AI VIET NAM – COURSE 2023

Visual Question Answering - Project

Ngày 4 tháng 12 năm 2023

Phần I: Giới thiệu

Visual Question Answering (VQA) là một bài toán phổ biến trong học máy, ứng dụng các kĩ thuật liên quan từ cả thị giác máy tính đến xử lý ngôn ngữ tự nhiên. Khái niệm cốt lõi của bài toán này là phân tích một hình ảnh và trả lời câu hỏi về hình ảnh đó. Các hệ thống VQA được sử dụng trong nhiều ứng dụng khác nhau, bao gồm hỗ trợ người dùng khiếm thị hiểu rõ môi trường xung quanh họ, công cụ giáo dục tương tác và tự động hóa quản lý hoặc điều tiết hình ảnh. Một chương trình VQA thường bao gồm 2 bước. Bước đầu là phân tính thông tin đầu vào, bao gồm sử dụng các kĩ thuật xử lý hình ảnh và xử lý câu hỏi đặt ra bằng ngôn ngữ tự nhiên. Sau đó hệ thống kết hợp thông tin thu được từ phân tích hình ảnh và ngữ cảnh của câu hỏi để tạo ra một câu trả lời phù hợp. Vì vậy, một chương trình có độ chính xác cao cần xây dựng tốt cả hai thành phần này.

Trong project này, chúng ta sẽ xây dựng một chương trình VQA sử dụng mô hình CNN cho hình ảnh và LSTM cho xử lý ngôn ngữ. Input và output của chương trình như sau:

- **Input:** Một cặp hình ảnh và câu hỏi bằng ngôn ngữ tự nhiên.
- **Output:** Câu trả lời cho câu hỏi về hình ảnh.

Phần II: Cài đặt chương trình

A. Phần lập trình

- **Visual Question Answering**

1. **Tải bộ dữ liệu:** Cho bộ dữ liệu visual question answering, các bạn tải bộ dữ liệu tại đây.
2. **Import các thư viện cần thiết:** Trong bài tập này, chúng ta sẽ sử dụng thư viện PyTorch để xây dựng và huấn luyện mô hình deep learning. Thêm vào đó, vì làm việc liên quan đến dữ liệu ảnh và text, chúng ta sẽ sử dụng thư viện PIL cho ảnh và spacy, nltk cho text:

```
import torch
import torch.nn as nn
import os
import pandas as pd
import matplotlib.pyplot as plt

from PIL import Image
from torch.utils.data import Dataset, DataLoader
from sklearn.model_selection import train_test_split
import torchtext
from torchtext.data.utils import get_tokenizer
from torchtext.vocab import build_vocab_from_iterator
import spacy

```

3. **Chia bộ dữ liệu train, val, test:** Vì bộ dữ liệu này đã được chia sẵn thành train, val, test, ta chỉ cần đọc dữ liệu lên từ file .txt

```
# Load train data
train_data = []

with open("/content/vaq2.0.TrainImages.txt", "r") as f:
    lines = f.readlines()
    for line in lines:
        temp = line.split('\t')
        qa = temp[1].split('?')
        answer = 0 if qa[1] == ' no\n' else 1
        data_sample = {
            'image_path': temp[0][:-2],
            'question': qa[0] + '?',
            'answer': answer
        }
        train_data.append(data_sample)

# Load val data
val_data = []

with open("/content/vaq2.0.DevImages.txt", "r") as f:
    lines = f.readlines()
    for line in lines:
        temp = line.split('\t')
        qa = temp[1].split('?')
        answer = 0 if qa[1] == ' no\n' else 1
        data_sample = {
            'image_path': temp[0][:-2],
            'question': qa[0] + '?',
            'answer': answer
```

```
        }
        val_data.append(data_sample)

# Load test data
test_data = []

with open("/content/vaq2.0.TestImages.txt", "r") as f:
    lines = f.readlines()
    for line in lines:
        temp = line.split('\t')
        qa = temp[1].split('?')
        answer = 0 if qa[1] == ' no\n' else 1
        data_sample = {
            'image_path': temp[0][:-2],
            'question': qa[0] + '?',
            'answer': answer
        }
        test_data.append(data_sample)


```

4. **Xây dựng bộ từ vựng:** chúng ta cần tiền xử lý text bằng cách biến đổi câu hỏi đầu vào thành các token bằng thư viện spacy và xây dựng bộ từ vựng cho model.

```
eng = spacy.load("en_core_web_sm") # Load the English model to tokenize English text

def get_tokens(data_iter):
    for sample in data_iter:
        question = sample['question']
        yield [token.text for token in eng.tokenizer(question)]


vocab = build_vocab_from_iterator(
    get_tokens(train_data),
    min_freq=2,
    specials= ['<pad>', '<sos>', '<eos>', '<unk>'],
    special_first=True
)
vocab.set_default_index(vocab['<unk>'])

```

5. **Xây dựng hàm tokenize:** Sau khi đã có bộ từ vựng cơ bản cho model, ta sẽ xây dựng một hàm tokenize để biến câu hỏi đầu vào thành những token tương ứng trong bộ từ vựng.

```
def tokenize(question, max_sequence_length):
tokens = [token.text for token in eng.tokenizer(question)]
sequence = [vocab[token] for token in tokens]
if len(sequence) < max_sequence_length:
    sequence += [vocab['<pad>']] * (max_sequence_length - len(sequence))
else:
    sequence = sequence[:max_sequence_length]

return sequence
```

6. **Xây dựng class pytorch dataset:** Chúng ta xây dựng class datasets cho bộ dữ liệu VQA như sau:

```
class VQADataset(Dataset):
def __init__(self, data, max_len=30, transform=None, root_dir='/content/val2014-resised/'):
```

```
    self.transform = transform
    self.data = data
    self.max_len = max_len
    self.root_dir = root_dir

def __len__(self):
    return len(self.data)

def __getitem__(self, index):
    image = Image.open(os.path.join(self.root_dir, self.data[index]['
    image_path'])).convert('RGB')
    if self.transform:
        image = self.transform(image)

    question = self.data[index]['question']
    question = tokenize(question, self.max_len)

    label = self.data[index]['answer']
    return image, torch.LongTensor(question), label
```

7. **Xây dựng hàm tiền xử lý ảnh (transforms):** Để dữ liệu ảnh đầu vào được đồng bộ về kích thước cũng như đơn giản hóa, chúng ta sẽ xây dựng hàm tiền xử lý ảnh như sau:

```
transform = T.Compose([
    T.Resize((224, 224)),
    T.ToTensor(),
    T.Normalize((0.485, 0.456, 0.406), (0.229, 0.224, 0.225)),
])


```

Các kỹ thuật được áp dụng: resize ảnh, đổi về tensor và chuẩn hóa giá trị pixel.

8. **Khai báo datasets object cho ba bộ train, val, test:**

```
train_dataset = VQADataset(
    train_data,
    transform=transform
)
val_dataset = VQADataset(
    val_data,
    transform=transform
)
test_dataset = VQADataset(
    test_data,
    transform=transform
)
```

9. **Khai báo dataloader:** Với ba object datasets trên, ta khai báo giá trị batch size và tạo dataloader như sau:

```
train_batch_size = 64
test_batch_size = 128

train_loader = DataLoader(
    train_dataset,
    batch_size=train_batch_size,
    shuffle=True
)
val_loader = DataLoader(
    val_dataset,
    batch_size=test_batch_size,
```

```
    shuffle=False
)
test_loader = DataLoader(
    test_dataset,
    batch_size=test_batch_size,
    shuffle=False
)
```

10. **Xây dựng model:** Trong phần này, ta sẽ dừng kiến trúc Resnet cho phần xử lý ảnh và kiến trúc LSTM cho phần xử lý text. Ta sẽ dùng thư viện timm để load kiến trúc ResNet50. Code của class model sẽ như sau:

```
import timm

class VQAModel(nn.Module):
    def __init__(self, image_model='resnet50', embed_dim=300, hidden_dim=128):
        super(VQAModel, self).__init__()
        self.image_encoder = timm.create_model(image_model, pretrained=True, num_classes=hidden_dim)

        self.embedding = nn.Embedding(len(vocab), embed_dim)
        self.lstm = nn.LSTM(input_size=embed_dim,
                            hidden_size=hidden_dim,
                            batch_first=True,
                            bidirectional=True)

        self.fc1 = nn.Linear(hidden_dim*3, 256)
        self.relu = nn.ReLU()
        self.dropout = nn.Dropout(0.1)
        self.fc2 = nn.Linear(256, 1)
        self.sigmoid = nn.Sigmoid()

    def forward(self, image, text):
        image_features = self.image_encoder(image)

        text_emb = self.embedding(text)
        lstm_out, _ = self.lstm(text_emb)

        lstm_out = torch.mean(lstm_out, 1)

        # Concatenate image and text features
        combined = torch.cat((image_features, lstm_out), dim=1)

        # Fully connected layers
        x = self.fc1(combined)
        x = self.dropout(self.relu(x))
        x = self.fc2(x)
        x = self.sigmoid(x)
        return x

```

Phần III: Câu hỏi trắc nghiệm

1. Mục tiêu của bài toán Visual Question Answering là gì?
   (*a*) Để tăng kích cỡ của tấm ảnh.
   (*b*) Để trả lời câu hỏi dự trên hình ảnh.
   (*c*) Để tạo sinh ra hình ảnh dựa trên câu mô tả.
   (*d*) Để tạo ra mô hình 3D từ hình ảnh 2D.

2. Trong VQA, mô hình thường nhận đầu vào là gì?
   (*a*) Chỉ là hình ảnh
   (*b*) Chỉ là câu hỏi văn bản
   (*c*) Hình ảnh và câu hỏi văn bản
   (*d*) Âm thanh

3. Visual Question Answering là sự kết hợp giữa 2 lĩnh vực nào sau đây?
   (*a*) Robotics và Xử lý ngôn ngữ tự nhiên.
   (*b*) Thị giác máy tính và học máy.
   (*c*) Thị giác máy tính và xử lý ngôn ngữ tự nhiên.
   (*d*) Học máy và robotics.

4. Trong VQA, câu trả lời có thể là gì?
   (*a*) Chỉ là "có" hoặc "không"
   (*b*) Chỉ là một số nguyên
   (*c*) Có thể là một câu trả lời văn bản hoặc câu trả lời "có" hoặc "không"
   (*d*) Chỉ là một màu sắc

5. Thành phần nào sau đây là một trong những thành phần chính của một mô hình VQA?
   (*a*) Nhận diện giọng nói.
   (*b*) Trích xuất đặc trưng từ hình ảnh.
   (*c*) Dịch câu hỏi ngôn ngữ tự nhiên.
   (*d*) Xử lý âm thanh.

6. Một số thách thức trong VQA bao gồm:
   (*a*) Hiểu biết ngôn ngữ tự nhiên
   (*b*) Nhận dạng đối tượng trong hình ảnh
   (*c*) Không có thách thức nào
   (*d*) Cả a) và b)

7. Một hệ thống VQA thường xử lý câu hỏi ngôn ngữ tự nhiên như thế nào?
   (*a*) Bằng cách chuyển đổi câu hỏi đó thành hình ảnh.
   (*b*) Bằng cách áp dụng sentiment analysis cho tấm ảnh đó.
   (*c*) Dịch câu hỏi đó qua ngôn ngữ khác.

(*d*) Bằng cách trích xuất đặc trưng câu hỏi đó bằng các kĩ thuật NLP.

8. Đoạn code sau đây dùng để làm gì?

```
eng = spacy.load("en_core_web_sm")

def get_tokens(data_iter):
    for sample in data_iter:
        question = sample['question']
        yield [token.text for token in eng.tokenizer(question)]

```

(*a*) Để load mô hình tiếng anh Spacy.

(*b*) Tạo ra một danh sách những từ từ những token.

(*c*) Thêm ký tự đặc biệt vào các câu hỏi.

(*d*) Chia văn bản tiếng Anh thành những token.

9. Mục đích của đoạn code sau là gì?

```
vocab = build_vocab_from_iterator(
    get_tokens(train_data),
    min_freq=2,
    specials= ['<pad>', '<sos>', '<eos>', '<unk>'],
    special_first=True
)
vocab.set_default_index(vocab['<unk>'])

```

(*a*) Xây dựng từ vựng từ tập train, bao gồm các token đặc biệt và token mặc định <unk> cho các từ chưa biết.

(*b*) Dịch tập data huấn luyện sang một ngôn ngữ khác bằng cách sử dụng các token đặc biệt.

(*c*) Sắp xếp các từ trong tập huấn luyện dựa trên tần suất xuất hiện của các từ.

(*d*) Đào tạo một mô hình mới để hiểu các mẫu ngôn ngữ trong tập data huấn luyện.

10. Mô hình học máy nào thường được sử dụng cho bài toán VQA?

(*a*) Mô hình dựa trên luật có sẵn.

(*b*) Các mô hình CNN.

(*c*) Mô hình linear regression.

(*d*) Mô hình Decision Tree.

11. Thách thức trong bài toán VQA là gì?

(*a*) Tăng độ phân giải của hình ảnh

(*b*) Sự mơ hồ trong câu hỏi ngôn ngữ tự nhiên

(*c*) Xử lý âm thanh.

(*d*) Tối ưu hóa phần cứng.

12. Output của mô hình VQA là:

(*a*) Bản báo cáo chi tiết.

(*b*) Con số cụ thể.

(*c*) Câu trả lời cho câu hỏi về hình ảnh.

(*d*) Một đồ thị.

13. Hệ thống VQA được ứng dụng trong việc:
    (*a*) Chơi các video game.
    (*b*) Hỗ trợ người dùng khiếm thị trong việc hiểu môi trường xung quanh.
    (*c*) Lọc email rác.
    (*d*) Xử lý âm thanh.

14. Dòng code sau đây có tác dụng gì?

```
        image_encoder = timm.create_model(image_model, pretrained=True,
    num_classes=hidden_dim))

```

    (*a*) Tải pretrained hình ảnh từ thư viện timm.
    (*b*) Chuyển đổi dữ liệu hình ảnh màu sang hình ảnh xám.
    (*c*) Tăng độ phân giải của hình ảnh.
    (*d*) Nén hình ảnh để xử lý nhanh hơn.

15. Output của image_encoder trong đoạn code sau là gì?

```
        image_encoder = timm.create_model(image_model, pretrained=True,
    num_classes=hidden_dim))

```

    (*a*) Ảnh được biến đổi từ tập dữ liệu.
    (*b*) Một vector embedding với số chiều là hidden_dim
    (*c*) Xác suất tấm ảnh đầu vào thuộc về từng class.
    (*d*) Chiều của tấm ảnh đầu vào.

16. Điều KHÔNG làm ảnh hưởng đến độ chính xác của một mô hình VQA?
    (*a*) Tốc độ Internet.
    (*b*) Chất lượng của tập dữ liệu đầu vào.
    (*c*) Độ nhiễu trong các tấm ảnh đầu vào.
    (*d*) Độ phức tạp trong câu hỏi từ ngôn ngữ tự nhiên.

17. Công nghệ nào là cần thiết để giải thích thành phần văn bản trong VQA?
    (*a*) Blockchain
    (*b*) Xử lý Ngôn Ngữ Tự Nhiên (NLP)
    (*c*) Tính Toán Lượng Tử
    (*d*) Mật mã học

- Hết -