RUB CASA Securing the Digital Society

Chair for Software Security

Hallucinating Certificates

Differential Testing of TLS Certificate Validation Using Generative Language Models

Talha Paracha[1,2] Kyle Posluns[2] Kevin Borgolte[1] Martina Lindorfer[3] David Choffnes[2]

[1]Ruhr University Bochum [2]Northeastern University [3]TU Wien

48th International Conference on Software Engineering
Rio de Janeiro, Brazil

Presented at Trajectory Labs, Toronto

Transport Layer Security (TLS)
Introduction

Transport Layer Security (TLS)
Introduction

- Used in the vast majority of websites

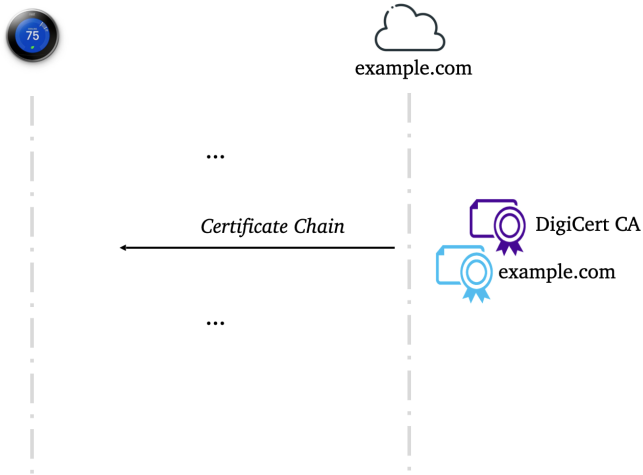
Transport Layer Security (TLS)
Introduction

- Used in the vast majority of websites
- Enabled-by-default in Android and iOS applications

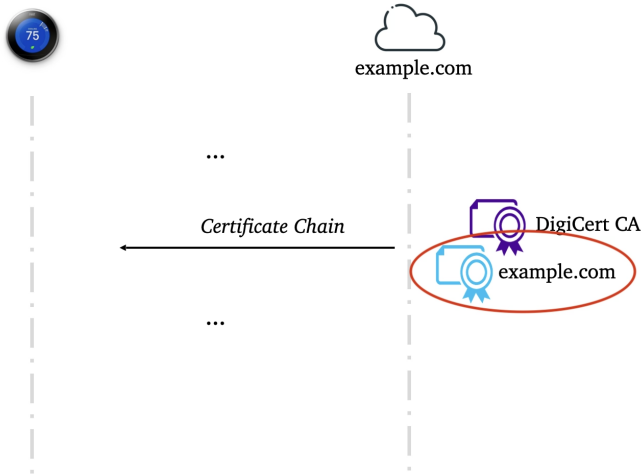
Transport Layer Security (TLS)
Introduction

- Used in the vast majority of websites
- Enabled-by-default in Android and iOS applications
- Default network security protocol in IoT devices

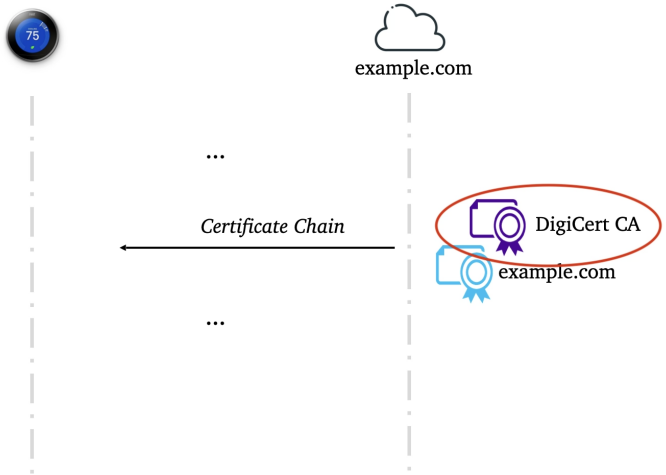
TLS Certificate Validation



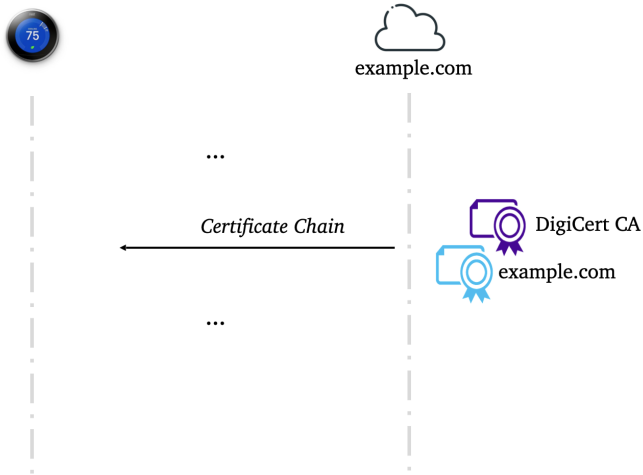
TLS Certificate Validation



TLS Certificate Validation



TLS Certificate Validation



Sample TLS Certificate (Encoded)

```
-----BEGIN CERTIFICATE-----
MIIFYjCCBEqgAwIBAgIQd70NbNs2+RrqIQ/E8FjTDTANBgkqhkiG9w0BAQsFADBX
MQswCQYDVQQGEwJCRTEZMBcGA1UEChMQR2xvYmFsU2lnbiBudi1zYTEQMA4GA1UE
CxMHUm9vdCBDQTEbMBkGA1UEAxMSR2xvYmFsU2lnbiBSb290IENBMB4XDTIwMDYx
........
9U5pCZEt4Wi4wStz6dTZ/CLANx8LZh1J7QJVj2fhMtfTJr9w4z30Z209fOU0iOMy
+qduBmpvvYuR7hZL6Dupszfnw0Skfths18dG9ZKb59UhvmaSGZRVbNQpsg3BZlvi
d0lIKO2d1xozclOzgjXPYovJJIultzkMu34qQb9Sz/yilrbCgj8=
-----END CERTIFICATE-----
```

Sample TLS Certificate (Decoded)

```
talha@talMBPper ~ % openssl x509 -in test.pem -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            77:bd:0d:6c:db:36:f9:1a:ea:21:0f:c4:f0:58:d3:0d
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=BE, O=GlobalSign nv-sa, OU=Root CA, CN=GlobalSign Root CA
        Validity
            Not Before: Jun 19 00:00:42 2020 GMT
            Not After : Jan 28 00:00:42 2028 GMT
        Subject: C=US, O=Google Trust Services LLC, CN=GTS Root R1
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (4096 bit)
                Modulus:
                    00:b6:11:02:8b:1e:e3:a1:77:9b:3b:dc:bf:94:3e:
                    b7:95:a7:40:3c:a1:fd:82:f9:7d:32:06:82:71:f6:
                    f6:8c:7f:fb:e8:db:bc:6a:2e:97:97:a3:8c:4b:f9:
                    2b:f6:b1:f9:ce:84:1d:b1:f9:c5:97:de:ef:b9:f2:
                    a3:e9:bc:12:89:5e:a7:aa:52:ab:f8:23:27:cb:a4:
                    b1:9c:63:db:d7:99:7e:f0:0a:5e:eb:68:a6:f4:c6:
                    5a:47:0d:4d:10:33:e3:4e:b1:13:a3:c8:18:6c:4b:
```

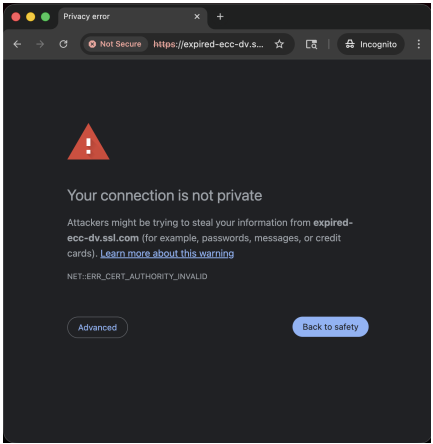
Sample TLS Certificate (Decoded)

```
        X509v3 extensions:
            X509v3 Key Usage: critical
                Digital Signature, Certificate Sign, CRL Sign
            X509v3 Basic Constraints: critical
                CA:TRUE
            X509v3 Subject Key Identifier:
                E4:AF:2B:26:71:1A:2B:48:27:85:2F:52:66:2C:EF:F0:89:13:71:3E
            X509v3 Authority Key Identifier:
                60:7B:66:1A:45:0D:97:CA:89:50:2F:7D:04:CD:34:A8:FF:FC:FD:4B
            Authority Information Access:
                OCSP - URI:http://ocsp.pki.goog/gsr1
                CA Issuers - URI:http://pki.goog/gsr1/gsr1.crt
            X509v3 CRL Distribution Points:
                Full Name:
                  URI:http://crl.pki.goog/gsr1/gsr1.crl

            X509v3 Certificate Policies:
                Policy: 2.23.140.1.2.1
                Policy: 2.23.140.1.2.2
                Policy: 1.3.6.1.4.1.11129.2.5.3.2
                Policy: 1.3.6.1.4.1.11129.2.5.3.3
    Signature Algorithm: sha256WithRSAEncryption
    Signature Value:
```

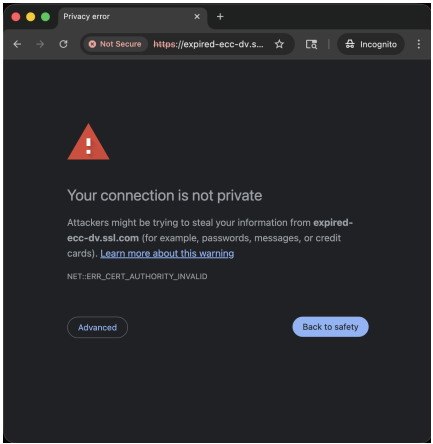
Challenges in Certificate Validation

Challenges in Certificate Validation



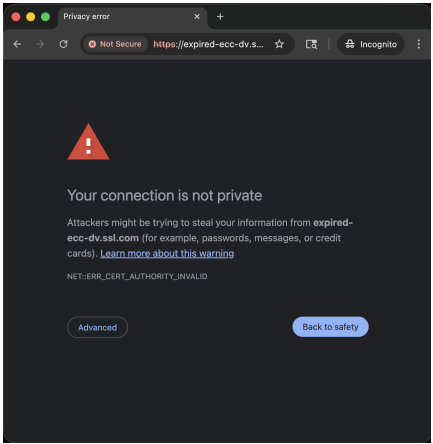
- Certificate validation can neither fail-open, nor fail-close

Challenges in Certificate Validation



- Certificate validation can neither fail-open, nor fail-close
- Validation software susceptible to programming bugs

Challenges in Certificate Validation

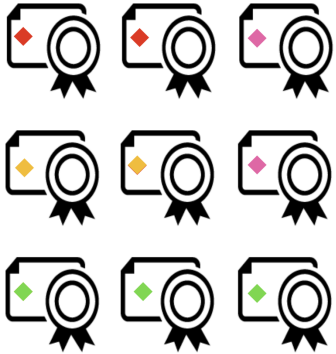


- Certificate validation can neither fail-open, nor fail-close
- Validation software susceptible to programming bugs
- Protocol RFCs are ambiguous

Testing TLS Certificate Validation

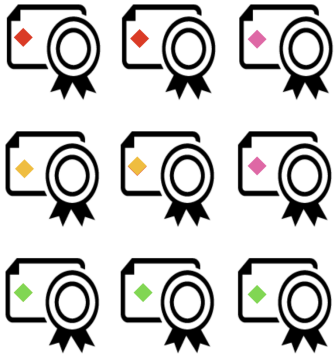
Testing TLS Certificate Validation

Approach 1: Use Real World Certificates



Testing TLS Certificate Validation

Approach 1: Use Real World Certificates

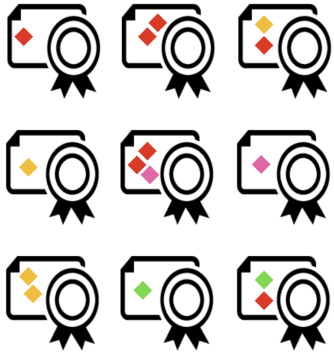


Issue

Edge cases are rare...

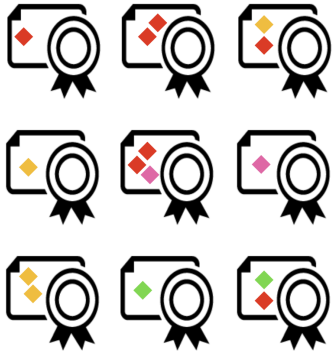
Testing TLS Certificate Validation

Approach 2: Use Mutated Certificates (*Frankencerts*)



Testing TLS Certificate Validation

Approach 2: Use Mutated Certificates (*Frankencerts*)

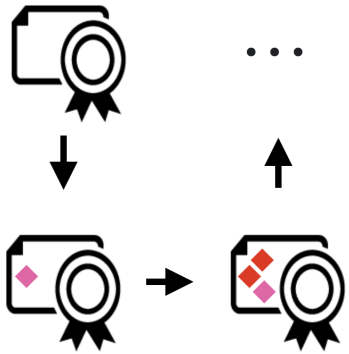


Issue

Mutations tend to make certificates invalid…

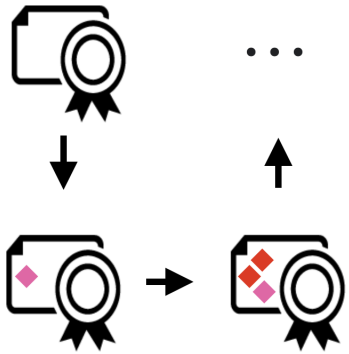
Testing TLS Certificate Validation

Approach 2: Use Code Coverage (*Transcert*)



Testing TLS Certificate Validation

Approach 2: Use Code Coverage (*Transcert*)

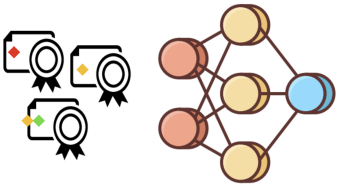


Issue

Certificate generation is slow and cannot be parallelized...

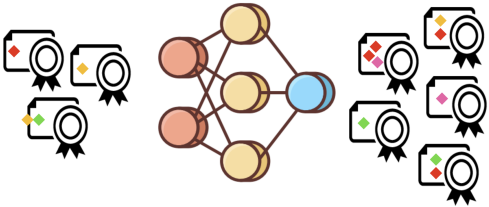
Testing TLS Certificate Validation

Our Approach: Use Generative Language Models (*MLCerts*)



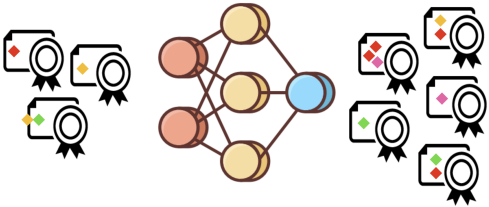
Testing TLS Certificate Validation

Our Approach: Use Generative Language Models (MLCerts)



Testing TLS Certificate Validation

Our Approach: Use Generative Language Models (MLCerts)



Features

- Can learn a quasi grammar
- Can be parallelized

Testing TLS Certificate Validation

MLCerts and Model Hallucinations

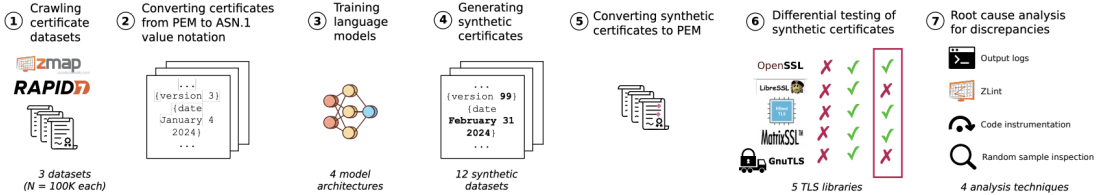
Prompt How many ‘r’ characters appear in the word “strawberry”

Answer There are two ‘r’ characters in the word “strawberry”

Model Hallucinations

Syntactically correct, semantically invalid outputs.

Methodology



Methodology

Step 1: Training Datasets

3 datasets
(N = 100K each)

Methodology

3 datasets
(N = 100K each)

Step 1: Training Datasets

Insight

Multiple datasets to learn different certificate features.

Methodology

3 datasets
(N = 100K each)

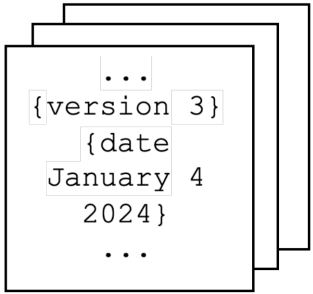
Step 1: Training Datasets

- IPv4, Modern, and Balanced datasets
- 100K TLS Certificates

Insight

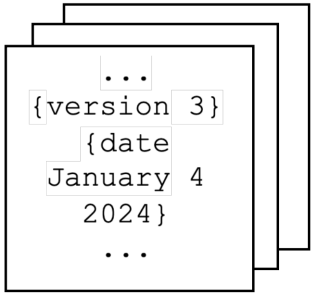
Multiple datasets to learn different certificate features.

Methodology



Step 2: Model Inputs

Methodology



Step 2: Model Inputs

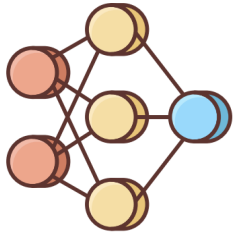
- PyCrate to convert certificates from PEM to ASN.1 value notation.

Insight

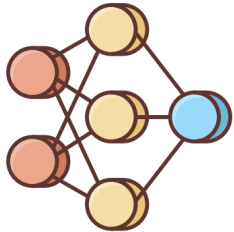
TLS certificates can be represented in a verbose textual notation.

Methodology

Step 3: Language Models



Methodology

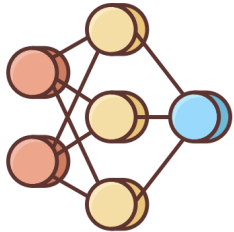


Step 3: Language Models

Insight

Unclear if large language models (LLMs) would help or hurt testing.

Methodology



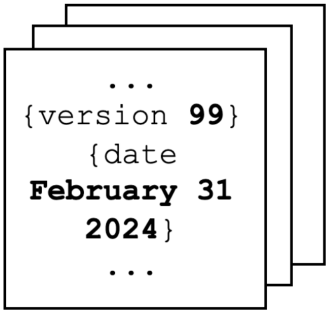
Step 3: Language Models

- Use RNNs and GPTs.
  - RNN-Small (1 mil. parameters).
  - RNN-Medium (10 mil. parameters).
  - GPT-Finetuned (125 mil. parameters).
  - GPT (125 mil. parameters).

Insight

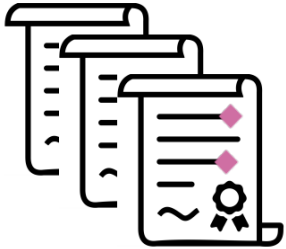
Unclear if large language models (LLMs) would help or hurt testing.

Methodology



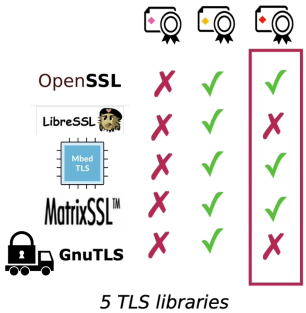
Step 4: Generate Synthetic Certificates

Methodology



Step 5: Convert into PEM format

Methodology



Step 6: Differential Testing

Methodology

Step 7: Discrepancy Analysis

Output logs

ZLint

Code instrumentation

Random sample inspection

4 analysis techniques

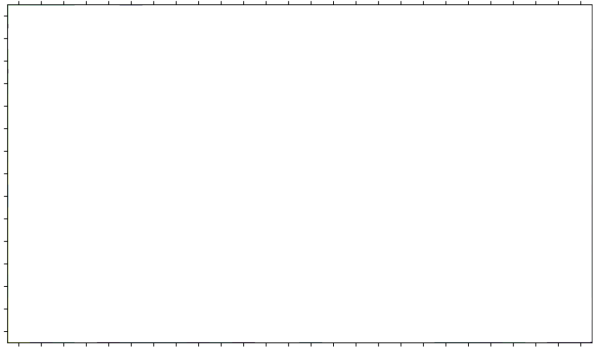
Key Result # 1

MLCerts Finds Unique Discrepancies

Key Result # 1

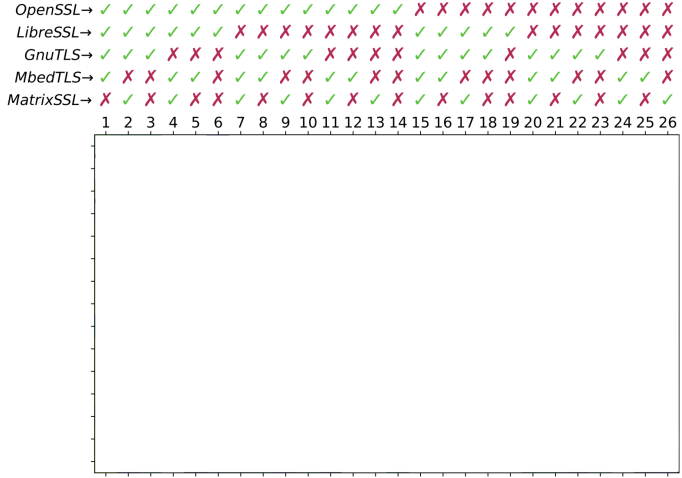
MLCerts Finds Unique Discrepancies

OpenSSL→ ✓
LibreSSL→ ✓
GnuTLS→ ✓
MbedTLS→ ✓
MatrixSSL→ ✗



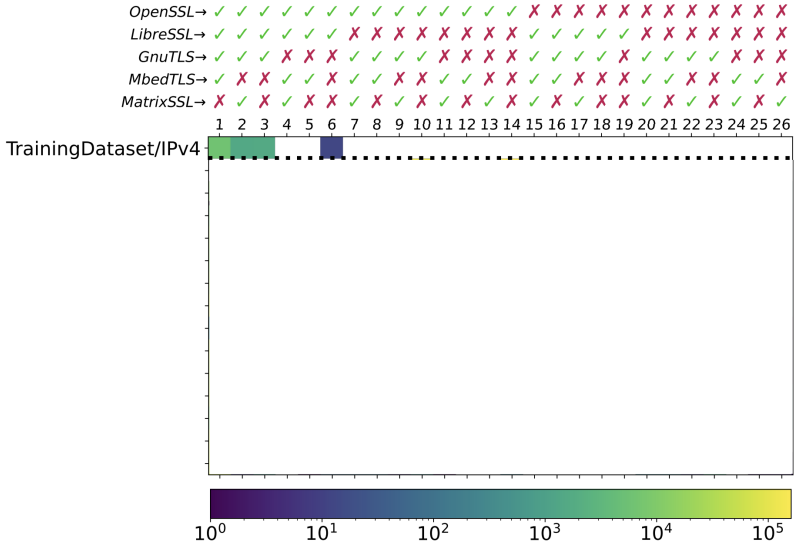
Key Result # 1

MLCerts Finds Unique Discrepancies



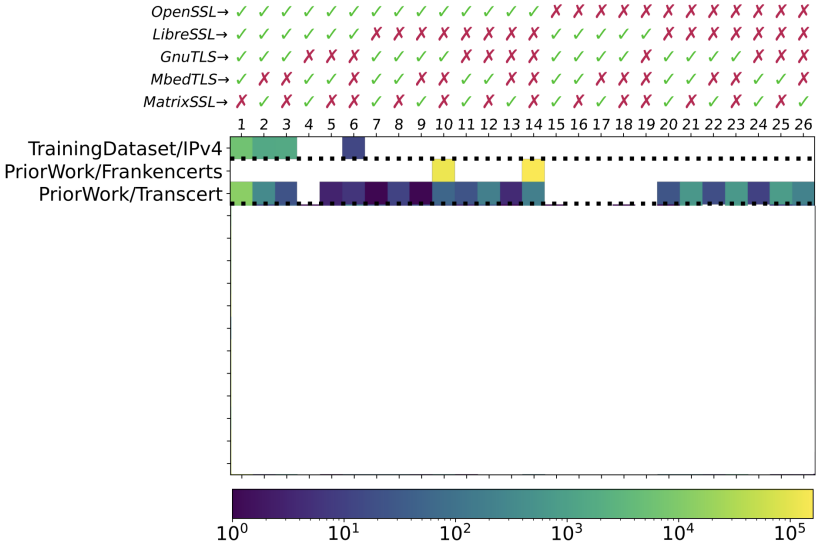
Key Result # 1

MLCerts Finds Unique Discrepancies



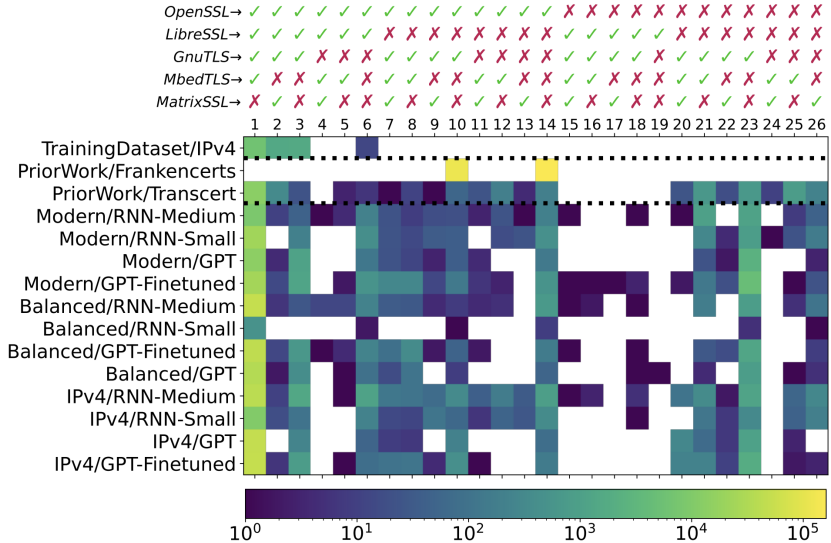
Key Result # 1

MLCerts Finds Unique Discrepancies



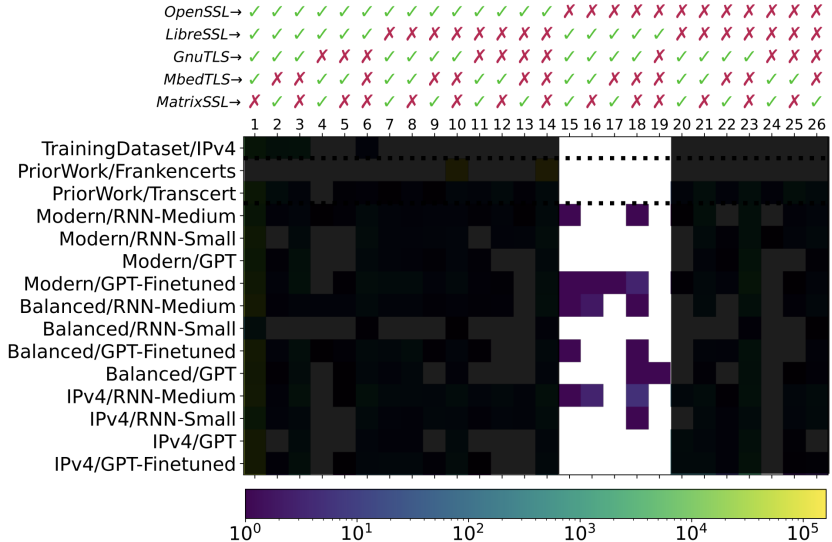
Key Result # 1

MLCerts Finds Unique Discrepancies



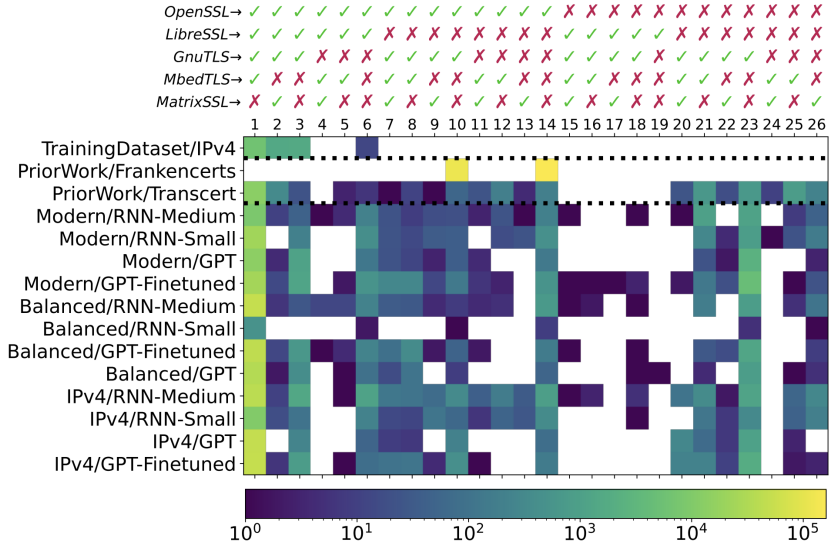
Key Result # 1

MLCerts Finds Unique Discrepancies



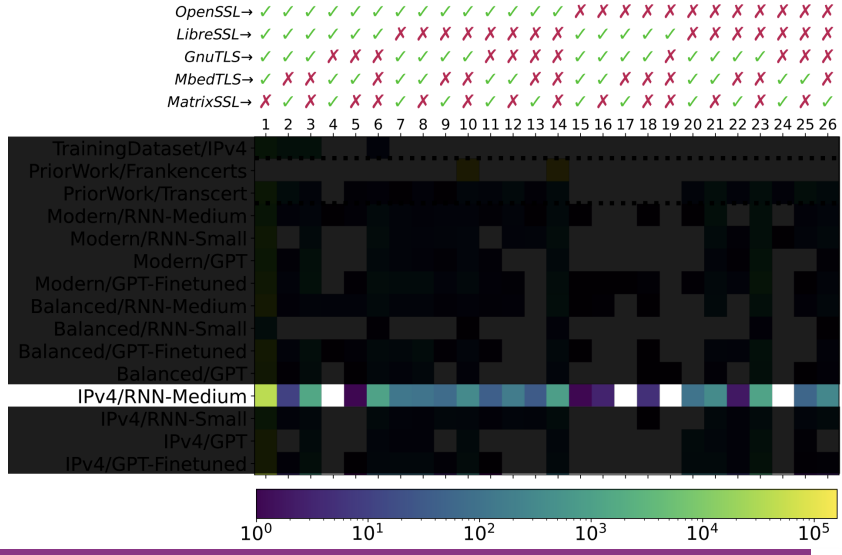
Key Result # 1

MLCerts Finds Unique Discrepancies



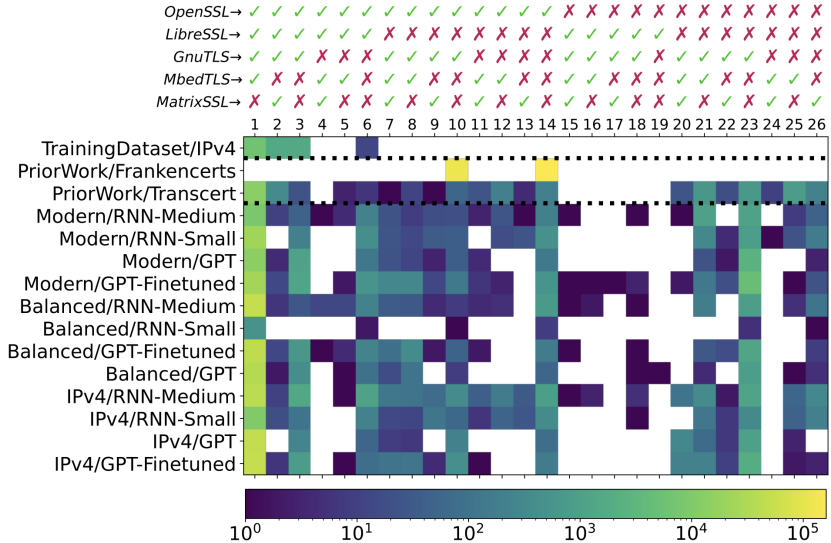
Key Result # 1

MLCerts Finds Unique Discrepancies



Key Result # 1

MLCerts Finds Unique Discrepancies

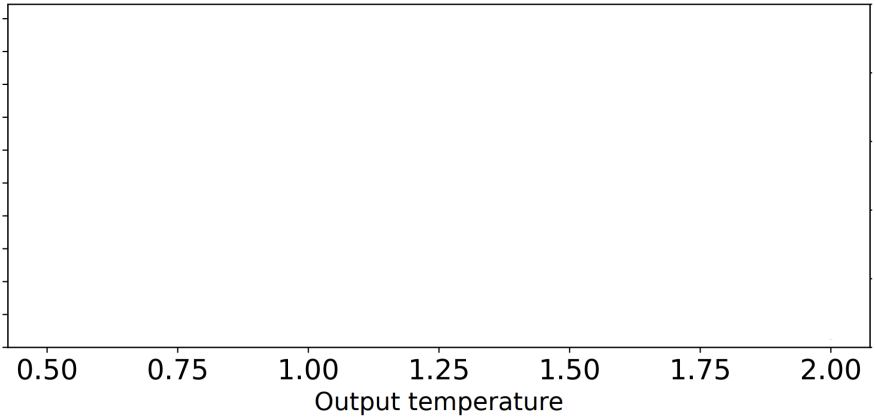


Key Result # 2

Model Hallucinations Help Trigger More Discrepancies

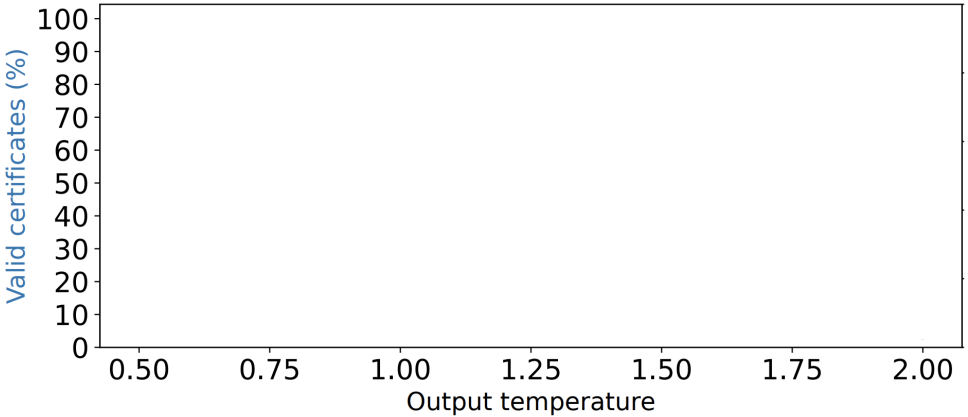
Key Result # 2

Model Hallucinations Help Trigger More Discrepancies



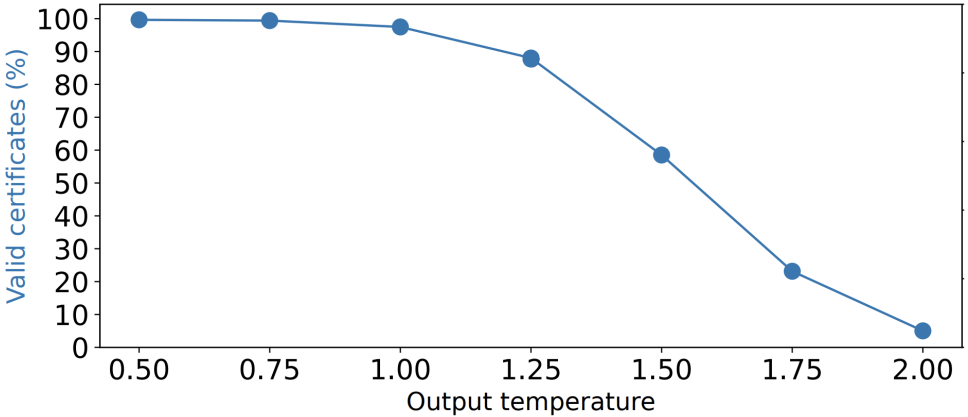
Key Result # 2

Model Hallucinations Help Trigger More Discrepancies



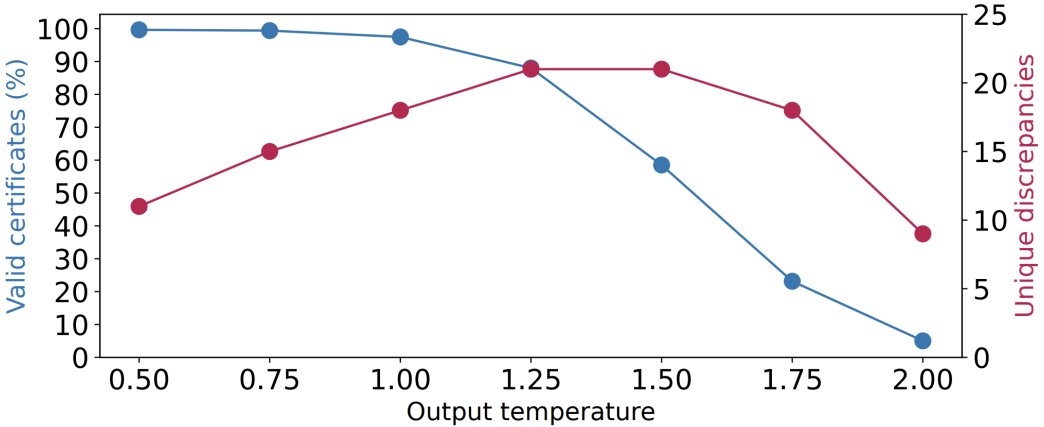
Key Result # 2

Model Hallucinations Help Trigger More Discrepancies



Key Result # 2

Model Hallucinations Help Trigger More Discrepancies



Key Result # 3
MLCerts Finds New Bugs

Key Result # 3
MLCerts Finds New Bugs

- GnuTLS accepts certificates expiring on *February 31st*.
- Some libraries allow for leap seconds in time value *YYMMDDHHMM**60*** instead of *YYMMDDHHMM**[0-59]***

Takeaways

- Language models can learn an approximate input grammar suitable for testing.
- Large language models are not necessarily needed.
- Techniques needed to “guide” the model towards producing diverse testcases.

Paper and artifacts:

`softsec.link`

Funded by

DFG Deutsche Forschungsgemeinschaft
German Research Foundation

CASA
Securing the Digital Society

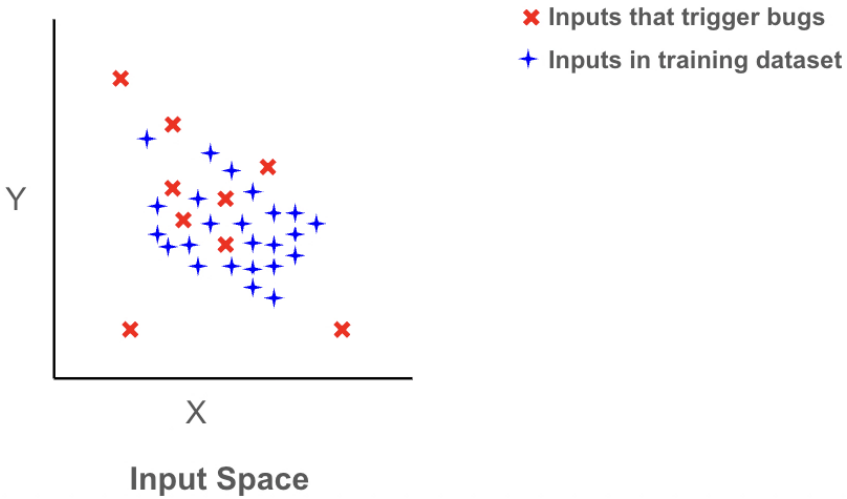
NSF

WWTF
Vienna Science and Technology Fund

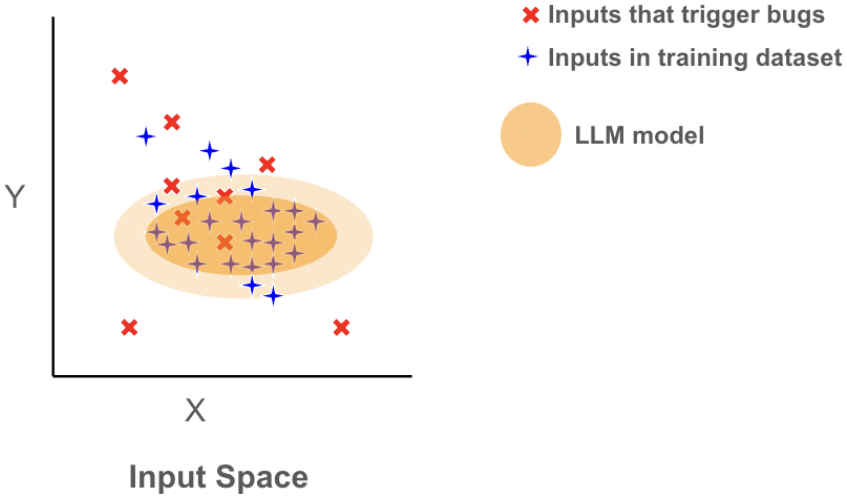
Exploring Input Space “Intelligently”

Exploring Input Space “Intelligently”

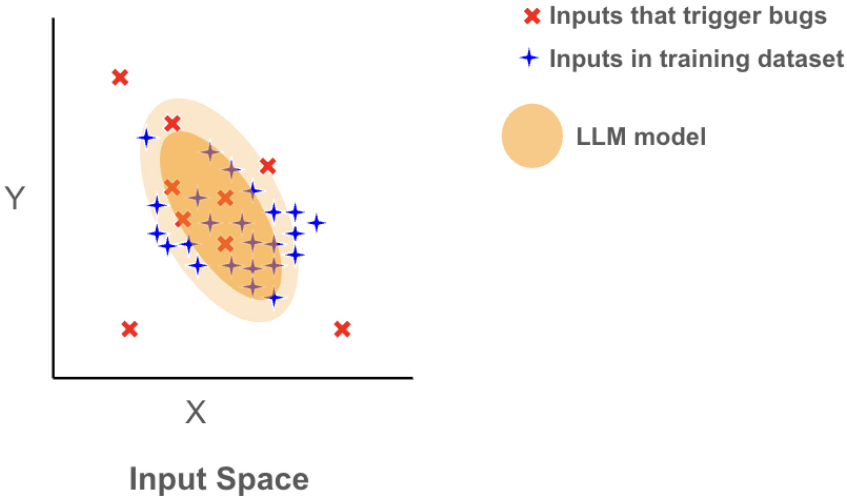
Exploring Input Space “Intelligently”



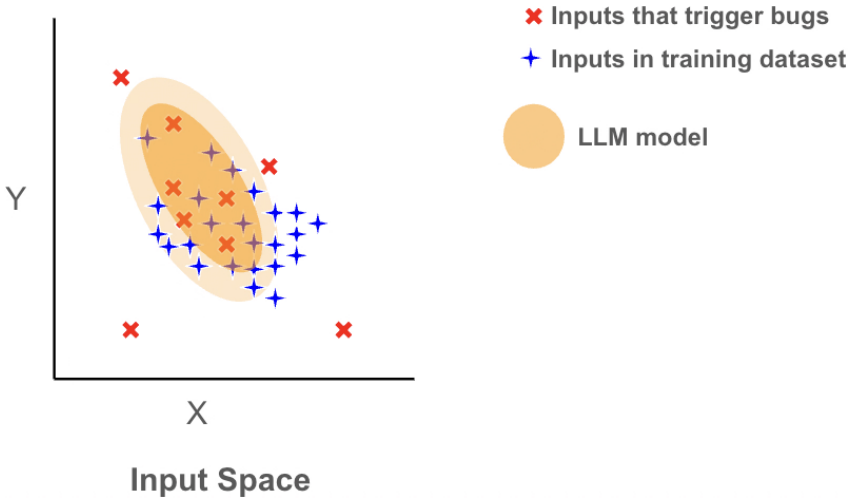
Exploring Input Space “Intelligently”



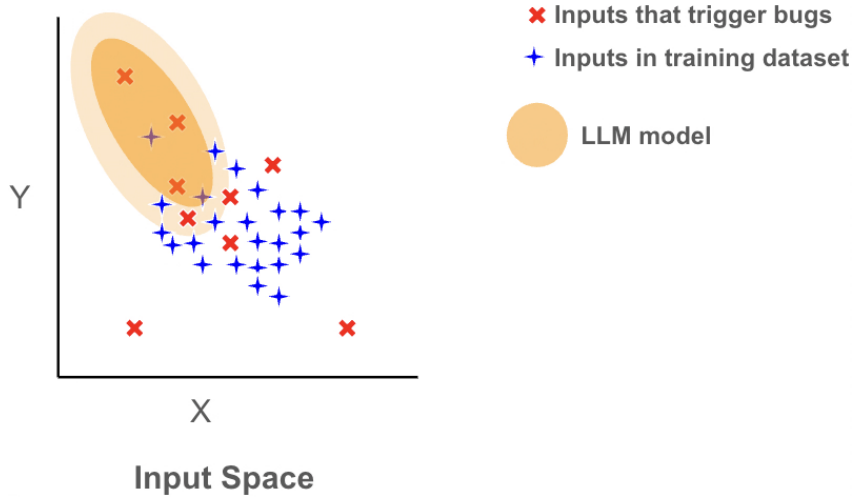
Exploring Input Space “Intelligently”



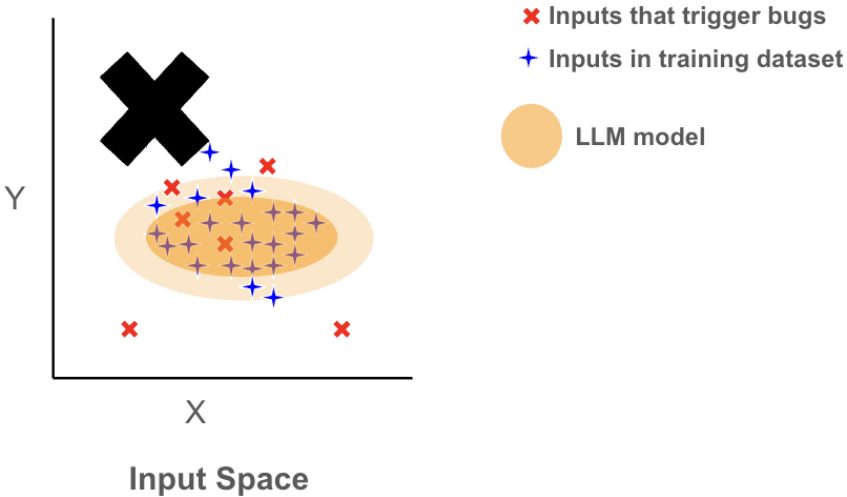
Exploring Input Space “Intelligently”



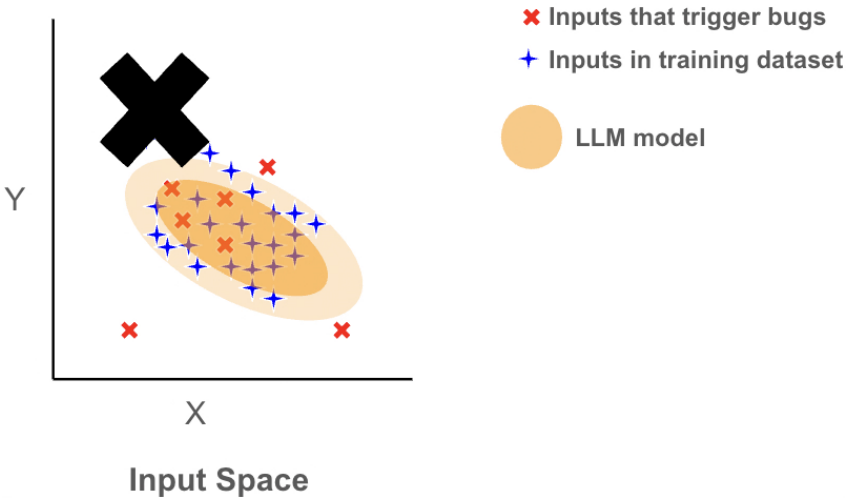
Exploring Input Space “Intelligently”



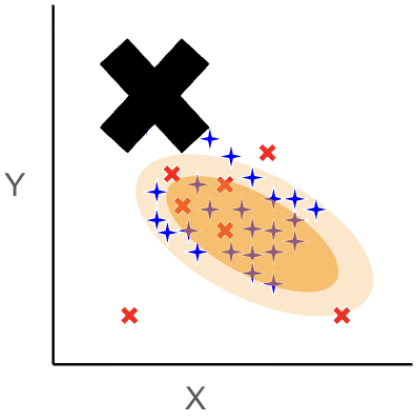
Exploring Input Space “Intelligently”



Exploring Input Space “Intelligently”



Exploring Input Space “Intelligently”



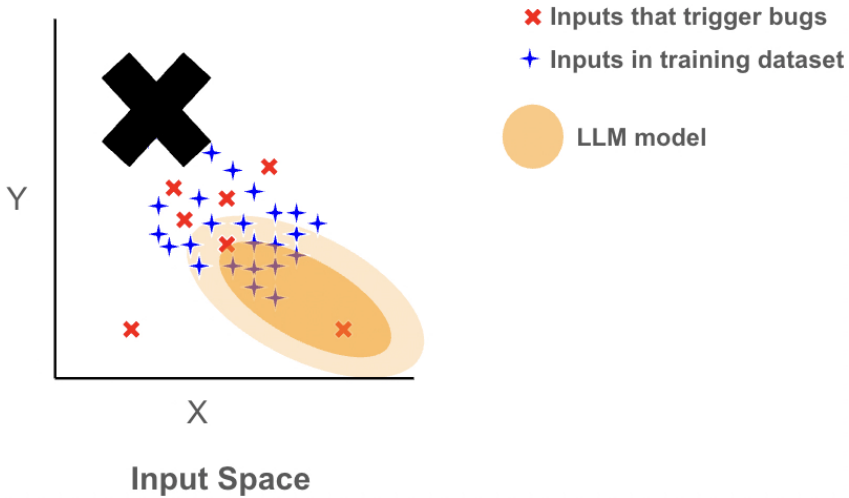
Input Space

Inputs that trigger bugs

Inputs in training dataset

LLM model

Exploring Input Space “Intelligently”



Exploring Input Space “Intelligently”