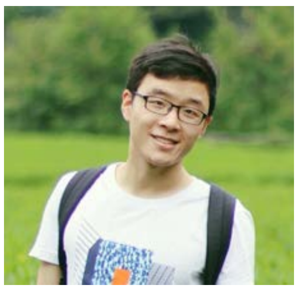
Neural-Machine-Translation-Based Commit Message Generation: How Far Are We?

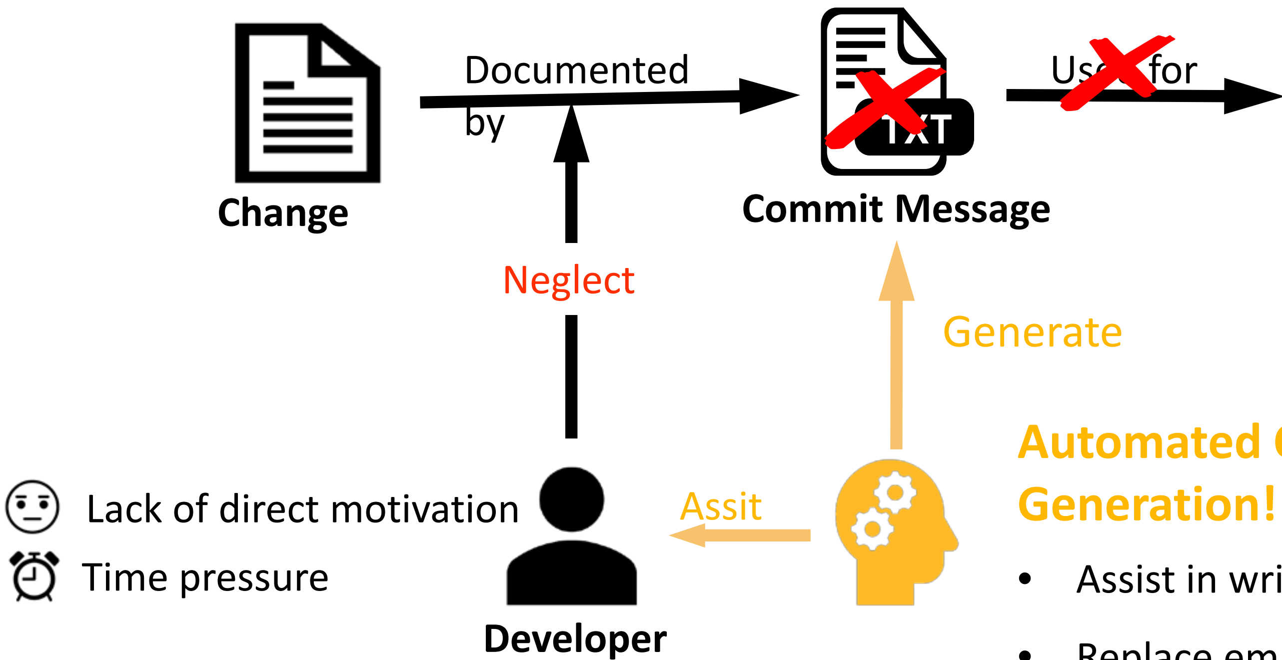
Zhongxin Liu, **Xin Xia**, Ahmed E. Hassan, David Lo, Zhenchang Xing and Xinyu Wang






ZHEJIANG UNIVERSITY 1897
MONASH University
Queen's UNIVERSITY
SMU SINGAPORE MANAGEMENT UNIVERSITY
ZHEJIANG UNIVERSITY 1897

Commit Messages



Program Comprehension

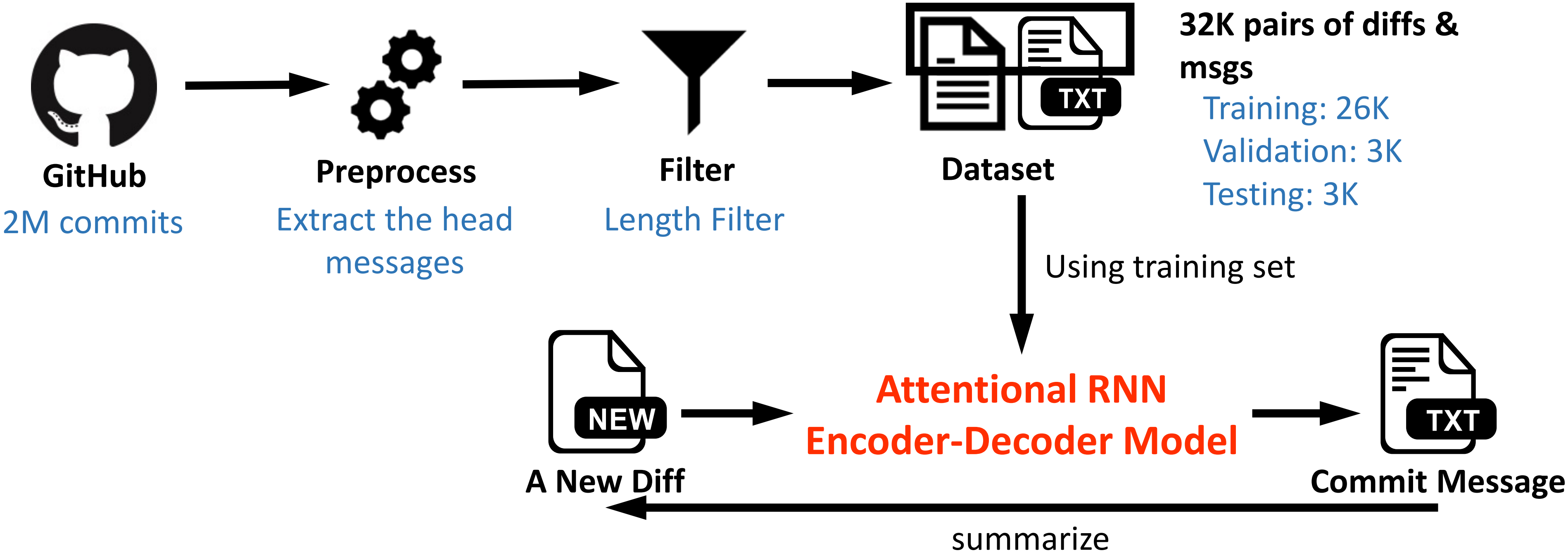
Software Maintenance

Automated Commit Message Generation!

- Assist in writing high-quality messages
- Replace empty messages

NMT-Based Commit Message Generation

- Recently, Jiang et al. proposed an approach, which uses a Neural Machine Translation (NMT) algorithm to generate one-sentence commit messages from diffs. [Jiang et al. ASE 2017]



Evaluation of *NMT*

- For convenience, we refer to Jiang et al.'s approach as *NMT*
- Jiang et al. evaluated *NMT* using the BLEU-4 score:
  - an accuracy measure that is widely used to evaluate machine translation systems

Model	Task	BLEU-4
NMT	diff -> commit msg	31.92

Model	Task	BLEU-4
Transformer[1]	En -> Fr	41.0
	En -> De	28.4

- *NMT*'s performance appears promising!

[1] Vaswani, Ashish, et al. "Attention is all you need." *Advances in Neural Information Processing Systems*. 2017.

However …

- Jiang et al. did not investigate the reasons behind *NMT*'s good performance.

RQ1: Why does NMT perform so well?

- *NMT* is complicated and slow!
  - Attentional RNN encoder-decoder model
  - 38 hours for training on a GPU

RQ2: Can a simpler and faster approach outperform *NMT?*

RQ1: Why does NMT perform so well?

Analyze *NMT Messages*

- *NMT messages:* commit messages generated by *NMT*

200 Test Commits

Randomly select 200 commits from Jiang et al.'s test set

Human Evaluation

- Scores range from 0-7
- Pick out *NMT messages* of score of 6 or 7 (*good messages*)

Manually Analyze
Good Messages

- There are some surprising findings.

Noisy Messages

- Many (37%) of the *reference messages* of these *good messages* are noisy.
- Two types of noisy messages:

liferay-continuous-integration

```
modules/apps/foundation/portal/.gitrepo CHANGED
@@ -3,7 +3,7 @@
3 3    ;
4 4    [subrepo]
5 5        cmdver = liferay
6   -      commit = 2f03e545085c159d922fb9eac9b166ee820a94c0
  6 +      commit = c3d68dbcaaa18c18e76bb46697c52e4d8ec6ffa9
7 7        mode = push
8   -      parent = ab9bdb710f55453499286b0269f60effb1c38e36
  8 +      parent = a1f017cdfb2581a936418d584058638f0262b47c
9 9        remote = git@github.com:liferay/com-liferay-portal.git
```

Reference Message:
ignore Update ' modules / apps / foundation / portal / .

Message Generated by NMT:
Ignore Update ' modules / apps / foundation / portal / .

Bot Message

Automatically generated by other dev tools

```
CHANGELOG.md DELETED
@@ -1,7 +0,0 @@
1   - # Changelog
2   -
3   - ## 0.1 (2014-02-20)
4   -
5   - Initial public release
6   -
7   - *
```

Reference Message:
update changelog

Message Generated by NMT:
Updated changelog

Trivial Message

Contains little and redundant information

Learning From or Producing Noisy Messages is of Little Value

modules/apps/foundation/portal/.gitrepo CHANGED

```
@@ -3,7 +3,7 @@
3  3    ;
4  4    [subrepo]
5  5        cmdver = liferay
6    -      commit = 2f03e545085c159d922fb9eac9b166ee820a94c0
   6 +      commit = c3d68dbcaaa18c18e76bb46697c52e4d8ec6ffa9
7  7        mode = push
8    -      parent = ab9bdb710f55453499286b0269f60effb1c38e36
   8 +      parent = a1f017cdfb2581a936418d584058638f0262b47c
9  9        remote = git@github.com:liferay/com-liferay-portal.git
```

Reference Message:
ignore Update ' modules / apps / foundation / portal / .

Message Generated by NMT:
Ignore Update ' modules / apps / foundation / portal / .

Bot Message

CHANGELOG.md DELETED

```
@@ -1,7 +0,0 @@
1    - # Changelog
2    -
3    - ## 0.1 (2014-02-20)
4    -
5    - Initial public release
6    -
7    - *
```

Reference Message:
update changelog

Message Generated by NMT:
Updated changelog

Trivial Message

- They contain little useful information.
- They can be generated through rule-based methods.

Identify Noisy Messages in Jiang et al.'s Dataset

- We find noisy messages from Jiang et al.'s dataset through our manually derived patterns.

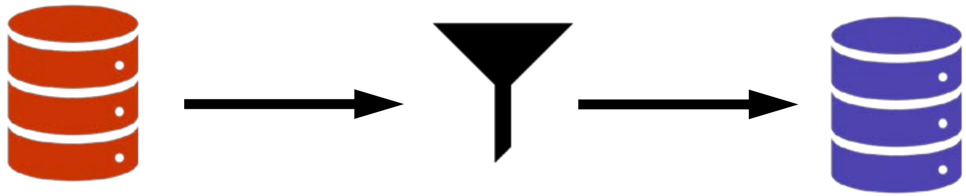
- Proportion of the noisy messages found by us.

Dataset	Bot	Trivial	Total
Training	12.6%	3.1%	15.6%
Validation	13.4%	2.9%	16.3%
Test	12.8%	3.2%	16.0%

Noisy messages are common in Jiang et al.'s dataset!

The Impact of Noisy Commits

- Will these noisy commits affect the performance of *NMT*?



- Train and test *NMT* on the cleaned dataset.

Dataset	BLEU-4
JIANG	31.92
Cleaned	14.19

Performance declines by a large amount!

RQ1: Why does NMT perform so well?

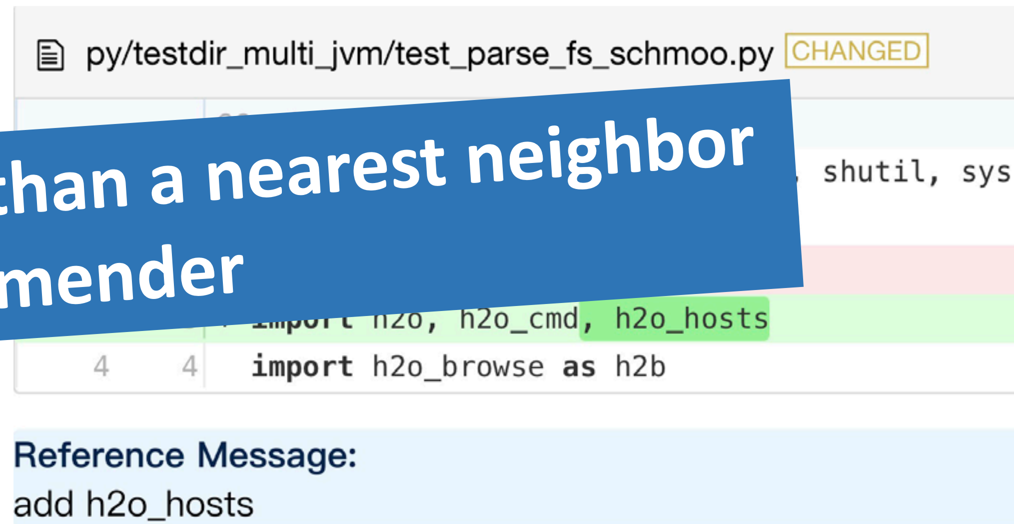
The good performance of *NMT* mainly comes from the noisy commits in Jiang et al.'s dataset!

RQ2: Can a simpler and faster approach outperform *NMT*?

Another Finding of Our Analysis

- For nearly every (70/71) *good message*, we can find out one or more similar training commits:

A Test Commit

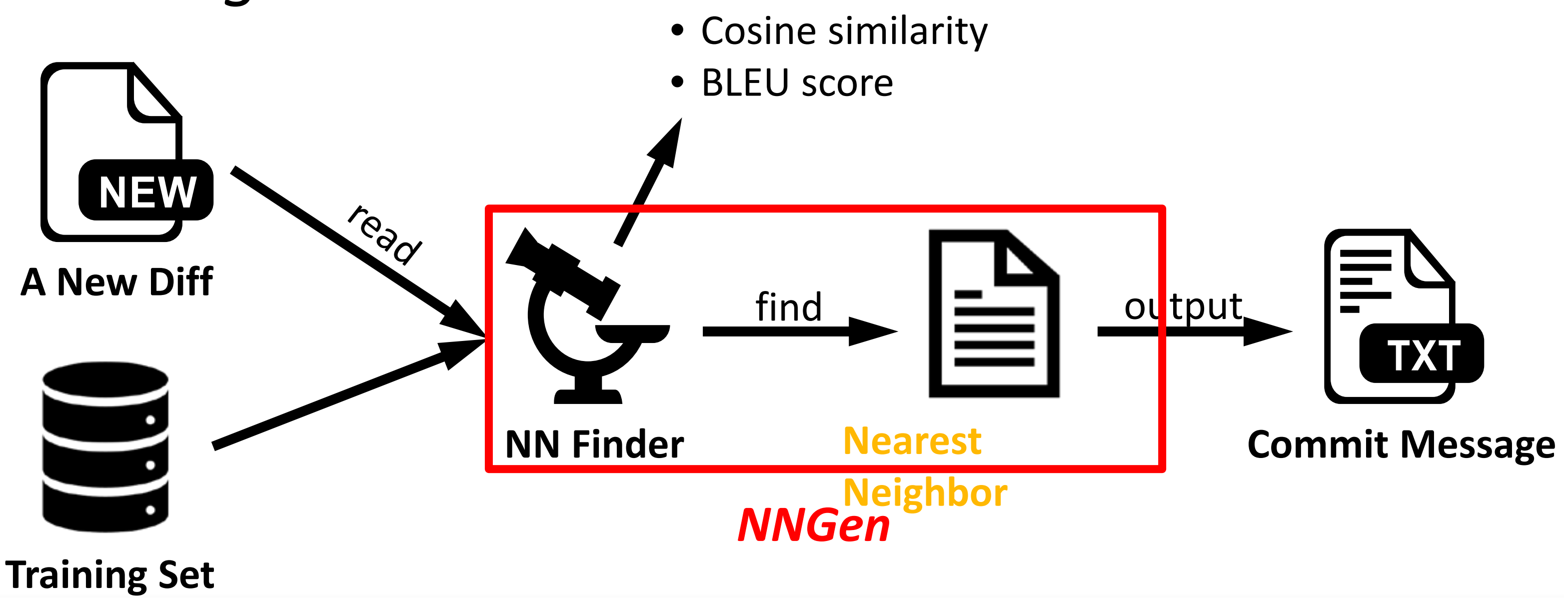


A Similar Training Commit

***NMT* seems no better than a nearest neighbor recommender**

NNGen

- A nearest-neighbor-based approach may be a nice and simple idea!
- We propose a nearest-neighbor-based approach, named *NNGen*
  – *Nearest Neighbor Generator*



Automatic Evaluation & Time Costs

Dataset	Approach	BLEU-4	
JIANG	*NMT*	31.92	
	NNGen	**38.55**	**↑ 21%**
Cleaned	*NMT*	14.19	
	NNGen	**16.42**	**↑ 16%**

Dataset	Approach	Device	Train	Test
JIANG	*NMT*	GTX 1070	38 hours	4.5 mins
	NMT	GTX 1080	34 hours	17 mins
	NNGen	**CPU**	N/A	**30 secs**
Cleaned	*NMT*	GTX 1080	24 hours	13 mins
	NNGen	**CPU**	N/A	**23 secs**

- GTX 1070: Nvidia GTX 1070 GPU, time costs reported by Jiang et al.
- GTX 1080: Nvidia GTX 1080 GPU, time costs on our server
- CPU: Intel i5 2.6GHz

Human Evaluation

- 200 commits randomly picked from the cleaned test set.
- 600 pairs of scores from 6 participants.

Approach	Low-quality	Medium-quality	High-quality	Mean Score
NMT	63.8%	8.8%	27.4%	1.34
NNGen	**57.9%**	**14.3%**	**27.8%**	**1.46**

- Wilcoxon signed-rank test: p-value 0.01 (Significant)

RQ2: Can a simpler and faster approach outperform *NMT*?

Our nearest-neighbor-based approach is considerably faster than *NMT*, and outperforms *NMT* by 16-21%

Implications

- Clean up the data carefully.
  – Noisy commits will affect performance.

- Consider simple approaches first. [Fu and Menzies FSE 2017]
  – Specifically, consider the nearest neighbor algorithm first for diff-msg "translation" tasks.
  – Little effort to understand data, sometimes leads to better performance

- We still have a long way to go to automatically generate high-quality commit messages.
  – The performance on the cleaned dataset is still not sufficient.

Importance of Critical Thinking

Practitioners' Expectations on Automated Fault Localization

Pavneet Singh Kochhar[1], Xin Xia[2]*, David Lo[1], and Shanping Li[2]
[1]School of Information Systems, Singapore Management University, Singapore
[2]College of Computer Science and Technology, Zhejiang University, China
{kochharps.2012,davidlo}@smu.edu.sg, {xxia,shan}@zju.edu.cn

2016 IEEE International Conference on Software Maintenance and Evolution

"Automated Debugging Considered Harmful" Considered Harmful

A User Study Revisiting the Usefulness of Spectra-Based Fault Localization Techniques with Professionals using Real Bugs from Large Systems

Xin Xia[*‡♯], Lingfeng Bao[*♯], David Lo[†], and Shanping Li[*‡]
[*]College of Computer Science and Technology, Zhejiang University, China
[†]School of Information Systems, Singapore Management University, Singapore
[‡]Hengtian Software Ltd., China
{xxia, baolingfeng, shan}@zju.edu.cn, davidlo@smu.edu.sg

Importance of Critical Thinking

2017 IEEE International Conference on Software Maintenance and Evolution

Supervised vs Unsupervised Models: A Holistic Look at Effort-Aware Just-in-Time Defect Prediction

Qiao Huang*, Xin Xia[†✓], and David Lo[‡]
*College of Computer Science and Technology, Zhejiang University, China
[†]Department of Computer Science, University of British Columbia, Canada
[‡]School of Information Systems, Singapore Management University, Singapore
tkdsheep@zju.edu.cn, xxia02@cs.ubc.ca, davidlo@smu.edu.sg

Importance of Critical Thinking

Perceptions, Expectations, and Challenges in Defect Prediction

Zhiyuan Wan, Xin Xia, Ahmed E. Hassan, David Lo, Jianwei Yin, and Xiaohu Yang

Abstract—Defect prediction has been an active research area for over four decades. Despite numerous studies on defect prediction, the potential value of defect prediction in practice remains unclear. To address this issue, we performed a mixed qualitative and quantitative study to investigate what practitioners think, behave and expect in contrast to research findings when it comes to defect prediction. We collected hypotheses from open-ended interviews and a literature review of defect prediction papers that were published at ICSE, ESEC/FSE, ASE, TSE and TOSEM in the last 6 years (2012-2017). We then conducted a validation survey where the hypotheses became statements or options of our survey questions. We received 395 responses from practitioners from over 33 countries across five continents. Some of our key findings include: 1) Over 90% of respondents are willing to adopt defect prediction techniques. 2) There exists a disconnect between practitioners' perceptions and well supported research evidence regarding defect density distribution and the relationship between file size and defectiveness. 3) 7.2% of the respondents reveal an inconsistency between their behavior and perception regarding defect prediction. 4) Defect prediction at the feature level is the most preferred level of granularity by practitioners. 5) During bug fixing, more than 40% of the respondents acknowledged that they would make a "work-around" fix rather than correct the actual error-causing code. Through a qualitative analysis of free-form text responses, we identified reasons why practitioners are reluctant to adopt defect prediction tools. We also noted features that practitioners expect defect prediction tools to deliver. Based on our findings, we highlight future research directions and provide recommendations for practitioners.

Index Terms—Defect Prediction, Empirical Study, Practitioner, Survey

Summary

Performance of *NMT* Declines Once Dataset is Cleaned

- Will these noisy commits affect the performance of *NMT*?

Our derived patterns

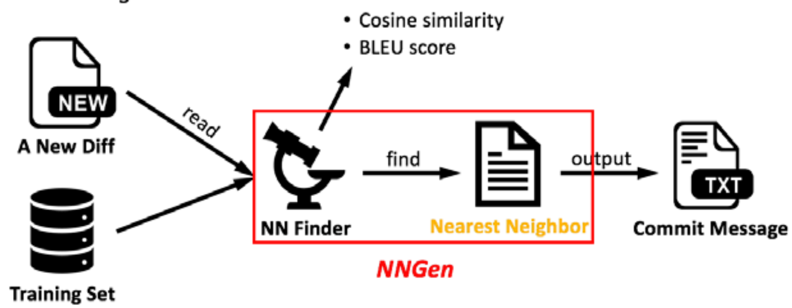
- Train and test *NMT* on the cleaned dataset.

Dataset	BLEU-4
JIANG	31.92
Cleaned	14.19

Performance declines by a large amount!

NNGen

- A nearest-neighbor-based approach may be a nice and simple idea!
- We propose a nearest-neighbor-based approach, named *NNGen*
  - *Nearest Neighbor Generator*



Automatic Evaluation & Time Costs

Dataset	Approach	BLEU-4	
JIANG	*NMT*	31.92	
	NNGen	38.55	↑ 21%
Cleaned	*NMT*	14.19	
	NNGen	16.42	↑ 16%

Dataset	Approach	Device	Train	Test
JIANG	*NMT*	GTX 1070	38 hours	4.5 mins
	NMT	GTX 1080	34 hours	17 mins
	NNGen	CPU	N/A	30 secs
Cleaned	*NMT*	GTX 1080	24 hours	13 mins
	NNGen	CPU	N/A	23 secs

- GTX 1070: Nvidia GTX 1070 GPU, time costs reported by Jiang et al.
- GTX 1080: Nvidia GTX 1080 GPU, time costs on our server
- CPU: Intel i5 2.6GHz

Implications

- Clean up the data carefully.
  - Noisy commits will affect performance.
- Consider simple appro...
  - ... leads to better performance

***NNGen* is a competitive baseline for diff->msg "translation" tasks.**

- We still have a long way to go to automatically generate high-quality commit messages.
  - The performance on the cleaned dataset is still not sufficient.

MONASH University