2022-2023, CVLab - EPFL + ECAL

Start: 20.09.2022
Finish: 13.01.2023

Heritage in the Digital Age

A Step-by-Step Guide to Augmenting Digitized Historical Images

Théo Gieruc

Young woman sitting on a bench and holding a umbrella, Fribourg

Professor: Pascal Fua
Supervisors: Mathieu Salzmann, Delphine Ribes

Contents

Abstract

Digitized heritage collections, such as those found in libraries and museums, have the potential to reach a larger audience and become more widely available through digital access. However, the use of these collections is often limited to remote access and research tools. This report presents a pipeline for augmenting digitized archive images with advanced computer vision and natural language processing techniques, with the goal of increasing their value and appeal. The pipeline consists of six steps: translation, text data preprocessing, phrase grounding, result postprocessing, result selection, and object segmentation through a novel class-agnostic segmentation model. The pipeline was applied to a dataset of 2,216 images from the BCUFR and the effectiveness of the approach was demonstrated. The results show that it is possible to enhance the value of digitized heritage collections through image augmentation, and potential avenues for future research are discussed.

Keywords: computer vision, natural language processing, phrase grounding, class-agnostic segmentation, referring expression segmentation

Chapter 1 Introduction

Many institutions, such as libraries and museums, have begun to digitize their collections in order to make them more accessible to a wider audience. However, the use of these digitized images has typically been limited to remote access and research tools. This projects aims to explore the potential for the valorization of these digitized archives by augmenting the images with the use of state-of-the-art computer vision and natural language processing techniques.

This project is part of a collaboration between the EPFL+ECAL Lab and the Cantonal University Library of the Canton of Fribourg (BCUFR) [1]. The dataset used is from the BCUFR, consists of 2,216 pictures from 1870 to 2003, with AI-generated captions and titles in French. The captions have been generated using an image captioning pipeline designed by Chenkai Wang, which consists of two main steps: image captioning with CLIP [2] and GPT-2 [3] and entity discovery and linking to detect and pair entities.

This report presents a pipeline for augmenting digitized heritage images, which includes several key steps, such as translation, phrase grounding and segmentation. The results of the pipeline, including segmentation results and different approaches for automatic selection of the best inference, are also presented. A user survey is conducted to assess the effectiveness of the pipeline. The report concludes with a discussion on future work.

Chapter 2 Literature review

Referring expression segmentation is a fundamental task in natural language processing and computer vision, which involves identifying and segmenting an object in an image based on a natural language expression describing it. In recent years, several approaches have been proposed to address this problem.

MattNet (2018) [4] decomposes expressions into three modular components - the subject, the location and object-subject relation - and uses language-based and visual attention to focus on relevant information. It has been trained on RefCOCO, RefCOCO+ [5] and RefCOCOg [6]. All three are based on MS-COCO [7]. The type of objects that appear in the referring expressions, the length of the expressions, and the relative size of the referred objects are the main differences between the datasets.[8]. DMN (2018) [8] combines elements of previous methods and takes advantage of intermediate information during upsampling as well as a synthesis module to process the linguistic and visual information jointly. It was also trained on RefCOCO, RefCOCO+ and RefCOCOg. Lang2Seg (2019) [9] uses spatial-aware dynamic filters to transfer knowledge from text to image and capture the spatial information of the specified object. The method also includes a caption generation network that takes features shared across both domains as input and improves both representations by enforcing consistency between the generated sentence and the given referring expression. It was trained on RefCOCO and RefCOCOg. MDETR (2021) [10] uses a transformer-based architecture to reason jointly over text and image by fusing the two modalities at an early stage of the model. It has been fine-tuned on several downstream tasks, such as phrase grounding with the PhraseCut dataset [11] and Flickr30k [12], and referring expression segmentation with RefCOCO, RefCOCO+ and RefCOCOg. LAVT (2021) [13] propose a method that involves early fusion of linguistic and visual features in the intermediate layers of a vision Transformer encoder network, rather than using a separate cross-modal decoder to fuse features extracted from separate vision and language encoders as in previous approaches. It was also trained on RefCOCO, RefCOCO+ and RefCOCOg.

Phrase grounding is closely related to referring expression segmentation. It involves identifying the objects or regions in an image that correspond to a given phrase or word in a description. GLIP (2021) [14] and MDETR [10] are two of the most successful approaches that have been developed for this task, according to Papers with code [15]. GLIP has an trasnformer-based architecture and has been declined in five variants to ablate its three core techniques: 1) unified grounding loss; 2) language-aware deep fusion; 3) and pre-training with both types of data. It has been trained on a set of modified datasets: Flickr30k [12], GQA [16], VG Caption [17], OpenImages [18] and Object365 [19]. All images included in MS-COCO have been removed, in order to keep them for evaluation.

In contrast to referring expression segmentation, which only detects the main subject of a description, phrase grounding detects every object mentioned in the description. This is demonstrated in Figure 2.1, where the referring expression segmentation model only detects the woman, while the phrase grounding model detects both the woman and the dog. Therefore, in this project, we will use phrase grounding instead of referring expression segmentation to more comprehensively detect all objects mentioned in a description.

(a) Referring expression segmentation (LAVT)

(b) Phrase Grounding (MDETR)

Figure 2.1: Expression : *A woman with her dog*

Chapter 3 The Pipeline

The pipeline consists of six steps, as illustrated in Figure 3.1. It takes as input an image with its title in French and an AI-generated alternative caption in English.

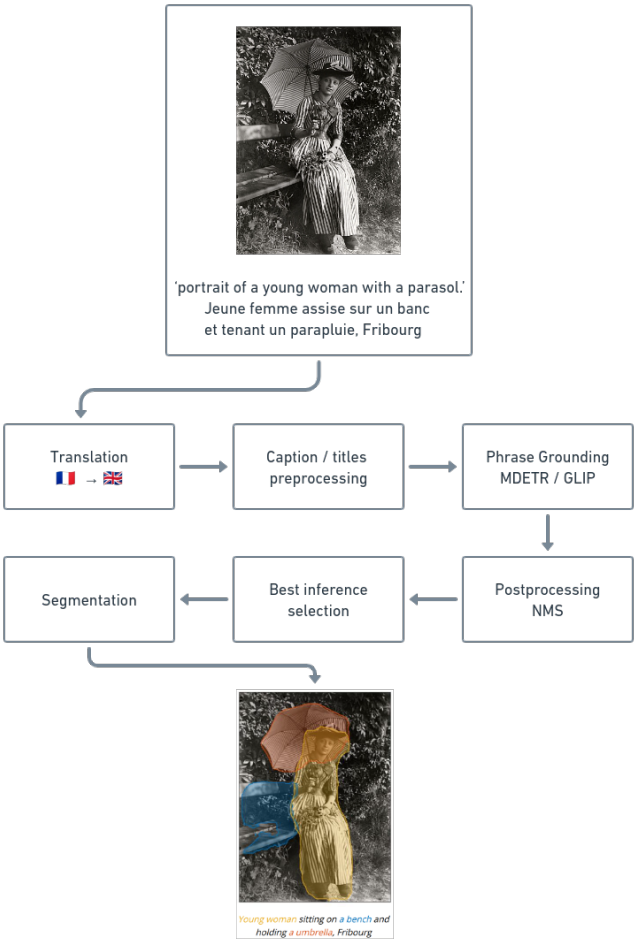

Figure 3.1: The six-step pipeline for valorizing digitized heritages.

3.1 Translating the Titles of Images

Since the phrase grounding models selected were trained on English captions, the titles of the images need to be translated from French to English. A pretrained machine learning model, MarianMT [20], which is a multilingual machine translation model trained on the OPUS [21] corpus, is used for this purpose. This specific model was selected because of its ease of implementation with HuggingFace [22].

Figure 3.2: Translation of the titles from French to English

3.2 Preprocessing the Captions and Titles

The captions and titles are preprocessed in preparation for phrase grounding. This is achieved by converting the text to lowercase and removing the following expressions: “portrait of”, “photograph of” and “black and white photo of”.
As this dataset consists of images from Fribourg, there are many mentions of Fribourg in the title, which can potentially confuse the phrase grounding models. Therefore, the following mentions were removed: “canton de fribourg”, “of fribourg”, “(fribourg)”, “[fribourg]” and “fribourg”.
The expressions “group of” and “a group of” have also been removed, as the phrase grounding algorithms would often give the label “group” rather than the subject of the group. For example, with the expression “a group of musicians”, the phrase grounding algorithm would choose the label “group” rather than “musicians”.

Figure 3.3: An example of preprocessing, with removal of ’Portrait of’ and ’[Fribourg]’, and casefolding

3.3 Phrase Grounding

Phrase grounding involves detecting the objects in an image that are mentioned in a caption. In this project, two state-of-the-art models were used for this task: GLIP [14] (GLIP-L - 430M parameters) and MDETR [23] (EfficientNet-B5 - 169M parameters). Inference was run on both the caption and the title for each image. An example of this process is shown in Figure 3.4, where an image and its caption are given as input to MDETR.

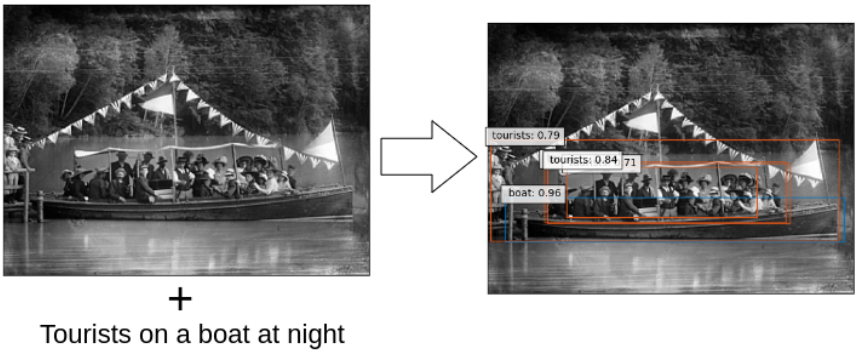

Figure 3.4: Inference on an image and its caption, using MDETR

3.4 Non-maximum Suppression

Both MDETR and GLIP tend to create several bounding boxes for an object, as shown in Figure 3.4. To address this, non-maximum suppression (NMS) is applied, a technique that removes bounding boxes that overlap too much with other bounding boxes. The standard NMS algorithm [24] is used twice: once label-wise with a threshold of 0.5, and once global with a threshold of 0.9. An example of the results on an inference from MDETR is shown in Figure 3.5.

3.5 Selecting the Best Phrase Grounding Results

For each image, there are four phrase grounding results: two from the caption and two from the title, as the models GLIP and MDETR were used. A user-friendly GUI is used to select the best results for each image. The definition of "the best result" can be somewhat arbitrary, and will be discussed in Chapter 7 User Survey. An example of this selection process is shown in Figure 3.6. The number of selected combinations is shown in Table 3.1.

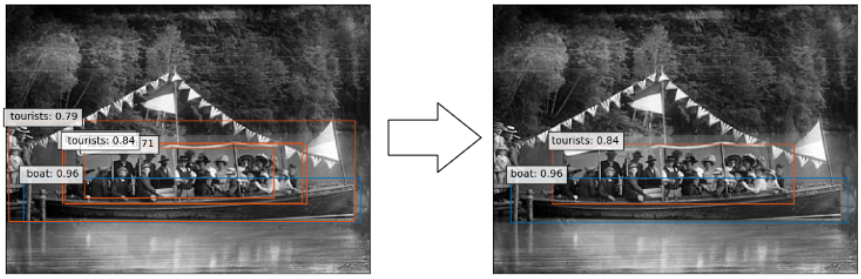

Figure 3.5: Example of Non-Maximum Supression

	MDETR	GLIP
Caption	78	1029
Title	15	224

Table 3.1: The number of time each combination was selected during the manual selection

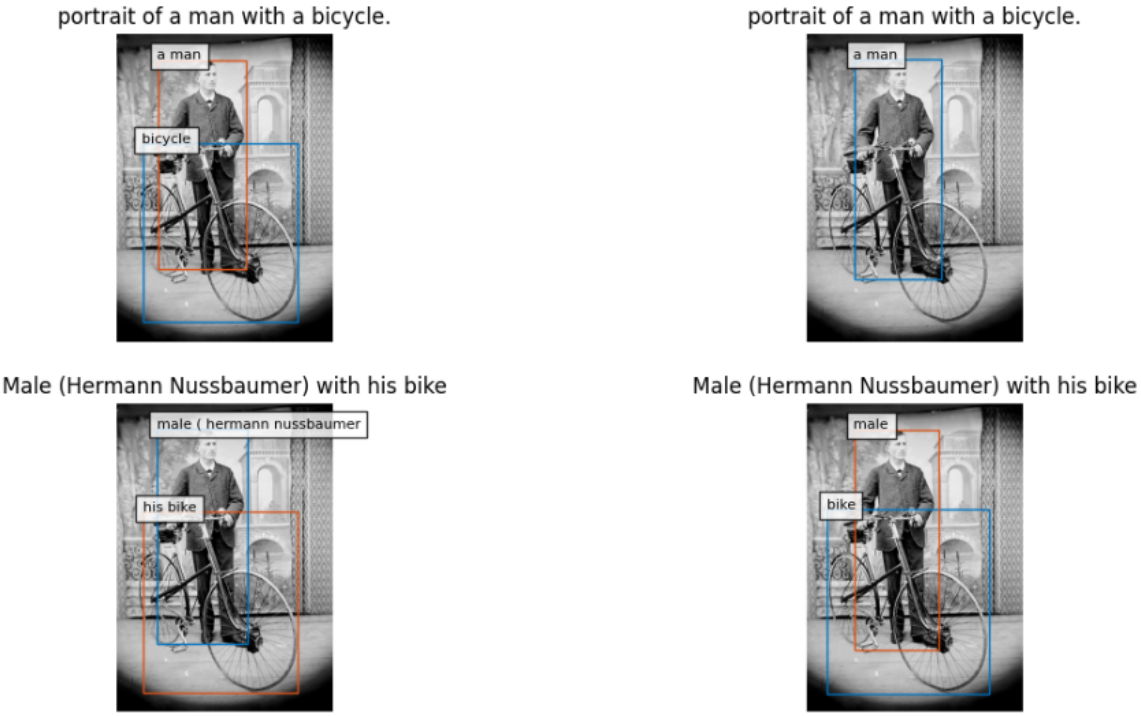

Figure 3.6: Example of the selection of the best inference between the four possibilities. On top, the AI-generated caption-induced inferences, on bottom the titles. On the left MDETR, on the right GLIP.

3.6 Segmenting the detected objects

In order to further augment the images, the detected objects are segmented in the images. The bounding boxes from the phrase grounding results are used to crop the objects and then segment them. Initially, GrabCut [25] was chosen for this task due to its simplicity, but its performance was disappointing, especially on black and white images, which make up the majority of the dataset. GrabCut is based on color information to distinguish between foreground and background pixels. In a black and white image, there is only one color channel, so there is less information available for GrabCut to use in making segmentation decisions. This led to designing a custom class-agnostic segmentation model on PyTorch [26].

3.6.1 Design of the model

The Segmentation Models library [27], which provides a wide range of pretrained segmentation models, was used to design the model. The chosen architecture is a U-Net [28] with a ResNet34 [29] backbone, pretrained on the Image-Net [30] dataset. An alternative model, with a DeepLabV3 [31] architecture and MobileNetV3 [32] backbone has also been trained for comparison.

3.6.2 Fine-tuning

The model was fine-tuned on a subset of the COCO-2014 [33] dataset, which consists of 5'000 images with bounding boxes and segmentation masks. The following process was used for fine-tuning the model:

1. The image is resized to 352x352, normalized, converted to black and white and then turned into a tensor. The bounding box is used to create a mask of the object in the image.

2. The image is multiplied by the mask to remove the background and is used as the input to the model.

3. The image is used as the input to the model, and the segmentation ground truth is used as the target for the model.

The model is trained for 100 epochs, using the Adam optimizer [34] and the Dice Loss [35] as the loss function, which is a useful loss function for image segmentation tasks because it is easy to optimize, handles class imbalances well, and is differentiable and continuous.

The process of fine-tuning the model is illustrated in Figure 3.7.

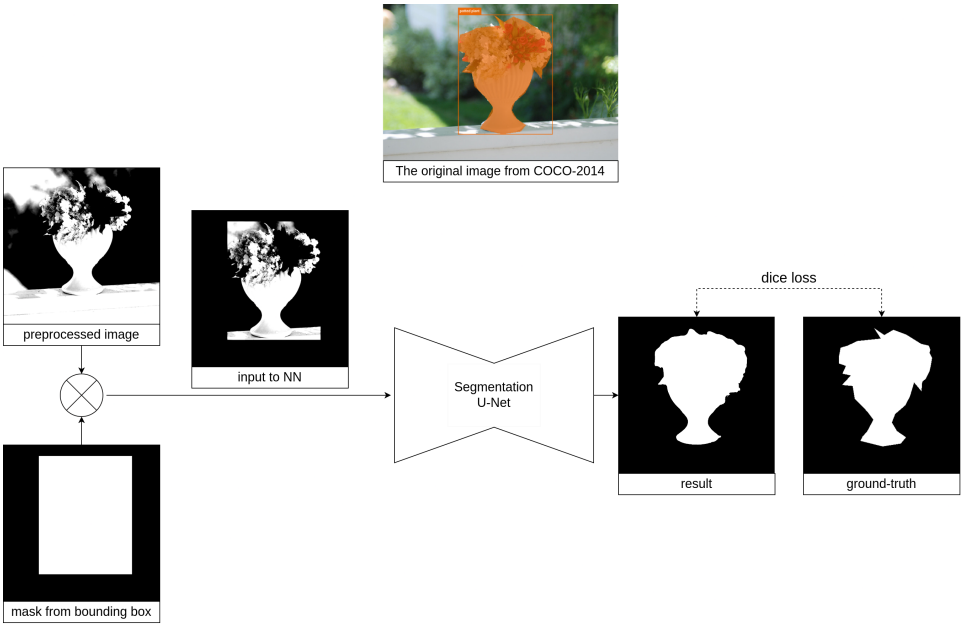

Figure 3.7: The fine-tuning process

3.6.3 Evaluation

The model is evaluated on a COCO-2014 validation subset, which consists of 1'000 images with bounding boxes and segmentation masks. It achieves a mean Intersection over Union (IoU) of 0.77, which is calculated as the ratio of the area of overlap between the predicted bounding box or segmentation mask and the ground truth, to the area of union between them:

$$IoU = \frac{(A \cap B)}{(A \cup B)}$$

where A is the predicted bounding box or segmentation mask and B is the ground truth. The IoU score ranges from 0 to 1, with higher values indicating better performance. A mean IoU of 0.77 indicates that the model is performing well but there is still room for improvement.

The model also achieves mean pixelwise precision and recall scores of 0.82 and 0.91, respectively. Pixelwise precision and recall are other common evaluation metrics used in object segmentation, and they measure the fraction of true positive pixels out of all predicted positive pixels (precision), and the fraction of true positive pixels out of all actual positive pixels (recall). These values provide additional insight into the performance of the model and can be used to identify any potential trade-offs between

precision and recall. A higher recall than precision seems to indicate that the model is more sensitive to detecting positive pixels but may be less accurate in predicting them. In other words, the model is more likely to identify most of the positive pixels (high recall), but some of the predicted positive pixels may be incorrect (lower precision).

In comparison, the alternative model with the DeepLabV3 architecture and MobileNetV3 backbone, has a mean IoU of 0.76, a mean pixelwise precision score of 0.79 and a recall score of 0.9. As this performs slightly worse than the U-Net + ResNet34 model, I will keep the latter.

3.6.4 Inference

The model was run on the images to segment the detected objects using the bounding boxes from the phrase grounding results. The preprocessing for this step was slightly different from the fine-tuning process: the image was resized to 352x352 with padding, normalized, and then turned into a tensor. After the inference, the segmentation was multiplied by the mask to remove any noise and was resized to the original size of the image. The process of running inference is illustrated in Figure 3.8.

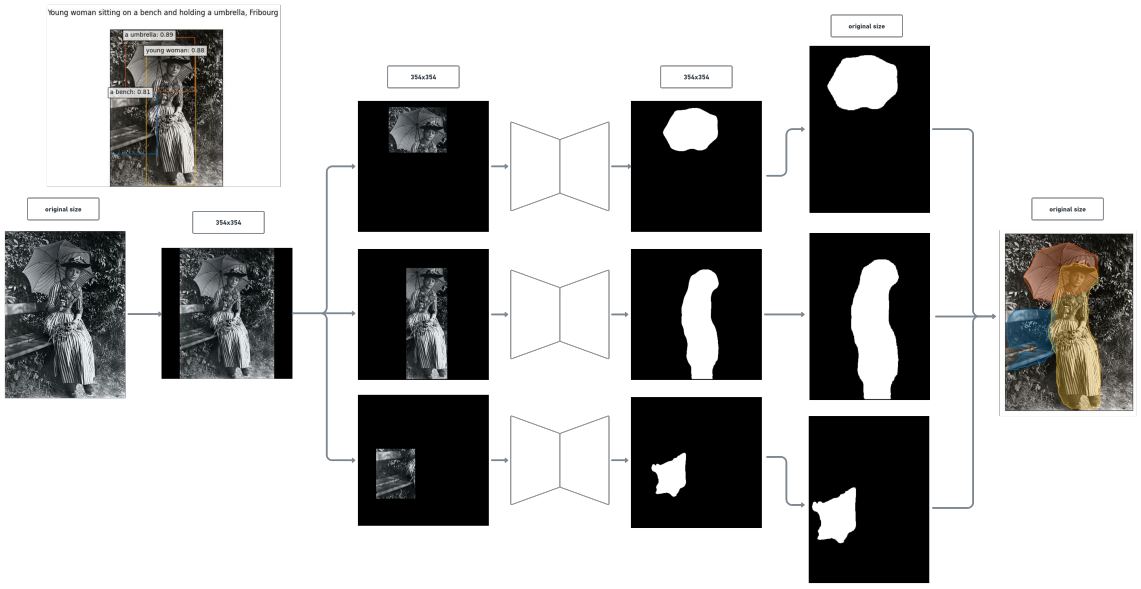

Figure 3.8: The inference process

3.6.5 Comparison with Grabcut

A comparison between GrabCut and the deep learning model is shown in Figure 3.9, on color and black & white images, for one and five inferences. On average, GrabCut takes 259ms per inference, whereas the deep neural net model only needs 47.3ms , which makes it not only better but also faster.

	1 inference	**5 inferences**
GrabCut	259ms	1240ms
Deep Segmentation Model	47ms	143ms

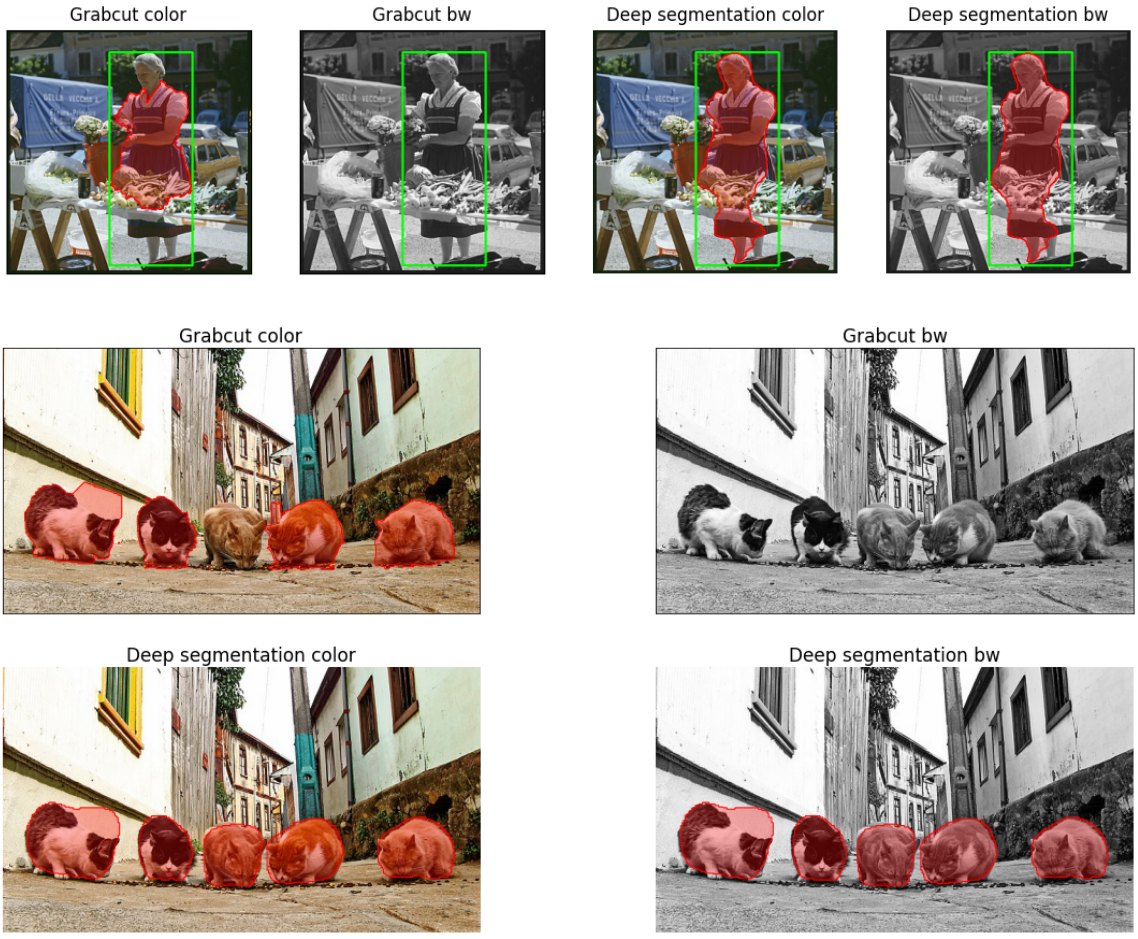

Figure 3.9: Comparison on color and black & white images between GrabCut and the Deep Segmentation model

Chapter 4 Results

We apply the pipeline to the dataset of 2,216 pictures from the BCUFR and demonstrate the effectiveness of the approach. Figure 4.1 shows some examples of the augmented images. All 1'500 images can be found here `https://tgieruc.github.io/Heritage-in-the-digital-age/`.

Figure 4.1: Examples of augmented images produced by the pipeline.

4.1 Segmentation

The segmentation model performs better on single objects than on groups of objects, as shown in the Figure 4.2. The left image shows that the "children" and "students" are better segmented than in the right image.

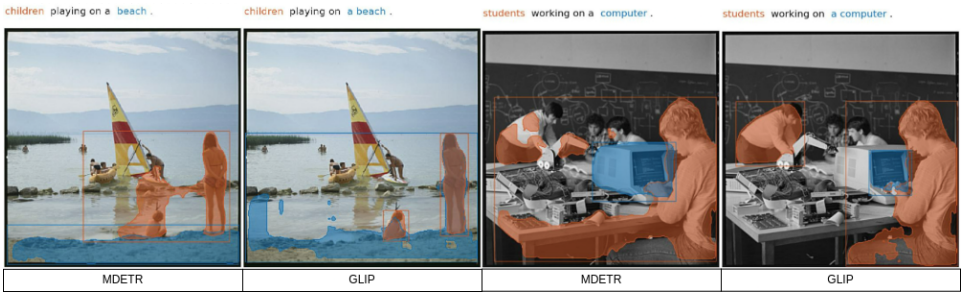

Figure 4.2: The segmentation model performs better on bounding box around a single object than a group of objects

The number of bounding boxes and their median area were compared between GLIP and MDETR to see which model performs better on phrase grounding. Figure 4.3 shows that GLIP tends to output more bounding boxes than MDETR, and they also tend to be smaller. This difference in output could be attributed to the different training data used by the two models. These results suggest that, due to its smaller and more numerous bounding boxes, GLIP is a more suitable option, as its bounding boxes would likely lead to better segmentation than MDETR.

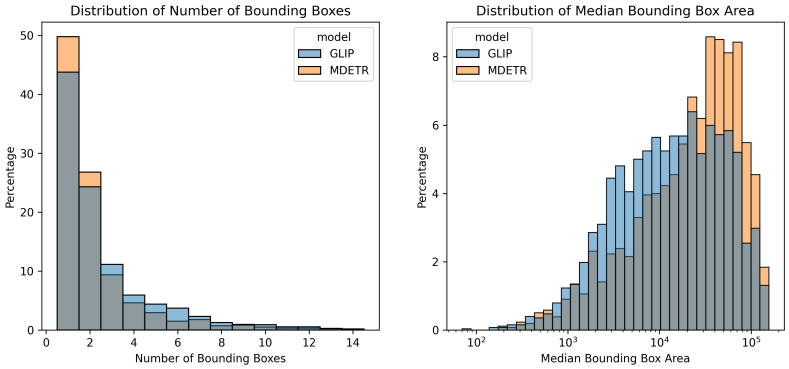

Figure 4.3: On average, GLIP produces more bounding boxes that are smaller in size compared to MDETR.

4.2 Automatic selection of the best inference

Manually selecting the best inference for each image in a dataset can be time-consuming, especially when dealing with large datasets. To address this issue, several different methods for automatic classification of the best inference were tried.

Both a random forest classifier and a MLP classifier were unable to perform better than always selecting the most popular combination. They were trained using the statistics collected during manual selection as features, such as the number of words in the caption and title, as well as the number of bounding boxes and their minimum, maximum, and median area for each combination of [caption, title] and [GLIP, MDETR].

DistilBERT-base-uncased [36] with HuggingFace [22] was used for two different approaches:

- Given the caption and the title, the model outputs which one is the best. It was trained on the manual selection data. However, it tends to always prefer the captions over the titles. The model can be found on HuggingFace [37].
- This model ranks a single expression (either a caption or a title) between 0 and 9. Over 400 expressions, extracted randomly from the titles and captions, were manually rated as training and test data. The model was trained for 15 epochs using the Adam optimizer [34]. The distribution of the grades, predictions, and absolute differences can be found in Figure 4.4 and the results are shown in Table 4.1. Overall, the model predicts with an accuracy of 81% the grade with an error of 1 or less. This model should be used by giving a rating to every caption and title, and then selecting the one with the highest grade. For ease of use, the model and a pipeline have been uploaded on HuggingFace [38] [39].

Absolute difference with ground truth								
0	1	2	3	4	5	6	7	8
66.5%	15.9%	6.6%	3.4%	2.4%	2.2%	0.97%	1.4%	0.5%

Table 4.1: The distribution of the absolute difference between the ground truth and the prediction

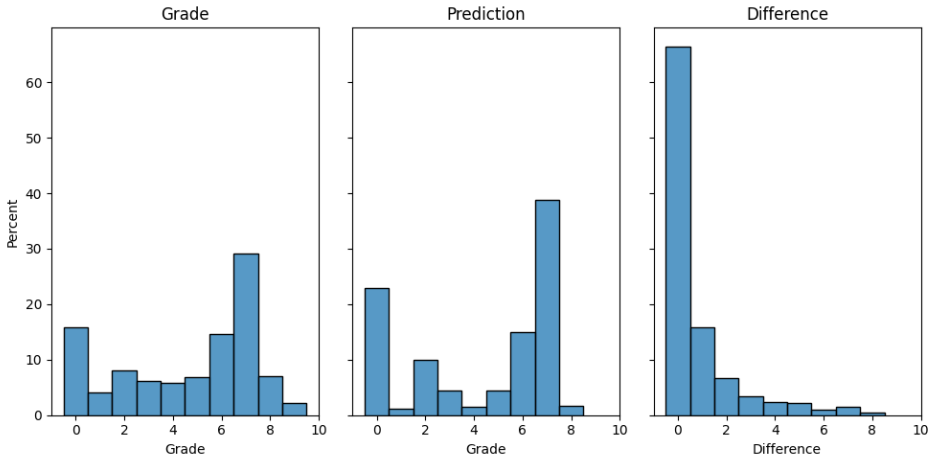

Figure 4.4: The distribution of the grades, the ground truth and the absolute difference between grades and ground truth

Chapter 5 Implementation

The implementation of the pipeline consists of one Python script per step, with the objective to make it as simple as possible to apply on different datasets.

5.1 Example of application of the whole pipeline

In this example, we will have the following architecture. A Jupyter Notebook demo hosted on Google Colab of the whole pipeline can be run here: `https://colab.research.google.com/github/tgieruc/Heritage-in-the-digital-age/blob/main/demo_pipeline.ipynb`.

```
main directory
├── data
│   └── BCU_database
│       ├── original_data.pkl
│       └── images
│           ├── ALBL00222_2k_324w.jpg
│           ├── ALBL00222_2k_2975h.jpg
│           └── [...]
└── pipeline
    ├── src
    │   └── GLIP [submodule]
    ├── 1_translate.py
    ├── 2_preprocessing.py
    ├── 3_GLIP_inference.py
    ├── 3_MDETR_inference.py
    ├── 4_apply_nms.py
    ├── 5_automatic_selection.py
    ├── 5_manual_selection.py
    └── 6_segmentation.py
```

`original_data.pkl` is a Pandas DataFrame serialized as a Pickle file that contains the following columns: *bcu_id* the ID of the image, *title* the title in French and *caption* the AI-generated alternative caption.

The images are stored in the `data/BCU_database/images/` folder and are in two resolutions: 324w and 2975h.

5.1.1 Translation

The following command is a Python script that translates the values in a specified column of a dataset stored in a pickle file:

```
python pipeline/1_translate.py
        --input_file data/BCU_database/original_data.pkl
        --column title
        --output_file data/1_translation.pkl
        --device cuda
```

This script takes several arguments:

- `--input_file` specifies the input file, which is a pickle file containing the dataset to be translated.
- `--column` specifies the column in the dataset that should be translated. In this case, the `title` column will be translated.
- `--output_file` specifies the output file, which is a pickle file where the translated dataset will be stored.
- `--device` specifies the device to use for translation. The value of this argument, `cuda`, indicates that a GPU should be used if one is available. If you do not want to use the GPU, you should write `cpu`.

This script adds a column `title_en` to the DataFrame, containing all the titles translated in English.

5.1.2 Preprocessing

The following command is a Python script that performs the preprocessing of the titles and captions and links each `bcu_id` to an image:

```
python pipeline/2_preprocessing.py
        --input_file data/1_translation.pkl
        --id_column bcu_id
        --image_directory data/BCU_database/images
        --output_file data/2_preprocessing.pkl
        --quality 324w
        --columns_to_preprocess caption title_en
```

This script takes several arguments:

- `--input_file` specifies the input file, which is a pickle file containing the dataset to be preprocessed.

- `--id_column` specifies the column in the dataset that contains the unique identifier for each record.
- `--image_directory` specifies the directory containing the images associated with the ids in the dataset.
- `--output_file` specifies the output file, which is a pickle file where the preprocessed dataset will be stored.
- `--quality` specifies the quality of the images. It can either be `324w` or `2975h`.
- `--columns_to_preprocess` specifies a list of columns in the dataset that should be preprocessed. In this case, the `caption` and `title_en` columns will be preprocessed.

This script creates three new columns `caption_preprocessed` and `title_en_preprocessed` containing the preprocessed expressions, and `filename` containing the filename of the images linked to the `bcu_id`.

5.1.3 Phrase Grounding Inference

As this part is the most computationally intensive and requires the most GPU memory, two Jupyter notebooks are available for execution on Google Colab, but it can also be run locally.

The first two commands run the `3_MDETR_inference.py` script to perform inference using MDETR on the `caption_preprocessed` and `title_en_preprocessed` columns of the dataset:

```
python pipeline/3_MDETR_inference.py
  --input_file data/2_preprocessing.pkl
  --output_file temp.pkl
  --expression_column caption_preprocessed
  --inference_column MDETR_caption
  --image_directory data/BCU_database/images


python pipeline/3_MDETR_inference.py
  --input_file temp.pkl
  --output_file temp.pkl
  --expression_column title_en_preprocessed
  --inference_column MDETR_title
  --image_directory data/BCU_database/images
```

The third command uses wget to download a pretrained model from a specified URL and save it to the `model` directory as `glip_large_model.pth`.

```
wget https://penzhanwu2bbs.blob.core.windows.net/data/GLIPv1_Open/
  models/glip_large_model.pth -O model/glip_large_model.pth
```

The final two commands run the `3_GLIP_inference.py` script to perform inference on the `caption_preprocessed` and `title_en_preprocessed` columns of the dataset using the pretrained GLIP model:

```
python pipeline/3_GLIP_inference.py
  --input_file temp.pkl
  --output_file temp.pkl
  --expression_column caption_preprocessed
  --inference_column GLIP_caption
  --image_directory data/BCU_database/images
  --config_file pipeline/src/GLIP/configs/pretrain/glip_Swin_L.yaml
  --weights_file model/glip_large_model.pth

python pipeline/3_GLIP_inference.py
  --input_file temp.pkl
  --output_file data/3_phrase_grounding.pkl
  --expression_column title_en_preprocessed
  --inference_column GLIP_title
  --image_directory data/BCU_database/images
  --config_file pipeline/src/GLIP/configs/pretrain/glip_Swin_L.yaml
  --weights_file model/glip_large_model.pth
```

These scripts take several arguments:

- `--input_file` specifies the input file, which is a pickle file containing the dataset on which to perform inference.
- `--output_file` specifies the output file, which is a pickle file where the resulting dataset will be stored.
- `--expression_column` specifies the column in the dataset containing the expressions to be grounded.
- `--inference_column` specifies the column in the resulting dataset where the grounded expressions will be stored.
- `--image_directory` specifies the directory containing the images associated with the records in the dataset.

- `--config_file` (for the GLIP model only) specifies the configuration file to use for the model.
- `--weights_file` (for the GLIP model only) specifies the file containing the pretrained weights for the model.

This will add four columns, for each combination of GLIP MDETR with caption or title, with the results of the inferences.

5.1.4 Postprocessing

The following command is a Python script that applies non-maximum suppression (NMS) to the values in specified columns of a dataset stored in a pickle file:

```
python pipeline/4_apply_nms.py
        --input_file data/3_phrase_grounding.pkl
        --output_file data/4_postprocess.pkl
        --columns_to_process GLIP_caption GLIP_title
                MDETR_caption MDETR_title
```

This script takes several arguments:

- `--input_file` specifies the input file, which is a pickle file containing the dataset on which to apply NMS.
- `--output_file` specifies the output file, which is a pickle file where the resulting dataset will be stored.
- `--columns_to_process` specifies a list of columns in the dataset on which to apply NMS. In this case, NMS will be applied to the `GLIP_caption`, `GLIP_title`, `MDETR_caption`, and `MDETR_title` columns.

This applies the NMS directly to the four columns containing the phrase grounding results.

5.1.5 Selection of best Phrase Grounding

The following commands are Python scripts that select the best phrase grounding for each expression in a dataset stored in a pickle file. One script performs automatic selection and the other provides a graphical user interface (GUI) for manual selection.

The automatic selection algorithm uses the expression ranking pipeline explained in section 4.2. It chooses GLIP if available, and selects the highest-ranked expression.

This command runs the `6_automatic_selection_best_phrase_grounding.py` script to perform automatic selection:

```
python pipeline/5_automatic_selection_best_phrase_grounding.py
        --input_file data/4_postprocess.pkl
        --output_file data/5_automatic.pkl
        --selection_column automatic_selection
```

This command runs the `6_manual_selection_best_phrase_grounding.py` script to provide a GUI for manual selection:

```
python pipeline/5_manual_selection_best_phrase_grounding.py
        --input_file data/4_postprocess.pkl
        --selection_column manually_selected
        --output_file data/5_manual.pkl
        --image_directory data/BCU_database/images
```

These scripts take several arguments:

- `--input_file` specifies the input file, which is a pickle file containing the dataset from which to select the best phrase groundings.
- `--output_file` specifies the output file, which is a pickle file where the resulting dataset will be stored.
- `--selection_column` specifies the column in the resulting dataset where the selected phrase groundings will be stored.
- `--image_directory` (for the manual selection script only) specifies the directory containing the images associated with the records in the dataset.

5.1.6 Segmentation

The following command is a Python script that performs segmentation on the images associated with a dataset stored in a pickle file:

```
python pipeline/6_segmentation.py
        --input_file data/5_automatic.pkl
        --output_dir data/6_segmentation
        --image_dir data/BCU_database/03_resized
        --selection_column automatic_selection
        --save_fig
        --save_segmentation_pickle
        --model_path model/model_segmentation.pth
        --save_colored_text_array
        --save_colored_text_html
        --device cuda
```

This script takes several arguments:

- `--input_file` specifies the input file, which is a pickle file containing the dataset associated with the images on which to perform text segmentation.
- `--output_dir` specifies the directory where the output of the segmentation will be saved.
- `--image_dir` specifies the directory containing the images on which to perform the segmentation.
- `--selection_column` specifies the column in the input dataset containing the selected phrase groundings.
- `--save_fig` specifies that the output images should be saved as figures.
- `--save_segmentation_pickle` specifies that each segmentation will be saved as a pickle file.
- `--model_path` specifies the file containing the pretrained model to use for segmentation.
- `--save_colored_text_array` specifies that the expression colored according to the segmentation should be added to the DataFrame.
- `--save_colored_text_html` specifies that the expression colored using HTML formatting according to the segmentation should be added.
- `--device` specifies the device to use for text segmentation. In this case, the script will use a GPU if available.

Chapter 6 Application of the pipeline to other projects

The pipeline has been successfully applied to other digitized collection projects, such as the MEL project, which focuses on digitizing books from the Musée de l'Élysée [40]. In this project, scans of the books are first processed using the Google ML Kit object detector [41] to identify and locate objects in the images. The resulting images and bounding boxes are then input into this pipeline's segmentation model for further processing. Some results are shown in Figure 6.1.

Other projects that wish to use this pipeline can refer to the detailed documentation and instructions provided in the project's repository, as well as the implementation process discussed in Chapter 5. This information should provide a strong foundation for successfully implementing the pipeline in other projects. Additionally, a demo of the segmentation model is available on Google Colab at `https://colab.research.google.com/github/tgieruc/Heritage-in-the-digital-age/blob/main/pipeline/notebooks/segmentation_demo.ipynb` for reference.

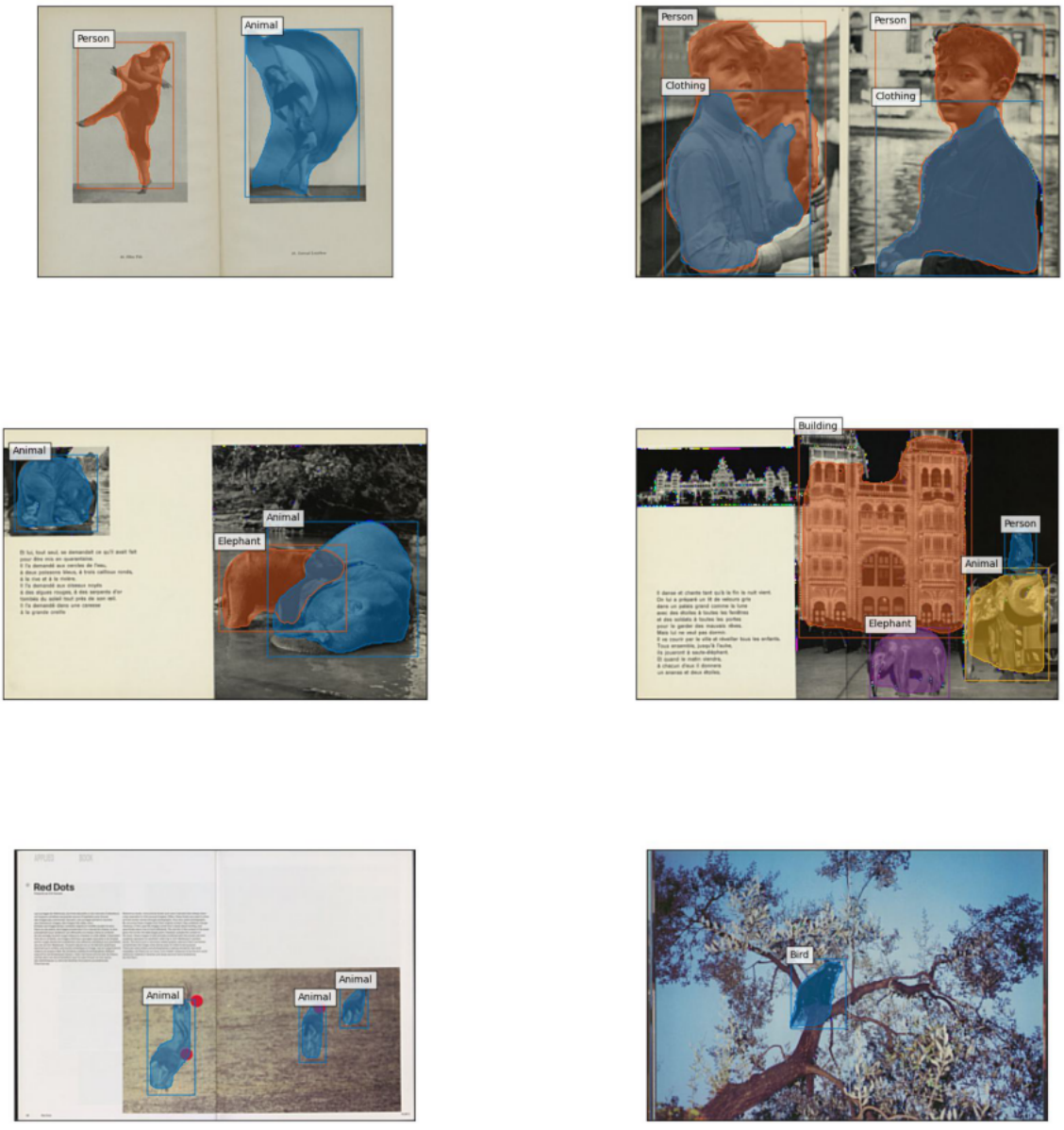

Figure 6.1: The results of applying the segmentation model on scans

Chapter 7 User survey

To assess the effectiveness of our approach, a user survey was conducted online and received 22 responses, including 6 from designers and 5 from engineers.

In the first part of the survey, participants were presented with a selection of images and asked to choose the inference that they found the most interesting from a list of five options: the combination of GLIP, MDETR, caption, title, or none. In the second part, participants were asked to rank a series of statements about the importance of different aspects in determining the most interesting inference.

Overall, the participants preferred GLIP over MDETR, and the title slightly over the caption, as shown in Table 7.1. In order to see if this is significant, we will run a t-test, with a Šidák corrected p-value of `0.0253`. The results are in Table 7.2 and show that there is a significant difference (p-value ¡ 0.0253) between MDETR and GLIP, and that people seem to prefer GLIP over MDETR.

	MDETR	**GLIP**
Caption	43	63
Title	47	79

Table 7.1: Combinations selected by participants

	t-statistic	**p-value**
MDETR over GLIP	-4.84	1.78e-6
Caption over title	-1.82	0.069

Table 7.2: Results of the t-tests

It can be interesting to look at the differences between engineers, designers and others. First we can look at their preferences of GLIP over MDETR and Caption over Title in Figure 7.1, with 95% Confidence Intervals. There seems to be no significant differences, but if we look at the difference between engineers and designers per questions, we can see that engineers seem systematically to have a higher preference to caption and MDETR than designers, as shown in Figure 7.2. However, it is important to remember that only 5 engineers and 6 designers have responded.

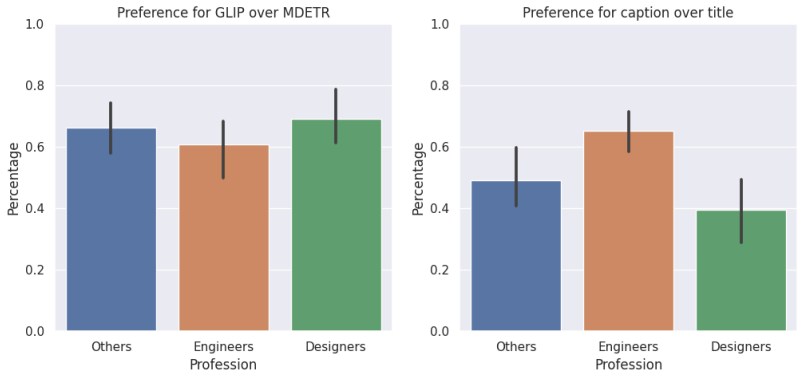

Figure 7.1: Preference of models and expressions according to profession

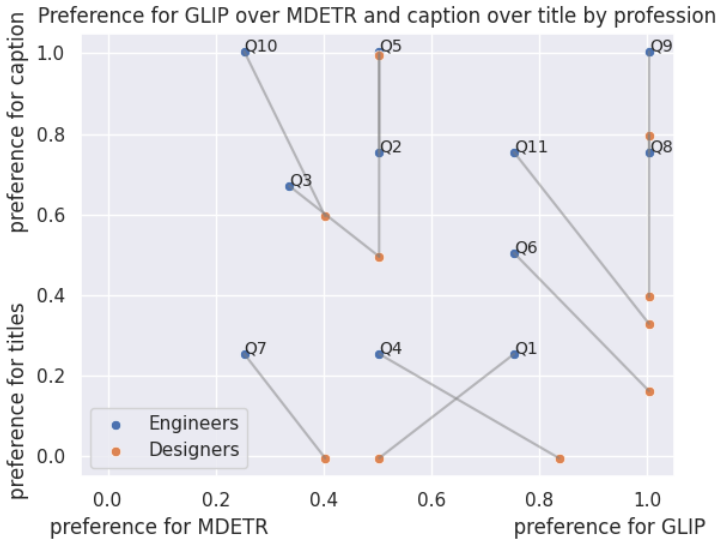

Figure 7.2: Preference of models and expressions according to profession, grouped by questions

In the second part of the survey, the following statements were ranked by the participants in order to estimate the importance of different aspects in determining the most interesting inference:

Rate the importance of the following factors in your choice of the most "interesting" inference :

1. *The degree to which the sentence accurately represents the content of the image*
2. *The precision of the segmentation in terms of dividing the image into relevant regions*
3. *The correctness of the association between the segmentations and the words*
4. *The level of detail or depth provided by the inference*
5. *The comical mismatch between the sentence and the image*

The results, grouped by profession, are displayed in Figure 7.3. As shown by the 95% confidence intervals, there is no significant differences between the different groups and questions. It is interesting to note that the 5th statement, "the comical mismatch between the sentence and the image" seems to be the most polarized.

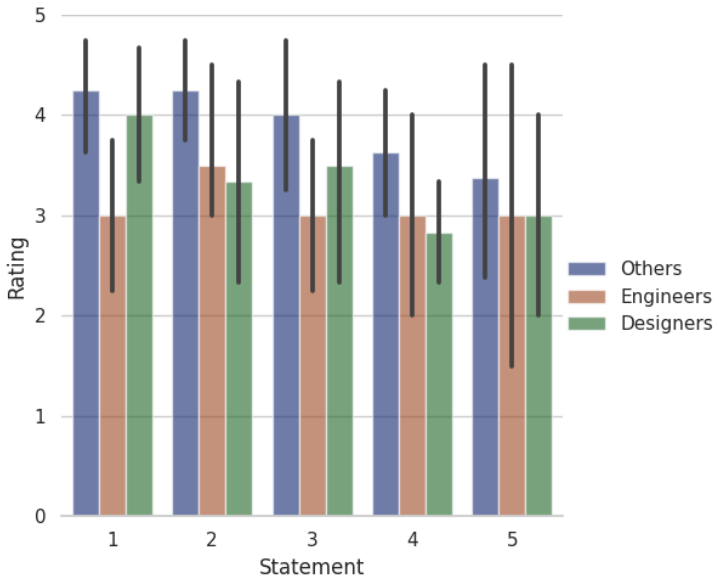

Figure 7.3: Preference of models and expressions according to profession, grouped by questions

Chapter 8 Conclusion and Future Work

In this project, the potential for valorizing digitized heritage collections was explored by augmenting images with information about the objects depicted in them using state-of-the-art computer vision and natural language processing techniques. A pipeline was developed that consists of six steps: translating the titles of images from French to English, preprocessing the captions and titles, running phrase grounding on the dataset using two state-of-the-art models, postprocessing the phrase grounding results, selecting the best results using a user-friendly GUI or an automated ranking model, and segmenting the detected objects in the images. The pipeline was applied to a dataset of 2,216 pictures from the BCUFR and the effectiveness of the approach was demonstrated.

There are several potential avenues for future work in this area. One possibility is to unify the phrase grounding and segmentation, in order to improve the performance of both tasks. GLIPv2 (2022) [42], whose implementation has not yet been released, uses this approach. The authors announced that it outperforms the previous version, GLIP, on all benchmarks when using the same pre-training data [42]. Once the code for GLIPv2 becomes available, it could be interesting to implement the model and evaluate its performance in unifying the phrase grounding and segmentation steps.

Other potential avenues for future work would be to investigate the potential applications of this approach, such as in education or tourism, and conduct user studies with larger sample sizes and more diverse datasets to further validate the effectiveness of the approach. Overall, the work highlights the potential for valorizing digitized heritage collections and creating more interesting, interactive, and fun ways to explore collections.

Bibliography

[1] Bibliothèque cantonale et universitaire. [Online]. Available: https://www.fr.ch/bcu

[2] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision." [Online]. Available: http://arxiv.org/abs/2103.00020

[3] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[4] L. Yu, Z. Lin, X. Shen, J. Yang, X. Lu, M. Bansal, and T. L. Berg, "Mattnet: Modular attention network for referring expression comprehension," 2018. [Online]. Available: https://arxiv.org/abs/1801.08186

[5] S. Kazemzadeh, V. Ordonez, M. Matten, and T. Berg, "ReferItGame: Referring to objects in photographs of natural scenes," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, pp. 787–798. [Online]. Available: https://aclanthology.org/D14-1086

[6] J. Mao, J. Huang, A. Toshev, O. Camburu, A. Yuille, and K. Murphy, "Generation and comprehension of unambiguous object descriptions." [Online]. Available: http://arxiv.org/abs/1511.02283

[7] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context." [Online]. Available: http://arxiv.org/abs/1405.0312

[8] E. Margffoy-Tuay, J. C. Pérez, E. Botero, and P. Arbeláez, "Dynamic multimodal instance segmentation guided by natural language queries," 2018. [Online]. Available: https://arxiv.org/abs/1807.02257

[9] Y.-W. Chen, Y.-H. Tsai, T. Wang, Y.-Y. Lin, and M.-H. Yang, "Referring expression object segmentation with caption-aware consistency," 2019. [Online]. Available: https://arxiv.org/abs/1910.04748

[10] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion, "Mdetr – modulated detection for end-to-end multi-modal understanding," 2021. [Online]. Available: https://arxiv.org/abs/2104.12763

[11] C. Wu, Z. Lin, S. Cohen, T. Bui, and S. Maji, "PhraseCut: Language-based image segmentation in the wild," version: 1. [Online]. Available: http://arxiv.org/abs/2008.01187

[12] X. Jia, E. Gavves, B. Fernando, and T. Tuytelaars, "Guiding long-short term memory for image caption generation." [Online]. Available: http://arxiv.org/abs/1509.04942

[13] Z. Yang, J. Wang, Y. Tang, K. Chen, H. Zhao, and P. H. S. Torr, "Lavt: Language-aware vision transformer for referring image segmentation," 2021. [Online]. Available: https://arxiv.org/abs/2112.02244

[14] L. H. Li, P. Zhang, H. Zhang, J. Yang, C. Li, Y. Zhong, L. Wang, L. Yuan, L. Zhang, J.-N. Hwang, K.-W. Chang, and J. Gao, "Grounded language-image pre-training." [Online]. Available: http://arxiv.org/abs/2112.03857

[15] Papers with code - flickr30k entities test benchmark (phrase grounding). [Online]. Available: https://paperswithcode.com/sota/phrase-grounding-on-flickr30k-entities-test

[16] D. A. Hudson and C. D. Manning, "Gqa: A new dataset for real-world visual reasoning and compositional question answering," 2019. [Online]. Available: https://arxiv.org/abs/1902.09506

[17] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and F.-F. Li, "Visual genome: Connecting language and vision using crowdsourced dense image annotations," 2016. [Online]. Available: https://arxiv.org/abs/1602.07332

[18] I. Krasin, T. Duerig, N. Alldrin, V. Ferrari, S. Abu-El-Haija, A. Kuznetsova, H. Rom, J. Uijlings, S. Popov, A. Veit, S. Belongie, V. Gomes, A. Gupta, C. Sun, G. Chechik, D. Cai, Z. Feng, D. Narayanan, and K. Murphy, "Openimages: A public dataset for large-scale multi-label and multi-class image classification." *Dataset available from https://github.com/openimages*, 2017.

[19] S. Shao, Z. Li, T. Zhang, C. Peng, G. Yu, X. Zhang, J. Li, and J. Sun, "Objects365: A large-scale, high-quality dataset for object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[20] Helsinki-NLP/opus-mt-fr-en · hugging face. [Online]. Available: https://huggingface.co/Helsinki-NLP/opus-mt-fr-en

[21] J. Tiedemann, L. Nygaard, and T. Hf, "The opus corpus – parallel and free," 01 2004.

[22] [Online]. Available: https://huggingface.co/

[23] A. Kamath, M. Singh, Y. LeCun, G. Synnaeve, I. Misra, and N. Carion, "MDETR – modulated detection for end-to-end multi-modal understanding." [Online]. Available: http://arxiv.org/abs/2104.12763

[24] S. K. Non-maximum suppression (NMS). [Online]. Available: https://towardsdatascience.com/non-maximum-suppression-nms-93ce178e177c

[25] C. Rother, V. Kolmogorov, and A. Blake, ""grabcut": Interactive foreground extraction using iterated graph cuts," in *ACM SIGGRAPH 2004 Papers*, ser. SIGGRAPH '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 309–314. [Online]. Available: https://doi.org/10.1145/1186562.1015720

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[27] P. Iakubovskii, "qubvel/segmentation_models.pytorch," original-date: 2019-03-01T16:21:21Z. [Online]. Available: https://github.com/qubvel/segmentation_models.pytorch

[28] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation." [Online]. Available: http://arxiv.org/abs/1505.04597

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition." [Online]. Available: http://arxiv.org/abs/1512.03385

[30] ImageNet. [Online]. Available: https://www.image-net.org/index.php

[31] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," 2017. [Online]. Available: https://arxiv.org/abs/1706.05587

[32] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," 2019. [Online]. Available: https://arxiv.org/abs/1905.02244

[33] COCO - common objects in context. [Online]. Available: https://cocodataset.org/#home

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[35] C. H. Sudre, W. Li, T. Vercauteren, S. Ourselin, and M. J. Cardoso, "Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations," vol. 10553, pp. 240–248. [Online]. Available: http://arxiv.org/abs/1707.03237

[36] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," 2019. [Online]. Available: https://arxiv.org/abs/1910.01108

[37] [Online]. Available: https://huggingface.co/tgieruc/Heritage-in-the-Digital-Age-expression-comparison

[38] [Online]. Available: https://huggingface.co/tgieruc/Heritage-in-the-Digital-Age-expression-ranking-pipeline

[39] [Online]. Available: https://huggingface.co/tgieruc/Heritage-in-Digital-Age-distilbert-base-uncased-expression-rating

[40] Photo elysée - musée cantonal pour la photographie. [Online]. Available: https://elysee.ch/

[41] Object detection and tracking | ML kit. [Online]. Available: https://developers.google.com/ml-kit/vision/object-detection

[42] H. Zhang, P. Zhang, X. Hu, Y.-C. Chen, L. H. Li, X. Dai, L. Wang, L. Yuan, J.-N. Hwang, and J. Gao, "Glipv2: Unifying localization and vision-language understanding," 2022. [Online]. Available: https://arxiv.org/abs/2206.05836