Data Management Final Project
Srikar Boggavarapu
srb271

Github link: https://github.com/srikarboga/datamgmtfinal

## Topic Modeling and Extraction using Clustering and Embeddings

Project Definition:

- This project aims to find complex and meaningful relationships between groups of images within a large dataset of images.
- Why is this a problem?
  - There are a large number of image datasets out there but the information in the datasets would be much more useful if the images were divided into classes. There are some datasets such as the CIFAR-100 that already do this, but many image datasets only provide the images with no distinction between images such as dog images and plane images.
- This project would eliminate this problem by grouping similar images together thus creating a distinction between images in an otherwise unorganized dataset. These meanings could be used to infer relationships between images within the dataset and help identify complex and underlying patterns between similar images. These relationships can be analyzed to find insights that can be used in several ways including for example to improve existing machine learning models that work with images.
- Strategies used:
  - The project achieves this by using techniques such as clustering and uses pre-trained machine learning models such as CLIP to generate embeddings for the images which can then be meaningfully clustered.
  - It uses several techniques learned in class such as data management strategies for cleaning and handling large amounts of data. Using python libraries such as Pandas to store the data in data frames. Using sqlite3 to create a database and store the final results in it. It employs machine learning techniques such as clustering and data modeling and visualization techniques to display the clusters as well.
  - It utilizes a model called BERTopic which is a topic modeling model that can work with CLIP which is what we need to generate and cluster the image embeddings.
- Dataset used:

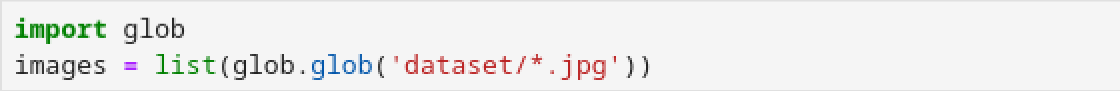
    - The dataset used is the Flickr8k dataset which is a dataset consisting of 8000 images. The dataset is widely available online and I have linked to it in the readme file.

Importance and Novelty:

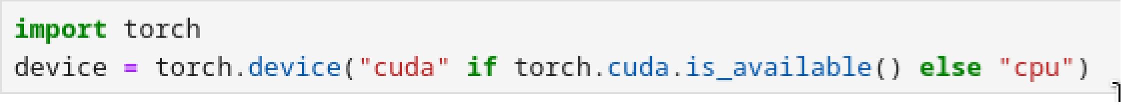
- This project can have a variety of uses and applications in the industry. It can help categorize the vast amounts of unlabeled and unorganized images that are available on the web. It can be used to improve search engines to help categorize the images and help people find what they are looking for easier. It can be used to improve image galleries on people's phones that they use every day, similar to the feature that already exists where images are sorted by faces, they can instead be sorted based on context such as hiking images vs campfire images vs airplane images and so on and so forth.
- I'm excited about this project because I believe it combines my interests and passions. Using a topic model such as BERTopic is something new that I have not done before and this project gave me the skills and confidence and served as an introduction to a whole new world. I am excited that after completing this project I have a new set of tools at my disposal for tackling a problem. There are prior works that have clustered documents but image clustering is very niche and the focus is more on image classification.
- Some existing data management issues with BERTopic are that it does not provide a way to easily analyze the topic clusters that are generated and to store them in a database such that they do not need to be generated every time which is an expensive process.
- This project lies at the intersection of both of these fields, it combines image clustering with image classification and image captioning. BERTopic is the prevailing model in the topic modeling space and this is also the model that this project uses. It is mainly used for clustering documents but can be adapted to be used for image datasets. This project streamlines its use for image datasets so it can be used with any image dataset. It also expands on it by connecting it to a database to store the resulting clusters which can be later used for further interpretation or for generating insights and used as training data for a new machine learning model.

The code:

I've included some relevant code snippets below and the explanation of how and why the project works. The full code can be found in the project submission or in the github link.

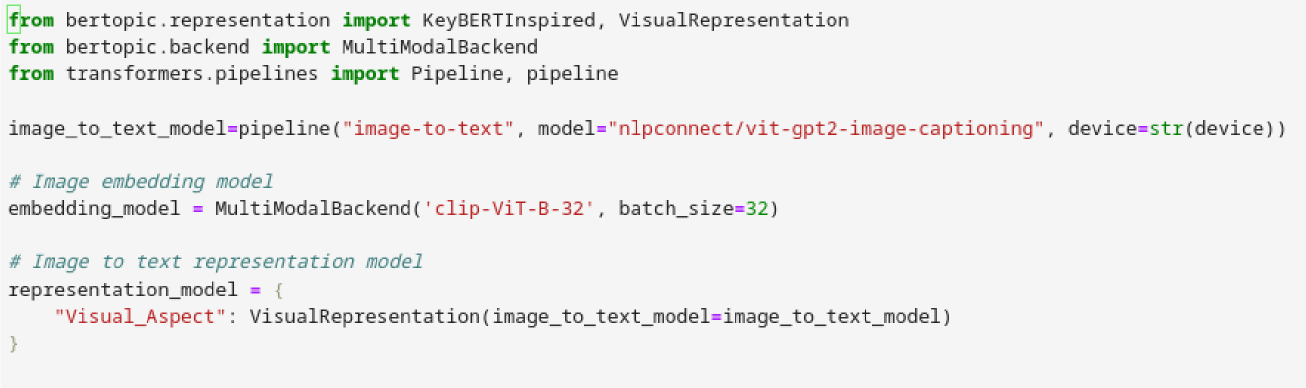


```
import glob
images = list(glob.glob('dataset/*.jpg'))
```



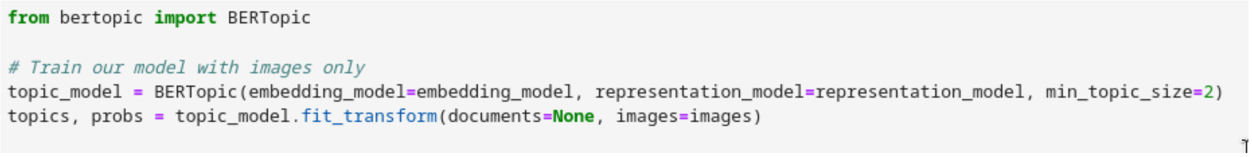
```
import torch
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

Here we load and initialize the dataset and pytorch which is necessary for BERTopic to run.



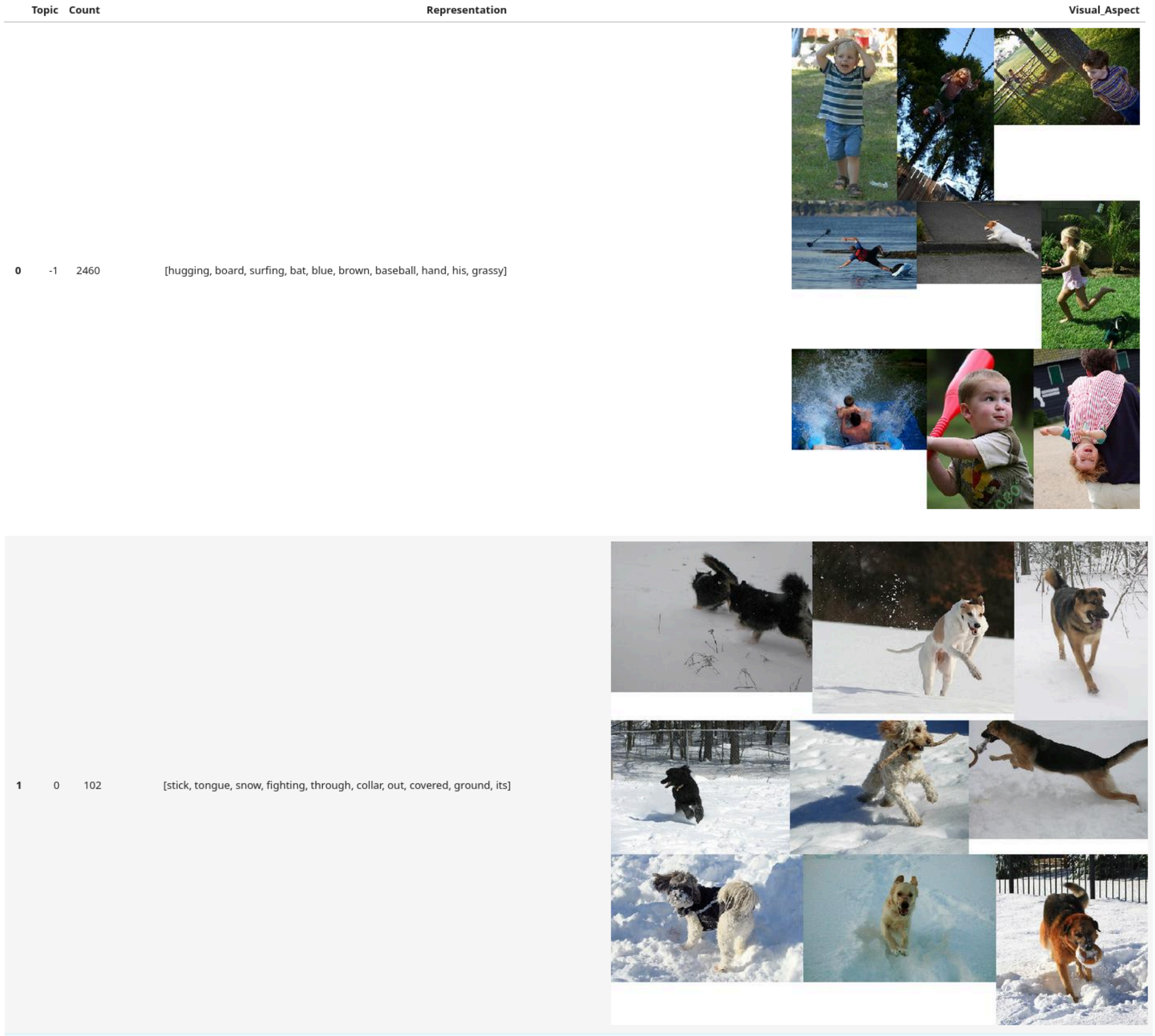
```
from bertopic.representation import KeyBERTInspired, VisualRepresentation
from bertopic.backend import MultiModalBackend
from transformers.pipelines import Pipeline, pipeline

image_to_text_model=pipeline("image-to-text", model="nlpconnect/vit-gpt2-image-captioning", device=str(device))

# Image embedding model
embedding_model = MultiModalBackend('clip-ViT-B-32', batch_size=32)

# Image to text representation model
representation_model = {
    "Visual_Aspect": VisualRepresentation(image_to_text_model=image_to_text_model)
}
```

This is the default configuration and initialization of the BERTopic model. We specify that we want to use CLIP as the backend which generates the embeddings for the images in our dataset. This model then uses a clustering algorithm to cluster those embeddings. We also specify that we want to use vit-gp2-image-captioning model to generate captions for the images in our dataset.



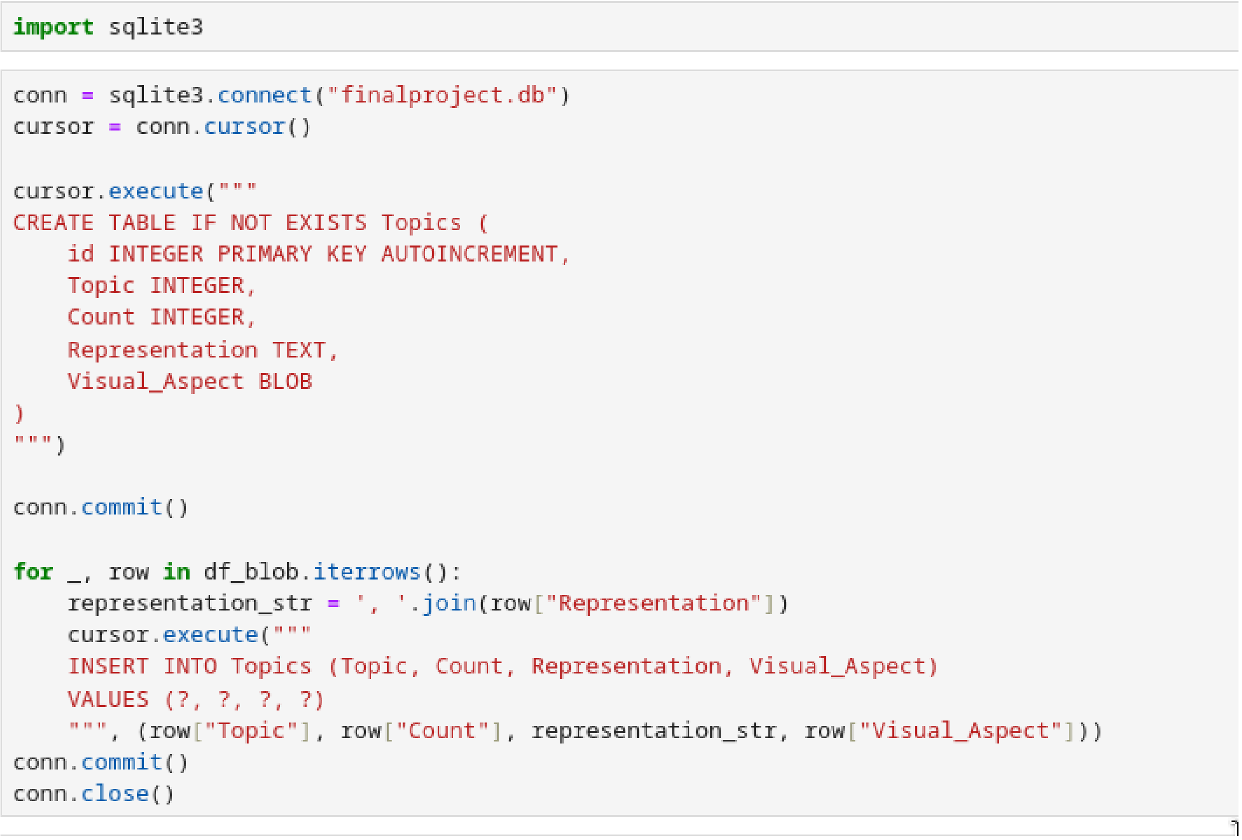
```
from bertopic import BERTopic

# Train our model with images only
topic_model = BERTopic(embedding_model=embedding_model, representation_model=representation_model, min_topic_size=2)
topics, probs = topic_model.fit_transform(documents=None, images=images)
```

This is the part where we call the BERTopic model to start training on our dataset and images. The model clusters the images in our dataset and captions them. The result will be our dataset divided into clusters of images with labels specifying topics for each cluster which as extracted from the captions, as shown below.



| | Topic | Count | Representation | Visual_Aspect |
|---|---|---|---|---|
| 0 | -1 | 2460 | [hugging, board, surfing, bat, blue, brown, baseball, hand, his, grassy] | |
| 1 | 0 | 102 | [stick, tongue, snow, fighting, through, collar, out, covered, ground, its] | |

Here we can see the cluster of images that the model thinks are similar on the right. And we can see the topics that it generated in the middle. We can see that our model clustered images of dogs in the snow together and on top it looks to be images of children clustered together. I have only shown two of the clusters here but in total the model divided the dataset into around 800 clusters. Not all clusters are the same size, but it is possible to adjust the number of clusters to ensure a more even distribution.



```
import sqlite3
```

```
conn = sqlite3.connect("finalproject.db")
cursor = conn.cursor()

cursor.execute("""
CREATE TABLE IF NOT EXISTS Topics (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    Topic INTEGER,
    Count INTEGER,
    Representation TEXT,
    Visual_Aspect BLOB
)
""")

conn.commit()

for _, row in df_blob.iterrows():
    representation_str = ', '.join(row["Representation"])
    cursor.execute("""
    INSERT INTO Topics (Topic, Count, Representation, Visual_Aspect)
    VALUES (?, ?, ?, ?)
    """, (row["Topic"], row["Count"], representation_str, row["Visual_Aspect"]))
conn.commit()
conn.close()
```

Finally we store the resulting clusters in a database using sqlite3 as shown above. Storing the results allows us to compare it to future runs with changes to the code and also allows us to use the results to train new models.

Advantages and limitations:

- This approach allows us to distinguish between dissimilar images within a large dataset. This allows us to quickly sort through the categories to find the images we are interested in and images similar to it.
- Limitations include the fact that the model is very sensitive to the number of clusters and changing this number can change the results drastically.
- Possible improvements would be to try other clustering algorithms. Try different captioning models. Adjust hyperparameters such as the number of clusters.

Changes after Proposal:

- In the proposal I originally planned on using BERTopic to cluster documents, but I found that clustering images yielded more interesting results so I chose to change the direction of my project from the initial proposal. Other than that most of the details of my project remain the same as in the proposal. I used the same BERTopic model but with different adaptations so it can work with images and I just had to use an image dataset rather than a document one.