Characteristic Faces
Group 08 *

Tianzheng Hu †
Vrije University of Amesterdam
t.hu2@student.vu.nl

Pengju Ma ‡
Vrije University of Amesterdam
p.ma2@student.vu.nl

Bowen Liang §
Vrije University of Amesterdam
b.liang@student.vu.nl

1. INTRODUCTION

With the advancement of modern technology, various types of face recognition and facial emotion detection software are becoming more widely used, bringing great convenience to humans. The faces of people living in different regions of the world are incredibly diverse. The "Characteristic Faces" represent the typical faces of a certain region, showing its unique facial features. This project aims to generate characteristic faces from different levels of granularity (country, continent, world) using face recognition algorithms. Finally, we will visualize the results and associate the characteristic faces of each region with their corresponding locations on an interactive map to create a "World characteristic face map".

Flickr is a photo-sharing website with a large amount of photo data, the majority of which includes GPS coordinates of where the photos were taken. We use a large dataset from Flickr in this project to generate characteristic faces. Section 2 posed the research questions that this paper tries to answer. The third section describes the data format, the meaning of each data field, data size, and distribution. This section also presents the technology we used in data preparation and the entire data processing pipeline in a clear and intuitive manner. Section 4 introduces in detail the methodology and technology we chose and how they are applied to the image dataset. The fifth section describes the final data product and visualization – a static HTML web page display with strong interactivity. The final section draws a conclusion and reflects on the research questions posed. The appendix provides a summary of each team member's joint contributions.

2. RESEARCH QUESTIONS

During this project, the following three questions will be researched:

1. What is 'Characteristic face'?
2. How many images do we need for each country in order to generate characteristic faces?
3. Main question: How to generate characteristic faces using existing image dataset?

The first question is about the definition of 'characteristic face'. Generally, we believe that a characteristic face is a face containing the facial features of people in a region. How to represent it specifically will be explored during the project.

The second question tries to answer the limitation of our project so that we can limit the number of images we'll analyze and process. The number of images used to generate characteristic faces will affect the quality of the final data product, so We need to make a trade-off between them.

And the third question is the central question of our project, it is about the methodology and models we're going to use and whether those methods can generate characteristic faces. The approach and model chosen directly impact the project's final outcome.

3. DATA PREPARATION

In this section, the main research task is the cleaning and filtering of the Flickr photo datasets after figuring out the stored format of data. Then finding the qualified face images for face operation.

The raw dataset is stored on databricks in parquet format at /mnt/lsde/datasets/flickr/parquet/. The total number of stored images is 100 million and about 48.5 million images have GPS coordinates, accounting for approximately 48.5%. The size of the first parquet file is 518086449 bytes and the remaining 33 files are smaller. The total size of the 34 files is 16877246393 bytes, which is around 15.7 GB. At the beginning of the data investigation, We filtered out the data that didn't contain GPS and managed to get the data size down to 8GB.

In the data investigation phase, the first parquet file used for testing had 519,081 rows containing GPS information. To read all the data in flickr, databricks took 32.52 seconds using the query: flickrs = spark.sql("select * from flickr where lat IS NOT NULL")

Table 1 shows the meanings of the data fields and sample data. Each photo has a unique photo identifier corresponding to it.

In order to calculate the countries to which the GPS belongs by longitude and latitude, we created a grid based on a worldly map. Since the latitude range is -180 to 180 degrees and the longitude range is -90 to 90 degrees, a 180*360 grid of the world map with an accuracy of 1 was created. The countries corresponding to each pair of longitude and latitude are calculated and all 64800 pairs of latitude and longitude and their corresponding countries and continents are

Proceedings of the VLDB Endowment, Vol. , No.
ISSN 2150-8097.
DOI:

Column	Description
Photo identifier	6985418911
Photo download URL	'http://farm8.staticflickr.com/7205/6985418911_df7747990d.jpg'
Path	4e2/f7a/4e2f7a26a1dfbf165a7e30bdabf7e72a.jpg
Timestamp	2012-02-16 09:56:37.0
Latitude	81.80488586425781
Longitude	24.55055809020996
Machine tags	canon,canon+powershot+hs+310, carnival+escatay,cruise,elph,hs+310, key+west+florida,powershot
Capture device	Canon+PowerShot+ELPH+310+HS

Table 1: Raw data format

stored in a dataframe using the **reverse-geocoder** package. The world map matrix we have generated is stored in /mnt/ lsde/group08/ Country-dictionary/ MapMatrix180-360.parquet and we use the **pycountry -convert** package to convert all the country names to continents, also in the dataframe. Finally, we have a complete map of latitude, longitude, country name and continent name.

By selecting the lon, lat, url, path columns of the raw Flickr dataframe and doing a join operation with the map grid, we obtain the country and continent names corresponding to 48469829 rows. Those data are stored as a parquet file in /mnt/lsde/ group08/Country-dictionary/ SpatialInfo.parquet. The result is a total of 235 countries or regions. And All the pre-data operation codes are stored in the file PreDate.py.

Figure 1 shows the top 10 countries with the most uploaded photos. The country with the most photos is the United States, with 17.58 million photos, far more than four times the second country. The country with the least number of photos in the top ten in Canada, which also has almost 960,000.

Figure 2 shows the statistics of the number of photos of each continent. It can be found that America and Europe have the largest number of photos, about 21.46 million and 18.86 million respectively.

After obtaining the basic spatial instances data, all images are subjected to face detection using OpenCV to obtain photos that include the rough faces at first. Since the time cost of face detection, the number of face photos on each granularity is also controlled at 500. In this step, there 4 countries have too few images to detect faces. So all the rough faces URLs of 231 countries or regions are stored in /mnt/lsde/group08/Country-dictionary/ FaceImages500.parquet . After this, the images are vectorised using StyleGAN2 to generate the feature faces at each granularity. Figure 3 shows the specific pipeline and Table 2 shows the data information for every step.

4. FACE OPERATION

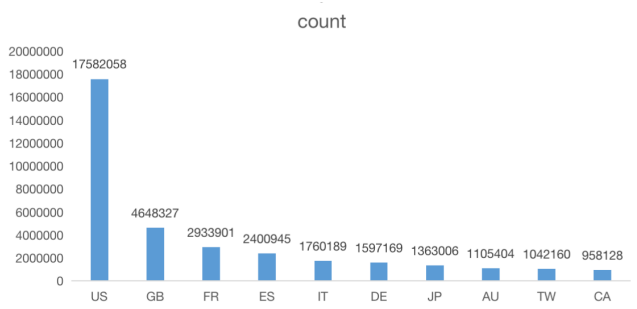

Figure 1: Top 10 countries' data distribution

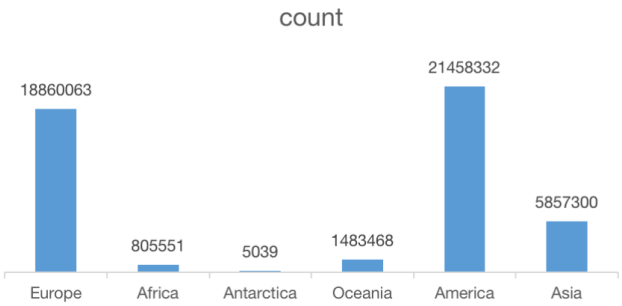

Figure 2: Continents' data distribution

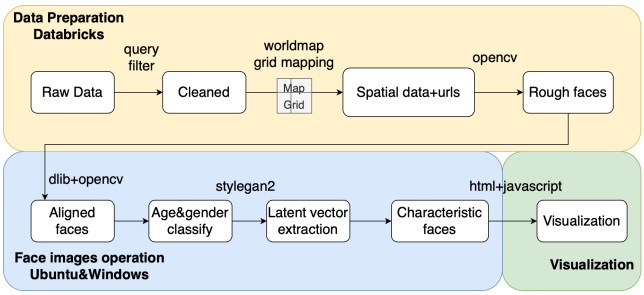

Figure 3: The data processing pipeline. The rounded squares with a black border on a white background in the picture are the status of the data. The words next to the solid line of arrows are the processing and tools performed on the data. The part with the yellow background is the part to get the appropriate urls for each country, all done on databricks, the blue background is the processing done on the image and the green is the visualisation.

In this section, we will introduce the relevant techniques and their concrete implementation. Before training the model to generate characteristic faces, we should first make sure that the input image data contains clear and frontal faces, rather than irregular photos such as landscape and side face photos.

4.1 Face detection

In the face detection part, Haar detection is used to first roughly get the qualified images rapidly. The basic principle of the Haar feature is that the pixel values of different areas

Status	Size	Instances	CTiame
Raw	15.7GB	100M	-
Cleaned+Spatial data+urls	7.53G	48M	2h
Rough faces urls	6.5MB	103K	39h
Rough faces	97.6MB	500	40mins
Aligned faces	390.6MB	500	3mins
Age/gender classify	97.6MB	500	0.5mins
Latent vector	1.44MB	40	80 mins
Characteristic faces	0.6MB	6	0.5mins

Table 2: For each data product, this table shows the size, the number of instances and deeded time to generate it. The section above the double horizontal line shows all the data products on databricks, the section below shows the data product per country because of the time limitation.

are different. It's a kind of boundary detection that is very similar to the filter of a convolutional neural network. For example, the eye and non-eye areas are very different, and this feature is used as a test criterion. Haar features are detected by subtracting the sum of pixels in the black area from the sum of pixels in the white area. The detection detects the face region from the input image and returns the coordinates of the face envelope. So the Haar detection is implemented on the databricks to get the first version of qualified images.

The URLs of all qualified images are downloaded into the local computer. Then we used Opencv and Dlib to align faces to a single face. Dlib is a toolset written in C++. Compared with deep learning libraries, Dlib internally encapsulates many traditional machine learning computational functions, such as regression analysis, support vector machines, and clustering, and provides both C++ and python interfaces externally. For face detection, the Dlib library works with the "shape-predictor-68-face-landmarks.dat" to get 68 points to mark important parts of the face [1] as Figure 4, for example, 18-22 points for the right eyebrow and 51-68 points for the mouth. It will be used in the "align-images.py" of the StyleGan2 model to align the images.

4.2 Age and Gender detection

For the age and gender identification part, two pre-trained models based on Caffe are used, one is for age prediction and another is for gender. Furthermore, the residual network of the DNN module is used in age and gender identification to detect faces even further. The steps are as followed:

1. Preload three network models
2. Load the image using video capture
3. Face detection is performed for each frame

Figure 4: Facial feature detection example.

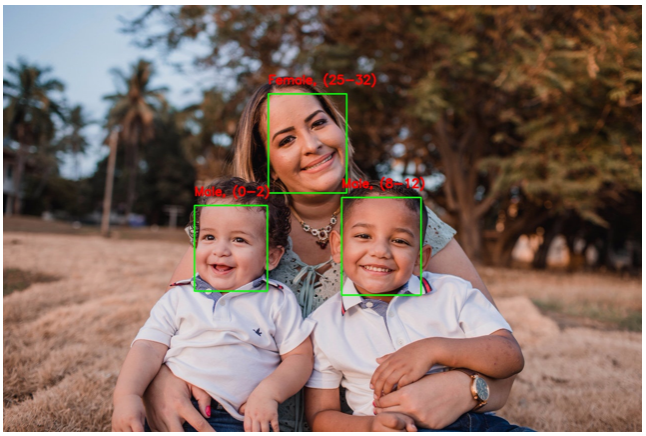

Figure 5: Sample of Age & Gender detection

4.3 Remove bad faces

To remove the bad faces that remain, there is no automatic way to filter the bag images such as scenery, side face, animals and so on. So we did this with human eyes and manually saved clear and accurate faces for the final qualified image dataset.

4.4 Extract latent vector

The bottom layer of the StyleGAN2 Encoder still relies on the StyleGAN2 core G-main class (generator), G-mapping class (mapping network), G-synthesis-stylegan2 class (synthesis network), D-stylegan2 class (discriminator), etc. StyleGAN2 is able to reproduce images similar to the images in the training set [3]. For this project, a pre-trained StyleGAN2 model from NVIDIA is used. The dataset on which the model is trained is the Flickr Faces high-quality (FFHQ) dataset, which contains 70K photos of faces from all over the world. The FFHQ set contains images of faces with some range in diversity in terms of ethnicity and age.

In the training process, the input iteration is realized by predicting and updating dlatent-avg (dlatent-avg is the average value of dlatents vector (18x512) in W-space), and noise is introduced in each iteration process, namely: Iteration input = iteration dlatent-avg + noise; At the same time, the learning rate is also changed in each iteration. The iterative input is used to generate the prediction image through

the StyleGAN2 network, and then VGG16 network is used to calculate the loss between the prediction image and the target image.

At the beginning of training, StyleGAN2 randomly generates 10,000 vectors and uses their average value as the starting point for training, simplifying the training process.

Instead of using the native methods of TensorFlow, StyleGAN2 uses dnnlib.tflib to build the optimization function. Dnnlib.tflib is a set of library functions built on TensorFlow by the NVIDIA team to facilitate code maintenance and understanding of code structure.

StyleGAN Encoder can project real faces into the dlatents space of StyleGAN, and break the restriction of 18 layers of data consistency in 18*512 vector space (i.e. Only use 1*512 vector to generate face), can maximize in the expanded vector space "fumble", the obtained dlatents optimal solution after StyleGAN model operation, the reconstructed image can be very close to the original image of the real face.

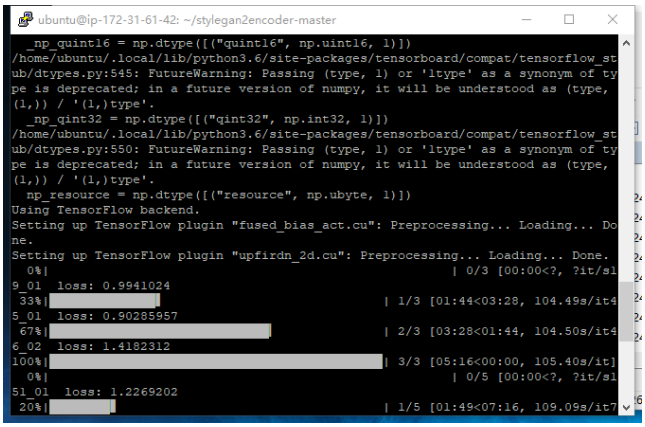

Figure 6: Sample of vector extraction

4.5 Generate the characteristic faces

The final process is to calculate the mean value of the latent vectors in each country and use the generator in stylegan2 to generate the characteristic faces.

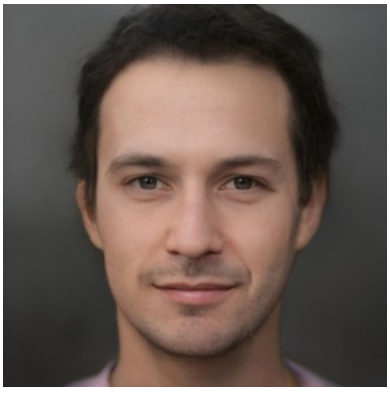

Figure 7: Sample of generating characteristic faces

4.6 Amazon Cloud Computing

StyleGAN2 requires a python 3.6 runtime environment, as well as a TensorFlow 1.14.0 environment with 12GB or more GPU cores and the cuda10.0. The P3.2xlarge Amazon EC2 in- stance was selected at AWS and includes 8 cores and 16GB of running memory, as it is the same configuration as the computer used for testing. Four machines are given to concurrently run the extraction of latent vectors. And due to the dimensions of StyleGan2Encoder, 18*512 and 500 times iteration still need 2 mins for each image.

5. RESULT

5.1 Data product

The final data product is an image dataset containing images with different parameters at different granularity. From world to continent to country, the dataset contains images of different age and gender combinations. The results images can be explored through the visualization website. A small sample of the generated characteristic faces will be shown here. Figure 8 shows four faces from four different continents. The characteristic faces of Asia, Europe, and Africa were sharper and of higher quality, while the image of Antarctica was a little blurry. This is because there are fewer face images in Antarctica compared to other continents, so the generated face image has lower accuracy and blurrier contours.

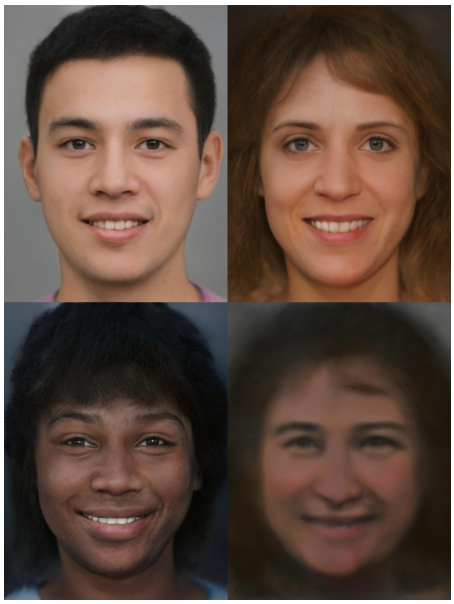

Figure 8: Four faces from four different continents with varying amounts of available source photos. Top left: Asia. Top right: Europe. Bottom left: Africa. Bottom right: Antarctica.

5.2 Visualization

5.2.1 Front-end

We used jsvectormap [2] to generate the static HTML website needed for the final visualization. Jsvectormap is a lightweight javascript library for creating interactive maps and nifty data visualizations. The HTML website consists of two components: an interactive map on the left and a dashboard on the right. The interactive map is implemented

through jsvectormap which provides almost all the basic functionality we need. Users can click on any region on the map and when the mouse hovers over a country, the name of that country will be displayed. The dashboard contains several kinds of parameters (i.e. age, gender and locality) and when people change different parameters, the generated characteristic face will also change.

5.2.2 *Back-end*

Two events will cause the displayed image to change: a different country region is clicked or a different parameter is chosen. When a country is clicked, the iso-code will be logged. We have created a dictionary that corresponds each country to the continent in which it is located. We set up some listener functions to monitor the user's requests, coding the information from the map in combination with the information from the dashboard. The same encoding is used as the name of the corresponding image in the **/static** folder, making it as simple as changing the source file of the dedicated HTML img-object to **src=/static/BFNL.jpg** to display the photo. For example, A request for both age (B) female (F) from the Netherlands (NL) is encoded as 'BFNL' as shown in figure 8. A request for both age (B) and female (F) from Europe (EU) is encoded as 'BF_continent_EU' as shown in figure 9. A request from both age (B) and female (F) from the world is encoded as 'BF_world' as shown in figure 10.

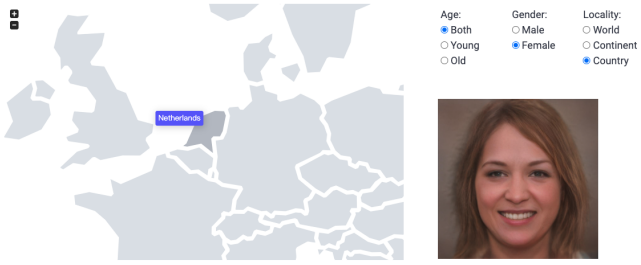

Figure 9: BFNL.jpg

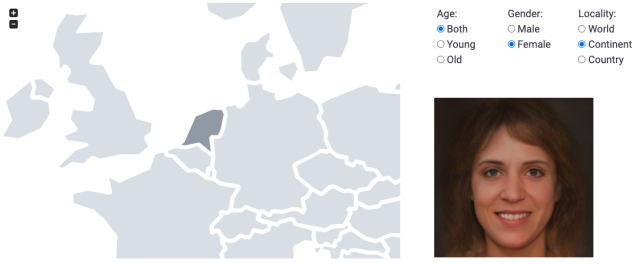

Figure 10: BF_continent_EU.jpg

6. DISCUSSION

There are some possible areas for improvement for this project.

Firstly, in the detection of rough faces, OpenCV did not show satisfactory results. There were still many images where no faces were detected as the greyscale could easily be misinterpreted. There are some paintings, sculptures

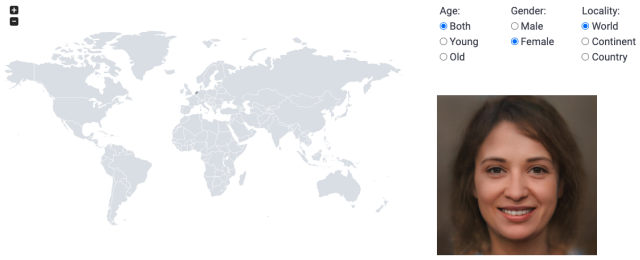

Figure 11: BF_world.jpg

or even scenic photos were been defined as faces. In addition, it was difficult to control the overall colour palette for greyscale detection, so there are some images too dark to affect the generation of result faces. As a result, the team had to manually remove these images.

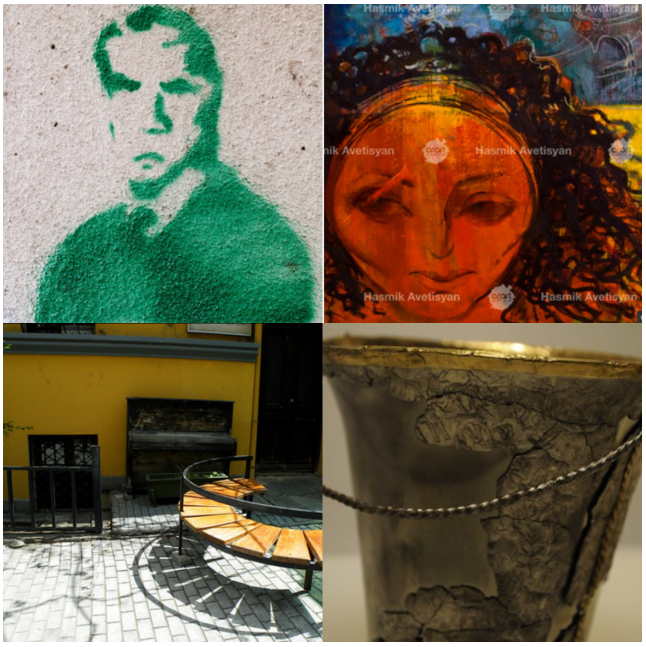

Figure 12: Examples of typical no faces image but face detection algorithms filtered out. Painting and Stickers or paintings of human figures, landscape photographs with complex compositional elements, photographs of still lifes shaped like human faces.

Secondly, the distinction between genders is not particularly obvious when generating characteristic faces. The difference between men and women is sometimes only in the length of the hair. So even when BOTH faces are generated, there is not much difference between the two genders alone.

Thirdly, acquiring the vectors was extremely time-consuming. In obtaining the vectors, the team tried to obtain more vectors of face images to achieve a variety of variations, but it also took a lot of time to obtain more face images. It would have taken up to two to three weeks of work to complete each country. Improving the model for this step or using more advanced methods would help to improve efficiency.

Finally, for some countries or regions with a very low number of uploaded photos, the project was discarded during

the project due to the inability to detect rough-face photos. However, for countries where there are still too few images for face manipulation, face images from countries or regions around the region are used. However, this method gives up some accuracy and relies only on the similarity of human looks on neighboring territories.

7. CONCLUSION

The goal of our project is to generate characteristic face images with different parameters for different levels of granularity (world, continent, country) and visualized the data product in a static HTML website.

To start, we explored the original datasets, analyzing the data format and data size. Before evaluating the data distribution, we filtered the datasets and removed images that did not contain GPS coordinates. A detailed pipeline diagram shows all the steps and methods we use to process the data. Through experiments, we finally decided to prepare 30-50 clear and usable face images for each country to generate characteristic faces, answering the second research question.

After the processing of the data is completed, we start to operate on the faces in the images. In the first step, we used Haar detection to recognize the face in the images. Secondly, two pre-trained models are implemented to identify age and gender. After these two steps, we manually removed the faces of bad quality and used the StyleGAN2 model to extract the latent vectors of each face. Finally, the mean value of the latent vectors in each country is calculated to generate the characteristic faces. The first research question was answered by defining a characteristic face as the average latent vector of a region. And the methodology explains how to generate the goal face images, which is our main research question.

For the visualization part, we created a static HTML website through Javascript. With an interactive map on the left and a dashboard on the right, the website allows users to see various characteristic faces by clicking on different regions or changing the parameter configuration. The methods and models we chose can generate images of good quality and allow users to see the facial diversity of people in different regions.

Code Work	**Team member**
Data preparation	Tianzheng Hu
Face operation	Pengju Ma
Visualization	Bowen Liang
Report	**Team member**
Introduction, Research question, Result, Conclusion	Bowen Liang
Data Preparation, Discussion	Tianzheng Hu
Face operation	Pengju Ma

Table 3: Division of work during project

8. REFERENCES

[1] Nataliya Boyko, Oleg Basystiuk, and Nataliya Shakhovska. Performance evaluation and comparison of software for face recognition, based on dlib and opencv library. In *2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP)*, pages 478–482. IEEE, 2018.

[2] Mustafa Omar. Jsvectormap. https://github.com/themustafaomar/jsvectormap.

[3] Yuri Viazovetskyi, Vladimir Ivashkin, and Evgeny Kashin. Stylegan2 distillation for feed-forward image manipulation. In *European conference on computer vision*, pages 170–186. Springer, 2020.

9. APPENDIX

The division of work among the group members is shown in Table 3.