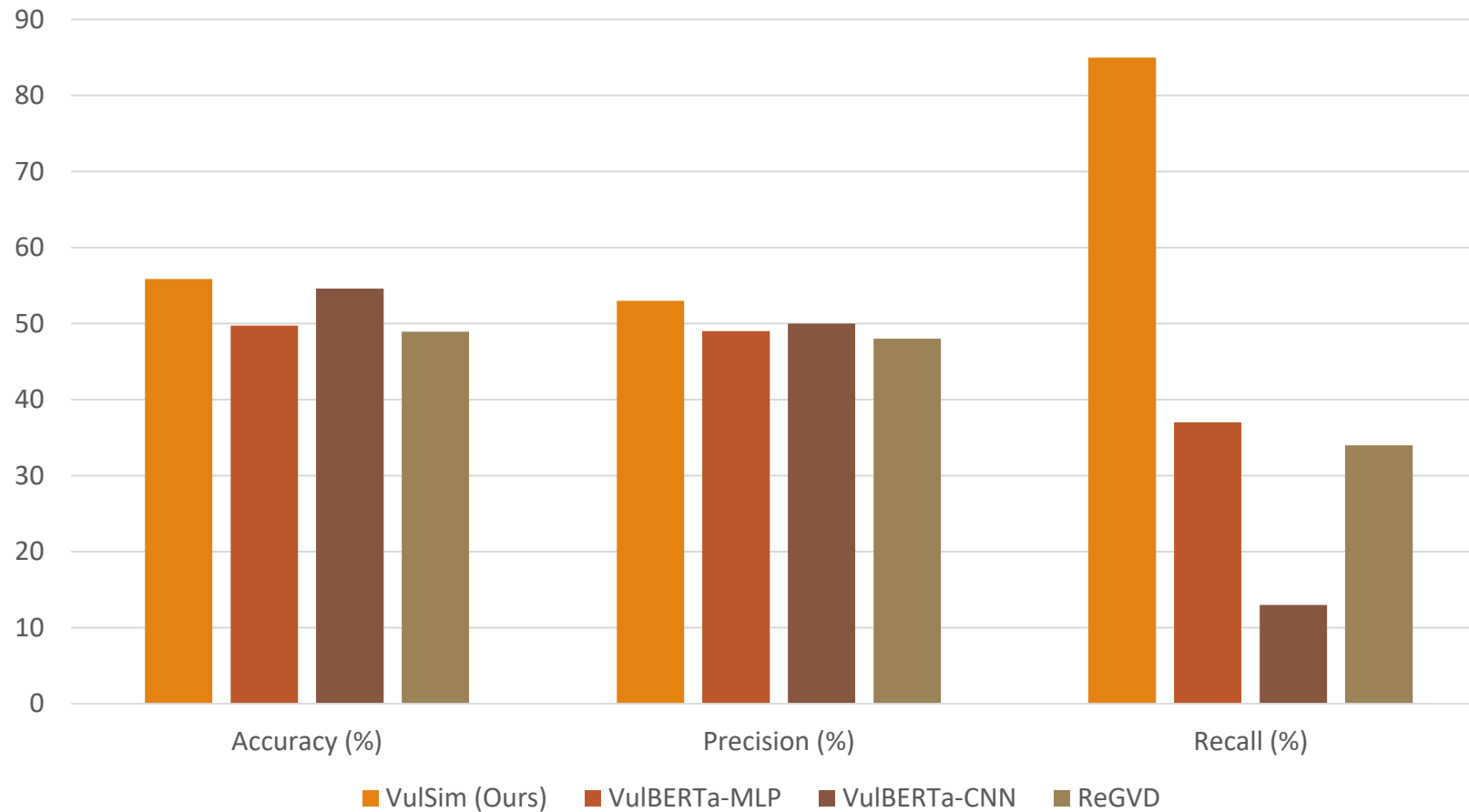
Evaluation: Comparison with SOTA Vulnerability Detection Tools



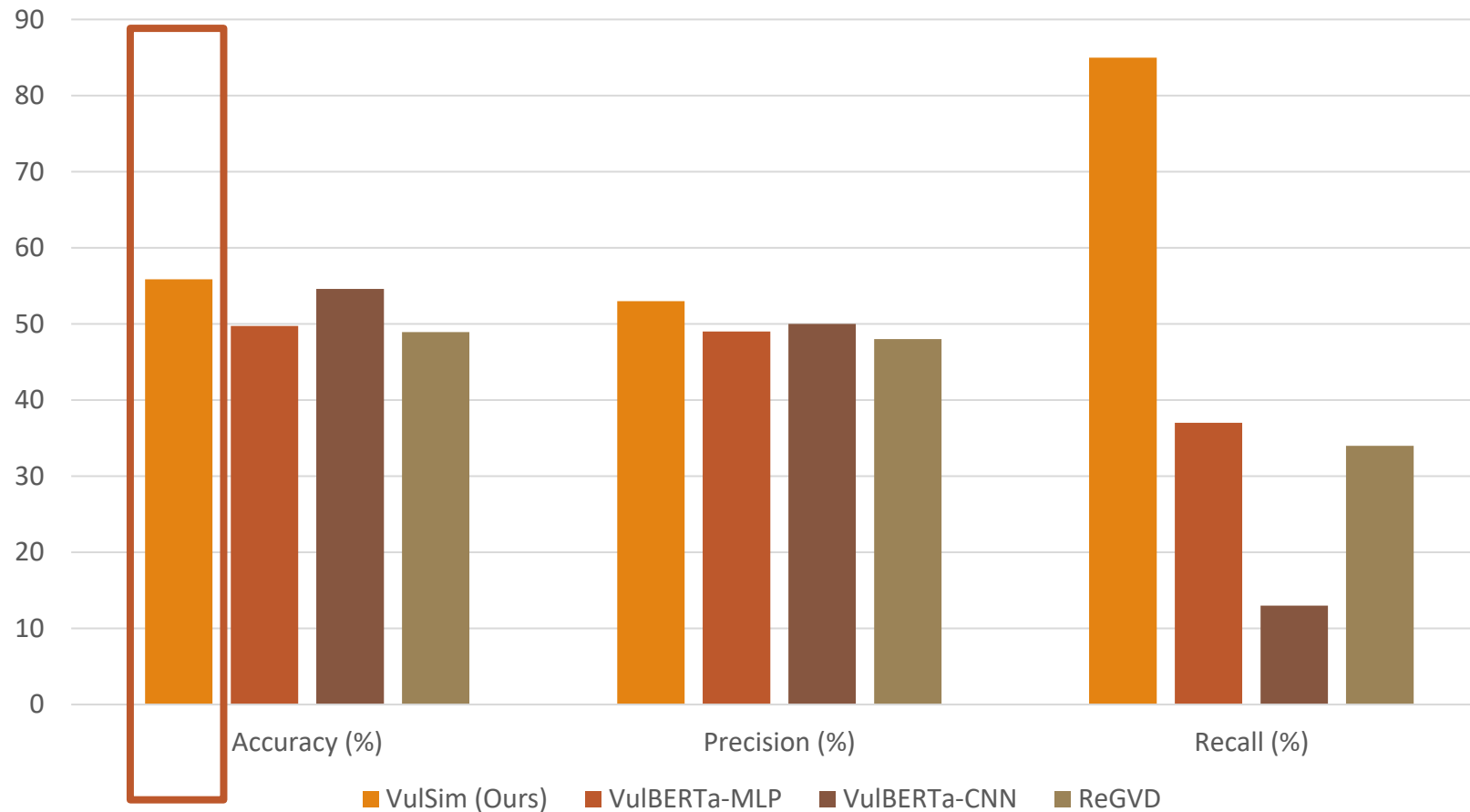
Evaluation: Comparison with SOTA Vulnerability Detection Tools



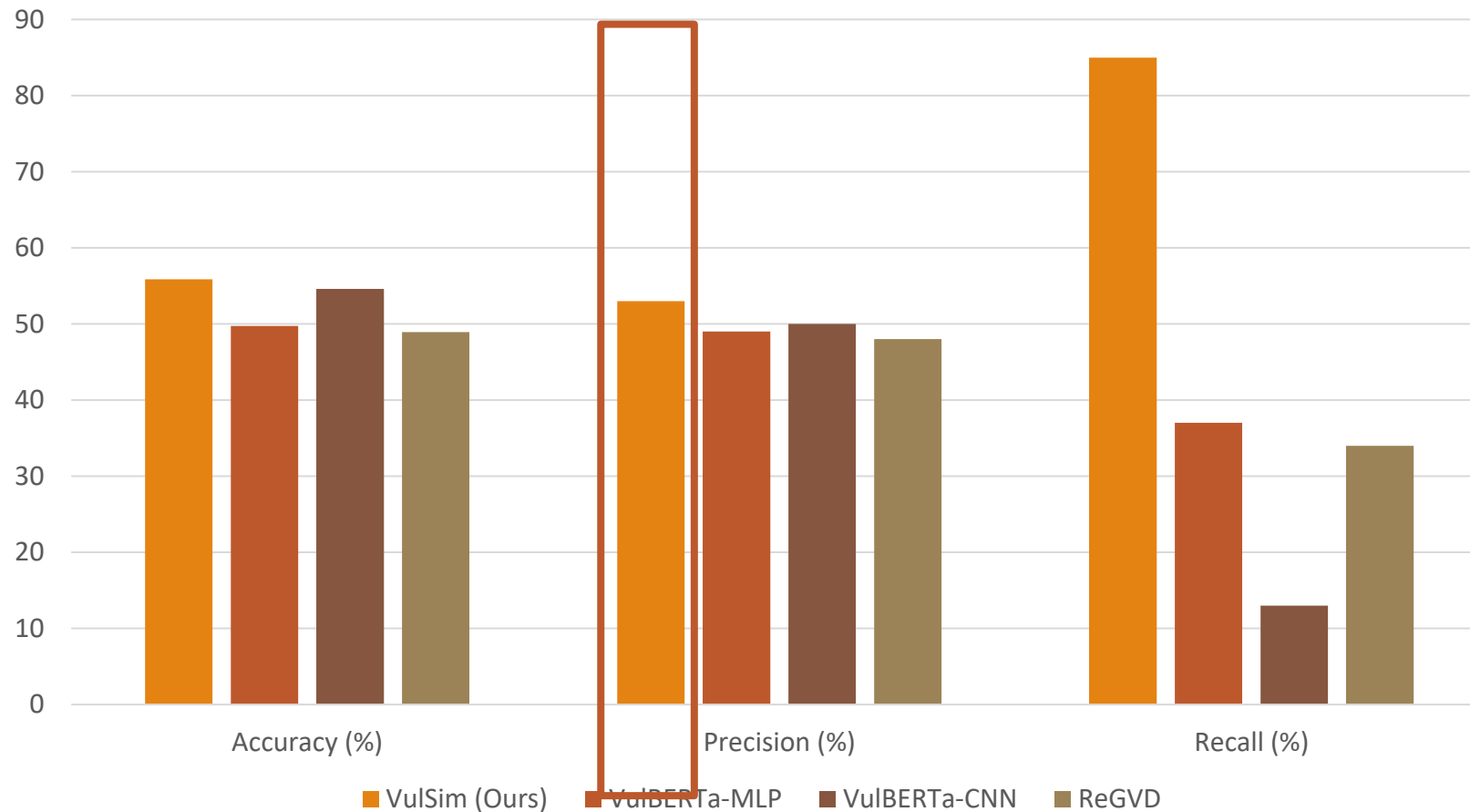
Evaluation: Generalizability on BigVul



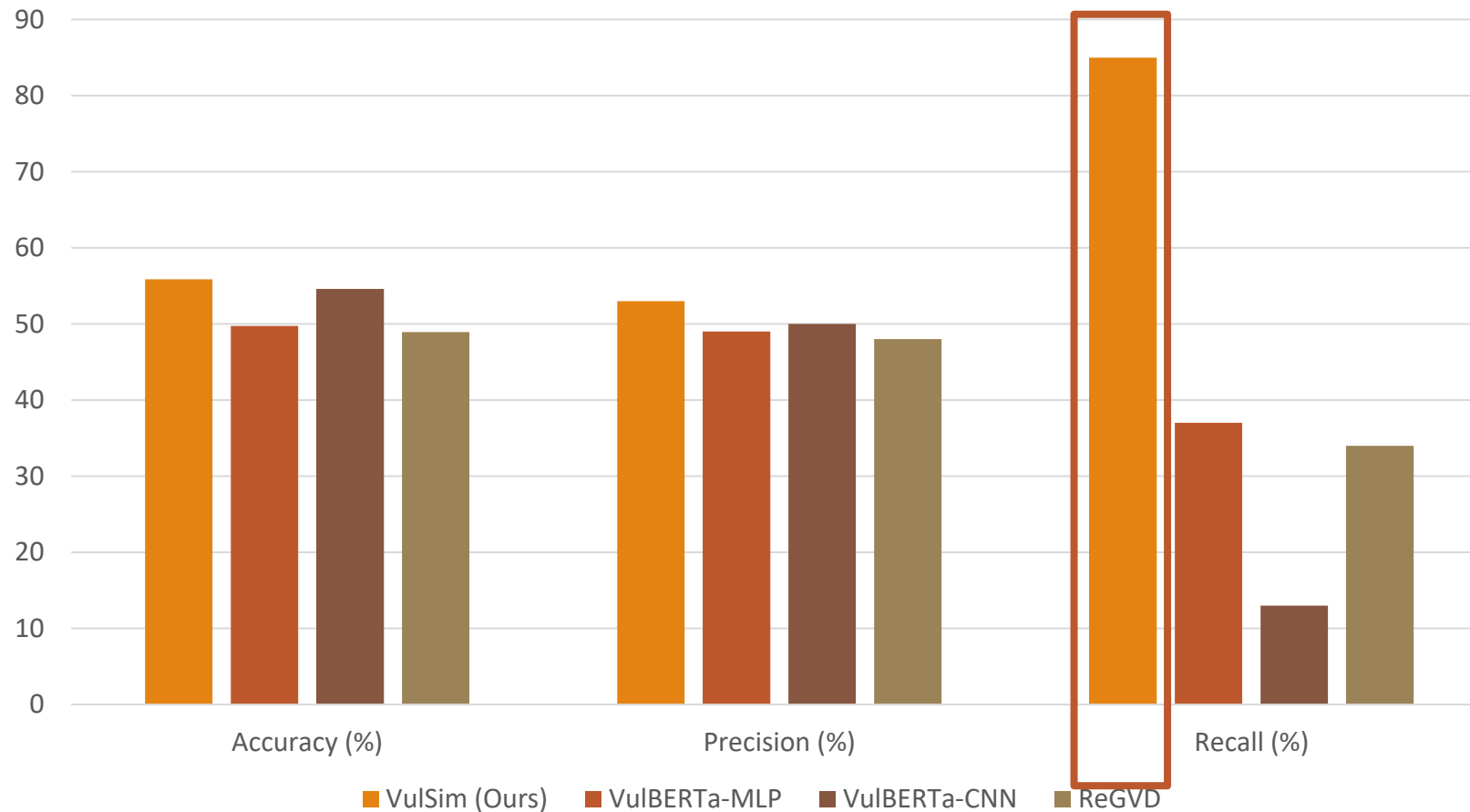
Evaluation: Generalizability on BigVul



Evaluation: Generalizability on BigVul



Evaluation: Generalizability on BigVul



Conclusion

- ❑ Existing datasets are not enough
- ❑ **VulSim**-> incorporates multiple dimension
- ❑ **VulSim**-> incorporates neighbor information
- ❑ Outperforms SOTA techniques
- ❑ Shows enhanced generalizability
- ❑ Our artifacts are available online¹



Thank you!!