A Framework for Fashion Data Gathering, Hierarchical-Annotation and Analysis for Social Media and Online Shop

TOOLKIT FOR DETAILED STYLE ANNOTATIONS FOR ENHANCED FASHION RECOMMENDATION

UMMUL WARA
Abstract

Due to the transformation of different recommendation system from content-based to hybrid cross-domain-based, there is an urge to prepare a social-network dataset which will provide sufficient data as well as detail-level annotation from a predefined hierarchical clothing category and attribute based vocabulary by considering user interactions. However, existing fashion-based datasets lack either in hierarchical-category based representation or user interactions of social network. The thesis intends to represent two datasets- one from photo-sharing platform Instagram which gathers fashionistas images with all possible user-interactions and another from online-shop Zalando with every cloths detail. We present a design of a customized crawler that enables the user to crawl data based on category or attributes. Moreover, an efficient and collaborative web-solution is designed and implemented to facilitate large-scale hierarchical category-based detail-level annotation of Instagram data. By considering all user-interactions, the developed solution provides a detail-level annotation facility that reflects the user's preference. The web-solution is evaluated by the team as well as the Amazon Turk Service. The annotated output from different users proofs the usability of the web-solution in terms of availability and clarity. In addition to data crawling and annotation web-solution development, this project analyzes the Instagram and Zalando data distribution in terms of cloth category, sub-category and pattern to provide meaningful insight over data. Researcher community will benefit by using these datasets if they intend to work on a rich annotated dataset that represents social network and resembles in-detail cloth information.

Keywords

Web-scraper; image-annotation; instagram; zalando; data pre-processing framework; customized-crawler; online-shopping website; deep fashion-recommendation
Sammanfattning


Nyckelord

Web-scraper; image-annotation; instagram; zalando; data pre-processing framework; customized-crawler; online-shopping website; deep fashion-recommendation
Acknowledgement

I would like to express my deep thanks to all individuals who help me directly or indirectly to complete this thesis work.

Foremost, I would like to express my sincere gratitude to my thesis examiner Prof. Mihhail Matskin, for his guidance, support and efforts throughout the project. His earnest observation towards the work provides great inspiration to improve the work.

Special thanks should be given to my supervisor Shatha Jaradat, who keeps faith on me by assigning this work. It was a great pleasure for me to work with her by receiving continuous encourages, constructive advices and useful feedbacks along the way of this research. Moreover, support with additional resource and willingness to give her time in a flexible manner are so much appreciated.

I would also like to extend my thanks to my colleagues- Mallu Goswami and Kim Hammer for their wonderful ideas and collaborations.

At last but not the least, my heartiest thanks goes to my parents, family and siblings, specially to my husband whose initiative encouraged me to pursue this master study and my daughter who sacrificed her time.
# Table of Contents

1 Introduction .................................................................................................................. 1  
1.1 Background .............................................................................................................. 1  
1.2 Problem ..................................................................................................................... 1  
1.3 Purpose ..................................................................................................................... 1  
1.4 Goal ........................................................................................................................... 2  
  1.4.1 Benefits, Ethics and Sustainability ...................................................................... 2  
1.5 Methodology .............................................................................................................. 2  
1.6 Delimitations ............................................................................................................. 3  
1.7 Outline ....................................................................................................................... 3  

2 Theoretical Background and Related Work ................................................................. 4  
2.1 Fashion Data Sources ............................................................................................. 4  
2.2 Available Fashion Datasets ..................................................................................... 4  
2.3 Comparison of Existing Fashion Datasets .................................................................. 6  
2.4 Annotation Approaches ........................................................................................... 7  
2.5 Requirement of Manual Web Annotation Solution .................................................. 8  

3 Data Collection, Preprocessing, Store and Developed Tools ................................. 10  
3.1 Data Collection Methods for Instagram and Zalando ............................................. 10  
  3.1.1 Instagram ............................................................................................................. 10  
  3.1.2 Zalando ............................................................................................................... 13  
3.2 Data Storage, Representation and Management ..................................................... 13  
  3.2.1 Data Storage ...................................................................................................... 13  
  3.2.2 Data Structure ................................................................................................... 15  
3.3 Developed Tools for Data Collection to Store ......................................................... 18  
  3.3.1 Web Solution to Consume Instagram API .......................................................... 18  
  3.3.2 Customizable Zalando Scraper .......................................................................... 19  

4 Detail-level Image Annotation Web Solution ............................................................. 24  
4.1 System Requirement ............................................................................................... 24  
4.2 Brief Description of System / System Design ......................................................... 25  
4.3 Framework Selection, Preprocessing and Implementation ..................................... 26  
  4.3.1 Framework Selection ......................................................................................... 26  
  4.3.2 Pre-processing and Database Prepare ............................................................... 27  
  4.3.3 Data Structure .................................................................................................. 29  
  4.3.4 Implementation .................................................................................................. 31  
4.4 Deployment and Secure Access .............................................................................. 33  
  4.4.1 Apache Server .................................................................................................... 33  
  4.4.2 HTTPSs secure access using Let’s Encrypt ....................................................... 34  

5 Evaluation and Analysis ............................................................................................. 36  
5.1 Statistics of Datasets ............................................................................................... 36  
5.2 Evaluation of Detail-level Image Annotation Web-service ...................................... 37  
5.3 Instagram API Application ...................................................................................... 42  
5.4 Analysis of Instagram Dataset and Zalando Dataset .............................................. 44  
  5.4.1 Analysis in Item-Category Level: ..................................................................... 44  
  5.4.2 Analysis in Attribute Level: ............................................................................. 45  
  5.4.3 Comparison of Zalando and Instagram Dataset: .............................................. 47  

6 Conclusion and Future Work ..................................................................................... 49  

7 References .................................................................................................................. 50
Appendix A

8.1 Detail-level Annotation Web-solution Web Interface:

- 8.1.1 Login page:
- 8.1.2 All fashionista Album Page:
- 8.1.3 Individual fashionista Album:
- 8.1.4 Image Annotation User Interface:
- 8.1.5 Complete Image Annotation view:
- 8.1.6 Image Annotation User Interface for Crowd Sourcing Worker:
List of Tables
Table 1. Comparing FashionRec with of different existing fashion....................7
Table 2. Summary of Image Annotation Methods towards the creation of a ground truth dataset.................................................................8
Table 3. Brief description of database and collection representation for both datasets. .................................................................15
Table 4 Brief description of user stories of Detail-level Image annotation tool.24
Table 5 List of the selected technology and their reason to choose. ............27
Table 6 Pseudo code for Dynamic Image assignment Algorithm. ...........28
Table 7 Statistics of Instagram and Zalando dataset ................................36
Table 8 Steps to consume Instagram API by developed web-application. ....46

List of Figures
Figure 1. Instagram Endpoints and their requested data.........................10
Figure 2. Fashion Dataset Retriever client view in Instagram Developer page.11
Figure 3. A snapshot of developed web-solution 'Fashion Dataset Retriever'..11
Figure 4. A set of requirement before any Instagram Permission Review request, taken from Instagram.................................................................12
Figure 5. An example of document that resides in a collection in MongoDB, extracted from [21]..................................................................14
Figure 6. Reference data structure, exported from [22]..................................15
Figure 7. An example of Embedded data structure, taken from [22]..............15
Figure 8. Embedded data structure for 'liketoknowit' and ‘useridtable’ collection documents. ..........................................................................................16
Figure 9. Embedded data structure for zalandoproductdetails’ collection............................17
Figure 10. Flow-chat of Instagram-scraper tool............................................19
Figure 11 Sample code snippet for python HTTP request and HTTP response parsing and element searching using Beautiful Soup. .....................20
Figure 12 Zalando web-shop’s cloth sub Category list retrieve from html.....21
Figure 13 Cloth item subcategory page info and items hyperlink....................22
Figure 14 Cloth item’s attribute retrieve, few are marked with blue line........22
Figure 15 Flow chart of Zalando Scraper .......................................................23
Figure 16 Visual work flow of detailed-annotation web-solution ..................26
Figure 17 A sample document that stores the Image Annotation Details in MongoDB..................................................................................29
Figure 18 Embedded data structure of documents used in annotation web-solution collections........................................................................30
Figure 19 Sequence Diagram of Annotation Web-solution............................32
Figure 20 Apache server configuration file for Annotation web-solution .....32
Figure 21 SSL server configuration file for annotation web-solution ..........34
Figure 22 (a) JSON structure for a Instagram post and (b) JSON structure for a Zalando cloth item.................................................................35
Figure 23 A view comprised of - Amazon MTurk request task creation(left) and description of the task addition view (right). .............................37
Figure 24 The View of the MTurk task assignment and reward Options ....38
Figure 25 A snapshot of MTurk worker qualification choosing options ....38
Figure 26 Highlights the annotation website URL load as IFrame
Figure 27 The view to publish MTurk HIT along with excel sheet
Figure 28 The webpage view where requester checks the result
Figure 29 A sample detail-level annotated Image by Amazon Crowd Sourcing worker
Figure 30 A sample image assigned to different annotators
Figure 31 Image shown in Figure30 annotated by Annotator 1
Figure 32 Annotated data by Annotator 2 for image in figure 30
Figure 33 Graphical representation of item categories aggregation of Instagram Data
Figure 34 Graphical representation of item categories aggregation for Zalando
Figure 35 Graph illustrates total number of items in each pattern
Figure 36 Graph plots total number of items in each pattern
Figure 37 Graphical view of the Item category compare of two dataset for a specific time duration
Figure 38 Graph plot for 'JumperAndCardigan' item subcategory for each dataset
Figure 39 Graph representing the data distribution of different 'Shoe' subcategories
1 Introduction

The thesis work is intended to prepare a hierarchical fashion-based dataset from Instagram\(^1\) and a popular online shop Zalando\(^2\) for a cross-domain fashion recommendation system \([1]\). This chapter describes the problem that encourages the work, its context, the goal of the thesis and the outline of the report.

1.1 Background

Data collection and preprocessing is a primary task to provide efficient and sufficient data for machine and deep learning models. In order to implement a deep learning model for fashion recommendation we need to collect a handful amount of fashionista’s data from social media platform like –Instagram. To understand efficiently the fashionista’s style, we also need to cross check cloth-related information with online shopping platform. The different attributes attached with each clothing category encourages the design of a customizable crawling script to be used to gather data from online-shopping website- Zalando. Besides these data collection purposes there is a necessity to implement a web application that will be used by a special annotation team to perform a detailed-level of annotation for fashion clothes’ images and prepare a fashion dataset that is rich in annotations.

1.2 Problem

The available fashion datasets are insufficient due to the lack of detail-level annotations. The deep learning model proposed in Deep Cross-Domain Fashion Recommendation \([1]\) requires a fashion dataset with enough level of style annotations that are needed to implement more accurate recommendation system. This requirement raises a question how to prepare a dataset with detail-level cloth information and style from a social media platform. Also a fine-grained dataset containing detailed information of clothing items from a popular online shop.

1.3 Purpose

The purpose of this project is to prepare two datasets with more accurate and detailed style annotations from two different platforms for more accurate recommendation system. To leverage the detailed-level annotation, an Image Annotation Web solution will be implemented to ease the annotating task by using visual and textual information for each Instagram posts as there could be a lack of information in the Instagram images. Moreover, the web solution

\[^1\] [https://www.instagram.com/developer/]
\[^2\] [https://www.zalando.co.uk/women-home/]
provides support to integrate pixel-by-pixel segmentation and localization info. The dataset will be used by Deep Cross-Domain Recommendation system as well as by other researchers who intend to work with all the requirements mentioned above.

1.4 Goal
Two crawler will be implemented in a customized way to gather sufficiently large dataset from Instagram and an online clothing shop (Zalando) and analyze the distribution of the gathered data. An Image annotation web solution will be designed and implemented for detail-level annotation by the mean of visual and textual information collected from each Instagram image. Finally, these tools will help to prepare a highly-annotated dataset, efficiently pixel-by-pixel segmented, rich with localization info and represents a fashion data on social network.

1.4.1 Benefits, Ethics and Sustainability
The thesis is intended to prepare a detail-level annotation dataset from a social network platform. This dataset will help the researcher community to train a deep learning model for cloth information retrieval, recommendation and many more applications in fashion industry.

The data gathering process from Instagram was initially started using its API. However, due to the change of the API3 access through other system the implemented API based crawler was prone to gather less data. Therefore, an open source library Instagram Scraper4 is used to to crawl public fashionista’s data. The customized Zalando crawler was also implemented using the web scraping library due to the unavailability of online-shop API access. The project only crawl public user’s data from Instagram to avoid the privacy intrusion. The Zalando crawler also follow the websites term and conditions during data crawling.

1.5 Methodology
There are two basic categories of research methods [2]. One is qualitative and another is quantitative. The qualitative research methodology is exploratory research [3], whereas the quantitative research methodology is used to quantify an underlying problem by collecting numerical data, analyzing the structural data in a scientific manner and revealing formulae or hidden pattern from data. By considering the underlying thesis problem, the quantitative research method is selected. The quantitative research method in this work consists of collecting data from different sources in a structured measurable way, analyzing those data to provide useful insight for image analysis or text analysis and at last using those data to create a detail-level

3 https://www.instagram.com/developer/
4 https://github.com/rarcega/instagram-scraper
image annotation web solution. The resulted data finally will be used by a deep learning based recommendation system for accurate recommendations.

1.6 Delimitations
The initial task of the project was collecting data from the social-photo-sharing-media Instagram and the popular online shop Zalando. Due to the change in Instagram login permission scope for Instagram API and the blockade of shop public API access of Zalando, the project followed different technique to collect data which also affects data collection duration. However, in order to get permission for Instagram API access this thesis work went through certain difficulties. Among them- two consecutive permission review rejection and API policy change are significantly mentionable. Moreover, connection drop while crawling Instagram data was another constraint to mention.

1.7 Outline
The project report is organized as follows. Chapter 2 explores the detail theoretical background study related to available fashion datasets, their limitation and the necessity to prepare new detail-level annotated dataset and develop annotation tools. Chapter 3 describes the tools developed for data collection. In chapter 4, the design and implementation of detail-level annotation tool is presented. Chapter 5 summarizes the evaluation of developed tools and the statistical analysis of datasets. Chapter 6 constitutes the conclusion and future work.

5 https://www.instagram.com/developer/authorization/
6 https://github.com/zalando/shop-api-documentation
2 Theoretical Background and Related Work

This chapter describes the existing fashion datasets, their limitations to fulfill the requirement of recommendation system in [1] and available tools and techniques for image detail-level annotation and available data collection strategy from Instagram and Zalando.

2.1 Fashion Data Sources

Different researchers use different data sources according to their research goal such as - clothing item retrieval, clothes parsing, clothing style and feature extractions, style recommendation etc. Those data source can be categorized into three kinds of sources for gathering fashion images and their metadata. They are - Social Fashion Networks, Fashion Shows or Photography Images from web and Online shopping websites. An example of social fashion network includes – chictopia.com. The chictopia.com is mentioned as globally online attraction for style admirers In [5]. Therefore author in [4] selected chictopia.com as data source. Another research [6] uses two online fashion network pose.com and chictopia.com.

In [7] author used two sources for data- one is an online retail store ModCloth7 for fashion data and the other one is real-world photograph of people wearing clothes in different settings. Another online shop Tabofocus8 along with a New York Fashion Show images from 2014 was used as data source for [8]. [9] gathers data from three online shopping websites amazon.com, zapos.com and shopbop.com.

The aforementioned sources of fashion data provide images and their metadata such as – clothing item or attributes description in a coarse or fine-grained way based on the type of data source. The shopping website provides sufficient and accurate metadata while social network provides less data. However, to facilitate an accurate recommendation system there is a need for providing images, their detailed-level-hierarchical metadata and all possible user interactions (comments, likes, etc.) from a popular social network such as – Instagram for a successful recommendation.

2.2 Available Fashion Datasets

Using various fashion data sources researchers in different communities prepare datasets according to their research goals. In [4], researchers used the online fashion-focused community (Chictopia) and created a dataset that is publicly available9 where visual contents focused on “outfit of the day” along with a title, description and several labels from each Chictopia posts. It

7 https://www.modcloth.com/
8 http://taobaofocus.com/
9 http://vision.is.tohoku.ac.jp/~kyamagu/research/chic-or-social/
contains 617k posts and their tags. Another research [6] crawled 15k images and their corresponding tags from Chichtopia and Pose social networks which comprise 31 classes with the goal of clothes parsing. The StreetToShop in [7] prepared a dataset to match a real-world cloth picture to an online-shop item that contains more than 20k street photos and more than 404k online photos. However, it focuses only on 11 clothing categories- bags, belts, dresses, eyewear, footwear, hats, leggings, outwear, pants, skirts and tops. Besides this there is inadequacy of clothing subcategories and absence of user interaction in an way.

In [8], two datasets are used – one gathered from New York fashion show with 8k photos and manually labelled. The other dataset crawled from largest online shop Taobao 10 and contains 0.5 million photos along with user-behaviour history. A large dataset is prepared by crawling the web for fashion models photographs in [10] with 136k photos and each annotated only with fashionistas, brand and demographics.

MVC [9] introduced a dataset that contains a large amount of images and attributes of a higher resolution. The dataset tried to gather all four different views such as – front, back, left and right of each images and collected 37k clothing items with 161k images. Moreover, all the images collected from three online shopping websites such as – Amazon11, Zapos12 and Shopbop13 in the aforementioned research. Another work [11] built a large-scale clothes dataset of over 8000k images called DeepFashion14 available online which gathered from Forever21 15 and Mogujie 16 online shops. DeepFashion is comprehensively annotated with 50 categories, 1k descriptive attributes, 4 to 8 landmarks and cross-pose/cross-domain pair correspondences. According to the researcher in [11] it is one of the large-scale fashion dataset devoted for research community. In [12] a dataset namely Coloured-fashion dataset is constructed from popular fashion network Chichtopia. There were 2k images in the dataset that annotated manually with colour-category tags from Chichtopia posts. The annotation process kept only 13 colors and 23 categories. This dataset has a lacking of work with multicoloured images. A classification-focused dataset is used for [13] and the data was crawled from the internet. This work prepared a benchmark dataset for clothing classification task that contains 80k images with 15 clothing classes. A large collection of dataset created by [14] namely what-to-wear which contains 10 occasion categories and 7 multi-value clothing attributes from an online shop and Flickr.

10 https://world.taobao.com
11 https://www.amazon.com
12 https://www.zappos.com
13 https://www.shopbop.com
15 https://www.forever21.com/eu/shop
16 http://www.mogujie.com
2.3 Comparison of Existing Fashion Datasets

Table 1 compares the created Instagram dataset called FashionRec with the existing fashion datasets. Each dataset varies in the number of images or sizes as well as other features. Most of the datasets annotation vocabulary focused on a smaller set of coarse-grained attributes. As per [11] larger amount of fine-grained attributes helps to perform better prediction in clothing feature space which lead to the detection and retrieval of cross-domain clothing images. Following the same goal in [1] the prepared dataset also facilitates a hierarchical-category list with 169 category spaces. This dataset also offers a detailed set of attributes list that is achieved through a cross-domain knowledge transfer from a fine-grained clothing attributes detailed data source, in our case Zalando.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#images</td>
<td>158k</td>
<td>32k</td>
<td>800k</td>
<td>70k</td>
<td>Handful amount of data for deep learning training model</td>
</tr>
<tr>
<td>#categories</td>
<td>53</td>
<td>262</td>
<td>50</td>
<td>Category: 13 SubCategory: 155</td>
<td>The only dataset that provides hierarchical level of annotations</td>
</tr>
<tr>
<td>#attributes</td>
<td>N/A</td>
<td>N/A</td>
<td>1000 (not hierarchical)</td>
<td>176 (hierarchical)</td>
<td>Provides attributes per category cloth item, which helps more accuracy and detail-level annotations</td>
</tr>
<tr>
<td>#localization</td>
<td>Bbox</td>
<td>N/A</td>
<td>4-8 landmarks</td>
<td>Bbox</td>
<td>Fashion cloth items localization</td>
</tr>
<tr>
<td>#segmentation</td>
<td>Super-pixel</td>
<td>N/A</td>
<td>N/A</td>
<td>Pixel-wise</td>
<td>There is no dataset that provide both detail-level annotation and pixel by pixel segmentation</td>
</tr>
<tr>
<td>#exact-pair</td>
<td></td>
<td>300k</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
<tr>
<td>#user-interaction</td>
<td>limited</td>
<td>N/A</td>
<td>N/A</td>
<td>In the form of comments, hashtags, captions</td>
<td>This dataset introduces both image and user-interactions and corresponding annotations helps in better analysis of user-interactions and its relation to</td>
</tr>
</tbody>
</table>
The existence of the users-interaction data provides remarkable insight of user’s behaviour and communication style mentioned in [1]. As a result, supports accurate recommendation for followers or fans. However, this user-interaction data is absent in all publicly available dataset which directed this work to prepare a scalable fine-grained attributes level annotated dataset focusing on social network by incorporating localisation and pixel-by-pixel segmentation information.

2.4 Annotation Approaches

The information associated with images can be categorized into different types as mentioned in [16]. They are:

**Content independent-metadata**: This type of metadata does not refer to the content of the images but rather describes the author’s name, date, location etc.

**Content dependent-metadata**: It includes the data that directly refers to the image content and provides features for low or intermediate levels. For example – colour, shape, texture etc.

**Content descriptive-metadata**: This metadata describes or relates an object that is also representation of real world object. Also refers as semantics of the image content.

The Content dependent or descriptive annotation improves the image accessibility. After selecting the information metadata, the type of annotation has to be chosen. Annotation can be one of three types – a) free text, b) keywords and c) ontology based. Free text based annotation does not provide any pre-defined structure. Whereas Keywords oriented annotation must have required predetermined restricted keywords. Ontology based annotation adds a hierarchical structure to a collection of keywords and defines relationship and rules between them.

The creation of ground truth dataset by annotating a set of images requires following different strategies according to application context. In a survey paper [17] the author summarizes the steps for creating an annotated dataset as follows:

<table>
<thead>
<tr>
<th>#source</th>
<th>Online shop and Flickr</th>
<th>Street and online shop</th>
<th>Instagram</th>
<th>images</th>
</tr>
</thead>
<tbody>
<tr>
<td>#availability</td>
<td>public</td>
<td>public</td>
<td>public</td>
<td>public</td>
</tr>
</tbody>
</table>

Table 1. A Comparison of FashionRec with different existing fashion datasets.
(1) Selection of annotation method—direct manual, intensive group annotation, collaborative annotation over the www and www search for illustrating keywords.
(2) Choice of the type of annotation—free text, restricted keyword or ontology based.
(3) Create the restricted vocabulary according to the type of annotation chosen.
(4) Decision about the whole image or the specific region annotation which is the scope of image localization.

A comparison and contrast of image annotation methods as described in [17] is summarized as following:

<table>
<thead>
<tr>
<th>Method</th>
<th>Associated Decision</th>
<th>Advantages</th>
<th>Disadvantages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Manual Annotation</td>
<td>N/A</td>
<td>Provide efficient annotation</td>
<td>Time-consuming and laborious.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Annotation differs among annotators.</td>
</tr>
<tr>
<td>Intensive Group Annotation</td>
<td>Event organize</td>
<td>Annotation inconsistency is less than the distributed annotation.</td>
<td>Event organization.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Annotation differs among groups.</td>
</tr>
<tr>
<td>Collaborative Annotation over the www</td>
<td>System Design</td>
<td>Large group of annotators.</td>
<td>Motivating the annotators.</td>
</tr>
<tr>
<td></td>
<td>Motivation for annotators.</td>
<td></td>
<td>Inconsistency between annotators.</td>
</tr>
<tr>
<td>www search for illustrating keywords</td>
<td>Design of a methodology and system</td>
<td>Large pool of illustrative images</td>
<td>Policy can be differed for images.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Inconsistent images from search result.</td>
</tr>
</tbody>
</table>

Table 2. Summary of Image Annotation Methods towards the creation of a ground truth dataset.

2.5 Requirement of Manual Web Annotation Solution

A set of images properly annotated with text describing each image's content is required to create a ground truth for the evaluation of deep learning algorithm for object recognition or various applications. For a successful evaluation of deep cross-domain recommendation algorithm [1] a successful ground truth is necessary. According to [17] the best method for creating such dataset is to create an application context specialized vocabulary and then
manually annotate the images using that vocabulary. To achieve this goal, the selection of the annotation method in this thesis work is decided in a combination of manual but collaborative www based. The choice of the method leverages the advantage of both methods and discard the time-consuming and laborious drawbacks of manual annotation method. However, this chosen method does not eliminate the inconsistency between the annotation from different annotators and the possibility of inaccurate annotations. Although, proper documentation regarding fashion will be provided in the web-solution to mitigate the inaccurate annotations. Nevertheless, the opportunity to reject the inaccurate annotation data could be possible while the task is done using Amazon Mechanical Turk\(^\text{17}\) service. Besides this the target annotation tools must provide additional user interaction information in the form of textual representation along with visual data which drastically improves the annotation accuracy by providing valuable knowledge. The aforementioned goals add a task in this thesis work to develop a complete web-based annotation that integrate all controlled vocabulary, incorporate user interactions information and provide a flexible annotated data representation. In addition to the annotation functionality, object localisation and segmentation can enrich the ground-truth dataset, which is beyond of the thesis scope.

\(^\text{17}\) https://www.mturk.com
3 Data Collection, Preprocessing, Storage and Developed Tools

This chapter describes the step-by-step methods followed in the thesis work elaborately. The first section 3.1 explains the data collection process. The next section 3.2 explains the data structure and the storage system used in this work. Finally, section 3.3 describes each of the developed tools from data collection to data storage.

3.1 Data Collection Methods for Instagram and Zalando

In order to collect data from Instagram or Zalando, there exists two options. One is endpoint API access and the other one is webpage scraping. In detail the data collection strategy, process and steps are described in this section for each data source.

3.1.1 Instagram

Instagram social network has a well-defined and structured endpoint API\textsuperscript{18} access for data request. For the hierarchical dataset preparation task Instagram post is analysed and information such as – image, image captions, hashtag, like, comments, all commented users-id and so on- are gathered. The correspondence between API endpoint and the requested data is shown in Figure 1. So, the work was decided to consume the endpoints for data request initially.

![Instagram Endpoints and their requested data.](image)

OAuth 2.0 Protocol\textsuperscript{19} is used by the Instagram API for authentication and authorization. The Instagram API requires authentication if the request is made on behalf of the user. Each authentication request requires an ‘access\_token’. The request is also provided with the scope of the Instagram API access. By default, the basic information of a user is accessible. Other data, like- public content, like, comments, tags and so on- need permission. To get the permission for other data - every application needs to be registered in sandbox mode, fully developed with privacy policy and sent for permission

\textsuperscript{18} https://www.instagram.com/developer/endpoints/
\textsuperscript{19} https://tools.ietf.org/html/draft-ietf-oauth-v2-12
review request by fulfilling certain criteria. After getting acceptance from reviewer the sandbox application will go live and can manipulate the permitted data. However, a rejected permission review request will not be able to request those data and need to submit new request.

As a consumer of the endpoint API we followed every rule to grant access for required data from Instagram that is described below.

1) Create a sandbox client namely ‘Fashion Dataset Retriever’ by clicking ‘Manage Client’ option using Instagram developer account. This step generated a client id and then used this client to request access token. Figure 2 shows this client and the corresponding status. However, the image in Figure 2 was taken after getting permission review request and thus showed ‘Reviewed’ status.

![Manage Clients](image)

Figure 2. Fashion Dataset Retriever client view in Instagram Developer page.

2) Developed a sandbox application that consists of a brief description of the application, data privacy policy, the requested data usage, Instagram login experience and so on. A snapshot of the solution is shown in Figure 3.

![Figure 3](image)

Figure 3. A snapshot of the developed web-solution 'Fashion Dataset Retriever' to consume Instagram API.
(3) After completion of fully functional web-solution and successfully crawled from sandbox user, we followed the step ‘Submit Review Request’ to acquire permission for any public Instagram user data gathering. Accomplishment of five criteria before submitting a review request is mandatory. They are – a) submission quality, b) video screencast quality, c) app development phase, d) brand and policy compliance and e) use-case and permission compliance. ‘Submission quality’ refers to clear and precise description and notes included in the app and review request form. In ‘video screencast quality’ a detailed video is uploaded including the Instagram login experience and permission scope. As fulfilment of third criteria, our web-app is fully functional and deployed in the publicly accessible IP. Then a privacy policy complied to Instagram policy is prepared and finally a specification of the use case of the intended dataset is mentioned. A described representation of the aforementioned criteria is provided in Figure 4 which is taken from Instagram.

<table>
<thead>
<tr>
<th>Criteria for Review</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submission quality: Notes must be clear, concise and in English. We will not approve submissions with insufficient notes.</td>
</tr>
<tr>
<td>Video screencast quality: The video screencast must show the Instagram login experience of your app, proper credentials and the usage of every permission you are requesting. We will not approve submissions if you do not provide a clear and working screencast.</td>
</tr>
<tr>
<td>App development phase: We only review final and production version apps. Submissions with test and development version apps will not be approved.</td>
</tr>
<tr>
<td>Brand and policy compliance: Your app must comply with Instagram Platform Policy and Instagram Brand Guidelines.</td>
</tr>
<tr>
<td>Use case and permissions compliance:</td>
</tr>
<tr>
<td>1. To help individuals share their own content with 3rd party apps</td>
</tr>
<tr>
<td>2. To help brands and advertisers understand and manage their audience and digital media rights</td>
</tr>
<tr>
<td>3. To help broadcasters and publishers discover content, get digital rights to media, and share media with proper attribution</td>
</tr>
</tbody>
</table>

Figure 4. A set of requirement before any Instagram Permission Review request, taken from Instagram.

Although our solution did not request anything on behalf of the user rather crawl basic information, media and each media data such as- tag, caption, comments, like and so on- but as a requester of API endpoint we thoroughly followed every steps including a development to deployment of web-app and successfully granted as basic information requester. The ‘Fashion Dataset Retriever’ receive permission for basic information of Instagram user in the case the user must give permission to this application. This success comprised two consecutive permission review rejection. Nevertheless, this permission gave access to basic data but due to the change of Instagram API policy no

https://www.youtube.com/watch?v=XlaGwAx2tnU
further permission for other data is given as in [18]. This limitation led us to develop another tool that crawls each Instagram post’s data such as – media, tag, caption, comments, likes and so on. In section 3.3, the developed Instagram data crawler app will be described elaborately.

3.1.2 Zalando
The popular online-shop ‘Zalando’ had a RESTful (Representational State Transfer) API endpoints and exchanges information in a JSON (JavaScript Object Notion) structure. The API was public and well documented in [19]. Due to change in Zalando’s principle [20], it is treating its API as product and transforming from online shop to expensive fashion platform comprising with a set of products. Therefore, the API endpoint mentioned in [19] is no longer accessible. This transformation directed our data gathering task to implement a customized scraping tool that will crawl different category and subcategory cloths and their all possible attributes and represent the data in a structured technique. In order to scrape data from Zalando website we first analyse the web content structure. By looking at the web content structure, we noted down about the common pattern of organizing the products under different categories. Each category contains a certain number of subcategory products. Under the subcategory products all the products are oriented in paginated structure. This subcategory product web page also contains the total number of products of that subcategory. From this total number and paginated item per page we traversed through each of the products webpage and gathered all the attributes of that product and stored in JSON structure. The implemented tools along with data crawling techniques and data representation will be described in section 3.3 in detail.

3.2 Data Storage, Representation and Management
This section describes the data storage tool, the data structure and the management of data used in this work.

3.2.1 Data Storage
All the data gathered from Instagram or Zalando is structured in JSON format and stored in MongoDB[21]. MongoDB is a popular document based and NO-SQL database. It stores data in a flexible schema-less JSON structure where fields can be varying between documents and the data structure can be changed. A MongoDB database constitutes of different collections and each collection comprises a group of documents. A document is represented in a key-value pairs. A sample document is given in Figure 5 taken from [21].

21 https://www.mongodb.com
Each of Instagram User’s post data is stored as a document in a ‘instagramscraper’ database. Fashionista falls in the same group stored in same collection for example- liketoknowit user’s post reside in ‘liketoknowit’ collection. Similar structure is also followed in Zalando dataset. A single cloth item is represented as a document in ‘zalandoproductdetails’ database. However, all clothing category resides in the same ‘zalandoproductdetails’ collection. Table 3 provides a brief idea of the databases and collections.

<table>
<thead>
<tr>
<th>Data Source</th>
<th>database</th>
<th>collections</th>
<th>name</th>
<th>usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instagram</td>
<td>instagramscraper</td>
<td>name</td>
<td>usernameid</td>
<td>All users username and their owner_id</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>liketoknowituser</td>
<td>Each posts crawled data as document for liketoknowit users</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>swedishuser</td>
<td>Each posts crawled data as document for swedish users</td>
</tr>
<tr>
<td>Zalando</td>
<td>zalandodataset</td>
<td>zalandoproductdetails</td>
<td>zalandoproductdetails</td>
<td>All clothing products details</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>zalandodatabycollar</td>
<td>Clothing products sorted by collar attribute</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>zalandodatabymaterial</td>
<td>Clothing products sorted by</td>
</tr>
</tbody>
</table>
### 3.2.2 Data Structure

A property of MongoDB – flexible schema- allows to represent a document in two ways. They are – References and Embedded Structure.

**References:** In this data structure the references or links of other document are stored to depict the relationships. To access the related data application need to resolve the stored references. This is also called normalized model. From [22] an example of reference model is taken and shown in Figure 6.

![Reference data structure](image1)

**Embedded:** In this document structure relationships between data are stored in a single document. Embedded data is kept data as field or array within a document. Document manipulation during read, write, update or delete is performed by single database operation. This model generalizes a denormalized data model. Figure 7 shows an example of this model taken form [22].

![Embedded data structure](image2)
The embedded data structure is used for both ‘instagramscraper’ and ‘zalandodataset’. Each document can be represented as an entity or an object as [22]. The entity representation of ‘liketoknowit’ and ‘usenameidtable’ collections for instagramscraper database is shown in Figure 8. The ‘usenameidtable’ collection only contains Instagram username and the related owner id. Another document ‘liketoknowit’ comprises id of image, username, ownerid, url, like count, comment count etc. Each ‘liketoknowit’ entity contains a list of comments. Each comment is composed of owner’s detail (commenters username, profile picture and id), commented text, when the comment is added and an id.

Figure 8. Embedded data structure for 'liketoknowit' and 'userndtable' collection documents.
The data model for the collection ‘zalandoproductdetails’ of the zalandodataset database is shown in Figure 9. It contains product id, item category, item sub-category, name, brand, shop-url, color, media, material, different attributes, different measurement attributes and local file system path. Media, attributes and material fields are a list of items.

```
{  
    _id: "",
    Id: "",
    Item_category: "",
    Item_subcategory: "",
    Name: "",
    Brand: "",
    shopUrl: "",
    Category_tag: "",
    Product_group: "",
    Color: "",
    Media: "",
    Material: {
        outerfacricmaterial: ""
        Innerfabricmaterial: ""
    },
    Detailsattributes: {
        Neckline: "",
        Fastening: "",
        Pattern: ""
    },
    Measurefittingattributes: {
        Modelheight: "",
        Fit: "",
        Lenght: "",
        sleevelength: "",
        Toatallength: "",
        back: "",
        cut: ""
    },
    Downloaddate: "",
    Path: ""
}
```

Figure 9. Embeded data structure for zalando document resides in ‘zalandoproductdetails’ collection.

**Database Access Tool:**

To extract data from the database we have used *PyMongo* library. *PyMongo* [23] is developed by mongoDB and recommended to work with MongoDB from python. We used *PyMongo* tool to perform create, read, update and delete operations know as *CRUD*. In order to connect with a running MongoDB database instance at first MongoClient module is imported using
code snippet ‘From pymongo import MongoClient’. Then a connection to the database is made using the url and port address for example- ‘client = MongoClient(‘localhost’,27017)’. Once the connection instance is established then any database can be accessed as attribute-style or dictionary-style. A simple example of database access using the aforementioned instance client in attribute based style is ‘db = client.db_name’ and dictionary based style is ‘db = client.db_name’. The code instructions ‘db.insert()’, ‘db.update()’, ‘db.find()’ and ‘db.delete()’ are used to add, update, find and delete document correspondingly.

3.3 Developed Tools for Data Collection and Storage

In order to prepare the Instagram and Zalando dataset - a web solution is developed for gathering user’s data upon permission to consume Instagram API and a customizable script is developed to scrape data from Zalando.

3.3.1 Web Solution to Consume Instagram API

Among two popular python web-frameworks, Flask [24] is used rather than Django [25]. Flask is simple and flexible. It gives the developer more control over choosing appropriate tools for example- choosing database. Due to Flask’s flexible property we have chosen MongoDB as database that facilitates document-based storage. Whereas Django framework follows ‘batteries included’ [26] style which implies developer can start work with this immediately. It also included basic database administrative, routing, forms, and many more built in functionalities.

In Figure 3, the home page for the API request web-solution is shown. It comprises three links. The first link directs the web page to the privacy policy. A brief description and the API permission scope is mentioned in the second link. The third link directs the users to the Instagram login window. After successful login a prompt window is opened asking the user to grant access for the application ‘Fashion Dataset Retriever’. If the grant is permitted, then a script is executed and user’s basic data is gathered. Because of the policy change in Instagram API request data gathering script only crawl basic data. Thus, tags, comments and other data was left. To accomplish all data scraping goals, an open source library Instagram scraper [27] is used that scrape public user’s data using Instagram’s public URLs. The library is developed in python and uses request framework to scrape data from Instagram and store in JSON structure. To cope with our required data, the library is modified programmatically in the following ways:

1. Store data in MongoDB database according to different fashionista group – liketoknowit and swedish.
2. Add new Instagram posts or new comments in any existing JSON file.
3. Clean or remove any users downloaded images without accompanying JSON file.
4. Add the file system path URI for each image in the JSON structure.
The flow chart of the Instagram-scraper is depicted in Figure 10.

![Flow chart of Instagram-scraper tool](image)

**Figure 10. Flow-chat of Instagram-scraper tool.**

### 3.3.2 Customizable Zalando Scraper

A customized tool is developed in python in order to scrape online clothing details that include high resolution images and all attributes from Zalando. This script focused on women clothing items. The intended product categories and their corresponding sub-categories are listed below in 'category > sub-category' format:

- **Blouse & Tunic >** Blouses, Shirts, Tunics
- **Dresses >** Work Dresses, Denim Dresses, Casual Dresses, Shirt Dresses, Cocktail Dresses, Knitted Dress, Maxi Dresses, Jersey Dresses
- **Coats >** Down Coats, Winter Coats, Wool Coats, Short Coats, Trench Coats, Parkas
- **Jeans >** Denim Shorts, Slim Fits, Skinny Fit, Flares, Straight Leg, Loose Fit, Boot cut
• Jumpers and Cardigans > Athletic Jackets, Hoodies, Sweatshirts, Fleece Jumpers, Cardigans, Jumpers
• Skirts > Maxi Skirts, Pleated Skirts, A-Line Skirts, Pencil Skirts, Mini Skirts, Denim Skirts, Leather Skirts
• Tops and T-shirts > Long Sleeve Tops, Vest Tops, Polo Shirts, T-Shirts
• Trousers and Shorts > Joggers & Sweats, Shorts, Chinos, Playsuit & Jumpsuits, Leggings, Trousers
• Tight and Socks > Knee High Socks, Leggings, Socks, Thigh High Socks, Tights, Tight & Socks
• Shoes > Sport Shoes, Flip Flop & Beach Shoes, Ballet pumps, Flats & Lace-Ups, Boots, Trainers, Outdoor Shoes, Heels, Sandals, Ankle Boots, Slippers, Mules & Clogs
• Bags > Phone Cases, Wash Bags, Tote Bags, Sport & Travel Bags, Laptop Bags, Shoulder Bags, Rucksacks, Clutch Bags
• Accessories > Belts, Gloves, Hats & Caps, Jewellery & Watches, Purses, Scarves & Shawls, Sunglasses

**Library used in Zalando Scraper:**
Zalando scraper used two python libraries to scrape data from zalando. One is ‘requests’ and another one is ‘BeautifulSoup’. To retrieve the data from a website it is required to send HTTP request. Python has several libraries for HTTP requests like – httplib, urllib, requests etc. The reasons behind choosing the ‘requests’ library are simplicity and minimal coding. Using the ‘requests’ library zalando scraper sends HTTP request to server and stores the response from the server as response object in html structure.

Having the html data in hand the next step is to parse the data and navigate different elements in the objective of search specific elements. To help in this task we utilized python ‘Beautiful Soup’ library to pull data from html files. It also provides the facility to navigate, search and modify the parse tree with minimal code. Beautiful soup converts the response data in a nested tree structure and dissect the data to extract the required element.

The function ‘url_to_get_beauty_soup’ shows the HTTP request, its response using ‘requests’ library and conversion to Beautiful soup object of html data for navigation or searching in Figure 11. Other functions – ‘soup_to_get_link’ and ‘soup_to_get_script_tag’ are used to search html ‘a’ tag and ‘script’ tag respectively in figure 11.

![Figure 11](image.png)

**Figure 11** Sample code snippet for python HTTP request and HTTP response parsing and element searching using Beautiful Soup.
Work flow of Zalando Scraper:
The working steps of the developed tool are comprised of the following:

**Step 1:** Take item category and item sub-category as input.

**Step 2:** The developed tool will first send a request to the women clothing section using url 'https://www.zalando.co.uk/womens-clothing/' and convert the response object in a soup object. Then we search for the requested item category in the soup object. After retrieving the item category, the tool will send a request to extract all the item-subcategory and their corresponding hyperlinks. For example, in Figure 12, all sub-categories hyperlink list is retrieved for ‘Blouse & Tunics’ category from the highlighted html `<a>` tag’s ‘href’ attributes in class ‘cat_tag_20vv5’.

**Step 3:** If the input item sub-category exists then go to the subcategory hyperlink.

**Step 4:** By requesting sub-category hyperlink we extract the total items information, the pagination information and current page’s items hyperlinks in a list. Also the number of pages is kept in memory to traverse all pages. Then for each page we repeat from step 5. To visualize this step as shown in Figure 13, we can imagine one example such as the scraper will request ‘https://www.zalando.co.uk/womens-clothing-blouses/’ for ‘blouses’ sub-category. After transforming the request object to beautiful-soup object, then we will search for script tag with id ‘z-nvg-cognac-props’, which contains a CDATA with information like- total product count, products in current page, hyperlinks for all products in current page etc. The script tag’s CDATA is processed and loaded as JSON data to retrieve the total product count, products in current page and each products hyperlink in ‘url_key’. Then all links are stored in a python list structure.
Step 5: Having each item’s hyperlink from step 4, the scraper requests the item page and gathers all required information of the item and saves in well-structured JSON and in database. All images of the item also download in local file system. This step is repeated until all the item information is crawled. A portion of the products data is marked and shown in Figure 14.

The flow chart of the zalando scraper is depicted in Figure 15.
Challenges:
One of the challenging tasks was retrieving product information’s from script tag. This task was completed by pulling all text inside a specific script tag, then searching CDATA instance and at last converting it to JSON structure to get article information. Another challenge was faced while developing this tool was pulling all items of a category using pagination. The HTML data request form sub-category link is parsed by BeautifulSoup and it contains only 8 items hyperlink, although the page visually presents 84 items. The first version of the tool first pulls eight links and paginates by appending the parameter ‘?p=pagenumber’ to the requested url. This approach could not crawl all the items. However, this limitation led a manual but intensive html analysis to get all items hyperlink reside in one page. As a result, we found the pagination information and first page items url name inside a CDATA text that placed in a specific script tag. Using these script information, the tool was able to create a list of all items in a page. Having the pagination and product information both in the HTML and CDATA section was the main reason for page manipulation error.
4 Detail-level Image Annotation Web Solution

This chapter highlights the systems requirement, the architecture and the deployment of the developed detail-level fashion-based image annotation web-solution.

4.1 System Requirements

The system requirements of our image annotation web-solution are highlighted in table 4 in the form of user stories [30]. User stories describe the feature of the system from the perspective of the user of the system. The structure of the user story “As a < type of user >, I want < some goal > so that < some reason >” is used in table 4.

<table>
<thead>
<tr>
<th>As a/an &lt;type of user&gt;</th>
<th>I want to &lt;some goal&gt;</th>
<th>So that &lt;some reason&gt; Desired User</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requester</td>
<td>Create or delete different annotator user account</td>
<td>Can assign image annotation task to different annotator</td>
</tr>
<tr>
<td>Requester</td>
<td>Dynamically assign a set of images to each annotator</td>
<td>Have control over the image assignment task for different annotators</td>
</tr>
<tr>
<td>Requester</td>
<td>Use the web-solution any time of the day</td>
<td>Can access or review different annotators annotated data without any time bound</td>
</tr>
<tr>
<td>Requester</td>
<td>Provide guideline about the annotation task and categories of items and styles</td>
<td>Can share same knowledge with different annotators</td>
</tr>
<tr>
<td>Requester</td>
<td>Access the web-solution securely</td>
<td>Can assign tasks to Amazon Turk Service</td>
</tr>
<tr>
<td>Annotator</td>
<td>Get the login credentials</td>
<td>Can login to web-solution</td>
</tr>
<tr>
<td>Annotator</td>
<td>Browse all fashionista’s profile gallery</td>
<td>Can select any fashionista from the all fashionista gallery view</td>
</tr>
<tr>
<td>Annotator</td>
<td>Browse an individual fashionistas image gallery</td>
<td>Have the option to select any images for detail-annotation task</td>
</tr>
<tr>
<td>Annotator</td>
<td>Annotate images with different clothing items</td>
<td>Provide detail-level annotated images</td>
</tr>
<tr>
<td>Annotator</td>
<td>Have a annotation web-page assisted with visual data and other data (textually/ graphically)</td>
<td>Can provide more accurate annotation for each images</td>
</tr>
<tr>
<td>Annotator</td>
<td>Edit previously annotated images</td>
<td>Correct any erroneous annotated data in any image</td>
</tr>
</tbody>
</table>

Table 4 Brief description of user stories of Detail-level Image annotation tool.

Table 4 represents both functional and non-functional requirements of the detail-level image annotation web-solution.
4.2 Brief Description of System Design

A detail-level annotation web-solution is a fashion-based image annotation tool to add label of clothing items corresponding attributes which is accessible via www at domain22.

(1) The home page of this solution provides login options for different annotators.

(2) After successful login the user is directed to the all fashionista’s profile gallery visible in a grid view. However, with an unsuccessful login the user gets back to the login page again. Each thumbnail picture in the grid appears with a red-circled-tick icon, which provides information about the completion of annotation for a fashionista.

(3) By selecting any incomplete fashionista grid, the annotator will be directed to fashionista’s all image page view. Like profile grid view, each image in the all images view contains a red-circled-tick icon to give information about a complete or incomplete image annotation.

(4) Then annotator will select any incomplete image for detail-level annotation and will be directed to image annotation view. At the top of this page, an image of the fashionista, a table of text-analysis data from different user interactions and a pie chart are shown. It uses the Instagram public fashionista’s posted image and their meta-data crawled by a script-based library mentioned in section 3.3.3. Below this view, the page includes a question asking the fashionista’s style and a drop-down list for different clothing item categories and attributes with ‘save’ and ‘finalize annotation’ options. To assist annotators with information of fashion style guideline, clothing item and non-fashion item, sufficient documents and instructions are added. Moreover, it provides different selections such as- easy accessing to fashionistas images, options for traversing next/previous image, going to all fashionista’s profile page and logging out from the system. All the added annotation is saved in the database ‘annotatorwebapp’ of MongoDB.

(5) After adding an annotation of an image, the view is updated by all the saved annotated data for an image in tabular view. The annotator has the choice to edit/delete an annotated data by clicking the edit/delete button for a non-finalized image. However, a finalize image annotation view restricted this edit or delete choice. Upon finalizing the annotation, the annotator is directed to the completed-annotated-image page view.

Prior to the use of this system a requester should create an annotator user’s account and assign a set of images to that annotator. Image assignment task has the property to dynamically assign images to any annotator for any fashionista. A manual script is accompanied with the system for the dynamic assignment task. The working procedure of the dynamic image assignment is described in section 4.3.2. Figure 16 represents the UI based aforementioned

22 https://shatha2.it.kth.se
step-by-step work flow of the web-solution. Screencast of each web-page of web-solution is provided in Appendix A.

Figure 16 Visual work flow of detailed-annotation web-solution.

4.3 Framework Selection, Preprocessing and Implementation

This section represents the framework selection, pre-processing and the implementation of detail-level annotation web-solution.

4.3.1 Framework Selection

Image annotation web-solution is implemented in Flask micro-framework in python due to its simplicity and flexibility. For data storage we chose MongoDB due its facility for document-based storage. The python backend is responsible for insert, update, query and delete operations in the database MongoDB and holding all the application logic.

Flask comes with template language jinja23. A template is a file that includes variable and/or expressions which are replaced by the values when it is rendered. This replacement of values is passed as the parameter of the render_template() function along with the html file name. Web-solutions UI is designed by front-end framework Bootstrap24 version 3.3. Besides this, JQuery is used for client-side scripting and AJAX is used for sending data to server or update page by restricting reloading the page. At last, the Apache Server is selected for deployment. Table 5 represents the summary of different components technology.

---

23 [http://jinja.pocoo.org](http://jinja.pocoo.org)
24 [https://getbootstrap.com/docs/3.3/](https://getbootstrap.com/docs/3.3/)
<table>
<thead>
<tr>
<th>Functionality</th>
<th>Selected Technology (language)</th>
<th>Reason of Usage</th>
</tr>
</thead>
</table>
| Web-framework              | Flask (Python)                | • Simple  
• Flexible                                        |
| Template Engine            | Jinja2                        | • Compatible with python  
• Comes with Flask                                     |
| Database                   | MongoDB                       | • Document-based Storage                              |
| Front-end                  | Bootstrap                     | • Responsive                                         |
| Front-end Script           | JQuery                        | • Cross-browser  
• lightweight                                         |
| Back-end Server Communication | AJAX                          | • Sending data to server  
• Asynchronous page update                             |

Table 5 List of the selected technology and their reason of usage.

4.3.2 Pre-processing and Database Prepare

The detail-level image annotation web-solution requires some data to be processed beforehand. They are the following:

(1) *Fashionista data*: Our instagram-scaper script collected nearly 70000 posts corresponding to 47 fashionistas from ‘liketoknowit’ community and they are the primary datasource for annotation web-solution stored in JSON structure in MongoDB.

(2) *Annotator Task Assign*: Each user will be assigned a certain number of images of a fashionista. Each image of a fashionista will be assigned at-least five annotators for labelling. A manual *script* is prepared for this purpose that runs on the intended user’s images and assign images to five annotators and the pseudo code is shown in table 6. After that anyone can dynamically assign task to different annotators. Every assignment of images stores a template of annotated information.
corresponding to that annotator, which updates either when annotator saves annotation of finalize image annotation.

<table>
<thead>
<tr>
<th>Pseudocode of Dynamic Image Assignment Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>File:</strong> assignImageToAnnotator_dynamic_v3.py</td>
</tr>
</tbody>
</table>

Function DynamicImageAssignment(a, f, n):

**INPUT:** valid Annotator username as ‘a’, valid Fashionistas username as ‘f’ and Number of image to Assign as ‘n’

**OUTPUT:** Add ‘n’ number of entry in collection ‘annotateddatadetailwithuser’

**SET:** annotatoruser_list, common_count, annotateddatadetailwithuser collection, annotatorimageidlist collection

**BEGIN**

01. Liketoknowit ← create a JSON file containing all the fashionista’s images list ‘liketoknowituser’ as {fashinista:[imageidlist]} format
02. Image_id_list ← From the above JSON file Read the fashionista ‘f’s imageidlist
03. If length of Image_id_list > n then
04. Assign_id_List ← Slice the shuffled Image_id_list by ‘n’
05. For each id in Image_id_list do
06. annotator_list ← take random common_count number of annotators
07. For each annotator in annotator_list do
08. If id exists in annotatorimageidlist collection for a, then
09. pass
10. else
11. add id to annotatorimageidlist
12. endif
13. End for
14. End for
15. End if
16. For each non-annotated id of annotator ‘a’ in annotatorimageidlist Do:
17. Insert template for annotator ‘a’ in ‘annotateddatadetailwithuser’
18. End for

**END**

**Table 6 Pseudo code for Dynamic Image assignment Algorithm.**

The template that was inserted after running the assignment algorithm in the collection ‘annotateddatadetailwithuser’ is shown in Figure 17. The ‘annotateddatajson’ is updated once the annotator adds annotation and ‘annotated’ field is updated once the annotation is complete for that image.
Figure 17 A sample document that stores the Image Annotation Details in MongoDB.

(3) **Image Processing:** For the gallery view of the all user folders all the fashionistas profile picture are used. Fashionistas profile pictures are downloaded and converted to base64\(^{25}\) binary format and stored as JSON file in the server root path- to decrease the frequent database access. The same process is done for all fashionistas image but stored in the mongo database. These images binary is used to create the thumbnail for the fashionistas album as well as annotation form UI. The reason behind the conversion of image binary and storing in database is – Instagram image URL was inaccessible and there was a restriction to read the image from file system through the client-side.

(4) **Text-analysis data representation and liketoknowit link processing:** Text-analysis data for each image contains different fashion related words and each word consists of ten most frequent data. For the data table and pie chart visualisation we sort the text analysis data in ascending order, we kept the last four words, normalised them and stored in database. The liketkit link also retrieved from the text-analysis data and stored in database.

### 4.3.3 Data Structure

The detailed-annotation web-applications data is stored in ‘annotatorwebapp’ database. Its constitutes a set of collections. They are- ‘users’, ‘imagebasebinary’, ‘imagetextanalysisdata’, ‘annotatorimagelist’ and ‘annotateddetailwithuser’. The data model for each of the collections is shown in Figure 18.

A brief description of each of the collection is given below:

1. **‘users’**: It holds the created annotator usernames as well as their passwords.
2. **‘imagebasebinary’**: The purpose of this collection is to hold all images base64 binary encoded data. Due to the inaccessibility of images from front-end html file for security reason, each images base64-encoded data is stored in MongoDB and retrieved to either create thumbnail for gallery view or render image in front-end.

\(^{25}\) [https://en.wikipedia.org/wiki/Base64](https://en.wikipedia.org/wiki/Base64)
(3) ‘imagetextanalysisdata’: To incorporate the text analysis data into web-solution we process text-analysis. The original text_analysis data contains top 10 predictions and their probabilities. We sort the data according to the descending order and we keep the top 4 predictions. Moreover, we also normalize their probability and store in this collection.

![Diagram of annotated data structure](image)

Figure 18 Embedded data structure of documents used in annotation web-solution collections.
31

(4) ‘annotatorimagelist’: This collection is needed to keep track of the assigned and annotated image list of an annotator. The reason is to avoid duplicate assignment for a new batch of images to a certain annotator while running the dynamic image assignment script.

(5) ‘annotateddetailwithuser’: In order to store the detail-level annotation of images this collection is created. As each image will be assigned to five different annotators, so annotation data corresponding to an image keeps the annotator name, fashionistas username and image id fields. The ‘imageinfo’ field holds the information about an image annotation. Moreover, this collection is used to select the ‘filled check-box’ and ‘unfilled check-box’ during the rendering of ‘allUserFolder.html’ and ‘singleUserFolder.html’ web-pages for different annotators. For example, an image is assigned to different annotator. One annotator-a1 completed annotation and other annotator-a2 did not. Then the image render in a1 webpage view with ‘filled check-box’ and in a2’s webpage as ‘unfilled check-box’. To achieve this goal this collection structure is designed.

4.3.4 Implementation

The sequence diagram of annotation web-solution is depicted in Figure 19. The web-solution working steps are described in implementation’s perspective as following:

1. Annotation web-solution goes to the url route ‘@annotation_app.route('/', methods=['GET','POST'])’ and renders the ‘login.html’ template which prompts a username and password.

2. After successful login the annotator is directed to all fashionistas album page from route ‘@annotation_app.route('/allUserFolder/<annotator_username>', methods=['GET', 'POST'])’ and then render ‘allUserFolder.html’. In the album the ‘filled’ checkbox implies that the fashionistas annotation is completed and ‘unfilled’ checkbox implies uncompleted annotation for that fashionista. Although each annotator all-users-folder view reads data from the same profile-picture-json file, the red checkbox data is updated corresponding to the annotated information stored for each image assigned to that particular annotator (from the template stored in ‘annotateddatadetailwithuser’). This view retrieves data from the database image annotation template and provides updated information to the red checkbox.

3. The image album view of a single fashionista and all assigned images to an annotator is rendered by ‘singleUserFolder.html’ from the back-end route ‘@annotation_app.route('/singleUserFolder/<insta_username>',methods=['GET', 'POST'])’ and sending all images information. Here, the ‘filled’ or ‘unfilled’ checkbox is also rendered.

4. By selecting any unannotated image the user is directed to the ‘fashionistasImageAnnotation.html’ page, which is rendered through
route @annotation_app.route ('/fashionistasImageAnnotation', methods=['GET', 'POST']). This view contains text analysis information of hashtags, comments and caption in a table view. The text-analysis table shows the top four word in each fashion vocabulary. Pie chart is shown according to the value of the fashion vocabulary. The annotated data is saved and updated by the same app route. The annotators save hierarchical item category and their corresponding attributes. Each item has certain mandatory attributes to be annotated. By saving each annotated fashion item the view will be updated by a table view, where all the saved items will be listed. The annotator can delete any saved annotation. After finalizing the annotation, the annotator will be directed to another view 'completeImageAnnotated.html'.

5. Upon the selection of a completed annotated image the use is directed to " page which is rendered using route @annotation_app.route ('/completeImageAnnotated', methods=['GET', 'POST']). Completed-annotation view shows all the text analysis table, pie chart, image etc as 'fashionistasImageAnnotation.htm' page. The only difference is there is a table showing all the detail of clothing annotated data and the style preference for that image.

![Sequence Diagram of Annotation Web-solution.](image)
4.4 Deployment and Secure Access

In order to make the annotation web-solution available to the end users and let them transfer data securely using HTTPs we deployed the solution in Apache server and used Let’s Encrypt for secure service. This section briefly describes each of them.

4.4.1 Apache Server

The detail-level image annotation web-solution is made available to the end user (users within team or Amazon Turk service participant) by deploying the solution in a server. Due to having some existing experience with Apache server, we chose Apache as deployment server. Apache 2.4.6 server is used as deployment server and mod_wsgi server module which provides WSGI compliant interface for hosting our python based web-application.

Steps to Install and Configure Apache Server:

We followed the following steps as described in [33] to install server in CentOS machine and web-solution in sever.

1. Update and install package repository: The server is installed using command “yum install httpd -y”.
2. Disable SELinux : By default SELinux is enabled in CentOS 7. It was disabled by editing SELINUX = enforcing to SELINUX = Disabled in the config file in /etc/selinux/config. Then restart the machine.
3. Allow Apache through the Firewall: Then it is needed to allow the default Apache port 80 (HTTP) and 443 (HTTPS) using commands: “sudo firewall-cmd --permanent --add-port=80/tcp” and “sudo firewall-cmd --permanent --add-port=443/tcp”. After enabling ports and to make the effects stable, “sudo firewall-cmd –reload” command is executed.
4. Install web-solution to the sever by copying source code in server root - /var/www/annotation and restart server using “sudo systemctl start httpd”.
6. Then create a virtual host configuration file called annotation.webapp.conf in the path - /etc/http/conf.d/annotation_webapp.conf and restart server to read new configuration. Figure 20 shows the aforementioned virtual host configuration file and highlights mandatory settings.

---

26 https://modwsgi.readthedocs.io/en/develop/
27 http://wsgi.tutorial.codepoint.net
4.4.2 HTTPS secure access using Let’s Encrypt

Annotation web-application provides secure service with SSL and accessible through HTTPs. An open source automated service Let’s Encrypt\(^{28}\) is used for this purpose. It issues domain validated free SSL\(^{29}\) certificate.

Reason to choose Let’s Encrypt over OpenSSL:

1. It’s certificate is trusted in different browsers and operating systems by default in comparison to the other free certificates of self-signed certificate [31]. As our web solution will be used by Amazon Turk Service, so we chose this Domain Validated(DV) SSL certificates provider to make our solution secure.

2. It supports automated renewal on every 90 days.

3. However, it does not provide a certificate for a bare IP address.

Steps to Enable SSL in Apache Server:

We followed the following steps according to reference [32] to install, issue SSL certificate and configure Apache server SSL configuration for detail-level image annotation web-solution.

1. Create a directory for example - /letsencrypt in the file system and clone Let’s Encrypt using command “git clone https://github.com/letsencrypt/letsencrypt”.

2. Browse to the created directory /letsencrypt and run command “./letsencrypt-auto --help”. This command will install required files and prepare system for requesting SSL certificate.

---

\(^{28}\) https://letsencrypt.org

\(^{29}\) http://info.ssl.com/article.aspx?id=10241
(3) After that run command ". /letsencrypt-auto --apache -d yourwebsite.com" where “yourwebsite.com” is a valid domain name. In our case it is “shatha2.it.kth.se”. We got “prevenence issuance” because there was no record of public DNS record for domain “shatha2.it.kth.se”.

(4) By adding a public DNS record the Let’s Encrypts server issued SSL certificate for our domain which creates a SSL server configuration file as shown in figure 21 for our web-application. There was another hinder during the SSL server configuration file generation process that is WSGIProcessGroup duplicate problem. After removing the WSGIProcessGroup declaration in our web-application configuration file Let’s Encrypt server was able to create SSL configuration file for the web-solution.

```
# WSGIDaemonProcess webapp user=apache group=apache threads=5 home=/var/www/annotation_webapp/ WSGIScriptAlias / /var/www/annotation_webapp/views.wsgi

<VirtualHost *:443>
  ServerName annotation.webapp
  ServerAlias annotation.webapp
  <Directory /var/www/annotation_webapp/>
    WSGIProcessGroup webapp
    WSGIApplicationGroup %{GLOBAL}
    WSGIScriptAliasOn
    Order deny,allow
    Allow from all
  </Directory>
  ServerAlias shatha2.it.kth.se
  SSLCertificateFile /etc/letsencrypt/live/shatha2.it.kth.se/cert.pem
  SSLCertificateKeyFile /etc/letsencrypt/live/shatha2.it.kth.se/privkey.pem
  Include /etc/letsencrypt/options-ssl-apache.conf
  SSLCertificateChainFile /etc/letsencrypt/live/shatha2.it.kth.se/chain.pem
</VirtualHost>
```

Figure 21 SSL server configuration file for annotation web-solution.
5 Evaluation and Analysis

This section demonstrates the statistics of data collected from Instagram and Zalando, the evaluation of annotation web-solution by Amazon Crowdsourcing Worker as well as team members, usage of Instagram API application and analysis of data in terms of clothing categories and attributes for example -pattern will be presented.

5.1 Statistics of Datasets

The scrappers implemented to gather data from Instagram and Zalando were able to collect sufficient data from each source. Table 7 shows the collected datasets statistics. Instagram dataset consists of 70k images and each of the images caption, tag, like count, comments with commented user's basic information is aggregated as JSON structure. The customized Zalando crawler gathered almost 62k cloth items in different categories which accumulated each cloth attributes details. This data also represented as JSON Structure.

<table>
<thead>
<tr>
<th>Instagram Dataset</th>
<th>Zalando Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Total Fashionista: 47</td>
<td>• Total Clothing Items: <strong>62K</strong></td>
</tr>
<tr>
<td>• Total Post: <strong>70k</strong></td>
<td>• Clothing Categories: <strong>12</strong></td>
</tr>
<tr>
<td>• Total Comments: <strong>4M</strong></td>
<td>• Cloth Attributes: <strong>In-detail</strong></td>
</tr>
<tr>
<td>• Duration: October, 2017 to March, 2018.</td>
<td>• Duration: March - April, 2018.</td>
</tr>
<tr>
<td>• Liketoknowit Community</td>
<td>• Online shop</td>
</tr>
</tbody>
</table>

Table 7 Statistics of Instagram and Zalando dataset.

Figure 22 (a) JSON structure for a Instagram post and (b) JSON structure for a Zalando cloth item.
Figure 22(a) presents the JOSN data structure gathered using Instagram crawler. The square highlighted the caption, comments, tags and image url. In figure 22(b) a JSON structure of Zalando dataset is provided and highlighted the cloth category, brand and attributes details.

5.2 Evaluation of Detail-level Image Annotation Web-service

A set of Instagram images is assigned to Amazon Crowdsourcing worker to annotate images in-detail using the developed web-solution. This section describes the step-by-step procedure to assign tasks to crowdsourcing people as well as evaluation of the results. Moreover, the modification of the web-solution to consume amazon MTurk service will be described.

Amazon Crowdsourcing Task Assignment:

To be able to assign task, one need to create an account as Amazon MTurk requester. However, there is another prerequisite to pass the assignment task in sand-box mode between requester and worker. After fulfilling all these prerequisite, we followed the step-by-step procedure mentioned below:

(1) Go to Amazon MTurk equester website: https://requester.mturk.com/.
(2) Start by creating a custom project from ‘Create menu’ select ‘Other project template’ as shown in Figure 23 left view.
(3) Then add meaningful information about the project like title- in our case ‘Fashionista’s Image Annotation task’ and a brief description. This information shown in figure 23 right view.

(4) After that the reward per assignment was decided. As well as, the time per assignment, the expiration time per hit, and other details were
configured during the project creation. Each task assignment in our case a single image to be annotated- is known as HIT. However, the same HIT can be assigned to multiple workers for more accurate image annotation.

(5) Next the decision of the level of workers whether they are Masters (Highly qualified with a record of approved hits) or non-Masters can be configured. In this project, the choice was for non-Masters but with a manually created qualification type (FashionSpecialist). This qualification was added after publishing the hits for workers, and choosing the workers who proved quality of work and assigning them this manually created qualification type. Figures 24 and 25 show the added value for step 4 and 5 respectively.

![Figure 24 The View of the MTurk task assignment and reward Options.](image)

![Figure 25 A snapshot of MTurk worker qualification choosing options.](image)

(6) Then in the design layout, we added brief information to guide the workers during annotation. This step resembles a front-end html-page
design that includes the guidelines along with the annotation-web-solution. We loaded the annotation website URL inside a IFrame highlighted in figure 26.

![Figure 26 Highlights the annotation website URL load as IFrame.](image)

The website URLs loaded from an excel sheet which uploaded upon publishing the task as shown in Figure 27.

![Figure 27 The view to publish MTurk HIT along with excel sheet.](image)

(7) Upon publishing the task, we got the calculation for the HITs. After successfully publishing the Hits, we checked the amazon MTurk worker that the HITs are correctly published.

(8) We checked the results from amazon MTurk requester account. From “Manage” menu, the result were visible. The webpage view is shown and result option is highlighted in Figure 28. Each URL in the results opened the HIT with the saved annotation details. If the results are satisfactory the HIT is approved and the worker gets paid. Otherwise the result can be rejected by mentioning appropriate reason.
Image Annotation by Amazon Crowdsourcing worker and Team members:

Annotation by Amazon Crowdsourcing worker:
We assigned some image annotation task to amazon crowd source worker and received a good quality result with detail-annotation. In Figure 29 we show one sample result from the amazon MTurk worker and annotated with 5 different categories- a) accessories, b) bags, c) shoes, d) trousers and shorts and e) jackets and their mandatory attributes. We modified our web-solution to provide less user interface for the amazon MTurk worker such as – removal of next and previous image browse button, pie representation omission, as well as the absence of non-fashion item or important item options. The web view is added in Appendix A.

Figure 28 The webpage view where requester checks the result.

Figure 29 A sample detail-level annotated Image by Amazon Crowd Sourcing worker.
Annotation by Team members:
The annotation web-solution is also evaluated by team members. We assigned a number of images to different members for detail-level annotation. A sample image in Figure 30 is assigned to two members for detailed annotation. Figure 31 shows that one member annotated only 3 items - jeans, jumper and cardigans and accessories. Another member annotated with 4 items - trouser and shorts, jumper and cardigans, shoes and accessories shown in Figure 32. Annotation result showed that manual image annotation led inconsistent data-label for different annotators. However, by using a voting algorithm, this problem can be solved by choosing the best suited label for the image which is out of this project scope.

Figure 30 A sample image assigned to different annotators.

Figure 31 Image shown in Figure 30 annotated by Annotator 1.

30https://scontent-arn2-1.cdninstagram.com/t51.2885-15/e35/23734962_302529360260575_8080536807676051456_n.jpg
5.3 Instagram API Application

A web-based application is developed to consume Instagram API to crawl any fashionista’s data. The application is granted for gather basic data by Instagram API reviewer. Upon successful login and give permission to the developed-API application to gather basic information, a script is needed to execute to crawl data for fashionistas recent media using Instagram API endpoints ‘https://api.instagram.com/v1/users/self/media/recent/?access_token = ACCESS-TOKEN’. Here, ACCESS-TOKEN is given by Instagram upon successful user login and give permission. To evaluate the application, we chose some sample Instagram account and gathered data from them. This task successfully gathered data for the users. Here, only one user - ‘umu2017’ sample will be described. The following table shows the process to consume Instagram API:

<table>
<thead>
<tr>
<th>Step/Goal</th>
<th>Description</th>
<th>UI/ Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Crawl Basic data for Instagram user ‘umu2017’ using API endpoint.</td>
<td>The image in column 3 shows information for user ‘umu2017’. Post: 6 Follower: 4 Following: 2 Images: 6</td>
<td><img src="image_url" alt="Image" /></td>
</tr>
<tr>
<td>Step</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Browse to the Instagram API application webpage. Login to Instagram using username and password.</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Fashion Dataset Retriever Permission. After successful login give permission to the application to crawl data. Highlighted in red square.</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Run Data Gathering Script and gather recent media data. Run a script that do use of endpoint: '<a href="https://api.instagram.com/v1/users/self/media/recent/?access_token">https://api.instagram.com/v1/users/self/media/recent/?access_token</a> = ACCESS-TOKEN' to crawl each image and comments.</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Run Data Gathering Script to gather user basic info data. Run a script 'instaScraperAPI' that do use of endpoint: '<a href="https://api.instagram.com/v1/users/self/?access_token=ACCESS-TOKEN%E2%80%99">https://api.instagram.com/v1/users/self/?access_token=ACCESS-TOKEN’</a> to crawl users basic info.</td>
<td></td>
</tr>
</tbody>
</table>
6. Run Data-Gathering Script to gather comments for a media using media id

Run a script 'instaScraperAPI' that uses an endpoint: 'https://api.instagram.com/v1/media/{media-id}/comments?access_token=ACCESS-TOKEN' to crawl all comments using media id get from recent media endpoint.

Media-id = 1524246243138891962_467508547

Table 8 Steps to consume Instagram API by developed web-application.

From rows 4, 5 and 6 in table 8, it can be seen that our Instagram API application successfully gathers user’s basic information, recent media and each media’s comments by accessing API endpoints. However, due to the restriction of the login and permission grant for each user this API access has lack of gathering a large amount of data from several user accounts.

5.4 Analysis of Instagram Dataset and Zalando Dataset

One of the objectives of the thesis is to analyse the Instagram data and the Zalando data to get insight of data distribution and relationships. To achieve this goal, we aggregated data at the level of clothing item categories and attributes. Moreover, clothing category is aggregated within a fixed time duration to get insight about the fashion and style for the specific time being.

5.4.1 Analysis in Item-Category Level:

Instagram Dataset:
In the category-based evaluation case, we used MongoDBs ‘group by’ functionality at first and then aggregated them. The aggregated data is plotted in Figure 33. Along X-axis we plotted all the item categories and in Y-axis we plotted the total number of items in each category.

Figure 33 Graphical representation of item categories aggregation of Instagram Data.
It is visible from the graph that three categories - ‘JumperAndCardigan’, ‘Accessories’ and ‘TrousersAndShorts’ – have highest frequency in the Instagram user-interaction data. ‘Shoes’, ‘Dresses’ and ‘TopsAndTshirts’ acquired the next third position sequentially. From this trend we can conclude that these six categories are the most frequent fashion items used by fashionistas. Moreover, by taking account these fashion items we can enrich our Zalando dataset by gathering these item categories which will finally help more accurate recommendation system. However, the total frequency of these item categories could surplus our Instagram datasets because of each image annotated more than one fashion items.

**Zalando Dataset:**
From the data distribution of Zalando dataset in Figure 34, we can observe that ‘Shoes’ item category has highest frequency in the dataset and next is ‘Dresses’, then ‘TopsAndTshirts’. The data distribution can be compared categorically with the Instagram dataset and from which can point which category has less data in Zalando. Then the selected item category can be used to crawl data using the customized Zalando crawler. As a result, it could be possible to provide more fine-gain data from Zalando for accurate recommendation.

![Figure 34 Graphical representation of item categories aggregation for Zalando.](image)

**5.4.2 Analysis in Attribute Level:**

**Instagram Dataset:**
Our predefined item attribute vocabulary includes- material, brand, pattern, occasion, collar etc. For this analysis case, we selected pattern attribute. Figure 35 shows the graphical representation of pattern-base analysis. Pattern attribute consists of sixteen words. Among all pattern ‘checked’ pattern possesses the highest count. Patterns like – ‘herringbone’, ‘colourful’, ‘striped’, ‘print’, ‘floral’, ‘plain’ and ‘marl’ are next prominent after ‘checked’. The rest of the pattern also has moderate frequency. By looking at the graph, the clothing pattern trend is noticeable. The log scale value of the item count is used as X-axis series.
Figure 35 Graph illustrates total number of items in each pattern.

Zalando Dataset:
Attribute-based Zalando dataset analysis is shown in Figure 36, where X-axis represents the logarithmic value of the total item count. In contrast to the Instagram dataset ‘Plain’ is the most prominent fashion clothing pattern in Zalando. There is also distribution similarity in ‘floral’, ‘Marl’ and ‘Print’ patterns. However, the ‘Checked’ pattern have the highest frequency in Instagram dataset, but in Zalando it has mid level of frequency.

Figure 36 Graph plots total number of items in each pattern.
5.4.3 Comparison of Zalando and Instagram Dataset:

In order to obtain the insight of the Instagram and Zalando dataset we compare item category total for two datasets for a specific time duration from 11th November, 2017 to 20th May, 2018. Figure 37 shows the data distribution of both datasets for 13 item categories. However, both datasets item count is scaled using log. Although the Zalando data quantity seems to higher in all categories compare to Instagram, but categories like – ‘shoes’, ‘TopsAndTshirts’, ‘Dresses’, ‘JumperAndCardigans’, ‘TrousersAndSocks’, ‘Accessories’ and ‘Coats’ has noticeable frequency in Instagram dataset compare to other categories.

![Zalando and Instagram Item Category for a time duration.](image)

**Figure 37** Graphical view of the Item category compare of two dataset for a specific time duration.

Our predefined item subcategory vocabulary consists of 155 subcategories that falls under 13 item categories. Another analysis between Instagram and Zalando dataset is done by considering the highest frequency item-category in each dataset. From Figure 33 and 34 we get the highest item category for Instagaram and Zalando. They are – ‘JumperAndCardigans’ and ‘Shoes’ for Instagram and Zalando respectively. By plotting the data in Figure 38 in item subcategory level, the analysis points the specific data gap between datasets. For example, ‘Cardigan’ and ‘FleeceJumper’ have extremely low frequency data in Zalando.

![Zalando and Instagram Jumper And Cardigan Sub-category data comparison](image)

**Figure 38** Graph plot for ‘JumperAndCardigan’ item subcategory for each dataset.
Considering the highest frequency item category ‘Shoe’ in Zalando, the data is plotted for different ‘shoe’ categories in Figure 39. It is visible that the Instagram data does not possess lower frequency in all sub-categories. However, Instagram sub-categories like - ‘Boots’, ‘Sandal’ and ‘sleeper’ has significant frequency compared to Zalando. Moreover, ‘Ballets’, ‘Flat&Lace’ and ‘AnkleBoot’ sub-categories also have noticeable frequency. Analyzing the data distribution in sub-category level, gives more insight about the data gap.

![Zalando and Instagram 'Shoe' sub-category comparison](image)

Figure 39 Graph representing the data distribution of different 'Shoe' subcategories.

The data distribution analysis of Instagram and Zalando dataset provides the information about data gap among them. As a result, the usage of customized Zalando crawler will be advantageous to gather specific items in category, subcategory or attributes like- pattern, material or brand.
6 Conclusion and Future Work

This project presents a framework to alleviate tasks—data collection to prepare a detail-level-style annotated large clothing dataset from photo-sharing-based social network Instagram. To ease the data collection task, two crawlers for both Instagram and Zalando is implemented in a customized way to gather a handful amount of data. In order to facilitate the detail-level annotation task for a large number of images, efficiently and accurately on time, an annotation web-solution is developed. Moreover, the significance of the web-solution is to simplify the image annotation task with a predefined hierarchical cloth category and their corresponding attributes. Besides this coarse-grain annotated dataset another fine-grain, clothing dataset is assembled from Zalando online shop to mitigate the annotation level gap presents in former dataset.

The developed annotation web-solution is utilized and evaluated by the team members and Amazon Turk Service for detail-level annotation. Although the service is available at any time, but the large numbers of image loading hinders the annotation task slightly, which guide us to improve the solution in future. Currently, the annotation web-solution only provides a platform to annotate attributes per cloth-item category. Incorporating the localization and pixel-by-pixel segmentation will also enrich the dataset. Another improvement of the web-solution is to combine customizable categories or attributes. These datasets will assist a deep cross-domain fashion recommendation system for accurate recommendation by considering the clothing contents and the user-interactions. Nevertheless, it will also facilitate researcher’s community who intend to work with a detail-annotated fashion-based dataset of a social network.
7 References


8 Appendix A

8.1 Detail-level Annotation Web-solution Web Interface:

8.1.1 Login page:

8.1.2 All fashionista Album Page:
8.1.3 Individual fashionista Album:

Image taken from a public fashionista’s Instagram account called ‘apincholovely’.

8.1.4 Image Annotation User Interface:

Image taken from a public fashionista Instagram account called ‘cellajaneblog’.
8.1.5 Complete Image Annotation view:

Image taken from a public fashionista’s Instagram account called ‘apincholovely’.

8.1.6 Image Annotation User Interface for Crowd Sourcing Worker:

Image taken from a public fashionista Instagram account called ‘hauteofftherack’.