# Team ID: PNT2022TMID20463

**Project Name:** SMART FASHION RECOMMENDER APPLICATION

**Team Leader:** Mohammed Azeer

**Team Member 1:** Menati Vasudeva Reddy **Team Member 2:** Loghadharshann D **Team Member 3:** Rajavignesh M

Abstract

Chatbots can bring innovation in online assistance and communication with customers. Due to the growth of e-commerce, fashion brands have been adopting chatbots to provide personalised consumer experiences. Research in the area of chatbots for fashion e-commerce has addressed technological advancements and consumer behaviour, but little has been done on analysing chatbot features through a holistic point of view. The aim of this paper is to offer an interdisciplinary review through a comprehensive categorisation of recent studies on the theme and inform future research in the area. To achieve that, a theme-based literature review was carried out through the analysis of specialised research. The collected work was categorised addressing both computational and non-computational perspectives. The ﬁndings show that Deep Learning, recommendation systems, audio recognition and integration of chatbots with other fashion applications are a few design opportunities to be applied in both research and practice.

al Library, Emerald, Google Scho-

lar, IEEE Xplore Digital Library, Mendeley (Elsevier),

Microsoft Academic, SagePub, ScienceDirect, Scopus,

Springer, Taylor and Francis Online, IBM’s TechDocs,

and Wiley Online Library. This covers 10 years of

research in this ﬁeld (preliminary searches on some of

the databases revealed no study on the theme prior to

2014 –see Figure 6). To avoid limitations, we adopted

a second search string that is not restricted to ‘e-com-

merce’AND ‘fashion’only. Variations for ‘fashion’

were also included such as apparel, garment, clothing,

cosmetics, and makeup. As for ‘chatbots’

The rapid progress of computer vision, machine learning, and artificial intelligence combined with the current growing urge for online shopping systems opened an excellent opportunity for the fashion industry. As a result, many studies worldwide are dedicated to modern fashion-related applications such as virtual try-on and fashion synthesis. However, the accelerated evolution speed of the field makes it hard to track these many research branches in a structured framework. Chatbots can bring innovation in online assistance and communication with customers. Due to the

growth of e-commerce, fashion brands have been adopting chatbots to provide personalised

consumer experiences. Research in the area of chatbots for fashion e-commerce has addressed

technological advancements and consumer behaviour, but little has been done on analysing

chatbot features through a holistic point of view. The aim of this paper is to oﬀer an

interdisciplinary review through a comprehensive categorisation of recent studies on the theme

and inform future research in the area. To achieve that, a theme-based literature review was

carried out through the analysis of specialised research. The collected work was categorised

addressing both computational and non-computational perspectives. The ﬁndings show that

Deep Learning, recommendation systems, audio recognition and integration of chatbots with

other fashion applications are a few design opportunities to be applied in both research and

pract

Chatbots can bring innovation in online assistance and communication with customers. Due to the

growth of e-commerce, fashion brands have been adopting chatbots to provide personalised

consumer experiences. Research in the area of chatbots for fashion e-commerce has addressed

technological advancements and consumer behaviour, but little has been done on analysing

chatbot features through a holistic point of view. The aim of this paper is to oﬀer an

interdisciplinary review through a comprehensive categorisation of recent studies on the theme

and inform future research in the area. To achieve that, a theme-based literature review was

carried out through the analysis of specialised research. The collected work was categorised

addressing both computational and non-computational perspectives. The ﬁndings show that

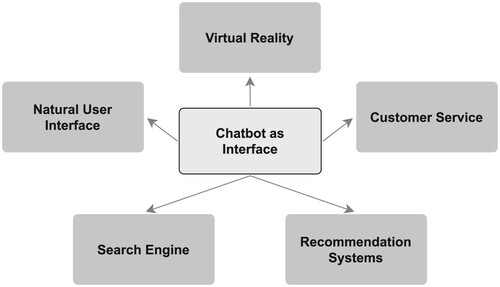
Deep Learning, recommendation systems, audio recognition and integration of chatbots with

other fashion applications are a few design opportunities to be applied in both rese

# 1.Introduction

Boosted by the COVID-19 pandemic, the fashion industry has been forced to speed up its digital transformation, bringing traditional offline services online. In the UK, 77% of retail businesses are employing AI in e-commerce through the use of chatbots.

Studies involving AI and fashion as a discipline have looked at factors that determine consumer acceptance forecasting and production management, trend analysis of colour palettes), accuracy of image labelling through machine learning (ML) Malware detection in self-driving vehicles using machine learning algorithms . Chatbots can provide personalised shopping experiences across physical and online channels and promote consumer wellbeing . However, the main challenge remains on the complexity of human language and the chatbot’s effectiveness within this context Chatbots may be viewed as Natural User Interfaces for users to interact with a computer more intuitively through natural language, easing acceptance of e-commerce, especially for fashion items. Thus, chatbots may be seen as an essential, relevant interface element for many fashion e-commerce tasks such as providing recommendations, exploring and searching huge catalogues, complementing virtual fitting room’s features, and delivering (post-sale) customer services.



When it comes to designing and adopting chatbots within the consumer journey, Pantano and Pizzi inform that: retailers might want to choose already designed chatbot platforms or to design their own; from a technological point of view, conversational agents are designed to mimic natural language; however, analytical skills to learn from consumer data are still required; and, social media (e.g. instant messengers such as Facebook Messenger) and voice-based AI (utilising audio recognition) such as Siri and Alexa, are the most used platforms. From a computational perspective, there is a trend to improve the responsiveness of the chatbot and patents were mostly designed to provide new algorithms related to Natural Language Processing (NLP), digital communication and consumer management.

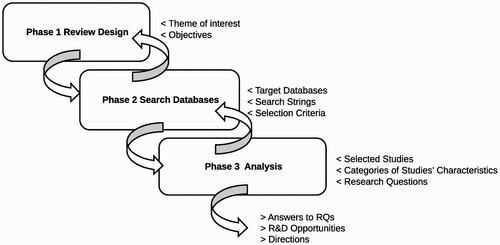
Since retailers are willing to opt for designing their own chatbot, it is crucial that managers and designers understand the current technological barriers and opportunities. Given this context, the aim of this paper is to provide design and research opportunities for future fashion chatbots applications by answering the following research questions:

* RQ1: Which are the current proposed chatbots design approaches for e-commerce in general, and particularly for fashion applications?
* RQ2: How could research on e-commerce chatbots be categorised in an integrated manner?
* RQ3: What are the research opportunities of chatbots design to deal with the specificities of fashion e-commerce applications?

The main contributions of this study are: (i) the proposal of an interdisciplinary categorisation with an integrated view of the theme of ‘chatbots in fashion e-commerce’ that can serve as support for guiding professionals and researchers, from academy to industry, interested in understanding such kind of agents; (ii) the provision of a synthesised basis for future reference and research on the theme.

## 2. Methodology and research design

The theme-based literature review was carried out according to a 3-phase methodology we interactively adjusted from that by Okoli ([2015](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) for simplicity and fidelity to actual procedures for the task at hand. Phases are not necessarily sequential: one may return to a previous phase after (partial) execution of a next phase – e.g. the review results provided a base to categorise chatbot studies and to enhance details of the review’s design and execution and associated analysis (see [Figure 2](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0002))



### 2.1. Search and selection strategies

*(Chatbot OR Dialog System OR Conversational Agent OR Virtual Assistant OR Digital Assistant)*

*AND*

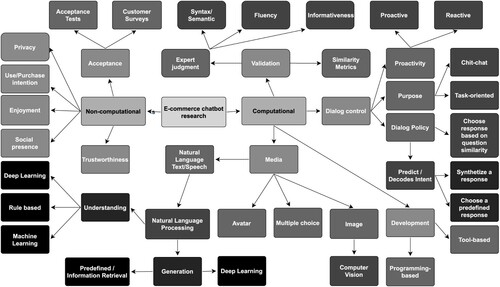
*((E-comm\* OR Ecomm\* OR Retail) OR (Fashion OR Garment OR Apparel OR Clothing OR Cosmetics OR Makeup OR Make-up))*

was conducted to collect papers published from 2011 to 2021 on the theme of interest from the following databases: ACM Digital Library, Emerald, Google Scholar, IEEE Xplore Digital Library, Mendeley (Elsevier), Microsoft Academic, SagePub, ScienceDirect, Scopus, Springer, Taylor and Francis Online, IBM’s TechDocs, and Wiley Online Library. This covers 10 years of research in this field (preliminary searches on some of the databases revealed no study on the theme prior to 2014 – see [Figure 6](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0006)). To avoid limitations, we adopted a second search string that is not restricted to ‘e-commerce’ AND ‘fashion’ only. Variations for ‘fashion’ were also included such as apparel, garment, clothing, cosmetics, and makeup. As for ‘chatbots’, we utilised dialog system, conversational agent, virtual assistant and digital assistant as variations.

A study was excluded if it was: i) written in a language other than English; ii) not available online for download; and iii) addressing the theme of chatbots but not for digital retail purposes. Three types of work were included: i) journal papers, ii) conference papers, and iii) book chapters.

### 2.2. Categorisation

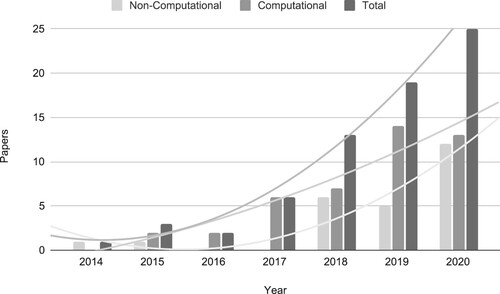
A first distinction of chatbots’ studies is along the lines of computational aspects (i.e. aspects related to the area of Computer Science or Information Technologies, such as the use of NLP) and non-computational aspects (i.e. all other aspects such as studying consumer acceptance). This first distinction appears in Jurafsky and Martin ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) and Diederich, Brendel, and Kolbe ([2019](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)). We expand on these earlier categorisations as illustrated in [Figure 3](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0003).



Chatbots as conversational recommender systems take a more sophisticated approach, offering a richer set of interactions that help improve preference elicitation and interact with users through natural language (NL). As such, chatbots can provide mechanisms to capture contextual information, as what has been intended by the so-called context-aware recommender systems. In the case of chatbots in fashion e-commerce, other features related to online sales could be required - e,g,, those that engage and persuade customers to buy a product, eventually involving negotiation dialogues, playing the role of a salesperson conversing with a customer (Jusoh, [2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)). This kind of system deals with the main challenges of natural-language-based interaction, such as NL understanding, information extraction, and NL generation, which can be specific for the fashion domain.

## 3. Results and discussion

After applying the proposed methodology, 76 out of 5959 papers were selected: 46 describing research on chatbot computational aspects; and 30 being surveys about the state of the art or the user experience using chatbots. Firstly, it is worth noting the growth of research on the theme of chatbots for (fashion) e-commerce since 2014 (see [Figure 4](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0004)). Such increasing interest may be explained by the advent of DL and the subsequent interest to comprehend non-computational aspects of chatbots.



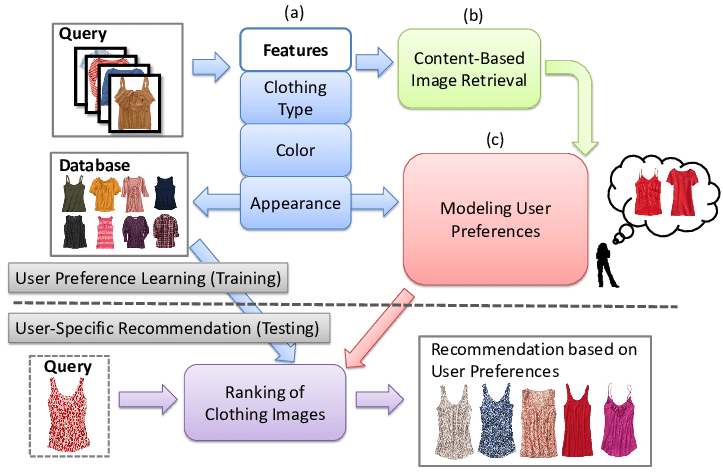
## Design of visual recommendation module

The fashion domain is a very popular playground of machine learning and computer vision. The main problem of this domain is produced by the high level of subjectivity and the semantic complexity of the features involved. Recent work has focused on a variety of approaches including attribute recognition, clothing retrieval, image generation and visual recommendation..

This section focuses on evaluating our system and deciding the stage to which it is able to fulfill the purpose for which it was created the performance of the system is analyzed in detail through several tests, from small scale to large scale. Firstly, the unit tests are done at the lower stages and then we proceed to the whole test system. Several machine purposes are also involved in the system.

**Visual recommendation module implementation :**

If the suggestion was cultivated for shopping, the storehouse would have contained pictures from online retail locations like Amazon, eBay, Pinterest, Instagram, etc. A subset of pattern datasets was used to test our proposed approach. At that point, the information had already been cleared of unimportant photos. Then, the photos were passed by means of the organizationand design vector pictures have been created from each photo. For the getting the suggestion, we first needed to build the individual style profile. This is brought out by taking one or more noteworthy pictures of the client's ideal attire things as they were entered and by making their style vector. These vectors are then blended to shape the framework of the individual style profile. The Figure 5 shown Pattern recommendation with similarity score.



**Recommender service**

**from flask import \***

**from flask import \***

**from connect import \***

**import datetime**

**from urllib.parse import urlparse**

**from flask\_mail import Mail,Message**

**import random**

**app= Flask(\_name\_)**

**mail=Mail(app)**

**app.config['SECRET\_KEY'] = ' \*\*\*\*\*\*\*\*\*\*\*\*'**

**app.config['MAIL\_SERVER']="smtp.gmail.com"**

**app.config['MAIL\_PORT']=587**

**app.config['MAIL\_USERNAME']='\*mail@gmail.com\*'**

**app.config['MAIL\_PASSWORD']=" \*\* User name\*\*"**

**app.config['MAIL\_USE\_TLS']=True**

**app.config['MAIL\_USE\_SSL']=False**

**mail=Mail(app)**

**import sqlite3, hashlib, os**

**from werkzeug.utils import secure\_filename**

**app = Flask(\_name\_)**

**app.secret\_key = 'random string'**

**UPLOAD\_FOLDER = 'static/uploads'**

**ALLOWED\_EXTENSIONS = set(['jpeg', 'jpg', 'png', 'gif'])**

**app.config['UPLOAD\_FOLDER'] = UPLOAD\_FOLDER**

**def getLoginDetails():**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**if 'email' not in session:**

**loggedIn = False**

**firstName = ''**

**noOfItems = 0**

**else:**

**loggedIn = True**

**cur.execute("SELECT userId, firstName FROM users WHERE email = ?", (session['email'], ))**

**userId, firstName = cur.fetchone()**

**cur.execute("SELECT count(productId) FROM kart WHERE userId = ?", (userId, ))**

**noOfItems = cur.fetchone()[0]**

**conn.close()**

**return (loggedIn, firstName, noOfItems)**

**@app.route("/")**

**def root():**

**loggedIn, firstName, noOfItems = getLoginDetails()**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute('SELECT productId, name, price, description, image, stock FROM products')**

**itemData = cur.fetchall()**

**cur.execute('SELECT categoryId, name FROM categories')**

**categoryData = cur.fetchall()**

**itemData = parse(itemData)**

**return render\_template('home.html', itemData=itemData, loggedIn=loggedIn, firstName=firstName, noOfItems=noOfItems, categoryData=categoryData)**

**@app.route("/add")**

**def admin():**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT categoryId, name FROM categories")**

**categories = cur.fetchall()**

**conn.close()**

**return render\_template('add.html', categories=categories)**

**@app.route("/addItem", methods=["GET", "POST"])**

**def addItem():**

**if request.method == "POST":**

**name = request.form['name']**

**price = float(request.form['price'])**

**description = request.form['description']**

**stock = int(request.form['stock'])**

**categoryId = int(request.form['category'])**

**#Uploading image procedure**

**image = request.files['image']**

**if image and allowed\_file(image.filename):**

**filename = secure\_filename(image.filename)**

**image.save(os.path.join(app.config['UPLOAD\_FOLDER'], filename))**

**imagename = filename**

**with sqlite3.connect('database.db') as conn:**

**try:**

**cur = conn.cursor()**

**cur.execute('''INSERT INTO products (name, price, description, image, stock, categoryId) VALUES (?, ?, ?, ?, ?, ?)''', (name, price, description, imagename, stock, categoryId))**

**conn.commit()**

**msg="added successfully"**

**except:**

**msg="error occured"**

**conn.rollback()**

**conn.close()**

**print(msg)**

**return redirect(url\_for('root'))**

**@app.route("/remove")**

**def remove():**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute('SELECT productId, name, price, description, image, stock FROM products')**

**data = cur.fetchall()**

**conn.close()**

**return render\_template('remove.html', data=data)**

**@app.route("/removeItem")**

**def removeItem():**

**productId = request.args.get('productId')**

**with sqlite3.connect('database.db') as conn:**

**try:**

**cur = conn.cursor()**

**cur.execute('DELETE FROM products WHERE productID = ?', (productId, ))**

**conn.commit()**

**msg = "Deleted successsfully"**

**except:**

**conn.rollback()**

**msg = "Error occured"**

**conn.close()**

**print(msg)**

**return redirect(url\_for('root'))**

**@app.route("/displayCategory")**

**def displayCategory():**

**loggedIn, firstName, noOfItems = getLoginDetails()**

**categoryId = request.args.get("categoryId")**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT products.productId, products.name, products.price, products.image, categories.name FROM products, categories WHERE products.categoryId = categories.categoryId AND categories.categoryId = ?", (categoryId, ))**

**data = cur.fetchall()**

**conn.close()**

**categoryName = data[0][4]**

**data = parse(data)**

**return render\_template('displayCategory.html', data=data, loggedIn=loggedIn, firstName=firstName, noOfItems=noOfItems, categoryName=categoryName)**

**@app.route("/account/profile")**

**def profileHome():**

**if 'email' not in session:**

**return redirect(url\_for('root'))**

**loggedIn, firstName, noOfItems = getLoginDetails()**

**return render\_template("profileHome.html", loggedIn=loggedIn, firstName=firstName, noOfItems=noOfItems)**

**@app.route("/account/profile/edit")**

**def editProfile():**

**if 'email' not in session:**

**return redirect(url\_for('root'))**

**loggedIn, firstName, noOfItems = getLoginDetails()**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT userId, email, firstName, lastName, address1, address2, zipcode, city, state, country, phone FROM users WHERE email = ?", (session['email'], ))**

**profileData = cur.fetchone()**

**conn.close()**

**return render\_template("editProfile.html", profileData=profileData, loggedIn=loggedIn, firstName=firstName, noOfItems=noOfItems)**

**@app.route("/account/profile/changePassword", methods=["GET", "POST"])**

**def changePassword():**

**if 'email' not in session:**

**return redirect(url\_for('loginForm'))**

**if request.method == "POST":**

**oldPassword = request.form['oldpassword']**

**oldPassword = hashlib.md5(oldPassword.encode()).hexdigest()**

**newPassword = request.form['newpassword']**

**newPassword = hashlib.md5(newPassword.encode()).hexdigest()**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT userId, password FROM users WHERE email = ?", (session['email'], ))**

**userId, password = cur.fetchone()**

**if (password == oldPassword):**

**try:**

**cur.execute("UPDATE users SET password = ? WHERE userId = ?", (newPassword, userId))**

**conn.commit()**

**msg="Changed successfully"**

**except:**

**conn.rollback()**

**msg = "Failed"**

**return render\_template("changePassword.html", msg=msg)**

**else:**

**msg = "Wrong password"**

**conn.close()**

**return render\_template("changePassword.html", msg=msg)**

**else:**

**return render\_template("changePassword.html")**

**@app.route("/updateProfile", methods=["GET", "POST"])**

**def updateProfile():**

**if request.method == 'POST':**

**email = request.form['email']**

**firstName = request.form['firstName']**

**lastName = request.form['lastName']**

**address1 = request.form['address1']**

**address2 = request.form['address2']**

**zipcode = request.form['zipcode']**

**city = request.form['city']**

**state = request.form['state']**

**country = request.form['country']**

**phone = request.form['phone']**

**with sqlite3.connect('database.db') as con:**

**try:**

**cur = con.cursor()**

**cur.execute('UPDATE users SET firstName = ?, lastName = ?, address1 = ?, address2 = ?, zipcode = ?, city = ?, state = ?, country = ?, phone = ? WHERE email = ?', (firstName, lastName, address1, address2, zipcode, city, state, country, phone, email))**

**con.commit()**

**msg = "Saved Successfully"**

**except:**

**con.rollback()**

**msg = "Error occured"**

**con.close()**

**return redirect(url\_for('editProfile'))**

**@app.route("/loginForm")**

**def loginForm():**

**if 'email' in session:**

**return redirect(url\_for('root'))**

**else:**

**return render\_template('login.html', error='')**

**@app.route("/login", methods = ['POST', 'GET'])**

**def login():**

**if request.method == 'POST':**

**email = request.form['email']**

**password = request.form['password']**

**if is\_valid(email, password):**

**session['email'] = email**

**return redirect(url\_for('root'))**

**else:**

**error = 'Invalid UserId / Password'**

**return render\_template('login.html', error=error)**

**@app.route("/productDescription")**

**def productDescription():**

**loggedIn, firstName, noOfItems = getLoginDetails()**

**productId = request.args.get('productId')**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute('SELECT productId, name, price, description, image, stock FROM products WHERE productId = ?', (productId, ))**

**productData = cur.fetchone()**

**conn.close()**

**return render\_template("productDescription.html", data=productData, loggedIn = loggedIn, firstName = firstName, noOfItems = noOfItems)**

**@app.route("/addToCart")**

**def addToCart():**

**if 'email' not in session:**

**return redirect(url\_for('loginForm'))**

**else:**

**productId = int(request.args.get('productId'))**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT userId FROM users WHERE email = ?", (session['email'], ))**

**userId = cur.fetchone()[0]**

**try:**

**cur.execute("INSERT INTO kart (userId, productId) VALUES (?, ?)", (userId, productId))**

**conn.commit()**

**msg = "Added successfully"**

**except:**

**conn.rollback()**

**msg = "Error occured"**

**conn.close()**

**return redirect(url\_for('root'))**

**@app.route("/cart")**

**def cart():**

**if 'email' not in session:**

**return redirect(url\_for('loginForm'))**

**loggedIn, firstName, noOfItems = getLoginDetails()**

**email = session['email']**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT userId FROM users WHERE email = ?", (email, ))**

**userId = cur.fetchone()[0]**

**cur.execute("SELECT products.productId, products.name, products.price, products.image FROM products, kart WHERE products.productId = kart.productId AND kart.userId = ?", (userId, ))**

**products = cur.fetchall()**

**totalPrice = 0**

**for row in products:**

**totalPrice += row[2]**

**return render\_template("cart.html", products = products, totalPrice=totalPrice, loggedIn=loggedIn, firstName=firstName, noOfItems=noOfItems)**

**@app.route("/removeFromCart")**

**def removeFromCart():**

**if 'email' not in session:**

**return redirect(url\_for('loginForm'))**

**email = session['email']**

**productId = int(request.args.get('productId'))**

**with sqlite3.connect('database.db') as conn:**

**cur = conn.cursor()**

**cur.execute("SELECT userId FROM users WHERE email = ?", (email, ))**

**userId = cur.fetchone()[0]**

**try:**

**cur.execute("DELETE FROM kart WHERE userId = ? AND productId = ?", (userId, productId))**

**conn.commit()**

**msg = "removed successfully"**

**except:**

**conn.rollback()**

**msg = "error occured"**

**conn.close()**

**return redirect(url\_for('root'))**

**@app.route("/logout")**

**def logout():**

**session.pop('email', None)**

**return redirect(url\_for('root'))**

**def is\_valid(email, password):**

**con = sqlite3.connect('database.db')**

**cur = con.cursor()**

**cur.execute('SELECT email, password FROM users')**

**data = cur.fetchall()**

**for row in data:**

**if row[0] == email and row[1] == hashlib.md5(password.encode()).hexdigest():**

**return True**

**return False**

**@app.route("/register", methods = ['GET', 'POST'])**

**def register():**

**if request.method == 'POST':**

**#Parse form data**

**password = request.form['password']**

**email = request.form['email']**

**firstName = request.form['firstName']**

**lastName = request.form['lastName']**

**address1 = request.form['address1']**

**address2 = request.form['address2']**

**zipcode = request.form['zipcode']**

**city = request.form['city']**

**state = request.form['state']**

**country = request.form['country']**

**phone = request.form['phone']**

**with sqlite3.connect('database.db') as con:**

**try:**

**cur = con.cursor()**

**cur.execute('INSERT INTO users (password, email, firstName, lastName, address1, address2, zipcode, city, state, country, phone) VALUES (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, ?)', (hashlib.md5(password.encode()).hexdigest(), email, firstName, lastName, address1, address2, zipcode, city, state, country, phone))**

**con.commit()**

**msg = "Registered Successfully"**

**except:**

**con.rollback()**

**msg = "Error occured"**

**con.close()**

**return render\_template("login.html", error=msg)**

**@app.route("/registerationForm")**

**def registrationForm():**

**return render\_template("register.html")**

**def allowed\_file(filename):**

**return '.' in filename and \**

**filename.rsplit('.', 1)[1] in ALLOWED\_EXTENSIONS**

**def parse(data):**

**ans = []**

**i = 0**

**while i < len(data):**

**curr = []**

**for j in range(7):**

**if i >= len(data):**

**break**

**curr.append(data[i])**

**i += 1**

**ans.append(curr)**

**return ans**

**if \_name\_ == '\_main\_':**

**app.run(debug=True)**

## 3.1. Computational studies

#### 3.1.1. Domain, language and implementation details

Most research on chatbot computational aspects had English as their primary language (76.3%), followed by papers on Indonesian chatbots (6.8%) and other languages like Chinese and Bangla. However, the resulting papers were mostly not fashion-specific (87.7%). Contrastingly, a few papers like Liao, Zhou, Ma, Hong, and Chua ([2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) and Vaccaro, Agarwalla, Shivakumar, and Kumar ([2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) were fashion-specific. It is also worth mentioning that, while non-computational research mainly employed a diversity of ready-to-use chatbot tools like Amazon Alexa, computational papers usually focus on chatbot development using a specific programming language.

#### 3.1.2. Media

Most recent papers already consider natural language text as the input method (79.4%), except for Aarthi ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)), Pricilla, Lestari, and Dharma ([2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) and Wintersberger, Klotz, and Riener ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) which allow the user to choose from a fixed list of options in specific situations. A few studies (15.1%) are voice-based, and others employ avatars to enhance users’ confidence and perception of the system (Eisman, Navarro, & Castro, 2016).

#### 3.1.3. Dialog control

A turn is each single contribution that composes a dialog (Jurafsky & Martin, [2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)). Reactive agents respond to user interactions, thus solely exploiting information explicitly provided by the user’s initiative. Conversely, proactive chatbots employ an engagement strategy to interact and influence users, making use of predictions about users’ demands.

Employed dialog turns were mostly reactive, except for Aarthi ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) and Liu, Jiang, Xiong, Yang, and Ye ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) systems, which could take control of the dialog flow. The dialog control options in Table 1 were well-explored. Neural-based controls seem to be a trending research direction. And, the majority are made to solve a specific task, such as answering questions about product characteristics or mediating a purchase. Most papers aimed at solving retail tasks with a minority addressing ‘chit-chat’ chatbots. This aspect could be potentially explored in fashion-specific chatbot research.

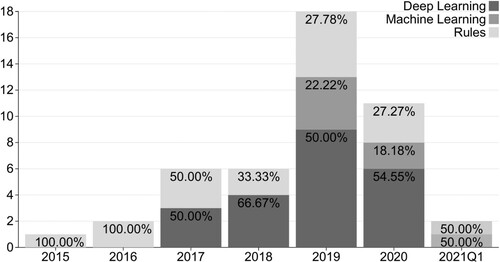
To aid users in solving a specific task, chatbots must have a dialog policy, which is responsible for deciding which action the system should take in the next iteration. Although many distinct approaches have been proposed, we identified three main subcategories in the literature:

1. Predict the user’s intent, and then choose a specific answer in a finite set of predefined responses;
2. Analyse the similarity between the user’s questions and questions in a dataset, and choose an answer accordingly; and
3. Sequence-to-sequence DL approaches decode the user intent first, and synthesise a response.

#### 3.1.4. Input processing and natural language understanding

The investigated dialog policies were based on variations of DL architectures such as Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN), as shown in [Figure 5](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0005). Interest in DL picked up starting in 2017. Nevertheless, classical ML approaches and rule-based methods are still employed in state-of-the-art research.

Figure 5. Input processing/NLU approaches over time.

[](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)

[Display full size](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)

When user interactions are enacted through buttons or multiple-choice interfaces, the expressed intents and/or other relevant information may be straightforwardly comprehended by the chatbot. However, when human natural language contained in unstructured data is allowed in the form of text (possibly extracted from speech), specific algorithms for Natural Language Understanding (NLU) must be employed. Rule-based algorithms employ handcrafted hard rules by using, for instance, the Artificial Intelligence Mark-up Language (AIML). Classical ML algorithms in general combine the extraction of handcrafted features from unstructured textual data, such as n-gram counts or the Term Frequency–Inverse Document Frequency (TF-IDF) statistical measure. DL algorithms present state-of-the-art performance by employing deep neural network architectures for sequence processing, which mainly include variations of CNN and RNN to recognise complex patterns from data.

Chatbot answers also follow approaches that resemble NLU algorithms, and thus may be categorised as rule-based, when sentences are generated from pre-written templates; or neural-based, when sentences are generated from textual training data. In the latter, variants of CNN, RNN, and Generative Adversarial Networks (GAN) are among the most popular approaches. For a comprehensive introduction to NLP algorithms, we refer the reader to Jurafsky and Martin

#### 3.1.5. Chatbot answers

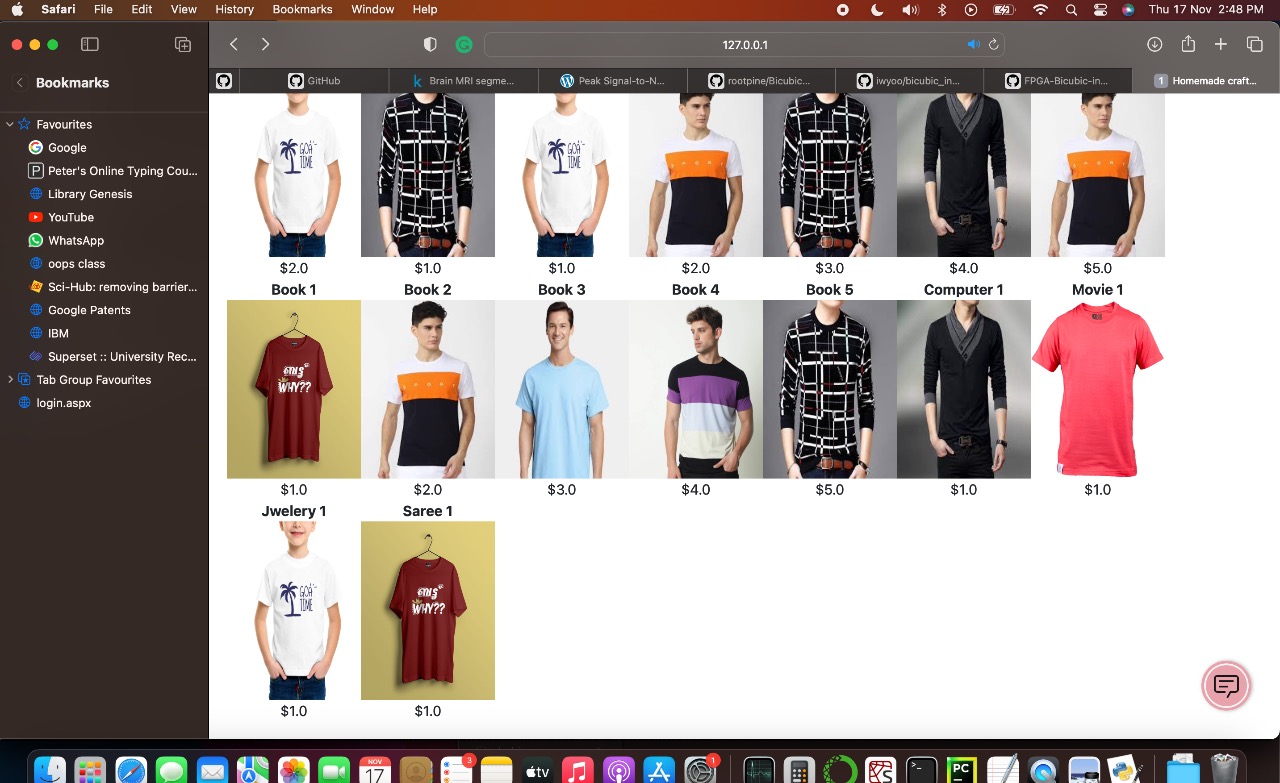
Algorithms to generate or to choose predefined chatbot answers have still been employing mostly rule-based approaches, in which answers are selected from a predefined set. Nonetheless, neural-based methods exploiting neural architectures such as RNN, CNN, and GAN have already been proposed, as shown in [Figures 6](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0006) and [7](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0007). In particular, IR and RNN methods prevail with over 93.1% of the selected papers addressing them ([Figure 6](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417#F0006)).

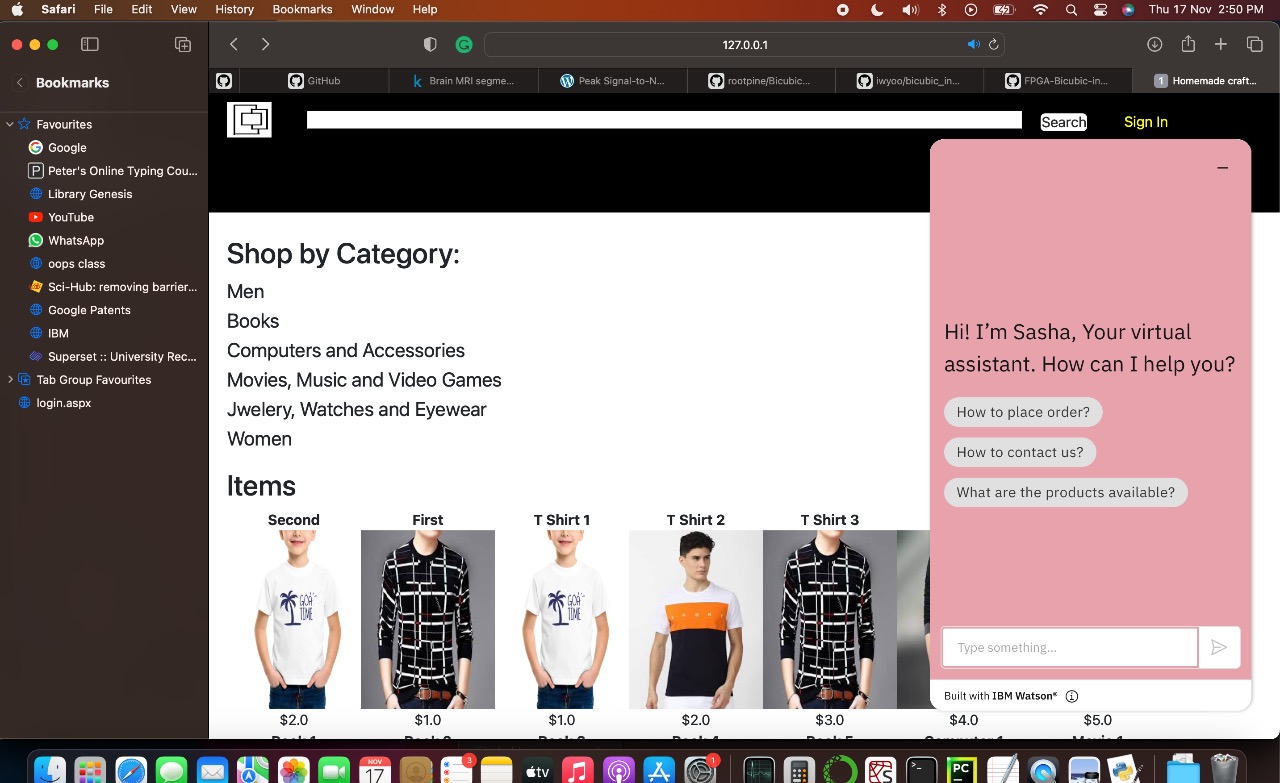
#### 4.1.1. Fashion-domain specific computational research opportunities

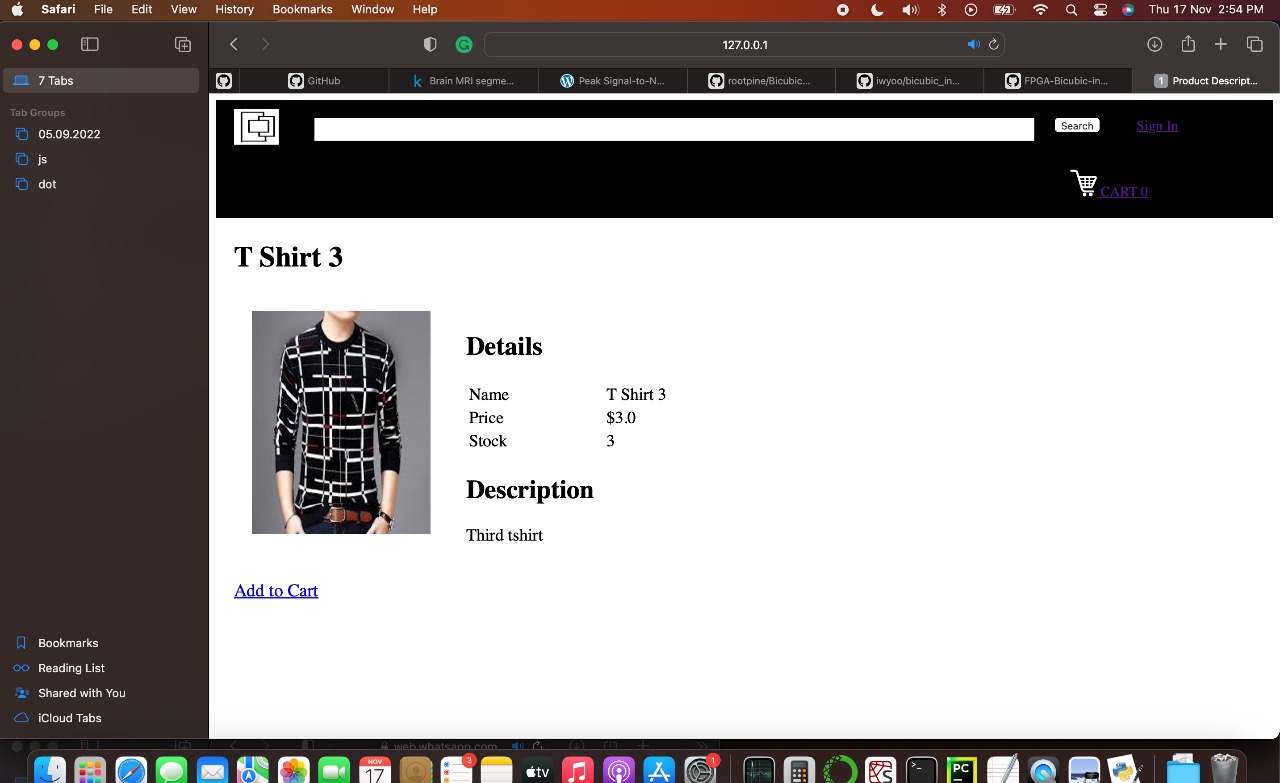
Four major opportunities may be highlighted:

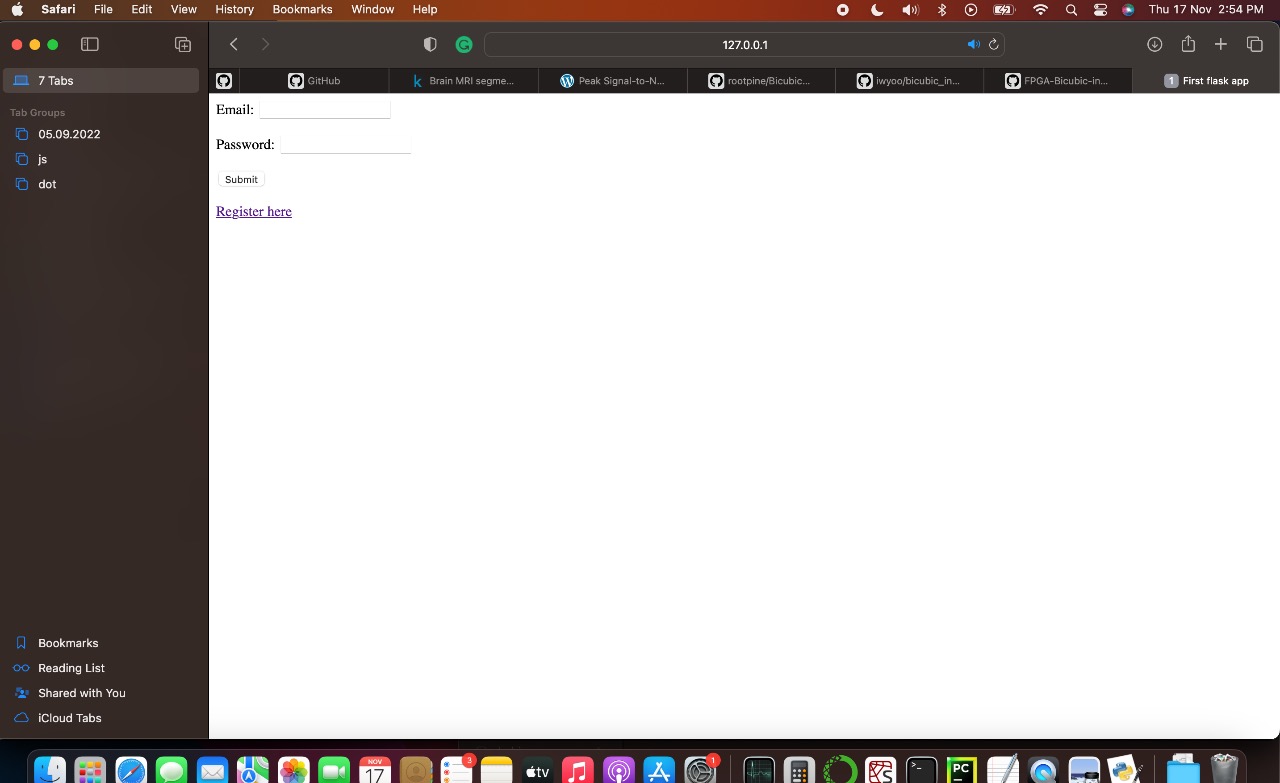
1. More encompassing, real-life, professional-grade datasets on fashion items are needed if fashion chatbots are to be more extensively trained and trusted: given that no large public dataset is readily available, Nazir et al. ([2019](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) alternatively collected data manually from clothing brands websites and from Facebook posts and comments. Others like Liao et al. ([2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) proposed to transfer knowledge from richer domains.
2. Applying chatbots to fashion e-commerce specific needs such as multimedia conversations (e.g. text, voice and images): Pantano, Passavanti, Priporas, and Verteramo ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) also revealed a lack of innovative technologies in the fashion luxury industry. De Carolis et al. ([2015](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) recommended new clothes based on users’ visual cues, Sapna et al. ([2019](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) make the chatbot ask users their preferences and Liao et al. ([2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) proposed a multimodal chatbot to gather user’s visual and textual clothing needs. Future work could explore chatbot’s retrieval of users’ short- and long-term preferences to better recommend fashion products.
3. Integration of chatbots to other fashion applications in different points within the consumer journey: since chatbots can provide personalised information for consumers across their journey (Vaccaro et al., [2018](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)), research that explores such integration, e.g. virtually trying a recommended item on, shows potential. Another opportunity is the integration with Augmented Reality (AR) – e.g. in Virtual Fitting Room (VFR) applications. Moriuchi, Landers, Colton, and Hair ([2020](https://www.tandfonline.com/doi/full/10.1080/17543266.2021.1990417)) compare e-commerce apps that use chatbots and those which use AR, but they do not propose integration of both to serve the fashion domain.

# IMPLEMENTATION AND OUTPUT SCREENSHOTS

****







# CONCLUSION

The present paper presents the development of a system that recognizes fashion similar images. We accomplish this by implementing an already existing CNN model with transfer learning for cloth image recognition using different libraries. For this purpose, we created a planfor collecting data and for developing the steps needed for preprocessing and cleaning up the data. We took into account features like patterns, machine, fabric, style etc. After extensive preprocessing and cleaning of data in a dataset, we constructed the model of stacked CNN to predict the features specific to these attributes and to train the models with the dataset to generate accurate predictions regarding almost all forms of images. A stacked CNN was used and implemented, with the help of this algorithm through which the system can recommend similar images This is the last test to assess if deep learning for style recovery is at a high development andcan be utilized in making fashion choices.

.