Horizon 2020

Understanding Europe's Fashion Data Universe

Product Taxonomy Linking

Deliverable number: D5.2

Version 2.0

Funded by the European Union's Horizon 2020 research and innovation programme under Grant Agreement No. 732328

Project Acronym:	FashionBrain
Project Full Title:	Understanding Europe's Fashion Data Universe
Call:	H2020-ICT-2016-1
Topic:	ICT-14-2016-2017, Big Data PPP: Cross-sectorial and cross-lingual data integration and experimentation
Project URL:	https://fashionbrain-project.eu

Deliverable type	Report (R)
Dissemination level	Public (PU)
Contractual Delivery Date	31 December 2018
Resubmission Delivery Date	4 February 2019
Number of pages	18, the last one being no. 12
Authors	Marko Jocić, Matthias Dantone - Fashwell
Peer review	Alan Akbik - Zalando Alessandro Checco - USFD

Change Log

Version	Date	Status	Partner	Remarks
0.1	07/12/2018	Draft	Fashwell	
0.2	13/12/2018	Full Draft	Fashwell	
1.0	20/12/2018	Final	Fashwell, Zalando, USFD	Rejected 30/01/2019
2.0	04/02/2019	Resubmitted Final	Fashwell, Zalando, USFD	

Deliverable Description

This tasks extends T5.1 by defining an attribute taxonomy of fashion products and extracts these attributes automatically, using computer vision and deep learning techniques. These are demonstrated through publicly available demonstration website.

Abstract

In this deliverable we show how we developed a deep learning classifier which automatically extracts attribute tags from fashion products. Developing such a classifier consists of a few steps: identifying relevant attribute groups for each fashion product category, collecting data, cleaning data, training a model and introducing some post-processing steps to make sure the classifier is production ready.

We identified 36 different attribute groups, 4 of them are general (applicable for every fashion product) and 32 which are category specific. Within these 36 attribute groups, a total of 574 relevant attribute tags were identified.

To collect training data for our deep learning classifier, we utilized proprietary product feeds and also Google image search. To reduce the noise in the collected data, we clean it by using crowdsourcing. In the end, the total number of collected data points exceeds half a million.

We train a classifier in a multi-task, multi-label setting. After the training, we address some failure cases by introducing post-processing of classifier's raw outputs. Our model achieves average accuracy of 82%.

After the classifier is trained, it is utilized to extract attributes from products in fashion blogger dataset, which was a result of deliverable D5.1. This resulted in a publicly available demonstration website. Additionally, statistics on distribution of extracted attribute tags on the dataset are reported.

Table of Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

API	Application Programming Interface
GDPR	General Data Protection Regulations
ROI	Region of Interest
SGD	Stochastic Gradient Descent

1 Introduction

As previously shown in D5.1, social media and influencers have transformed the fashion landscape by becoming an advertising and marketing source for brands and retailers. In this deliverable, we delved a step deeper into the products we see in images to extract attribute information about every product in the form of text labels. It may be relevant for a shopper to see a similar sneaker to that they see in social media image, however it may be even more interesting for a retailer to understand deep levels of product information in the form of text that could be aggregated and understood for trend scouting, seasonal buys or understanding from a much more concrete perspective - what products are popular in images on social media. The product tagging technology does just this, it analyzes an image to assign keywords and phrases to products.

For this deliverable we took the same Instagram fashion bloggers dataset which resulted from D5.1 and analyzed the attributes present in these products. This is made public on the following website: https://fashionbrain-project.eu/product-taxonomy-linking/.

Scope This deliverable extends the work of D5.1 by extracting attributes of detected fashion products in Instagram images.

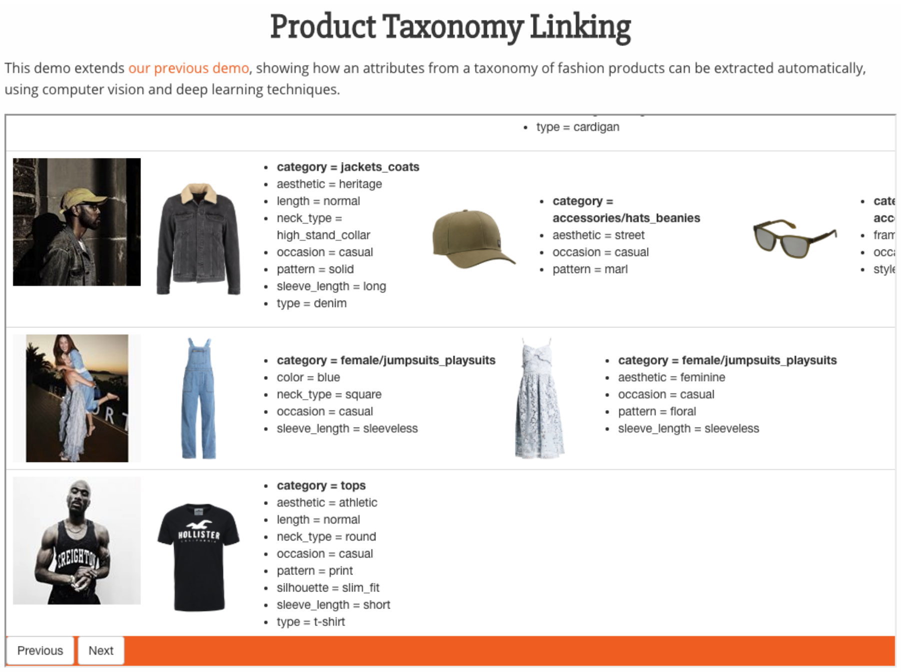
Dependencies This deliverable is using the dataset resulting from deliverable D5.1.

Contribution This deliverable will contribute to all the tasks concerning fashion trend prediction (D5.3, D5.4, D5.5).

Datasets This deliverable is providing a new dataset by enriching the dataset that resulted from D5.1.

2 Demonstration Website

The demonstration website (https://fashionbrain-project.eu/product-taxonomy-linking) displays the images with all the identified products, as well as attributes each product. Figure 2.1 shows an example screenshot of the website.



Product Taxonomy Linking

This demo extends our previous demo, showing how an attributes from a taxonomy of fashion products can be extracted automatically, using computer vision and deep learning techniques.

- type = cardigan

- **category = jackets_coats**
- aesthetic = heritage
- length = normal
- neck_type = high_stand_collar
- occasion = casual
- pattern = solid
- sleeve_length = long
- type = denim

- **category = accessories/hats_beanies**
- aesthetic = street
- occasion = casual
- pattern = marl

- **cate**
  acc
- fram
- occa
- style

- **category = female/jumpsuits_playsuits**
- color = blue
- neck_type = square
- occasion = casual
- sleeve_length = sleeveless

- **category = female/jumpsuits_playsuits**
- aesthetic = feminine
- occasion = casual
- pattern = floral
- sleeve_length = sleeveless

- **category = tops**
- aesthetic = athletic
- length = normal
- neck_type = round
- occasion = casual
- pattern = print
- silhouette = slim_fit
- sleeve_length = short
- type = t-shirt

Previous Next

Figure 2.1: Screenshot of the publicly available website that shows the collected fashion blogger images, annotated products and their attributes.

This website demo is compliant with the General Data Protection Regulations (GDPR)[1]. Of special note, the consortium does not directly host these Instagram images, instead they are served by Instagram. More specifically, we follow Article 17 of GDPR [1], which concerns the user's rights to data erasure. This means if a user chooses to delete his/her data from Instagram, it will also be automatically deleted from this demonstration website.

Moreover, a Consent Manager[2] has been introduced to allow influencers to withdraw consent, should they wish to do so.

[1]We refer to Deliverable D9.3 for more information.

[2]https://fashionbrain-project.eu/data-ethics-and-privacy, see Deliverable D9.2 and D9.3 for detailed information.

3 Attributes

The extracted attributes are either general for all fashion objects or specific for one or several fashion product categories. For every category there are different attribute tags that are relevant. Take for example tops: tops have silhouettes and length whereas a shoe is described by a heel height, fastener and ankle height. On the other hand, color, pattern and occasion are general attributes that are relevant for any fashion product. We have built a classifier which is able to automatically extract such tags from images. We will get into how we collect data and train the classifier to perform such task but for now take a look at the attributes grouped by category that are made available for tagging:

- **General attributes**: color, pattern, occasion, aesthetic
- **Tops&Blouses**: neck type, sleeve length, silhouette, type, length
- **Pants&Trousers**: type, length, cut, denim detail
- **Skirts**: silhouette, length
- **Dresses**: silhouette, length, sleeve length, neck type
- **Jackets&Coats**: type, length, sleeve length, neck type
- **Bags**: type, strap type, size, closer, leather type
- **Eyewear**: style, frame
- **Shoes**: type, heel type, heel height, toe type, ankle height, fastener

4 Data Collection

The first step in creating the attribute classifier is to identify relevant attribute data and execute data collection. This means collecting at least few thousands of images per category sub-tag, labeling them and verifying the data set is comprehensive before kicking off training. We took a few steps in order to collect the necessary data.

1. Due to our access to many shop feeds and product descriptions we can collect product images that match the relevant tag. We use keyword matching to mine the descriptions for a set of words that likely will match a tag. For example, for animal print we would look for keywords like "zebra" and "leopard". Once a keyword was identified this image could be assigned to the Pattern - animal bucket.
2. In order to insure high quality of collected data, we had to make sure that the keywords we chose to sample certain attributes are relevant. We did this by conducting a small scale user study where we asked different subjects to describe certain products by attributes. These results were combined with our expert knowledge of the fashion landscape to obtain the final keywords.
3. In cases where we had difficulty collecting sufficient amount of data from shop feeds, we used Google image search with keywords to collect additional data. For example searches such as "red dress, red pants, red shoes" were done to collect image data for the color sub-tag "red".
4. After collecting sufficient amount of data, we had to ensure that all the selected images were good ground truth examples. In order to do this we had to clean the collected data. We have developed an internal tool to view multiple images at a time and then easily manually remove those that do not correspond to a certain tag. Data cleaning was done by Upwork [3] crowd which were specifically trained for this task.
5. The whole data collection and later data cleaning process had several iterations, as we had ensure a good distribution of different image types per every category. For example, we want similar number of wild images, professional model photos and in-shop clean product shots. Another example is when collecting images of high-top shoes - we need to make sure we collect examples of high-top boots, but also examples of high-top sneakers. This is important so the classifier can recognize the attributes in a variety of different image type scenarios.

We managed to collect a dataset with more then half million images through the described process. Table 4.1 shows number of images collected per attribute group.

Table 4.1: Statistics on collected images for each attribute group.

Attribute group	No. of images	No. of tags
class_attributes/bags/closer	10966	5
class_attributes/bags/leather-type	1588	8
class_attributes/bags/size	7179	5
class_attributes/bags/strap-type	8788	22
class_attributes/bags/type	20490	6
class_attributes/dress/length	4856	9
class_attributes/dress/silhouette	29670	10
class_attributes/eyewear/frame	5372	4
class_attributes/eyewear/style	11896	16
class_attributes/jacket-coat/length	4474	4
class_attributes/jacket-coat/type	27272	13
class_attributes/pants/cut	7226	6
class_attributes/pants/denim-detail	6601	4
class_attributes/pants/length	7704	8
class_attributes/pants/type	11450	13
class_attributes/shoes/ankle-height	7997	3
class_attributes/shoes/fasteners	8255	6
class_attributes/shoes/heel-height	4683	10
class_attributes/shoes/heel-type	7280	3
class_attributes/shoes/toe-type	7181	4
class_attributes/shoes/type	25706	12
class_attributes/skirt/length	7256	38
class_attributes/skit/silhouette	9843	3
class_attributes/tops/length	5991	9
class_attributes/tops/silhouettes	6029	8
class_attributes/tops/type	18078	5
general_attributes/aesthetics	16266	4
general_attributes/color	108076	13
general_attributes/neck_type	60834	6
general_attributes/occasion	20511	23
general_attributes/pattern	31891	4
general_attributes/sleeve_length	25791	4
Total	537200	288

5 Classifier

5.1 Training a Classifier

With the images data collected and all tagged with the appropriate tag, the classifier is ready to be trained. In total we have 537200 labeled images which we split 70%, 20% and 10% for the train, validation and test sets respectively. We utilized Keras, an API that allows us to build and train models in TensorFlow.

We describe some details on how we train the attribute classifier model:

- We use ResNet-50 [2] as the backbone model, and we add a classification head for each attribute group made up of a fully connected layer followed by a softmax. Essentially, we create a multi-task, multi-label classifier. Multi-task means we can train a single sample on multiple attribute groups simultaneously (e.g. color and pattern). Multi-label means each sample can have multiple ground truths for one attribute group (e.g. a t-shirt doesn't have to be in one color, but in multiple). This is implemented by calculating loss weights for each sample - the weights make sure the loss is balanced between different classes, and also that loss is 0 (zero) for all the outputs that are not present in the sample.
- We use Stochastic Gradient Descent (SGD) to optimize parameters the model, where each mini-batch consists of 64 samples. We train the model with learning rate of 0.01.
- Images and their labels are fed into the model as input and output. For each image not all attribute groups have a ground truth annotated (e.g. for shoes we don't have any ground truth values for "sleeve length" attribute). During training we make sure the model does not back-propagate a loss for the attribute groups that do not have a ground truth label.
- The algorithm is evaluated by averaging accuracy per attribute group and also per sample. For attribute groups that perform poorly, we try to improve performance by collecting more data with clearly defined distinctions between borderline cases (for example defining the difference between casual sneakers and sports sneakers).

Our attribute classifier achieves accuracy averaged per attribute group of 82% on the test data split. Table 5.1 shows accuracy score for each attribute group. It is interesting to note that even though quantitatively "color" attribute has one of the lowest performances, we observe that qualitatively it yields very accurate results. We claim that this is due to the fact that we have multiple attribute tags for each color (e.g. blue/dark, blue/medium, blue/light) and thus most of the classifier's confusion is in these kind of cases.

Table 5.1: Attribute classifier accuracy per attribute group (class)

Attribute group	Accuracy (%)
class_attributes/bags/closer	72
class_attributes/bags/leather-type	94
class_attributes/bags/size	93
class_attributes/bags/strap-type	83
class_attributes/bags/type	85
class_attributes/dress/length	87
class_attributes/dress/silhouette	80
class_attributes/eyewear/frame	88
class_attributes/eyewear/style	75
class_attributes/jacket-coat/length	93
class_attributes/jacket-coat/type	85
class_attributes/pants/cut	72
class_attributes/pants/denim-detail	77
class_attributes/pants/length	75
class_attributes/pants/type	82
class_attributes/shoes/ankle-height	93
class_attributes/shoes/fasteners	91
class_attributes/shoes/heel-height	82
class_attributes/shoes/heel-type	89
class_attributes/shoes/toe-type	85
class_attributes/shoes/type	88
class_attributes/skirt/length	88
class_attributes/skirt/silhouette	82
class_attributes/tops/length	82
class_attributes/tops/silhouettes	80
class_attributes/tops/type	85
general_attributes/aesthetics	56
general_attributes/color	63
general_attributes/neck_type	83
general_attributes/occasion	83
general_attributes/pattern	79
general_attributes/sleeve_length	90

5.2 Post-processing of Classifier Outputs

Before being able to use the classifier in a production setting, we first need to do some post-processing of its raw outputs. This mainly arises from the fact that during test time, the model is not provided with a ground truth value and basically is not restricted to predict invalid attribute for particular product. For example, there is no constraint in the model itself to make sure it does not output sleeve length values for a shoe product. Also, if a classifier is not confident enough about a certain prediction, we would prefer not to show it. These constraints have to imposed explicitly by a human expert.

We list some examples of the the post-processing steps that are executed on the classifier's raw outputs:

- **Filter the attribute groups depending on the fashion product category**. For example if the detected category is shoes then we only return shoe type, ankle-height, color etc. and do not return for example neck type or sleeve length.
- **Filter results based on a threshold**. We return only the tags where the probability returned by the model is over a certain threshold. We define these thresholds manually through some visual evaluation. Typically the threshold is somewhere between 0.3 to 0.5. This way it can happen that the model returns multiple tags per attribute group yet often they can both be correct or borderline (e.g. returning red and orange red when the color of a product is borderline).
- **Filter tags based on the location of the Region of Interest (ROI) with respect to the image**. We do this to handle cases where products are not fully visible in the image. For example, the blouse is most likely not fully visible when the ROI touches the top of the image and thus we don't show the value of "neck type" attribute.

5.3 Results on the Fashion Blogger Dataset

We used the attribute classifier to extract attributes for all products found in images in our Instagram fashion blogger dataset. In this section we show distribution of predicted attributes for a few general attributes and also sunglasses "style" attribute.

Figure 5.1 shows the distribution of extracted color attribute tags in our dataset. We can see that the most popular colors by far are white and black, followed by blue and red. The least present garment colors are silver, metallic, purple and turquoise, as well as multi-colored fashion products.

We did the same analysis for pattern (Figure 5.2), occasion (Figure 5.3) and sunglasses style (Figure 5.4). These histograms tell us that the most fashion products in Instagram images have solid pattern and are worn casually. Sunglasses styles which are currently "in" are round, aviator and butterfly, which is not the case for rectangular sunglasses.

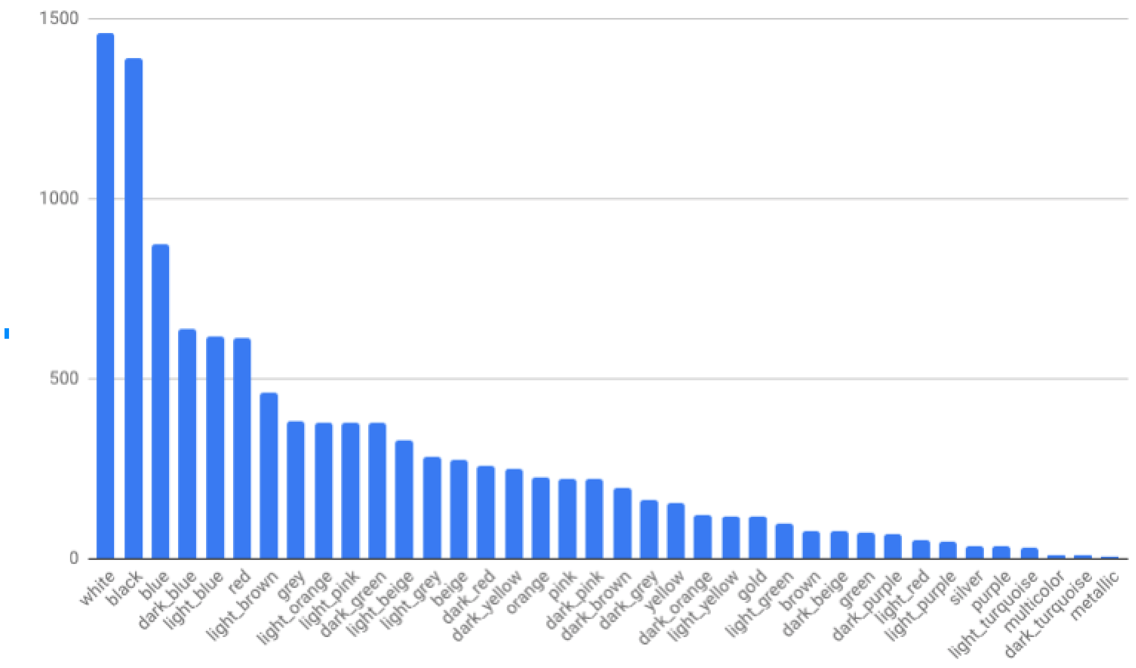


Figure 5.1: Distribution of **color** attribute tags in our dataset.

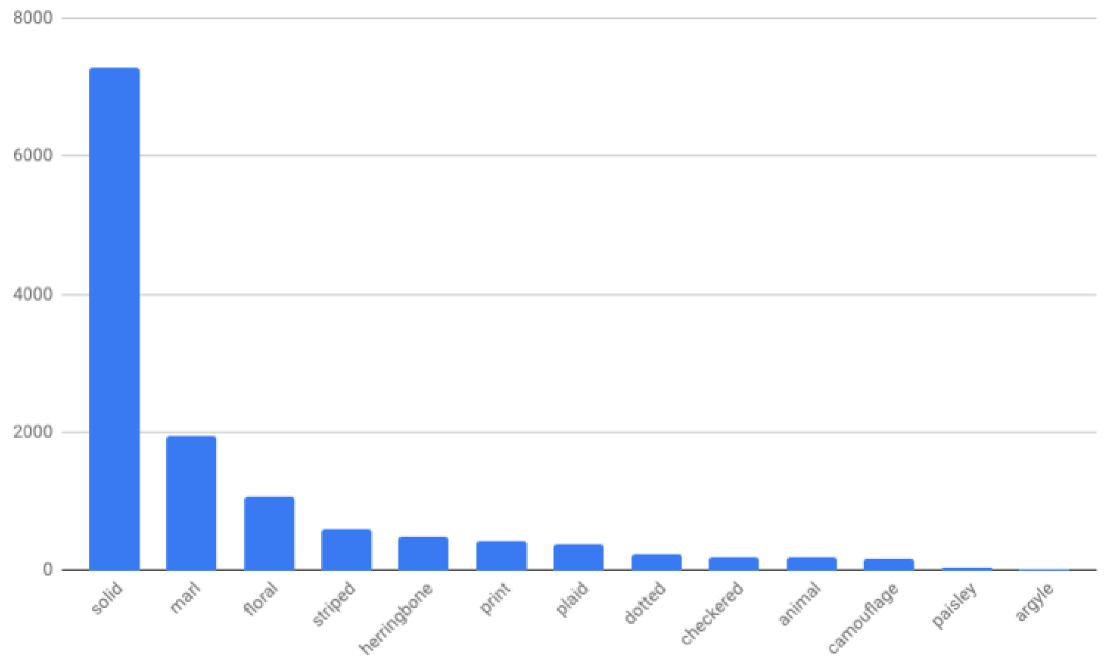


Figure 5.2: Distribution of **pattern** attribute tags in our dataset.

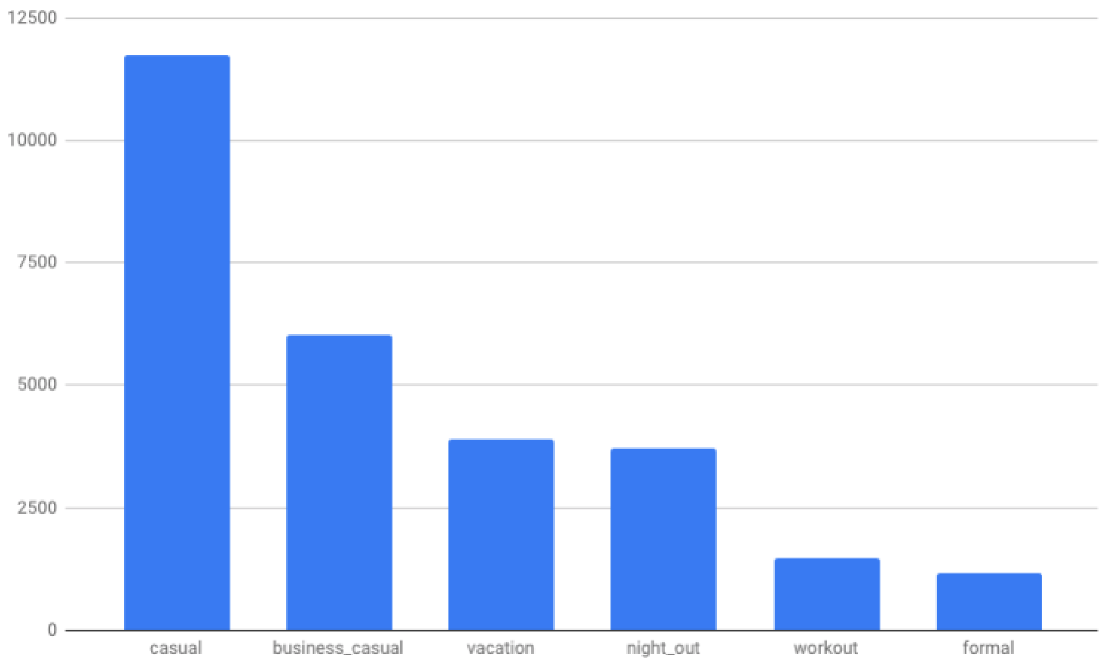


Figure 5.3: Distribution of **occasion** attribute tags in our dataset.

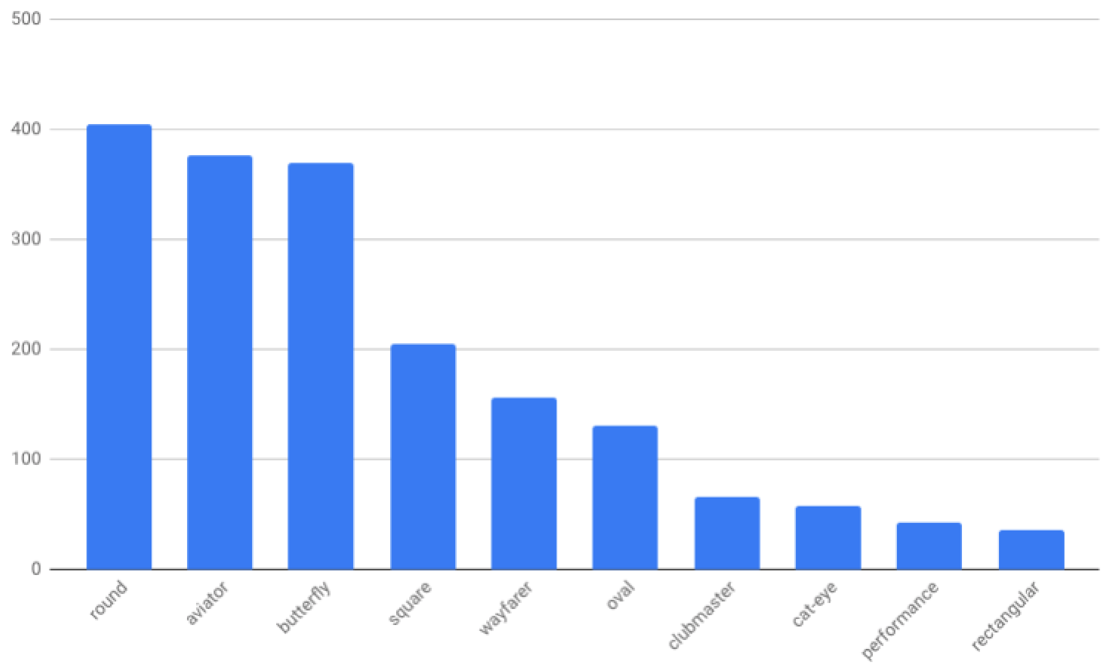


Figure 5.4: Distribution of **sunglasses style** attribute tags in our dataset.

Bibliography

[1] GDPR. General data protection regulations, 2018. URL https://ec.europa.eu/commission/priorities/justice-and-fundamental-rights/data-protection/2018-reform-eu-data-protection-rules_en.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

[3] Upwork. Upwork crowdsourcing platform, 2018. URL https://www.upwork.com.