Can Image-to-Text Models be useful for generating Image Captions as a Feature in a Recommender System?

An Artificial Intelligence Research Project

Dries De Ryck
Master in Computer Science
Data Science and Artificial Intelligence
University of Antwerp
dries.deryck@student.uantwerpen.be

1. Introduction

In this report I will describe my research for the Artificial Intelligence Project course. For this project we looked at the *H&M Personalized Fashion Recommendations* [1] competition on Kaggle. The goal of this competition is to make fashion item recommendations to a number of customers. Several datasets were provided, about the customers, the articles and previous purchases. Also included were images of the fashion articles. These images are the starting point of my research.

In the provided dataset about the articles, there are descriptions that describe what the article is. While doing exploratory data analysis, I found that some of these descriptions are quite simple and short. For example:

Jersey top with narrow shoulder straps.

This gave me the idea that maybe the images could help to give more information about the articles. If I could generate new descriptions of the articles based on the images, maybe they would contain more information than what the original description contained. I had no real idea if these descriptions were at all useful for making recommendations, but the course did not require us to deliver work that would increase the recommendation score. So because I found it interesting to know if I could use these images to generate new descriptions, I decided that this would be my research question:

Can Image-to-Text models be useful for generating image captions as a feature in a recommender system?

2. Background

At the start of this course I had no experience with data science in practice and a very limited experience with artificial intelligence. So for me this course was a way to learn new things, to experience what it is like to apply concepts from these fields. I had to learn how to use several new data science libraries, even *Pandas* was something I had not previously worked with.

I had seen some theory in a course *Information Retrieval* that links with this project: concepts like Vector Space Models, Dimensionality Reduction and Natural Language Processing. As I will explain later, I encountered these concepts while I was working on this research project.

3. Methodology

Originally my research question was a bit different:

Can we use AI that generates a description given an image to improve a content-based recommender system that uses the current descriptions?

The plan was to calculate similarities between articles using the article descriptions and generated descriptions. These could then be used to try to recommend items similar to previously bought articles. These similar items could be used for candidate generation and later ranking. In Results I will discuss why I changed my research question to be more focused on feature engineering.

My methodology for my new research question went as follows: First I had to find an image-to-text model that works. I decided to start with this because my research would fail if the image-to-text model would not work out as expected. Also I only needed the images as input for the model, which I had access to. The way I evaluated the image-to-text model was by sampling some random articles from the dataset and trying the model on these samples. I would judge by eye if the generated description would match the image. Apart from the quality of the generated descriptions, this was also a way to test the efficiency of the model. Since there are 105.100 images to work with, the model had to be fast enough.

When I found the right model, I had to actually generate the image descriptions. Even with a fast model, generating descriptions for all the images would be computationally very expensive.

The framework for candidate generation and ranking that I used was the *Radek's LGBMRanker starter-pack notebook* [2] that was provided in the course. To be able to use these generated descriptions for making recommendations, I had to make a feature out of them that could be included in ranker training. Therefore, I needed to transform the generated descriptions into embeddings. Then I could add the embeddings as multiple features.

After this I would train the ranker and evaluate the feature importance of the embeddings. I would also look at the competition score to see if the generated descriptions could increase it.

4. Results

For my original research question I tried vectorising the descriptions using TfidfVectorizer from SKLearn [3]. I was able to calculate cosine similarities between these vectors, but since the dataset was so large, it was computationally impossible to create a matrix with similarities between each pair of articles. Therefore I changed my research to be more focussed on embeddings that I would use as a feature.

The first Image-to-Text model I looked at was the Vision Transformer Model [4] (ViT). This model was not able to generate descriptions with the quality that I was looking for. An example of an image and the description that was generated for it is given below.

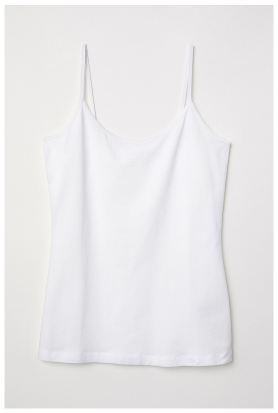

Figure 1. An article image from the dataset.
Caption by ViT: ”a white shirt and a black shirt”

After testing a bit more, I got better results from the BLIP 2 model. This model ”leverages frozen pre-trained image encoders and large language models (LLMs) by training a lightweight, 12-layer Transformer encoder in between them”. A visual representation of the architecture is included below. To give an example, the description that was generated for the same image as with the ViT model is ”a white tank top with spaghetti straps and a low cut neckline”. [5]

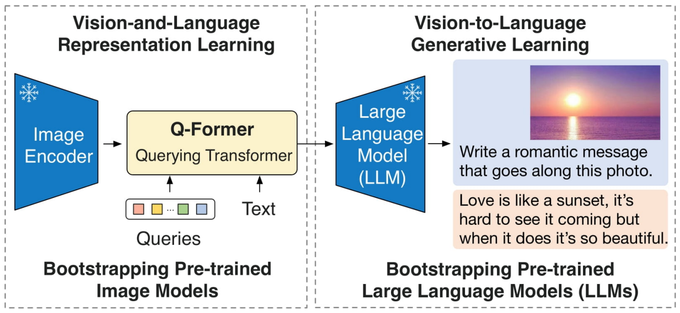
Vision-and-Language
Representation Learning
Vision-to-Language
Generative Learning
Image Encoder
Q-Former
Querying Transformer
Queries
Text
Large Language Model (LLM)
Write a romantic message that goes along this photo.
Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.
Bootstrapping Pre-trained Image Models
Bootstrapping Pre-trained Large Language Models (LLMs)

Figure 2. BLIP 2 Architecture

I decided to stick with this since these were the best results I had gotten from a pretrained image-to-text model. Now I had to generate descriptions for more articles. I chose to limit the number of articles to a thousand to limit computation time. Since the most popular items would be highly likely to be recommended to users, I chose to generate descriptions for the thousand bestseller items.

Now it was time to create embeddings for these descriptions. Using the Sentence Transformer model ”all-MiniLM-L12-v2” [6], I was able to map the descriptions to a 384-dimensional vector space. I could not add the embeddings as features just yet, as I had to do some principal component analysis to reduce the dimensionality. When I looked at the first 50 principal components, they had a variance of 84% of the original variance of the vector space, while being much more space efficient. This way I was able to add the 50 principal components as features to the data and give this to the ranker to train.

Looking at the results from the ranker, the six most important features without the bestseller feature included three principal components of the embeddings. The most important feature was principal component 3 with 30%. A table with percentage of importance is given below.

pc3	0.30%
article_id	0.08%
pc42	0.06%
product_type_no	0.05%
pc22	0.03%
department_no	0.03%

TABLE 1. THE 6 MOST IMPORTANT FEATURES

Although 30% seems like a good result, the fact that this principal component was much more important than other features and other principal components, without a reasonable explanation, already made me question the result. Still I checked the competition score and compared it to results of the same data trained without the principal components.

	private score	public score
Trained with principal components	0.01175	0.01175
Trained without principal components	0.0143	0.01417

TABLE 2. COMPETITION SCORES

Unfortunately, the competition scores show that the extra features did not raise the score.

5. Discussion

I will now try to answer my research question. Unfortunately the results that I got were not successful. A possible answer to why my approach did not work could be that the ranker did not have enough valuable information to be able to train and actually learn something. Perhaps with more features, it would have been able to pick up on valuable information and do a better job. The popularity baseline was not good for trying out features, because it would be so important that other features were drowned out in training the model. But perhaps a different baseline for comparing features would have shown whether the principal components did really add valuable information.

That said, I was really quite happy with the results from the image-to-text model. The tests that I did with it were really promising and sometimes the generated descriptions were more informative than the original descriptions. If someone was able to use the original descriptions in a meaningful way, I am positive that the image-to-text model could add value to this.

6. Conclusion

In conclusion, although my research did not result in an improvement of the competition score, I do see a place for Image-To-Text models in projects like this. In this project almost all of the items had original descriptions, but imagine a dataset where more descriptions are missing and images are available. Then the images could certainly be used for generating the missing descriptions.

In other cases, and possibly this project, Image-to-Text models might not be the best solution. If there are already descriptions available, you might not need to generate new ones. If you wanted to use the images, perhaps an embedding of the image itself might be more logical than transforming it to text and then making embeddings. So there are times when image-to-text models are useful and times when they are not.

Apart from the result, I did personally learn a lot while working on this project, like learning new libraries and principal component analysis. Working with these large datasets definitely posed a challenge, and in some cases I was able to find a way to make it work.

References

[1] H&M Personalized Fashion Recommendations Kaggle Competition, https://www.kaggle.com/competitions/h-and-m-personalized-fashion-recommendations

[2] Radek's LGBMRanker starter-pack Notebook, https://www.kaggle.com/code/marcogorelli/radek-s-lgbmranker-starter-pack/

[3] TfidfVectoriser Module, https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[4] Vision Transformer Model, https://huggingface.co/docs/transformers/model_doc/vit

[5] Blip 2 Model, https://huggingface.co/docs/transformers/model_doc/blip-2

[6] Sentence Transformers all-MiniLM-L12-v2 Model https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2