

Spring 5-22-2017

## Shopbot: An Image Based Search Application for E-Commerce Domain

Nishant Goel  
*San Jose State University*

Follow this and additional works at: [https://scholarworks.sjsu.edu/etd\\_projects](https://scholarworks.sjsu.edu/etd_projects)



Part of the [Artificial Intelligence and Robotics Commons](#)

---

### Recommended Citation

Goel, Nishant, "Shopbot: An Image Based Search Application for E-Commerce Domain" (2017). *Master's Projects*. 516.

DOI: <https://doi.org/10.31979/etd.r7a5-6dzf>

[https://scholarworks.sjsu.edu/etd\\_projects/516](https://scholarworks.sjsu.edu/etd_projects/516)

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact [scholarworks@sjsu.edu](mailto:scholarworks@sjsu.edu).



# SAN JOSÉ STATE UNIVERSITY

Shopbot: An Image Based Search Application for E-Commerce Domain

A Project Report

Presented to

Department of Computer Science

San Jose State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

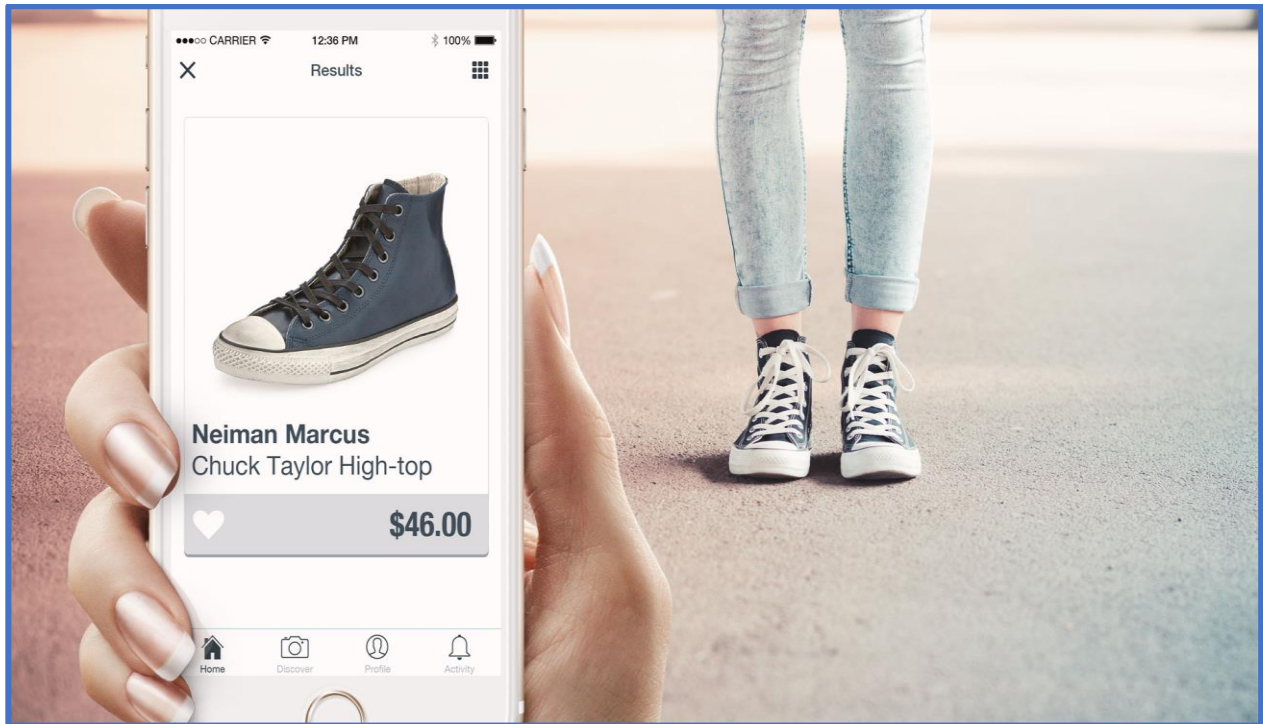
By

Nishant Goel

May 2017

# Shopbot

## An Image Based Search Application for E-Commerce Domain



The Designated Project Committee Approves the Project Titled

Shopbot: An Image Based Search Application for E-Commerce Domain

By

Nishant Goel

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

May 2017

Dr. Robert Chun

Department of Computer Science, SJSU

Dr. Jenny Lam

Department of Computer Science, SJSU

Mr. Nimit Patel

Software Engineer, F5 Networks

## **ABSTRACT**

For the past few years, e-commerce has changed the way people buy and sell products. People use this business model to do business over the Internet. In this domain, Human-Computer Interaction has been gaining momentum. Lately, there has been an upsurge in agent based applications in the form of intelligent personal assistants (also known as Chatbots) which make it easier for users to interact with digital services via a conversation, in the same way we talk to humans. In e-commerce, these assistants offer mainly text-based or speech based search capabilities. They can handle search for most products, but cannot handle search that is based on product features, for instance color or pattern of a T-shirt. Most of the times, it is difficult for users to define these characteristics while searching for a product. Furthermore, a growing number of consumers rely on social media to make a purchasing decision. They try to find out what is trending right now and look for similar items. This brings us the need of a virtual shopping assistant or a shopbot which recommends products based on an image of the product provided by a user. It will be designed to provide relevant responses to the user queries by performing image recognition. This report explains the proposed approach along with the implementation for the virtual shopping assistant.

## **ACKNOWLEDGMENT**

I would like to take this opportunity to thank my project advisor, Dr. Robert Chun, for his guidance and support throughout my project and graduate studies. He always motivated me whenever and wherever, I got stuck in my project.

I am very thankful to the committee member, Dr. Jenny Lam for being very encouraging and helpful throughout the project.

I would also like to thank Mr. Nimit Patel for his constant support and guidance. He always showed keen interest in knowing if I am facing any technical issues while working on the project.

Finally, I would like to thank Department of Computer Science at San Jose State University for giving me this opportunity to pursue Master's degree in Computer Science.

## TABLE OF CONTENTS

<b>1. Introduction</b> .....	9
1.1. <i>Problem Statement</i> .....	9
1.2. <i>Solution</i> .....	9
1.3. <i>Scope</i> .....	10
1.4. <i>Target Users</i> .....	10
<b>2. Background</b> .....	12
<b>3. Motivation</b> .....	22
3.1. <i>Research Question</i> .....	22
<b>4. Related Work</b> .....	23
<b>5. Product Design</b> .....	27
5.1. <i>Overview</i> .....	27
5.2. <i>Architecture</i> .....	29
5.2.1. <i>User Interface</i> .....	30
5.2.2. <i>Middleware</i> .....	30
5.2.3. <i>Backend</i> .....	40
5.3. <i>Implementation</i> .....	42
<b>6. Experimental Results</b> .....	49
<b>7. Conclusion</b> .....	53
<b>8. Future work</b> .....	55
<b>9. References</b> .....	57
<b>10. Appendix</b> .....	61

## LIST OF FIGURES

1. Social Media Images .....	11
2. Intelligent Assistants .....	12
3. NLP based Shopping bot.....	14
4. Speech to text Conversion .....	16
5. Stages of Text Processing .....	16
6. Response Conversion .....	17
7. Main steps in Human Computer Interaction .....	18
8. Overview of Image Recognition Tasks .....	22
9. Architecture of Shopbot .....	29
10. Scraped Data Set .....	32
11. Metadata file .....	32
12. Test Image for Research.....	33
13. Microsoft Computer Vision API .....	34
14. Google Vision API .....	35
15. Response of Google Vision API .....	36
16. Amazon AWS Rekognition .....	37
17. CloudSight API.....	39
18. Shopstyle Product API .....	41
19. GET Request of Backend API .....	42
20. Image Upload .....	44
21. Image Classification .....	45
22. Retrieval .....	46



23. Product Listings .....	47
24. Making a Purchase .....	48
25. Actual Product Page .....	48

## **1. INTRODUCTION**

### **1.1 Problem statement**

Nowadays, intelligent agents are the trend when it comes to online shopping. These agents or chatbots are often used as customer service applications in order to reduce the waiting time for users by answering frequently asked questions. In online shopping, product search is a key component [1]. When a user is online to buy a particular product, in most of the cases, he finds it by entering product related information on search engines like Amazon or eBay. Based on the input provided, a list of similar products is recommended to the user. If the user is not sure of the product information or he has just seen something which he wants to buy, searching for that item online following the traditional way could be tedious. These search engines allow users to search based on a criterion like keywords, categories, etc. However, they do not provide any search functionality based on images. For instance, suppose you see someone wearing a pair of Nike shoes which you like a lot. Now going online and finding the same pair of shoes could be time consuming as you might not know the exact brand of the shoes or its exact category. It is not easy to describe product based features like shape and texture. Another instance could be specifying featured-based products like T-shirt with a particular pattern or logo. Hence, a capability is needed which will allow you to take a picture of an item and recommends you that same or similar items from various e-commerce web sites. The agent based assistants can be trained to provide this recommendation capability for a particular search query via image along with text and speech.

### **1.2 Solution**

The solution is to build a chatbot based Image Search & Recommendation system. We need to build an intelligent assistant that works with image recognition along with natural language

processing (NLP). This system captures the image and extracts all the information related to different objects and use that information to retrieve and recommend products from e-commerce websites.

### **1.3 Scope**

The scope of the project is to come up with a virtual shopping assistant having image recognition capabilities. Existing systems have only text and speech recognition capabilities for product search. However, they do not have image based search functionality. Hence, this project will focus on image recognition to offer users with a powerful and seamless search capability to search for various kinds of items on e-commerce websites using just images. As far as the current scope is concerned, we target Shopstyle as an e-commerce platform due to its availability since it is open source. Moving forward, we plan to integrate other e-commerce platforms like Amazon, eBay, etc.

### **1.4 Target Users**

The target users for this system are people of all age groups. Everyone is considered as the target user irrespective of their language or educational background. To consider a few, there are segments of population which are less likely to buy products online due to less exposure to technology or their attitude towards online shopping altogether. This shopping assistant will surely help in closing the gap between them and ecommerce. It will make things simpler for them, to purchase items by uploading images of the items they want to buy. For instance, elderly people who stay alone at home and do not know much about how to search for products online, can just search for the products by uploading items' images. Other age group which actively shops online is Millennials. They are the largest segment of population worldwide which shops online at all times as they are very active on social media and like everything

online to be fast and seamless. Hence, they become the tempting targets for any e-commerce based service as well. Furthermore, with the upsurge in use of social media across the world, people rely heavily on it to drive their purchasing decisions. They generally look for items which are trending and hence, prefer buying those over options fallen out of fashion. They follow celebrities and public figures and emulate their style by wearing similar apparel or shoes. So, having an application based on visual search will help them keeping up with current fashion and having a great online shopping experience at the same time.

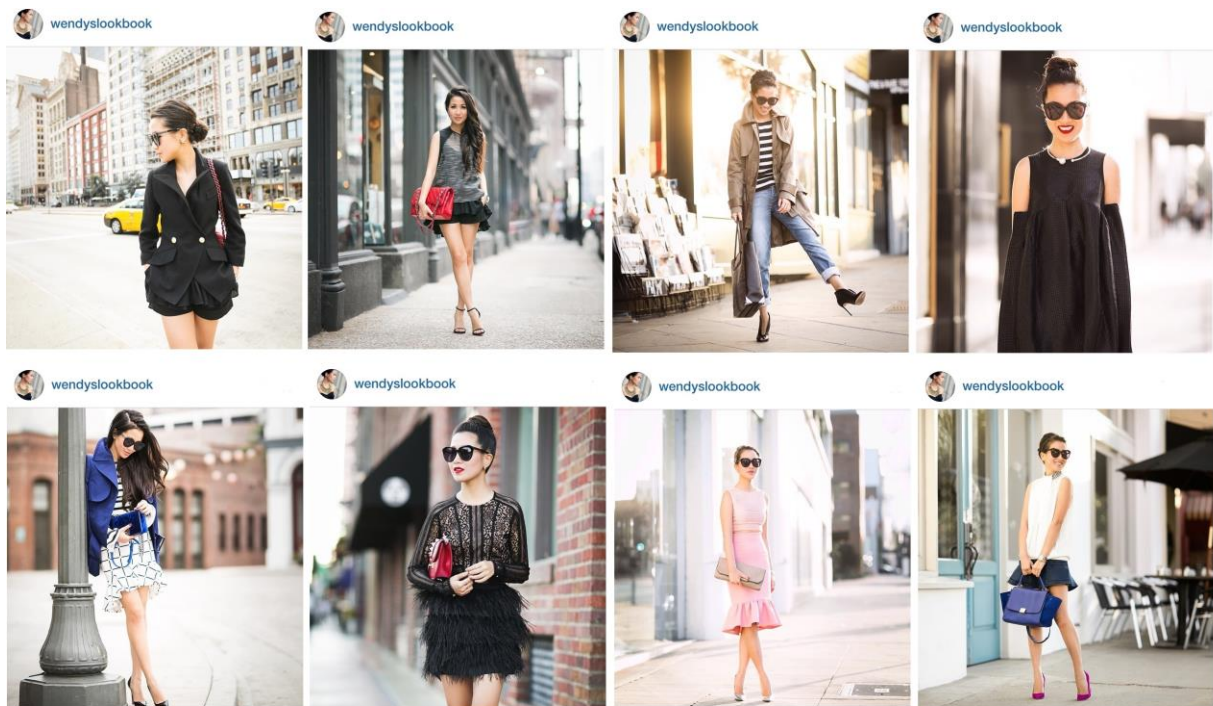


Fig. 1. Social media images of various models

## 2. BACKGROUND

The aim is to build a virtual shopping assistant or chatbot which is an intelligent assistant trained for e-commerce, to provide a seamless search capability for searching products on e-commerce websites using image recognition. But, before we start with explaining the proposed system, we should understand what a chatbot is and different NLP techniques implemented currently. Then, we study how we can implement image recognition for searching products on e-commerce websites.

Nowadays, chatbots in the form of conversational user interfaces are the latest fad [2]. This concept is being hailed by various influential people across the Silicon Valley as a prime shift in the way we interact with digital content. A chatbot is a software application which is developed to simulate a conversation with users through text or speech. It can be designed for small talk or as a means of interaction with users by answering regular questions. It analyzes the user's intent and message's context and tries to answer with a response. The concept of chatbots has been there since a long time. However, various artificial intelligence techniques are being explored to incorporate them to resolve real life problems. In this project, we are trying to build an image based chatbot.



Fig. 2. Intelligent Assistants (Chatbots) [3]

In the e-commerce domain, a chatbot acts as a shopping bot (shopbot) which assists consumers with online shopping by searching, identifying and comparing various products across numerous websites [4]. These websites maintain a vast inventory of a wide range of products in various categories. These inventories are spread across various catalogues on numerous web pages. These websites provide navigation via menus and search bars. Browsing through these pages can be tedious and a painstaking process. The search facilities offer search by keyword or by category to retrieve products based on a user's input query. However, there are many situations in which the results are not relevant to the user or might be unsatisfactory. For instance, if you want to buy a jacket with a specific pattern, you might be recommended with products which are not relevant to your query. Also, there can be cases when a user does not know much about the product he expects to purchase.

The chatbot tries to resolve these issues by making it easier to interact with e-commerce websites. It understands and interacts with the user in natural language. After being integrated with an ecommerce website, it accesses varied products available in that company's inventory and helps you make a decision for buying a product. When you do not know much about the product you want to buy, it will converse with you as your personal shopping assistant to help you narrow down your selection criteria for that product. For instance, if you want to buy a pair of shoes but have not decided on the brand name or the color, it will help you select the shoes of your fondness. Fig. 3 shows an NLP based shopping bot [5].

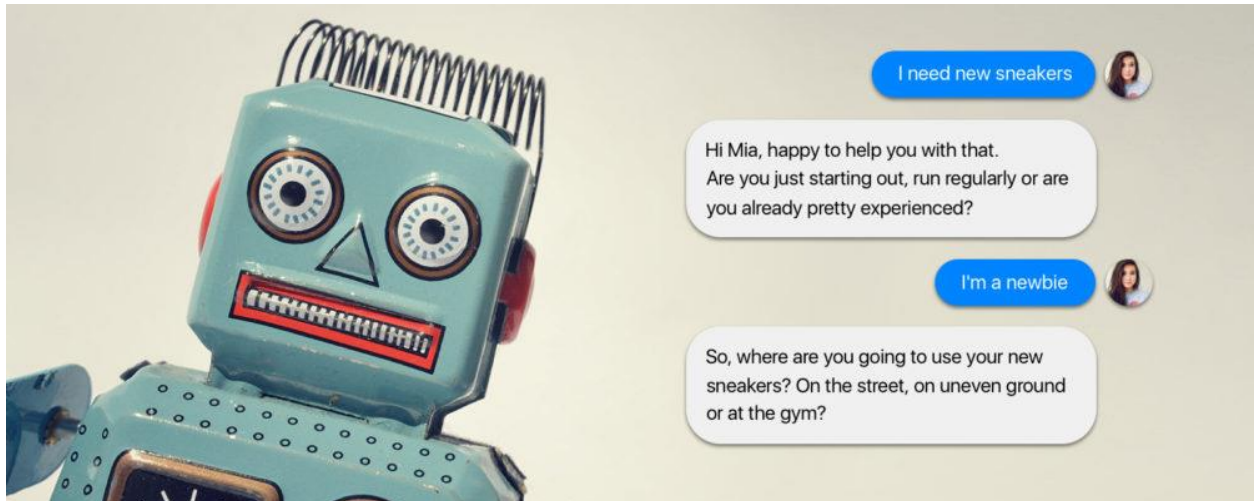


Fig. 3. An NLP based Shopping bot [5]

Existing chatbots are developed using NLP in which they mainly deal with text and speech based recognition. However, we will mainly focus on an image based search application which will use image recognition techniques to extract various product features and provide users with exact and similar matches from the website's inventory.

Now, firstly we will discuss various design techniques which are used to build chatbots. Moving forward, we will see how speech analysis and image processing work in order to build chatbot as an intelligent assistant.

There are various core design techniques that a chatbot designer should be familiar with; some of them are as follows [6]:

- **Pattern Matching:** This is used to design question-answer based chatbots. It helps in answering natural language queries and finds semantic meaning out of them.
- **Parsing:** This is a technique used to analyze text using various NLP functions like Trees in NLTK library of Python. This library offers built-in methods for NLP.

- **Artificial Intelligence Markup Language (AIML):** This is a language which simplifies conversational modelling in the context of a two-way conversation. It depends on tags which act as identifiers in the code of a chatbot. It is based on XML language.
- **Chat Script:** This is a technique which is used in situations when nothing matches in AIML. It provides a meaningful default answer by working on the most appropriate syntax. It has various features like logical OR, logical AND, facts, and variables.
- **Markov Chain:** This is a technique which is used to build probabilistically correct responses for chatbots. It is based on the fact that the probability of a word or letter occurrence in a piece of text is fixed.
- **Semantic Networks:** This concept is used to figure out the relations between language elements like synonyms, hyponyms, etc. These relations can be represented in graphs to enable searching based on reasoning rules.

Now, we discuss how speech analysis works. There are mainly three stages in speech analysis.

- 1) Recognizing speech and converting it to corresponding text,
- 2) Processing text, and
- 3) Taking appropriate steps to execute an action

First of all, a speech is given as an input to a digital signal processing module. This module converts the speech input into a sequence consisting of speech related information. Then, this information is converted into text per specific instructions. At this point, we obtain text from the input speech for further processing [6]. Fig. 4 illustrates the above-mentioned process.



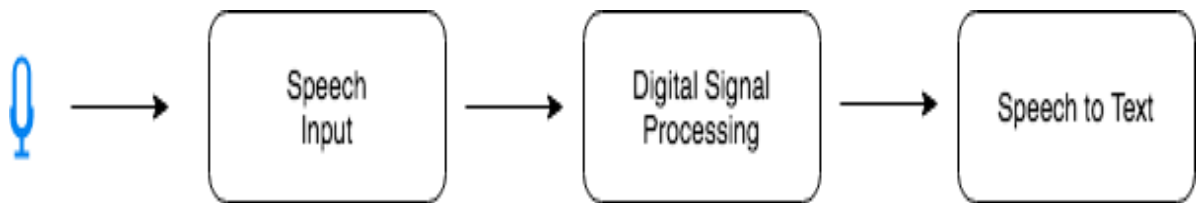


Fig. 4. Phase of Speech to Text Conversion

Once you get the text from the speech input, text processing phase begins where the resulting text of the previous phase is divided into individual words which get tagged based on their position in the sentence. Parts of speech based labels are used to perform tagging. Then, chunking operation is used to create phrases from tagged words. Redundant words are removed to retrieve keywords from the phrases which are later, validated for correctness. Fig. 5. shows this phase of text processing.

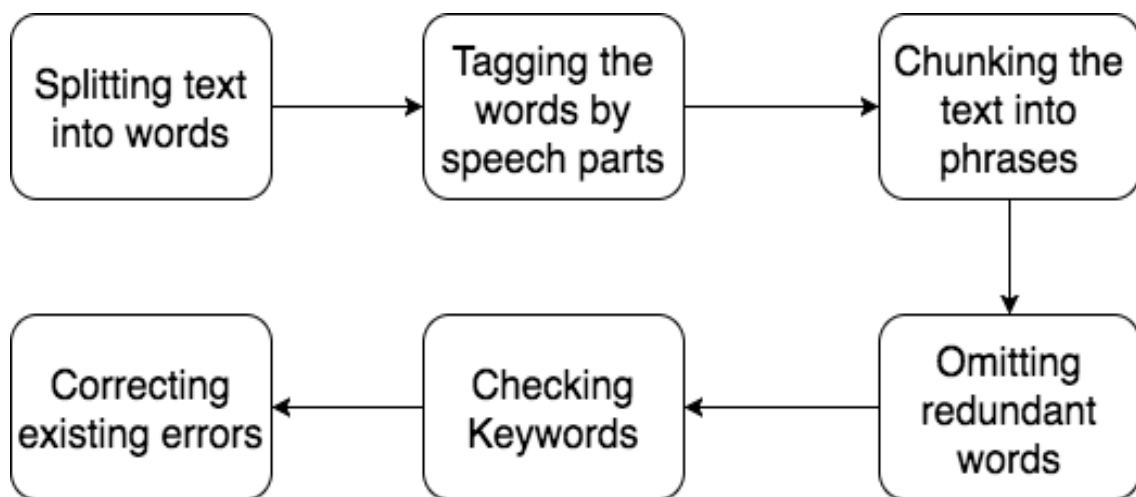


Fig. 5. Phase of Text Processing

Finally, the output of the previous phase of speech-text processing i.e., keywords, is passed to a chatbot device which will give us an output in the form of a programmed response. Fig. 6 illustrates an overview of the third phase.

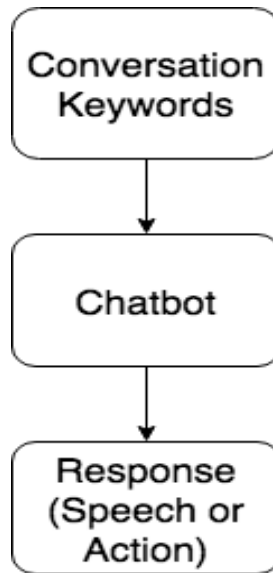


Fig. 6. The Phase of Generating the Response or Action

As explained earlier, human-computer interaction can be performed using both text or speech. can be done by conversing either in text or speech. Once speech is converted to text, both ways have very similar processing. Fig. 7 illustrates the complete process of speech recognition.

In conversational systems like chatbots, the quality of human-computer interaction is mainly affected by the parameters stated as follows:

- 1) Text Analysis to generate keywords using various grammar sets,
- 2) Pattern matching along with techniques to access database, and
- 3) Response type based on the specific application.

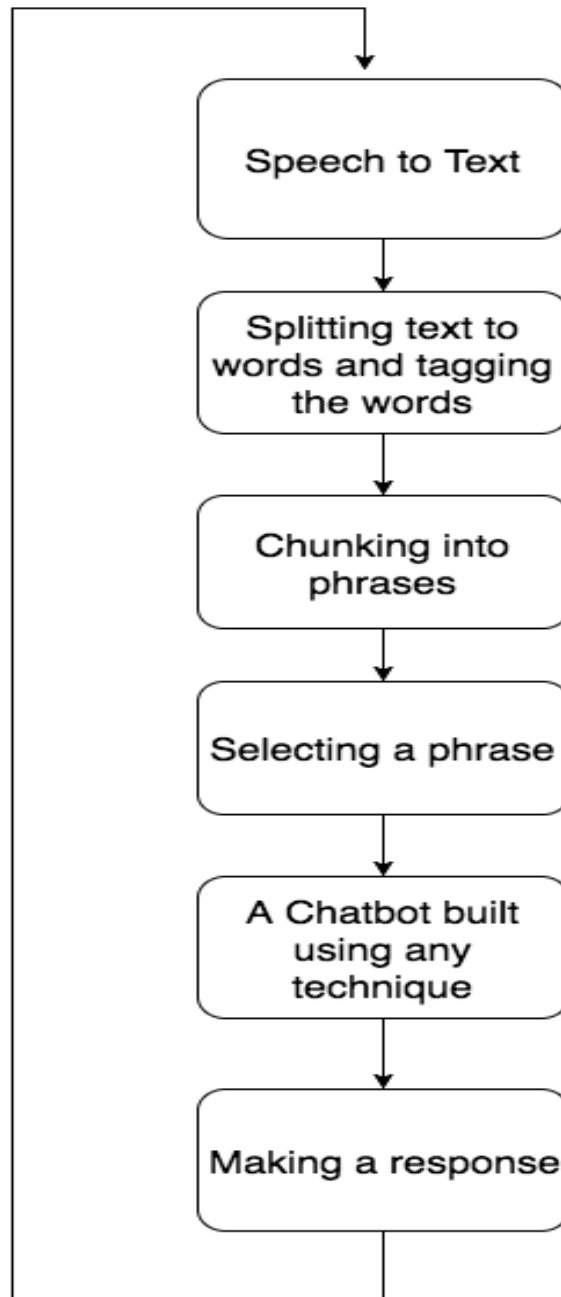


Fig. 7. Complete process of Speech Recognition

So, this is how speech recognition works. Now, we will discuss about the approaches to perform image recognition on the images uploaded by user.

The conventional problem in computer vision, especially in image processing is to determine if an image contains some specific objects, features, or activities [7]. Based on this, there are multiple types of image recognition that exist with different goals:

- **Object Recognition:** This is the way to recognize one or more pre-learned objects or their classes, when present as either in two-dimensional or three-dimensional space. For instance, Google Goggles and Blippar which are visual search based applications which perform object recognition.
- **Identification** – This is used to recognize an individual instance of an object. For instance, identifying a specific person's face, handwritten digits recognition, etc.
- **Detection** – This is used to recognize an image data for a specific scenario. For example, detection of uncharacteristic tissues or cells in images in the field of medical science. This is sometimes used to find particular regions of interesting data from the images in order to further analyze them by using computationally advanced techniques to gain maximum accuracy.

The state of the art algorithms in image recognition are based on a system called Convolutional Neural Networks (CNN). To demonstrate the offerings of CNN, ImageNet is used as a benchmark in detecting and classifying objects. This is a Visual Recognition Challenge which is happened at large scale using millions of images from various classes. Several experiments proved that CNN on ImageNet data set gives highly accurate results and performance almost close to a human response. However, there are certain challenges for which these algorithms struggle. These challenges are related to recognizing small or thin objects like small ant on a stem of a flower [8]. Also, they face difficulty in classifying images having filters such as images with Instagram filters. But, it's not at all difficult for humans to recognize those images.

However, recognizing fine-grained classes from an image could pose trouble to humans such as finding species of a bird, which can be easily classified using CNN.

Other image recognition dedicated application are based on tasks [9], such as:

- Optical character recognition
- Facial recognition
- Content Detection
- Pose estimation
- 2D Code Reading

In computer vision, there are some applications which works as individually to solve specific problems of object detection. However, other applications are organized as sub entities in a larger system to handle multiple functionalities such as image acquisition, feature extraction, etc. Their specific implementation depends on the use cases of the problem. Many functional tasks are specific to the application.

An image recognition system consists of typical functional tasks stated below [9].

- **Image Acquisition:** An image is acquired using various image sensors. There are various types of sensors available such as light-sensitive cameras, ultra-sonic cameras, radar, and others. Based on the sensor type, an image is retrieved which could be either a two-dimensional image, a three-dimensional scene, or a sequence of images [10].
- **Pre-processing:** Prior to the process of feature extraction, the input image is pre-processed based on certain assumptions or criteria like aspect ratio, resolution, color scale, etc. Some of the pre-processing tasks are histogram equalization, re-sampling, reducing the noise, etc.

- **Feature extraction:** This is one of the important phases in image recognition because all the main features of the images are extracted in this phase [10]. Some of those typical features which are extracted are:
  - Complex features such as color, texture, or shape
  - Eigen vectors
  - Localized interest points
- **Segmentation:** While processing the image, various image regions and points are selected which seem to be relevant for further processing [10]. For instance, breakdown of various one image into multiple image regions where each region contains a single object.
- **High-level Processing:** This phase comprises of processing of input coming from the segmentation phase in the form of relevant image points or regions containing specific objects [10]. The input data is verified if it satisfies the application or model specific assumptions. Then, various parameters related to object size and shape are estimated. Finally, the objects are classified into multiple categories based on their features.
- **Decision making:** This step is to make final decision about whether there is a match or not within the application [10].

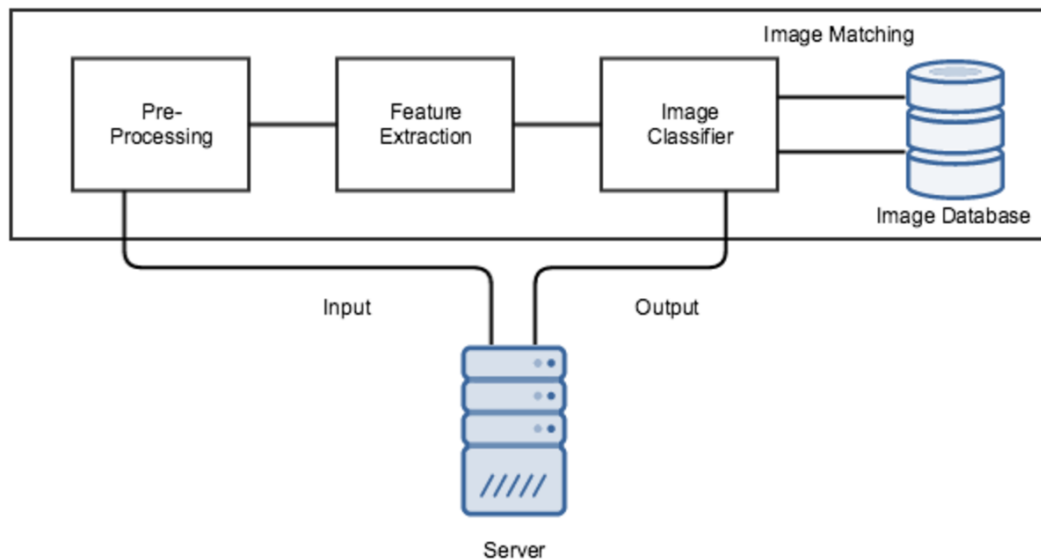


Fig. 8. High Level Overview of Image Recognition Tasks

Fig. 8. shows a high level overview of image recognition system comprising of various functional tasks.

### 3. MOTIVATION

The motivation for this project came by understanding and analyzing the various problems that people face while shopping online. To resolve a few of these problems and make their life easier, this system can certainly help in a big way.

#### 3.1 Research Question

As part of our research, our main focus has been to take a step back and try to understand the needs of the users who love shopping online and the challenges they face while doing that. Our goal is to resolve as many challenges as we can by making a robust and intelligent system in the form of a chatbot which can make their life easier and organized. When we started with the

research, we were primarily focused on finding existing systems in this domain, how those are being leveraged by users, and identifying various other opportunities which can enhance the user experience of online shopping. This gives rise to our research question: “Could we develop a Virtual Shopping Assistant based on Image Recognition, allowing users’ to search for similar products online by uploading an image?” This question is going to be the base of our future research and implementation.

#### **4. RELATED WORK**

Over the past years, Artificial Intelligence (AI) has been revolutionizing the mainstream computing. It has given applications various powerful abilities in the form of machine learning algorithms for computer vision and NLP which were next to impossible before. With the advent of these technologies, we have seen so much new innovations being happening around us especially in e-commerce in the form of intelligent assistants.

In 1960s, one of the applications which involved understanding natural language was ELIZA [11] which mainly dealt with recognizing phrases and cue words from the input and provided pre-typed responses in order to have a meaningful conversation [12]. Though it followed an approach of string substitution and providing pre-recorded responses, it was considered as an intelligent assistant by a lot of people. Then, came various chatbots like ALICE [13] and CleverBot [14] which used AIML based techniques to provide generic responses to converse with the user. As explained in section 2, AIML is the language to build an agent based chatbot. These kinds of bots are being used as part of various applications. As most of the bots were using AIML technique, Salvatore La Bua [15] came up with a different technique of Latent Semantic Analysis. It is basically a statistical modelling technique to compute semantic



similarity between different parts of information. As the research progressed, different functionalities were incorporated within chatbot in order to make it intelligent.

In e-commerce domain, these intelligent assistants are agent based applications equipped with knowledge to transform information between vendors and consumers. They help to meet the requirements of the users when it comes to online shopping. Based on what users are looking for, they help them by searching, identifying and comparing products in lesser time. For instance, Inktomi Shopping Engine [16] and mySimon [17] which are price comparison based search engines compare prices of products across various websites and provide you with the best price. Also, they offer searching for related information like product reviews and users' comments.

Nowadays, a diverse class of intelligent assistants are being emerging which are specifically called intelligent personal assistants, for example, Facebook Bots, Skype Bots, Slack Bots, Google Now, etc. These are software applications that take inputs from users in the form of text or voice to answer their queries by utilizing concepts of NLP. The response could be in the form of actions or formulating recommendations. These assistants are evolving as the smart applications over the internet as they are being deployed on major platforms like Facebook, Slack, Skype, Android, iOS, and many more, making them global [18]. Moreover, their use cases are rapidly growing in different domains like smart wearables. Considering the increase in demand of wearables along with the heavy reliance of the design of these devices on text and voice input, implies a major growth in intelligent personal assistants. In contrast to the traditional searching techniques, they leverage NLP in the form of speech recognition and semantic analysis of text, to provide users with a speech-driven or text-based question-answer system [19]. This is about intelligent assistants with speech recognition capabilities.

Considering image recognition capabilities, there are applications available which perform image analysis to classify an image based on users' input, for example, Google Goggles, SnapTell, CamFind, Slyce, etc. Google Goggles is a smartphone application which was developed by Google to perform image recognition [20]. It is capable of performing label detection, barcode identification and optical character recognition (OCR) to search similar items along with their prices. However, it does not produce successful results for categories like Apparel, Shoes, Furniture. SnapTell is one of the popular acquisitions by Amazon back in 2009 to improve the shopping experience of its mobile applications [21]. It mainly used with products like CDs/DVDs, books or games where users can upload an image of the cover of these products to find their prices and reviews from Amazon inventory. However, this application does not work well with products from categories like shoes, electronics, crockeries, cars, etc. Then, there are applications like Nokia Point & Find [22], CamFind and Slyce which provide price based comparisons of different products across different e-commerce websites.

Within e-commerce, there is an advent of intelligent assistants to provide recommendations for different products based on text and speech analysis [23]. However, intelligent assistants with image search and recognition capabilities to provide recommendations are not present. This is the area which has not been explored much. Before I go ahead and explain my implementation plan for my project, I would like to give a brief overview about recommendation systems.

A Recommendation system is a service which provides users with product information relevant to their interests to help them decide what products to purchase. It uses statistics and knowledge discovery techniques to come up with similar products [24]. It can be implied that there will

be recommendations to select from, if there are multiple people and their choices are considered as a deciding factor. In e-commerce, online vendors offer large collection of products. It becomes really difficult for the users to select from such a large inventory of products just by using a computer. Hence, vendors provide recommendations to the users by understanding and analyzing their buying patterns and purchase history. Nowadays, e-commerce vendors like Amazon and eBay have very efficient recommendation engines which provide a personalized recommendation to each user. There are basically two kinds of recommendation systems: first one is online shopping recommendation environment where, each user is recommended with a set of products based on his/her interests. Another type is web based search engine which recommends products based on a user's query from the available inventory in the form of listings. Currently used techniques to build a recommendation system are: collaborative filtering, content-based, knowledge-based, and association rules. In this project, we will integrate a web based search engine to fetch relevant product listings from e-commerce websites.

So far, we have seen how an intelligent assistant works, what are the capabilities of the existing systems, how speech and text recognition have done wonders in the field of e-commerce, and how recommendation helps users in making a decision to buy a product. Most importantly, we have studied what are the challenges users face while shopping online and how we can resolve them by improving upon the existing systems and expanding those using new technologies like image recognition. Considering the scope of the project, we are planning to build a virtual shopping assistant based on image recognition for now.

## 5. PRODUCT DESIGN

The concept of machine learning is being used in variety of application domains such as predicting diseases, stock price prediction, weather forecasting and many more. It has minimized the computation efforts to a great extent. However, there is an area in machine learning which is rapidly growing i.e., Computer Vision. It is defined as the process of extracting, analyzing and understanding valuable information from an image or a sequence of images [25]. Anecdotally, approximately 70% of the neural activity in the human brain is related to vision either directly or indirectly. An important application of Computer Vision is Image Recognition. There is extensive research being done in this field of Image Recognition, based on a maxim that an image is worth a thousand words. However, on the world-wide web, converse may be true; it is easy to perform text based search on Google as compared to a visual search using an uploaded image which may return erratic results. With the easy availability of image data and a high rise in use of mobile devices with in-built cameras and internet connection, this field of image recognition in visual search will continue to grow in future.

### 5.1 Overview

In our project, we have worked with image recognition extensively to extract features from a query image in order to perform label detection and classification. Based on the output of an image classifier, we can use our product retrieval system to provide recommendations in the form of same or similar products' listings from the e-commerce website.

In e-commerce, there are myriad of products which are sold online like clothing, electronics, shoes, etc. Considering the scope of the project, we could not consider all the items for our classification problem. Therefore, we have focused on Shoes to be our product of choice

because of its popularity among consumers and a wide-variety available in the market. It is sold on the basis of its appearance such as its color, shape, texture, etc. Its visual appeal is what makes you buy it.

In our project, our approach is to develop a virtual shopping assistant in the form of a chatbot which is based on image recognition and will perform visual search to recommend user with most similar products relevant to the input image. For instance, suppose you meet someone who is wearing a pair of fancy shoes. You really like those and want to search for them online. You just need to take a picture of those shoes, upload it on our shopping bot and you are done. You will be shown same or related shoes from our e-commerce service which you can buy at any time. In the next section, we will discuss the architecture we built to develop our virtual shopping assistant.

## 5.2 Architecture

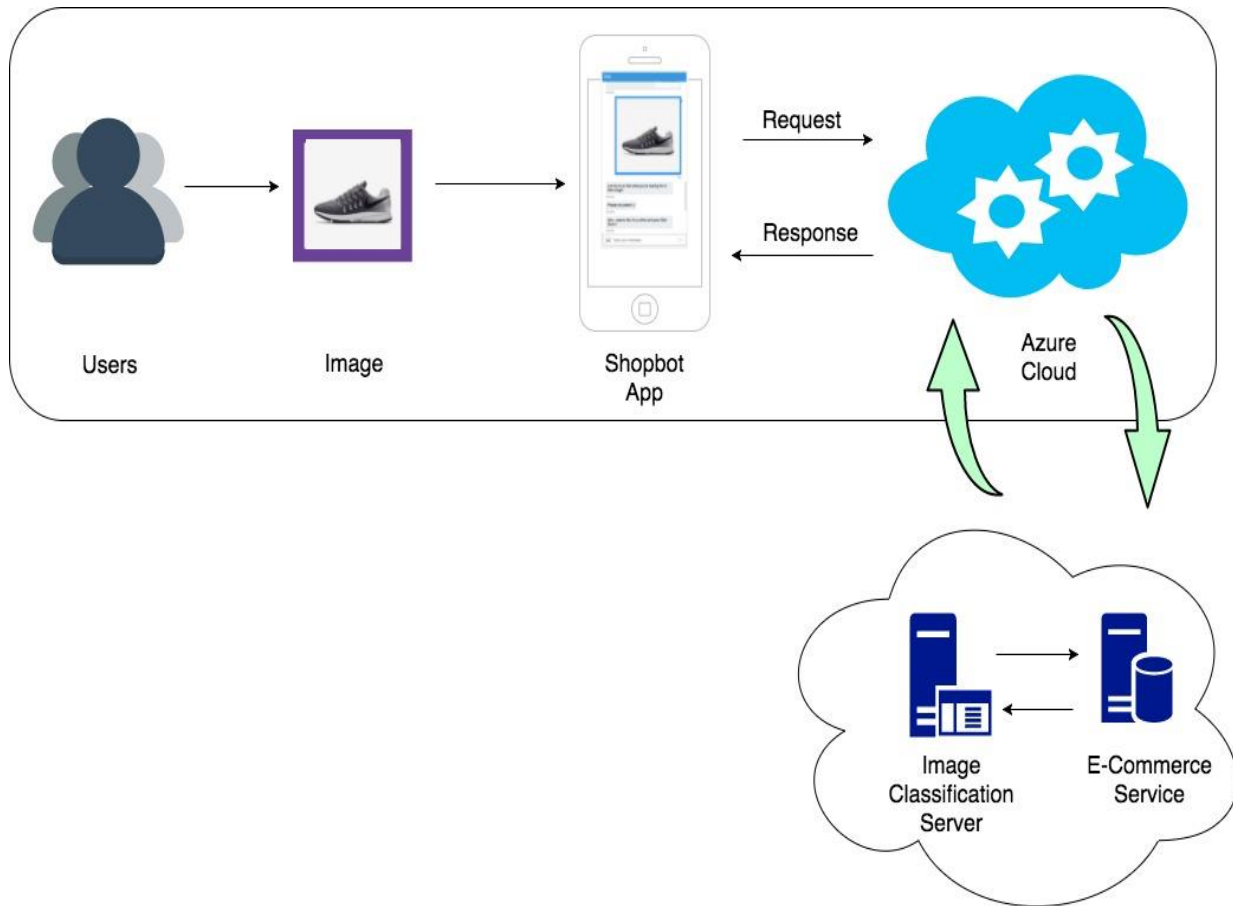


Fig. 9. Architecture of Shopbot, an Image Based Search Application

This architecture is developed in the form of a pipeline due to the presence of multiple tasks such as conversational interface, image recognition and product retrieval, which are required to be performed in a sequential manner. Hence, each of these tasks is handled individually.

The overall architecture of this project consists of the following components:

- **User Interface:** It is a necessary part of any user based application. It helps in engaging users and providing them with a great experience.

While deciding on the user interface, one question came up. For the virtual shopping assistant, should we build a chatbot or a web-based application. Anecdotally, we have found that people like to upload an image in a chat-based interface; rather than submit it to a web-based interface. Also, we feel that in this specific case of visual search, the conversational interfaces seem to be more natural.

Hence, we did research on the ways to build the conversational interfaces or chatbots.

We tried to understand the requirements we need to cater to in order to make the user experience better. After considering various options, we finally decided on Microsoft Bot Framework to build the front-end of our application. It offers various advantages such as ease of integration with Facebook, Skype, Slack, etc. along with cross-platform nature of its bots.

- **Middleware:** This layer comprises of an Image Recognition based web service. Our whole project somewhat revolves around this service, which makes it one of the main components in our project. To build this service, we require two subtasks i.e., a Dataset and an Image Classifier. These subtasks are described below in detail.

### **Dataset**

Initially, we decided to use a dataset called UT-Zap50K from Zappos.com, a well-known e-commerce website. The data was of size 260MB collected by a research group

at the University of Texas from Zappos.com. It had more than 50,000 shoes that were organized in four main categories such as boots, sandals, shoes and slippers. At first, it seemed to be a well-labelled and pruned dataset. However, random testing of this dataset showed us that it was unusable to use for our project. There were far too many limitations in the dataset. For instance, there were various incorrectly categorized images with vague labels. There was no standard in terms of image size, disproportion in the number of shoes for men and women, and the image categories were very generic with respect to our project objectives. These limitations made it clear that this dataset won't be able to train an image classifier to perform accurate recognition. Therefore, we had to look for other options.

However, with the amount of images this dataset had, we decided to stick with it as our source of shoe images. Instead of using it directly, there was a need of a specific data set per our needs. So, we chose to build our own dataset using these images by performing web scraping using web crawlers over Zappos.com. After a lot of trial and error and much efforts, we could build our dataset of approximately 970MB, consisting of more than 12,000 men shoes and 14,000 women shoes of numerous brands and categories. The interesting thing with this dataset was that each shoe item would have two key properties attached with it. One is the shoe images and other is a metadata file containing image information like brand name, color, gender and a specific category corresponding to the brand. For instance, see fig. 9 shown below.





Fig. 10. Shoe image no. 25 in the scraped dataset

```
"25": {  
    "image_url": "/dataset/images/25.png",  
    "labels": {  
        "gender": ["men", "women"],  
        "brand": "Nike",  
        "category": "low top sneakers",  
        "color": "maroon"  
    }  
}
```

Fig. 11. Metadata for the shoe shown above

This above section described how we got our dataset. Now, we will discuss the experiments we did to come up with an image recognition solution.

### **Image Processing or Classification**

We are a strong believer in open source technologies. During the past few years, these technologies have come a long way and the active contributions from the community

has been the main reason in its success. As a result, various companies have started using these technologies which will keep on increasing in coming years. Open source technologies come with a lot of benefits such as the continuous improvement in the software, zero cost, cross-platform compatibility, and many more [26].

Due to increased contributions in open source technologies, the field of Computer Vision and Image Recognition has evolved to a great extent lately [27]. Considering these advantages, we realized that working on a solution from scratch would not help us solve our problem rather than working on a solution using already existing technologies in an improved and more efficient way. In this task, we performed various experiments with the various available and recommended image recognition solutions from top-notch players like Google, Microsoft and Amazon, in order to come up with a perfect solution. As a test image for experiments, we used a shoe image shown in fig. 12.



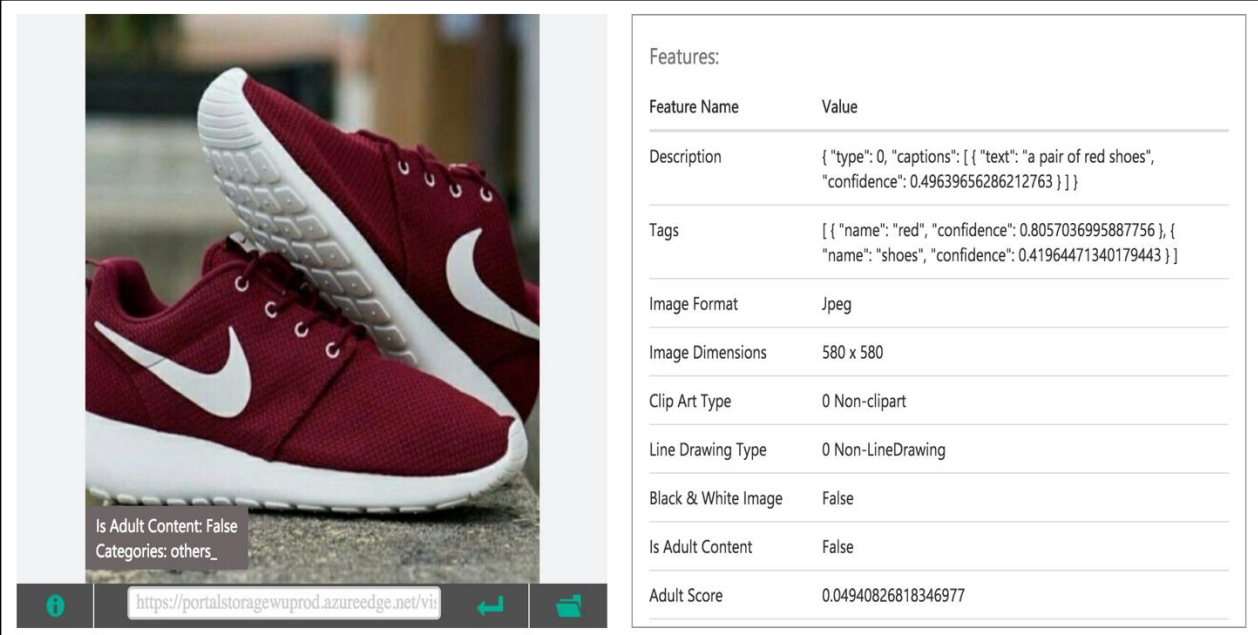
Fig. 12. Test Image for Research

The experiments performed are described below along with my findings.

- **Microsoft’s Computer Vision API**

The first experiment was performed using the solution from Microsoft Cognitive Research, also known Computer Vision API [32]. This API gives information about the content present in the input image. It performs object detection to provide descriptions and tags to classify images.

Now, let’s look at the result of classifying the test image:



Features:	
Feature Name	Value
Description	{ "type": 0, "captions": [ { "text": "a pair of red shoes", "confidence": 0.49639656286212763 } ] }
Tags	[ { "name": "red", "confidence": 0.8057036995887756 }, { "name": "shoes", "confidence": 0.41964471340179443 } ]
Image Format	Jpeg
Image Dimensions	580 x 580
Clip Art Type	0 Non-clipart
Line Drawing Type	0 Non-LineDrawing
Black & White Image	False
Is Adult Content	False
Adult Score	0.04940826818346977

Fig. 13. showing the result of Microsoft Computer Vision API

As we can see, this API returns a set of image features like description, tags, image format, images dimensions, and others. For the test image, we have got a description as “a pair of shoes” with tags like “red” and “shoes” which is certainly very generic. This is not what we really want as a result from our image recognition service. Although, there is an option of domain specific model for training, the only model available so far is of celebrities.

It certainly gave us the information related to the image content. However, the contextual accuracy was not there. We would require more specific categorization of the product. Hence, it's not really a solution for our virtual shopping assistant where we would need image classification fairly specific to shoes.

- **Google Vision API**

The second experiment was performed using a Vision API offered by Google which offers developers an image classifier with in-built capabilities like face-detection, object detection, text identification and others [33]. It rapidly classifies an image into multiple categories.

Now, let's look at the result of classifying the test image:

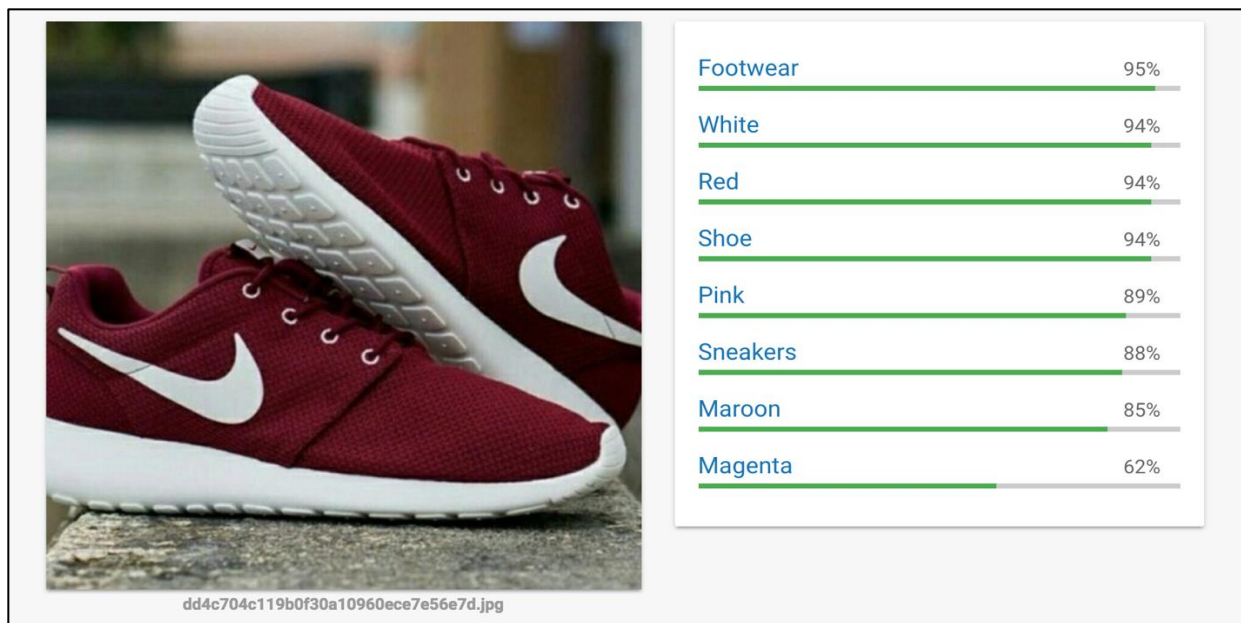


Fig. 14. showing result of Google Vision API

```
{
  "labelAnnotations": [
    {
      "mid": "/m/09j5n",
      "description": "footwear",
      "score": 0.9542664
    },
    {
      "mid": "/m/083jv",
      "description": "white",
      "score": 0.9429403
    },
    {
      "mid": "/m/06fvc",
      "description": "red",
      "score": 0.9406342
    },
    {
      "mid": "/m/06rrc",
      "description": "shoe",
      "score": 0.9392204
    },
    {
      "mid": "/m/01fklc",
      "description": "pink",
      "score": 0.8862301
    },
    {
      "mid": "/m/09kjl",
      "description": "sneakers",
      "score": 0.88332796
    }
  ]
}
```

Fig. 15. showing the JSON response of the categories with average classification score.

Thus, this API returns image classification in multiple categories along with colors breakdown and classification score. For the test image, we have got result like footwear, white, red, shoe, pink, sneakers, maroon, magenta. This is slightly better than what we got from Microsoft Computer Vision API. However, still it is not something we could use to recommend similar products from our e-commerce inventory.

- **Amazon AWS Rekognition**

The third experiment was performed with Amazon's offering in Image Recognition [34]. This API provides you with capabilities like object detection, scene identification, face recognition and it also enables developers to do search

and comparison based on faces. It uses state of the art deep learning based machine learning models which can be utilized to perform visual search and image classification.

Now, let's look at the result of classifying the test image:

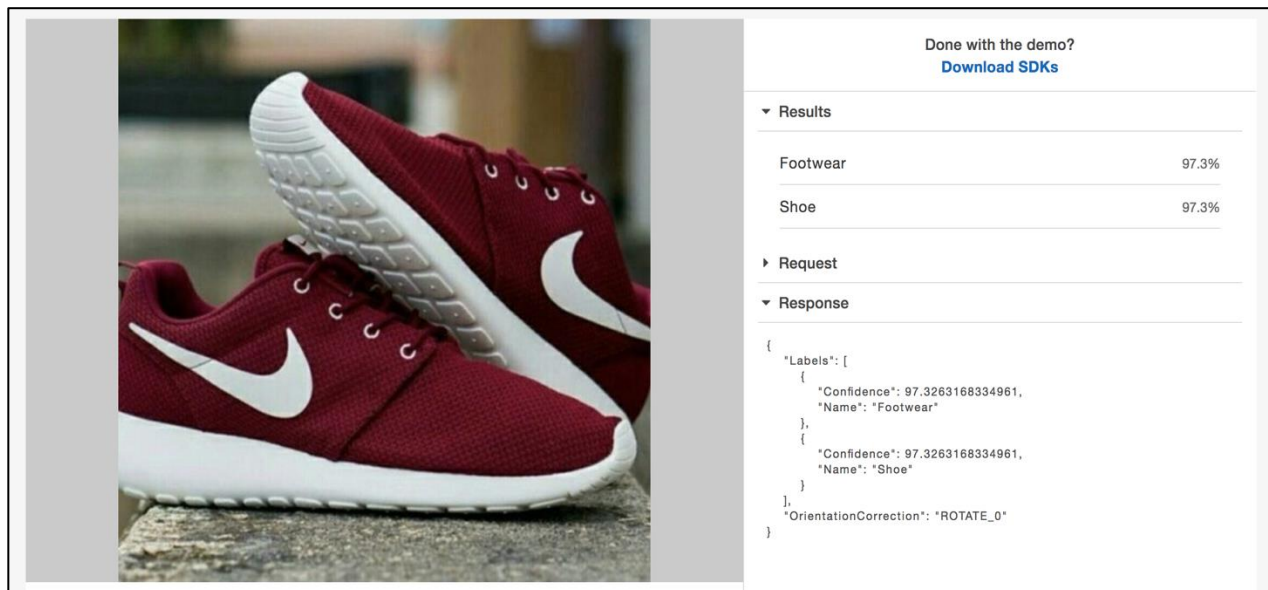


Fig. 16. showing result of Amazon AWS Rekognition API

This API returns result in the form of categories based on the input image along with the confidence score of the labels. For the test image, we have got labels like footwear and shoe. This result is very similar to the Google's API.

Till now, these were the APIs that we tried working with, which could not solve our problem because of their generic results.

After doing some more research, we decided to approach this problem in a different way. Instead of using the existing technologies directly, we thought that it might be a good idea to “train” an image recognition solution using a known set of image dataset with images and their corresponding labels and descriptions such as a shoes’ catalogue.

So, we looked out for an image recognition solution which offers deep learning models with pre-existing capabilities like color detection and object detection and enables us to “train” those models on our own image data. During our research, we found an API from CloudSight Inc. which offers exactly we were looking for. Below is the detailed description of our experiment with this API.

- **CloudSight API**

This is an API offered by CloudSight Inc. which runs on Amazon Elastic Container Service having Nvidia/CUDA images running within Docker Containers [29]. It provides us with deep learning models with in-built capabilities like color detection, face recognition, etc. And, it gives us the capability to train its image classifiers on outside dataset. This is what we were looking for. Hence, we trained these models on our scraped dataset of shoes to come up with more specific image classification. Now, let’s look at the result of the modified classifier trained on our scraped dataset:

```
{
  "token": "ULIw70p9RHHWdR73cg4v9A",
  "url": "http://assets.cloudsight.ai/uploads/image_request/image/208/6/208336695/dd4c704c119b0f30a10960ece7e56e7d.jpg",
  "ttl": 54,
  "status": "completed",
  "gender": ["men", "women"],
  "color": ["red", "white"]
  "brand": ["Nike"]
  "category": ["shoes", "sneakers", "low top sneakers"]
}
```

Fig. 17. shoes results from CloudSight API

So, finally we have achieved our goal. For the test image, you can see all the parameters like brand name, shoe categories, color breakdown and gender in the response. Our idea worked. Now, we can go ahead and generate a product description using our script to show to the user. For instance, we would generate a description based on the API's response like "a red and white Nike low top sneakers". Although, we achieved what we were looking for, we had a doubt that if big players like Google, Amazon and Microsoft could not classify the test image with good accuracy, how could this API be successful with a good amount of accuracy. We found out that it was possible because of the training of classifiers on a more specific dataset that we provided. We trained the models on a dataset with specific details like brand name, categories, etc.

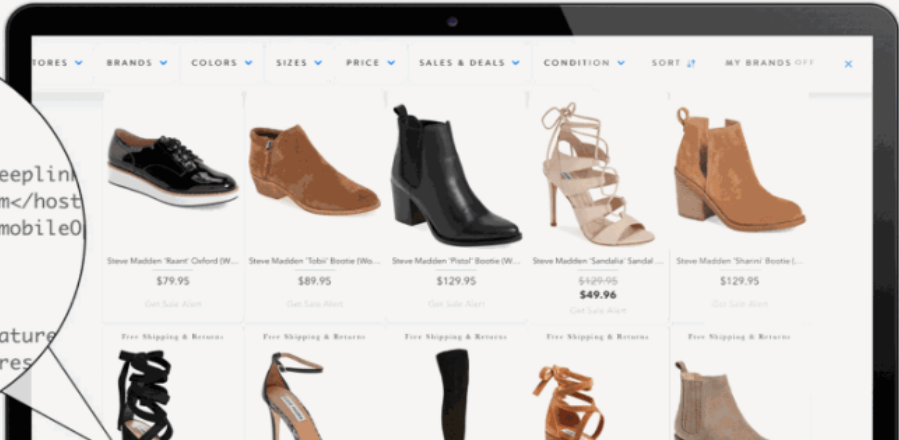


So, overall we have found a solution to perform image based classification for the shoe images we will get from the user-interface. As a result, this layer will provide an output in the form of multiple labels and keywords to the backend layer which is described in the next section

- **Backend:** This would handle a product recommendation service. Its role is to take the output from the middleware layer in the form of labels and keywords and use them to produce same or similar listing of products from our e-commerce inventory. For our back-end service, initially we considered Amazon or eBay as our source inventory. However, we found out that there were no open-source APIs available for our use case where we would get access to their inventory to perform search based on keywords. Also, we preferred an e-commerce service which is more specific to fashion products particularly shoes. Hence, we found Shopstyle. It is a fashion marketplace widely used online by consumers. It has listings of over 10 million products across shoes, apparel, beauty products, etc. from multifarious vendors across the world. Fig. 18 shows the Shopstyle product API [35].

## ACCESS OUR PRODUCT API

Building a shopping platform? Use the same API that powers ShopStyle to access thousands of global brands and millions of products. [Click here](#) for more information.



```

</id>
</id>
<name>Nordstrom</name>
<deeplinkSupport>true</deeplinkSupport>
<hostDomain>nordstrom.com</hostDomain>
<mobileOptimized>>false</mobileOptimized>
</logo>
<sizes>
<featured>
<sizeName>feature
<url>http://res
</featured>
</default?x>

```

Fig. 18. showing Shopstyle Product API

Shopstyle has partnerships with certain third-party companies which use web crawlers to collect data in the form of images and affiliate links and provide this data to Shopstyle in the form of JSON and XML feeds. Therefore, using Shopstyle as our source inventory not only gives us a large collection of shoes to perform our search, but also an affiliate program which we can use to buy any product as well. Based on the keywords that we provide in a simple HTTP GET request, we receive a list of similar products relevant to the search in JSON format, and we show these product listings in the form of a carousel in the Shopbot along with details like price, brand name, product image and its affiliate links so that user can visit the product page and make a purchase. Fig. 19 shows the result of the HTTP GET request.

```

{
+ metadata: {...},
- products: [
  - {
    id: 608436443,
    name: "MSGM Fall 2015 Sleeveless Top",
    brandedName: "MSGM Fall 2015 Sleeveless Top",
    unbrandedName: "Fall 2015 Sleeveless Top",
    currency: "USD",
    price: 265,
    priceLabel: "$265",
    inStock: true,
+ stock: [...],
+ retailer: {...},
+ brand: {...},
    locale: "en_US",
    description: "Look 4 of the Fall 2015 Collection. Fuchsia MSGM sleeveless top featuring mock neck, oversize bow accent at front, dual seam pockets closure.",
    clickUrl: "https://api.shopstyle.com/action/apiVisitRetailer?id=608436443&pid=uid1889-37128833-30",
    - image: {
      - sizes: {
        - Small: {
          sizeName: "Small",
          url: "https://img.shopstyle-cdn.com/pim/68/f9/68f98d8810fcf4a72495a5aa0086d2ae_small.jpg",
          width: 32,
          height: 40,
          actualWidth: 21,
          actualHeight: 40
        },
        - XLarge: {
          sizeName: "XLarge",
          url: "https://img.shopstyle-cdn.com/pim/68/f9/68f98d8810fcf4a72495a5aa0086d2ae_xlarge.jpg",
          width: 328,
          height: 410,
          actualWidth: 220,
          actualHeight: 410
        },
        - Medium: {
          sizeName: "Medium",
          url: "https://img.shopstyle-cdn.com/sim/68/f9/68f98d8810fcf4a72495a5aa0086d2ae_medium/msgm-fall-2015-sleeveless-top.jpg",
          actualWidth: 220,
          actualHeight: 410
        }
      }
    }
  }
]
}

```

Fig. 19. showing the response of HTTP GET request to Shopstyle API

In the above section, we have seen the architecture of the Shopping bot along with its different components. Now, we will see the implementation steps of the Shopping bot.

### 5.3 Implementation

Before we started with the actual implementation, we firstly decided on the tools and technologies we should use for the implementation. The technological stack is stated below:

- **User Interface:** the front-end interface of the Shopbot is developed using the Microsoft Bot Framework. It is an open source SDK for building conversational interfaces, also known as chatbots. It offers you many benefits over other frameworks available. For instance, its self-scaling capability based on users' traffic and Bot Connector Service which allows you to connect to multiple social platforms like Facebook, Slack, etc. with a flip of a switch.
- **Image Recognition:** After doing an extensive research on the existing technologies in this domain, we decided to utilize the open-ended Cloud Sight APIs which provided us with pre-defined classifiers which could be trained again on new datasets to improve performance and accuracy.
- **Web Services:** we develop and utilize various REST APIs as part of this project to ensure seamless interactions across the platform. For instance, one of the APIs is Shopstyle API.
- **Programming Languages:** C# is used for developing our front-end and Python is used for training our machine learning classifier.
- **Cloud Services:** Microsoft Azure is used to deploy the Shopbot to make it readily available to the users. It is a cloud service offered by Microsoft to build, manage and deploy applications. We have used its platform as a service to deploy our Shopping bot. It also provides reliable and secure access to our applications' data. Azure automatically balances the network load to make your application run seamlessly.
- **Hardware Resources:** Apple MacBook Pro having 8 cores CPU with RAM of 16 GB and Nvidia GT graphics card, is used for development.

Now, we will discuss various steps that are required to be implemented to build an Image Recognition based Virtual Shopping Assistant. These steps are as follows:

- **Image Upload:** The Shopbot asks user to enter an image. The image can be of formats like .png and .jpeg.

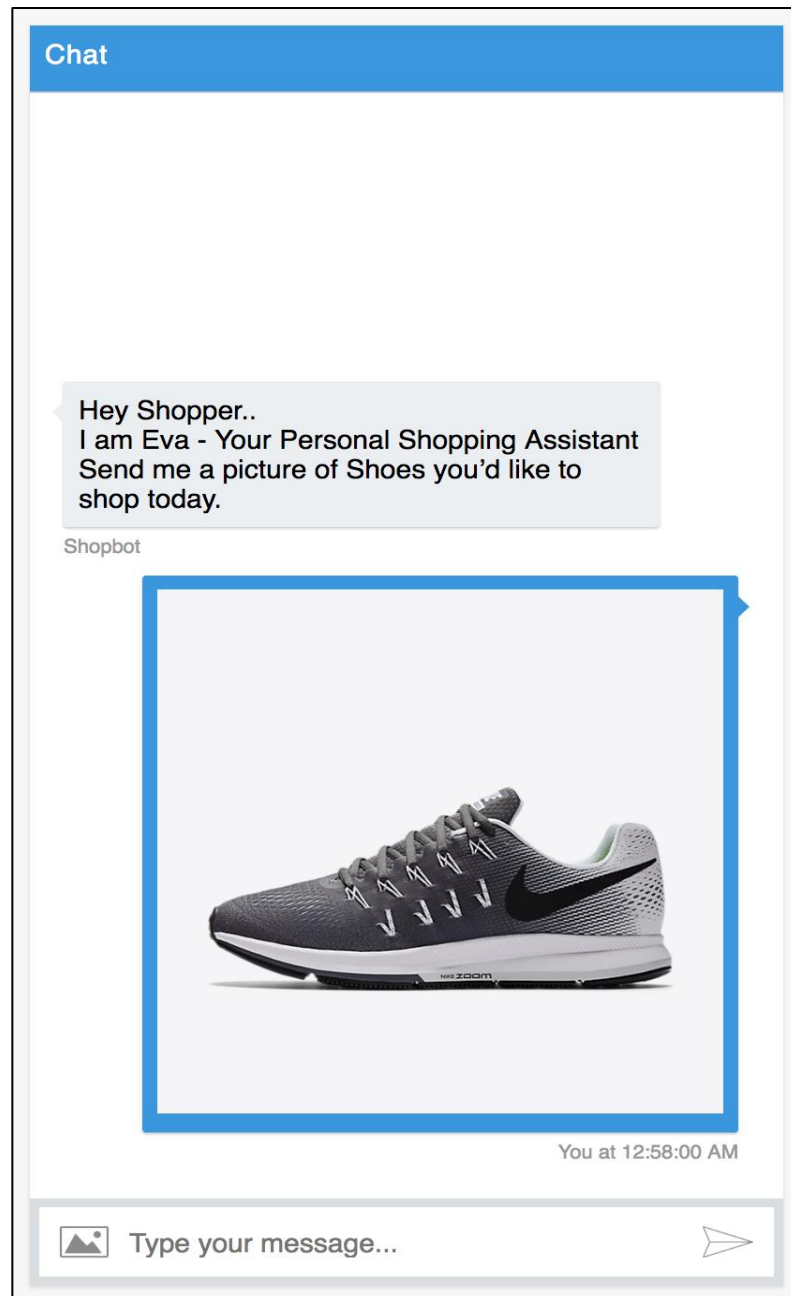


Fig. 20. showing an image uploaded by an user

- **Classification:** Once the user uploads an image, it goes to an Image Recognition based REST service which processes it and returns with various product features like color, category, brand, etc. The Shopbot creates a description of the product by using these features it finds in the image.

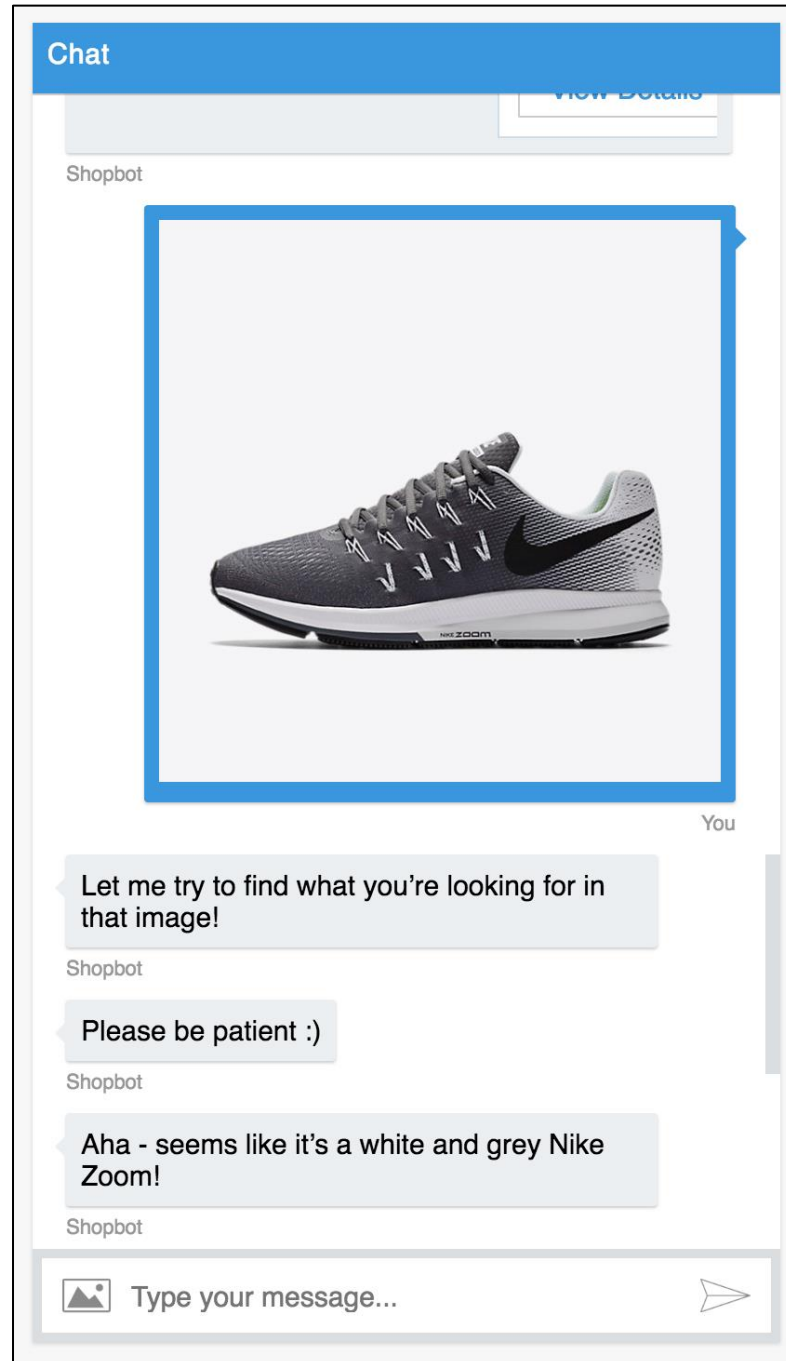


Fig. 21. showing description of the query image

- **Retrieval:** Once we have the features for a specific product, we use our back-end service to send these to the Shopstyle API which returns us with similar products, if present in the inventory.

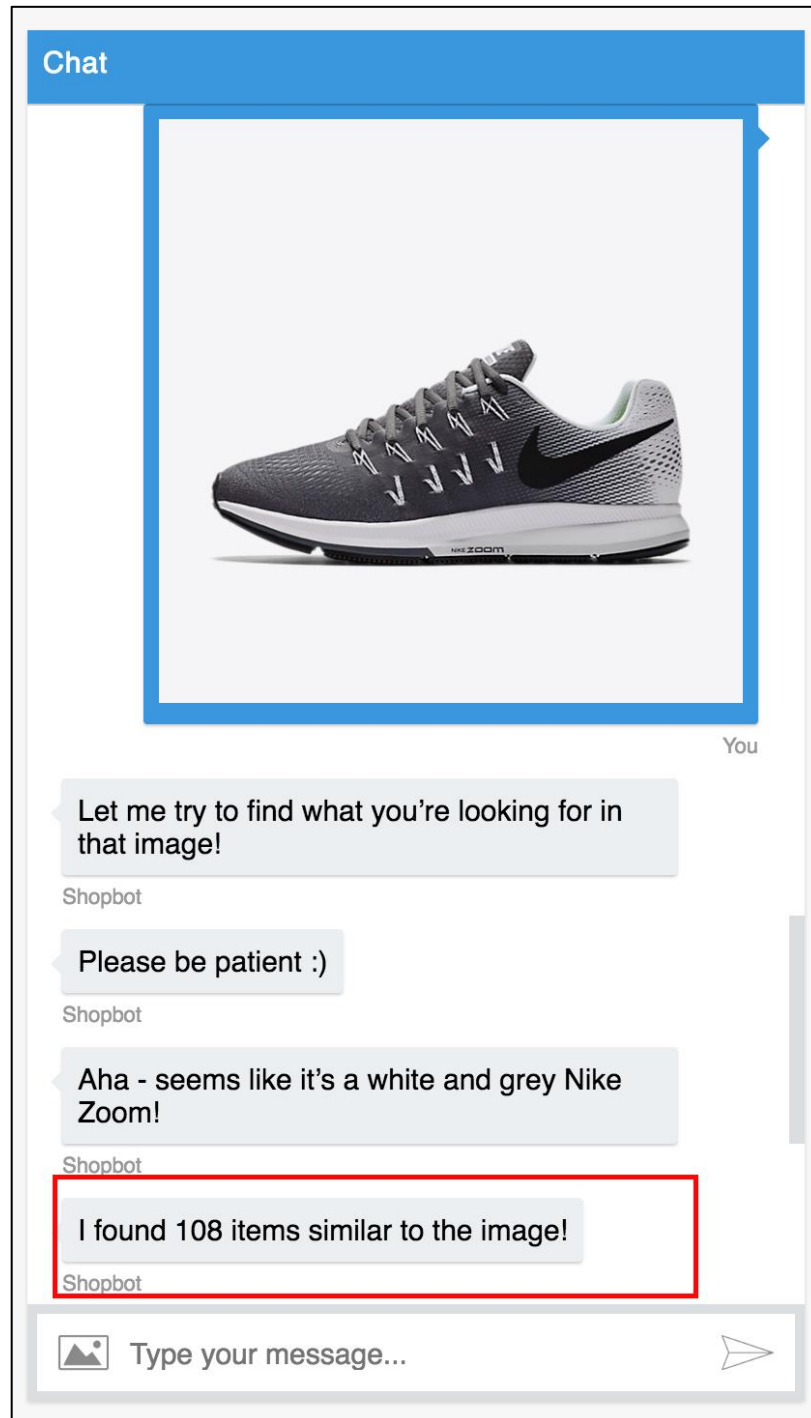


Fig. 22. showing a count of the similar products

- **Product Listings:** The similar products received from back-end service is organized in a carousel in the form of Hero cards with information like product image, price and affiliate links. This carousel is then returned to the Shopbot. If user taps on any item he likes, they will be redirected to that item's webpage to make the purchase.

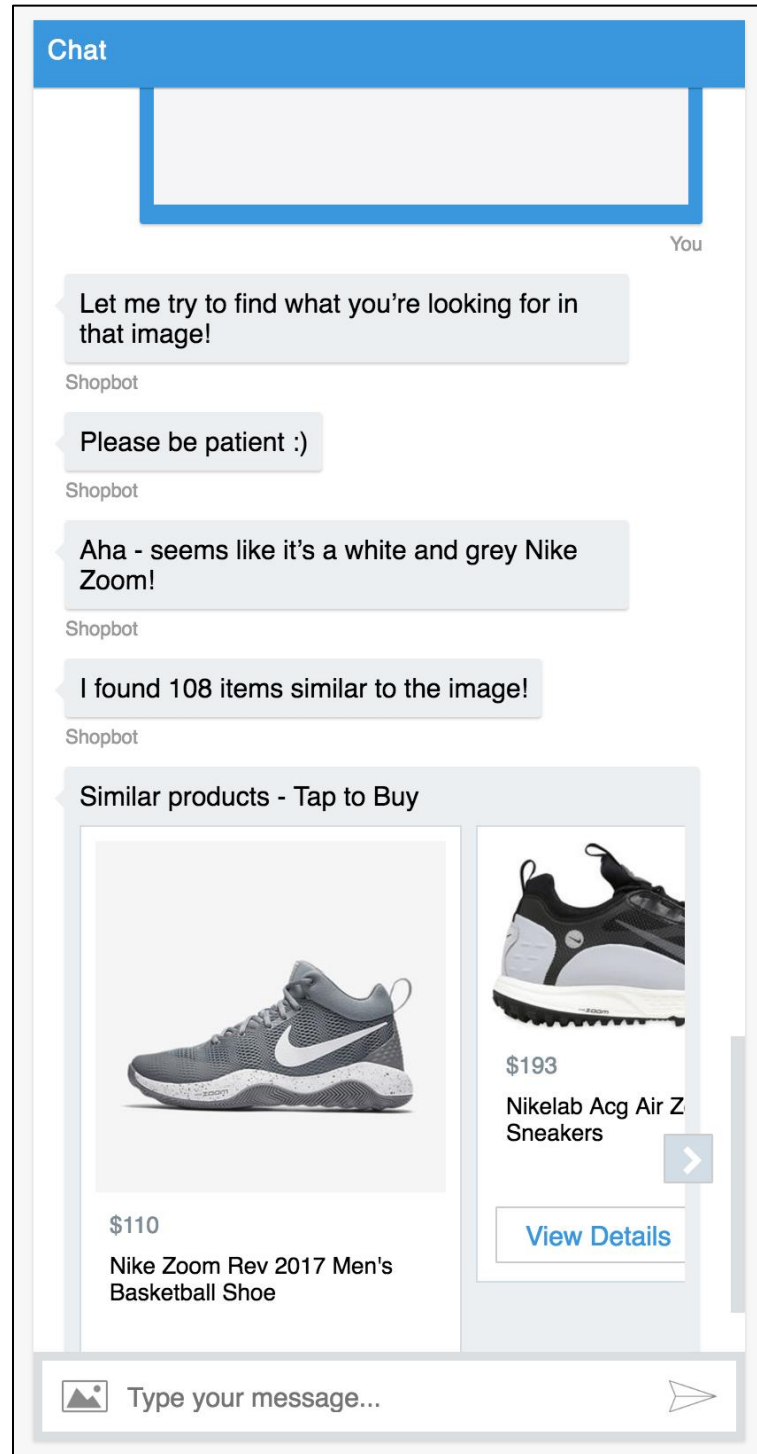


Fig. 23. showing a carousel of products



- **Make a purchase:** Based on the retrieved results from the inventory, user can now access the top similar items to its queried product and visit the selected product page by clicking the **View Details** button to gain more information about the product like its features or applicable discounts.



Fig. 24. Forwarding to the actual product webpage

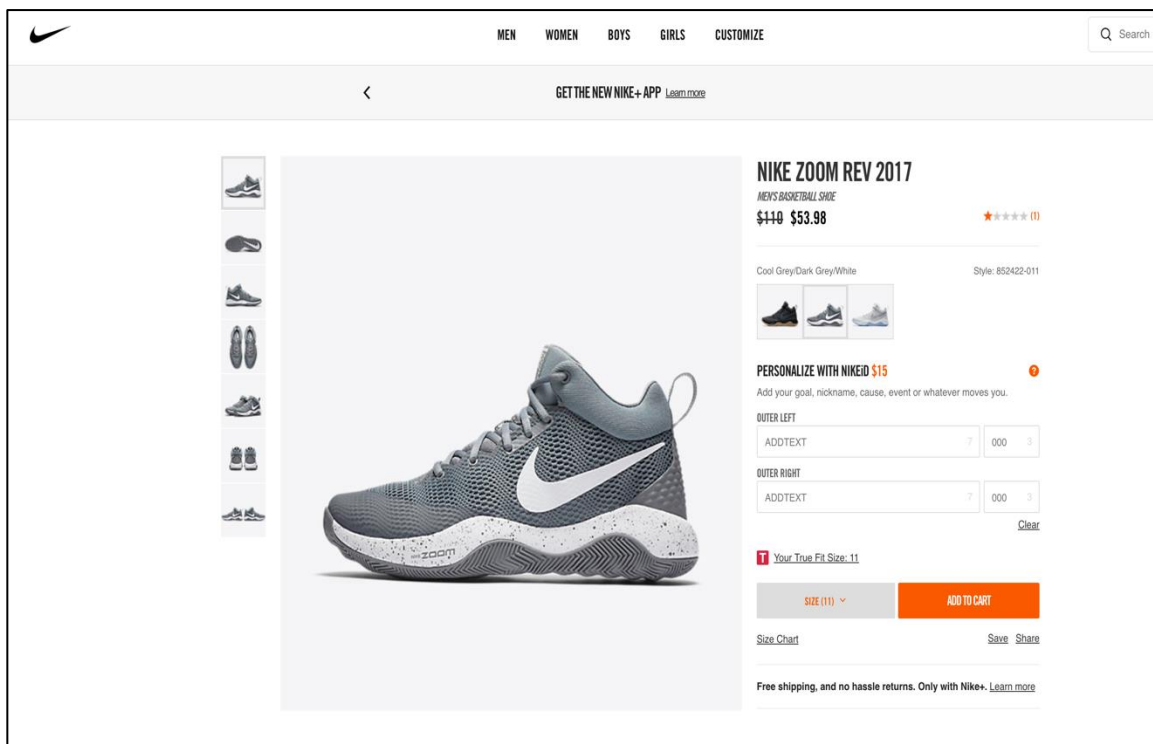


Fig. 25. Showing actual product page on Nike's website

## 6. EXPERIMENTAL RESULTS

In previous section, we discussed about the architecture of our Shopping bot and its underlying implementation steps. In this section, we will summarize the results of our experiments that we performed to develop the image recognition and product retrieval services.

Like I explained during the implementation phase how I carried out various experiments with existing machine learning models for building the image recognition service. Most of the models gave very generic results in terms of finding the image features. The image used for experiments is shown in fig. 11 and the results are shown below in Table 1.

<b>Image Recognition API</b>	<b>Average Confidence Score</b>	<b>Results</b>	<b>Machine Learning Models with Re-training Ability</b>
Microsoft Vision API	49.6%	a pair of shoes	No
Google Cloud Vision API	91.4%	footwear, shoes, white, red, pink	No
Amazon Rekognition	96.3%	footwear, shoes	No
CloudSight API	97.95%	sneakers, shoes, red, white	<b>Yes</b>

Table 1. showing comparison of existing image recognition services

We got very similar results using all the above image recognition services. Each API gave us generic results in terms of image classification, for example, footwear, shoes, red, white, etc. These results were not specific enough to be considered for our image recognition use case. However, there was one difference among these services i.e., CloudSight API which offers pre-trained

machine learning models with existing capabilities which can be trained again on any domain-specific dataset. Due to this special feature, we decided to move forward with this image recognition service.

We trained the existing image classification model provided by CloudSight using our dataset. We experimented with different split ratios of training and testing data to figure the best model accuracy. The results comparison is shown in Table 2.

<b>Pre-trained/ Re-trained Classifier</b>	<b>Split Ratio (Training: Testing)</b>	<b>Model Accuracy (%)</b>	<b>Training time (seconds)</b>
Pre-trained	-	96.2%	-
Re-trained on Scraped Dataset	60:40	93.1%	~726
	70:30	93.8%	~754
	80:20	95.2%	~809
	<b>90:10</b>	<b>96.8%</b>	<b>~ 861</b>

Table 2. showing model accuracy based on various split-ratios of training and testing data

After this experiment, we found out the best model accuracy could be achieved by training the machine learning model on 90:10 ratio of training and testing image data. Hence, we followed this approach. Once the model got trained, we proceeded with building a RESTful API which would be used directly within our application. Then, we performed actual testing of this service using multiple query images of different brands, shapes, categories and colors selected randomly from

the internet to test the classification accuracy of our image recognition service. The results are as follows:






Query Image	Approx. Time taken to Classify an Image	Results (brand, category, colors, gender)	Confidence Score
	5 seconds	<ul style="list-style-type: none"> <li>• Men</li> <li>• Brown</li> <li>• Leather Lace Up</li> <li>• Shoes</li> </ul>	97.6%
	3 seconds	<ul style="list-style-type: none"> <li>• Men and Women</li> <li>• Asics</li> <li>• Grey and Blue</li> <li>• Athletic Shoes</li> </ul>	98.7%
	3.5 seconds	<ul style="list-style-type: none"> <li>• Women</li> <li>• Red</li> <li>• Suede Stiletto</li> <li>• Shoes</li> </ul>	98.8%
	2.4 seconds	<ul style="list-style-type: none"> <li>• Men and Women</li> <li>• Nike</li> <li>• Grey and Black</li> <li>• Running Shoes</li> </ul>	98.6%
	4.4 seconds	<ul style="list-style-type: none"> <li>• Women</li> <li>• Black and Beige</li> <li>• Boots</li> <li>• Shoes</li> </ul>	98.9%

Table 3. showing results returned by the image recognition service

Based on the above shown confidence scores and time taken in image classification, our image recognition service resulted in 98.7% of accuracy with approximately 3.5 seconds of processing time to recognize a specific image.

Similarly, we performed experiments using our product retrieval service as well. Since, we did not use any similarity metrics directly to come up with relevant products. We tried to find out the accuracy of results based on visually similar images. In the below table, you can see top relevant products returned by the backend service based on the labels generated by image recognition phase for each query image.

Query Image	Relevant Products			
				
				
				
				
				

Table 4. showing relevant results returned from the backend service

As shown in Table 4, the retrieved products seem very similar to the query images. However, there may be some instances when you would get different products or no products at all for a specific image because of non-availability of a specific pair of shoes on Shopstyle.

From the above experiments, we can derive that this shopbot application achieved desirable results.

In the next section, we will summarize our research and implementation work.

## **7. CONCLUSION**

In this research, we have tried to find out an answer to our research question of building a virtual shopping assistant using state of the art image recognition technologies to improve the overall online shopping experience for the users. We have proposed a novel approach of providing relevant product listings to the users based on visual search. The proposed application has three main components, i.e., chatbot based user interface, image recognition service and a product retrieval service. Each of these components is an important part of this application. However, image recognition service is the core functionality of the application which enables users to extract meaningful information from the images which are hard to describe in words, for example, categories, brand name, color features of a product.

With the extensive use of social media sites like Facebook, Twitter, people have become very keen to know about the trending things happening around them. They look to follow various people, for example their favorite celebrities. Their purchasing decisions get influenced by social media a lot. So, having a product search application to find relevant products by providing just an image is very empowering. Now, users have a better way of finding products perfect for them. They are no longer required to go to traditional text-based search platforms. Instead, they can allow this application to find relevant products for them. All they need to do is provide an image of the

product they are looking for.

While working on this project, we understood the importance of the dataset required to train the machine learning model in order to achieve results of greater accuracy and precision. After performing various experiments, we realized that having a good quality data with more granularity is far more important than having just a large dataset of low quality and down-sampled images, especially for our use case where we are trying to classify shoes based on images. Building our own dataset with features like brand names and more granular shoes categories, have certainly helped us in achieving quality results and high performance.

In product retrieval service, we have used only one e-commerce API i.e., Shopstyle to retrieve similar products. Although it provides satisfactory results, having more number of e-commerce inventories integrated within this service will surely increase the listings of relevant products and thus, its performance. For instance, a user can specify his/her favorite e-commerce portal to retrieve relevant products.

The experiments carried out in this project manifest the power of Computer Vision and Machine Learning. The results show the benefits of having a Visual or Image based search application which will bridge the communication gap between humans and computers when it comes to finding relevant products online based users' real product intentions. We believe that this application is a good starting point in taking the user shopping experience to the next level altogether. We hope to continue refining this solution to make it more efficient and effective for the users.

## 8. FUTURE WORK

Based on our research, we have developed a fully-functional image based search application with satisfactory results. However, due to the limited scope of the project, we could not refine it to achieve higher accuracy and performance. So, as part of the future work, we would like to take it forward by refining it and building upon it by adding more functionalities. Some of them are defined below:

- **Training Data Set:** We have seen how important a training data set can be for building an accurate image classification model. During our experiments with the pre-trained classifier, we saw the generic nature of results which could not be used as part of the solution. Hence, having a specific dataset with well-defined features is of paramount importance. Furthermore, the dataset must contain data for multiple products and not only shoes to build an image recognition service which can classify different items available on e-commerce. To build this kind of dataset, we would perform data wrangling on multiple datasets to come up with a collective data source comprising of multiple products.
- **Social Media Integration:** To increase user engagement with this application, we would require to publish it on various social media sites like Facebook Messenger, Slack, Skype, etc. as real world users are the ones who would help improve this application by providing their valuable feedback.
- **E-commerce Inventories:** Having a standalone e-commerce inventory for retrieving relevant product items, confines users' purchasing options. For instance, if a user wants to



buy a pair of jeans, he will be restricted to the options currently available on that inventory. They won't have the option to choose their favorite shopping portal. So, we would integrate support for various other shopping portals and not just one. This will certainly enhance the user shopping experience.

- **Multiple Object Recognition:** Based on the problem that we are trying to solve, having a multi-object recognition technology is inevitable. It is very unlikely that user would use an image showing only a single object to perform search. So, performing an image segmentation based on multiple objects is what we would work on next.
- **Hardware Resources:** Performing image processing at a brisk rate requires more hardware resources along with their optimized utilization. In this project, we have used an 8 core CPU machine with 16GB of RAM. However, building and training a Convolutional Neural Network based on images needs a lot more resources. So, we would evaluate the current processing of images per second and will optimize the computational results by incorporating a GPU based architecture.
- **Natural Language Processing (NLP):** This concept is being highly used in user-based application. So, along with image based processing, we would integrate NLP services within this application to offer users a holistic online shopping experience. Then, users would be able to use either speech, text or image based search to retrieve their favorite products from various e-commerce inventories.

## 9. REFERENCES

- [1] Christopher C. Yang, S. H. Kwok, and Milo Yip, "Image Browsing for Infomediaries," in Proc. 35th Hawaii International Conference on System Sciences, vol. 14, 2002. [Online]. Available: <http://ieeexplore.ieee.org/document/994209/>
- [2] Neil Baptista, "Chatbot and Conversational UI," December 2016. [Online]. Available: <https://medium.com/chat-bots-weekly/chatbot-conversational-ui-start-here-2f9250e8cde0#.wrqxofor9>
- [3] Rizwan Ahmed, "Facebook's New Chatbot Platform: The Future of Ecommerce Business," July 2016. [Online]. Available: <http://www.cyberockk.com/2016/07/facebooks-new-chatbot-platform-future.html>
- [4] Siddharth Gupta et al, "An E-Commerce Website based Chatbot," International Journal of Computer Science and Information Technologies, vol. 6 (2), 2015, 1483-1485. [Online]. Available: <http://ijcsit.com/docs/Volume%206/vol6issue02/ijcsit20150602125.pdf>
- [5] Angela Sokolovska, "From E-Commerce to Conversational Commerce: Chatbots and Virtual Assistants," August 2016. [Online] Available: <http://www.guided-selling.org/from-e-commerce-to-conversational-commerce/>
- [6] S. A. A. Kader, John Woods, "Survey on Chatbot Design Techniques in Speech Conversation Systems," International Journal of Advanced Computer Science and Applications, vol. 6, No. 7, 2015.
- [7] Tim Morris, *Computer Vision and Image Processing*. Palgrave Macmillan (2005).
- [8] O. Russakovsky et al., "ImageNet Large Scale Visual Recognition Challenge", 2014.

- [9] King-sun Fu, Azriel Rosenfeld, "Pattern Recognition and Image Processing," IEEE Transactions on Computers, vol. C-25, no. 12, December 1976.
- [10] E. Roy Davies, *Machine Vision: Theory, Algorithms, Practicalities*. Morgan Kaufmann. 2005.
- [11] "ELIZA - Wikipedia, the free encyclopedia.," [Online]. Available:  
<http://en.wikipedia.org/wiki/ELIZA>.
- [12] J. Weizenbaum, "ELIZA - A Computer Program for the study of Natural Language Communication between Man and Machine," Communications of the ACM, vol. 9, no. 1, pp. 53-62, January 1966.
- [13] "A. L. I. C. E. The Artificial Linguistic Internet Computer Entity," A.L.I.C.E. AI Foundation, Inc., [Online]. Available: <http://www.alicebot.org/about.html>
- [14] "CleverBot - Wikipedia, the free encyclopedia.," [Online]. Available:  
<http://en.wikipedia.org/wiki/Cleverbot>.
- [15] "LSA-Bot", [Online]. <http://www.slblabs.com/projects/lisabot>
- [16] Inktomi Corporation, [Online] Available: <http://www.inktomi.com/>
- [17] mySimon, [Online] Available: <http://www.MySimon.com/>.
- [18] Johann Hauswald et al., "Sirius: An Open End-to-End Voice and Vision Personal Assistant and Its Implications for Future Warehouse Scale Computers," In *Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems*. ACM, New York, USA, 223-238. [Online] Available:  
<http://dx.doi.org/10.1145/2694344.2694347>

- [19] Marti A. Hearst. 'Natural' Search User Interfaces. *Commun. ACM*, vol. 54, No. 11, pp 60–67, November 2011. [Online]. Available: <http://dx.doi.org/10.1145/2018396.2018414>
- [20] Google Goggles, "Goggles overview and requirements". [Online]. Available: <https://support.google.com/websearch/answer/166331?hl=en>.
- [21] SnapTell, "Image Recognition Startup SnapTell Acquired by Amazon Subsidiary A9.com". [Online]. Available: <http://techcrunch.com/2009/06/16/image-recognition-startup-snaptellacquired-by-amazon-subsidiary-a9com/>.
- [22] Nokia Point & Find, "Nokia Point & Find - Wikipedia, the free encyclopedia". [Online]. Available: [http://en.wikipedia.org/wiki/Nokia\\_Point\\_%26\\_Find](http://en.wikipedia.org/wiki/Nokia_Point_%26_Find).
- [23] J. Ben Schafer, Joseph Konstan, and John Riedl., "Recommender systems in e-commerce," *In Proceedings of the 1st ACM conference on Electronic commerce*. ACM, New York, USA, 1999, 158-166. [Online]. Available: <http://dx.doi.org/10.1145/336992.337035>
- [24] J. Ben Schafer, Joseph A. Konstan, and John Riedl, "E-Commerce Recommendation Applications," *Data Mining and Knowledge Discovery*. 5, 1-2, January 2001, 115-153. [Online]. Available: <http://dx.doi.org/10.1023/A1009804230409>
- [25] The British Machine Vision Association and Society for Pattern Recognition. [Online]. Available: <http://www.bmva.org/visionoverview>
- [26] Advantages of Open Source Software. [Online]. Available: <http://www.cioinsight.com/it-strategy/application-development/slideshows/nine-advantages-of-open-source-software.html#sthash.9EHXfaio.dpuf>

- [27] Visually Intelligent Bots – The Future of Fashion Personalization. [Online]. Available: [http://  
http://www.iamwire.com/2016/09/visually-intelligent-bots-the-future-of-fashion-  
personalisation/141095](http://http://www.iamwire.com/2016/09/visually-intelligent-bots-the-future-of-fashion-personalisation/141095)
- [28] Brad Folkens, “What is Visual Cognition?” [Online]. Available: [http://  
https://blog.cloudsight.ai/what-is-visual-cognition-85e3086af298](http://https://blog.cloudsight.ai/what-is-visual-cognition-85e3086af298)
- [29] Brad Folkens, “Deep Learning Recognition Using GPUs in Amazon ECS Docker Containers,” 2016. [Online]. Available: [http://  
https://blog.cloudsight.ai/deep-learning-image-recognition-  
using-gpus-in-amazon-ecs-docker-containers-5bdb1956f30e](http://https://blog.cloudsight.ai/deep-learning-image-recognition-using-gpus-in-amazon-ecs-docker-containers-5bdb1956f30e)
- [30] Microsoft Bot Framework Documentation. [Online]. Available: [http://  
https://docs.botframework.com/en-us/csharp/builder/sdkreference/](http://https://docs.botframework.com/en-us/csharp/builder/sdkreference/)
- [31] Andrej Karpathy. *Convolutional Neural Networks*. 2015. [Online]. Available: [http://  
cs231n.github.io/convolutional-networks/](http://cs231n.github.io/convolutional-networks/).
- [32] Microsoft Computer Vision API Documentation. [Online]. Available: <https://docs.microsoft.com/en-us/azure/cognitive-services/Computer-vision/Home>
- [33] Google Cloud Vision API Documentation. [Online]. Available: <https://cloud.google.com/vision/docs/>
- [34] Amazon Rekognition Documentation. [Online]. Available: <https://aws.amazon.com/documentation/rekognition/>
- [35] Shopstyle Product API Documentation. [Online]. Available: <https://www.shopstylecollective.com/api/product>

## 10. APPENDIX

This section consists of snippets of the source code we wrote to implement our project. The programming language is C# and the development environment is Visual Studio.

The main source code is divided into four main parts which are displayed below.

- Message Controller

```
using System;
using System.Linq;
using System.Net;
using System.Net.Http;
using System.Threading.Tasks;
using System.Web.Http;
using System.Web.Http.Description;
using Microsoft.Bot.Connector;
using Newtonsoft.Json;
using Microsoft.Bot.Builder.Dialogs;

namespace Shopbot
{
    [BotAuthentication]
    public class MessagesController : ApiController
    {
        /// <summary>
        /// POST: api/Messages
        /// Receive a message from a user and reply to it
        /// </summary>
        public virtual async Task<HttpResponseMessage> Post([FromBody]Activity activity)
        {
            // check if activity is of type message

            if (activity.Type == ActivityTypes.Message)
            {
                await Conversation.SendAsync(activity, () => new ShopbotDialog());
            }
            else
            {
                ConnectorClient connector = new ConnectorClient(new Uri(activity.ServiceUrl));
                var reply = HandleSystemMessage(activity);

                if (reply != null)
                {
                    await connector.Conversations.ReplyToActivityAsync(reply);
                }
                return new HttpResponseMessage(System.Net.HttpStatusCode.Accepted);
            }
        }
    }
}
```

```
private Activity HandleSystemMessage(Activity message)
{
    if (message.Type == ActivityTypes.DeleteUserData)
    {
        // Implement user deletion here
        // If we handle user deletion, return a real message

    }
    else if (message.Type == ActivityTypes.ConversationUpdate)
    {
        // Handle conversation state changes, like members being added and removed
        // Use Activity.MembersAdded and Activity.MembersRemoved and Activity.Action for info
        // Not available in all channels

        if ((message.MembersAdded[0].Name == "Bot") || (message.MembersAdded[0].Name == "Shopbot")) {
            string replyMessage = string.Empty;
            replyMessage += $"Hey Shopper.. \n\n";
            replyMessage += $"I am Eva - Your Personal Shopping Assistant \n\n";
            replyMessage += $"Send me a picture of Shoes you'd like to shop today.\n";
            return message.CreateReply(replyMessage);
        }

    }
    else if (message.Type == ActivityTypes.ContactRelationUpdate)
    {
        // Handle add/remove from contact lists
        // Activity.From + Activity.Action represent what happened

        |
    }
    else if (message.Type == ActivityTypes.Typing)
    {
        // Handle knowing tha the user is typing

    }
    else if (message.Type == ActivityTypes.Ping)
    {

    }

    return null;

}
}
```

- Method to receive message

```
using Microsoft.Bot.Builder.Dialogs;
using Microsoft.Bot.Connector;
using Newtonsoft.Json;
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net.Http;
using System.Net.Http.Headers;
using System.Threading;
using System.Threading.Tasks;
using System.Web;

namespace Shopbot
{
    [Serializable]
    public class ShopbotDialog : IDialog<object>
    {
        public async Task StartAsync(IDialogContext context)
        {
            context.Wait(MessageReceived);
        }

        public async Task MessageReceived(IDialogContext context, IAwaitable<IMessageActivity> argument)
        {
            var message = await argument;
            string[] products = new string[0];
            // default to message text, so we can bypass the image recognition
            var productdescription = message.Text;
            if (message.Text != null) {
                await context.PostAsync($"No!!! I need an image. Please upload an image.");
            }

            // is there an image in the message?
            var image = message.Attachments?.
                FirstOrDefault(x => x.ContentType.ToLowerInvariant().Contains("image"));

            // if so, fire off image recognition
            if (image != null)
            {
                var connector =
                    new ConnectorClient(new Uri(message.ServiceUrl));
            }
        }
    }
}
```



```
// if so, fire off image recognition
if (image != null)
{
    var connector =
        new ConnectorClient(new Uri(message.ServiceUrl));

    // if the image was passed as byte data in Content, use that
    var data = image.Content as byte[] ??

        // if not, download it from the ContentUrl
        connector
            .HttpClient
            .GetByteArrayAsync(image.ContentUrl)
            .Result;

    productdescription = await ProcessImage(context, message, data);

    if (productdescription.Contains(';'))
    {
        products = productdescription.Split(';');

        foreach (string product in products)
        {
            await
                context.PostAsync($"For {product}..");

            // find matching products and display them
            await ShowProductListing(context, product);
        }
    }
    else {
        // find matching products and display them
        await ShowProductListing(context, productdescription);
    }
}

context.Wait(MessageReceived);
}
```

- Method to generate product listings in a Carousel

```
private static async Task ShowProductListing(IDialogContext context, string productdescription)
{
    // build the shopstyle query
    var shopClient = new HttpClient
    {
        BaseAddress = new Uri("http://api.shopstyle.com/api/v2/")
    };

    var jsonproductresponse = await
    (await
        shopClient.GetAsync(
            "products?" +
            $"pid=uid356-30731934-21&" +
            $"fts={HttpUtility.UrlEncode(productdescription)}&" +
            "offset=0&limit=10")
        .Content
        .ReadAsStringAsync());

    // create a dynamic object from the json response
    dynamic productresponse
    = JsonConvert.DeserializeObject(jsonproductresponse);

    // did we find any results?
    if (productresponse.metadata.total > 0)
    {
        await context.PostAsync($"I found {productresponse.metadata.total} items similar to the image!");

        var productlist = new List<Attachment>();

        // show a max of 10 items
        int productMax =
            productresponse.metadata.total < 10 ?
            productresponse.metadata.total : 10;

        for (var i = 0; i < productMax; i++)
        {
            // create a link to the product as a Card Action
            var buttons = new List<CardAction>
            {
                new CardAction
                {
                    Title = "View Details",
                    Type = "openUrl",
                    Value = productresponse.products[i].clickUrl
                }
            }
        };
    }
};
```

```
// try to get an image if there is one
var imgs = new List<CardImage>();
string img = productresponse
    .products[i]?
    .image?
    .sizes?
    .XLarge?
    .url;

if (!string.IsNullOrEmpty(img))
{
    imgs.Add(new CardImage(img));
}

// add the Card Action and the image to a Hero Card attachment
var attachment = new HeroCard
{
    Text = productresponse.products[i].name,
    Images = imgs,
    Subtitle = productresponse.products[i].priceLabel,
    Buttons = buttons
};
productlist.Add(attachment.ToAttachment());
}

// create the carousel from the product list
var carousel = context.MakeMessage();
carousel.Attachments = productlist;
carousel.AttachmentLayout = AttachmentLayoutTypes.Carousel; //or List
carousel.Text = "Similar products - Tap to Buy";

await context.PostAsync(carousel);
}
else
{
    await
        context.PostAsync($"Sorry, didn't find anything. Try again!");
}
}
```

- Method to process Image and perform Image Recognition

```
private static async Task<string> ProcessImage(IDialogContext context,
        IMessageActivity message,
        byte[] data)
{
    await context.PostAsync($"Let me try to find what you're looking for in that image!");
    await context.PostAsync($"Please be patient :)");

    // build the request's content - a very specific request structure
    var content =
        new MultipartFormDataContent("Upload-----" + DateTime.Now)
        {
            {
                // the image byte data
                new StreamContent(new MemoryStream(data)),
                "image_request[image]", "image.jpg"
            },
            {
                new StringContent("en-GB"),
                "image_request[locale]"
            }
        };

    var imgClient =
        new HttpClient
        {
            BaseAddress = new Uri("https://localhost:8080/imageRecognition/")
        };

    imgClient.DefaultRequestHeaders.Authorization =
        new AuthenticationHeaderValue("CloudSight", "mI15EWumC92bGgQgQmt7uQ");

    // Send the image for processing
    var responseMessage =
        await imgClient.PostAsync("image_requests", content);

    // Get the token for this request from the response
    var jsonimageresponse =
        await responseMessage.Content.ReadAsStringAsync();

    // get a dynamic object using Newtonsoft.Json
    dynamic imageresponse =
        JsonConvert.DeserializeObject(jsonimageresponse);

    // check the image processing status using the token
    var jsonimagestatus = await
        (await imgClient.GetAsync($"image_responses/{imageresponse.token}"))
        .Content
        .ReadAsStringAsync();
}
```

```
// check the image processing status using the token
var jsonimagestatus = await
    (await imgClient.GetAsync($"image_responses/{imageresponse.token}"))
    .Content
    .ReadAsStringAsync();

dynamic imagestatus = JsonConvert.DeserializeObject(jsonimagestatus);

// prepare the "typing" response..
var typing = context.MakeMessage();
typing.Type = ActivityTypes.Typing;

// if it's not Completed or Timed Out yet, wait and poll
while (imagestatus.status != "completed" && imagestatus.status != "timeout")
{
    // not done yet, show the chatbot loading spinner..
    await context.PostAsync(typing);
    Thread.Sleep(2000);

    jsonimagestatus = await
        (await imgClient.GetAsync($"image_responses/{imageresponse.token}"))
        .Content
        .ReadAsStringAsync();

    imagestatus = JsonConvert.DeserializeObject(jsonimagestatus);
}

// Did it Complete or Time Out?
string productdescription = "";
string[] products = new string[0];
if (imagestatus.status != "timeout")
{
    // Got a result!
    productdescription = imagestatus.name;
    if (productdescription.Contains(';'))
    {
        products = productdescription.Split(';');
        await
            context.PostAsync($"I must say, Nice Choice! - seems like it's a ");
        foreach (string product in products)
        {
            await
                context.PostAsync($"{product}!");
        }
    }
    else {
        await
            context.PostAsync($"Aha - seems like it's a {productdescription}!");
    }
}
```

```
        products = productdescription.Split(';');
        await
            context.PostAsync($"I must say, Nice Choice! - seems like it's a ");
        foreach (string product in products)
        {
            await
                context.PostAsync($"{product}!");
        }
    }
    else {
        await
            context.PostAsync($"Aha - seems like it's a {productdescription}!");
    }
}
else
{
    // Timed Out
    await
        context.PostAsync("Ah, couldn't find anything this time, sorry.");
}
return productdescription;
}
}
```