preprocessing-data

December 10, 2023

[20]:
```
import re
import os
import glob
```

[21]:
```
scrape_dir = os.path.join('..', 'data-scrapes')
print(scrape_dir)
```

```
..\data-scrapes
```

[22]:
```
import datetime, time
ts = time.time()
st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d-%H%M%S')

print("Converting sequences ... ")
out_file = os.path.join('..', 'data', 'protein-seqs-' + st + '.txt')

print("Writing to: %s" % out_file)
```

```
Converting sequences …
Writing to: ..\data\protein-seqs-2023-12-10-203549.txt
```

[23]:
```
num_proteins_done = 1000    # TODO: Remove (here to reduce complexity)

# All files are read like this:
fasta_files = glob.glob(scrape_dir + "/*.fasta")
print(fasta_files)
```

```
['..\\data-scrapes\\all-human-0001.fasta']
```

[24]:
```
# helper function

def dump_to_file(protein_id, sequence):
    with open(out_file, "a") as f:
        f.write(protein_id + "," + sequence + "\n")
```

[25]:
```
for fname in fasta_files:
    print("Converting: %s: " % fname)
    proteins = {}   # will hold all proteins in this form ->  id: seq

    with open (fname, 'r') as f:
        protein_seq = ''
        protein_id = ''

        for line in f:

            # Match this:   >[two chars]|[alphanumeric chars]|

            match = re.search(r'^>([a-z]{2})\|([A-Z0-9]*)\|', line)
            if match:
                # we matched one of the header lines
                # - that means we're either starting the first protein record
                # - or we're starting ANOTHER one ... in this case, we need to␣
↪write the previous one to a file
                if protein_id != '':
                    dump_to_file(protein_id, protein_seq)


                # to make sure we process only a few points during␣
↪experimentation
                num_proteins_done += 1
                if num_proteins_done > 1000: break   # TODO: Remove


                # starting a new sequence
                protein_id = match.group(2)
                protein_seq = ''

            else:
                # Header line not found. So, we must be seeing the protein␣
↪sequences
                protein_seq += line.strip()



        if protein_id != '':  # we also need the last one dumped
            dump_to_file(protein_id, protein_seq)
```

```
Converting: ..\data-scrapes\all-human-0001.fasta:
```

[26]:
```
# convert function
print("Converting functions ...")
out_file_fns = os.path.join('..', 'data', 'protein-functions-' + st + '.txt')
print(out_file_fns)
target_functions = ['0005524']   # just ATP binding for now
```

```
Converting functions …
..\data\protein-functions-2023-12-10-203549.txt
```

[27]:
```
annot_files = glob.glob(scrape_dir + "/*annotations.txt")
print(annot_files)
```

```
['..\\data-scrapes\\all-human-0001-annotations.txt']
```

[28]:
```
has_function = []  # a dictionary of protein_id: boolean  (which says if the␣
↪protein_id has our target function)

for fname in annot_files:
    with open (fname, 'r') as f:
        for line in f:
            match = re.search(r'([A-Z0-9]*)\sGO:(.*);\sF:.*;', line)
            if match:
                # we got the match correctly (should always happen)
                protein_id = match.group(1)
                function = match.group(2)

                if function not in target_functions:
                        continue

                # We found the function for this protein, so the class will be␣
↪'True'
                has_function.append(protein_id)

    import json
    with open(out_file_fns, 'w') as fp:
        json.dump(has_function, fp)

    # Take a peek
    print(has_function[:10])
```

```
['P27361', 'P53779', 'Q9UHC1', 'Q9NYL2', 'O15440', 'P33527', 'Q92887', 'O15438',
'O15439', 'Q5T3U5']
```

[ ]:
```

```