Automatically Generating Commit Messages from Diffs Using Neural Machine Translation

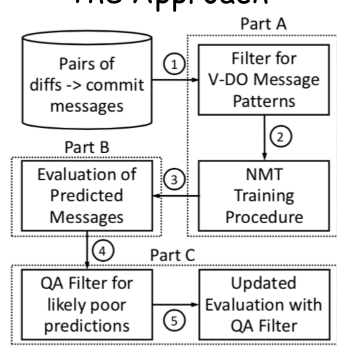
Problem Statement

- Commit messages are important for software change comprehension
- However, developers do not always write good commit messages

Contributions

- This paper adapts a neural machine translation (NMT) algorithm to automatically generate commit messages from program diffs
- The commit messages generated either have very high or very low quality
- Open source the data sets and software

The Approach



Preparing a Data Set for NMT

- Initially started with 2M commits from the top 1k Java projects
  - As 82% of the commit messages have only one sentence, so this paper aims to generate one-sentence commit message
  - Remove issue ids from the extracted sentences and removed commit ids from diffs
    - Perhaps the commit ids should be also removed from the commit messages?

Preparing a Data Set for NMT (cont'd)

- Removed merge and rollback commits-> 1.8M
- Set the maximum length of diffs as 100 tokens, because the tool does not work well for the settings 50 and 200 -> 75k
- Remove messages not matching the V-DO patterns (verb/direct-object) -> 32k
- Split the 32k messages so that 26k are used for training, 3k for testing, and 3k for validation

NMT Training and Testing

- Evaluation
  - RQ1: Compared to the messages generated by a baseline, are the messages created by NMT more or less similar to the oracle?
    - Compared with MOSES, a popular statistical machine translation software
  - RQ2: With V-DO filter enabled or disabled, how does the NMT model create messages?
    - V-DO filter should be used to effectively remove messages with low quality or complex messages

RQ1 & RQ2

TABLE I: BLEU SCORES (%) OF MOSES AND OUR MODELS ON THE TEST SET

Model	BLEU	Len_{Gen}	Len_{Ref}	p_1	p_2	p_3	p_4
MOSES	3.63	129889	22872	8.3	3.6	2.7	2.1
NMT1	31.92	24344	22872	38.1	31.1	29.5	29.7
NMT2	32.81	21287	22872	40.1	34.0	33.4	34.3
	23.10*	20303	18658	30.2	23.3	20.7	19.6

MOSES is the baseline model. NMT1 is the NMT model with V-DO filter described in Section IV-B. NMT2 is a model trained without V-DO filter described in Section V-D. Len_{Gen} is the total length of the generated messages (c in Equation (11)). Len_{Ref} is the total length of the reference messages (r in Equation (11)). The modified n-gram precision p_n, where $n = 1, 2, 3, 4$, is defined in Equation (8).
* This BLEU score is calculated on a test set that is not V-DO filtered described in Section V-D. The other BLEU scores are tested on a V-DO filtered test set described in Section IV-A4.

RQ1 (cont'd)

TABLE II: BLEU SCORES (%) ON DIFFS OF DIFFERENT LENGTHS

Diff Length	BLEU	Len_{Gen}	Len_{Ref}	p_1	p_2	p_3	p_4
≤ 25	6.46	870	655	18.6	6.9	4.3	3.1
$> 25, \leq 50$	9.31	3627	3371	23.1	10.8	6.6	4.5
$> 50, \leq 75$	12.67	4779	4418	24.8	14.1	9.8	7.6
> 75	43.33	15068	14428	47.1	42.3	41.7	42.3

See Table I for explanation of each column name. The BLEU scores are calculated based on the test results generated by Model1, the NMT model with V-DO filter trained in Section IV-B.

Human Evaluation

- BLEU is a widely used metric to compare translation models, but it is not recommended for evaluating individual sentences
- Human evaluation can assess how similar a generated message is to the original human-created message

An Example

Example 1 of 3

message 1: "Added X to readme"
message 2: "edit readme"

Recommended score: 6

Explanation: The two messages have only one shared word, "readme". But the two messages are very similar in the meaning, because "Added" is a type of "edit".

Results

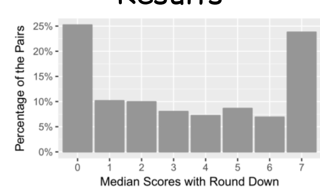


Fig. 9: The distribution of the median scores obtained in the human study. There are 983 scores in the figure. Each score is the median score of the scores made by one to three human experts for a generated message. The scores range from 0 to 7, where 0 denotes the generated message is not similar to the reference message at all, and 7 denotes the generated message is identical to the reference message. The most frequent scores are 0 and 7. There are 248 messages scored 0 and 234 messages scored 7. For the rest of the scores, the number of messages ranges from 68 to 100.

QA Filter

- To filter out the generated bad messages with SVM
  - Bad: 0, 1
  - Not bad: 2-7

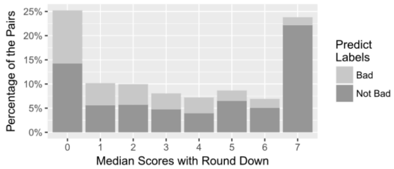


Fig. 11: The predict results of the cross evaluation of QA filter. QA filter reduced 108 messages that are scored 0, 45 messages that are scored 1, 42 messages that are scored 2, 32 messages that are scored 3, 32 messages that are scored 4, 21 messages that are scored 5, 18 messages that are scored 6, and 16 messages that are scored 7. We note that although we trained the QA filter with binary labels, "bad" and "not bad", the evaluation result shows that QA filter is able to reduce more messages for lower scores.

13

Results

- QA filter reduced 44% of the "bad" messages at a cost of 11% of the "good" messages (6-7)

14

An Exemplar Generated Good Message

TABLE III: Example Result

Diff:

```
-- a/core/.../CursorToBulkCursorAdaptor.java
+++ b/core/.../CursorToBulkCursorAdaptor.java
@@ -143,8 +143,7 @@ public final class
CursorToBulkCursorAdaptor ...
 public void close() {
 maybeUnregisterObserverProxy();
- mCursor.deactivate();
-
+ mCursor.close();
 }
 public int requery(IContentObserver observer, ...
```

Generated Message:
"CursorToBulkCursorAdapter . Close must call mCursor . Close instead of mCursor . Deactivate . "

Reference Message:
"Call close ( ) instead of deactivate ( ) in CursorToBulkCursorAdaptor . close ( ) "

Scores: 7, 6, 7

15