Carregando_arquivos

May 29, 2024

1 Busca de Índice e Consulta de documento

Autor: Davi J. Leite Santos
Versão: 0.0.3
Data: 25 de Abril de 2024
Localização: Ribeirão das Neves, Minas Gerais - Brasil

1.1 Contato

- **Endereço:** Ribeirão das Neves, Minas Gerais - Brasil
- **Email:** davi.jls@outlook.com
- **LinkedIn:** davi-j-leite-santos
- **Website:** davijls.com.br

1.2 Principais Competências

- **Cibersegurança**
- **Segurança da Informação**
- **Operações de TI**

1.2.1 Reconstruir o documento

Para reconstruir o documento usando o índice invertido.

[ ]:
```
import os
import json
```

2 Processo de carregar os arquivos e guarda-los

Essa parte server para acessar cada documento e fazer o indice invertido de cada um, justamente para armazena-los de alguma forma.

[ ]:
```
# Lendo e processando os arquivos
def process_file(file_path, vocab, index, doc_id):
    if not os.path.exists(file_path):
        print(f"File {file_path} does not exist.")
        return
    with open(file_path, 'r', encoding='utf-8') as f:
        words = f.read().split()
```

```
    for pos, word in enumerate(words):
        word = word.lower().strip(".,!?;")
        if word not in vocab:
            vocab[word] = len(vocab)
        word_id = vocab[word]
        if word_id not in index:
            index[word_id] = {}
        if doc_id not in index[word_id]:
            index[word_id][doc_id] = []
        index[word_id][doc_id].append(pos)

# Salvando os index do indice invertido
def save_index(vocab, index, vocab_file, index_file):
    with open(vocab_file, 'w', encoding='utf-8') as f:
        json.dump(vocab, f)
    with open(index_file, 'w', encoding='utf-8') as f:
        json.dump(index, f)

#Lendo o progresso dos arquivos
def load_progress(progress_file):
    if os.path.exists(progress_file) and os.path.getsize(progress_file) > 0:
        with open(progress_file, 'r', encoding='utf-8') as f:
            progress = json.load(f)
    else:
        progress = {'last_processed': -1}
    return progress

# Salvando o progresso de analise dos arquivos para saber onde parou caso o␣
 ↪codigo de algum bug e pare
def save_progress(progress, progress_file):
    with open(progress_file, 'w', encoding='utf-8') as f:
        json.dump(progress, f)

# Criando o indice invertido
def create_inverted_index(files_dir, index_dir, vocab_dir, progress_file):
    # Dando Load no progresso
    progress = load_progress(progress_file)

    # Para ler cada arquivo com base no progresso
    files = sorted(os.listdir(files_dir))
    for i, file_name in enumerate(files):
        if i <= progress['last_processed']:
            continue
        file_path = os.path.join(files_dir, file_name)

        # Criando os indices
        vocab = {}
```

```
        index = {}
        process_file(file_path, vocab, index, 0)

        # Salvando em formato de arquivo
        vocab_file = os.path.join(vocab_dir, f'vocab{i+1}.json')
        index_file = os.path.join(index_dir, f'index{i+1}.json')
        save_index(vocab, index, vocab_file, index_file)

        # Salvando o progresso
        progress['last_processed'] = i
        save_progress(progress, progress_file)

        print(f'Processed {file_name}')
```

[ ]:
```
# Função para juntar todos indices (Vocab e Index)
def merge_indices_and_vocabs(index_dir, vocab_dir, index_geral_file,␣
 ↪vocab_geral_file):
    geral_index = {}
    geral_vocab = {}
    current_word_id = 0

    # Organizando os arquivos
    index_files = sorted(os.listdir(index_dir))
    vocab_files = sorted(os.listdir(vocab_dir))

    # Carregando esses arquivos
    for index_file, vocab_file in zip(index_files, vocab_files):
        with open(os.path.join(index_dir, index_file), 'r', encoding='utf-8')␣
 ↪as f:
            index = json.load(f)
        with open(os.path.join(vocab_dir, vocab_file), 'r', encoding='utf-8')␣
 ↪as f:
            vocab = json.load(f)

        # Criando o indice invertido em apenas um arquivo
        for word, word_id in vocab.items():
            if word not in geral_vocab:
                geral_vocab[word] = current_word_id
                current_word_id += 1
            geral_word_id = geral_vocab[word]

            if geral_word_id not in geral_index:
                geral_index[geral_word_id] = {}
            for doc_id, positions in index[str(word_id)].items():
                doc_id = int(doc_id)
                if doc_id not in geral_index[geral_word_id]:
                    geral_index[geral_word_id][doc_id] = []
```

```
                geral_index[geral_word_id][doc_id].extend(positions)

    # salvando esse indice maior em dois arquivos diferentes
    with open(vocab_geral_file, 'w', encoding='utf-8') as f:
        json.dump(geral_vocab, f)
    with open(index_geral_file, 'w', encoding='utf-8') as f:
        json.dump(geral_index, f)
```

[ ]:
```
# Defina os caminhos dos diretórios e arquivos
files_dir = '../Motor_de_busca-WebCrawler/sites_visitados'
index_dir = 'indexs'
vocab_dir = 'vocabs'
progress_file = 'progress.json'
```

[ ]:
```
# Crie os diretórios se não existirem
os.makedirs(index_dir, exist_ok=True)
os.makedirs(vocab_dir, exist_ok=True)
```

[ ]:
```
# Crie o índice invertido
#create_inverted_index(files_dir, index_dir, vocab_dir, progress_file)
```

[ ]:
```
# Defina os caminhos dos arquivos gerais
index_geral_file = 'index_geral.json'
vocab_geral_file = 'vocab_geral.json'
```

[ ]:
```
# Mesclar os índices e vocabulários individuais nos arquivos gerais
#merge_indices_and_vocabs(index_dir, vocab_dir, index_geral_file,␣
 ↪vocab_geral_file)
```

3 Fazendo as leituras das operações

[ ]:
```
def convert_to_list_format(index_geral_file, list_geral_file):
    with open(index_geral_file, 'r', encoding='utf-8') as f:
        index = json.load(f)

    list_format = []
    for word_id, docs in index.items():
        for doc_id, positions in docs.items():
            list_format.append({
                'word_id': int(word_id),
                'doc_id': doc_id,
                'positions': positions
            })

    with open(list_geral_file, 'w', encoding='utf-8') as f:
        json.dump(list_format, f)
```

[ ]:
```
# Defina o caminho do arquivo de lista geral
list_geral_file = 'list_geral.json'
```

[ ]:
```
# Converta o índice geral para o formato de lista
convert_to_list_format(index_geral_file, list_geral_file)
```

[ ]:
```
def load_geral_files(index_geral_file, vocab_geral_file, list_geral_file):
    with open(index_geral_file, 'r', encoding='utf-8') as f:
        index_geral = json.load(f)
    with open(vocab_geral_file, 'r', encoding='utf-8') as f:
        vocab_geral = json.load(f)
    with open(list_geral_file, 'r', encoding='utf-8') as f:
        list_geral = json.load(f)

    return index_geral, vocab_geral, list_geral
```

[ ]:
```
# Carregar os arquivos gerais
index_geral, vocab_geral, list_geral = load_geral_files(index_geral_file,␣
 ↪vocab_geral_file, list_geral_file)
```

[ ]:
```
def convert_to_tuple_format(index_geral_file, tuple_format_file):
    with open(index_geral_file, 'r', encoding='utf-8') as f:
        index = json.load(f)

    tuple_format = []
    for word_id, docs in index.items():
        for doc_id, positions in docs.items():
            tuple_format.append((int(word_id), int(doc_id), positions))

    with open(tuple_format_file, 'w', encoding='utf-8') as f:
        json.dump(tuple_format, f)
```

[ ]:
```
# Defina o caminho do arquivo de formato de tupla
tuple_format_file = 'tuple_format.json'
```

[ ]:
```
# Converta o índice geral para o formato de tupla
convert_to_tuple_format(index_geral_file, tuple_format_file)
```

[ ]:
```
def load_all_files2(index_geral_file, vocab_geral_file, tuple_format_file):
    with open(index_geral_file, 'r', encoding='utf-8') as f:
        index_geral = json.load(f)
    with open(vocab_geral_file, 'r', encoding='utf-8') as f:
        vocab_geral = json.load(f)
    with open(tuple_format_file, 'r', encoding='utf-8') as f:
        tuple_format = json.load(f)

    return index_geral, vocab_geral, tuple_format
```

[ ]:
```
# Carregar os arquivos gerais e o arquivo de formato de tupla
index_geral, vocab_geral, tuple_format = load_all_files2(index_geral_file,␣
↪vocab_geral_file, tuple_format_file)
```

4 Variaveis

5 Exemplo de como usar os dados carregados

#print("Índice Geral:", index_geral) #print("Vocabulário Geral:", vocab_geral) #print("Lista Geral:", list_geral) #print("Formato de Tupla:", tuple_format)