MLinter: Learning Coding Practices from Examples —Dream or Reality?

SANER 2023 - RENE

Corentin LATAPPY
Jean-Rémy FALLERI
Xavier BLANC
Quentin PEREZ
Christelle URTADO
Cédric TEYTON
Thomas DEGUEULE
Sylvain VAUTTIER

Orchid Room
Thursday, March 23, 2023

LaBRI
IMT Mines Alès
École Mines-Télécom
promyze

Coding practices

eqeqeq

Require the use of `===` and `!==`

It is considered good practice to use the type-safe equality operators `===` and `!==` instead of their regular counterparts `==` and `!=`.

Examples of **incorrect** code for this rule:

```
1    a == b
2    foo == true
```

Examples of **correct** code for this rule:

```
1    a === b
2    foo === true
```

Rule « eqeqeq » from linter ESLint
(https://eslint.org/docs/latest/rules/eqeqeq)

src/server.ts · 33ed2672641f55c0af000c4a8195d042e163369c · promyze / v2OverfullBackend · GitLab (0) | SendCorrection.java (1) | ApiService.java (0)

```
36
37          final Correction correction = new Correction(craftTagId, spaceId, contentFile);
38
39          final ICraftTagRepo craftTagRepo = CraftTagRepo.getInstance();
40
```

Cédric Teyton a noté **Use Task.Backgroundable for background tasks**

```
41          ProgressManager.getInstance().run(new Task.Backgroundable(null, "Promyze") {
```

Commenter — Effacer le tag — Tag suivant

```
42              @Override
43              public void run(@NotNull ProgressIndicator indicator) {
44                  try {
45                      craftTagRepo.addCorrectionToCraftTag(correction);
46
47                      PromyzeNotifier.notifyUserSuccess("Correction added !");
48                  } catch (AbstractPromyzeException exception) {
49                      if (exception instanceof NotFoundException) {
50                          storageService.clearLastCraftTagNegativeAdded();
51
52                          PromyzeNotifier.notifyUserError("Correction not added... Negative tag not found.");
53                      } else {
54                          PromyzeNotifier.notifyPromyzeError(exception);
55                      }
56                  }
57              }
58          });
59      }
60  }
61 }
62
```

Mode lecture — Pratiques existantes

Toutes À valider (3)

Rechercher une pratique

Toutes | Clean Code | JS | TS | DDD | Test | Voir plus

TS

Use a specific type for objects with unknown keys

Arthur Magne 17/09/2021

Error

Preserve original context of errors

Cédric Teyton 17/09/2021

Jetbrains

Use Task.Backgroundable for background tasks

Cédric Teyton 17/09/2021

Extract Component if main component is too big or doesn't follow SRP

Quentin 17/09/2021

Use loader while APIs working

Quentin 17/09/2021

CSS

Use global style

Quentin 17/09/2021

nodejs

Use 'cluster' module from node to run multi-threads in node js

Cédric Teyton 10/09/2021

Clean Code

File is specific or agnostic

Quentin 10/09/2021

Clean Code React

Use stylesheet if possible (aka

React

Use dedicated service to make

Clean Code TS +4

Extract shared interface in

Clean Code

Créer une pratique

craftTagReferenceVectorsManagementModal.component.ts

```
15  import { getFocusedCraftTagSelectorVM } from './viewModel';
16  import {
17      selectCraftTagVector,
18      unselectCraftTagVector,
19  } from './../../../../corelogic/usecases/craftTagReferenceVectorsManagement/toggleCraftTagVector/toggleCraf
20  import {CodeMirrorEditor} from "themis-app/adapters/primary/shared/codeMirror/codeMirrorService";
21
22  type Props = {
23      focusedCraftTag: FocusedCraftTag | null;   Magne, 12/08/2021 09:13 • ✨ 🚧  Vector management working in
24  };
25
26  type State = {
27      currentTagIndex: number;
28      craftTagIds: string[];
29      craftTagReferenceName: string;
30      editorOptions: {};
31      editor: any;
32      hasNoExampleAvailable: boolean;
33  };
34  export default class CraftTagReferenceVectorsManagementModal {
35      props: Props = {
36          focusedCraftTag: null,
37      };
38
39      state: State = {
40          currentTagIndex: 0,
41          craftTagReferenceName: '',
42          craftTagIds: [],
43          editorOptions: {},
44          editor: null,
```

Props › focusedCraftTag

Promyze

Practices | Information

Suggested practices (BETA) :

- Use specific types
- Set null default value for attributes in classes for c
- Don't use array as types

Search a practice by name or category :

- Promyze - Alway use a translation system
- Promyze - Avoid duplicating code
- Promyze - Avoid magic number and prefer constan
- Promyze - Centralize code mirror modes into a glob
- Promyze - Create a model specific to the database
- Promyze - Create simple selectors in selectors fold
- Promyze - Don't use array as types [TS]
- Promyze - Extract Component if main component is
- Promyze - Extract global helper methods [Clean Co
- Promyze - Extract method to keep readability [Clea
- Promyze - Extract reused services [Clean Code]
- Promyze - Extract shared interface in specific folde
- Promyze - File is specific or agnostic [Clean Code]
- Promyze - Files should only have one responsibility
- Promyze - HTTP calls to external services should h

Git | TODO | Problems | Terminal | Event Log

23:19 LF UTF-8 4 spaces* ø/up-to-date Blame: Magne 12/08/2021 09:13 TypeScript 4.2.4 core

Research Questions

Learning Coding Practices from Examples

Research Questions

Learning Coding Practices from Examples

RQ1: How many examples are required to learn a practice?

Research Questions

Learning Coding Practices from Examples

RQ1: How many examples are required to learn a practice?

RQ2: What are the best code examples to learn a practice?

MLinter
Design

MLinter
Design

Create a binary classifier for each rule

MLinter
Design

Create a binary classifier for each rule

Two input types to train:

MLinter
Design

Create a binary classifier for each rule

Two input types to train:

- Non-compliant code

```
if (age == '42')
```

MLinter
Design

Create a binary classifier for each rule

Two input types to train:

- Non-compliant code
- Compliant code

```
if (age == '42')
```

MLinter
Design

Create a binary classifier for each rule

Two input types to train:

- Non-compliant code
- Compliant code
  - Fixed code

```
if (age == '42')
```

```
if (age === '42')
```

MLinter
Design

Create a binary classifier for each rule

Two input types to train:

- Non-compliant code
- Compliant code
  - Fixed code
  - Extant code

```
if (age == '42')
```

```
if (age === '42')
```

```
spaceRepository.getAll()
```

MLinter
CodeBERT

MLinter
CodeBERT

Pretrained model on source code in 6 major languages

MLinter
CodeBERT

Pretrained model on source code in 6 major languages

Feature Extraction

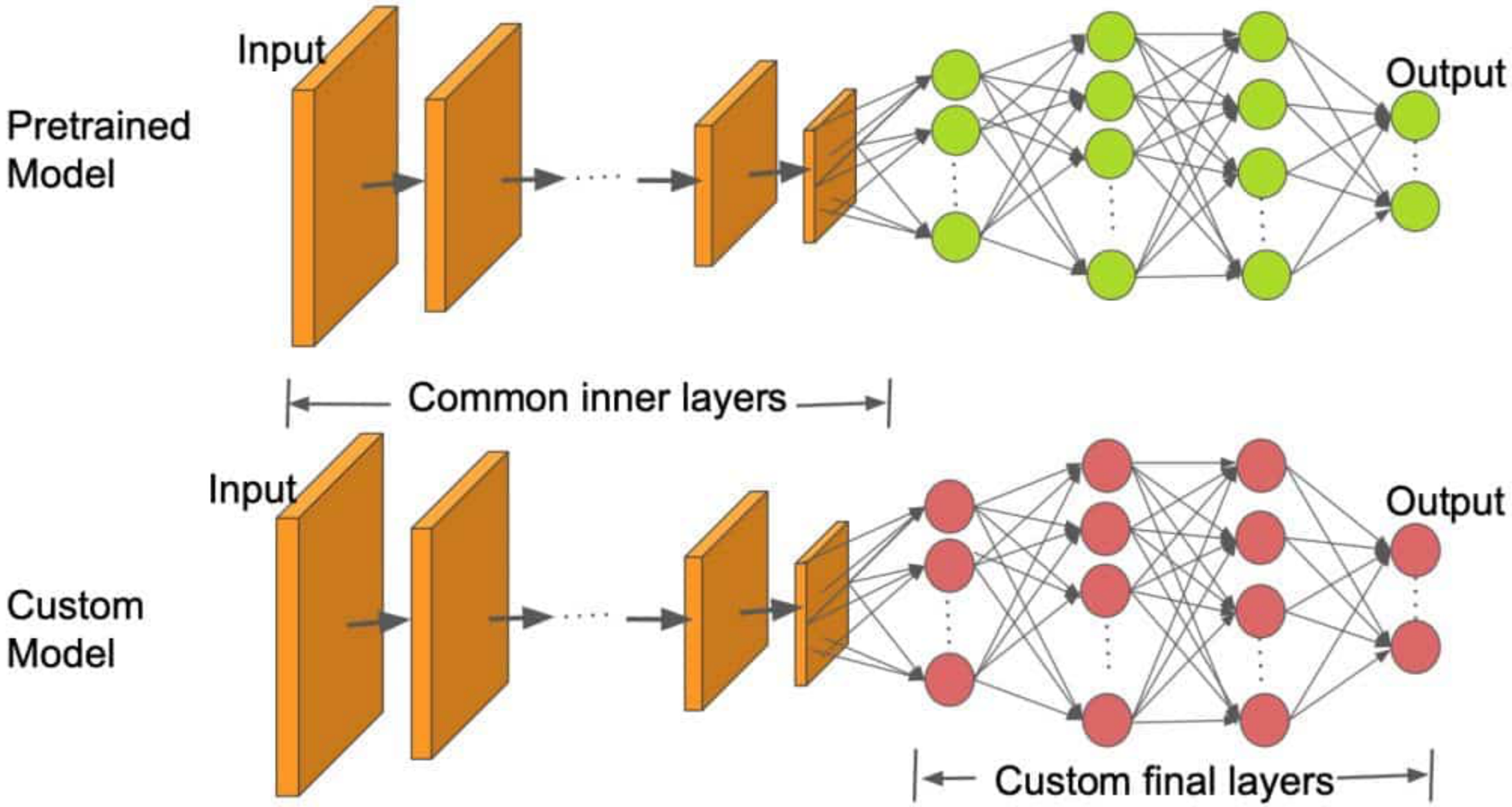
if (x is not None) and (x>1) | Compute

Computation time on cpu: 0.038 s

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
0	-0.035	0.255	-0.054	0.092	0.046	-0.244	0.002	0.088	0.152	-0.127	0.272	0.260	-0.164	-0.004	0.372	-0.125	0.060	0.228
1	0.242	0.105	0.111	0.163	1.507	-0.269	0.275	0.135	-0.138	0.240	-0.318	0.207	0.376	0.019	0.973	-0.504	0.239	0.248
2	-0.324	0.200	0.354	0.064	-0.597	-0.422	-0.737	0.363	0.772	0.742	-0.640	0.494	-0.400	-0.507	1.042	0.391	1.070	0.329
3	-0.165	0.469	0.673	0.057	-0.080	0.366	0.005	0.663	0.670	0.730	-0.002	0.196	-0.101	0.020	0.704	0.074	0.778	0.356

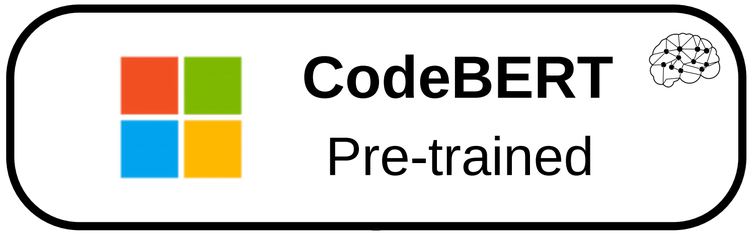
Base CodeBERT output
(https://huggingface.co/microsoft/codebert-base)

MLinter
Fine-tune CodeBERT on a new task



Transfer learning model
(https://learnopencv.com/image-classification-using-transfer-learning-in-pytorch/)

MLinter
Fine-tune CodeBERT on a new task



Code from practice ***P***

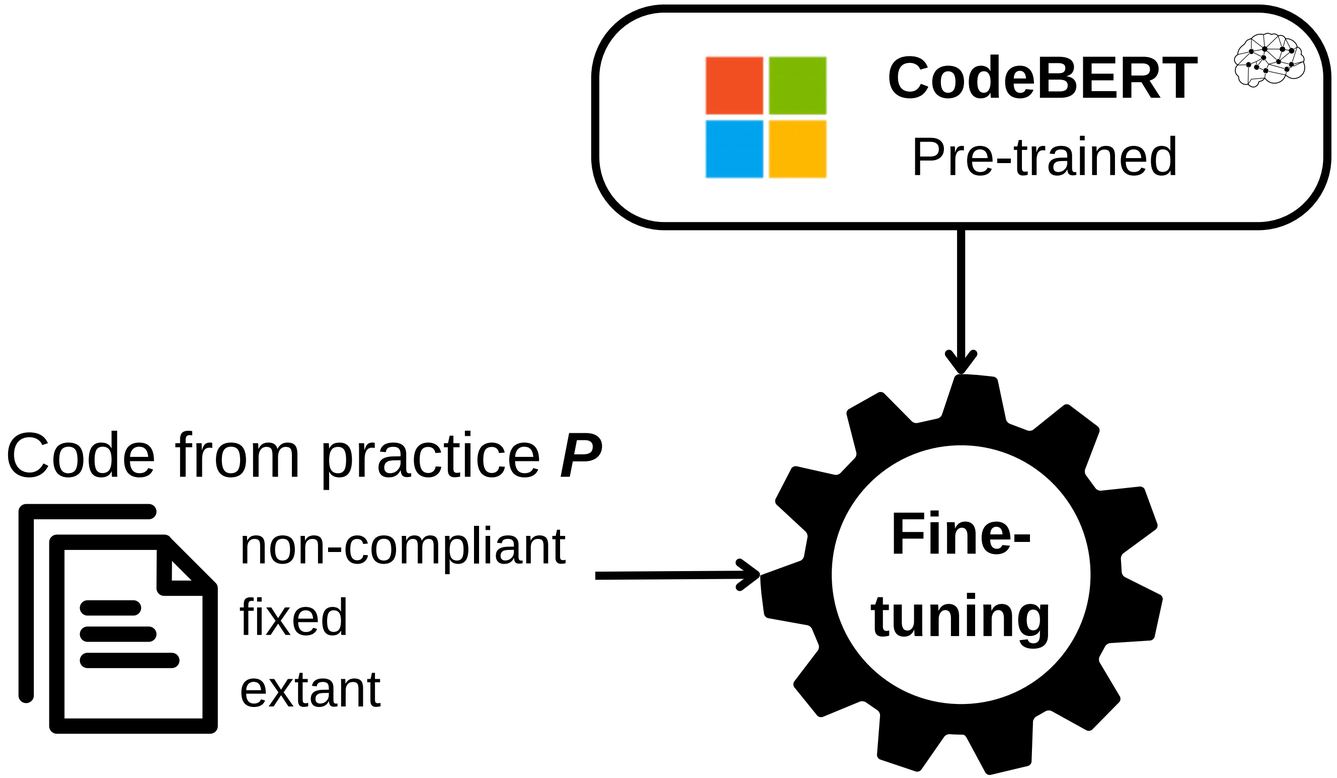
non-compliant

fixed

extant

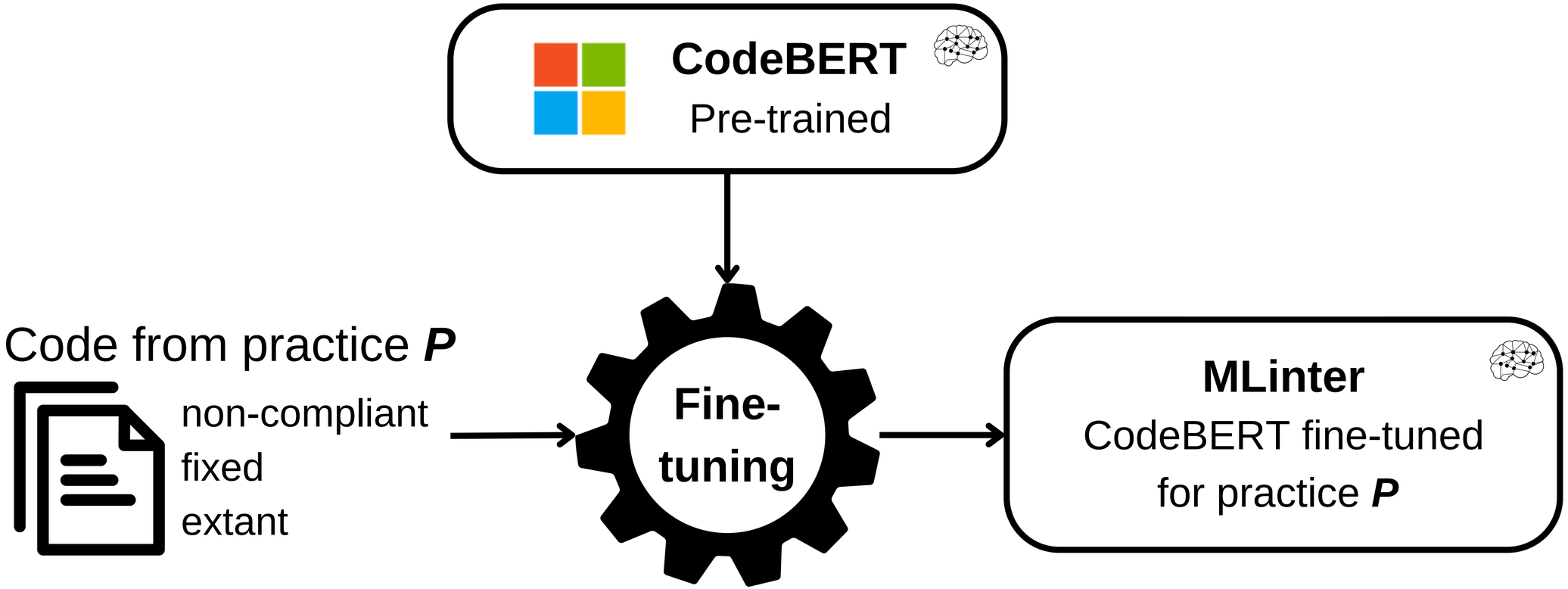
MLinter

Fine-tune CodeBERT on a new task



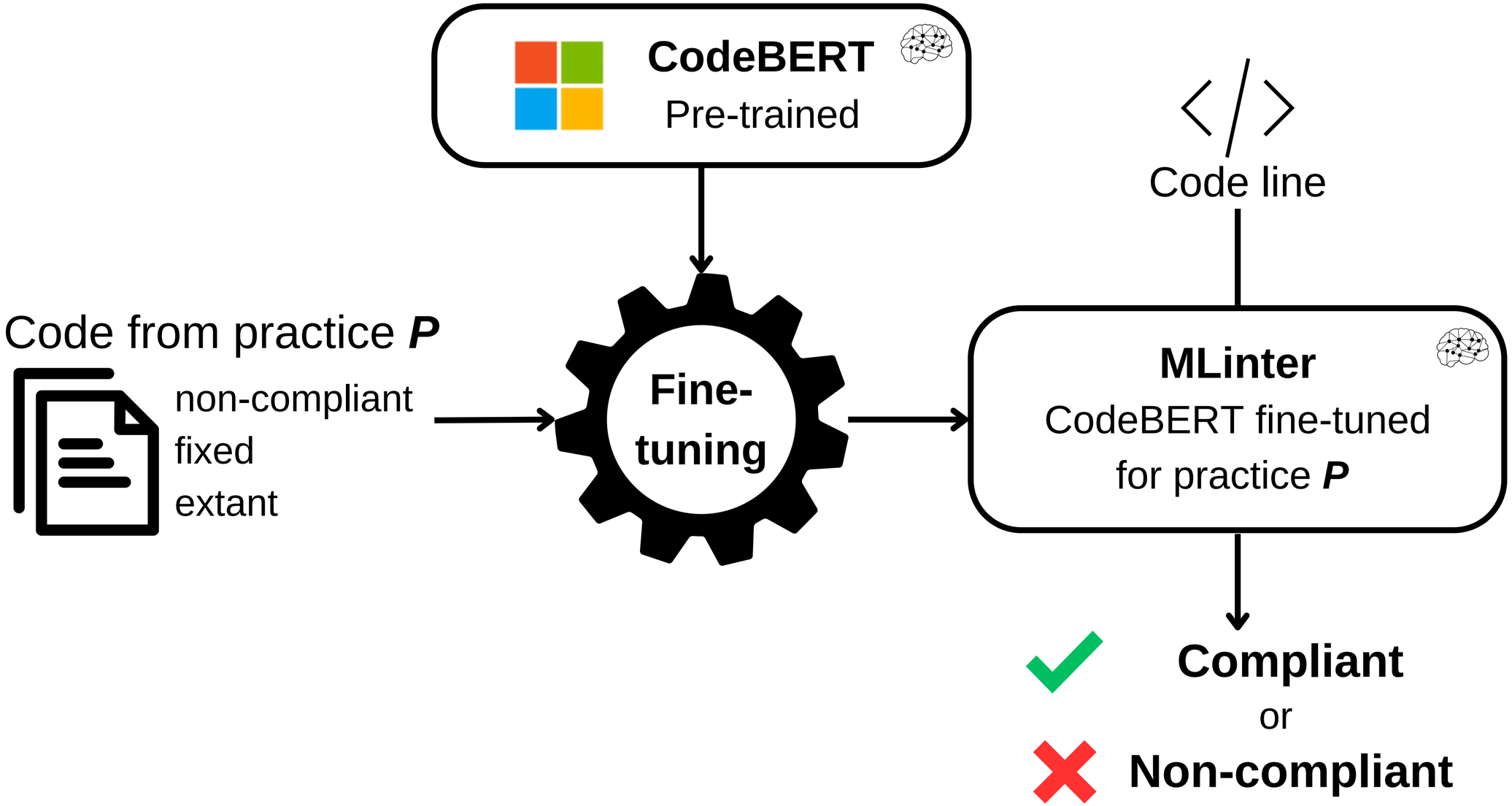
MLinter

Fine-tune CodeBERT on a new task

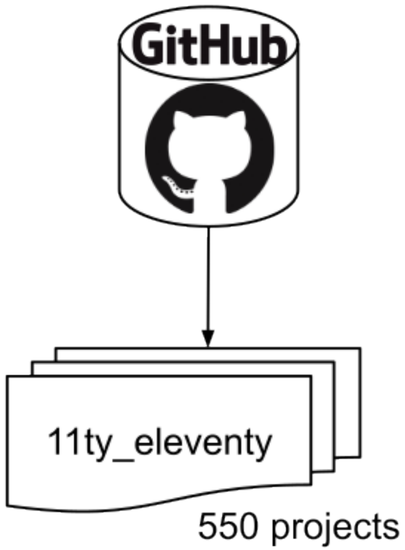


MLinter

Fine-tune CodeBERT on a new task



Dataset Creation



Use of the GitHub API to gather all public projects in JavaScript with more than 10K ⭐

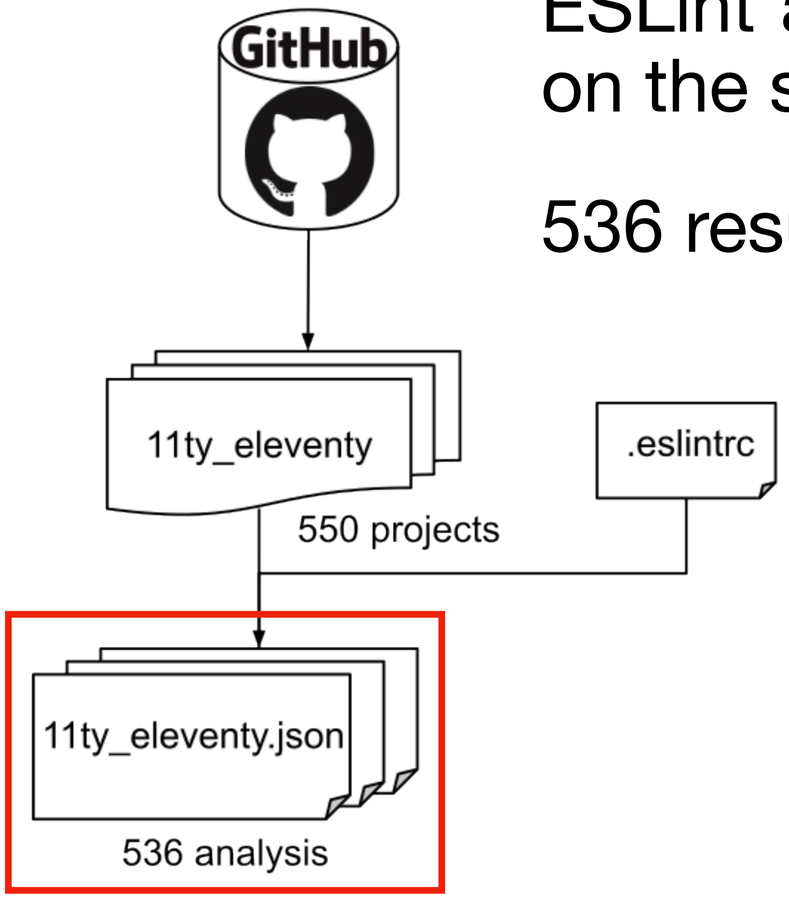
550 repositories cloned

Dataset Creation

Selection criterias to enable ESLint rules: auto-fixable and one liner

54 rules enabled

Dataset Creation



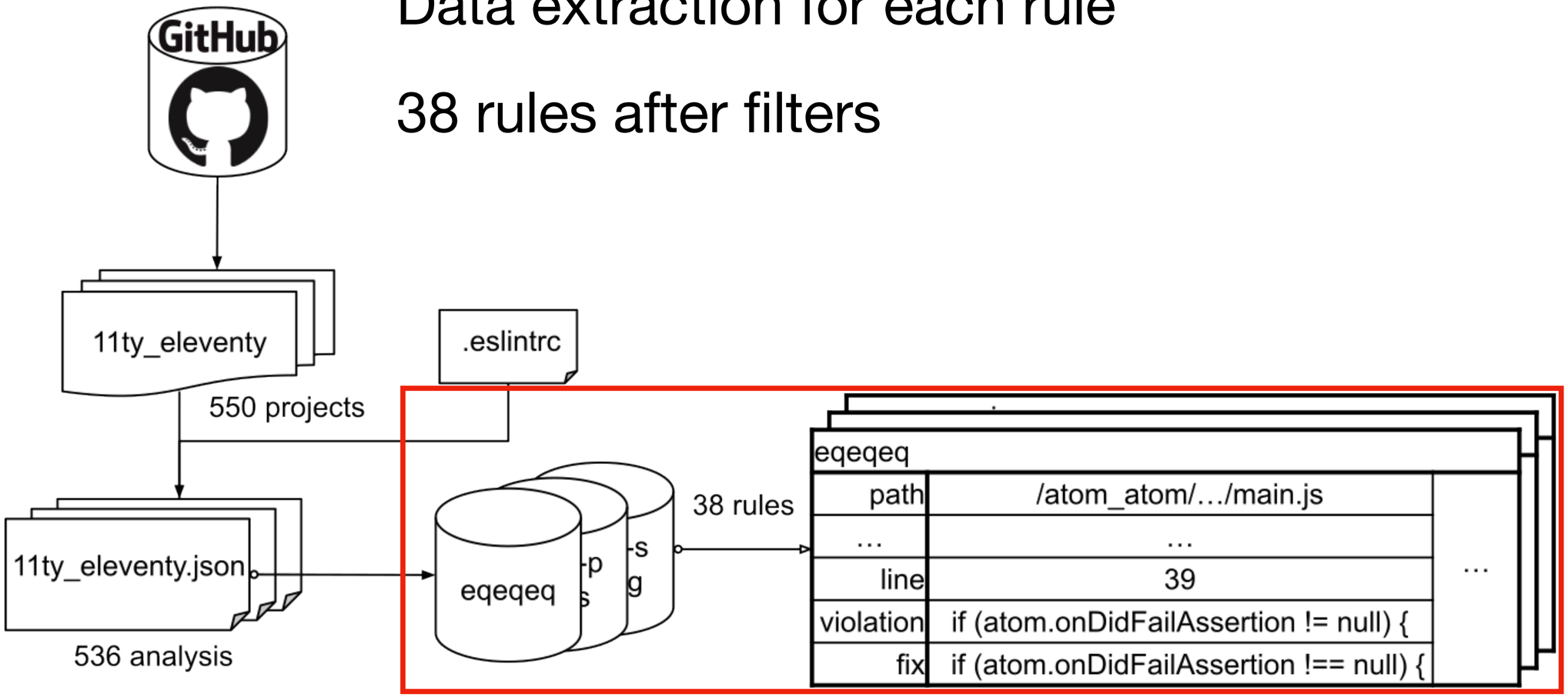
ESLint analysis with the specified configuration on the selected projects

536 result files

Dataset Creation

Data extraction for each rule

38 rules after filters



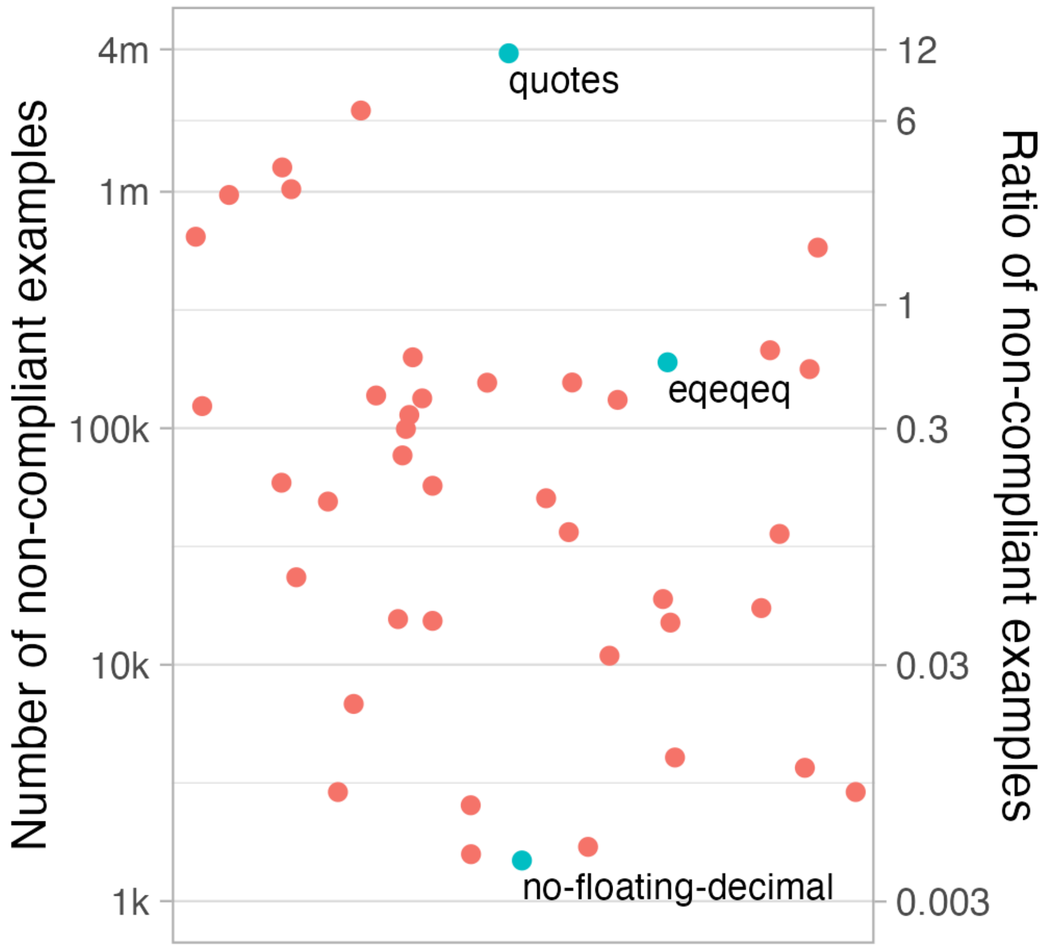
Dataset Creation

Dataset Creation

550 projects
218.530 files
+33M code lines
~13M violations for 38 rules

Dataset Creation

550 projects
218.530 files
+33M code lines
~13M violations for 38 rules



⚠️ Strong imbalance between rules

Experimental Protocol

Experimental Protocol

RQ1 How many examples are required to learn a practice?

Experimental Protocol

RQ1 How many examples are required to learn a practice?

3 Learning sizes

Experimental Protocol

RQ1 How many examples are required to learn a practice?

3 Learning sizes

Small	10
Medium	100
Large	1 000

Experimental Protocol

RQ2 What are the best code examples to learn a practice?

Experimental Protocol

RQ2 What are the best code examples to learn a practice?

Non-compliant (50%) **Compliant (50%)**

Experimental Protocol

RQ2 What are the best code examples to learn a practice?

Non-compliant (50%)	**Compliant (50%)**
Violation (50%)	

Experimental Protocol

RQ2 What are the best code examples to learn a practice?

<table>
<tr><th>Non-compliant (50%)</th><th>Compliant (50%)</th></tr>
<tr><td rowspan="3">Violation (50%)</td><td>Extant (50%)</td></tr>
<tr><td>Fixed (50%)</td></tr>
<tr><td>Fixed (25%)</td><td>Extant (25%)</td></tr>
</table>

Experimental Protocol

RQ2 What are the best code examples to learn a practice?

<table>
<tr><th></th><th colspan="2">Non-compliant (50%)</th><th colspan="2">Compliant (50%)</th></tr>
<tr><td>VE</td><td colspan="2" rowspan="3">Violation (50%)</td><td colspan="2">Extant (50%)</td></tr>
<tr><td>VF</td><td colspan="2">Fixed (50%)</td></tr>
<tr><td>VFE</td><td>Fixed (25%)</td><td>Extant (25%)</td></tr>
</table>

Experimental Protocol

	10 (S)	**100 (M)**	**1 000 (L)**
VE	S/VE	M/VE	L/VE
VF	S/VF	M/VF	L/VF
VFE	S/VFE	M/VFE	L/VFE

9 configurations for each practice

Experimental Protocol

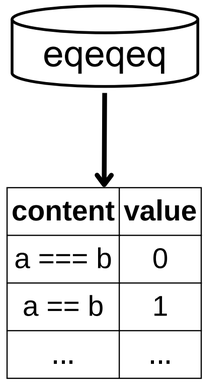
For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)

Experimental Protocol

For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)

Experimental Protocol

For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)

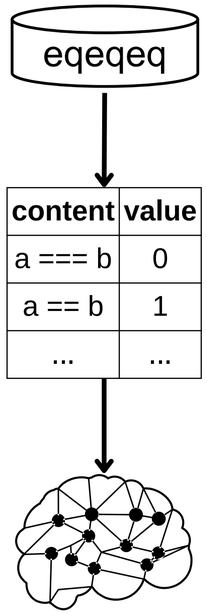


content	value
a === b	0
a == b	1
...	...

Creating a random learning set using dataset from ***P*** respecting ***C***

Experimental Protocol

For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)

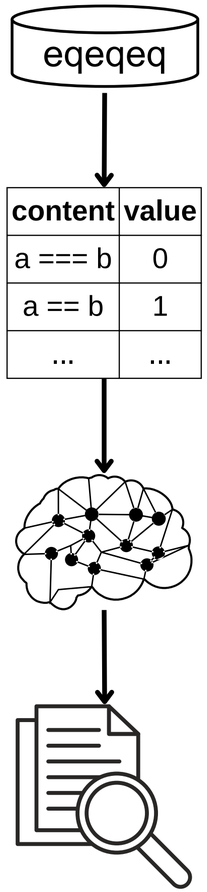


Creating a random learning set using dataset from ***P*** respecting ***C***

Fine-tune CodeBERT and obtain a new classifier

Experimental Protocol

For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)



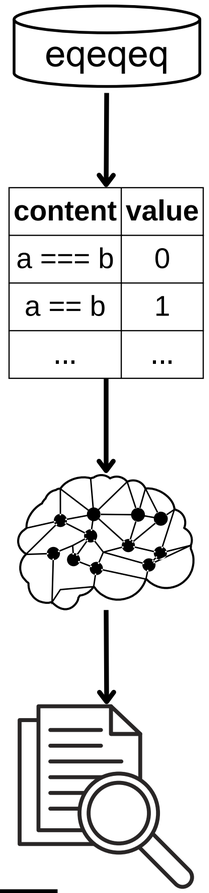
Creating a random learning set using dataset from ***P*** respecting ***C***

Fine-tune CodeBERT and obtain a new classifier

Evaluating classifier to measure Precision and Recall
using two methods: Balanced and Realistic

Experimental Protocol

For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)



Creating a random learning set using dataset from ***P*** respecting ***C***

Fine-tune CodeBERT and obtain a new classifier

Evaluating classifier to measure Precision and Recall using two methods: Balanced and Realistic

x 100

Experimental Protocol

Experimental Protocol

Balanced validation

Experimental Protocol

Balanced validation

Set with same size and balance as training

Experimental Protocol

Balanced validation

Set with same size and balance as training

Realistic validation

Experimental Protocol

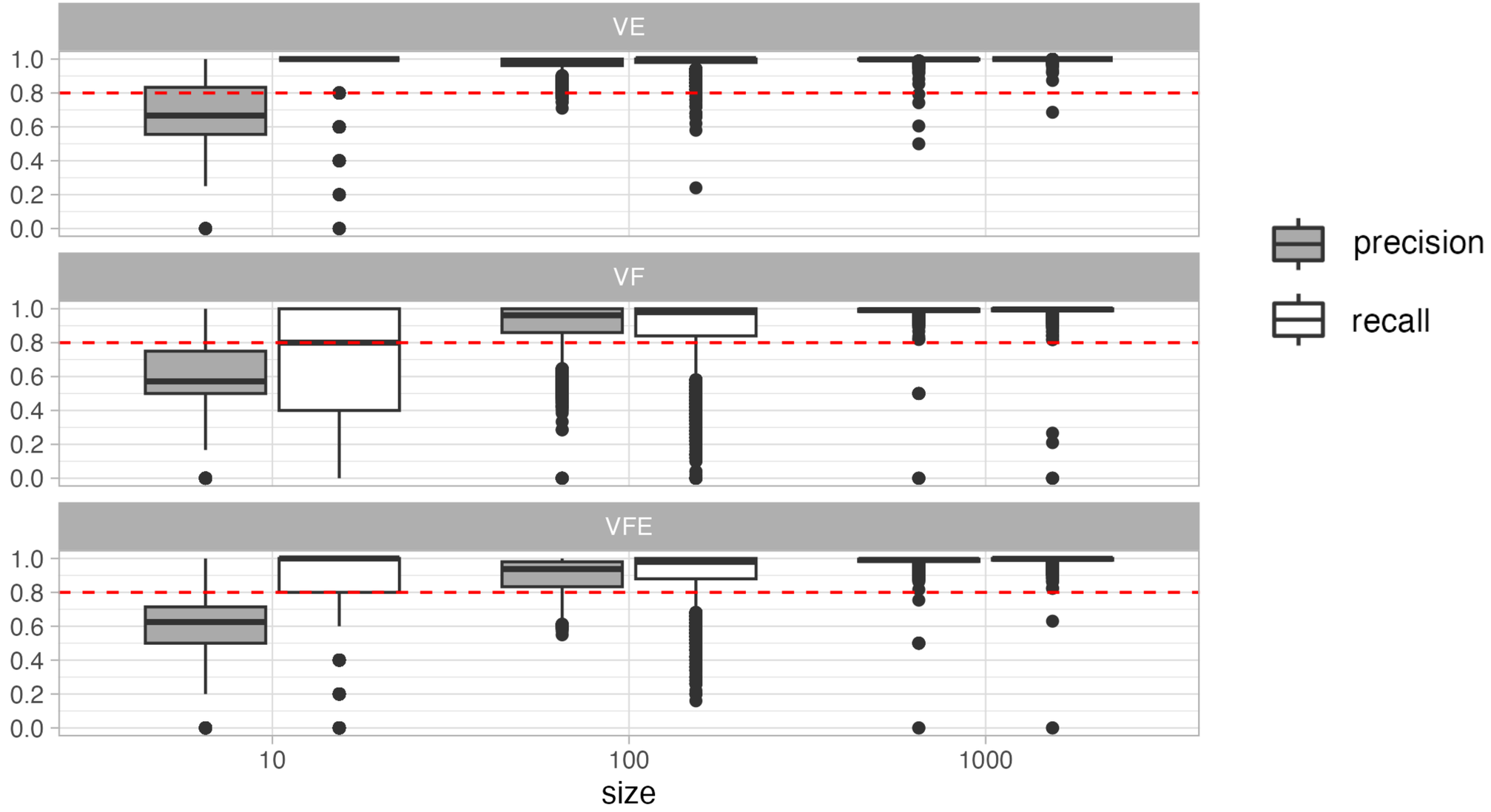
Balanced validation

Set with same size and balance as training

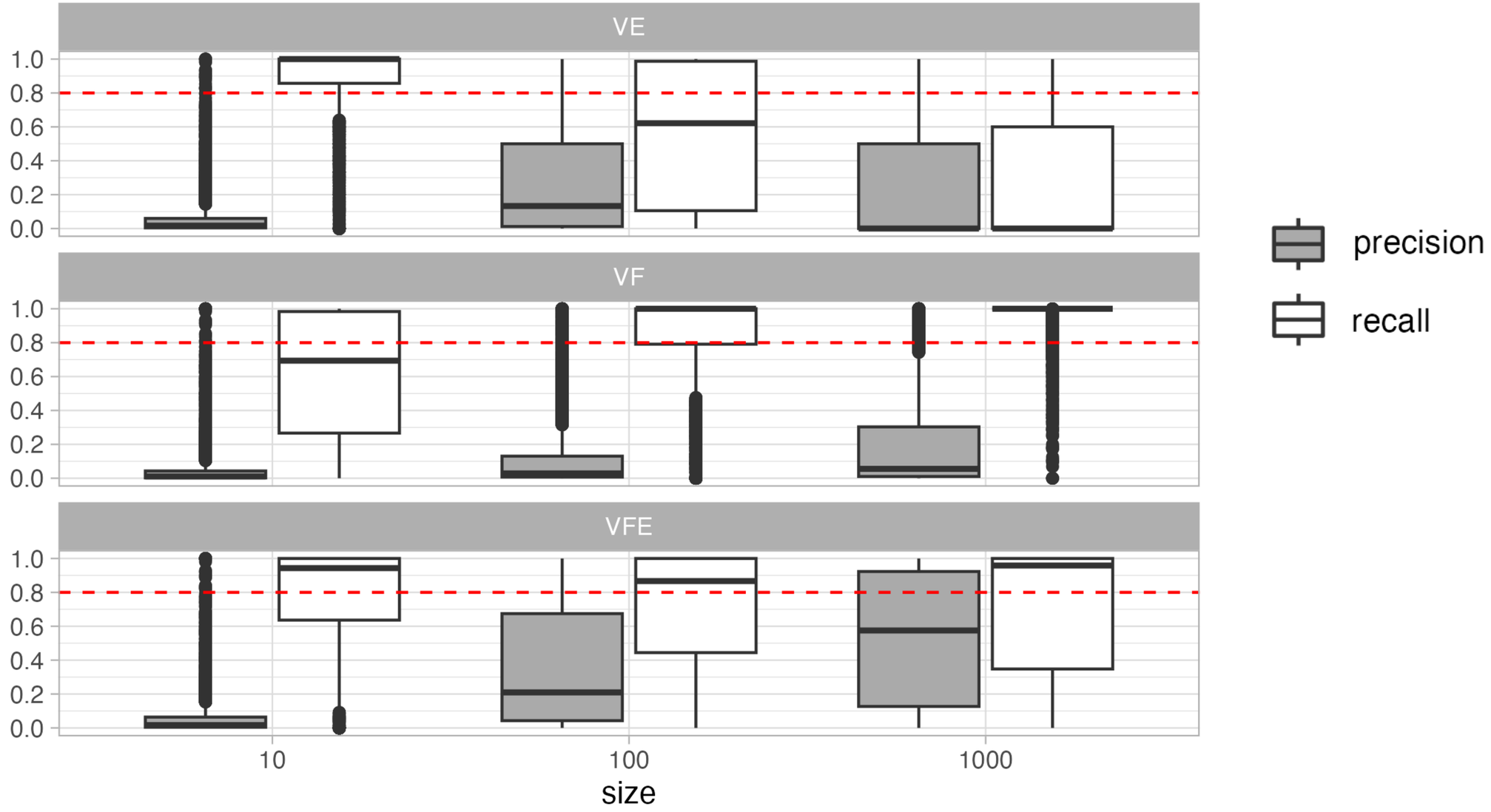
Realistic validation

Set with a balance similar to real source code files

Results
Balanced validation



Results
Realistic validation



Results
Conclusion

Results
Conclusion

RQ1:

Results
Conclusion

RQ1:

- Bigger training set sizes yield better results

Results
Conclusion

RQ1:

- Bigger training set sizes yield better results
- Size 10 clearly insufficient

Results
Conclusion

RQ1:

- Bigger training set sizes yield better results
- Size 10 clearly insufficient
- Size 100 similar to size 1 000

Results
Conclusion

RQ1:

- Bigger training set sizes yield better results
- Size 10 clearly insufficient
- Size 100 similar to size 1 000

RQ2:

Results
Conclusion

RQ1:

- Bigger training set sizes yield better results
- Size 10 clearly insufficient
- Size 100 similar to size 1 000

RQ2:

- No configuration emerges

Results
Conclusion

Results
Conclusion

CodeBERT can obtain good results on few examples

Results
Conclusion

CodeBERT can obtain good results on few examples

MLinter currently not usable in production

Results
Conclusion

CodeBERT can obtain good results on few examples

MLinter currently not usable in production

Alternative approach produces better outcomes

Results
Conclusion

CodeBERT can obtain good results on few examples

MLinter currently not usable in production

Alternative approach produces better outcomes

=> Replication study

Coding practices

eqeqeq — ESLint

Require the use of `===` and `!==`

It is considered good practice to use the type-safe equality operators `===` and `!==` instead of their regular counterparts `==` and `!=`.

Examples of **incorrect** code for this rule:

```
1  a == b
2  foo == true
```

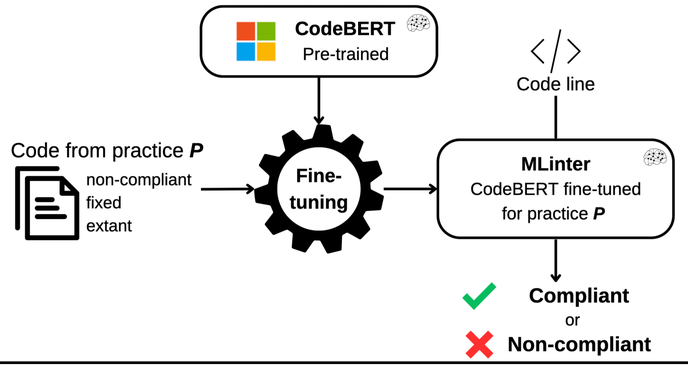
Examples of **correct** code for this rule:

```
1  a === b
2  foo === true
```

Rule « eqeqeq » from linter ESLint
(https://eslint.org/docs/latest/rules/eqeqeq)

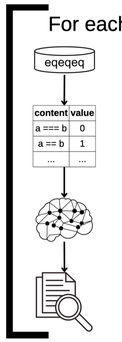
2

MLinter
Fine-tune CodeBERT on a new task



11

Experimental Protocol



For each practice ***P*** (eqeqeq) and each configuration ***C*** (M/VF)

Creating a random learning set using dataset from ***P*** respecting ***C***

Fine-tune CodeBERT and obtain a new classifier

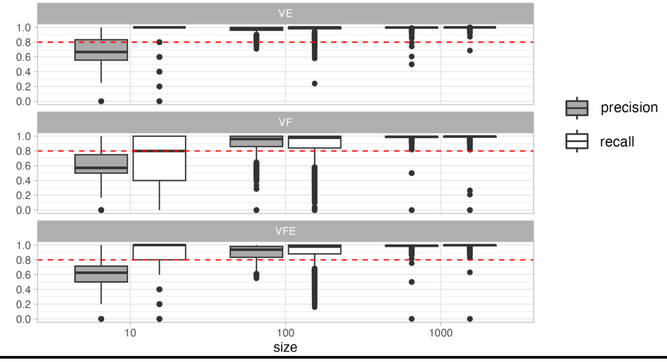
Evaluating classifier to measure Precision and Recall
using two methods: Balanced and Realistic

x 100

20

MLinter: Learning Coding Practices from Examples—Dream or Reality?

Results
Balanced validation



22

Results
Conclusion

CodeBERT can obtain good results on few examples

MLinter currently not usable in production

Alternative approach produces better outcomes

=> Replication study

25

Orchid Room

Corentin LATAPPY ✉ **corentin.latappy@labri.fr**

Thursday, March 23, 2023

LaBRI — IMT Mines Alès, École Mines-Télécom — promyze