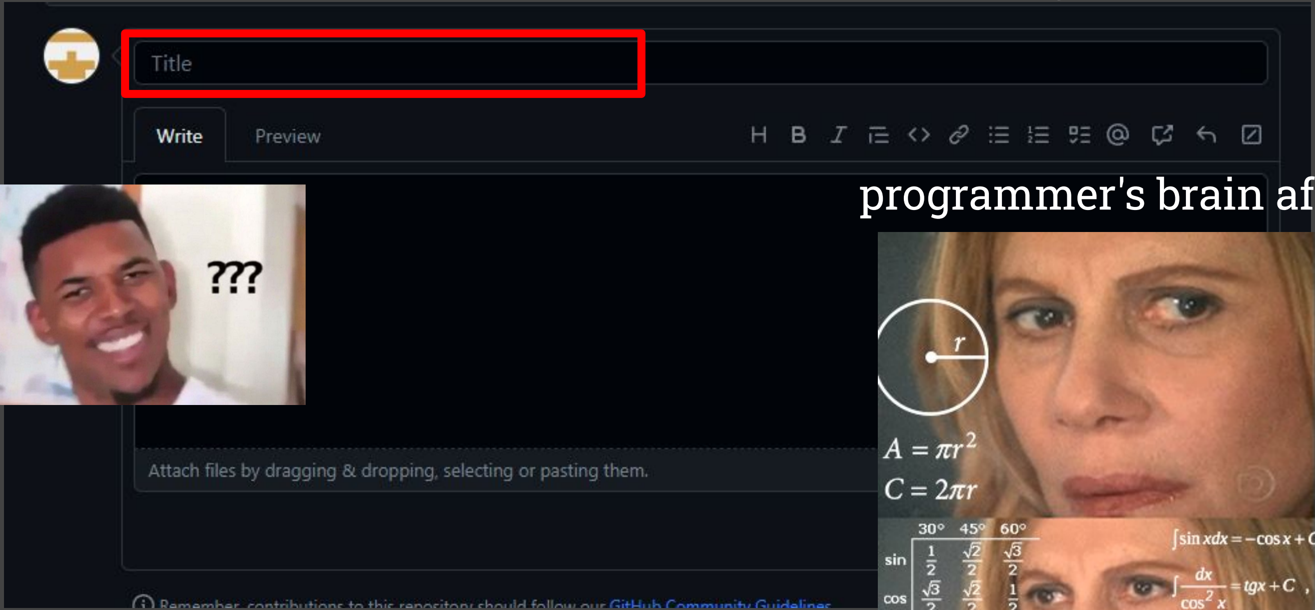
Pull Request Title Generation

NLP Final Project - No.1 NLP

Introduction

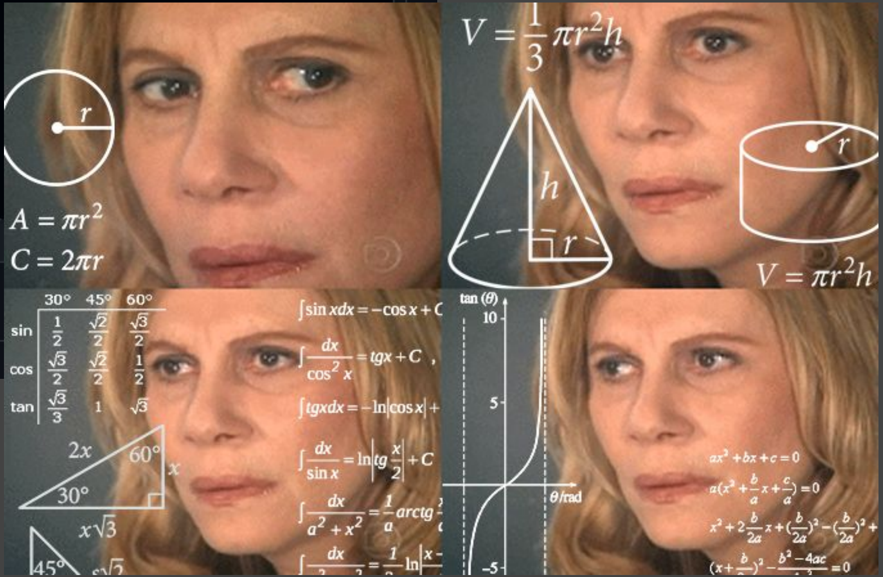
What title should I use for my PR?



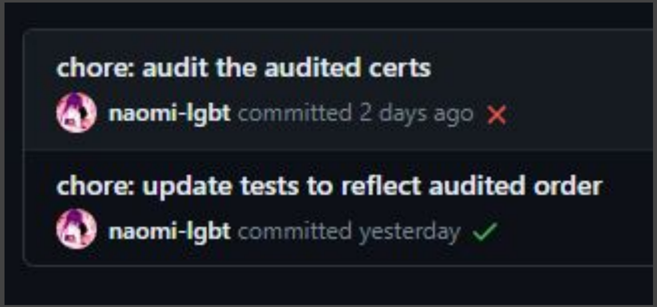
Attach files by dragging & dropping, selecting or pasting them.

Remember, contributions to this repository should follow our GitHub Community Guidelines

programmer's brain after hours of hard work



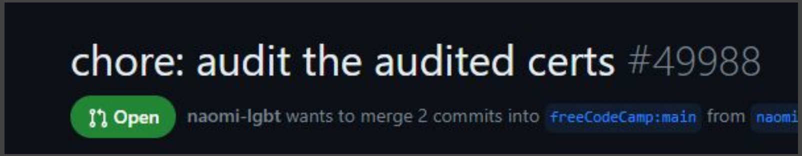
commit


chore: audit the audited certs
naomi-lgbt committed 2 days ago ✗

chore: update tests to reflect audited order
naomi-lgbt committed yesterday ✓

Generate

pull request title


chore: audit the audited certs #49988
Open naomi-lgbt wants to merge 2 commits into `freeCodeCamp:main` from `naomi`

Automatic Pull Request Title Generation

Ting Zhang, Ivana Clairine Irsan, Ferdian Thung, DongGyun Han, David Lo and Lingxiao Jiang
School of Computing and Information Systems, Singapore Management University
Email: {tingzhang.2019, ivanairsan, ferdianthung, dhan, davidlo, lxjiang}@smu.edu.sg

***Abstract*—Pull Requests (PRs) are a mechanism on modern collaborative coding platforms, such as GitHub. PRs allow developers to tell others that their code changes are available for merging into another branch in a repository. A PR needs to be reviewed and approved by the core team of the repository before the changes are merged into the branch. Usually, reviewers need to identify a PR that is in line with their interests before providing a review. By default, PRs are arranged in a list view that shows the titles of PRs. Therefore, it is desirable to have a precise and concise title, which is beneficial for both reviewers and other developers. However, it is often the case that developers do not provide good titles; we find that many existing PR titles are either inappropriate in length (i.e., too short or too long) or fail to convey useful information, which may result in PR being ignored or rejected. Therefore, there is a need for automatic techniques to help developers draft high-quality titles.**

In this paper, we introduce the task of automatic generation of PR titles. We formulate the task as a one-sentence summarization

to help the *readers* (not limited to integrators, but refers to anyone reading the PR) grasp the context and purpose of the PR. Frequently, a PR is linked with one or more issue reports. Issue reports in issue tracking systems (e.g., GitHub issues) are used to keep track of bugs, enhancements, or other requests. Therefore, many PRs contain the identifiers of the linked issues in their titles or descriptions. Since contributors often neglect to write a PR description, the role of the PR titles is significant. For example, 219,909 PRs (16.4%) do not have descriptions among our collected 1,341,790 PRs.

Another common case is that PRs are displayed in a list view by default so that only the title and other meta-information (e.g., author name, tags, and PR id) are available. In the absence of a PR description, a high-quality title becomes more important for readers to understand the intention of a PR.

Dataset

PRTiger dataset

- Dataset for automatic pull request title generation.

- First dataset that can be leveraged for PR title generation.

- 43,816 Pull Request.

- 495 Repositories

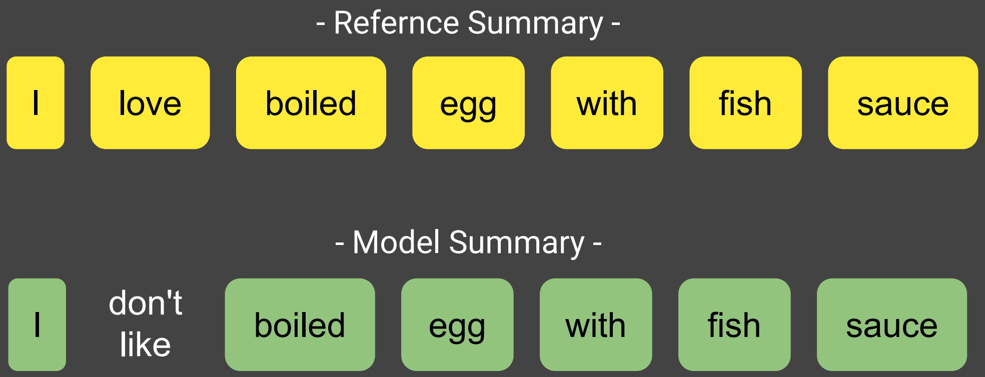
Evaluation

ROUGE

- Evaluate the quality of text summarization systems.

- Measures the overlap between the words from a machine-generated and reference/ human-written.

ROUGE-1



$$\text{Recall} = \frac{\text{Number of word matches}}{\text{Number of words in reference}} = \frac{6}{7}$$

$$\text{Precision} = \frac{\text{Number of word matches}}{\text{Number of words generated}} = \frac{6}{8}$$

$$\text{ROUGE-1} = 2\left(\frac{\text{Precision x Recall}}{\text{Precision} + \text{Recall}}\right) = 0.8$$

ROUGE-2

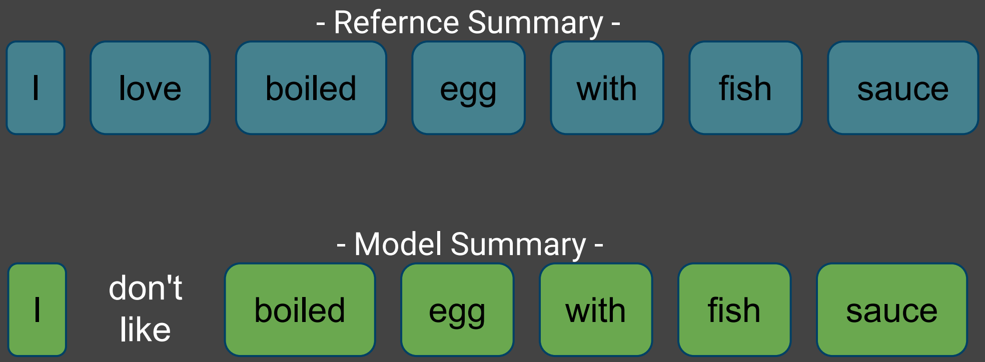
- Refernce Summary -	- Model Summary -
I love	I don't
love boiled	don't like
boiled egg	like boiled
egg with	boiled egg
with fish	egg with
fish sauce	with fish
	fish sauce

$$\text{Recall} = \frac{\text{Number of bigram matches}}{\text{Number of bigrams in reference}} = \frac{4}{6}$$

$$\text{Precision} = \frac{\text{Number of bigram matches}}{\text{Number of bigram generated}} = \frac{4}{7}$$

$$\text{ROUGE-1} = 2(\frac{\text{Precision x Recall}}{\text{Precision} + \text{Recall}}) = 0.615$$

ROUGE-L



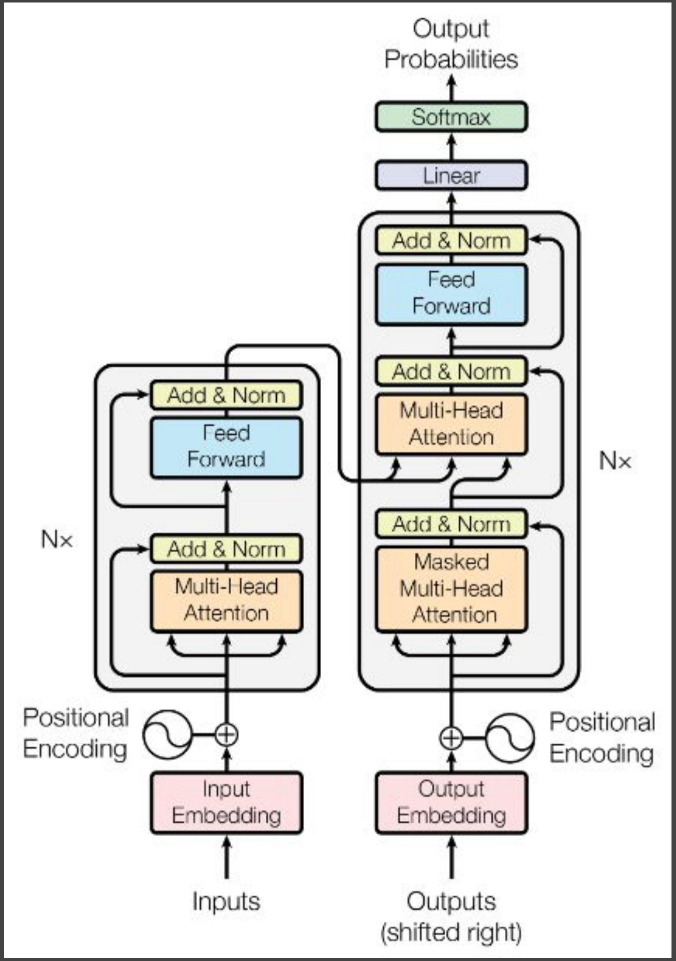
$$\text{Recall} = \frac{\text{LCS(gen, ref)}}{\text{Number of words in reference}} = \frac{6}{7}$$

$$\text{Precision} = \frac{\text{LCS(gen, ref)}}{\text{Number of words generated}} = \frac{6}{8}$$

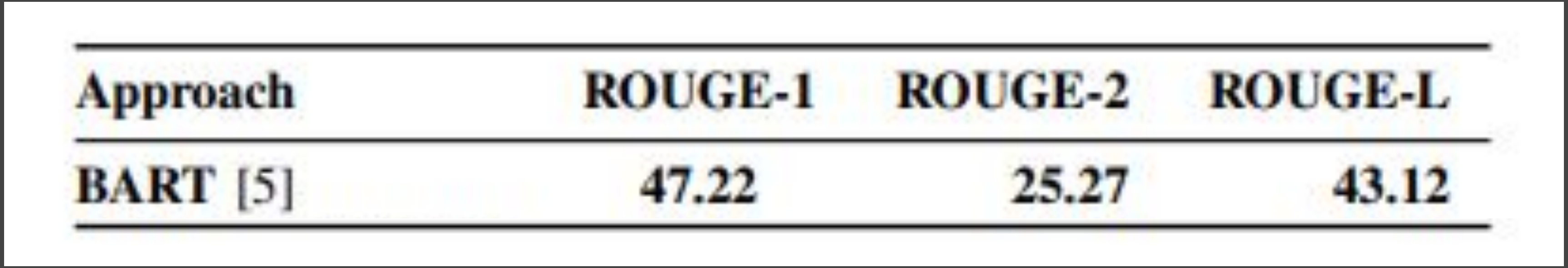
$$\text{ROUGE-1} = 2(\frac{\text{Precision x Recall}}{\text{Precision} + \text{Recall}}) = 0.8$$

Baseline Model

- BART(facebook/bart-base)



Baseline: paper



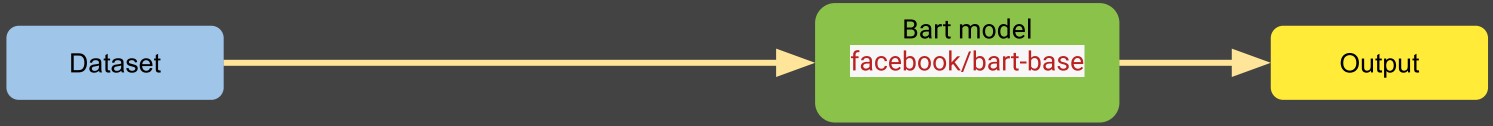
Approach	ROUGE-1	ROUGE-2	ROUGE-L
BART [5]	47.22	25.27	43.12

Our Approach

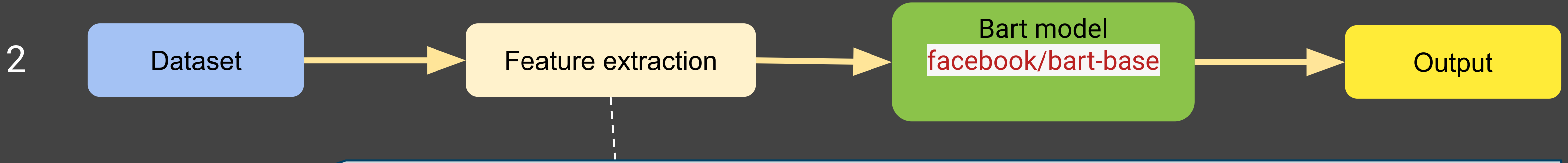
Approachs

From paper

1



Approaches



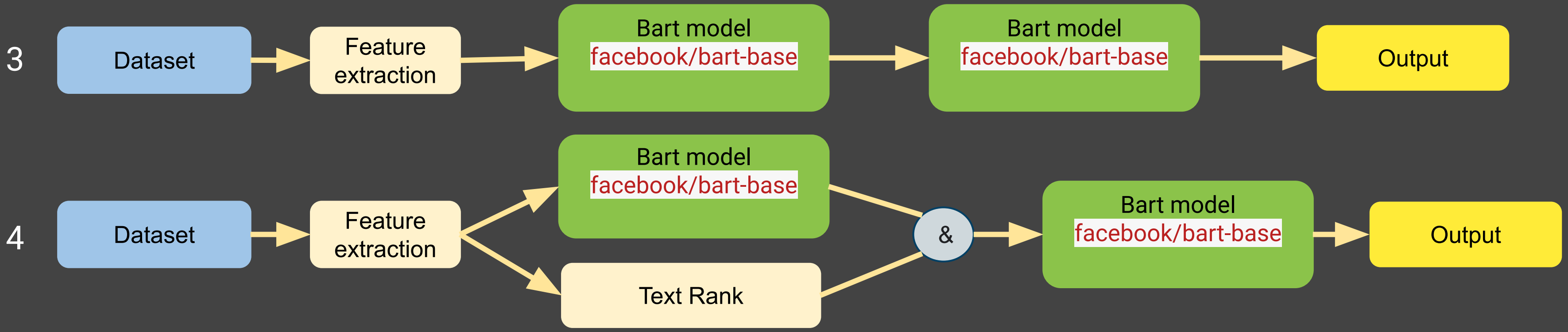
Before

to run, try this: sentry tsdb query organizations --since "10 am pst" --until "12 pm pst" organization_total_received organization_total_rejected right now this is limited to organization statistics. this could fairly easily support projects or other entities with a bit of cleanup, as well as different aggregation methods. @getsentry/infrastructure @mattrobenolt   sort runner commands.   add tsdb query organizations command.

After

organization_total_received,organization_total_rejected. to run, try this: sentry tsdb query organizations --since "10 am pst" --until "12 pm pst" organization_total_received organization_total_rejected right now this is limited to organization statistics. this could fairly easily support projects or other entities with a bit of cleanup, as well as different aggregation methods. @getsentry/infrastructure @mattrobenolt   sort runner commands.   add tsdb query organizations command.

Approaches


3
Dataset
Feature extraction
Bart model facebook/bart-base
Bart model facebook/bart-base
Output
4
Dataset
Feature extraction
Bart model facebook/bart-base
Text Rank
&
Bart model facebook/bart-base
Output

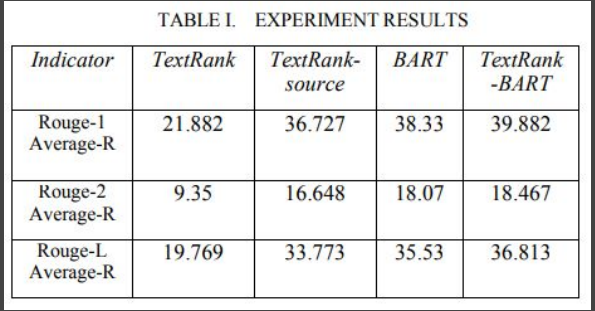


TABLE I. EXPERIMENT RESULTS

Indicator	*TextRank*	*TextRank-source*	*BART*	*TextRank-BART*
Rouge-1 Average-R	21.882	36.727	38.33	39.882
Rouge-2 Average-R	9.35	16.648	18.07	18.467
Rouge-L Average-R	19.769	33.773	35.53	36.813

News Text Summarization Method based on BART-TextRank Model

Reference: https://ieeexplore.ieee.org/document/9390683

TextRank

- Graph-based ranking algorithms
- is to extract the **most representative sentences** of the article as the guiding clue

Results

Results - facebook

Model	ROUGE-1	ROUGE-2	ROUGE-L
BART	46.8684	25.0862	42.9303
BART+ preprocessing	46.5925	24.7061	42.7126
BART+ preprocessing+ BART	45.3242	23.6018	41.5316
BART+ preprocessing+ TextRank + BART	46.0233	24.4411	42.3475

Results - distril

Model	ROUGE-1	ROUGE-2	ROUGE-L
BART	47.0203	24.9569	42.7383
BART+ preprocessing	46.9456	24.9711	42.7546
BART+ preprocessing+ BART	46.6103	24.5855	42.4746
BART+ preprocessing+ TextRank	46.9729	24.966	42.7417

Sample results: Bart-TextRank

Generated result	Target result
fix relative paths in tests (attempt 1)	fix relative paths in tests, part 1
disallow null being a valid parameter in get_class()	get_class() disallow null parameter rfc
small code optimizations and clean up the build.	fixed n_support_ attr for oneclasssvm and svr
resolves checkstyle errors for api-gateway lazy-loading leader-election	resolves checkstyle errors for api-gateway, lazy-loading, leader-election

Results (Bart-TextRank)

1. Source: this pr has enhancements to the reporter (in gatsby cli) and "local reporter" (in gatsby). the reporter class has added errormap -> plugins can register new structured errors in onpreinit by calling reporter.seterrormap pass an error map to the reporter.error pass an error map to the reporter.error

Generated output: pass error map to the reporter class

2. **Source:** e-commerce. adding a reference guide for working with square to help shore up the e-commerce documentation. i've gone back and forth on this a bit, trying to decide how far to take it. having gone through the square documentation, it seems to me like the existing plugin doesn't correctly place the required reference to square's library. i feel like it's probably worth creating a starter to make this easier for people but also that something like that goes beyond

Generated output: add reference guide for working with square

Sample Result

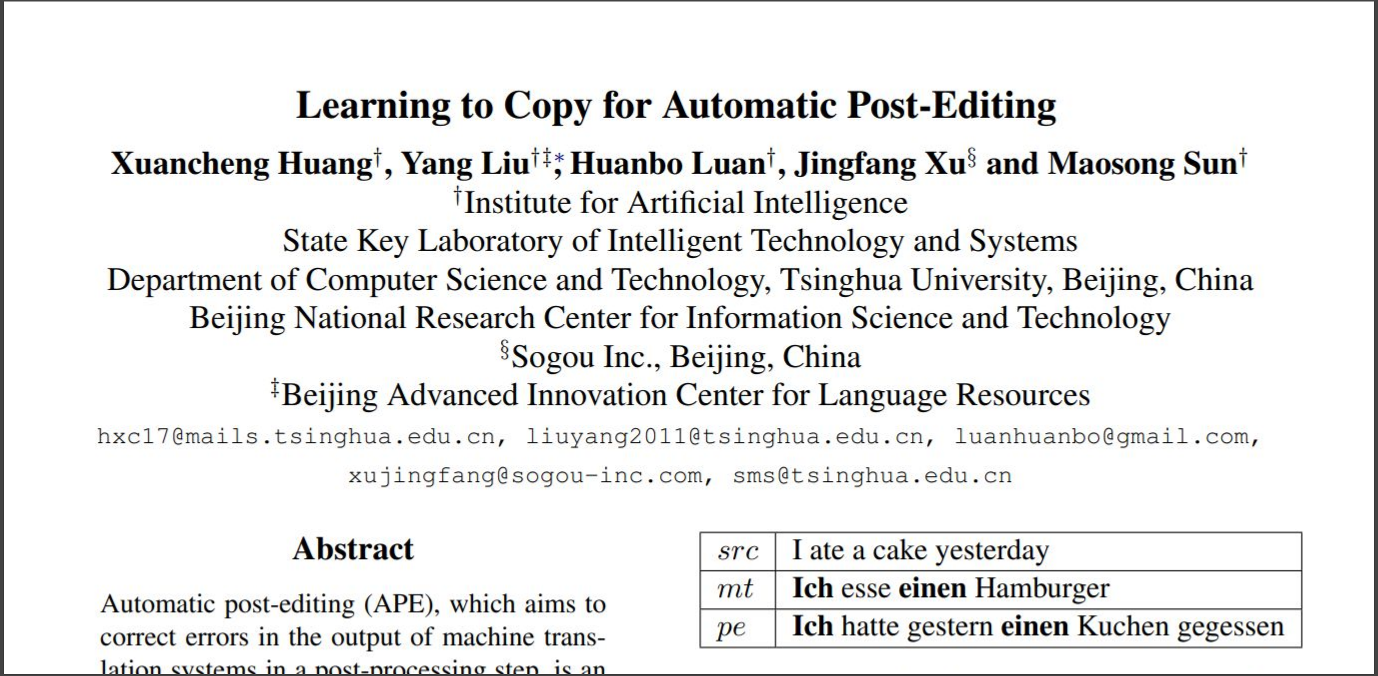
Source : type-checking,type-check. 20e6ff2: fix #24059, fix #25034 fix #22462, fix #22556, fix #21996 (all are duplicates of the same issue). ced9fca: does not fix any particular issue, but it's related to #21461 30f02dd: fix #24022 gdscript: fix type-check of indexed values   gdscript: clarify error message about cycles   they may happen with any cyclic dependency, not only with inheritance.   gdscript compiler: check if subclass exists before comparison otherwise these checks might trigger the insertion of an empty value,   leading to crashes.   gdscript parser has issues with static typing outer class in inner class.   if a member of an inner class has the same name as a variable in the outer class, the type of the outer variable is used for type-checking   optional typing conflict if a class and one of its member variables share a common named attribute with different expected types   crash when downcasting to an inner class type within an inner class   variable type is overwritten by a local variable   assignment bug with typed gdscript if class variables has same name as node members

Generate After Bart Ouput

Generated Output: fix type-check of indexed values in inner class

Future Plan

Future Plan



Learning to Copy for Automatic Post-Editing

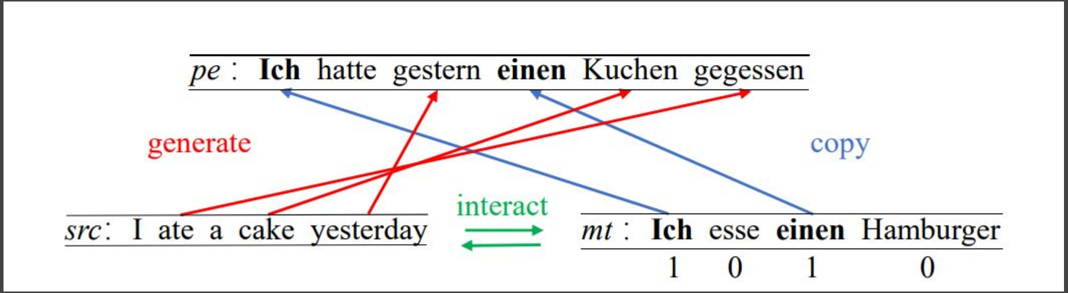
Xuancheng Huang[†], **Yang Liu**[†‡*], **Huanbo Luan**[†], **Jingfang Xu**[§] **and Maosong Sun**[†]
[†]Institute for Artificial Intelligence
State Key Laboratory of Intelligent Technology and Systems
Department of Computer Science and Technology, Tsinghua University, Beijing, China
Beijing National Research Center for Information Science and Technology
[§]Sogou Inc., Beijing, China
[‡]Beijing Advanced Innovation Center for Language Resources
hxc17@mails.tsinghua.edu.cn, liuyang2011@tsinghua.edu.cn, luanhuanbo@gmail.com,
xujingfang@sogou-inc.com, sms@tsinghua.edu.cn

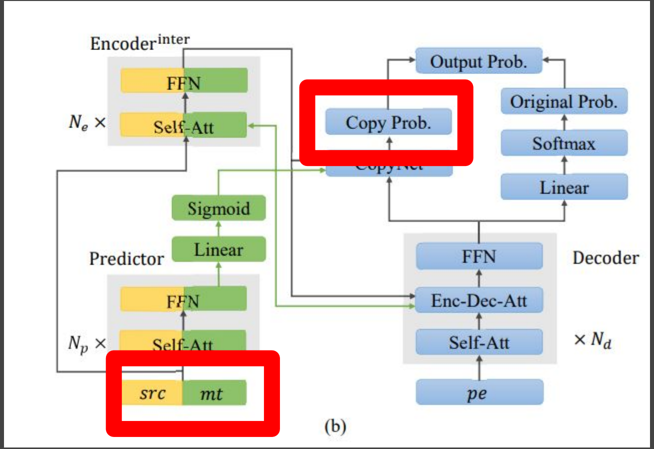
Abstract

Automatic post-editing (APE), which aims to correct errors in the output of machine translation systems in a post-processing step, is an

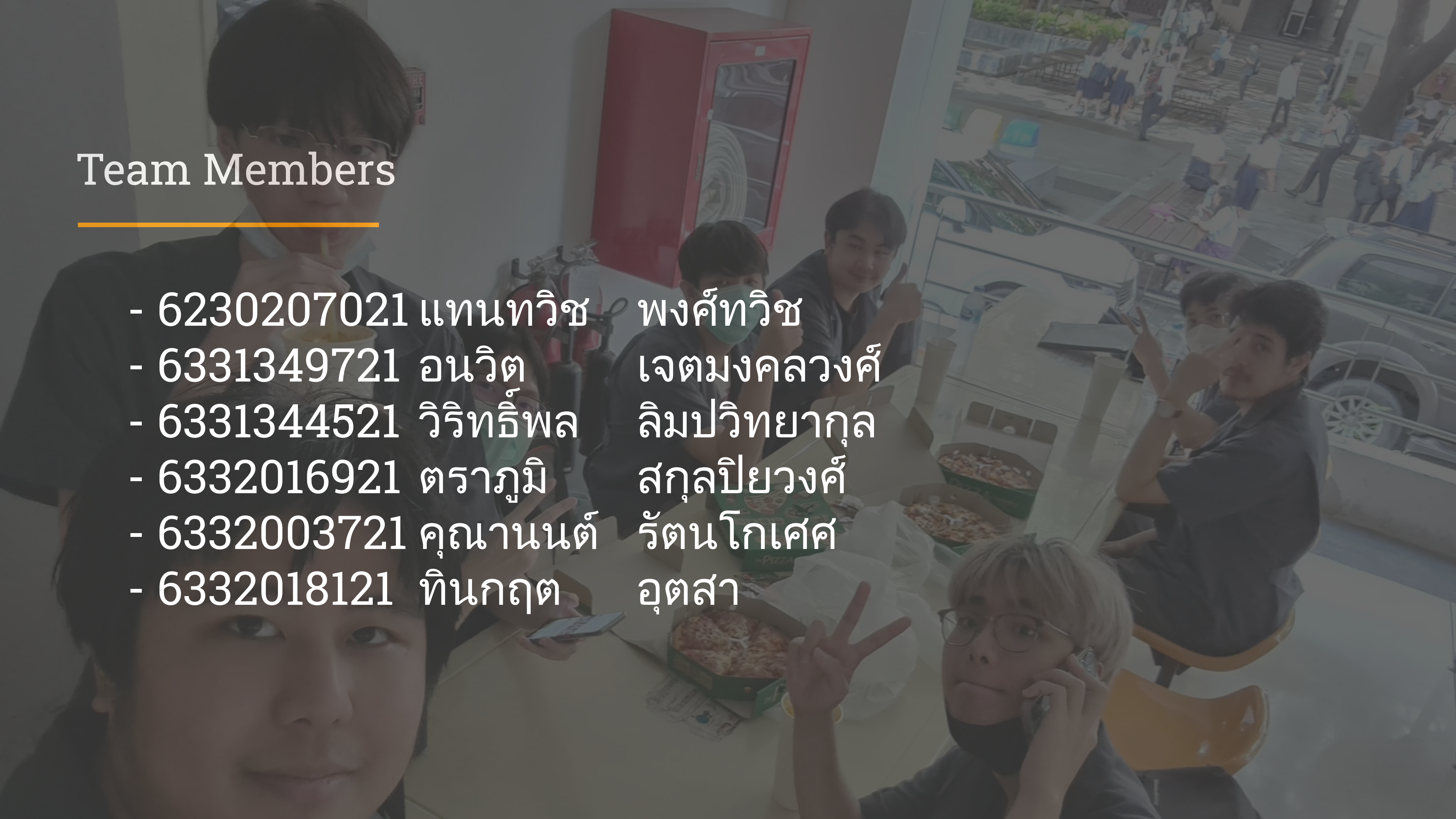
src	I ate a cake yesterday
mt	**Ich** esse **einen** Hamburger
pe	**Ich** hatte gestern **einen** Kuchen gegessen

Future Plan


pe : Ich hatte gestern einen Kuchen gegessen
generate
copy
interact
src : I ate a cake yesterday
mt : Ich esse einen Hamburger
1 0 1 0



(b)



Team Members

- 6230207021 แทนทวิช พงศ์ทวิช
- 6331349721 อนวิต เจตมงคลวงศ์
- 6331344521 วิริทธิ์พล ลิมปวิทยากุล
- 6332016921 ตราภูมิ สกุลปิยวงศ์
- 6332003721 คุณานนต์ รัตนโกเศศ
- 6332018121 ทินกฤต อุตสา