Content Based Fashion Recommendation System

Kushal Doshi, Nishant Kumar, Nidhi Arun Harwani
Rutgers University
Piscataway, NJ, USA
Email: kushal.doshi@rutgers.edu, nishant.kumar@rutgers.edu, nidhi.arun.harwani@rutgers.edu

I. Abstract

The purpose of our project is to get similar items based on user searches also known as a content-based recommendation system for Amazon.com. We will be using the database provided by Amazon in JSON format from their publicly available S3 bucket. We use attributes such as product type, brand, color, etc, to fetch similar items using text-based similarity. This project uses techniques like bag-of-words, word2vec, TFIDF to feature the text into numerical vectors and eventually gauge the similarity between them. This will help us attain the goal of obtaining a similar set of items for possibly each item. Nearest Neighbour search will be used as the final algorithm to find the similarities.

***Index Terms*—Natural Language Processing, Nearest Neighbor Similarity, Bag of Words, tf-IDF, Word2vec, text pre-process, Keras, Tensorflow, Content Based Recommendation System**

II. Introduction

The aim of this project is to fetch similar items for users based on their text searches for a particular product. We will be primarily focusing on the woman's apparel in this project. Using machine learning and various recommendation systems, Amazon.com successfully up its sales by 35 percent, that sums up to be 40 billion dollars annually. Our aim is to simulate a similar recommendation engine. Let's take an example here to clarify what we want to achieve. Say a user searches for "Polka dots round neck top". The result for this search will automatically be apparel tops which have polka dots. Although there will be thousands of such in the database, the customer will click on and go to the description page of only those similar products that she likes. The basic idea would be to add a section of similar items on the product page itself. For example, the customer selected some item and its description reads 'white polka dots on a black top with round neck and cut shoulders' then our system will fetch all those items that are similar to this description and display those to the customer in our recommendation system. The user can then skim through those results and check out similar items. This is a plus-plus scenario for the website as well as the user. The customer's experience upgrades as their time are cut short in searching for what exactly they want with the help of the recommendations. Eventually, they shop for more due to getting apt results and more products as per their desire, which helps in the increase of the total sales of the website.

III. Related Work and Preliminaries

There has been a lot of work done in this field. For example, one very popular algorithm is Collaborative Filtering.[1] One type of collaborative filtering is user-based collaborative filtering. It starts by finding a set of customers who have purchased and rated similar items with the target users purchasing history. It aggregates items from these similar customers and uses the ratings from other similar users to predict the ratings from this user. There are other algorithms too. Random walks is one which could be used in predicting links in complex graphs in a very efficient manner. Also, if we model the user and product graph as a bipartite graph, then it becomes feasible to use a Bipartite Projection algorithm to calculate the relevance between two customers. Thus the predicted rating is essentially based on the other relevant customers ratings. In this project, will be using Content-based (Item-based) filtering.

Before applying Natural Language Processing algorithms, we need to pre-process the data.[2] Because we have real-world data, it will be generally incomplete. It will lack certain attribute values, would contain noisy data such as outliers, erroneous values and would be inconsistent: a lack of relatability between two or more attributes. Price and color are important features of our data, hence we removed the rows which have no values or contains NULL for these features. We will recommend the products looking at the users past purchases or searches. Hence, we analyzed the feature title and gained as much information we could about the users interests. We deleted the rows which had less than four words in title as implementing text similarity on them would be not very useful for our recommendation system. Next, we removed the duplicates which differed only at the end as it will be a piece of redundant information about the product and also the products whose titles are not adjacent but very similar. And finally, we removed stop words from our title feature.

IV. Problem Formalization

The User will input a query. We will perform Natural Language Processing algorithms on the query and make recommendations using the Nearest Neighbor Search. We will start by converting the user query into numerical vectors using different techniques like Bag of words, TF-IDF and Weighted Word2Vec.[3] We then calculate the euclidean distance of the query to rest of the points and recommend top 10 similar products based on the distance calculated. The output will be

title of the product, its euclidean distance from the user query and the product image.

V. THE PROPOSED MODEL

The primary feature that we have used to make the recommendations is the title of the product. Our first objective is to convert the title into numerical vectors.

- **Natural Language Processing Techniques**

1) **Bag of Words**

In this approach, we use the tokenized words for each and find out the frequency of each token.
Example:
Doc1 = John Likes movies.
Doc2 = Mary Likes movies too

John	Likes	Movies	Mary	Too
1	1	1	0	0
0	1	1	1	1

Fig. 1. Bag of Words example

2) **TF-IDF**

It stands for Term Frequency - Inverse Document Frequency.
TF (t, D) = frequency of word t in Document D / Total number of words in Document D
IDF (t) : Log (Total Number of Documents in the corpus / Number of Documents containing word t)
Example:
Doc1 = John Likes movies.
Doc2 = Mary Likes movies too

John	Likes	Movies	Mary	Too
0.7049	0.5015	0.5015	0	0
0	0.4099	0.4099	0.5761	0.5761

Fig. 2. TF-IDF example

3) **Word2Vec**

Word2vec is a group of related models that are used to produce word embedding. Tese models are shallow, two-layer neural networks that are trained to reconstruct linguistic contexts of words. Word2vec takes as its input a large corpus of text and produces a vector space, typically of several hundred dimensions, with each unique word in the corpus being assigned a corresponding vector in the space. Word vectors are positioned in the vector space such that words that share common contexts in the corpus are located in close proximity to one another in the space. Once the title is converted into numerical vectors, we calculate the euclidean distance between the user query and each data point in our data. We make recommendations by arranging the points in increasing order of the distances and select top 10 closest points. This technique is called as the nearest neighbor search. In general, for an n-dimensional space, the distance is :

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(p_1-q_1)^2+(p_2-q_2)^2+\cdots+(p_i-q_i)^2+\cdots+(p_n-q_n)^2}$$

- **Image Feature Extraction**

We have used title or text feature to recommend products to the user. Other important feature which could be used to find similar products is Image. Image has attributes like edges, corners, blob, ridges, color etc which can be compared to other image to find similarity between them. To incorporate this in our model, we need to represent each image to a vector.

Keras is used for data flow and differentiable programming across a range of tasks. We used Keras to implement Visual Geometry Group Convolutional Neural Networks (VBB CNN) on 16042 training data points of images(dim 224,224) to convert them to dense vectors of length 25088.[4] CNN being a regularised version of multilayer perceptrons, each image was passed through a stack of convoluted layers, where filters were used with a small respective field. Three Fully-Connected (FC) layers follow a stack of convolutional layers, which has a different depth in different architectures (first two has 4096 channels, the third has 1000 channels each). The final layer was the softmax layer[5]
What we had as the output of the neural network was a vector representation of each image. When a query point was taken as the input, the euclidean distance between the query image and rest of the images was calculated from their vector representations. Distances were sorted in ascending order and its top vectors row were the recommended product image for the user.

VI. EXPERIMENTS

Below are the results from the techniques used for recommendation:

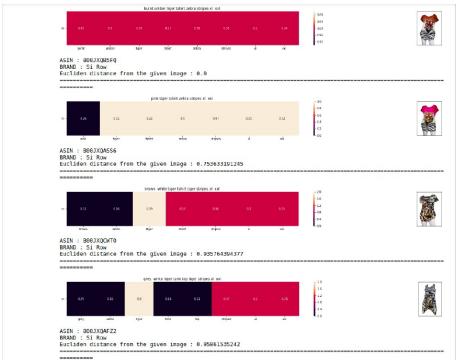

Fig. 3. Bag of Words

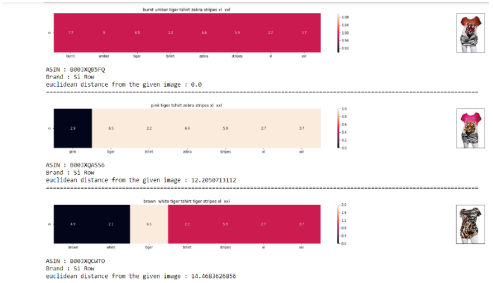

Fig. 4. TF-IDF

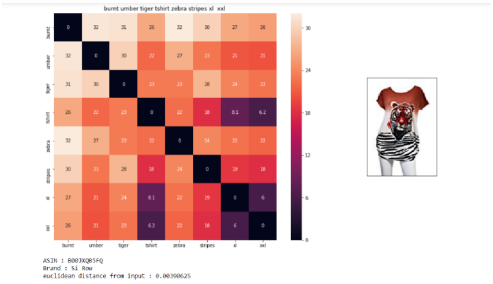

Fig. 5. Word2Vec Query

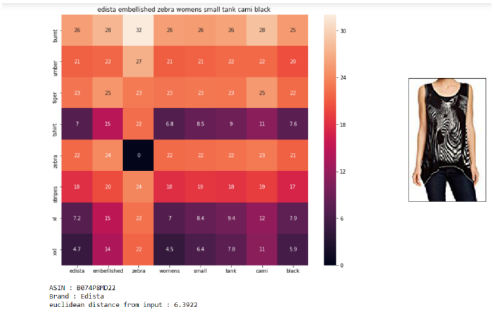

Fig. 6. Word2Vec

Fig. 7. Image Feature Extraction - OpenCV

VII. Deployment and Evaluation

As seen in the above sections, we tried various text based similarity models and an image based model for recommending similar apparel. After evaluating between the various text based models, we found out that all the models work with almost the same precision. Thus, we chose one of them, Bag of Words, as the text based similarity model for deployment on our website. With this, we also used image based OpenCV model to show how the image based is different than the text-based model. Figures 8 and 9 are examples of the two deployed models:

Fig. 8. Bag of Words model

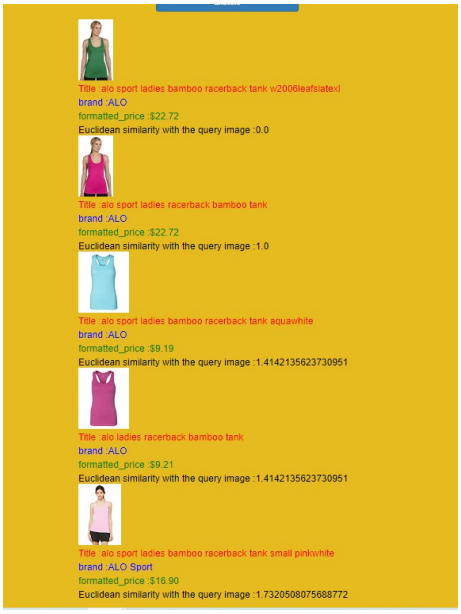

Fig. 9. OpenCV model

For the evaluation part, we tested query points for deep learning model using CNN and NLP technique - Bag of Words. We humanly assessed the results for each query point and gave a score from 1 to 5. This assessment was based as to how likely we would select the recommended apparel based on the original one. At the end, we got an average score of 4.05 for the OpenCV model and an average score of 3.86 for Bag of Words. Hence, we can conclude that the recommendation system works better when the model accepts the features of the image instead of the tittle describing the image of the product.

VIII. Conclusion

We have used the row id (asin) as our query point. Features like title : text feature and image were converted to vectors. Using Euclidean distances, we compared our query point to the rest of the data and recommended top vectos with least distances. Comparing NLP techniques,Bag of Words and Tf-Idf gave us similar results but Word2Vec gave us varied and relatable outputs(looking at the linguistic contexts of words from 'title'). Recommending using Images usually gave outputs with similar color and the shape of the figures printed on the product. Lastly, we found that image based recommendations were more likely to be chosen rather than the text based one.

IX. Contribution

This project was created by Kushal Doshi, Nishant Kumar, and Nidhi Arun Harwani under the guidance of Professor Richard Martin. There are two major components of this project: Back-end and Front-end. The back-end which constitutes cleaning and analyzing the data and building the various models using NLP techniques and deep learning techniques were done by Kushal and Nidhi. Nishant worked on the front-end which constituted creating the User Interface and also integrating the generated models to the UI. All three of us worked together for the evaluation of the models, project reports, and presentations.

References

[1] A. Nenkova and K. McKeown, *Automatic Summarization*, 06 2011, vol. 5.

[2] J. Stewart, V. Mittal, J. Carbonell, and M. Kantrowitz, "Multi-document summarization by sentence extraction," *NAACL-ANLP 2000 Workshop on Automatic Summarization*, 05 2002.

[3] R. Mitkov, *The Oxford handbook of computational linguistics*. Oxford University Press, 2019.

[4] C. Mead, *Analog VLSI and Neural Systems*. USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[5] Chiueh Tzi-Dar and Goodman, "High-capacity exponential associative memories," in *IEEE 1988 International Conference on Neural Networks*, 1988, pp. 153–160 vol.1.