·····ESMFold·····

·····ESMFold·····PDB·····

ESMFold ····· token ·····
····· ESMFold scripts/fold.py ·····

·····

·· FASTA··· ID ····· 128 aa ·····
·····128 ·····

```
from pathlib import Path
from collections import defaultdict
import re

records, name, seq = [], None, []
for line in Path('proteins.fa').read_text().splitlines():
    line = line.strip()
    if not line:
        continue
    if line.startswith('>'):
        if name is not None:
            records.append((name, ''.join(seq)))
        name, seq = line[1:].split()[0], []
    else:
        if name is None:
            raise ValueError('·····FASTA··')
        seq.append(line)
if name is not None:
    records.append((name, ''.join(seq)))
if not records or len({n for n, _ in records}) != len(records):
    raise ValueError('FASTA···ID··')
alphabet = set('ACDEFGHIKLMNPQRSTVWYX')
if any(not s or set(s) - alphabet or not re.fullmatch(r'[A-Za-z0-9][A-Za-z0-9_.-]*', n)
       for n, s in records):
    raise ValueError('·····ID···')
groups = defaultdict(list)
for n, s in records:
    groups[(len(s) - 1) // 128].append((n, s))
out = Path('fold-batches')
out.mkdir(exist_ok=False)
with (out / 'plan.tsv').open('x') as plan:
    for bucket, rows in sorted(groups.items()):
        job = f'len_{bucket * 128 + 1:05d}_{(bucket + 1) * 128:05d}'
        (out / f'{job}.fa').write_text(''.join(f'>{n}\n{s}\n' for n, s in rows))
        plan.write(f'{job}\t{len(rows)}\n')
```

·····

····· CLI · --max-tokens-per-batch ···· GPU ···· token
·····--chunk-size ·····
·····

```
set -euo pipefail
: "${ESM_REPO:?·····/···ESM····}"
: "${TOKENS:?·····token··}"
: "${CHUNK:?·····chunk·}"
test -s fold-batches/plan.tsv
mkdir esmfold-run-001
python -c 'import torch; print(torch.__version__, torch.cuda.is_available())'   > esmfold-run-001/environment.txt
while IFS=$' ' read -r job count; do
  out="esmfold-run-001/$job"
  mkdir "$out"
  /usr/bin/time -v -o "$out/time.txt"     python "$ESM_REPO/scripts/fold.py"       -i "fold-batches/$job.fa" -o "$out/pdb"       --max-tokens-per-batch "$TOKENS"
 --chunk-size "$CHUNK"       > "$out/run.log" 2>&1
  printf 'expected_records=%s\n' "$count" > "$out/expected.txt"
  python -c "from pathlib import Path; import sys; d,f=map(Path,sys.argv[1:]); e={x[1:].split()[0] for x in f.read_text().splitlines() if x.startswith('>')}; a={
x.stem for x in d.glob('*.pdb') if x.stat().st_size>0}; m=e-a; assert not m, 'Missing structures; check OOM log: '+','.join(sorted(m)); print(f'Output check pass
ed: {len(a)} non-empty PDBs')" "$out/pdb" "fold-batches/$job.fa"
done < fold-batches/plan.tsv
```

····· CUDA OOM···· t
oken ··· chunk·····
·····

·····

····· ID ··· PDB ····· ID··· PDB ·····
·····ESMFold ·····
·····

···· | ···· | ···

····· | ·····CUDA··· | ·····

···OOM | ···token···chunk··· | ·····

··ID··· | ···ID····· | ·····

····· | ····· | ·····

····· CUDA OOM ·····ID·····ID·····
·····token·····chunk·····

·····token·chunk·····
·····

···· FASTA·····ESM ·····PyTorch/CUDA ····· ID ·····
·····ESM ·····
·····

····

ESM··fold.py···token·chunk····· https://github.com/facebookresearch/esm/blob/main/scripts/fold.py

ESM····· https://github.com/facebookresearch/esm