Product Recommendation System with Deep Learning

Zihe Yang, Minyuan Xu, Xueyan Liu, Jiamin Zhong

***Abstract*—This project aims to create a fashion product recommendation system based on the product image in customers' searching, viewing and purchase history. For achieving the goal, it is split to 3 parts: 1. Image Classification: to implement CNN model tagging the image by multiple categories. 2. Image Similarity Check: to find out the similar products by Eigenfaces and ResNet50 model. 3. Customer Preference Prediction: to help predict the property of the products (such as 'if fragile'), and creating more detailed customer segmentation.**

I. INTRODUCTION

The coronavirus pandemic is rapidly changing our behavior toward e-commerce, and the shifts are likely to stick post-pandemic. According to Howard University, over 56% of surveyed U.S. adults stated that they shopped more online, 67% reported going to the shop less, and even those who had never used an e-commerce service felt motivated to do so[2].

This growing e-commerce industry presents us with a large dataset waiting to be researched on. In this project, we plan to start with fashion product images with several label attributes (category, color, size and etc.) to train an images classifier, to predict features based on product description and other attributes, and to build similarity check models taking images as input and output several prioritized images, helping fashion recommendation system.

II. RELATED WORK

Convolutional Neural Network Model: CNN is one of the most widely used deep learning methods for image recognition and classifications. Each input image will pass it through a series of convolution layers with kernels, pooling, fully connected layers and apply Softmax function to classify an object with probabilistic values between 0 and 1.

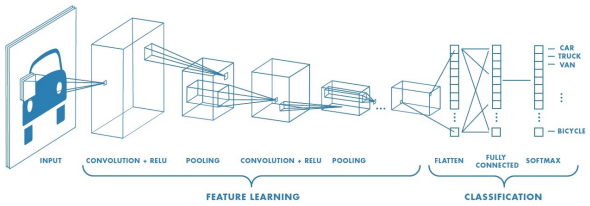

Figure 1: Complete flow of CNN model to process and classify an input image

CNN is used in the project for multi-class and multi-label classification.

Eigenface Model: It helps decompose the images and retrieve PCA for each image. Next by training the eigenface model, we are able to find a most similar image of the input image based on the components (eigenfaces).

Residual Net 50 Model: Residual Network is trained to take input of images and represent them as a set of vectors (in our case, these corresponding vectors are the fashion image embeddings). We generate our image embeddings of the first 5000 articles using ResNet50.

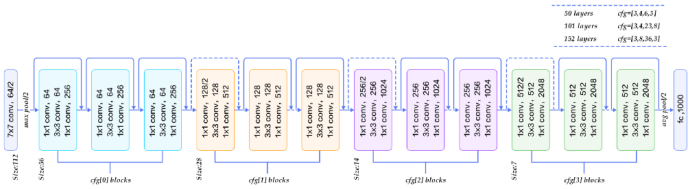

Figure 8: Image Courtesy of https://medium.com/@siddharthdas_32104/cnns-architectures-lenet-alexnet-vgg-googlenet-resnet-and-more-666091488df5

Feedforward Neural Network Model: The goal of a feedforward network is to approximate some function f*. For example, for a classifier, y = f*(x) maps an input x to a category y. A feedforward network defines a mapping y = f(x;θ) and learns the value of the parameters θ that result in the best function approximation.[3] We will use this model for prediction of whether a certain item is harmful or not and help the customer make a better decision that suits his or her own needs.

III. DATASET

Fashion Product Images dataset from Kaggle is used in this project. It contains the information of 44k fashion products such as clothes, jewelry, sunglasses, etc. with multiple category labels, descriptions and high-res images. In this dataset, every product is assigned masteCategory, subCategory, and articleType which are used as labels for classification, and season, gender, basecolor, usage, year, etc. as features, and otherFlags such as isFragile, isReturnable, etc. which are used as labels for prediction for customer segmentation.

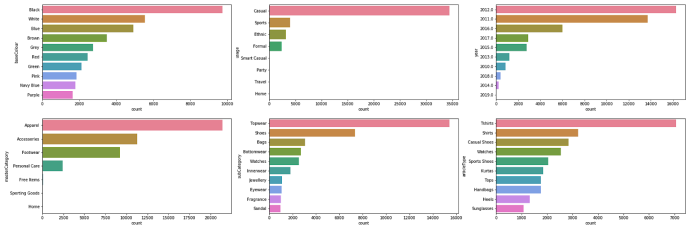

Figure 2: Some categories of dataset

Data Preprocessing: All the methods are implemented with the cleaned dataset below. We have hundreds of categories in the original dataset, here we only choose top 10 categories: "Shirts", "Watches", "Sports Shoes", "Tops", "Handbags", "Heels", "Sunglasses", "Backpacks", "Jeans", "Shorts" from And in the end, 8880 images are used. You can find the distribution of our categories here.

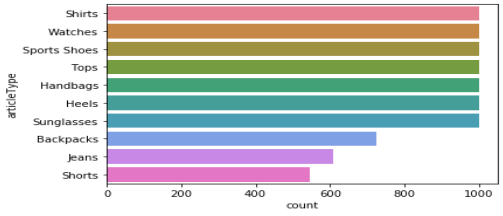

Figure 3 : Top 10 article type categories

Then sample 80% for training, 20% for tests.

IV. METHODS

Classification-CNN Model: Convolutional neural network models are ubiquitous in the image data space. They work phenomenally well on computer vision tasks like image classification, object detection, image recognition, etc. Here, we will be building CNN classification models using CNN by 2 different types of classifier:

1. *Multi-Class classifier*. A classification works with one set of tags that includes more than two categorical variables, each image only being assigned one tag. We select ReLU as the activation function for Cov layers and Softmax as the last layer of the CNN model.
2. *Multi-Labels classifier*. A classification works with multiple sets of tags. An image can be assigned more than one tag. (As the *Table1* shown, an image can be marked as "Men" and "Women" at the same time to representing "Unisex"). Here, we set multiple Sigmoid activation functions on the last layer of the CNN model.

	Multi-Classes	Multi-Labels
unisex	[Girls,Boys,Men,Unisex,Women]	[Girls,Boys,Men,Women]
	[0,0,0,1,0]	[0,0,1,1]
Girl-Winter	[Girl-Winter,Boy-Spring,Girl-Spring]	[Girl,Boy] [Spring,Winter]
	[1,0,0]	[1,0] [0,1]

Table 1: Multi-Class & Multi-Label example

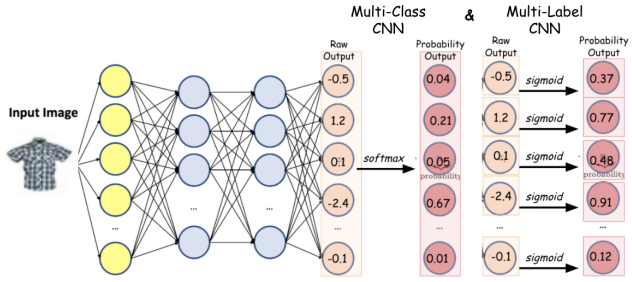

Figure 4: Multi-Class & Multi-Label CNN model output example

Similarity Check- Eigenfaces is using PCA (Eigenvalues and EigenVectors) to reduce image dimensions and project a training sample on small feature space. Then, the reduced basis of features can be linearly combined to re-create and match to any image in the original dataset of images.

Let's use an example to understand the training algorithm of eigenvalues. We assume to take a training set of m images of dimension N*N. First, to convert these images into vectors of size N^2 , and calculate the average of all these face vectors and subtract it from each vector. Next, compute the eigenvalues of $A^T A$. Next, we choose the k highest eigenvalues and corresponding eigenvectors. Then to represent each face vector in the linear combination of these best K eigenvectors.

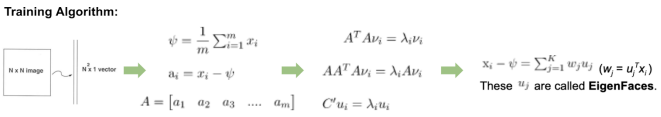

Figure 5: Training Algorithm of Eigenfaces Method

Figure 6: Image and 12 eigenvectors with highest eigenvalues

After training the eigenface model, we could use any image to test the model. Given an unknown image, we need to first subtract the face from the average face. Then, we project the normalized vector into eigenspace to obtain the linear combination of eigenfaces. From the above projection, we generate the vector of the coefficient. Here we subtract it from the training image to get the minimum distance between the training vectors and testing vectors. If this error is below the tolerance level, then our unknown image is recognized as that image with minimum distance from the training set.

Test Algorithm:

Step 1: normalize Γ: $\Phi = \Gamma - \Psi$

Step 2: project on the eigenspace

$$\hat{\Phi} = \sum_{i=1}^{K} w_i u_i \quad (w_i = u_i^T \Phi)$$

Step 3: represent Φ as: $\Omega = \begin{bmatrix} w_1 \\ w_2 \\ \dots \\ w_K \end{bmatrix}$

Step 4: find $e_r = \min_l \|\Omega - \Omega^l\|$

Step 5: if $e_r < T_r$, then Γ is recognized as face l from the training set.

Figure 7: Eigenfaces test algorithm

Similarity Check-CNN Model:

After retrieving the set of vectors (image embeddings), we will determine the most similar items by cosine similarity. By finding neighbors with the largest cosine similarity in the embedding space, we can make recommendations based on the user interests or cluster categories.

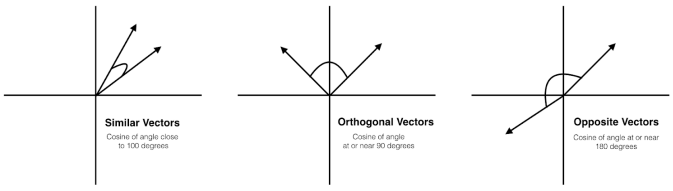

Figure 9: Illustration of cosine similarity between vectors

Since our latent space is high-dimensional, we will next reduce the dimensionality to two-dimensional and visualize the data using t-SNE, an algorithm that learns to project data in high dimensional space into lower dimensional space while preserving global and local distances.

Prediction-FFNN model: Feedforward neural networks are models where the information only travels forward through the input nodes, then through the hidden layers, and finally to the

output nodes. To handle the non-linear decision boundary between input and the output for the prediction, we are using the Multi-layered Network of Neurons. The package of pyTorch is used for this part.

V. RESULTS

Classification: We firstly applied these two type classifiers to every single categorical column (such as gender or season). Then one hot encoded all columns to 23 binary columns or combined all the columns to a single categorical column that has 99 different elements, fitting the multi-classes multi-labels CNN models. In sum, we create the 12 CNN models by different classifiers and categories.

The following *Table2* shows the highest accuracy and time spending of our classifications by the various classifiers. If we build a model for each individual class, then our final classification accuracy of all tags is 97%*97%*81%*70% *93% = 50% and it takes more than1 hour in total. Obviously, the performance of building multiple single-label multi-class models is far worse than building a single multi-class multi-label CNN model.

	Accuracy		Time
Type	Multi-Class	Multi-Labels	
Master Category	97%	99%	880s
Sub-Category	97%	81%	890s
Gender	81%	85%	880s
Season	70%	67%	880s
Usage	93%	13%	880s
Multi-Category	62%	32%	990s

Table 2: Highest accuracy of each model

The following figure is the visualization of each model's accuracy curve for both the training and validation data. In general, for the most single category models*(plot2-6)*, the accuracies of the two type classifiers are similar, but when we faced the classification model having multi-outputs multi-class category, the model using softmax as output function is much more accuracy*(plot1)*

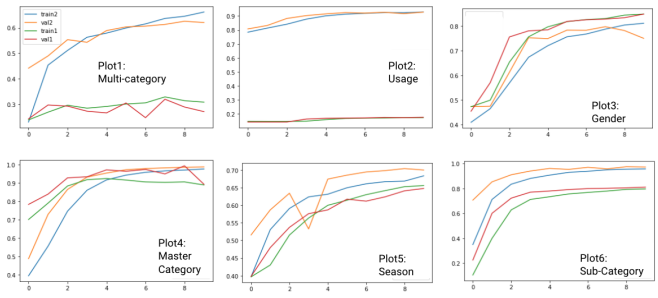

Figure 10:
Note: The blue and orange line represents the multi-class model that used Softmax as output function; green and red lines represent the multi-label model that used Sigmoid as the output function

Similarity Check-Eigenfaces: With the method, first, we got this average image. Then we select top 30 eigenfaces which are listed below.

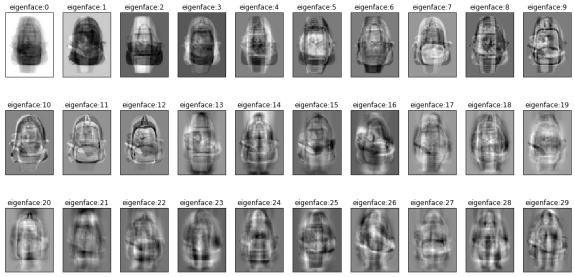

Figure 11: Top 30 Eigenfaces

Last, we get the most similar image from databases with given images. Results are listed below but the accuracy of this method is low.

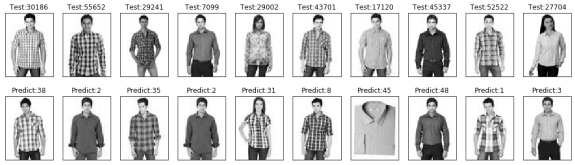

Figure 12: Result of Eigenfaces Method

Similarity Check-CNN Model: The ResNet50 model enables commercial websites to create a more accurate recommendation system based on the purchase and browsing history of each customer. We randomly test an image indexed 1989 (on the left), and have 6 returned recommendations (on the right).

Figure 13:input image and six returned recommendations

Apparently the returned recommendation images are highly aligned with the input image.

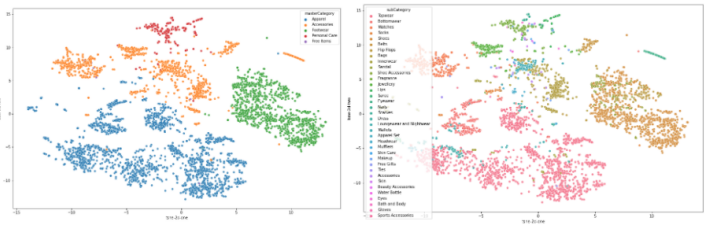

Figure14: t-SNE 2-D plot for MasterCategory(left) and SubCategory(right)

The master categories are nicely separated,meanwhile it appears that some Sub Categories overlap with each other.

Prediction: We fit a FFNN model on 6660 training data to predict the feature of 'isFragile', and the plot of the loss is shown in Figure 15, which seems to converge very well. And later, when we train the model on the left 2220 data, the accuracy is 88%.

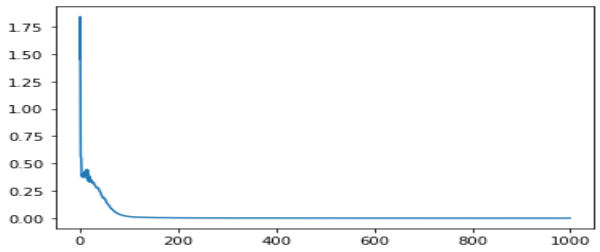

Figure 15: FFNN loss

VI.DISCUSSION

Classification: Our ultimate goal is to accurately tag the product images. Since our dataset has complex categories, it is a great challenge for us finding the most effective way to build the classification model. From the results, the comprehensive category classification model is more accurate and consumes less time. But if we only need to tag the image by one category. It is undoubtedly a better choice to build a single-label multi-class model, the accuracy can be as high as 92%.

As for classifier selection: multi-label model or multi-class, multi-class classifier model has the conspicuous advantage. It not only has higher accuracy but also is less sensitive to imbalanced data. For example, in the 'Usage' category*(plot2, figure*)*, 3 of 7 labels accounted for 98% data, reducing the accuracy of the multi-labels classifier to 13%, compared to 67% accuracy of the multi-class classifier. We think it is caused by the output function. In mathematically, the sigmoid output of each category is independent. In contrast, the outputs of a softmax are all interrelated. The elements that lacking of data make the sigmoid function not have enough information to make an accurate classification, while the softmax function can classify according to other elements.

In addition, when we use the multi-class classifier model to handle the multi-label category, it requires us to convert all the labels and classes into individual binary classes, in other words, binary encode all the categorical variables[4].

Similarity Check-Eigenfaces:

Advantages:

- Easy to implement and computationally less expensive.
- No knowledge of the image required.

Limitations :

- Proper centered subject is required for training/testing.
- The algorithm is sensitive to lightning, shadows and also the scale of face in the image .
- Front view of the image is required for this algorithm to work properly.

Similarity Check-CNN:

Advantages:

- Dimensionality Reduction provides a more efficient representation
- Contextual Similarity enables a more expressive representation

Limitations:

- For high-cardinality variables — those with many unique categories — the dimensionality of the transformed vector becomes unmanageable.
- The mapping is completely uninformed: “similar” categories aren’t placed closer to each other in embedding space.

Prediction: This prediction part is used for customer segmentation. Customers can use the information they already get to predict the features they want to know to help them decide whether to buy the product. In the code provided, we only train the model to predict the feature of if the product is fragile. For other features needed to be checked, customers can just change the string of ‘isFragile’ to the name of the feature, and then run the code.

VII. CONCLUSION

Image classification, similarity check and customer segmentation, bringing these three steps into this product recommendation system improves the accuracy to predict people’s preferences. Experiments also show that the recommendation system built in this project has a good effect to help customers choose their products. When the customer in the e-commerce site starts to click some product detail page, the recommendation system will analyze the product image to generate combined labels and present the most similar products under these labels at that detail page. When customers hesitate whether the purchase should be made or not by these existing product properties, they could use the system to predict extra properties to help them make the decision.

The limitation of this system could be these noisy product images in the real world. Unlike these images used in this project, product images in e-commerce websites could contain a lot of objects and noises. Extracting the particular object we are interested in before training the model will definitely help improve the system accuray. This might be the future direction of this project.

A picture speaks louder than a thousand words. We live in the age of Instagram, YouTube, and Twitter where images and video (a sequence of images) dominate the way we consume information. Over the past few years, deep learning techniques have enabled rapid progress in helping us understand images. This project is a good start for us, we look forward to exploring more on deep learning techniques in the future.

REFERENCE

[1] Aggarwal, P. (2019) Kaggle Fashion Product Images Dataset(Version 1), 44k products with multiple labels.https://www.kaggle.com/paramaggarwal/fashion-product-images

[2] Draelos, R., Draelos, P., & Draelos, R. (2019, September 08). Multi-label vs. Multi-class Classification: Sigmoid vs. Softmax. Retrieved December 16, 2020, from https://glassboxmedicine.com/2019/05/26/classification-sigmoid-vs-softmax/

[3] Ian Goodfellow and Yoshua Bengio and Aaron Courville,Deep Learning, MIT Press, Retrieved 2016, from http://www.deeplearningbook.org

[4] C. Li, Q. Kang, G. Ge, Q. Song, H. Lu and J. Cheng, "DeepBE: Learning Deep Binary Encoding for Multi-label Classification," 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, 2016, pp. 744-751, doi: 10.1109/CVPRW.2016.98.

[5] S. A. Shahriyar, K. M. Rokibul Alam, S. S. Roy and Y. Morimoto, "An Approach for Multi Label Image Classification Using Single Label Convolutional Neural Network," 2018 21st International Conference of Computer and Information Technology (ICCIT), Dhaka, Bangladesh, 2018, pp. 1-6, doi: 10.1109/ICCITECHN.2018.8631970.