IT-314 Software Engineering
Lab-5

Name: Kushal Soni
St. Id: 202001058

Static Analysis Tools:

- ### Pylint:

  Repo link for used code :https://github.com/togethercomputer/OpenChatKit/blob/main/data/OIG-moderation/prepare.py#L1

  #### Example 1:

```
43      checkpoint = torch.load(os.path.join(input_path, f'prank_{i}_checkpoint.pt'), map_location=torc
44
45      if i == 0:
46          _tmp = {k[len(f"{0}."):]:v for k,v in checkpoint.items() if k.startswith(f"0.")}
47          # torch.save(_tmp, os.path.join(output_path, f'pytorch_embs.pt'))
48          model.gpt_neox.embed_in.weight.data[:] = _tmp['embed_in.weight']
49
50          for j in range(n_layer_per_stage):
51              _tmp = {k[len(f"{j+1}."):]:v for k,v in checkpoint.items() if k.startswith(f"{j+1}.")}
52              if len(_tmp) == 0:
53                  break
54              # torch.save(_tmp, os.path.join(output_path, f'pytorch_{j}.pt'))
55              model.gpt_neox.layers[j].load_state_dict(_tmp)
56
57      elif i == n_stages - 1:
58          for j in range(n_layer_per_stage):
59              if i*n_layer_per_stage + j == 44:
60                  break
61              _tmp = {k[len(f"{j}."):]:v for k,v in checkpoint.items() if k.startswith(f"{j}.")}
62              if len(_tmp) == 0:
63                  break
```

Error:

```
PROBLEMS 7   OUTPUT   DEBUG CONSOLE   TERMINAL   JUPYTER

new.py:51:0: C0301: Line too long (102/100) (line-too-long)
new.py:64:0: C0301: Line too long (102/100) (line-too-long)
new.py:82:0: C0301: Line too long (102/100) (line-too-long)
```

Example 2:

```
85      return model
86
87
88  if __name__ == '__main__':
89
90      parser = argparse.ArgumentParser(description='Convert HF checkpoints')
91      parser.add_argument('--ckpt-path', type=str, default=None,
92                          help='model-name')
93      parser.add_argument('--save-path', type=str, default=None,
94                          help='model-name')
95      parser.add_argument('--n-stages', type=int, default=8,
96                          help='pipeline group size')
```

Error :

```
new.py:89:0: C0303: Trailing whitespace (trailing-whitespace)
new.py:91:62: C0303: Trailing whitespace (trailing-whitespace)
new.py:93:62: C0303: Trailing whitespace (trailing-whitespace)
new.py:95:58: C0303: Trailing whitespace (trailing-whitespace)
new.py:97:67: C0303: Trailing whitespace (trailing-whitespace)
new.py:100:0: C0303: Trailing whitespace (trailing-whitespace)
new.py:103:0: C0303: Trailing whitespace (trailing-whitespace)
new.py:113:0: C0303: Trailing whitespace (trailing-whitespace)
```

Example 3:

```
110     load_decentralized_checkpoint(
111         model, args.ckpt_path, n_stages=args.n_stages, n_layer_per_stage=args.n_layer_per_stage,
112     )
113
114     model.save_pretrained(args.save_path)
115     config.save_pretrained(args.save_path)
116     tokenizer.save_pretrained(args.save_path)
```

Error:

```
new.py:113:0: C0303: Trailing whitespace (trailing-whitespace)
new.py:116:0: C0304: Final newline missing (missing-final-newline)
```

Example 4:

new.py 7

C: > Users > student > Downloads > new.py > ...

```python
import torch
import torch.nn as nn

import argparse

from transformers import GPTNeoXForCausalLM

from transformers import AutoConfig, AutoTokenizer

from transformers.modeling_utils import no_init_weights
import os


def create_empty_gptneox(config):

```

Error:

PROBLEMS 7 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

```
new.py:1:0: C0114: Missing module docstring (missing-module-docstring)
new.py:1:0: E0401: Unable to import 'torch' (import-error)
new.py:2:0: R0402: Use 'from torch import nn' instead (consider-using-from-import)
new.py:2:0: E0401: Unable to import 'torch.nn' (import-error)
new.py:6:0: E0401: Unable to import 'transformers' (import-error)
new.py:8:0: E0401: Unable to import 'transformers' (import-error)
new.py:10:0: E0401: Unable to import 'transformers.modeling_utils' (import-error)
```

Example 5:

```python

def create_empty_gptneox(config):

    import torch
    import torch.nn as nn

    _reset_parameters_linear = nn.Linear.reset_parameters
    def dummy(*args, **kargs):
        pass
    nn.Linear.reset_parameters = dummy

    # 1. disable init for faster initialization
    # 2. avoid tie token embeddings with lm_head, as we train them separately.
    with no_init_weights(_enable=True):
        model = GPTNeoXForCausalLM(config).eval()

    nn.Linear.reset_parameters = _reset_parameters_linear
```

Error:

```
new.py:14:0: C0116: Missing function or method docstring (missing-function-docstring)
new.py:14:25: W0621: Redefining name 'config' from outer scope (line 107) (redefined-outer-name)
new.py:16:4: W0621: Redefining name 'torch' from outer scope (line 1) (redefined-outer-name)
new.py:17:4: W0621: Redefining name 'nn' from outer scope (line 2) (redefined-outer-name)
new.py:27:8: W0621: Redefining name 'model' from outer scope (line 109) (redefined-outer-name)
```

Example 6:

```
16      import torch
17      import torch.nn as nn
18
19      _reset_parameters_linear = nn.Linear.reset_parameters
20      def dummy(*args, **kargs):
21          pas=10
22      nn.Linear.reset_parameters = dummy
23
24      # 1. disable init for faster initialization
25      # 2. avoid tie token embeddings with lm_head, as we train them separately.
26      with no_init_weights(_enable=True):
27          model = GPTNeoXForCausalLM(config).eval()
28
```

Error:

```
new.py:20:0: W0613: Unused argument 'args' (unused-argument)
new.py:20:0: W0613: Unused argument 'kargs' (unused-argument)
new.py:21:8: W0612: Unused variable 'pas' (unused-variable)
new.py:16:4: W0611: Unused import torch (unused-import)
```

Example 7:

```
28
29      nn.Linear.reset_parameters = _reset_parameters_linear
30
31      return model
32
33  > def load_decentralized_checkpoint(model, checkpoint_path, n_stages=2, n_layer_per_stage=14):
86
87
88  if __name__ == '__main__':
89
90      parser = argparse.ArgumentParser(description='Convert HF checkpoints')
91      parser.add_argument('--ckpt-path', type=str, default=None,
```

Error:

```
new.py:33:0: R0912: Too many branches (13/12) (too-many-branches)
```

Types of errors identified by Pylint:

- Line-too-long
- Trailing-whitespace
- Missing-final-newline
- Import-error
- Redefined-outer-name
- Unused variable
- too-many-branches

False positive errors:

From these above errors the ‘too-many-branches’ error is false positive as only pylint is identifying it as an error and code is also running properly.

Error detection accuracy of pylint is very good as I only found one false positive error where it detected almost 10.