Installation

We have uploaded `NLQF` to Python Package Index (PyPi) and you can install it by the following command:

```
pip install nlqf
```

The source code of `NLQF` is available at https://github.com/v587su/NLQF.
You can also download the source code from the above link and install it directly with the following command:

```
pip install -r requirements.txt
python setup.py install
```

Besides, `nlqf` should be used with GPU-enabled Pytorch. You can refer to https://pytorch.org/get-started/locally/ to install Pytorch. Pytorch 1.3 is recommended.

Usage example

To test the whether the installation is successful, you can run the following script:

It is noteworthy that, to use the `model_filter`, you need to download the pre-trained model `./resource/vae.model` and the word vocabulary `./resource/word_vocab.json` form https://github.com/v587su/NLQF.

```python
import nlqf
import torch
import json

vocab_path = './word_vocab.json'
model_path = './vae.model'
with open(vocab_path, 'r') as f:
    word_vocab = json.load(f)
model = torch.load(model_path)

raw_comments = ['======','create a remote conection','return @test','convert int to string']

# Select the rules to be applied from the following default ruleset:
# 'contain_any_url': remove comments containing any url
# 'contain_any_javadoc_tag': remove comments containing any javadoc tag
# 'contain_any_non_English': remove comments containing any non-English words
# 'not_contain_any_letter': remove comments not containing any letter
# 'less_than_three_words': remove comments with less than three words
# 'end_with_question_mark': remove comments ending with question mark
# 'contain_html_tag': remove comments containing html tag
# 'detach_brackets': detach the content in brackets from the comment
# 'detach_html_tag': detach the content in html tag from the comment
rule_list = ['contain_any_javadoc_tag','not_contain_any_letter']

# Define your own rule using functions.
# Return False if you want to discard the comment string.
# Return True if you want to retain the comment string.
# Return String if you want to replace the comment string with your String.
def my_rule1(comment_str):
    return len(comment_str) < 10

def my_rule2(comment_str):
    if comment_str.startswith('Return'):
        return comment_str.replace('Return', '')
    else:
        return True

# The key should start with 'detach' if your rule is for detachable content.
# Otherwise, name the key as you like.
my_rule_dict = {
    'length_less_than_10':my_rule1,
    'detach_return':my_rule2
}

comments,idx = nlqf.rule_filter(raw_comments, \
        selected_rules=rule_list,defined_rule_dict=my_rule_dict)
print(comments,idx)
# ['create a remote conection', 'convert int to string'] [1, 3]

comments,idx = nlqf.model_filter(raw_comments, word_vocab, model, \
        with_cuda=True, query_len=20,num_workers=1, max_iter=1000)
print(comments,idx)
# ['create a remote conection', 'convert int to string'] [1 3]
```

If both `model_filter` and `rule_filter` print valid outputs, the installation is successful.