Neural Computing and Applications
https://doi.org/10.1007/s00521-024-09901-w

ORIGINAL ARTICLE

An AI pipeline for garment price projection using computer vision

Rodrigo Rico Gómez[1] · Joe Lorentz[1,2] · Thomas Hartmann[1] · Arda Goknil[3] · Inder Pal Singh[2] · Tayfun Gökmen Halaç[4] · Gülnaz Boruzanlı Ekinci[5]

Received: 16 June 2023 / Accepted: 23 April 2024

Abstract

The fashion industry's traditional price-setting methods, based on historical sales and Fashion Week trends, are inadequate in the digital era. Rapid changes in collections and consumer preferences necessitate advanced Artificial Intelligence (AI) techniques. These AI methods should analyze data from various sources, including social media and e-commerce, to predict future fashion trends and prices. In this paper, we propose, apply, and assess a data analytics approach, i.e., FashionXpert, employing several image processing and machine learning techniques in an AI pipeline for garment price prediction. It integrates various heterogeneous data sources (e.g., textual and image data from e-stores, brand websites, and social media) to obtain more consistent, accurate, and beneficial information. We evaluated its effectiveness with an industrial data set obtained by a fashion search tool from the electronic commerce sites of clothing brands. FashionXpert predicted garment prices with an average Mean Absolute Error (MAE) of 15.31 EUR on a data set that has a standard deviation of 72.99 EUR.

Keywords Image segmentation · Multi-label classification · Price regression

1 Introduction

The fashion industry is known to be enormously fast-moving and short-lived. There are persistent threats, even for well-established brands, such as little brand loyalty due to high market fragmentation, high cost of handling online returns, as high as 50%, and an ever-increasing number of fashion seasons. Some fashion businesses produce up to 52 micro-seasons[1] every year. Some manufacturers and retailers in the fashion industry already use automated solutions that collect and analyze data from social media and e-stores to generate statistical reports about trends and prices, e.g., follow.fashion [1] and Edited [2]. However, the statistical postmortem information is insufficient for fashion industry players to keep up in today's fast-paced fashion world. Therefore, the industry is moving toward using Artificial Intelligence (AI) to learn from the collected data and predict fashion trends and prices.

Fashion data are often images from various sources, e.g., e-stores, brand websites, Facebook, and Instagram. These data are usually, to some extent, enriched by metadata such

✉ Arda Goknil
arda.goknil@sintef.no

Rodrigo Rico Gómez
rodricgom97@gmail.com

Joe Lorentz
joe.lorentz@datathings.com

Thomas Hartmann
thomas.hartmann@datathings.com

Inder Pal Singh
inder.singh@uni.lu

Tayfun Gökmen Halaç
tayfunhalac@galaksiya.com

Gülnaz Boruzanlı Ekinci
gulnaz.boruzanli@ege.edu.tr

[1] DataThings, 5 rue de l'industrie, 1811 Luxembourg, Luxembourg

[2] SnT, University of Luxembourg, Campus Kirchberg, 1359 Luxembourg, Luxembourg

[3] SINTEF Digital, Oslo, Norway

[4] Galaksiya Information Technologies, Izmir, Turkey

[5] Department of Mathematics, Ege University, Izmir, Turkey

[1] https://medium.com/@andreaazevedo_32670/the-effects-of-the-52-micro-seasons-on-the-environment-edd87951b74f.

Published online: 17 May 2024

as color, garment category, brand, size, location and date. Examples of publicly available data sets are the DeepFashion database [3] and iMaterialist.[2] Extracting as much information as possible from fashion images is crucial to building Machine Learning (ML) models able to learn and predict trends and prices. Our objective is to develop, integrate, and leverage image processing and ML techniques to provide the online fashion industry, not only purely statistical information about online fashion products (e.g., the number of items sold online), but also predictions about online consumer behavior such as emerging items, fabrics, and colors among consumers. Those predictions will enable the fashion industry to optimize production lines and supply chains and align pricing strategies.

In this paper, we propose, apply, and assess a data analytics approach, i.e., FashionXpert, employing several image processing and ML techniques in an AI pipeline for price prediction for the fashion industry (see Fig. 1). The proposed approach integrates multiple heterogeneous data sources (e.g., textual and image data from e-stores, brand websites, and social media) to produce more consistent, accurate, and beneficial information. Textual data (e.g., product descriptions or customer comments on social media) do not provide enough information for price prediction. It needs to be augmented with data extracted through image processing, such as colors, garment category, fabric types, and cutting complexity. Therefore, we employ advanced, state-of-the-art computer vision models (i.e., Mask-RCNN [4], ResNet [5], and U-Net [6]) to extract additional and missing information from the collected images. FashionXpert first segments the image to localize individual garments. It analyzes each garment using ML models and image processing to extract additional information. FashionXpert then combines the collected metadata with the data extracted from the images to train ML models for price prediction.

There are several challenges and limitations we face in FashionXpert. They are mostly related to image processing tasks. The first one is about obtaining data. Some ML models FashionXpert employs require a fashion image data set labeled with ground truth masks and bounding boxes for garment detection and segmentation. It is costly and laborious to produce such training data sets. FashionXpert 's performance depends on the generation of labeled images. We use data augmentation techniques, i.e., horizontal flipping and random rotation ($\pm 20^{\circ}$) of labeled images, to address this limitation (see Fig. 13). The second major challenge concerns the garment classification problem. As presented in the following sections, we have defined sixty-six categories among which garments should be classified. The differences among these categories can be quite subtle and hard for the ML models to differentiate (see Fig. 14). Therefore, we split the classification problem into two subproblems (superclass and subclass identification) and define a superclass-subclass tree.

Another challenge is the segmentation of human skin to avoid propagating skin pixels through the pipeline, which is critical for the color extraction part. We employ a pre-trained skin segmentation model [6] to remove skin pixels after garment mask generation. Finally, the price prediction model has to deal with different target price distributions depending on the garment type (superclass). In addition, the most significant brands and countries vary between superclasses. To address this challenge, we build one price prediction model per each superclass and assign each superclass the set of relevant brands and countries we consider. The models are robust against outliers and missing data in order to ensure that the output follows the price distribution of each superclass.

We have evaluated FashionXpert on an industrial data set. Although not ideal, we consider the performance of Mask-RCNN effective with reliable results for superclass prediction. The total F1-score for superclass prediction is 67%, and the IoU-score for mask extraction is 81%. FashionXpert performed remarkably well in predicting subclasses (45% on average), taking into account the complexity of the problem to solve (see Fig. 14). The only prediction having accuracy under 40% is for the subclasses of the superclass *dress* due to its several subclasses (nineteen subclasses). The average (among all superclasses) Mean Absolute Error (MAE) of the price prediction model is 15.31 EUR, which is relatively a good result when considering the standard deviation of the price distribution in our data set (i.e., 72.99 EUR). Our main contributions are as follows.

- *Development of a novel AI pipeline* for garment price projection, integrating advanced computer vision techniques with market trend analysis. This pipeline not only predicts prices based on visual attributes of garments but also incorporates dynamic market data, setting it apart from existing methodologies.
- *Introduction of a comprehensive data set* that combines visual features with temporal market trends, enabling a more nuanced understanding of fashion pricing dynamics. This data set serves as a foundation for both training our AI model and facilitating future research in the field.

The rest of the paper is structured as follows: In Sect. 2, we present the background and related work. Section 3 describes the overview of FashionXpert. Sections 4 and 5 provide the core technical solutions. In Sect. 6, we report on the results of the empirical validation. We conclude the paper in Sect. 7.

[2] https://www.kaggle.com/c/imaterialist-fashion-2019-FGVC6/rules.

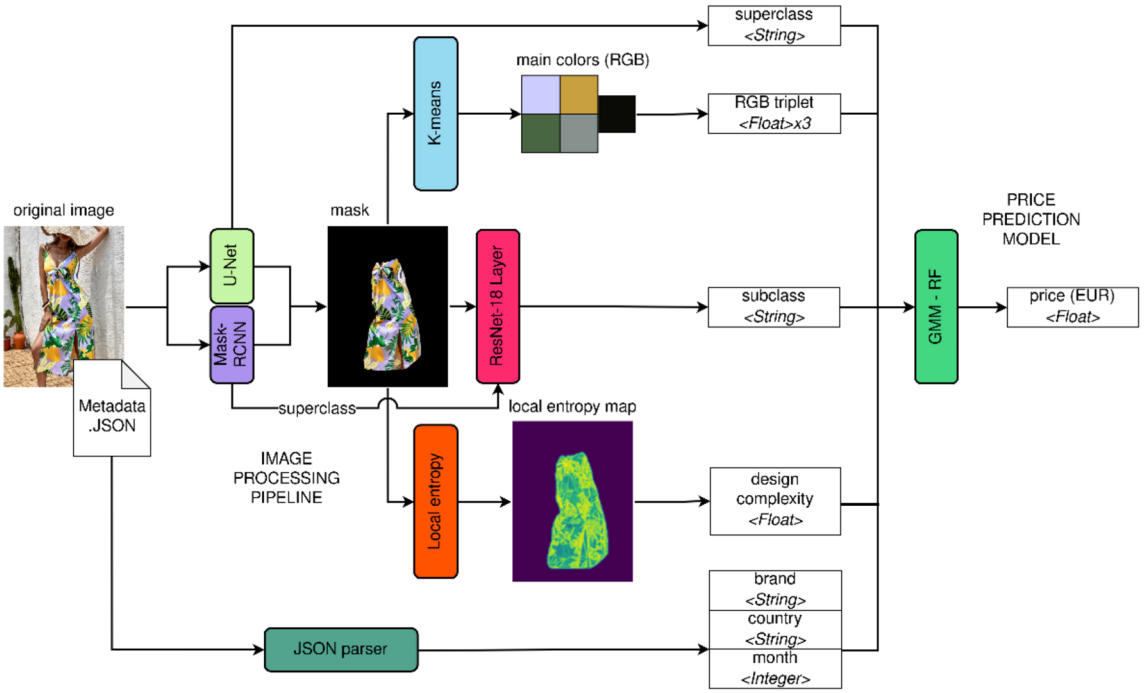

Fig. 1 The AI Pipeline of FashionXpert. It is composed by two main modules: image processing pipeline (IPP) and price prediction model (PPM). The IPP (Fig. 2) takes an image of the garment and retrieves visual fashion attributes: category (superclass and subclass), color, design complexity. We combine these attributes with metadata that influences the price (brand, country, month, size) and we input all these features in the PPM, that uses both a GMM (Gaussian Mixture Model) and RF (Random Forest), and outputs the predicted price

2 Related work

We review the literature on image processing (Sect. 2.1), price prediction (Sect. 2.2), and the application of AI to the fashion industry (Sect. 2.3).

2.1 Image processing

Our pipeline extracts fashion attributes (e.g., garment type, color, design complexity) from fashion images to build an efficient and robust price prediction model. These attributes constitute a natural vocabulary to describe styles in clothing and apparel. Chen et al. [7] introduce a new fashion data set from New York Fashion Show and follow a learning-based approach that adapts Support Vector Machines (SVM) to extract the fashion attributes from images. They apply pose estimation to retrieve the body region of the fashion model (i.e., part of the human wearing the garments in the image). Instead, we get the fashion attributes from an image processing pipeline composed of several computer vision models without any previous clue about the position or pose of the fashion model. Chen et al. aim to produce a data set, while our goal is to predict the price of the garments.

Yamaguchi et al. [8] propose one of the first computer vision approaches for the fashion domain [9]. They introduce a new parsing method to predict a person's outfit from both the image and garment tags using a Conditional Random Field (CRF) model. Earlier, Borràs et al. [10] develop a content-based image segmentation and interpretation method, describing garments using their color, texture, and structural compositions. The main difference between these two works above and our work is that they need to perform human pose prediction first, which is not ideal due to the inconsistent targets between pose estimation and garment detection [9].

Liang et al. [11] propose an approach similar to ours. They use deep Convolutional Neural Networks (CNN), decompose human images into semantic fashion/body regions, and formulate the human parsing task as an Active Template Regression (ATR) problem. They use two different CNNs in parallel to retrieve the mask of the regions. In contrast, we use Mask-RCNN for this task. We also extend their work by retrieving color and design complexity, apart from performing garment price prediction. Another example of using CNNs in the fashion domain is

Where To Buy It, the application developed by Kiapour et al. [12]. The application extracts fashion attributes (garment category, color, pattern, and material) from street fashion images to query shop websites for similar garments. Although our goal is to predict future garment prices, our feature extraction problem is similar. The main limitation of the *Where To Buy It* application is that it requires the user to specify the location of the garment inside the image, while we do not.

Some other recent works [13, 14] use CNNs too. They exploit the latest advancements in ML by training a deep neural network for the feature extraction part. More specifically, they adopt ResNet [5] and AlexNet [15] for the problem of fashion attribute extraction. For instance, Al-Halah et al. [14] learn the fashion attributes (e.g., color, fabric, shape, texture) using AlexNet-like architecture [15] and augment them by employing a smoothing model using text data to discover fashion styles. Jin et al. [13] extract the fashion attributes in the fashion images by using a CNN [15] trained on DeepFashion [3] data set. In contrast, instead of adopting ResNet or AlexNet, we build our pipeline based on Mask-RCNN and add additional models on top of Mask-RCNN. We use ResNet to extract the garment type but combine it with other models (e.g., U-Net, K-means, local entropy) to extract other attributes (RGB colors and design complexity).

Jia et al. [16] introduce *Fashionpedia*, where they adopt Mask-RCNN to perform garment attribute extraction on top of garment detection and classification. Our image processing pipeline is similar to Fashionpedia. However, we employ a skin segmentation module using U-Net [6], a layer of ResNet-18 [5] models for better garment classification, color extraction, and design complexity estimation.

There are some approaches addressing skin segmentation in the literature. Sreekumar et al. [17] propose a technique using U-Net architectures for hand segmentation for the first time. Also, Xie et al. [18] use a U-Net-based method successfully to segment faces, necks, and shoulders in images. Our pipeline similarly uses a U-Net architecture to remove faces, hands, legs, and other skin elements that can potentially introduce noisy pixels in garment masks. As for color extraction, Pavan Kumar et al. [19] probe the suitability of the K-means clustering algorithm [20] for unsupervised color extraction. We use U-Net and K-means for the same purpose these works do but in a different domain. The two works of Xie et al. and Pavan Kumar et al. provided proof of the suitability of U-Net and K-means for two tasks (i.e., skin segmentation and color extraction, respectively) in our work.

The main difference between our work and the existing works mentioned above is that we introduce a novel, full-fledged AI pipeline using Mask-RCNN followed by U-Net as the core model with three branches (i.e., K-means, ResNet-18, and local entropy) that extract fashion attributes (e.g., garment type, colors, design complexity). FashionXpert predicts not only the garment superclass (e.g., top, dress, bottom) but also the subclass (e.g., evening dress, blouse top, wrapped, polo). Instead of using labeled data to predict the fabric or pattern, we measure the design complexity of garments by using local entropy. Also, similar to Pavan Kumar et al. [19], we extract the color in an unsupervised way, i.e., with K-means clustering, without requiring labeled data. Getting the design complexity and color using unsupervised methods is crucial due to the cost of data sets.

Our image processing framework incorporates cutting-edge computer vision techniques available until 2022, exploring novel methodologies, particularly for garment detection. Innovations such as visual transformers [21] and masked autoencoders for self-supervised learning [22], culminating in comprehensive models like Meta's Segment Anything Model [23], herald new avenues for enhancing image segmentation. While our methodology primarily employs supervised learning and CNNs for garment detection, the advent of Transformer models presents exciting prospects for advancing object detection capabilities [24, 25], reflecting a significant shift toward experimental approaches in garment detection leveraging the latest AI advancements.

2.2 Price prediction

Some existing approaches [26–28] employ the time-series modeling of historical price data and Recurrent Neural Networks (RNN) (or its variant called Long-short term memory (LSTM) [29] networks) to predict the future price of a fashion item. For instance, Kedia et al. [26] leverage machine learning to find the optimal price point for products in fashion e-commerce platforms. To this end, they generate price-demand pairs using the concept of price elasticity, i.e., as the price increases, the demand goes down and vice versa. They employ an ensemble of multiple regressors (LSTM, ARIMA, Random Forest, and XGBoost) to build a demand prediction model that predicts the demand for a particular product for the next day based on its current price (discounted).

FashionXpert relies on computer vision and metadata rather than time-series modeling. Even though the works mentioned above achieve good performance in price prediction, their architecture complexity is mostly high as they follow time series forecasting, i.e., predicting the future value by observing the historical ones, modeled using

LSTMs. On the contrary, FashionXpert provides a novel combination of multiple computer vision tasks for fashion attribute prediction (e.g., color, category, design complexity) and other metadata (e.g., country, brand) to achieve price prediction. More specifically, we learn fashion attributes on a pixel level using Mask-RCNN [4] and other models on top of it, which helps extract the mask of the fashion item along with its classification and localization in the image. We propose using Gaussian Mixture-based regression to effectively map the metadata to the learned fashion attributes.

Advancements in AI and ML have significantly impacted price prediction research not only in the fashion domain but also in other domains like stock trading. Ferreira et al. [30] systematically reviewed AI in stock market trading, covering various models and their performance. FashionXpert distinguishes itself from those models by applying AI to the fashion industry, proposing a novel use case of AI for garment price prediction based on design complexity and visual features. Warner et al. [31] explored hybrid genetic algorithms and BERT for stock market prediction, highlighting AI's potential in financial forecasting. While this study emphasizes the integration of textual analysis and historical data, FashionXpert focuses on visual attributes of garments for price prediction, indicating a domain-specific application of AI in understanding fashion items' value.

Mohanty et al. [32] proposed a hybrid model combining autoencoders and kernel extreme learning machines for financial market prediction. This research underlines the efficacy of deep learning in capturing complex market dynamics. Similarly, FashionXpert leverages deep learning, particularly CNNs, for feature extraction from fashion images, showcasing the adaptability of deep learning techniques across different contexts. Mokhtari et al. [33] discussed machine learning algorithms' role in stock market predictions, focusing on technical and fundamental analysis approaches. In contrast, FashionXpert applies AI to decode visual fashion attributes, illustrating the diverse applicability of AI across sectors.

2.3 AI in the Fashion Industry

The fusion of AI with the fashion industry has seen remarkable growth, introducing innovative solutions for design creation, sustainability, and customer experience optimization [34–38]. For instance, Ciklacandir et al. [34] employed several feature extraction methods, including Discrete Cosine Transform, Principal Component Analysis, Gray Level Co-occurrence Matrix, and deep learning architectures like ResNet18 and GoogLeNet, for fabric defect detection. The extracted features were classified using K-Nearest Neighbor, Support Vector Machine, and Decision Tree classifiers to enhance defect detection in textiles. This contrasts with FashionXpert, which applies AI for garment price prediction, analyzing visual attributes through computer vision. While the former addresses quality control in textile manufacturing by identifying defects, FashionXpert leverages AI to understand market-driven aspects of fashion, such as pricing strategies based on garment features. Zhong et al. [35] introduced a data set (TextileNet) for ML models for textile material classification to support sustainable fashion initiatives, featuring over 760,949 images classified into detailed fiber and fabric categories. While TextileNet focuses on material identification crucial for sustainability, FashionXpert applies AI to determine garment prices, reflecting the diverse applications of AI in enhancing both sustainable practices and commercial strategies within the fashion industry.

FashionXpert distinguishes itself by specifically leveraging AI for garment price prediction, a niche yet critical aspect of the fashion industry's economic landscape. While the reviewed works demonstrate AI's expansive role in enhancing creativity, sustainability, and operational efficiency, FashionXpert uniquely applies AI to decode the complexities of fashion pricing, offering innovative and commercially valuable insights.

3 Overview of the approach

Figure 1 presents an overview of FashionXpert which has two parts: (i) the image processing pipeline and (ii) the price prediction model. These two parts are fully automated. Sections 4 and 5 provide their details. The image processing pipeline contains all the computer vision models that extract relevant features required for price prediction from the images (see Fig. 2). We propose using Mask-RCNN to perform garment segmentation and classification. We employ a U-Net architecture model to retrieve the skin from the images. To provide a detailed garment classification, we use ResNet-18. We also propose K-means clustering to get the primary colors of the garment, while we get the design complexity using the local entropy of the image.

Following the feature extraction through the image processing pipeline, the price prediction model constitutes the second critical component of FashionXpert. The model integrates the extracted features, including garment segmentation, skin detection, detailed classification, primary colors, and design complexity, to forecast the garment's price. Utilizing a Gaussian Mixture Model (GMM) with a Random Forest Model (RFM), the model is trained on a data set comprising these features alongside historical price data, allowing it to learn the intricate relationships between a garment's visual attributes and market value. By feeding

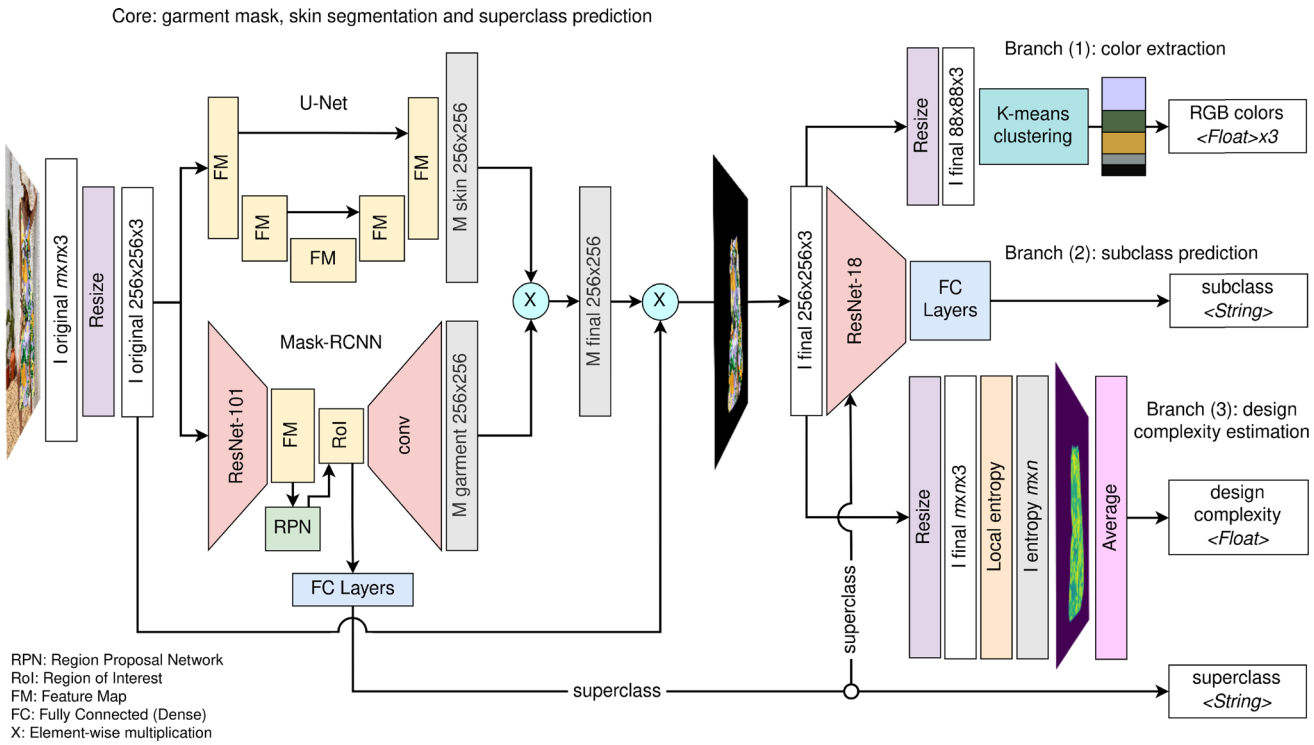

Fig. 2 The image processing pipeline in FashionXpert. The core model is Mask-RCNN that performs garment mask extraction and superclass prediction, we also apply a skin segmentation network with U-Net architecture. On top of it we build three branches: (1) color extraction using K-means, (2) subclass prediction using a layer of ResNet-18 models and (3) design complexity estimation using the local entropy of the garment mask

the comprehensive feature set into the price prediction model, FashionXpert achieves a nuanced understanding of how different attributes contribute to pricing, facilitating accurate and dynamic price predictions. The integration of these components forms a cohesive and automated system, detailed further in Sects. 4 and 5, designed to revolutionize garment price prediction by leveraging the full potential of AI and machine learning technologies.

4 Image processing pipeline

We propose using a combination of models (see Fig. 2) to predict the taxonomy of the garments in the input image, the principal colors of each garment as RBG, and its design complexity. The pipeline includes (i) a Mask-RCNN we use to have the initial garment classification and the mask of the garments in the image, (ii) a U-Net performing skin segmentation, (iii) a layer of ResNet-18 models to address further the garment classification, and (iv) K-means clustering for color extraction and local entropy to estimate the design complexity. Figure 3 presents the image processing pipeline step-by-step with an example.

4.1 Garment Taxonomy and Classification Problem

We provide a garment taxonomy having nine superclasses and sixty-six subclasses for FashionXpert. Given the low number of samples on our data set (13,138), it is difficult to train an ML model that distinguishes 66 categories with enough accuracy. Therefore, we first train a model to solve a 9-category classification. For each category, we train another model to predict the subclasses. In other words, we split the classification problem into two subproblems (superclass and subclass identification) and define a superclass-subclass tree, as shown on Fig. 4.

Each garment has a tuple of superclass-subclass that should be predicted using the image. The taxonomy in Fig. 3 provides what type of garment (e.g., dress, top) and subclass (e.g., wrap dress, evening dress) we have in the input image. The taxonomy structure is a tree-like graph of $depth = 2$. Nodes with $depth = 1$ are superclasses (S), and nodes with $depth = 2$ are subclasses (T). Therefore, each subclass is only connected to a single superclass, which enables us to split the problem into two subproblems. First, we segment the image by localizing garments and predicting superclass labels ($s_k \in S$). Second, we determine subclasses to refine the labels ($t_{kl} \in T_k$).

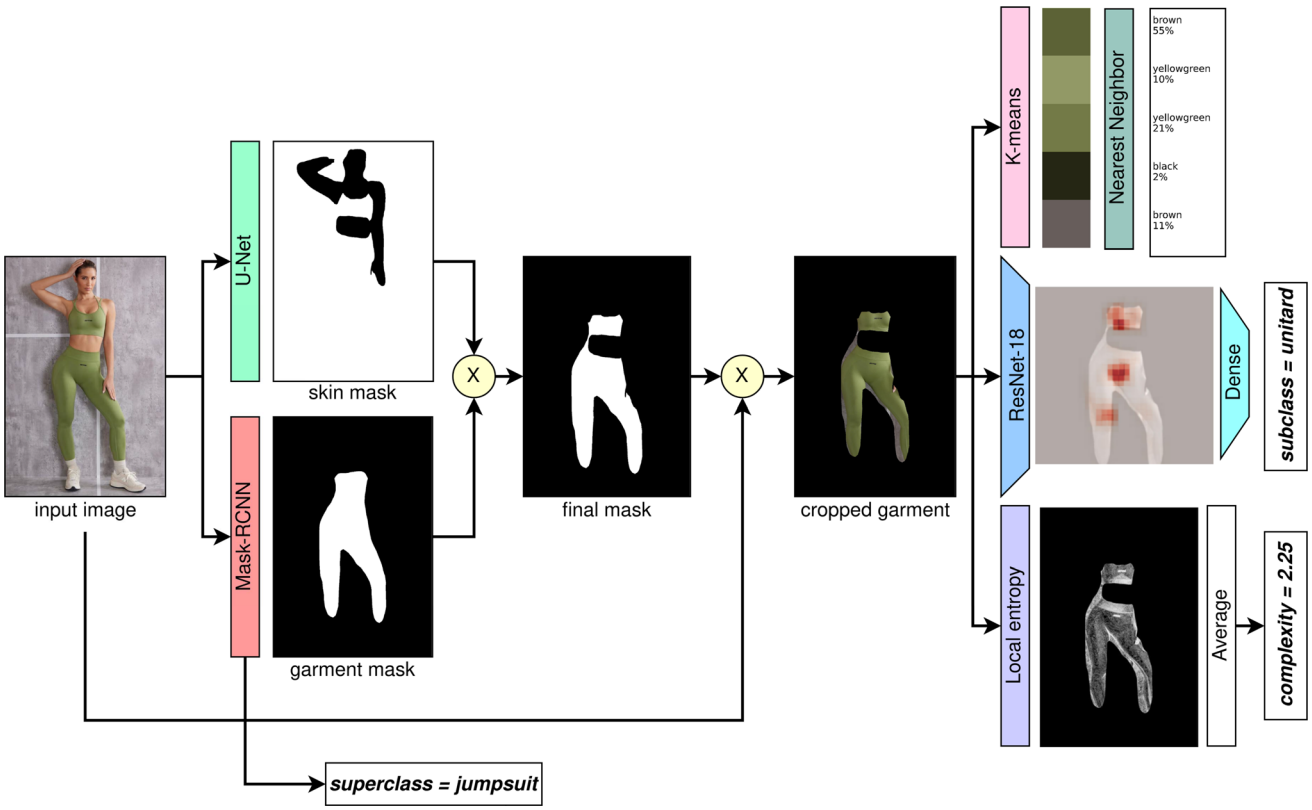

Fig. 3 Illustration of the image processing pipeline steps in Fig. 2 using an example image

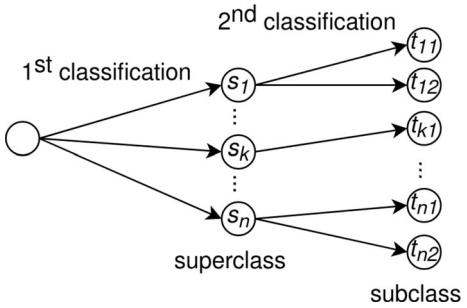

Fig. 4 Taxonomy with superclasses and subclasses

4.2 Mask extraction and superclass prediction

Before predicting the superclasses, we need to localize garments in the image. More specifically, we identify the image pixels that belong to a specific garment (i.e., its mask). To do so, we use Mask-RCNN [4]. Mask-RCNN performs object detection, segmentation, and classification [4]. We propose using it for mask extraction and superclass prediction, playing a central role in the application. Figure 2 illustrates the role of MASK-RCNN in our pipeline. Mask-RCNN performs the following function:

$$\mathcal{F} : I^{(\text{original})} \longrightarrow (M^{(\text{garment})},\ s_k \in S,\ p \in [0,1]) \tag{1}$$

where $I^{(\text{original})} \in [0,255]^{256\times256\times3}$ is the input resized image, $M^{(\text{garment})} \in \{0,1\}^{256\times256}$ is the mask identifying the pixels of a garment, S is the set of superclasses, and p the normalized prediction confidence.

Next, we identify the garment image $I^{(\text{garment})} \in [0,255]^{256\times256\times3}$ by computing the intersection between the mask and the input image:

$$I^{(\text{original})} \odot M^{(\text{garment})} = I^{(\text{garment})} \tag{2}$$

Since Mask-RCNN mostly returns more than one prediction for each image, we have to choose one. By default, we select the prediction with the highest probability p. However, we allow FashionXpert users to choose among the alternatives.

4.3 Skin segmentation

Once Mask-RCNN retrieves the mask, the pipeline employs a U-Net model [6] to perform skin segmentation (Fig. 2). The model weights have been transferred from an external repository.[3] without further training. The skin segmentation is crucial to avoid propagating noisy skin pixels to the later stages of our pipeline.

[3] https://github.com/MRE-Lab-UMD/abd-skin-segmentation.

The U-Net model receives the image $I^{(\text{original})}$ and produces the skin mask $M^{(\text{skin})}$ that we subtract from the garment mask $M^{(\text{garment})}$ to produce the final output mask $M^{(\text{final})}$:

$$M^{(\text{garment})} \odot M^{(\text{skin})} = M^{(\text{final})} \tag{3}$$

where $M^{(\text{skin})} \in \{0,1\}^{256\times256}$ is computed by the U-Net model ($\mathcal{U}$).

$$\mathcal{U} : I^{(\text{original})} \longrightarrow M^{(\text{skin})} \tag{4}$$

We compute the intersection of $I^{(\text{original})}$ and $M^{(\text{final})} \in \{0,1\}^{256\times256}$ for the final segmentation image $I^{(\text{final})}$:

$$I^{(\text{original})} \odot M^{(\text{final})} = I^{(\text{final})} \tag{5}$$

4.4 Garment color extraction

We identify the garment colors in the image as they may have a crucial impact on the product price. The image processing pipeline in Fig. 2 employs the k-means clustering algorithm [20] ($\mathcal{K}$) to extract the colors in the final image:

$$\mathcal{K} : (I^{(\text{final})},\ k) \longrightarrow C \in [0,255]^{3\times k} \tag{6}$$

where the hyper-parameter k indicates the number of clusters to produce and C is the set of positions in RGB coordinates of the k clusters' centroids. k is set to 5 by default. FashionXpert users can change the default value when using the application.

Every pixel is treated as a point in a three-dimensional space defined by the RGB color representation components (range of 0 to 255). We consider the primary colors as the RGB coordinates of the centroids. The algorithmic complexity of K-means is $o(n^2)$ [20], which is a potential bottleneck for FashionXpert and may slow down its real-time responsiveness. One way of significantly reducing the execution time of K-means might be to decrease the input image resolution. Therefore, we tried several image resolutions and measured the execution time for each one. Since there is no labeled data for garment colors, we checked the correct performance of color extraction qualitatively. We observed that, beyond 88×88, the color extraction stops being the bottleneck of the pipeline, and its execution time is already comparable with the rest of the stages of the pipeline. At this size, reducing image resolution does not make a big difference in the accuracy of the model but it leads to a significant reduction in execution time. With images of 88×88 resolution, the clustering algorithm encounters no substantial problems.

Once we have a set of RGB triplets as the output of K-means, we assign each triplet a label using a Nearest Neighbor approach (see Fig. 5). We define a set of principal colors whose coordinates we know in the RGB space. We then measure the Euclidean distance from the output RGB triplet to each principal color in the RGB space. The color label showing the closest distance is the label we assign to our unlabeled triplet.

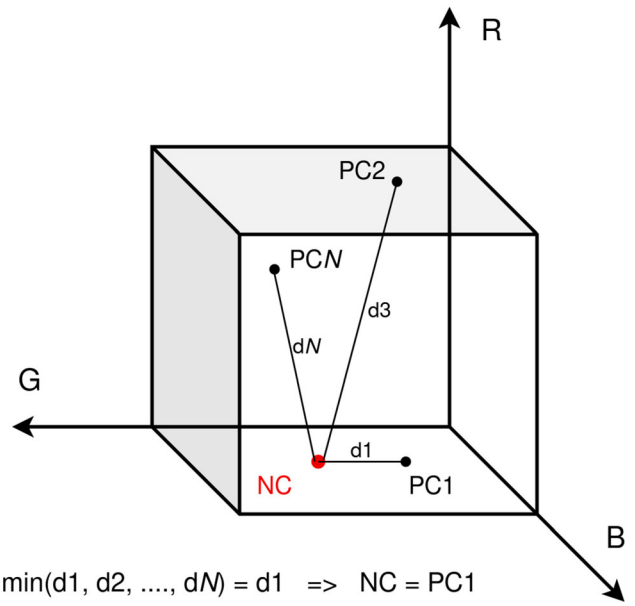

Fig. 5 Color categorization by Nearest Neighbor. PC stands for Principal Color, and NC stands for New Color

4.5 Subclass prediction

The pipeline employs a ResNet-18 model [5] to predict subclasses (see Fig. 3). The input of this model is the image $I^{(\text{final})}$ computed after skin-pixel removal. The model classifies $I^{(\text{final})}$ among a set of categories T_k that corresponds to the superclass s_k. Subclass prediction ($\mathcal{R}$) is done exclusively by the ResNet-18 architecture:

$$\mathcal{R} : I^{(\text{final})} \longrightarrow t_{kl} \in T_k \tag{7}$$

The pipeline uses a different ResNet-18 model based on the superclass s_k obtained by Mask-RCNN. The architecture is identical among all the superclasses, but the models are trained specifically to classify the image among the subclasses corresponding to the given superclass.

4.6 Design complexity

In addition to garment type and color, the complexity of a garment design may also impact its price under certain circumstances. Thus, we devised a novel method in the image processing pipeline to quantify the design complexity of a garment using its image. The method uses the local entropy measurements of the final garment image

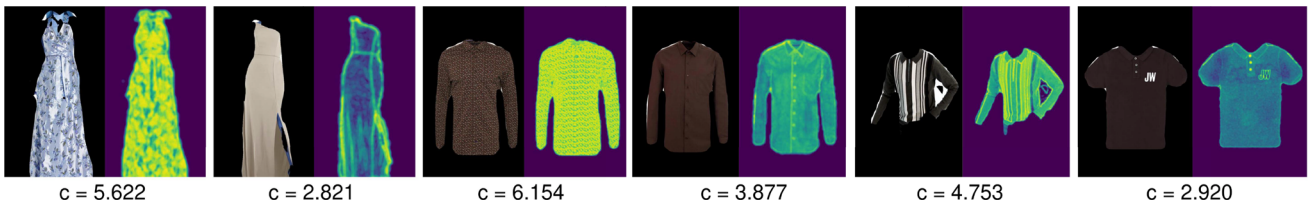

Fig. 6 Examples of design complexity calculations. We show both the segmented mask image (reshaped to original size) and the local entropy image. The number c represents the design complexity of the garment. Higher value indicates higher design complexity

$I^{(\text{final})}$. Considering only $I^{(\text{final})}$, we avoid noise from the background and skin.

Local entropy is related to the complexity contained in a given pixel neighborhood [39]. In our case, the local entropy computation is a function $\mathcal{S}$ that takes an RGB image $J \in [0, 255]^{m \times n \times 3}$ and gives another image $K \in (\mathbb{R}^+)^{m \times n}$:

$$\mathcal{S} : (J,\ r) \longrightarrow K \tag{8}$$

The idea of the local entropy is to compute per-pixel entropy using a sliding window that covers the pixel itself and the surrounding pixels up to a certain radius. The formula is just an adaptation of the Shanon entropy Eq. (9). Each element i, j of K is the local entropy value of the neighborhood of pixel i, j in the image J. The neighborhood is defined under the disk radius $r \in \mathbb{N}$ [39].

$$\mathcal{S}(J_r) = \sum_{i=1}^{r} P(X_i) \log_2 P(X_i) \tag{9}$$

We compute the global entropy (or complexity) of a garment by averaging over all the pixels' values of its local entropy image (see Fig. 6 for an example design complexity calculations).

5 Price prediction model

FashionXpert employs a hybrid approach for price prediction, combining a Gaussian Mixture Model (GMM) with a Random Forest Model (RFM) (refer to Fig. 7). The image processing pipeline extracts features, including superclasses, subclasses, primary RGB colors, and design complexity, which are then combined with metadata such as country, month, and brand to form a comprehensive feature vector. Most of these features are categorical, with the exception of design complexity and primary RGB colors, which are represented numerically. The RFM utilizes the numerical attributes (color and design complexity) to generate a price estimate, which serves as a correction factor for the price predicted by the GMM. The RFM's price estimation output, in conjunction with categorical features (superclass, subclass, brand, month, and country), is then fed into the GMM for the final price prediction (see Sect. 5.1 for detailed information on the GMM).

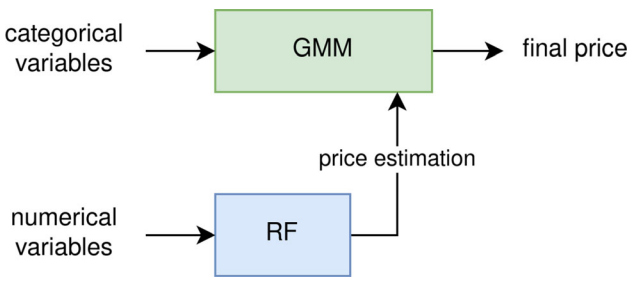

Fig. 7 Architecture of the price prediction approach

Due to the preponderant impact of categorical variables on price, and the prevalence of such predictors in our data set, we have opted to utilize a Gaussian Mixture Model (GMM). GMMs are inherently robust against outliers and noise in the data. In our context, we define noise as data samples automatically sourced from fashion websites without accurately reflecting reality, e.g., some suits in the data set have been labeled with a much lower price as the web price only accounts for the bottom part. These errors in data acquisition contribute to a highly noisy data set. Additionally, a GMM can accommodate missing data when no samples match all the combinations of categorical variables. For instance, no data is available for Adidas leggings in France, while price prediction for Adidas leggings in France are expected from production data. After experimenting with various architectures of Random Forest and Neural Networks, we have discovered that data noise significantly influences the accuracy of the predictions. Despite the fact that the training procedure results in an acceptable convergence of the loss function, the predictions tend to deviate significantly from the actual price distributions when provided with inputs that differ from those utilized in the model's training. However, by utilizing a GMM, which possesses robustness, we can ensure that, regardless of the combination of categorical variables that we input, the predicted price will align with the observed price distributions in the market.

We have designed the GMM algorithm to deal exclusively with categorical variables. But we still have numerical variables in the input feature vector. Since these

variables have much less impact on the price than the categorical ones, we have decided not to include them in the GMM architecture. Then, we need another way to decode its information. We have decided to process them using a RFM. This model will decode the information of the numerical variables and will output a price estimation based on it. This estimation will be used later as an additional input for the GMM to perform the price predictions. The details of this operations are explained in Sect. 5.1, but essentially the resulting price is mixed with the mean prices output by the GMM. The key in this process is converting color and complexity information in price information that can be used later in the algorithm without further processing.

5.1 Gaussian mixture model

The categorical features, namely superclass, subclass, country, brand, and month, undergo processing via a Gaussian Mixture Model (GMM) approach. We develop a unique GMM per superclass, with subclass, country, brand, and month serving as inputs to this section of the Price Prediction Model. These variables are utilized to partition the total data set into Gaussian Slots (GS), which comprise subsets of data possessing a specific combination of the four input variables (see Fig. 8). Given a set of values for these variables, the GMM output represents the mean price of the GS defined by this particular set of values. For instance, to determine the price of a Zara t-shirt in Germany during March, we obtain the mean price of the garment offered in the data set with matching attributes (i.e., t-shirt, Zara, Germany, and March). Up to this point, we have a rudimentary implementation of the GMM, which performs the following operation:

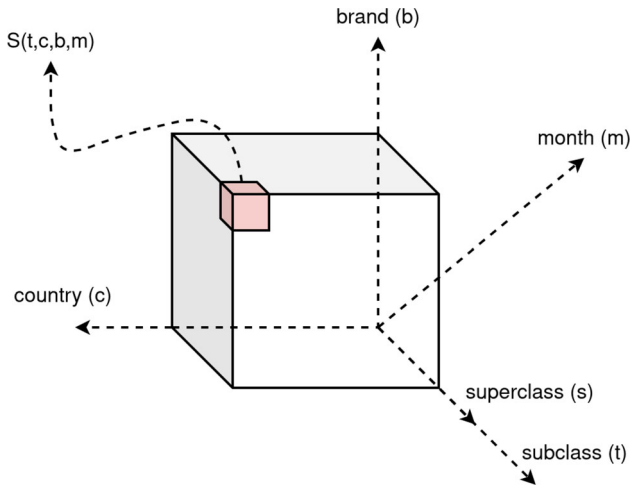

Fig. 8 Graphical representation of the concept of Gaussian Slot (GS)

$$\begin{aligned} &\mathcal{GMM} : (t_{kl}, c_i, b_j, m_h) \longrightarrow \text{mean}(S) \\ &\text{with} \quad S = P(t = t_{kl} | c = c_i | b = b_j | m = m_h) \end{aligned} \tag{10}$$

where P is the total price data set, t_{kl} is the subclass, c_i is the country, b_j is the brand, m_h is the month, and S is the GS subset. The expression $P(c_1|...|c_n)$ denotes the subset of P that satisfies all the conditions $c_1, ..., c_n$. In terms of logical operations, $(a|b)$ would be (a AND b).

Ideally, the data set should contain sufficient samples for all possible combinations of input variables to ensure that each Gaussian Slot (GS) has a representative number of samples. However, missing data is a common issue that should be addressed. In such cases, we infer the content of a GS from other higher-level GSs that have enough samples. For instance, if there are insufficient samples in a GS, (e.g. t-shirt, Zara, Germany, March), we relax the matching conditions and estimate the price of a t-shirt in Germany in March, irrespective of the brand. Alternatively, we use the mean price of Zara garments to improve the estimation. To estimate the content of a GS, we rely on two other GSs, namely GS1 (t-shirt, Germany, March) and GS2 (Zara). If GS1 still lacks sufficient samples, we split it until we obtain adequate samples for price estimation. Our GMM uses this strategy to deal with the absence of training data. However, each time we split a GS, we introduce an error in the prediction since we must infer a price distribution. Figure 9 shows all the prediction steps of our approach using a GMM.

Algorithm 1 outlines the steps involved in the GMM-based price prediction process of FashionXpert. To provide a comprehensive understanding of the process, we will delve into each step of the algorithm. The algorithm requires various inputs, including the price data set (P) for training, superclass (s_k), subclass (t_kl), country (c_i), brand (b_j), month (m_h), and an array of numerical features (v) - the RGB components of the principal color and the design complexity. First, we obtain subsets of the total price distribution (Lines 1–3). The price distribution is filtered for a given superclass $S^{(s)}$ (Line 1), for superclass, subclass, country, and brand $S^{(stcb)}$ (Line 2), and for superclass and month $S^{(sm)}$ (Line 3). The process begins by defining two Gaussian slots (refer to Fig. 9a). We calculate r_m (Line 4), which is the ratio of $mean(S^{(sm)})$ to $mean(S^{(s)})$. This coefficient r_m represents the influence of the variable $m = m_h$ on the price prediction. We also compute r_{tcb} (Line 5) and obtain p_v (Line 6) by feeding v into the RFM. Finally, we calculate r_v (Line 7), which is the ratio of p_v to $mean(S^{(s)})$.

Algorithm 1 Price Prediction

Require: $P, s_k, t_{kl}, c_i, b_j, m_h, v$
Ensure: p
1: $S^{(s)} \leftarrow P(s = s_k)$
2: $S^{(stcb)} \leftarrow P(s = s_k | t = t_{kl} | c = c_i | b = b_j)$
3: $S^{(sm)} \leftarrow P(s = s_k | m = m_h)$
4: $r_m = mean(S^{(sm)})/mean(S^{(s)})$
5: $r_{tcb} = mean(S^{(stcb)})/mean(S^{(s)})$
6: $p_v = RF(s = s_k, v)$
7: $r_v \leftarrow p_v/mean(S^{(s)})$
8: **if** $len(S^{(stcb)}) \geq 4$ **then**
9: $\quad p \leftarrow mean(S^{(stcb)}) \cdot r_{tcb} \cdot a(r_m) \cdot a(r_v)$
10: **else**
11: $\quad S^{(scb)} \leftarrow P(s = s_k | c = c_i | b = b_j)$
12: $\quad S^{(st)} \leftarrow P(s = s_k | t = t_{kl})$
13: $\quad r_{cb} \leftarrow mean(S^{(scb)})/mean(S^{(s)})$
14: $\quad r_t \leftarrow mean(S^{(st)})/mean(S^{(s)})$
15: $\quad$ **if** $len(S^{(scb)}) \geq 4$ **then**
16: $\quad\quad p \leftarrow mean(S^{(s)}) \cdot r_{cb} \cdot r_t \cdot a(r_m) \cdot a(r_v)$
17: $\quad$ **else**
18: $\quad\quad S^{(sc)} \leftarrow P(s = s_k | c = c_i)$
19: $\quad\quad S^{(sb)} \leftarrow P(s = s_k | b = b_j)$
20: $\quad\quad r_c \leftarrow mean(S^{(c)})/mean(S^{(s)})$
21: $\quad\quad r_b \leftarrow mean(S^{(b)})/mean(S^{(s)})$
22: $\quad\quad p \leftarrow mean(S^{(s)}) \cdot r_c \cdot r_b \cdot r_t \cdot a(r_m) \cdot a(r_v)$
23: $\quad$ **end if**
24: **end if**

The adequacy of the available samples (threshold = 4) in $S^{(stcb)}$ is verified (Line 8). We chose 4 because, due to the low number of data in our training data set, we consider it acceptable to provide a price estimation. If this condition is satisfied, the price is estimated as the product of $mean(S^{(s)})$ and the three impact coefficients r_{tcb}, r_m, and r_v (Line 9). To account for outliers, the coefficients r_m and r_v are subjected to an activation function, which takes the form of a sigmoid function with an upper bound of 1.2 and a lower bound of 0.8, thereby smoothing the impact of the month and the numerical variables on the final price prediction by allowing fluctuations of up to 20%. This percentage value has been set intuitively, and then we have checked that we obtain the outcome we pursue.

In case there are insufficient samples in $S^{(stcb)}$, the algorithm proceeds to split the Gaussian slot into two higher-level ones: $S^{(scb)}$ (Line 11) and $S^{(st)}$ (Line 12). In contrast to filtering the price data set for country, brand, and subclass, we filter it for brand and country and for subclass, combined with the previously defined month $S^{(sm)}$, representing the second attempt of the algorithm to obtain the price estimation (Fig. 9b). We compute the impact coefficients of the brand and country (Line 13) and the subclass (Line 14). The impact coefficients of the month and the numerical features were calculated beforehand. If there are sufficient samples in $S^{(scb)}$ (Line 15), we determine the price by applying the impact coefficients to $S^{(s)}$ and passing r_m and r_v through the activation function a (Line 16).

In the case where there are still insufficient samples in $S^{(scb)}$, the Gaussian slot is divided into $S^{(sc)}$ (Line 17) and $S^{(sb)}$ (Line 18). We derive the impact coefficient of the country r_c (line 20) and the brand r_b (Line 21) separately. The price is computed (Line 22) in the same manner as before (Lines 9 and 16). This marks the final attempt of the price prediction model (see Fig. 9c). Since all slots are filtered using only one variable in addition to the superclass, further division is not feasible.

Throughout the price prediction steps, we assess sample sufficiency (4 samples) in $S^{(stcb)}$ (Line 8) and $S^{(scb)}$ (Line 15) due to the fact that these are the only slots that filter the data set based on the superclass s in combination with at least two other variables *tcb* or *cb*). The remaining slots filter the data set on the superclass and one other variable (at most), where the latter is guaranteed to be non-empty, due to the restriction of working only with categories available within each superclass. Specifically, for a given superclass s, there is a list of its most frequent brands $B^{(s)}$, and thus for any possible brand b_j in the list of the superclass $B^{(s)}$, it is impossible for $S^{(sb)} = \emptyset$. This property extends to all categorical variables.

Each of the three splits made in the algorithm separates one variable from the others on the filter. The first split separates the month variable m from the rest (Lines 2 and 3), followed by the separation of the subclass variable t from the country and brand variables (Lines 11 and 12) in the second split, and finally, the separation of country and brand variables in the last split (Lines 18 and 19). The order of these separations is based on the variable independence level and the impact of the variables on the price. The month variable is the most independent and thus split first to minimize the error. The brand and country variables are highly dependent and have a higher impact on the price, so they are avoided until the last attempt to minimize the error. The algorithm splits the variables less costly to separate, in terms of error, before the more expensive ones.

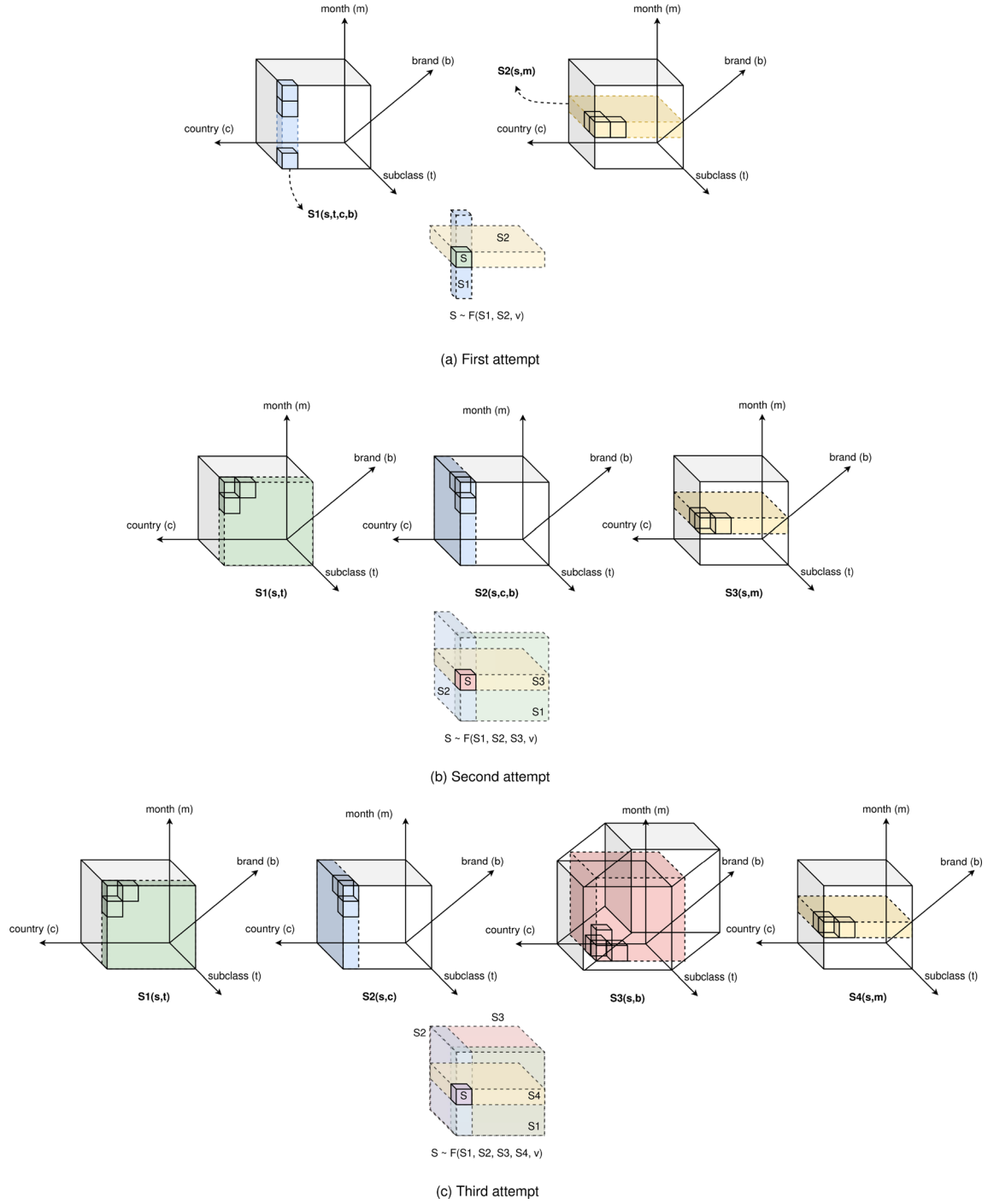

Fig. 9 The price prediction steps of FashionXpert using a GMM

6 Evaluation

In this section, we investigate, based on our industrial data set in the fashion domain, the following Research Questions (RQs):

- *RQ1*. Do the fashion attributes we extract affect garment pricing? The goal of this research question is to investigate whether the fashion attributes we extract affect garment pricing and, in turn, whether we can use them in our price prediction model.
- *RQ2*. Is the superclass prediction and garment segmentation model effective? The goal of this research question is to assess whether FashionXpert identifies, in a reliable manner, the superclasses of our garment

taxonomy in input images while getting an accurate mask of the garments.

- *RQ3*. Is the subclass prediction model effective? The goal of this research question is to assess whether the ResNet-18 Layer of FashionXpert identifies, in a reliable manner, the subclasses of our garment taxonomy in input images.
- *RQ4*. Is the price prediction model effective? The price prediction model depends on the image processing pipeline output and metadata. We investigate in this research question whether the price prediction model of FashionXpert mitigates the effects of missing data and outliers in the data sets and predicts garment prices accurately.

All models employed by our solution described in Sects. 4 and 5 are built using supervised learning algorithms. Therefore, they should be trained with labeled data (training data set) and tested using a data set different than the training one. We address each RQ in a separate subsection. We first present how the models are trained and tested for each RQ (in each subsection except for RQ1). Then, we report the results obtained for RQs. Due to the lack of labeled data, some parts of the pipeline are unsupervised, (e.g., color extraction and design complexity estimation). So, it is not possible to provide an accuracy metric, because there is not ground truth data. Since we use their outputs to perform price prediction (see RQ4), we consider them successful as long as they contribute to reaching good results when evaluating the price prediction model.

6.1 Subjects of the evaluation

We used an industrial data set (combining two sub-data sets) to evaluate image processing and price prediction in FashionXpert: (i) the bounding box data set and (ii) the price prediction data set. Table 1 compares our data set with various data sets in the literature used for object detection and classification in the fashion domain.

6.1.1 Bounding box data set

The data set used to train and evaluate the models of the image processing pipeline consists of 13,137 images. The images are divided into nine superclasses and sixty-six subclasses. Table 2 gives the number of images per superclass. Each image contains one labeled garment with a bounding box and a mask that indicates the location and pixels of the garment.

We used *follow.fashion* [1] (i.e., a fashion search tool automatically visiting the electronic commerce sites of clothing brands) to collect garment images for the bounding box data set. *follow.fashion* enables users to monitor the price and stock changes on clothing products. It visits each website in 24-hour periods and obtains the price and stock information of the clothing products together with product images. We gathered the product images from the follow.fashion database and used the *labelme* tool[4] (i.e., an online annotation tool to build image databases for computer vision research) to draw the bounding boxes manually for the training data set.

6.1.2 Price prediction data set

This data set is a CSV file in which each row represents the metadata of the garment offer (i.e., the price, date, brand, year, and country of the garment sold). We added the ground truth superclass, subclass, and garment image to this metadata. All these data (superclass, subclass, brand, country, year, month, complexity, RGB components, and price) are input into the price prediction model. We applied the previously trained image processing pipeline to the garment image to obtain the design complexity and RGB components. Table 2 gives the number of rows and offers per superclass (in total, 128.4K) that have been used for training and testing our models. One garment image has several offers assigned, one for each combination of metadata values (i.e. brand, country, month, year). In order to build the price prediction data set, we use a different set of images than the ones in the bounding box data set.

The final price prediction data set CSV had several preprocessing stages. We first converted all the currencies to Euro (EUR) while considering the currency conversion rate in the European Central Bank database.[5] We addressed the high cardinality in the "brand" variable (3437 different brands as categorical data in the data set). Since we implemented a different price prediction model per superclass, we got for each the list of 35 most frequent brands. Samples with a different brand will fall into the category *other*. We kept only 5% of the samples with brand *other*, in order to avoid a huge imbalance of the data set on this variable. This way, we reduce the cardinality from 3437 to 36.

The price prediction data set includes the data acquired between 2016 and 2022. We obtained the data set from the follow.fashion database. Follow.fashion has a fashion ontology and supports natural language processing. It extracts the title, description, category, price, and stock

[4] http://labelme.csail.mit.edu/.

[5] We used Python Currency Converter.

Table 1 Different data sets in the literature used for object detection and classification

	FashionAI [40]	WTBI [12]	DeepFashion2 [41]	iMaterialist [42]	Our Data
# Images	357 K	425 K	491 K	1 M	78 K
# Categories	41	11	13	228	66
# Bounding boxes	100 K	39 K	801 K	1 M	13 K
# Masks	NO	NO	801 K	1 M	13 K
# Price prediction	NO	NO	NO	NO	65 K

All of them belong to the fashion domain. Note that FashionAI data set contains landmarks instead of bounding boxes. #Images indicates the total number of images in the data set. #Categories the number of classes. #Bounding boxes the number of garments labeled with a bounding box, #Masks the number of garments labeled with a mask. #Price prediction the number of garments labeled with its price

Table 2 Number of images per superclass in the bounding box data set (*N* bbox images)

Superclass	*N* subclasses	*N* bbox images	*N* offers
Co_ord	3	510	4585
Suits	1	200	3601
Outerwear	8	1295	17,001
Jumpsuit	7	1251	13,745
Dress	19	3569	40,419
Shirts	3	672	4464
Knitwear	2	700	6813
Bottom	8	1343	14,666
Top	15	3598	23,125

We also display the number of offers used for training and evaluating in the price prediction data set (*N* offers)

information from the product descriptions in natural language, and stores them in its database.

6.2 RQ1: Do the fashion attributes we extract in FashionXpert affect garment pricing?

We investigated the fashion attributes affecting garment pricing to answer which fashion attributes we can use in our prediction model. We used the price prediction data set having the garment prices (the target variable) along with the ground truth superclass, subclass, color,[6] brand, month, and country. We used the Shapiro-Wilk test [43] to test, for each variable, whether the price distribution is normal. The outcome was negative in all cases (i.e., the price distribution cannot be assumed normal), which abstained us from using ANOVA and forced us to use a nonparametric test (the Mann–Whitney U test [44]). This test gives a p value for each category of each independent variable. The p value solves the hypothesis contrast on Eq. (11).

$$\begin{aligned} H_0 &: P(x_i = k) = P(x_i \neq k) \\ H_1 &: P(x_i = k) \neq P(x_i \neq k) \end{aligned} \tag{11}$$

where $P(x_i = k)$ is the price distribution of the samples in which the independent variable x_i takes the value k, and $P(x_i \neq k)$ is the price distribution of the rest of the data set. If the p value is more than 0.05, we confirm the null hypothesis H_0; otherwise, we reject it and assume the alternative hypothesis H_1. Rejecting the null hypothesis implies $P(x_i = k) \neq P(x_i \neq k)$ and, in turn, that $x_i = k$ affects the price distribution. We consider a variable useful for price prediction when the p value < 0.05 for more than half of the categories of the variable. For instance, if the p value is lower than 0.05 for the category "Zara" of the variable "brand", that category affects the price, i.e., useful to predict the price. If the p value < 0.05 for more than half of the categories (brands in this case), then we consider the variable "brand" useful to predict the price.

Answer to RQ1: For each variable, we selected in the price prediction data set, more than half of the categories fulfill p value < 0.05. Therefore, we conclude that the fashion attributes we extract in FashionXpert affect garment pricing, and, in turn, we can use them in our price prediction model.

6.3 RQ2: Is the superclass prediction and garment segmentation model effective?

6.3.1 Model training

We trained Mask-RCNN for 30k optimization iterations with a batch size of four images. We determined the number of iterations empirically. Figure 10 presents the evolution of the top-1 classification accuracy of Mask-RCNN for superclass prediction (defined as the average of the values obtained performing fourfold cross-validation). The accuracy stops increasing in the interval between 30,000 and 37,500 iterations. We fixed the number of iterations as 30,000 to prevent over-fitting. We started from a model that had been pre-trained using the data set COCO

[6] Ground truth color comes in a categorical format (back, blue, red, etc.) instead of RGB components.

[45]. The learning rate during training was 2.5×10^{-4}. We used a multitask loss function to train classification, bounding box[7] regression, and mask prediction: $L = L_{clf} + L_{bbox} + L_{mask}$, where L_{clf} is the term of the loss function that measures the classification error, L_{bbox} quantifies the error in the bounding box estimation, and L_{mask} measures the error in the mask prediction. For a more detailed description of the loss function of Mask-RCNN, the reader is referred to He et al. [4].

Mask-RCNN's first operation was to get a feature map using a backbone network [4], i.e., a ResNet-101 formed by a convolutional stage and four residual blocks (five backbone network blocks in total). We relied on the pre-training to perform the first operations of feature extraction. Since we imported pre-trained weights, we ignored the first layers of the network while training. We froze the first two blocks of the backbone (the convolutional stage and the first residual block) and trained the rest of the network blocks with the parameters established. We implemented the setup using *detectron2*,[8] a Mask-RCNN API using PyTorch.

6.3.2 Experiment setup, model testing, and results

Mask-RCNN performs both superclass prediction and mask extraction (see Fig. 2). To measure the performance of this model, we used four metrics: precision, recall, and F1-score for superclass prediction and intersection over union (*IoU*) for mask extraction. Table 3 presents the results per superclass.

We computed *IoU* using the predicted (pred) and the ground truth (gt) masks. It is the quotient of their intersection and union:

$$IoU = \frac{N^{(1)}(M^{(\text{gt})} \cap M^{(\text{pred})})}{N^{(1)}(M^{(\text{gt})} \cup M^{(\text{pred})})}, \quad \forall M \in \{0,1\}^{m \times n} \tag{12}$$

where $N^{(1)}$ is a function that takes a binary array of a dimension and returns the number of 1 elements. The intersection ($\cap$) is an element-wise *AND* operation between two masks ($1 \times 1 = 1$, $1 \times 0 = 0$ and $0 \times 0 = 0$). The union ($\cup$) is an element-wise *OR* operation ($1 \times 1 = 1, 1 \times 0 = 1$ and $0 \times 0 = 0$). We assessed the performance of mask prediction using the average of all the *IoU*s computed for the testing set (see Table 3).

Each result shown in Table 3 is the average of the four values obtained by fourfold cross-validation. In each fold, we used 75% of the data for training and 25% for testing (randomly split).

Answer to RQ2: Although not ideal, the performance of Mask-RCNN is good, and Mask-RCNN has reliable results on superclass prediction and mask extraction. The F1-score for overall superclass prediction is 67%, which can be improved. As we can see in Fig. 11, the errors of Mask-RCNN are focused on certain matrix elements. Table 4 presents the main confusions. The model has problems to differentiate knitwear, shirts, and outerwear (low F1-score). For future work, we plan to increase the number of labeled images in our data set and fine-tune the training setup to improve the F1-score. The *IoU* score for mask prediction is 0.81, which is a good result due to the complexity of the garments' boundaries. The top is the only superclass with an IoU score under 80% because most models in the pictures are women with long hair. The hair is the source of noisy pixels propagating through the rest of the models. Therefore, we plan to implement a hair segmentation model in the image processing pipeline.

6.4 RQ3: Is the subclass prediction model effective?

6.4.1 Model training

We started with pre-trained weights on the ImageNet data set. We performed model training for subclass prediction in four stages. Figure 12 shows a scheme of the stages; Table 5 gives the stage parameters. The loss function used in model training is the Cross-Entropy, i.e., $L = -\sum_{i=1}^{N} b_i \log(p_i)$, where b_i is a binary label (1 if the truth class is class i) and p_i is the *softmax* probability. We decreased the learning rate and included more layers in each stage to slow down the weight update as we approached the optimal solution. We followed, in general terms, the principles of the transfer learning scheme [46]. When the test loss did not reach an absolute minimum during three consecutive epochs, we concluded the stage and started the following one to prevent over-fitting. In other words, each training stage concluded when: (i) the test loss did not reach an absolute minimum in three consecutive epochs or (ii) we reached the number of epochs stated in Table 5. The training data set was the garment images which Mask-RCNN output after skin segmentation ($I^{(\text{final})}$ in Eq. (5)).

We performed data augmentation on images in the training data set due to the complexity of subclass

[7] In addition to mask and category, Mask-RCNN returns four vertices (called the bounding box) that enclose the mask in a rectangle. We do not use it in our pipeline, but it is used internally by Mask-RCNN and it requires a term in the loss function.

[8] https://github.com/facebookresearch/detectron2.

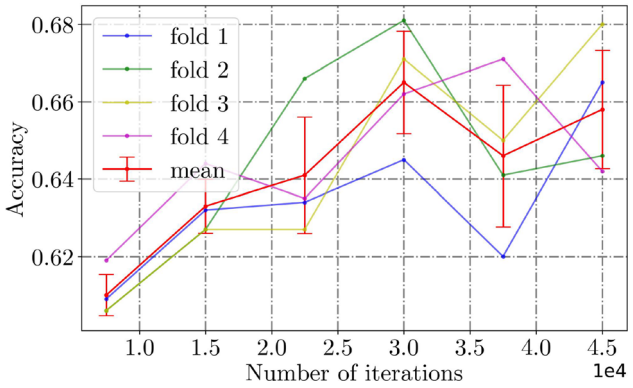

Fig. 10 Empirical determination of the optimal number of iterations to train Mask-RCNN

Table 3 Mask-RCNN results on superclass prediction and mask extraction

	Precision	Recall	F1-score	IoU-score
Co_ord	0.60	0.69	0.64	0.83
Suits	0.83	0.90	0.86	0.86
Outerwear	0.40	0.66	0.50	0.83
Jumpsuit	0.64	0.68	0.65	0.82
Dress	0.79	0.69	0.73	0.80
Shirt	0.75	0.41	0.53	0.87
Knitwear	0.70	0.43	0.51	0.85
Bottom	0.80	0.86	0.83	0.85
Top	0.70	0.63	0.66	0.76
Total	0.69	0.66	0.67	0.81

We show precision, recall and F1-score for the superclass prediction problem and IoU score for the mask extraction problem

prediction. We did not change the nature of the images we input into the models. For instance, we did not flip the image vertically because the garments would look upside down (something that does not happen in reality). Figure 13 presents some data augmentation examples on images. We included two more versions of the training images through the horizontal flip and random rotations not higher than 20°. These transformations were introduced to our data loader and applied in each training iteration. We used the horizontal flip with a probability of 0.5. Random rotation could take any value in the range of ±20° for each iteration.

6.4.2 Experiment Setup, Model Testing, and Results

As mentioned before, the garment taxonomy prediction is a 66-categories classification problem. Section 4 presents how we split the problem into two steps: (i) superclass prediction using Mask-RCNN and (ii) subclass prediction using a different ResNet-18 model for each superclass. The alternative is to predict the subclasses directly using Mask-RCNN in a single step. We tested both approaches. Splitting the problem into two steps had a 5% better F1 score for subclass prediction than the single-step approach (see Table 6). In addition, the main reason to split this problem into superclass and subclass prediction is to ensure at least one confident feature related to the garment taxonomy. Superclass prediction has an F1-score of 67%; subclass prediction has an F1-score of 45% after propagating the errors. We implemented the two steps subclass prediction in the final version of our pipeline.

Subclass prediction was performed once we knew the superclass result. We trained a different ResNet-18 model for each superclass outcome. We performed the fourfold cross-validation to test the accuracy of each ResNet-18 model separately (i.e., in each fold, we used 75% of the data for training and 25% for testing). We measured the performance of each ResNet model in the same way we did to evaluate Mask-RCNN for superclass prediction, i.e., top-1 accuracy. Table 6 gives the average F1-score for each superclass' model. The classification errors propagate through the pipeline. Therefore, the results in Table 6 for subclass prediction are for an ideal scenario in the superclass prediction. The model has a better subclass prediction accuracy for superclasses having fewer subclasses to differentiate (N). The accuracy decreases for superclasses having visually similar subclasses (e.g., ballerina and a-line dresses, formal and sport shirts, or henley and t-shirt tops). Figure 14 gives some images for different subclasses that are hard to distinguish.

The F1-score for subclass prediction, knowing the superclass beforehand, is 66%. This number is reduced up to 45% due to the errors committed by Mask-RCNN on superclass prediction (F1-score = 67% Table 3).

Answer to RQ3: As seen in Table 6, all the ResNet-18 models have good performance (considering the number of subclasses for each superclass and the similarity between the subclasses). The only models with accuracy under 50% are the ResNets for the superclasses *dress* and *top* due to the number of subclasses (19 and 15 subclasses, respectively). Even though all the ResNet-18 models perform well, the errors propagated from Mask-RCNN make the subclass prediction accuracy 45%. To increase the accuracy, we should enhance the accuracy of Mask-RCNN in superclass prediction. As we see in Table 6, the best approach is to use Mask-RCNN for superclass prediction and then ResNet-18 for subclass prediction (instead of using Mask-RCNN in a single step). Even with the propagation of errors, the accuracy is 6% higher in the first case.

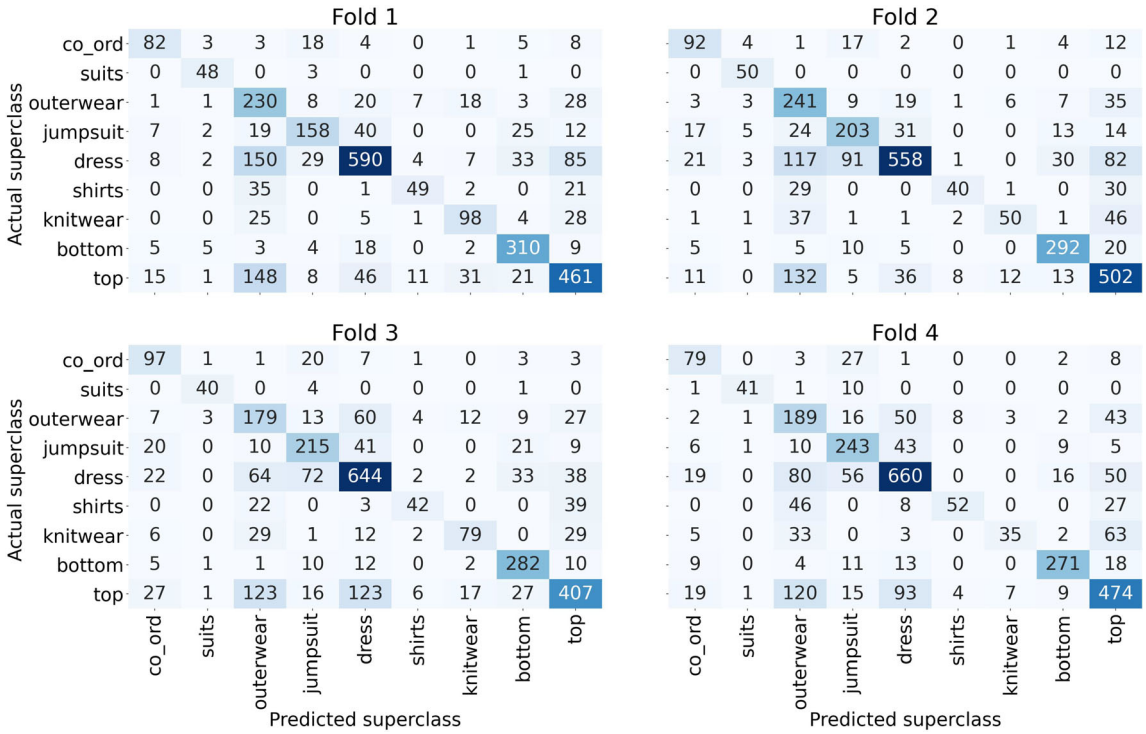

Fig. 11 Full results of Mask-RCNN on superclass prediction presented as confusion matrix of each of the four folds in the cross validation

Table 4 Most common confusions of Mask-RCNN in superclass prediction

Actual superclass	Wrong prediction	Frequency
Shirts	Outerwear	0.29
Knitwear	Top	0.28
Knitwear	Outerwear	0.21
Shirts	Top	0.21
Top	Outerwear	0.18

We show the actual superclass along with what the model has predicted. The Frequency is the percentage of elements misclassified. E.g. in the first row represents the percentage of shirts misclassified as outerwear

6.5 RQ4: Is the price prediction model effective?

6.5.1 Model training

As explained in Sect. 5, the price prediction model is formed by a Gaussian Mixture Model (GMM) and a Random Forest (RF). The training of the GMM consisted of getting the mean price of each of the Gaussian Slots reachable by Algorithm 1. As for the RF, we have set it to hold ten trees and seven nodes as a maximum per tree.

6.5.2 Experiment setup, model testing, and results

We have split the data set into 80% samples for training and 20% for testing. The price prediction model has been evaluated using the Mean Absolute Error with the standard deviation of the price data set of the testing samples. We propose a metric (we call r) defined by Eq. (13).

$$r = \frac{\sigma_{\text{test}} - MAE_{\text{test}}}{\sigma_{\text{test}}} \tag{13}$$

$r = 1$ in the ideal scenario where $MAE = 0$. If $MAE \sim \sigma$, our predictor is equivalent to a model retrieving the mean of the data set as a prediction. In this case, we consider that the model provides a prediction in the range of the real prices of the data set but does not learn the variance of the target with respect to the given input variables ($r = 0$). If $MAE < \sigma$, the model learns from the training process ($0 < r < 1$). The closer MAE gets to 1, the more the model learns. For $r < 0$ (if $MAE > \sigma$), the model performs worse than a random guesser ($r = 0$), which indicates low accuracy.

We propose r as a performance evaluator instead of MAE because it provides a normalized score whose critical values (0 and 1) are more interpretable by users. In addition, using r, we can compare the performance of different models tested on different target distributions. As mentioned earlier, we have trained one different model per

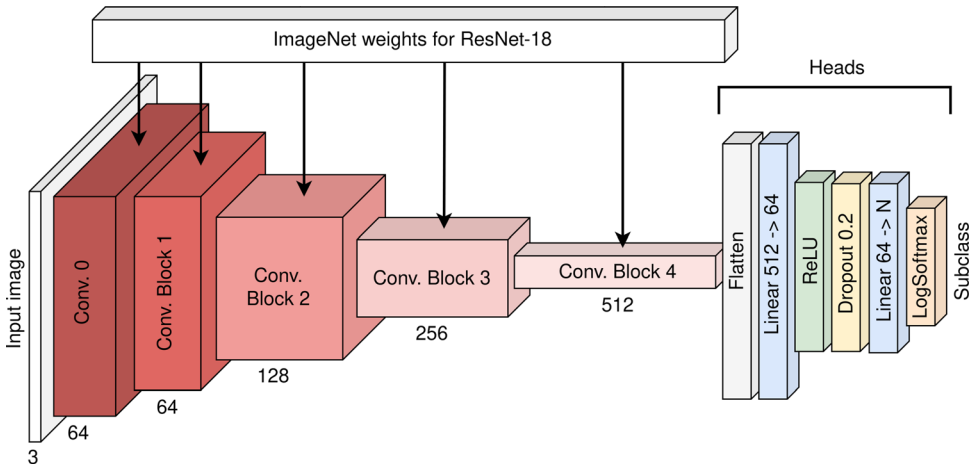

Fig. 12 ResNet-18 architecture and training stages. We build Head layers on top of the ResNet-18 architecture obtained from [5]. ImageNet weights are loaded on each ResNet-18 block. Then we perform fine tuning starting with the layers closer to the classification output

Table 5 Training stages of ResNet-18

Stage	Weights (see Fig. 8)	Epochs	L.R
1	Heads	50	10^{-3}
2	Heads + Block 4	30	2×10^{-4}
3	Heads + Blocks 4, 3	15	2×10^{-4}
4	Heads + Blocks 4, 3, 2	10	1×10^{-4}
5	Heads + Blocks 4, 3, 2, 1	10	1×10^{-4}

Each stage involves more layers than the stage before. Also, we decrease the learning rate and the number of epochs we train. Layers closer to the Heads are trained more epochs than layers closer to the input, since we have loaded the weights from ImageNet data set

superclass. Assume the following two models: (i) the model estimating the price of *tops* with $MAE = 4$ EUR and (ii) the model estimating the price of *suits* with $MAE = 40$ EUR. Although one could claim that the first model is ten times more accurate, predicting a suit's price is more complex since the target price distribution presents significantly more variance.

Answer to RQ4: As seen in Table 7, for all the superclasses, the prediction model has an r score higher than 0.5 (mostly above 0.6). The average r score is 0.67, which we consider satisfactory due to the problem's difficulty. Since we do not consider the market trends, it is hard to predict a garment's price using only static information while ignoring temporal evolution.

6.6 Threats to the validity

This section addresses potential limitations and challenges that may affect the validity of our study's findings.

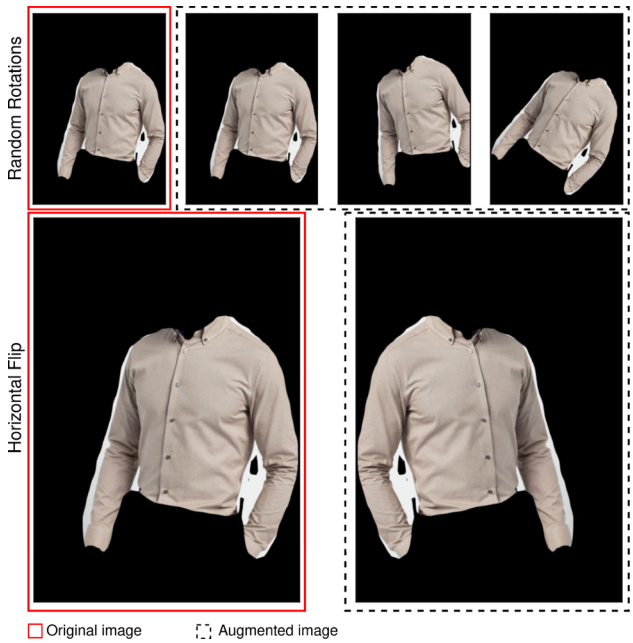

Fig. 13 Data augmentation illustration. We perform both random rotations in the range of $\pm 20°$ and horizontal flip of each image in the data set. That way we got two augmented images per each original image

6.6.1 Threats to the external validity

Model obsolescence: the selection of models for our AI pipeline was guided by a strategic balance between cutting-edge technology and practical applicability within the dynamic field of computer vision. At the time of implementation, our chosen models represented the state-of-the-art, offering the best available balance of accuracy, computational efficiency, and adaptability to the nuanced task of garment price projection. This decision was underpinned by the models' proven performance in similar image recognition tasks, their robustness in handling diverse fashion data sets, and their flexibility in integrating with

Table 6 ResNet-18 accuracy results on subclass prediction

Superclass	*N* subclasses	F1-score
Co_ord	3	0.72
Suits	1	–
Outerwear	8	0.54
Jumpsuit	7	0.63
Dress	19	0.35
Shirts	3	0.79
Knitwear	2	0.80
Bottom	8	0.54
Top	15	0.45
ResNet-18 layer (isolated)	66	0.66
Mask-RCNN + ResNet-18	66	0.45
Single-step (Mask-RCNN)	66	0.40

Last three rows present results in the total data set: (i) ResNet-18 Layer (isolated) presents the accuracy of the ResNet-18 layer isolated in an ideal scenario, (ii) Mask-RCNN + ResNet-18 presents the accuracy of the our pipeline (concatenation of Mask-RCNN and ResNet-18 layer) in subclass prediction, taking into account the propagation of errors from Mask-RCNN, and (iii) Single-step (Mask-RCNN) presents the accuracy if we use the approach in which Mask-RCNN predicts directly the subclass, without prior superclass prediction

Table 7 Evaluation results of the price prediction model of each superclass

Superclass	σ_{test} (EUR)	MAE_{test} (EUR)	r
Co_ord	28.66	8.09	0.72
Suits	376.23	60.34	0.83
Outerwear	53.09	14.89	0.59
Jumpsuit	13.30	5.66	0.57
Dress	20.75	6.77	0.67
Shirts	54.33	19.29	0.64
Knitwear	10.26	4.44	0.56
Bottom	92.23	15.16	0.84
Top	8.02	3.19	0.60
Mean	72.99	15.31	0.67

We provide both the standard deviation and the r score we have defined

temporal market trend analysis. Given the rapid advancement in computer vision technologies, the models we selected as state-of-the-art at the time of our study's implementation may now be surpassed by newer models. This fast-paced evolution could impact the generalizability and future applicability of our results.

Market variability: our approach assumes a degree of consistency in market behavior that may not hold in all cases. Unforeseen economic events, shifts in consumer behavior, or changes in the fashion industry's structure could alter the effectiveness of our model.

Global variability: our research is potentially limited by its focus on specific markets or regions. The global fashion industry encompasses a wide range of consumer behaviors, economic conditions, and regulatory environments. These differences can affect the generalizability of our results across different geographical contexts.

Comparison with other approaches: we encountered significant challenges in implementing and comparing our method with certain advanced methods directly due to a variety of factors. Firstly, the availability of implementations for some of the latest methods is limited, with many not being publicly accessible or requiring proprietary data sets for replication. Secondly, the documentation and reproducibility of these methods vary greatly, making it difficult to ensure a fair and accurate comparison. Finally, the resource-intensive nature of training and testing multiple sophisticated models exceeds our current capabilities, particularly in terms of computational resources and access to equivalent data sets.

6.6.2 Threats to the internal validity

We critically examined the inherent challenges within our evaluation of FashionXpert. We identified potential biases in model selection, acknowledging that while our models were state-of-the-art at the time of implementation, the

Fig. 14 Example images for different subclasses hard to distinguish. We show four cases where it is hard even for a human to classify the subclasses

fast-paced advancements in AI could soon render them less effective. To mitigate these threats, we employed robust validation methods, including cross-validation and sensitivity analysis, ensuring our model's performance is reliably assessed against various data scenarios. We also highlighted efforts to combat overfitting through careful data set partitioning and model regularization techniques. Furthermore, we detailed our approach to selecting evaluation metrics that accurately reflect the model's predictive capabilities, ensuring a comprehensive assessment of its performance. Through these strategies, we aimed to bolster the internal validity of our findings, maintaining transparency and adaptability in our research approach.

7 Conclusion

In this paper, we presented FashionXpert, which performs price prediction for the fashion industry. FashionXpert provides an AI pipeline that employs several image processing and ML techniques with data from heterogeneous data sources. Our contributions include (i) an AI pipeline using computer vision and ML techniques for price prediction, (ii) a data set to assess computer vision and ML techniques for the fashion industry, and (iii) an empirical assessment of these techniques in our pipeline using our data set.

Our empirical results showed that FashionXpert performed well while predicting superclasses (with a total F1 score of 67%) and subclasses (45% on average). Moreover, the price prediction model has been evaluated with the metric r-score [Eq. (13)] in all the superclasses showing an average value of 0.67 and an average MAE of 15.31 EUR. Considering the standard deviation of the target price data set (72.99 EUR on average among superclasses) and given the complexity of the problem, we can conclude that this result is very satisfactory.

For future work, we aim to add a trend prediction feature (i.e., predicting fashion trends, including colors, fabrics, silhouettes, patterns, styles, and more for clothing collections in upcoming seasons) in our pipeline. Trend predictions allow the fashion industry to know what will be fashionable in the future and plan the clothing collections and merchandising appropriately. On the other hand, emerging trends affect garment pricing for upcoming seasons. Therefore, we also plan to improve our price prediction model with trend predictions.

Acknowledgements This work was supported by the TrendXpert innovation project funded by Luxembourg Ministry of Economy and the Scientific and Technological Research Council of Turkey (under Project #119N537) under the IraSME framework (September 2019 call). The research of Joe Lorentz for this publication is supported by the National Research Fund Luxembourg (Grant 14297122). We would like to thank to the following people from Galaksiya for their contributions to the TrendXpert project which led to this research paper: Erdem Eser Ekinci, Rabia Hatapoğlu, Uğur Üntürk, Gizem Abalı, İremnur Kulaksız, Kaan Sağlam, Berkay Akdal, Pınar Göçebe Bayhan, and Ozan Türker.

Funding Open access funding provided by SINTEF.

Data availability The data sets generated during and/or analyzed during the current study are not publicly available due to confidentiality reasons but are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors declare that there is no Conflict of interest.

References

1. follow.fashion. https://follow.fashion/
2. Edited. https://edited.com/
3. Liu Z, Luo P, Qiu S, Wang X, Tang X (2016) Deepfashion: powering robust clothes recognition and retrieval with rich annotations. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1096–1104
4. He K, Gkioxari G, Dollár P, Girshick R (2017) Mask-RCNN. arXiv preprint arXiv:1703.06870v3
5. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
6. Ronneberger O, Fischer P, Brox T (2015) U-Net: convolutional networks for biomedical image segmentation. arXiv preprint arXiv:1505.04597v1
7. hen K, Chen K, Cong P, Hsu WH, Luo J (2015) Who are the devils wearing prada in new york city? In: Proceedings of the 23rd ACM international conference on multimedia
8. Yamaguchi K, Kiapour MH, Ortiz LE, Berg TL (2012) Parsing clothing in fashion photographs. In: 2012 IEEE conference on computer vision and pattern recognition, pp 3570–3577. https://doi.org/10.1109/CVPR.2012.6248101
9. Cheng WH, Song S, Chen CY, Hidayati SC (2021) Fashion meets computer vision: a survey. ACM Comput Surv 54(4):72. https://doi.org/10.1145/3447239
10. Borràs A, Tous F, Lladós J, Vanrell M (2003) High-level clothes description based on colour-texture and structural features. In: Perales FJ, Campilho AJC, de la Blanca NP, Sanfeliu A (eds) Pattern recognition and image analysis. IbPRIA 2003. Lecture

notes in computer science, vol 2652. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-44871-6_13
11. Liang X et al (2015) Deep human parsing with active template regression. IEEE Trans Pattern Anal Mach Intell 37(12):2402–2414. https://doi.org/10.1109/TPAMI.2015.2408360
12. Kiapour MH, Han X, Lazebnik S, Berg AC, Berg TL (2015) Where to buy it: matching street clothing photos in online shops. In: 2015 IEEE international conference on computer vision (ICCV), pp 3343–3351. https://doi.org/10.1109/ICCV.2015.382
13. Jin B, Lu B, Wu H, Shi W, Li Y (2021) Fashion style forecasts based on different price ranges. In: 2021 IEEE 5th advanced information technology, electronic and automation control conference (IAEAC), vol 5. IEEE, pp 2296–2302
14. Al-Halah Z, Stiefelhagen R, Grauman K (2017) Fashion forward: forecasting visual style in fashion. In: Proceedings of the IEEE international conference on computer vision, pp 388–397
15. Krizhevsky A, Sutskever I, Hinton GE (2012) Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, pp 1097–1105
16. Jia M et al (2020) Fashionpedia: ontology, segmentation, and an attribute localization dataset. In: Vedaldi A, Bischof H, Brox T, Frahm JM (eds) Computer vision—ECCV 2020. ECCV 2020. Lecture notes in computer science, vol 12346. Springer, Cham
17. Sreekumar A, Geetha M (2020) Hand segmentation in complex background using UNet. In: 2020 2nd international conference on inventive research in computing applications (ICIRCA), pp 440–445. https://doi.org/10.1109/ICIRCA48905.2020.9183215
18. Xie H -X, Lin C -Y, Zheng H, Lin P -Y (2018) An UNet-based head shoulder segmentation network. In: 2018 IEEE international conference on consumer electronics-Taiwan (ICCE-TW), pp 1–2. https://doi.org/10.1109/ICCE-China.2018.8448587
19. Pavan Kumar I, Hara Gopal VP, Ramasubbareddy S, Nalluri S, Govinda K (2020) Dominant color palette extraction by K-means clustering algorithm and reconstruction of image. In: Raju K, Senkerik R, Lanka S, Rajagopal V (eds) Data engineering and communication technology. Advances in intelligent systems and computing, vol 1079. Springer, Singapore
20. Jin X, Han J (2011) K-means clustering. In: Sammut C, Webb GI (eds) Encyclopedia of machine learning. Springer, Boston. https://doi.org/10.1007/978-0-387-30164-8-425
21. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S, Uszkoreit J (2021) An image is worth 16×16 words: transformers for image recognition at scale. Computer Science, Computer Vision and Pattern Recognition.arXiv:2010.11929
22. He K, Chen X, Xie S, Li Y, Dollár P, Girshick R (2021) Masked autoencoders are scalable vision learners. Computer Science, Computer Vision and Pattern Recognition, arXiv
23. Kirillov A, Mintun E, Ravi N, Mao H, Rolland C, Gustafson L, Xiao T, Whitehead S, Berg AC, Lo WY, Dollár P (2023) Segment anything. Computer Science, Computer Vision and Pattern Recognition. arXiv:2304.02643
24. Li Z, Nie Y, Han K, Guo J, Xie L, Wang Y (2022) A transformer-based object detector with coarse-fine crossing representations. In: 2022, 36th conference on neural information processing systems (NeurIPS 2022)
25. He L, Todorovic S (2022) DESTR: object detection With split transformer. In: 2022, proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 9377–9386
26. Kedia S, Jain S, Sharma A (2020) Price optimization in fashion e-commerce. arXiv preprint arxiv:2007.05216
27. Nguyen HD, Tran KP, Thomassey S, Hamad M (2021) Forecasting and anomaly detection approaches using LSTM and LSTM autoencoder techniques with the applications in supply chain management. Int J Inf Manag 57:102282. https://doi.org/10.1016/j.ijinfomgt.2020.102282
28. He QQ, Wu C, Si YW (2022) LSTM with particle swam optimization for sales forecasting. Electron Commer Res Appl 51:101118. https://doi.org/10.1016/j.elerap.2022.101118
29. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
30. Ferreira Fernando GDC, Gandomi Amir H, Cardoso Rodrigo TN (2021) Artificial intelligence applied to stock market trading: a review. IEEE Access 9:30898–30917
31. Warner B, Crook A, Cao R (2020) Predicting the DJIA with news headlines and historic data using hybrid genetic algorithm/support vector regression and BERT. In: Big data–BigData 2020: 9th international conference, held as part of the services conference federation, SCF 2020, Honolulu, HI, USA, September 18–20, 2020, Proceedings 9. Springer, pp 23–37
32. Mohanty DK, Parida AK, Khuntia SS (2021) Financial market prediction under deep learning framework using auto encoder and kernel extreme learning machine. Appl Soft Comput 99:106898
33. Mokhtari S, Yen KK, Liu J (2021) Effectiveness of artificial intelligence in stock market prediction based on machine learning. arXiv preprint arXiv:2107.01031
34. Yaşar Çıklaçandır FG, Utku S, Özdemir H (2023) Determination of various fabric defects using different machine learning techniques. J Text Inst 115(5):733–743
35. Zhong S, Ribul M, Cho Y, Obrist M (2023) TextileNet: a material taxonomy-based fashion textile dataset. arXiv preprint arXiv:2301.06160
36. Xing W, Liu Y, Xin B, Zang L, Deng N (2022) The application of deep and transfer learning for identifying cashmere and wool fibers. J Nat Fibers 19(1):88–104
37. Zhu Y, Liu R, Hu G, Chen X, Li W (2023) Accurate identification of cashmere and wool fibers based on enhanced ShuffleNetV2 and transfer learning. J Big Data 10(1):152
38. Kahraman Y, Durmuşoğlu A (2023) Deep learning-based fabric defect detection: a review. Text Res J 93(5–6):1485–1503
39. Plot entropy in Scikimage post. https://scikit-image.org/docs/dev/auto-examples/filters/plot-entropy.html
40. Zou X, Kong X, Wong W, Wang C, Liu Y, Cao Y (2019) FashionAI: a hierarchical dataset for fashion understanding. In: 2019 IEEE/CVF conference on computer vision and pattern recognition workshops (CVPRW), pp 296–304. https://doi.org/10.1109/CVPRW.2019.00039
41. Ge Y, Zhang R, Wang X, Tang X, Luo P (2019) DeepFashion2: a versatile benchmark for detection, pose estimation. In: Segmentation and re-identification of clothing images, CVPR
42. Guo S, Huang W, Zhang X, Srikhanta P, Cui Y, Li Y, Adam H, Scott MR, Belongie S (2019) The iMaterialist fashion attribute dataset. In: Proceedings of the IEEE/CVF international conference on computer vision workshops
43. Shapiro SS, Wilk MB (1965) An analysis of variance test for normality (complete samples). Biometrika 52(3/4):591–611. https://doi.org/10.2307/2333709
44. Nachar N (2008) The Mann–Whitney U: a test for assessing whether two independent samples come from the same distribution. Tutor Quant Methods Psychol. https://doi.org/10.20982/tqmp.04.1.p013
45. Lin TY, et al (2014) Microsoft coco: common objects in context. In: Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13. Springer International Publishing, pp 740–755
46. Pan SJ, Yang Q (2009) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359