Facilitating Software Evolution through Natural Language Comments and Dialogue

Sheena Panthaplackel

Dissertation Proposal

Committee Members: Ray Mooney, Jessy Li, Milos Gligoric, Greg Durrett, Charles Sutton

Software is Constantly Evolving

Developers regularly

- Incorporate new functionality
- Improve existing functionality
- Refactor the code base

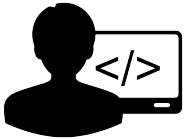
elastic / elasticsearch Public → **60K+ commits**
1.6K contributors

Developers may unintentionally introduce vulnerabilities, and these should be identified
Goal #1: Uphold software quality amidst constant changes

Critical changes should be prioritized
Goal #2: Facilitate prompt implementation of critical changes

Can we guide developers in making more *methodical changes* through natural language?

Software is Constantly Evolving



Developers use natural language in various ways...

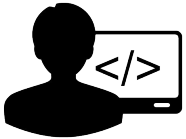
Comments for documenting code

Commit ***messages***

Dialogue for reporting and discussing issues

Queries for search

Software is Constantly Evolving



Developers use natural language in various ways...

Comments for documenting code

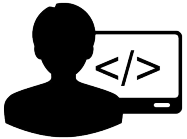
Commit ***messages***

Dialogue for reporting and discussing issues

Queries for search

Goal #1: Uphold software quality amidst constant changes

Software is Constantly Evolving



Developers use natural language in various ways...

Comments for documenting code

Commit ***messages***

Dialogue for reporting and discussing issues

Queries for search

Goal #1: Uphold software quality amidst constant changes

Goal #2: Facilitate prompt implementation of critical changes

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Source Code Comments

Document functionality, usage, implementation, error cases, ...

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
    return Collections.max(scores);

}
```

Source Code Comments

When developers make code changes, they often fail to update comments accordingly.

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
-    return Collections.max(scores);
+    return Collections.min(scores);
}
```

Source Code Comments

When developers make code changes, they often fail to update comments accordingly.

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
-    return Collections.max(scores);
+    return Collections.min(scores);
}
```

Leads to time-wasting confusion and vulnerability to bugs

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Inconsistency Detection

Post Hoc Inconsistency Detection

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
        return Collections.min(scores);
}
```

✗ **Inconsistent**

```
/** Computes the highest value from the list of scores */
public double getBestScore() {
        return Collections.max(scores);
}
```

✓ **Consistent**

Inconsistency Detection

Just-In-Time Inconsistency Detection

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
-        return Collections.max(scores);
+        return Collections.min(scores);
}
```

```
/** Computes the highest value from the list of scores */
- public int getBestScore() {
+ public double getBestScore() {
         return Collections.max(scores);
}
```

Post Hoc Inconsistency Detection

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
         return Collections.min(scores);
}
```

✗ **Inconsistent**

```
/** Computes the highest value from the list of scores */
public double getBestScore() {
         return Collections.max(scores);
}
```

✓ **Consistent**

Prior work

Rule-based approaches constrained to specific domains/templates and traditional ML approaches

We studied

Detecting inconsistency upon code changes

Inconsistency Detection

Just-In-Time Inconsistency Detection

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
-        return Collections.max(scores);
+        return Collections.min(scores);
}
```

```
/** Computes the highest value from the list of scores */
- public int getBestScore() {
+ public double getBestScore() {
         return Collections.max(scores);
}
```

Post Hoc Inconsistency Detection

```
/** Computes the highest value from the list of scores */
public int getBestScore() {
         return Collections.min(scores);
}
```

✗ **Inconsistent**

```
/** Computes the highest value from the list of scores */
public double getBestScore() {
         return Collections.max(scores);
}
```

✓ **Consistent**

Prior work

Rule-based approaches constrained to specific domains/templates and traditional ML approaches

We studied

Detecting inconsistency upon code changes through a general framework which encodes the syntactic structure of code/comments with a deep neural network.

Task

C_{old}

```
/** Computes the highest value from the list of scores
*/
```

M_{old}

```
public int getBestScore() {
        return
Collections.max(scores);
}
```

M_{new}

```
public int getBestScore() {
        return
Collections.min(scores);
}
```

<u>Problem Setting:</u>

Suppose M_{new} is merged into the code base (with no comment changes).

Task

C_{old}

```
/** Computes the highest value from the list of scores
*/
```

M_{old}

```
public int getBestScore() {
        return
Collections.max(scores);
}
```

➡

M_{new}

```
public int getBestScore() {
        return
Collections.min(scores);
}
```

<u>Problem Setting:</u>

Suppose M_{new} is merged into the code base (with no comment changes).

Determine whether the comment (C_{old}) becomes inconsistent with the corresponding code (M_{new}).

Task

C_{old}

```
/** Computes the highest value from the list of scores
*/
```

M_{old}

```
public int getBestScore() {
        return
Collections.max(scores);
}
```

➡

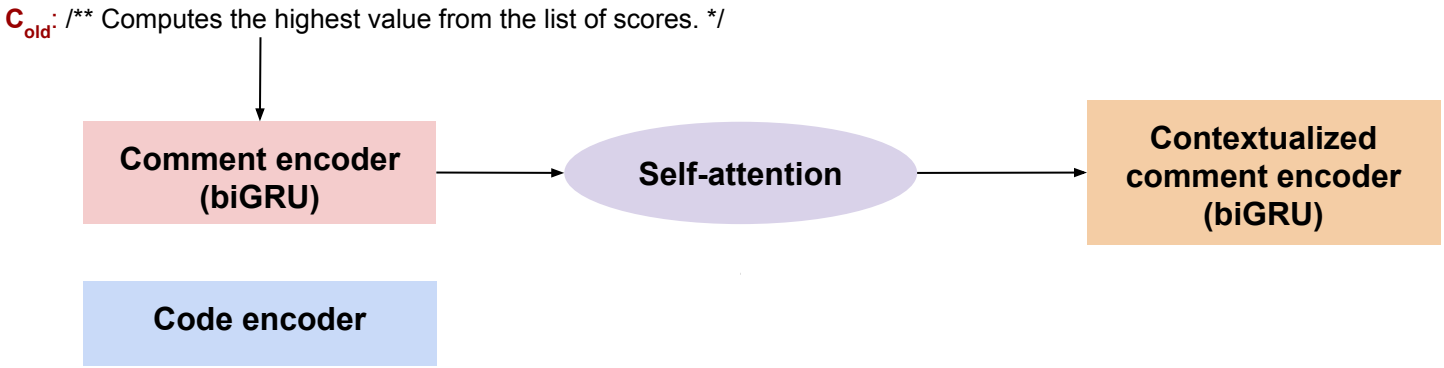
M_{new}

```
public int getBestScore() {
        return
Collections.min(scores);
}
```

<u>Problem Setting:</u>
Suppose M_{new} is merged into the code base (with no comment changes).
Determine whether the comment (C_{old}) becomes inconsistent with the corresponding code (M_{new}).

<u>Post Hoc:</u> Given C_{old} and M_{new}
<u>Just-In-Time:</u> Given C_{old}, M_{new}, and M_{old}

Architecture



GRU *[Cho et al. 2014]*; ***Multi-head attention*** *[Vaswani et al. 2017]*

Code Representations

Sequence-based

Post Hoc (M_{new}): M_{new} as a sequence of tokens

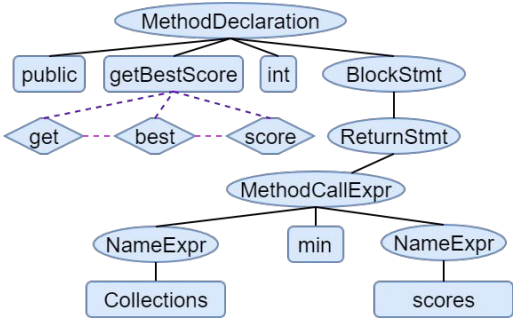
```
public int getBestScore ( ) { return Collections . min ( scores ) ; }
```

Just-In-Time (M_{edit}): Edits between M_{old} and M_{new} as a sequence of tokens

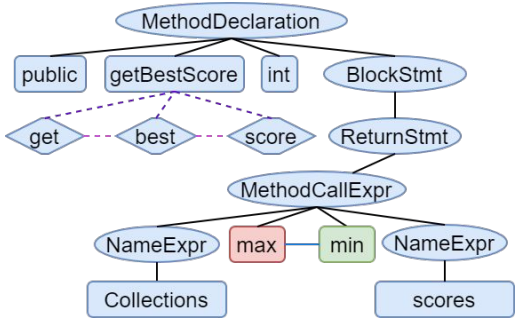
```
<Keep> public int getBestScore(){ return Collections. <KeepEnd>
<ReplaceOld> max <ReplaceNew> min <ReplaceEnd><Keep> (scores);} <KeepEnd>
```

AST-based

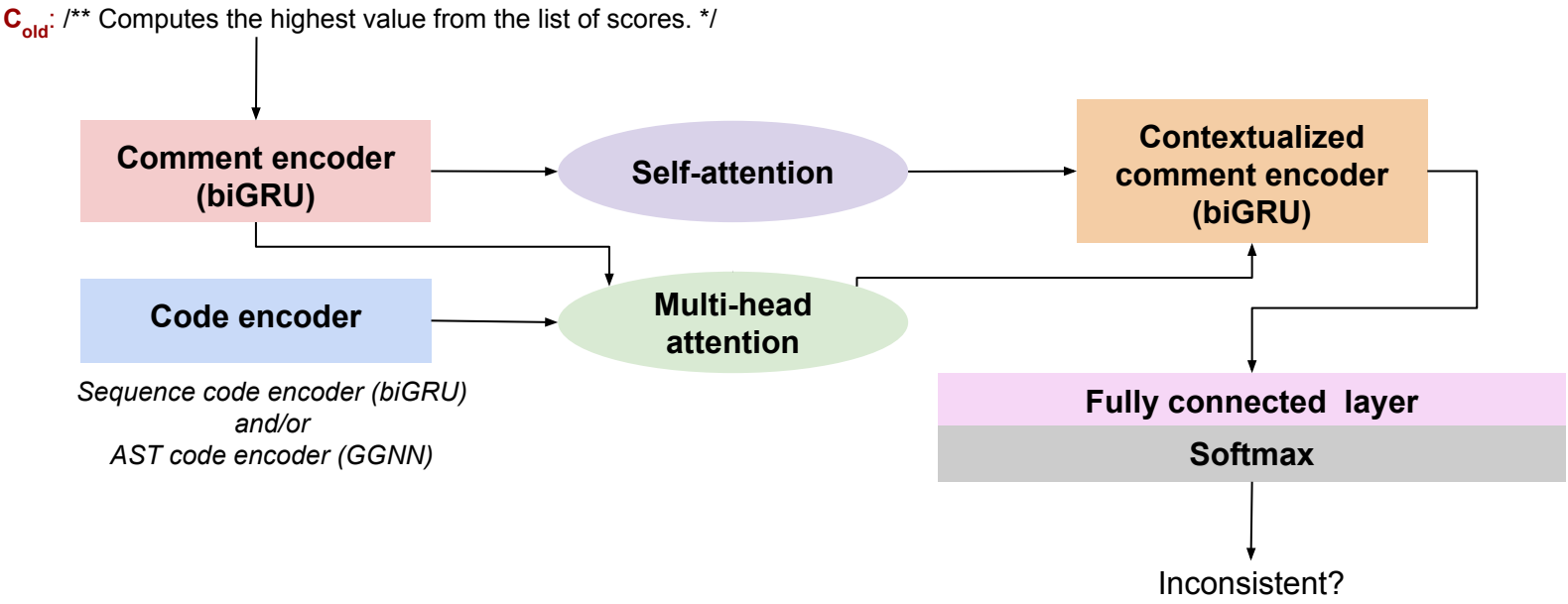
Post Hoc (T_{new}): Graph representation of AST nodes in M_{new}



Just-In-Time (T_{edit}): Graph representation of AST node edits between M_{old} and M_{new}

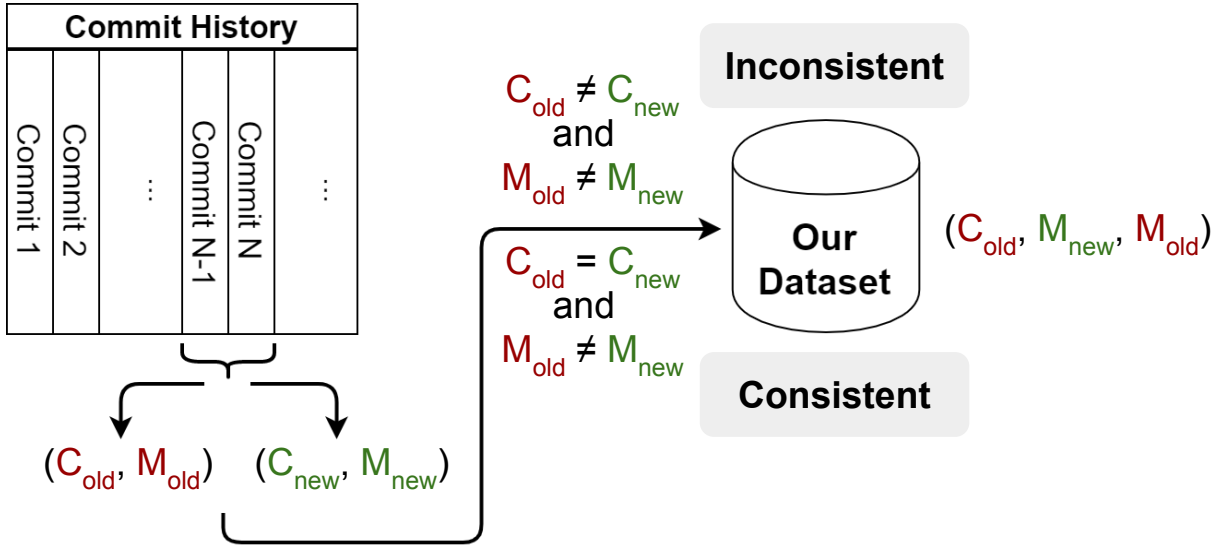


Architecture



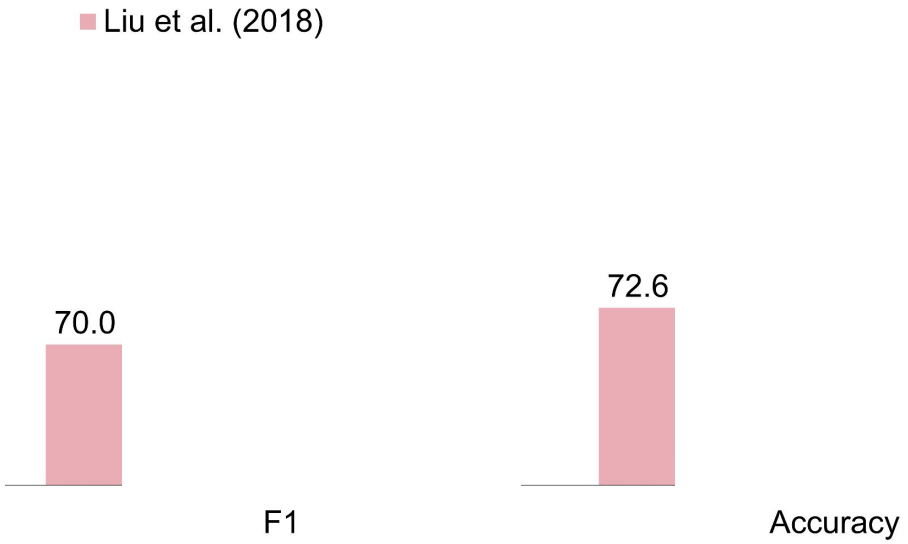
GRU *[Cho et al. 2014]*; ***Multi-head attention*** *[Vaswani et al. 2017]*; ***GGNN*** *[Li et al. 2016]*

Data Collection

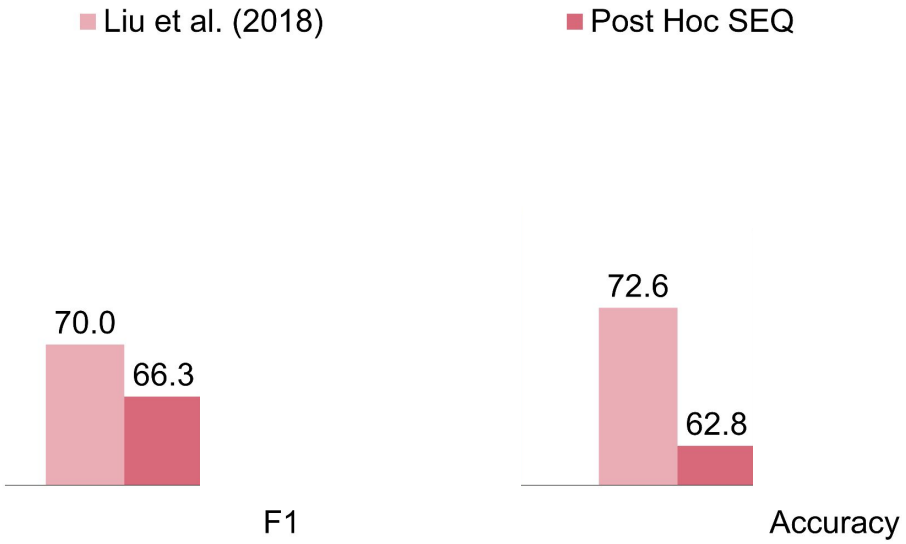


Balanced dataset with ~41K examples from ~1.5K projects

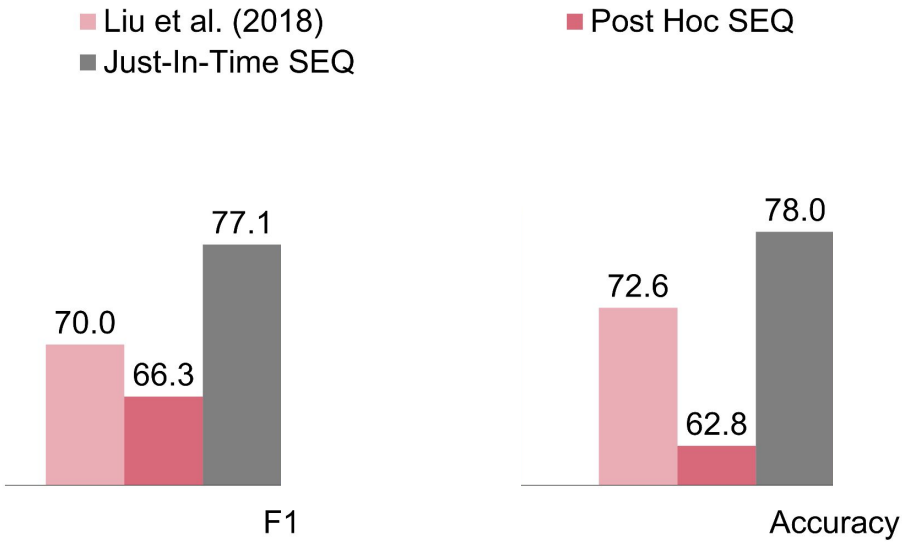
Results



Results

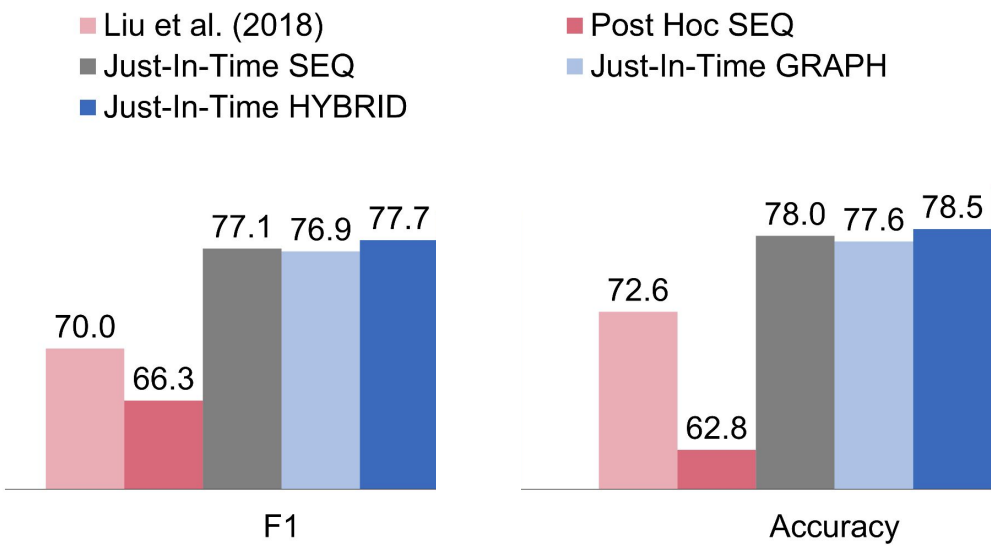


Results



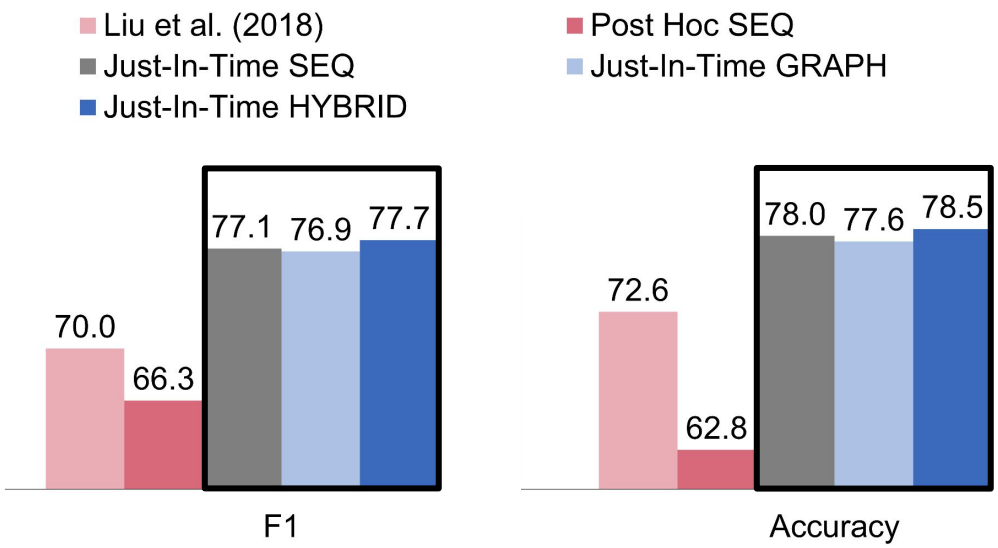
- Our Just-In-Time approach can outperform post hoc and baseline models

Results



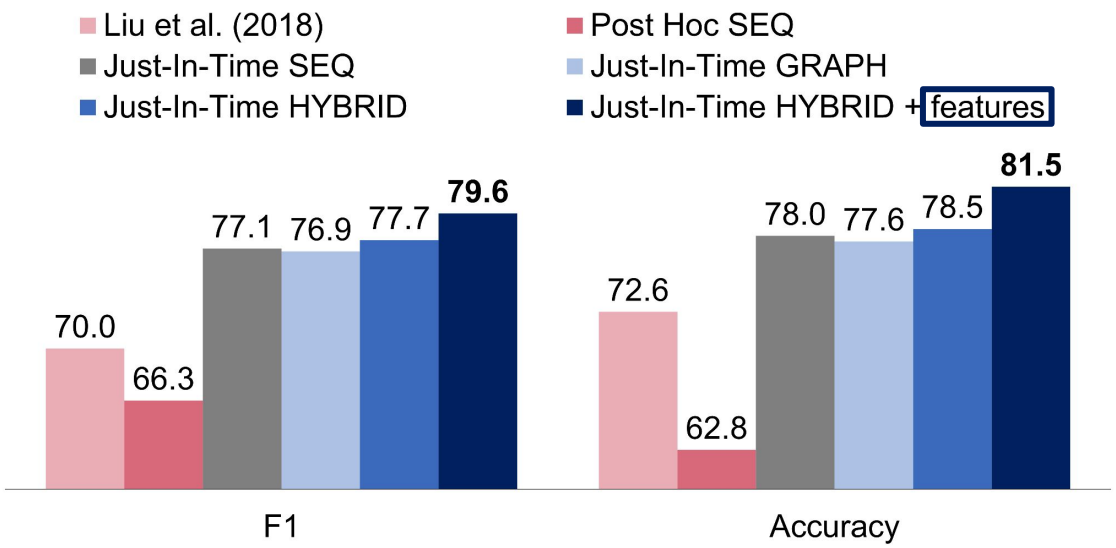
- Our Just-In-Time approach can outperform post hoc and baseline models

Results



- Our Just-In-Time approach can outperform post hoc and baseline models
- No significant difference between SEQ, GRAPH, and HYBRID approaches

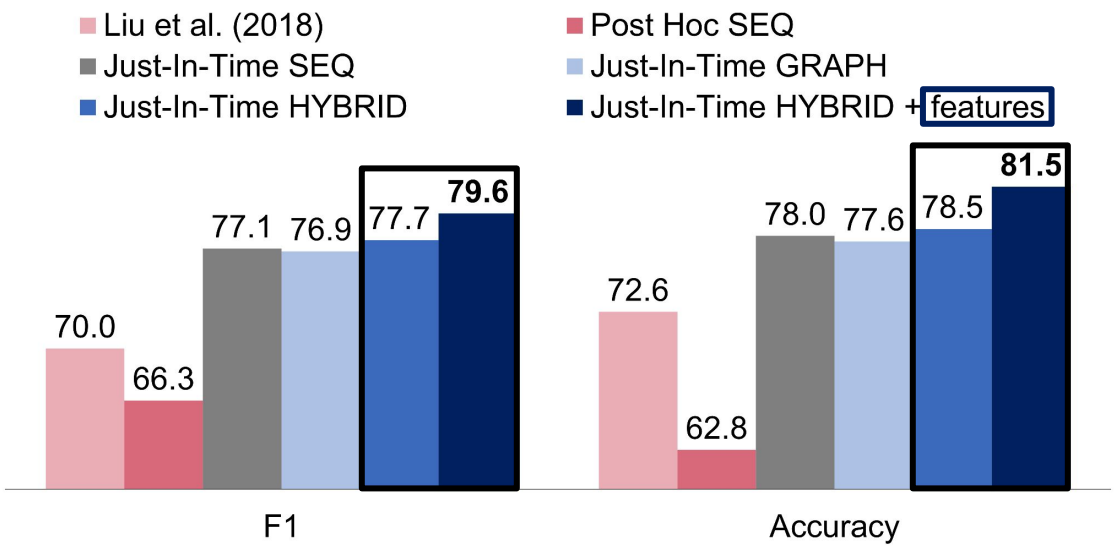
Results



e.g., lexical overlap, is Java keyword *[**Associating Natural Language Comment and Source Code Entities**; Panthaplackel et al. AAAI 2020]*

- Our Just-In-Time approach can outperform post hoc and baseline models
- No significant difference between SEQ, GRAPH, and HYBRID approaches

Results



e.g., lexical overlap, is Java keyword *[**Associating Natural Language Comment and Source Code Entities**; Panthaplackel et al. AAAI 2020]*

- Our Just-In-Time approach can outperform post hoc and baseline models
- No significant difference between SEQ, GRAPH, and HYBRID approaches
- Incorporating auxiliary features can further boost performance

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Updating Inconsistent Comments

C_{old}

```
/** Computes the highest value from the list of scores
*/
```

M_{old}

```
public int getBestScore() {
        return
Collections.max(scores);
}
```

M_{new}

```
public int getBestScore() {
        return
Collections.min(scores);
}
```

<u>Problem Setting:</u>
Suppose inconsistency is detected upon code changes (i.e., C_{old} is inconsistent with M_{new}).

Updating Inconsistent Comments

C_{old}

```
/** Computes the highest value from the list of scores
*/
```

M_{old}

```
public int getBestScore() {
        return
Collections.max(scores);
}
```

C_{new}

```
/** Computes the lowest value from the list of scores */
```

M_{new}

```
public int getBestScore() {
        return
Collections.min(scores);
}
```

<u>Problem Setting:</u>
Suppose inconsistency is detected upon code changes (i.e., C_{old} is inconsistent with M_{new}).

Updating Inconsistent Comments

C_{old}

```
/** Computes the highest value from the list of scores
*/
```

M_{old}

```
public int getBestScore() {
        return
Collections.max(scores);
}
```

➡

C_{new}

```
/** Computes the lowest value from the list of scores */
```

M_{new}

```
public int getBestScore() {
        return
Collections.min(scores);
}
```

<u>Problem Setting:</u>
Suppose inconsistency is detected upon code changes (i.e., C_{old} is inconsistent with M_{new}).

Automatically produce an updated comment (C_{new}) that is consistent with the new version of the code (M_{new}).

Code Summarization/Comment Generation

Given a body of code (M_{new}), generate a NL summary/comment (C_{new})

StackOverflow answer code snippet to question title [1]

Method body to method name [2]

Methods/classes to comments [3]

[1] *Iyer et al. 2016, Yao et al. 2018, Yin et al. 2018*
[2] *Allamanis et al. 2016, Xu et al. 2019, Alon et al. 2019, Fernandes et al. 2019*
[3] *Sridhara et al. 2011, Movshovitz-Attias and Cohen 2013, Hu et al. 2018, Liang and Zhu 2018, LeClair et al. 2019, Fernandes et al. 2019, Ahmad et al. 2020, Yu et al. 2020*

Code Summarization/Comment Generation

Given a body of code (M_{new}), generate a NL summary/comment (C_{new})

StackOverflow answer code snippet to question title [1]

Method body to method name [2]

Methods/classes to comments [3]

Ignores rich context from C_{old} and code changes between M_{old} and M_{new}

[1] *Iyer et al. 2016, Yao et al. 2018, Yin et al. 2018*
[2] *Allamanis et al. 2016, Xu et al. 2019, Alon et al. 2019, Fernandes et al. 2019*
[3] *Sridhara et al. 2011, Movshovitz-Attias and Cohen 2013, Hu et al. 2018, Liang and Zhu 2018, LeClair et al. 2019, Fernandes et al. 2019, Ahmad et al. 2020, Yu et al. 2020*

Code Summarization/Comment Generation

Given a body of code (M_{new}), generate a NL summary/comment (C_{new})

StackOverflow answer code snippet to question title [1]

Method body to method name [2]

Methods/classes to comments [3]

Ignores rich context from C_{old} and code changes between M_{old} and M_{new}
Deviates from how developers update comments

We studied

Learning to <u>edit</u> $C_{old} \rightarrow C_{new}$ rather than <u>generate</u> C_{new} from scratch.

[1] *Iyer et al. 2016, Yao et al. 2018, Yin et al. 2018*
[2] *Allamanis et al. 2016, Xu et al. 2019, Alon et al. 2019, Fernandes et al. 2019*
[3] *Sridhara et al. 2011, Movshovitz-Attias and Cohen 2013, Hu et al. 2018, Liang and Zhu 2018, LeClair et al. 2019, Fernandes et al. 2019, Ahmad et al. 2020, Yu et al. 2020*

Code Edits → NL Edits

M_{old}

```
public int getBestScore() {
      return Collections.max(scores);
}
```

M_{new}

```
public int getBestScore() {
      return Collections.min(scores);
}
```

M_{edit}: Code edits between M_{old} and M_{new}

```
<Keep> public int getBestScore(){ return Collections. <KeepEnd>
<ReplaceOld> max
<ReplaceNew> min
<ReplaceEnd>
<Keep> (scores);} <KeepEnd>
```

C_{old}

```
/** Computes the highest value from the list of scores */
```

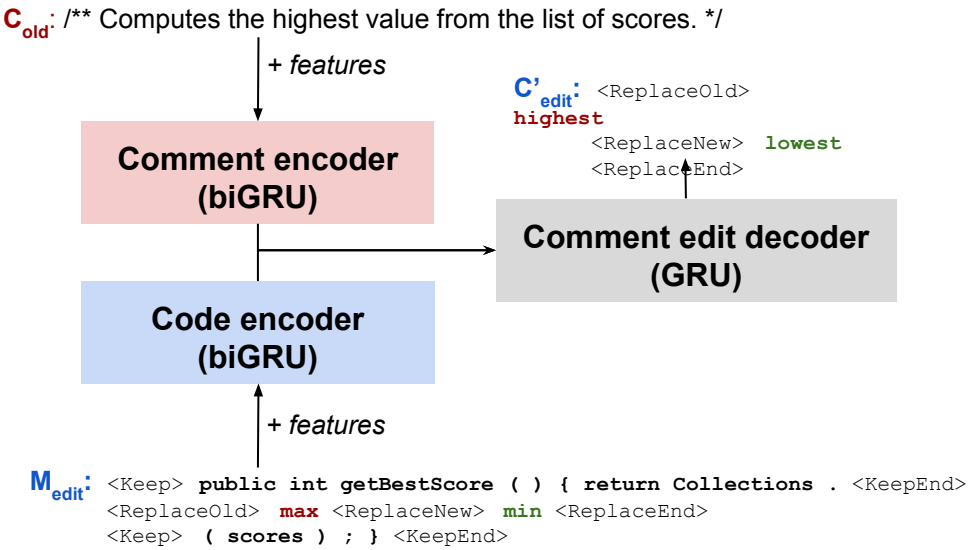
C_{new}

```
/** Computes the lowest value from the list of scores */
```

C_{edit}: NL edits between C_{old} and C_{new}

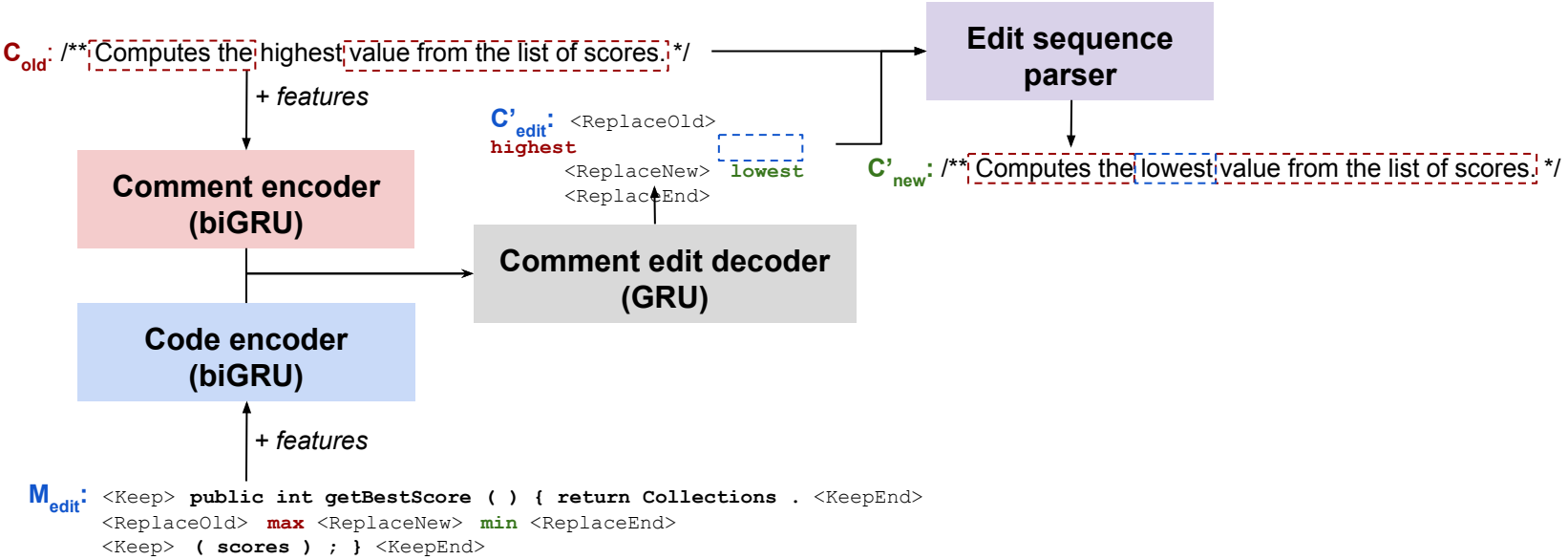
```
<ReplaceOld> highest
<ReplaceNew> lowest
<ReplaceEnd>
```

Edit Model



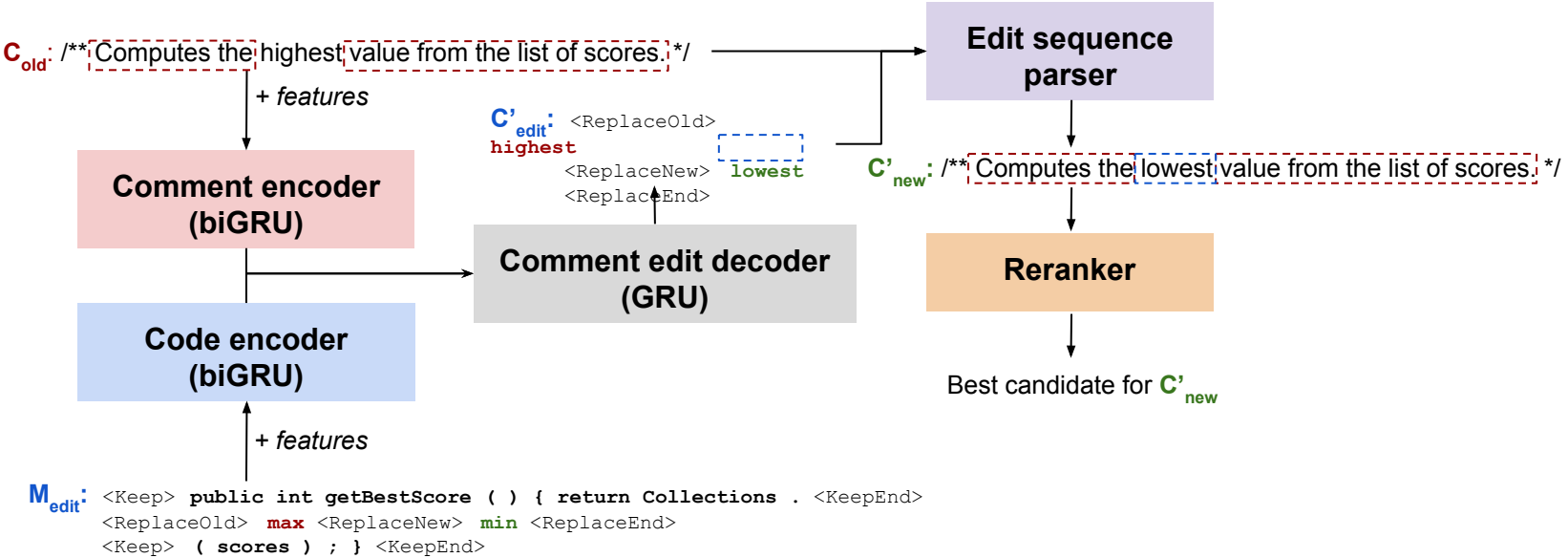
Inference
for C'_{edit} in Beam search candidates:

Edit Model



Inference
for C'_{edit} in Beam search candidates:
C'_{new} = Edit sequence parser (C'_{edit})

Edit Model



Inference

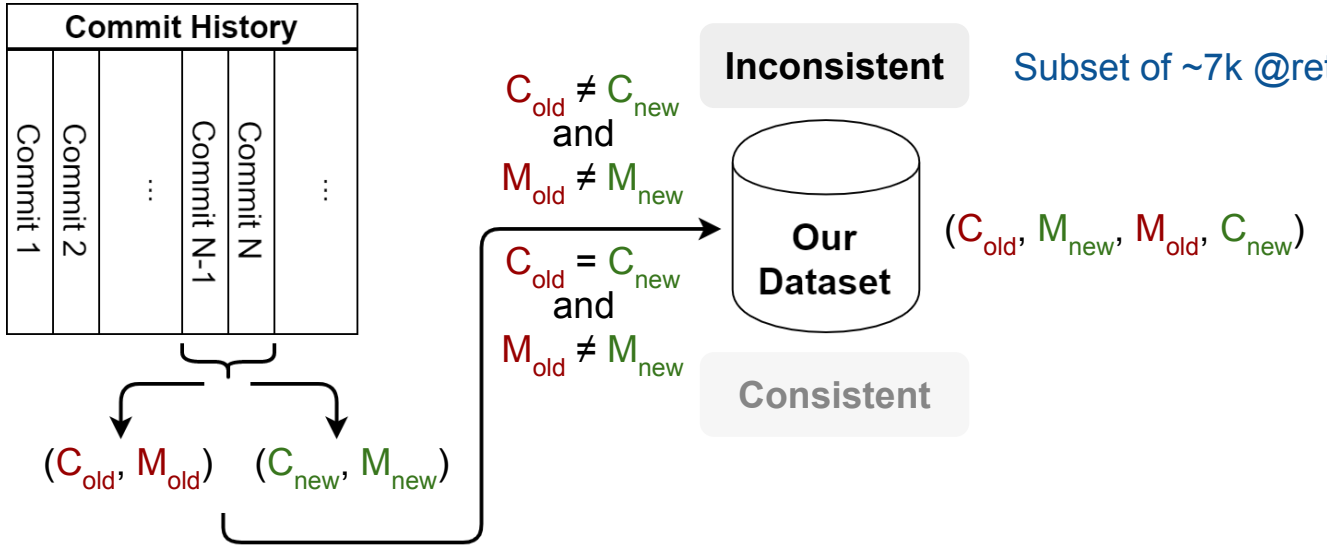
for C'_{edit} in Beam search candidates:

C'_{new} = Edit sequence parser (C'_{edit})

$\text{Rerank}(C'_{new}) = \lambda_1 \text{Beam}(C'_{edit}) + \lambda_2 \text{METEOR}(C_{old}, C'_{new}) + \lambda_3 P(C'_{new} | M_{new})$

METEOR *[Banerjee and Lavie 2005]*

Data Collection



Subset of ~7k @return examples

Balanced dataset with ~41K examples from ~1.5K projects

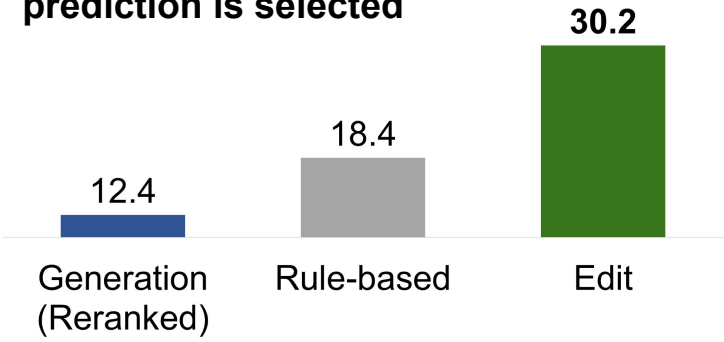
Results: Human Evaluation

Annotation task

Given C_{old} and code diff:

- Select the most suitable C'_{new}
- Select **None** if all options are bad or if C_{old} does not need to be updated

Percent of times each model's prediction is selected



Results: Human Evaluation

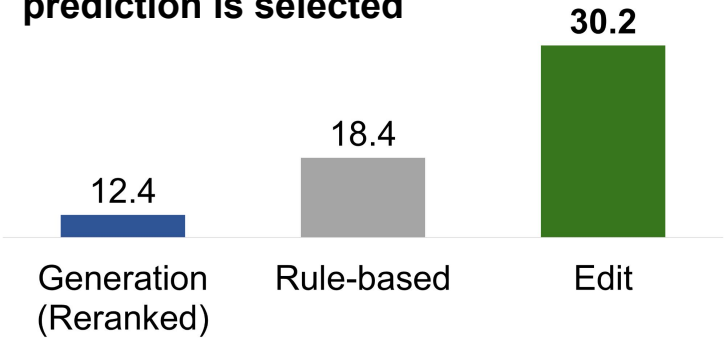
Annotation task

Given C_{old} and code diff:

- Select the most suitable C'_{new}
- Select **None** if all options are bad or if C_{old} does not need to be updated

- Edit model outperforms generation and rule-based baselines

Percent of times each model's prediction is selected



Results: Human Evaluation

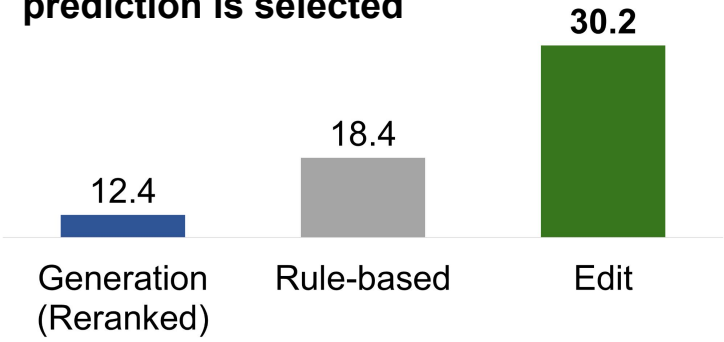
Annotation task

Given C_{old} and code diff:

- Select the most suitable C'_{new}
- Select **None** if all options are bad or if C_{old} does not need to be updated

- Edit model outperforms generation and rule-based baselines
- Users selected **None** 55% of the time

Percent of times each model's prediction is selected



Results: Human Evaluation

Annotation task

Given C_{old} and code diff:

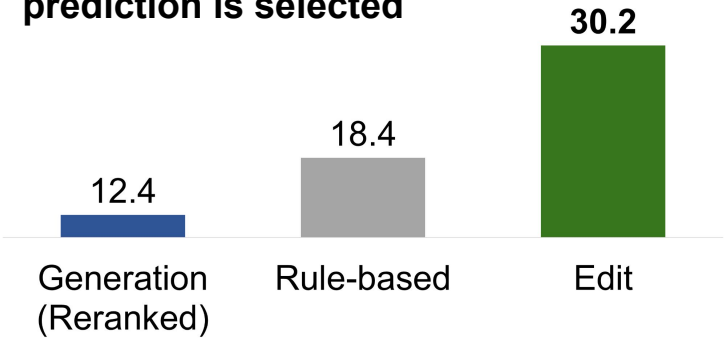
- Select the most suitable C'_{new}
- Select **None** if all options are bad or if C_{old} does not need to be updated

- Edit model outperforms generation and rule-based baselines
- Users selected **None** 55% of the time
- Not all code changes warrant comment updates

We studied

Combining inconsistency detection and update models through pipelining and joint training.

Percent of times each model's prediction is selected



Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Dialogue in Bug Report Discussions

When a bug is reported, developers engage in a dialogue to collaboratively understand it and ultimately resolve it.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

dev007 added a commit that referenced this issue

1) User reports bug

2) Developers engage in the discussion
(understand problem, diagnose cause, propose solution)

3) Bug is resolved with code changes

Dialogue in Bug Report Discussions

When a bug is reported, developers engage in a dialogue to collaboratively understand it and ultimately resolve it.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

dev007 added a commit that referenced this issue

To expedite bug resolution:

Prior work

- Predicting severity [1]
- Assigning relevant developers [2]
- Localizing “buggy” code [3]

[1] *Lamkanfi et al. 2010, Chaturvedi and Singh 2012, Tian et al. 2012, Yang et al. 2014, Gomes et al. 2019, Arokiam and Bradbury 2020*
[2] *Avik 2006, Baysal et al. 2009, Xi et al. 2018, Balock et al. 2021*
[3] *Saha et al. 2013, Rahman and Roy 2018, Loyola et al. 2018, Zhu et al. 2020*

Dialogue in Bug Report Discussions

When a bug is reported, developers engage in a dialogue to collaboratively understand it and ultimately resolve it.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

dev007 added a commit that referenced this issue

To expedite bug resolution:

Prior work

- Predicting severity [1]
- Assigning relevant developers [2]
- Localizing “buggy” code [3]

[1] *Lamkanfi et al. 2010, Chaturvedi and Singh 2012, Tian et al. 2012, Yang et al. 2014, Gomes et al. 2019, Arokiam and Bradbury 2020*
[2] *Avik 2006, Baysal et al. 2009, Xi et al. 2018, Balock et al. 2021*
[3] *Saha et al. 2013, Rahman and Roy 2018, Loyola et al. 2018, Zhu et al. 2020*

Dialogue in Bug Report Discussions

When a bug is reported, developers engage in a dialogue to collaboratively understand it and ultimately resolve it.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

dev007 added a commit that referenced this issue

To expedite bug resolution:

Prior work

- Predicting severity [1]
- Assigning relevant developers [2]
- Localizing “buggy” code [3]

Solution is often formulated in discussion but buried under large amount of text.

[1] *Lamkanfi et al. 2010, Chaturvedi and Singh 2012, Tian et al. 2012, Yang et al. 2014, Gomes et al. 2019, Arokiam and Bradbury 2020*
[2] *Avik 2006, Baysal et al. 2009, Xi et al. 2018, Balock et al. 2021*
[3] *Saha et al. 2013, Rahman and Roy 2018, Loyola et al. 2018, Zhu et al. 2020*

Dialogue in Bug Report Discussions

When a bug is reported, developers engage in a dialogue to collaboratively understand it and ultimately resolve it.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

dev007 added a commit that referenced this issue

To expedite bug resolution:

Prior work

- Predicting severity [1]
- Assigning relevant developers [2]
- Localizing "buggy" code [3]

Solution is often formulated in discussion but buried under large amount of text.

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time

[1] *Lamkanfi et al. 2010, Chaturvedi and Singh 2012, Tian et al. 2012, Yang et al. 2014, Gomes et al. 2019, Arokiam and Bradbury 2020*
[2] *Avik 2006, Baysal et al. 2009, Xi et al. 2018, Balock et al. 2021*
[3] *Saha et al. 2013, Rahman and Roy 2018, Loyola et al. 2018, Zhu et al. 2020*

Dialogue in Bug Report Discussions

When a bug is reported, developers engage in a dialogue to collaboratively understand it and ultimately resolve it.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

To expedite bug resolution:

Prior work

- Predicting severity [1]
- Assigning relevant developers [2]
- Localizing "buggy" code [3]

Solution is often formulated in discussion but buried under large amount of text.

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time

[1] *Lamkanfi et al. 2010, Chaturvedi and Singh 2012, Tian et al. 2012, Yang et al. 2014, Gomes et al. 2019, Arokiam and Bradbury 2020*
[2] *Avik 2006, Baysal et al. 2009, Xi et al. 2018, Balock et al. 2021*
[3] *Saha et al. 2013, Rahman and Roy 2018, Loyola et al. 2018, Zhu et al. 2020*

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Generating Solution Descriptions

Primary Task: Generation

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Given: Title and utterances

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

Generate a concise NL description of the solution

Generating Solution Descriptions

We mine 12K bug reports reports for open-source Java projects on GitHub Issues which are linked to a single commit/PR.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Given: Title and utterances $[U_1, U_2, U_3, U_4]$

dev007 added a commit that referenced this issue

↓ *Commit message/PR*

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

Filtering Techniques for Reducing Noise

- **Generic descriptions** (e.g., *fix bug*)
- **Uninformative descriptions** (e.g., restate problem mentioned in title: *black screen appears when we seek over an AdGroup*)
- **Discussions with insufficient context** (e.g., solution not adequately discussed)

Title: CSE NPE

Utterance #1
com.facebook.presto.spi.PrestoException: Compiler failed
at com.facebook.presto.sql.planner.LocalExecutionPlanner$Visitor.visitScanFilterAndProject(LocalExecutionPlanner.java:1320)...

Utterance #2
cc: @rongrong

Utterance #3
Do you have a repro? Or PM me the query that failed.

Generating Solution Descriptions

Benchmarking existing approaches

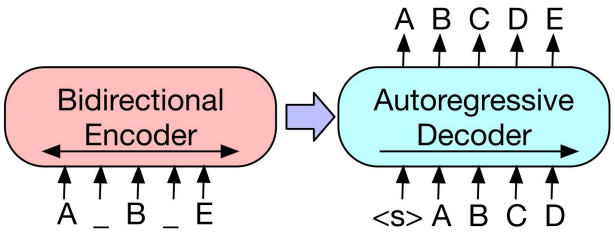
- Copy Title

Generating Solution Descriptions

Benchmarking existing approaches

- Copy Title
- Fine-tune PLBART *[Ahmad et al. 2021]*

BART *[Lewis et al. 2020]*

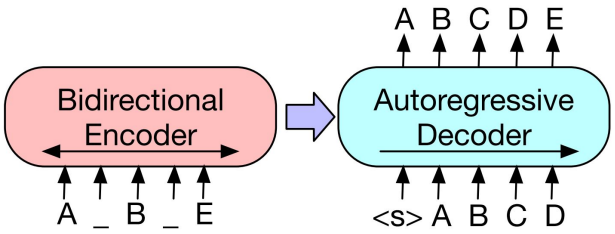


Generating Solution Descriptions

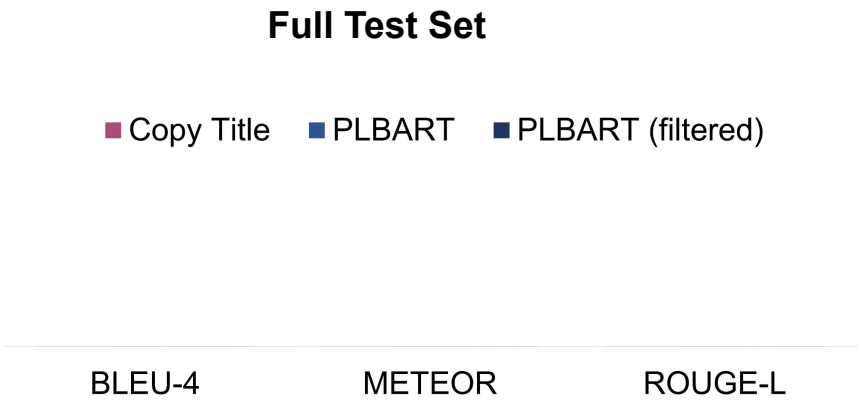
Benchmarking existing approaches

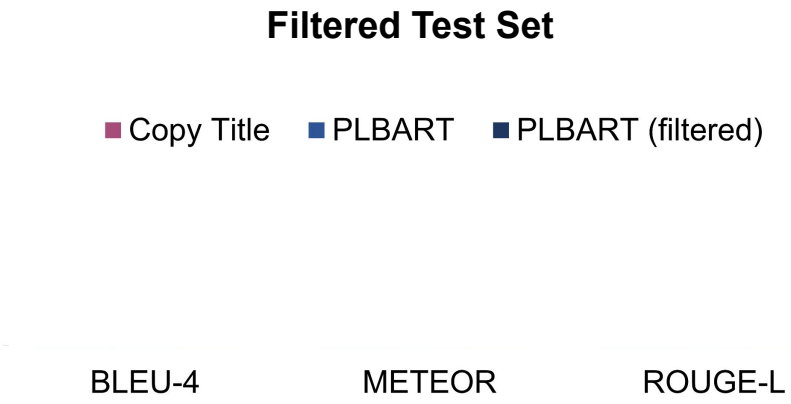
- Copy Title
- Fine-tune PLBART *[Ahmad et al. 2021]*
  - Full training set
  - Filtered training set

BART *[Lewis et al. 2020]*



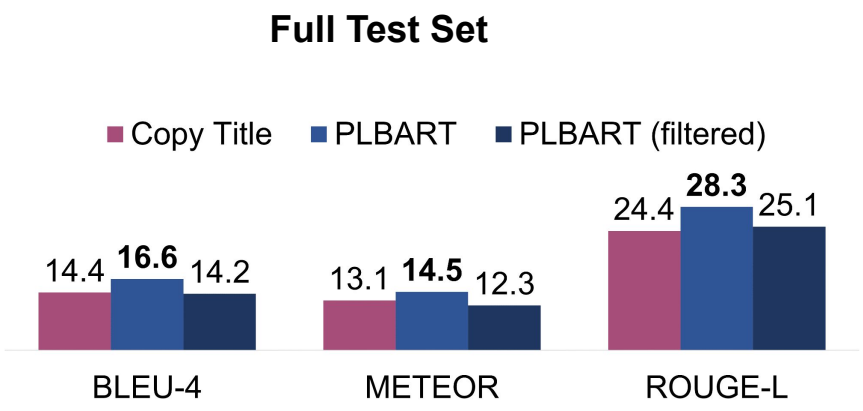
Results: Automatic Metrics

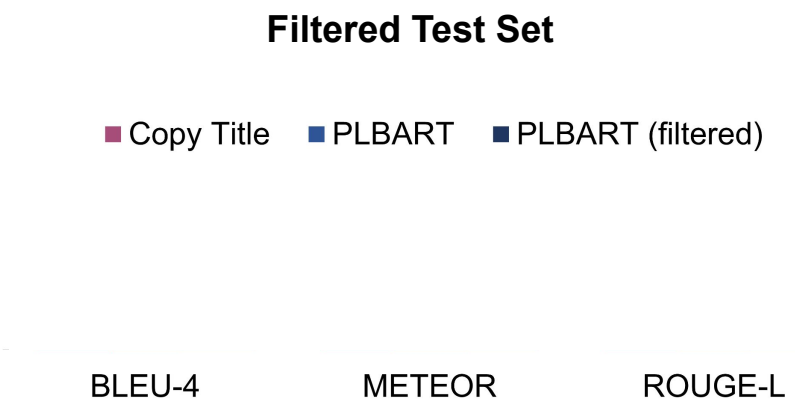




METEOR *[Banerjee and Lavie 2005];* ***BLEU-4*** *[Papineni et al. 2002];* ***ROUGE*** *[Lin et al. 2004]*

Results: Automatic Metrics

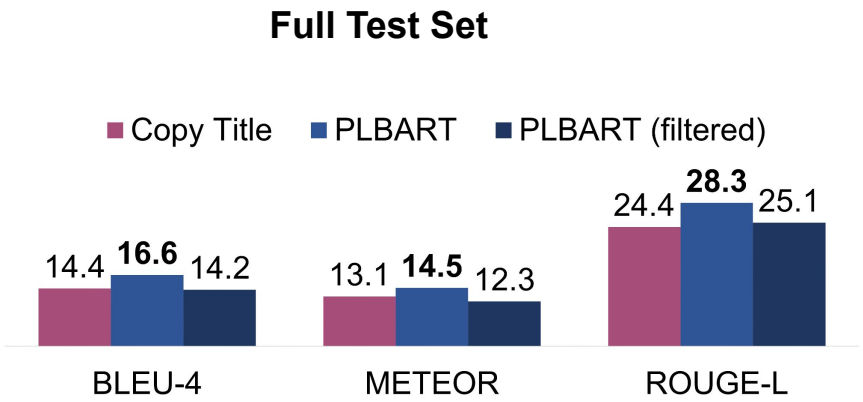


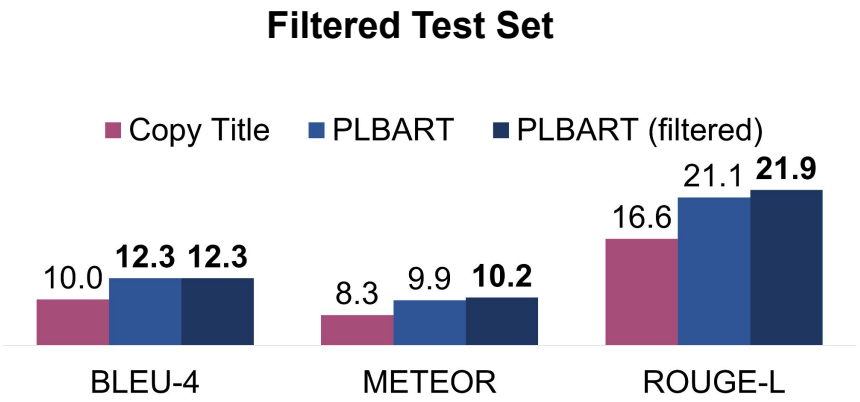


PLBART is the best model on the **full** test set

METEOR *[Banerjee and Lavie 2005];* ***BLEU-4*** *[Papineni et al. 2002];* ***ROUGE*** *[Lin et al. 2004]*

Results: Automatic Metrics





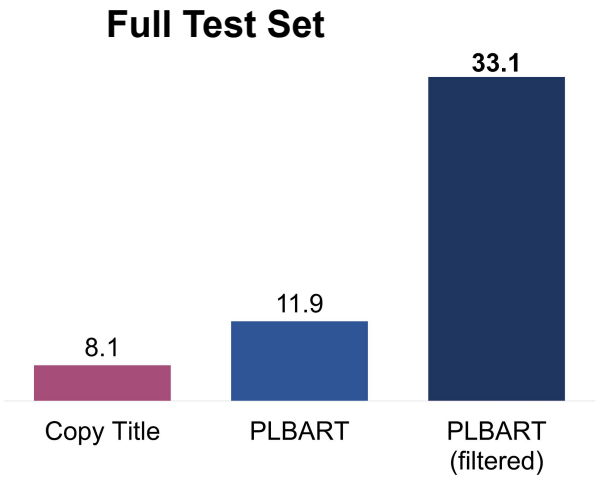
PLBART is the best model on the **full** test set

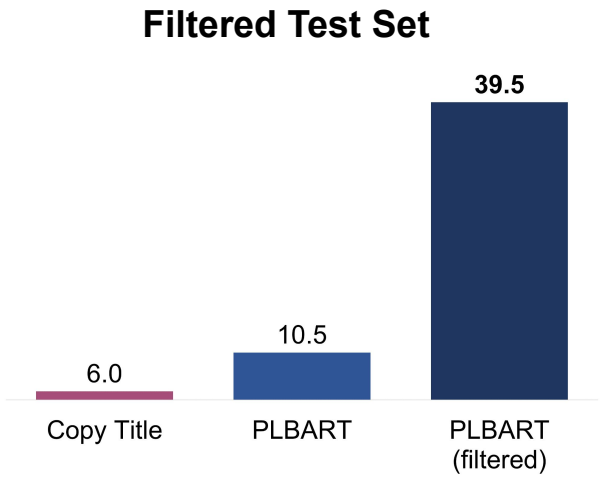
PLBART (filtered) performs slightly better on **filtered** subset

METEOR *[Banerjee and Lavie 2005];* ***BLEU-4*** *[Papineni et al. 2002];* ***ROUGE*** *[Lin et al. 2004]*

Results: Human Evaluation

- Annotators are shown issue title and discussion ($U_1 ... U_g$)
- Presented with model predictions
- Select the one(s) that is/are most informative towards resolving the bug





PLBART (filtered) performs better on both the full and **filtered** test sets

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Determining When to Generate Description

Secondary Task: Classification

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time

After each new utterance U_t, make binary prediction

Once positive label is predicted at t_p, generate

Determining When to Generate Description

We mine 12K bug reports reports for open-source Java projects on GitHub Issues which are linked to a single commit/PR.

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Given: Title and utterances $[U_1, U_2, U_3, U_4]$

dev007 added a commit that referenced this issue

Commit message/PR

Time step of commit/PR

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

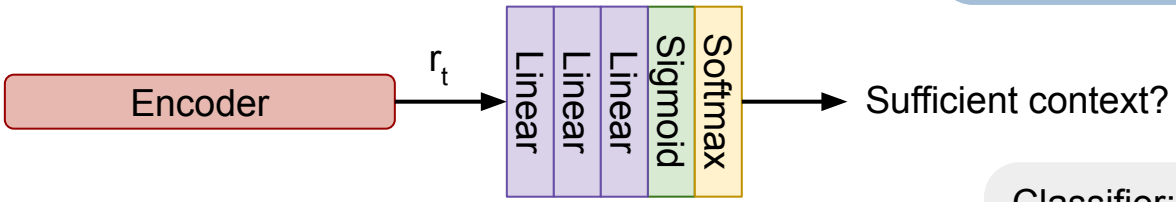
$t_g = 4$

Classifier

Secondary Task: Classification

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time



$t_p = t_g$	32.5%
$t_p < t_g$	45.8%
t_p = **None**	21.6%

Classifier:

- Transformer-based encoder to learn a representation, r_t, for U_t
- Input feeding with r_{t-1}

Combined System

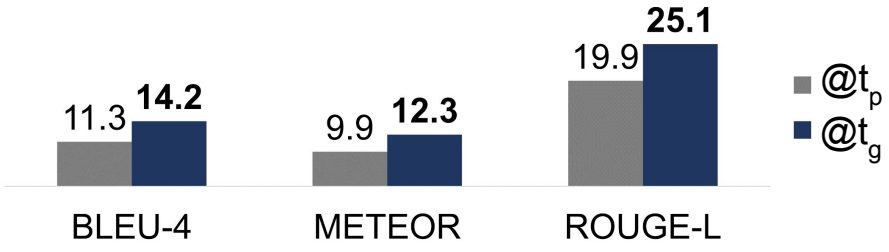
We pipeline classification and generation models:

- **Training:** Classification and generation models are trained separately
- **Inference:**
  (1) Classifier predicts t_p
  (2) Generation model generates description given title, $U_1 ... U_p$

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time

Automatic metrics for PLBART (filtered) using context available @t_p vs @t_g



Gap in performance due to error propagation from classifier

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

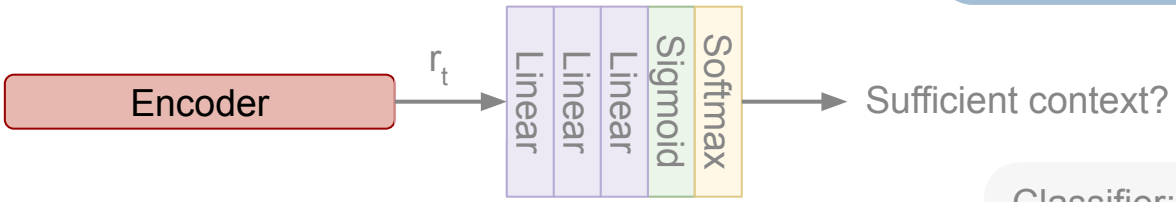
[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Improving Classifier

Secondary Task: Classification

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time



Classifier:

- Transformer-based encoder to learn a representation, r_t, for U_t
- Input feeding with r_{t-1}

We propose

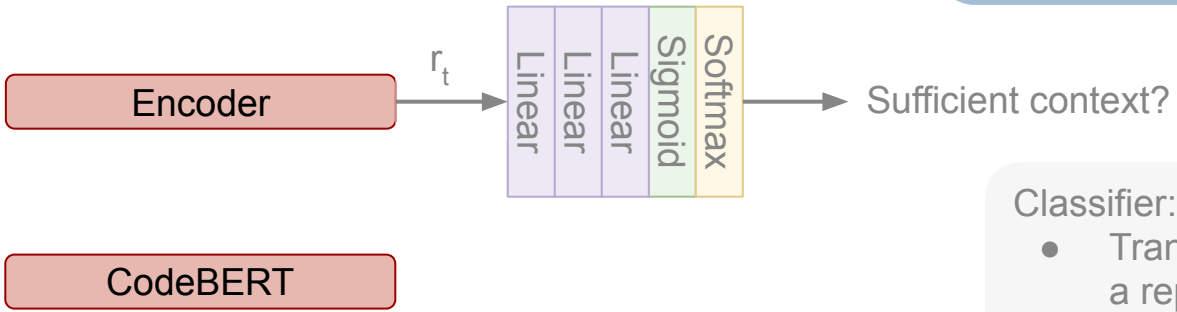
- Fine-tuning pretrained encoders

Improving Classifier

Secondary Task: Classification

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time



Classifier:

- Transformer-based encoder to learn a representation, r_t, for U_t
- Input feeding with r_{t-1}

We propose

- Fine-tuning pretrained encoders

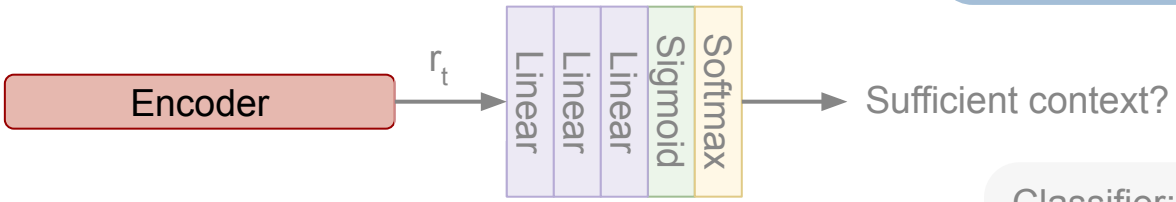
CodeBERT *[Feng et al. 2020]*

Improving Classifier

Secondary Task: Classification

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time



CodeBERT

BERTOverflow

Classifier:

- Transformer-based encoder to learn a representation, r_t, for U_t
- Input feeding with r_{t-1}

We propose

- Fine-tuning pretrained encoders

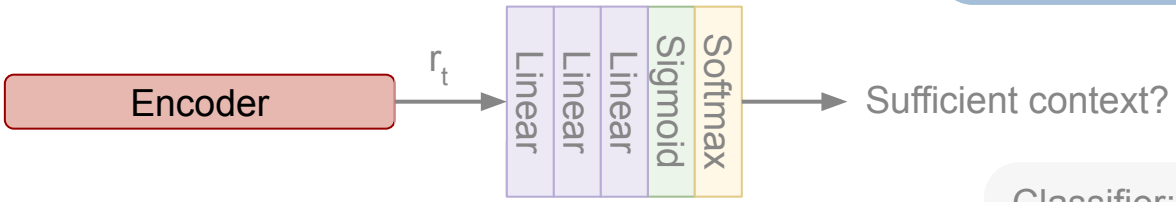
CodeBERT *[Feng et al. 2020];* ***BERTOverflow*** *[Tabassum et al. 2020]*

Improving Classifier

Secondary Task: Classification

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time



Classifier:

- Transformer-based encoder to learn a representation, r_t, for U_t
- Input feeding with r_{t-1}

CodeBERT

BERTOverflow

PLBART

We propose

- Fine-tuning pretrained encoders

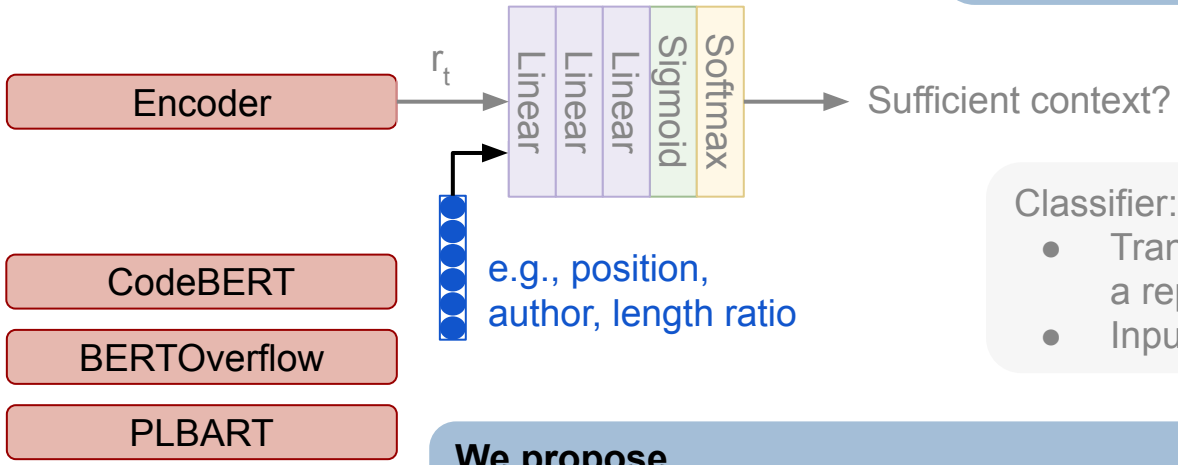
CodeBERT *[Feng et al. 2020];* ***BERTOverflow*** *[Tabassum et al. 2020];* ***PLBART*** *[Ahmad et al. 2021]*

Improving Classifier

Secondary Task: Classification

We studied

Generating solution descriptions by synthesizing relevant content in the discussion when it emerges in real-time



Classifier:

- Transformer-based encoder to learn a representation, r_t, for U_t
- Input feeding with r_{t-1}

We propose

- Fine-tuning pretrained encoders
- Injecting explicit knowledge through features

CodeBERT *[Feng et al. 2020];* ***BERTOverflow*** *[Tabassum et al. 2020];* ***PLBART*** *[Ahmad et al. 2021]*

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Jointly Trained Combined System

We pipeline classification and generation models:

- **Training:** Classification and generation models are trained separately
- **Inference:**
  (1) Classifier predicts t_p
  (2) Generation model generates description given title, $U_1 \ldots U_p$

Generation and classification tasks are inherently intertwined

Jointly Trained Combined System

~~We pipeline classification and generation models:~~

- ~~**Training:** Classification and generation models are trained separately~~
- **Inference:**
  (1) Classifier predicts t_p
  (2) Generation model generates description given title, $U_1 ... U_p$

Generation and classification tasks are inherently intertwined

We propose
Jointly training on generation and classification

Jointly Trained Combined System

~~We pipeline classification and generation models:~~

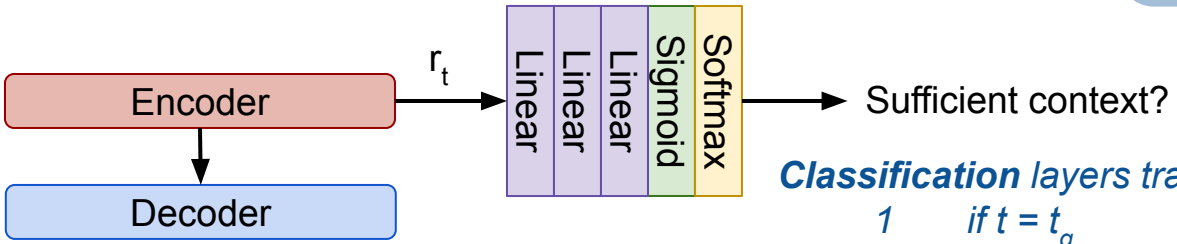
- ~~**Training:** Classification and generation models are trained separately~~
- **Inference:**
  (1) Classifier predicts t_p
  (2) Generation model generates description given title, $U_1 ... U_p$

Generation and classification tasks are inherently intertwined

We propose
Jointly training on generation and classification

Initialize encoder-decoder from PLBART

Shared encoder



*Decoder fine-tuned on **generating** description*

***Classification** layers trained on predicting:*

1 if $t = t_g$
0 else

Jointly Trained Combined System

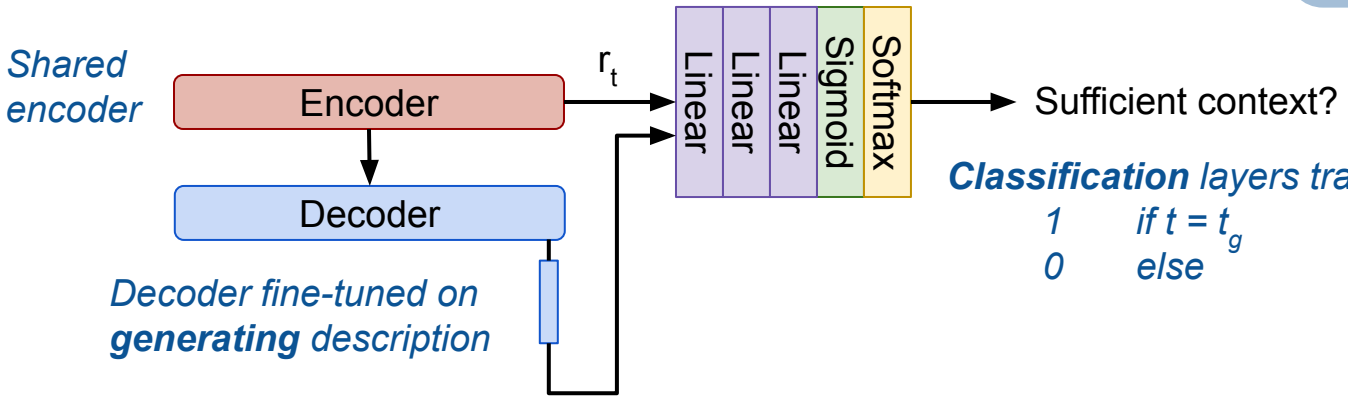
~~We pipeline classification and generation models:~~

- ~~**Training:** Classification and generation models are trained separately~~
- **Inference:**
  (1) Classifier predicts t_p
  (2) Generation model generates description given title, $U_1 ... U_p$

Generation and classification tasks are inherently intertwined

We propose
Jointly training on generation and classification

Initialize encoder-decoder from PLBART



***Classification** layers trained on predicting:*

1 if $t = t_g$
0 else

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Suggesting Bug-Resolving Code Changes Based on Discussions

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

```
public int getL1Distance (int x1, int x2) {
-       return x1-x2;
+       return Math.abs(x1-x2);
}
```

How should this materialize as concrete code changes?

We propose
Generating suggested code changes

Suggesting Bug-Resolving Code Changes Based on Discussions

Generating bug-resolving code changes

Buggy code

```
public int getL1Distance (int x1, int x2) {
      return x1-x2;
}
```

Fixed code

```
public int getL1Distance (int x1, int x2) {
      return Math.abs(x1-x2);
}
```

Recent work

Incorporating NL description can guide code edits

[Chakraborty and Ray 2021, Tufano et al. 2021, Elgohary et al. 2021]

We propose

Incorporating bug report discussions to guide code edits

Suggesting Bug-Resolving Code Changes Based on Discussions

Generating bug-resolving code changes

Buggy code

```
public int getL1Distance (int x1, int x2) {
      return x1-x2;
}
```

Fixed code

```
public int getL1Distance (int x1, int x2) {
      return Math.abs(x1-x2);
}
```

Recent work

Incorporating NL description can guide code edits

[Chakraborty and Ray 2021, Tufano et al. 2021, Elgohary et al. 2021]

We propose

Incorporating bug report discussions to guide code edits

Incorporating bug report discussions and solution descriptions to guide code edits

Suggesting Bug-Resolving Code Changes Based on Discussions

We propose

Incorporating bug report discussions to guide code edits

Incorporating bug report discussions and solution descriptions to guide code edits

Tailor PLBART

- Incorporate copy mechanism [1]
- Decode edits
- Leverage flattened AST sequences [2]

[1] *Vinyals et al. 2015, Panthaplackel et al. 2021*
[2] *Hu et al. 2018*

Suggesting Bug-Resolving Code Changes Based on Discussions

We propose

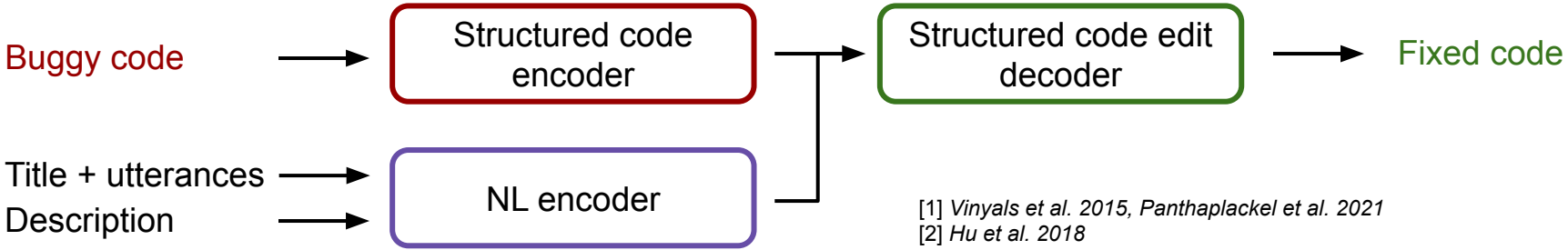
Incorporating bug report discussions to guide code edits

Incorporating bug report discussions and solution descriptions to guide code edits

Tailor PLBART

- Incorporate copy mechanism [1]
- Decode edits
- Leverage flattened AST sequences [2]

Integrate with structured edit models [3]



[1] *Vinyals et al. 2015, Panthaplackel et al. 2021*
[2] *Hu et al. 2018*
[3] *Tarlow et al. 2020, Yao et al. 2021, Mesbah et al. 2019*

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Interactively Generating NL Descriptions to Drive Code Changes

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

Interactively Generating NL Descriptions to Drive Code Changes

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

t_p

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

Utterance #5
Let's also log it before returning

Given: *title and* $[U_1, U_2, U_3, U_4]$

Generate description_p

We propose
Generating descriptions at multiple time steps

Cannot react to new utterances after generation

Interactively Generating NL Descriptions to Drive Code Changes

Title: Incorrect distance

Utterance #1
Seeing negative distance when using 1D grid.

Utterance #2
Probably a bug in `getL1Distance(int x1, int x2)`

Utterance #3
We do `x1 - x2`, which will be negative if `x1 < x2`.

Utterance #4
We should compute its absolute value.

t_p

Description
Compute absolute value of `x1 - x2` in `getL1Distance`

Generate description$_p$

Utterance #5
Let's also log it before returning

t_{p+k}

Description
Log distance before returning from `getL1Distance`

Generate description$_{p+k}$

Given: *title and* $[U_1, U_2, U_3, U_4]$

Given: *title and* $[U_1, U_2, U_3, U_4]$ $[\text{description}_p, U_5]$

We propose
Generating descriptions at multiple time steps

Cannot react to new utterances after generation

Interactively Generating NL Descriptions to Drive Code Changes

Driving code changes in PR discussion interactions

Title: Compute absolute value of `x1 - x2` and log it in `getL1Distance`

Author

```
/** Computes distance as difference between x1 and x2 */
/** Computes distance as  magnitude of difference between x1 and x2
*/
public int getL1Distance (int x1, int x2) {
-       return x1-x2;
+       int distance = Math.abs(x1-x2);
+       log.debug(String.format(" (%d)", distance));
+       return distance;
}
```

Reviewer
Please make the log message more descriptive.

Author
Will add in something about it being L1 distance. Anything else that should be included?

Reviewer
Maybe that it's for the 1D grid?

Author

```
+       log.debug(String.format("(%d)", distance));
+       log.debug(String.format(" L1 Distance in 1D  (%d)", distance));
}
```

Prior work

- Recommending reviewers [1]
- PR prioritization [2]
- Determining where to post review comment [3]
- Previewing changes [4]

We propose

Building an agent to simulate the role of the **reviewer** by prescribing code changes

[1] *Yu et al. 2014*
[2] *van der Veen et al. 2015*
[3] *Hellendoorn et al. 2021*
[4] *Tufano et al. 2021*

Overview

<u>Comments</u>
Goal #1:
Uphold software quality amidst constant changes

Completed Work
- Detecting inconsistent comments [1]
- Updating inconsistent comments [2]

<u>Dialogue</u>
Goal #2:
Facilitate prompt implementation of critical changes

- Generating solution descriptions based on bug report discussions [3]
- Initial study of real-time system for generating solution descriptions [3]

Proposed Work

Short-Term Goals
- Improving classifier for assessing context in ongoing discussion
- Using joint training for real-time system

Long-Term Goals
- Suggesting bug-resolving code changes based on discussions
- Interactively generating NL descriptions to drive code changes

[1] *Panthaplackel et al. AAAI 2021*
[2] *Panthaplackel et al. ACL 2020*
[3] *Panthaplackel et al. preprint 2021*

Questions/comments?