

Cross-Border E-Commerce Products Based on Image Recognition and Deep Learning

Submitted in partial fulfillment of the requirements for
the award of
Bachelor of Engineering degree in Computer Science and Engineering

by

NULI MEGHANA (37110523)

VATTUMALLISAIMOUNIKA(37110811)



**DEPARTMENT OF COMPUTER SCIENCE AND
ENGINEERING SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600 119**

MARCH - 2021



SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

Accredited with “A” grade by NAAC
Jeppiaar Nagar, Rajiv Gandhi Salai, Chennai – 600 119

www.sathyabama.ac.in



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this project report is the bonafide work of **NULI MEGHANA(37110523)** and **VATTUMALLI SAI MOUNIKA (37110811)** who carried out the project entitled “**Cross-Border E-Commerce Products Based on Image Recognition and Deep Learning**” under my supervision from **NOVEMBER 2020 TO MARCH 2021**.

Internal Guide

DR. L.LAKSHMANAN

Head of the Department

Dr. S. Vigneshwari, M.E., Ph.D.,

Dr. L. Lakshmanan, M.E., Ph.D.,

Submitted for Viva voce Examination held on_____

Internal Examiner

External Examiner

DECLARATION

I **NULI MEGHANA** hereby declare that the Project Report entitled “**Cross-Border E-Commerce Products Based on Image Recognition and Deep Learning**” is done by me under the guidance of **Dr. L.Lakshmanan** Department of Computer Science and Engineering at Sathyabama Institute of Science and Technology is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: CHENNAI

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala, M.E., Ph.D., Dean**, School of Computing, **Dr. S. Vigneswari, M.E., Ph.D., and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.L.Lakshmanan**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

ABSTRACT

We focus on the problem of parsing fashion images for detecting various types of clothing and style. The current state-of-the-art techniques for the problem are mostly based on variations of the Seg Net model. The techniques formulate the problem as segmentation and typically rely on geometrical shapes and position to segment the image. However, specifically for fashion images, each clothing item is made of specific type of materials with characteristic visual texture patterns. Exploiting the texture for recognizing the clothing type is an important clue which has been ignored so far by the state of-the-art. In this paper, we propose a two- stream deep neural network architecture for fashion image parsing.

To handle various challenges, we formulate fashion recognition into a multi-task learning problem, including landmark detection, category and attribute classifications, and solve it with the proposed deep convolutional neural network. We design two knowledge sharing strategies which enable information transfer between tasks and improve the overall performance. The proposed model achieves state-of- the-art results on large-scale fashion dataset comparing to the existing methods, which demonstrates its great effectiveness and superiority for fashion recognition.

TABLE OF CONTENTS

	ABSTRACT	V
	LIST OF FIGURES	VIII
	LIST OF ABBRIVATIONS	IX
CHAPTER NO	TITLE	PAGE NO
1	INTRODUCTION	
	1.1 INTRODUCTION	1
	1.2 DOMAIN OVERVIEW	1
2	LITERATURE SURVEY	
	2.1 LITERATURE SURVEY 1	5
	2.2 LITERATURE SURVEY 2	6
	2.3 LITERATURE SURVEY 3	8
3	AIM AND SCOPE OF THE PRESENT INVESTIGATION	
	3.1 AIM OF THE PROJECT	9
	3.1.1 OBJECTIVE	9
	3.2 SCOPE OF THE PROJECT	9
4	SYSTEM ANALYSIS AND DESIGN AND IMPLEMENTATION METHODOLOGY	
	4.1 EXSISITING SYSTEM	10
	4.2 PROPOSED SYSTEM	11
	4.3 REQUIREMENT SPECIFICATION	11
	4.3.1 HARDWARE REQUIREMENT	11
	4.3.2 SOFTWARE REQUIREMENT	11
	4.4 SOFTWARE DISCRPTION	12
	4.5 SYSTEM ARCHITECTURE	15
	4.6 SYSTEM IMPLEMENTATION	16
	4.6.1 PRODUCT DATA EVALUATION	16
	4.6.2 PRE – PROCESSING THE DATA	16
	4.6.3 IMAGE PRE – PROCESSING	17
	4.6.4 FEATURE EXTRACTION	17
	4.6.5 TRAINING THE MODEL	18
	4.6.6 PRODUCT CATEGORY	19

	PREDICTION	
5	RESULTS AND DISCUSSION	22
6	CONCLUSION AND FUTURE ENHANCEMENT	
	6.1 CONCLUSION	25
	6.2 FUTURE ENHANCEMENT	26
	REFERENCES	27
	APPENDIX	28
	A. PLAGIARISM REPORT	28
	B. JOURNAL PAPER	29
	C. SOURCE CODE	33

LIST OF FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
1.1	Block diagram of Deep Learning	2
1.2	Multiple layers of Neural Network	3
1.3	Architecture	16
1.4	Output for Men's clothing	23
1.5	Output for Women's clothing	24

LIST OF ABBREVIATIONS

ABBREVIATIONS

EXPANSIONS

API	APPLICATION PROGRAM INTERFACE
CNN	CONVOLUTION NEURAL NETWORK
CNTK	THE MICROSOFT COGNITIVE TOOLKIT
DL	DEEP LEARNING
ML	MEACHINE LEARNING
UML	UNIFIED MODELING LANGUAGE

CHAPTER 1

INTRODUCTION

1.1. INTRODUCTION

We do have conducted some experiments on the bidirectional message passing between the parallel network, and we observe that the feature sharing to the landmark detection network hurt the performance of landmark localization. We argue that landmark detection only requires the structural feature that represents the overall structure of the clothes, regardless of the detail feature. Sharing the feature from the classification stream may confuse the landmark detection network, leading to the performance distortion. This architecture can be formulated more as the asymmetrical multi-task learning. The weight of each neuron in the convolution kernel of average pooling sampling is 0.25, and the step size of the convolution kernel is 2 when sliding on the input tensor. The result of mean sampling is equivalent to reducing the input mapping blur to the original $1/4$. Maximum sampling uses a similar method to slide on the input, and uses the maximum neuron (a certain pixel in the picture problem) detected by the sampling window as the sampling result. The effect shown in the figure is that the size of the original image is reduced.

1.2. DOMAIN OVERVIEW

Deep learning is a computer software that mimics the network of neurons in a brain. It is a subset of machine learning and is called deep learning because it makes use of deep neural networks.

Deep learning algorithms are constructed with connected layers.

- The first layer is called the Input Layer
- The last layer is called the Output Layer
- All layers in between are called Hidden Layers. The word deep means the network join neurons in more than two layers.

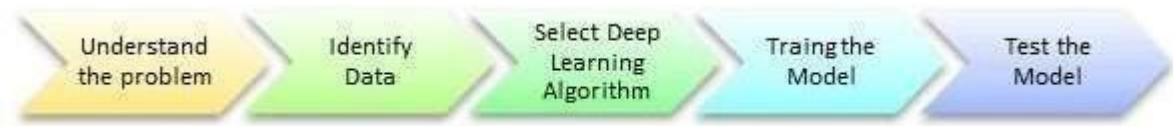


Fig 1.1: Block diagram of Deep Learning

To grasp the idea of deep learning, imagine a family, with an infant and parents. The toddler points objects with his little finger and always says the word 'cat.' As its parents are concerned about his education, they keep telling him 'Yes, that is a cat' or 'No, that is not a cat.' The infant persists in pointing objects but becomes more accurate with 'cats.' The little kid, deep down, does not know why he can say it is a cat or not. He has just learned how to hierarchies complex features coming up with a cat by looking at the pet overall and continue to focus on details such as the tails or the nose before to make up his mind.

A neural network works quite the same. Each layer represents a deeper level of knowledge, i.e., the hierarchy of knowledge. A neural network with four layers will learn more complex feature than with that with two layers. The learning occurs in two phases.

- The first phase consists of applying a nonlinear transformation of the input and create a statistical model as output.
- The second phase aims at improving the model with a mathematical method known as derivative.

The neural network repeats these two phases hundreds to thousands of time until it has reached a tolerable level of accuracy. The repeat of this two-phase is called an iteration. Classification of Neural Networks

- Shallow neural network: The Shallow neural network has only one hidden layer between the input and output.
- Deep neural network: Deep neural networks have more than one layer. For instance,
- Google LeNet model for image recognition counts 22 layers.

The computational models in Deep Learning are loosely inspired by the human brain.

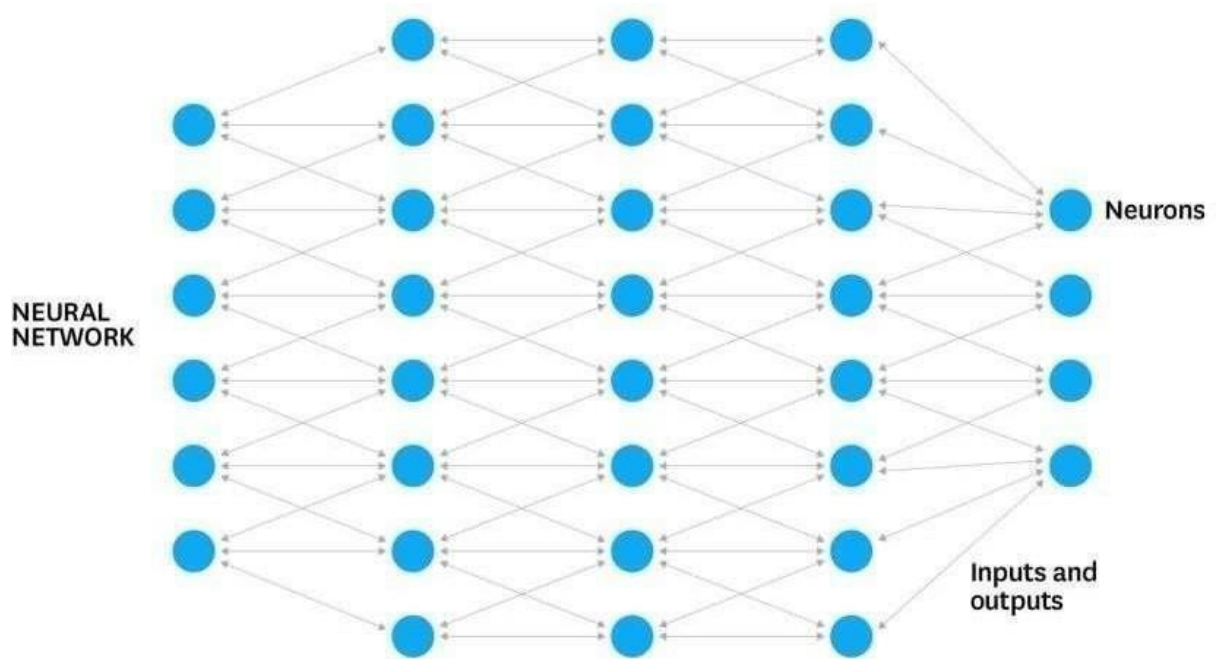


Fig 1.2: Multiple layers of Neural Network

A.NEURON

Artificial Neural Networks contain layers of neurons. A neuron is a computational unit that calculates a piece of information based on weighted input parameters. Inputs accepted by the neuron are separately weighted. Inputs are summed and passed through a non-linear function to produce output. Each layer of neurons detects some additional information, such as edges of things in a picture or tumors in a human body. Multiple layers of neurons can be used to detect additional information about input parameters. **B.NODES**

Artificial Neural Network is an interconnected group of nodes akin to the vast network of layers of neurons in a brain. Each circular node represents an artificial neuron and an arrow represents a connection from the output of one neuron to the input of another.

C.INPUT

Inputs are passed into the first layer. Individual neurons receive the inputs, with each of them receiving a specific value. After this, an output is produced based on these values.

D.OUTPUT

The outputs from the first layer are then passed into the second layer to be processed. This continues until the final output is produced. The assumption is that the correct output is predefined. Each time data is passed through the network, the end result is compared with the correct one, and tweaks are made to their values until the network creates the correct final output each time.

E. CONVOLUTIONAL NEURAL NETWORK

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand- engineered, with enough training, ConvNets have the ability to learn these filters/characteristics. The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image.

ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low- Level features such as edges, color, gradient orientation, etc.

With added layers, the architecture adapts to the High Level features as well, giving us a network which has the wholesome understanding of images in the dataset, similar to how we would.

CHAPTER 2 LITERATURE SURVEY

2.1 Literature Survey 1

Title	Caveat Emptor: How Lay Technical and Professional Communicators Sell Technical Products in C2C E-Commerce.
Authors	VINCENT D. ROBLES
Published Year	2019
Efficiency	These customer engagement techniques might encourage a seller culture that facilitates higher quality content among these users.
Drawbacks	This research implies that the product description content figures largely in mitigating this risk to buyers.
Description	Consumer-to-consumer (C2C) e-commerce involves consumers re-selling products to other consumers using online platforms. Research identifies trust as a major factor in this exchange. It concludes that seller generated product descriptions can mitigate mistrust. Further, technical and professional communication research can reveal what content sellers tend to provide and can reveal how platform design may encourage that content. Literature review: C2C e-commerce and TPC researchers agree that mistrust can be mitigated by detailed content, and they call for platform designers to help improve platform and seller reputations. Research questions: 1. What content do sellers provide about their technical products? 2. How do the platforms' web form designs and the associated documentation about listing a product for sale encourage certain content types? Research methodology: Four platforms

were chosen using specific criteria. Product descriptions were collected once per week for six weeks, generating 1900 product descriptions. These descriptions were unitized and given reliable content categories, a methodology called quantitative content analysis. Further, the documentation and processes for posting items were explored to determine how they may encourage content types. Results/discussion: Sellers mostly provide product information and sales procedures, and they rarely give benefits and goodwill to the buyer. The platform design seems to encourage this content because of the content-entry process, the content-entry options, and the required and unrequired content entry. Conclusions: This study invites technical and professional communicators to provide more guidelines for users about the kinds of content they may include, and designers to explore the content entry process using usability and user-experience research.

2.2 Literature Survey 2

Title	Supervised Topic Modeling using Hierarchical Dirichlet Process-based Inverse Regression: Experiments on E-Commerce Applications
Authors	Weifeng Li, Junming Yin, and Hsinchun Chen, Fellow, IEEE
Published Year	2017
Efficiency	Evaluation of HDP-IR in comparison with the state-of-the-art baseline techniques reveals that both increasing the topic structure flexibility and using sufficient dimension reduction could improve the predictive performance on user-generated

review text in e-commerce applications, and the proposed inference algorithm is highly effective in terms of its scalability.

Drawbacks

Most supervised topic models adopt sampling-based inference algorithms, which require hundreds of iterations over each variable across all documents before convergence. Therefore, the scalability of these models is limited.

Description

The proliferation of e-commerce calls for mining consumer preferences and opinions from user-generated text. To this end, topic models have been widely adopted to discover the underlying semantic themes (i.e., topics). Supervised topic models have emerged to leverage discovered topics for predicting the response of interest (e.g., product quality and sales). However, supervised topic modeling remains a challenging problem because of the need to prespecify the number of topics, the lack of predictive information in topics, and limited scalability. In this paper, we propose a novel supervised topic model, Hierarchical Dirichlet Process-based Inverse Regression (HDP-IR). HDP-IR characterizes the corpus with a flexible number of topics, which prove to retain as much predictive information as the original corpus. Moreover, we develop an efficient inference algorithm capable of examining large-scale corpora (millions of documents or more). Three experiments were conducted to evaluate the predictive performance over major e-commerce benchmark testbeds of online reviews. Overall, HDP-IR outperformed existing state-of-the-art supervised topic models. Particularly, retaining sufficient predictive information improved predictive R-squared by over

17.6 percent; having topic structure flexibility contributed to predictive R-squared by at least 4.1 percent.

2.3 Literature Survey 3

Title	A Model of e-Learning Value Creation Based on Triple-Factor
Authors	Sfenrianto
Published Year	Oct 2016
Efficiency	Better understanding of the existence of these factors to support a model of value creation in e-Learning process.
Drawbacks	The focus for measuring the performance of e-Learning is still rarely done.
Description	This sytem proposes , An understanding of factors that influence the success of students in e-Learning process should be considered in order to improve learning outcomes. To achieve this improvement, this study proposed learning style, motivation, and knowledge ability (triple-factor) to add value creation model of an e-Learning system. The results of the study showed that there was a tendency that e-Learning values (good (Y2) and very good (Y3)) were influenced by learning styles (X2 (30.23%) and X3 (46.51%)), motivation (X5 (27.91%) and X6 (48.84%)), knowledge ability factor (X9 (44.19%) and X10 (30.23%)). Meanwhile the e-learning value of less than satisfactory (Y1) influenced by Learning Styles (X1 (23.26%)), motivation (X4 (23.26%)), knowledge ability factor (X7 (20.93%) and X8 (4.65%)). Thus, this study has proven that the existence of triplefactor that reflect the relationship between e-learning value, and those factors (learning styles, motivation, and knowledge ability). The findings of this study contribute to a better understanding of the existence of these factors to support a model of value creation in e-Learning process.

CHAPTER 3

AIM AND SCOPE OF THE PRESENT INVESTIGATION

3.1. AIM OF PROJECT

The goal of this post is to build intuition and understanding of how neural networks can be trained to identify the micro category of a product using its images. And also for identifying an unified and Robust pre segmentation processing for fashion image identification using Neural Network Algorithms.

Construct a new type of cross-border e-commerce product classification technology by combining image recognition and deep learning technology. This technology can effectively improve the rationality of the functional design of the e-commerce platform, thereby improving the efficiency of cross-border commerce business.

3.1.1. OBJECTIVE

The main purpose of doing this project is:

- Building user-friendly application where user can easier interact easily.
- User can get all the reviews of search query.
- It is also used to build a real time project.

3.2. SCOPE OF PROJECT

The main scope of this project therefore are:

- Data Analysis
- Pre – processing the data
- Training the featured data
- Predict the data
- Test the predicted product

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN AND IMPLEMENTATION METHODOLOGY

4.1 EXSISTING SYSTEM

Cross-border e-commerce is a concrete manifestation of "Internet + foreign trade", which is conducive to improving the people's material living standards and quality, and is also the key to promoting my country's industrial transformation and upgrading. E-commerce platform is the carrier of cross- border e-commerce, and product classification plays a very important role in the e-commerce platform. Aiming at the shortcomings of existing product classification, this paper develops a new classification technology. First, obtain the relevant parameters of the product image through image recognition technology. Secondly, a cross-border e-commerce product classification model is constructed through deep learning technology, and Gray and RGB features of convolution and recursive networks are combined when training the classifier to enhance feature expression. Finally, the applicability and efficiency of related models are verified through performance and functional tests. The convolution kernel is usually represented by a small square. On the feature map, the entire map is scanned sequentially in pixels according to the preset step size. There can be multiple convolution kernels working on a feature map at the same time, and each convolution kernel represents the detection of a certain feature on this feature map. The use of convolution kernel structure greatly reduces the amount of network parameters, speeds up the calculation, and also reduces the risk of overfitting. In the sub-sampling layer, there are usually two forms, maximum pooling sampling and average pooling sampling, which can be regarded as a special convolution process.

The weight of each neuron in the convolution kernel of average pooling sampling is 0.25, and the step size of the convolution kernel is 2 when sliding on the input tensor. The result of mean sampling is equivalent to reducing the input mapping blur to the original $1/4$.

4.2 PROPOSED SYSTEM

Close attention has been paid to represent the wavelengths accurately such that all the essential frequencies in the input images are captured. We choose only small values of λ i.e. from 3 to 8 pixels, since texture on the clothing items is inherently a fine-grained semantic attribute of the cloth and adding responses for larger wavelengths to the set of feature maps does not add value. We further validated the choices for the texture descriptor parameters by conducting experiments on a surrogate task to classify cropped clothing images based upon their texture descriptors alone. Multiresolution analysis is accomplished by varying the neighborhood radius and the number of points to be considered in the circularly symmetric neighborhood. We experiment with radii starting from 2 pixels and extending to 9 pixels and observe a nearly 2% increase in accuracy for the set of multiresolution feature maps comprising of 2, 3, 4 radius values.

To achieve the object of knowledge sharing at the semantic level, we propose a feature map sharing method called structure awareness between the two parallel streams. In specifics, we concatenate feature maps from the end of the landmark detection network with the middle-level feature maps in fashion classification network. The feature maps from the upper stream are directly supervised and contain rich structural information. Being able to be inserted in the middle part of other tasks, these features can interact with other types of features, co-adapt and cooperate, and benefit to the downstream network for feature extraction.

4.3 REQUIREMENT SPECIFICATION

4.3.1. HARDWARE REQUIREMENTS:

- Minimum requirement - 30 GB or more, 10GB or more for foundation.
- Processor - 1.4 GHz 64 bit
- Memory - 512 MB
- Display - (800 * 600) Capable video adapter and monitor.

4.3.2. SOFTWARE REQUIREMENTS:

- Python
- Numpy
- pandas TensorFlow, Keras(Python Libraries)

- Web technologies and bootstrap.

4.4. SOFTWARE DESCRIPTION:

A.PYTHON

- Python is a free, open-source programming language. Therefore, all you have to do is install Python once, and you can start working with it. Not to mention that you can contribute your own code to the community. Python is also a cross-platform compatible language. So, what does this mean? Well, you can install and run Python on several operating systems.
- Python is also a great visualization tool. It provides libraries such as Matplotlib, seaborn and bokeh to create stunning visualizations.

B.PANDAS

- Pandas is a popular Python package for data science, and with good reason: it offers powerful, expressive and flexible data structures that make data manipulation and analysis easy, among many other things. The Data Frame is one of these structures. Pandas is a high-level data manipulation tool developed by Wes McKinney. It is built on the Numpy package and its key data structure is called the Data Frame. Data Frames allow you to store and manipulate tabular data in rows of observations and columns of variables.
- Pandas is built on top of the NumPy package, meaning a lot of the structure of NumPy is used or replicated in Pandas. Data in pandas is often used to feed statistical analysis in SciPy, plotting functions from Matplotlib, and machine learning algorithms in Scikit-learn. There are two types of data structures in pandas: Series and Data Frames.
 - a) Series: A pandas Series is a one dimensional data structure
 - b) Data Frame: A pandas Data Frame is a two (or more) dimensional data structure basically a table with rows and columns.

C.NUMPY

- Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object.

- NumPy is, just like SciPy, Scikit-Learn, Pandas, etc. one of the packages that you just can't miss when you're learning data science, mainly because this library provides you with an array data structure that holds some benefits over Python lists, such as: being more compact, faster access in reading and writing items, being more convenient and more efficient.
- NumPy is a Python library that is the core library for scientific computing in Python. It contains a collection of tools and techniques that can be used to solve on a computer mathematical models of problems in Science and Engineering. One of these tools is a high-performance multidimensional array object that is a powerful data structure for efficient computation of arrays and matrices. To work with these arrays, there's a vast amount of high-level mathematical functions operate on these matrices and arrays.

D.MATHPLOTLIB

- Plotting of data can be extensively made possible in an interactive way by Matplotlib, which is a plotting library that can be demonstrated in Python scripts. Plotting of graphs is a part of data visualization, and this property can be achieved by making use of Matplotlib.
- Matplotlib is a plotting library like GNUplot. The main advantage towards GNUplot is the fact that Matplotlib is a Python module. Due to the growing interest in python the popularity of matplotlib is continually rising as well.
- Another characteristic of matplotlib is its steep learning curve, which means that users usually make rapid progress after having started. The official website has to say the following about this: Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc, with just a few lines of code.
- Matplotlib makes use of many general-purpose GUI toolkits, such as wxPython, Tkinter, QT, etc.,

E. TENSORFLOW

- TensorFlow is an open-source end-to-end platform for creating Machine Learning applications. It is a symbolic math library that uses dataflow and differentiable programming to perform various tasks focused on training and inference of deep neural networks. It allows developers to create machine learning applications using various tools, libraries, and community resources.
- Currently, the most famous deep learning library in the world is Google's TensorFlow. Google product uses machine learning in all of its products to improve the search engine, translation, image captioning or recommendations.
- TensorFlow architecture works in three parts:
 - a) Pre-processing the data
 - b) Build the model
 - c) Train and estimate the model

F. KERAS

- Keras is an Open Source Neural Network library written in Python that runs on top of Theano or TensorFlow. It is designed to be modular, fast and easy to use. It was developed by François Chollet, a Google engineer. Keras doesn't handle low-level computation.
Instead, it uses another library to do it, called the "Backend."
- Keras is high-level API wrapper for the low-level API, capable of running on top of TensorFlow, CNTK, or Theano.
- Keras High-Level API handles the way we make models, defining layers, or set up multiple input-output models. In this level, Keras also compiles our model with loss and optimizer functions, training process with fit function.
- Keras in Python doesn't handle Low-Level API such as making the computational graph, making tensors or other variables because it has been handled by the "backend" engine.

4.5. SYSTEM ARCHITECTURE:

An architecture diagram is a graphical representation of a set of concepts, that are part of an architecture, including their principles, elements and components. It is an important tool as it provides an overall view of the physical deployment of the software system and its evolution roadmap.

It consists of :

- a) Product data evaluation
- b) Pre – processing the data
- c) Image pre – processing
- d) Feature extraction
- e) Training the model
- f) Product category prediction

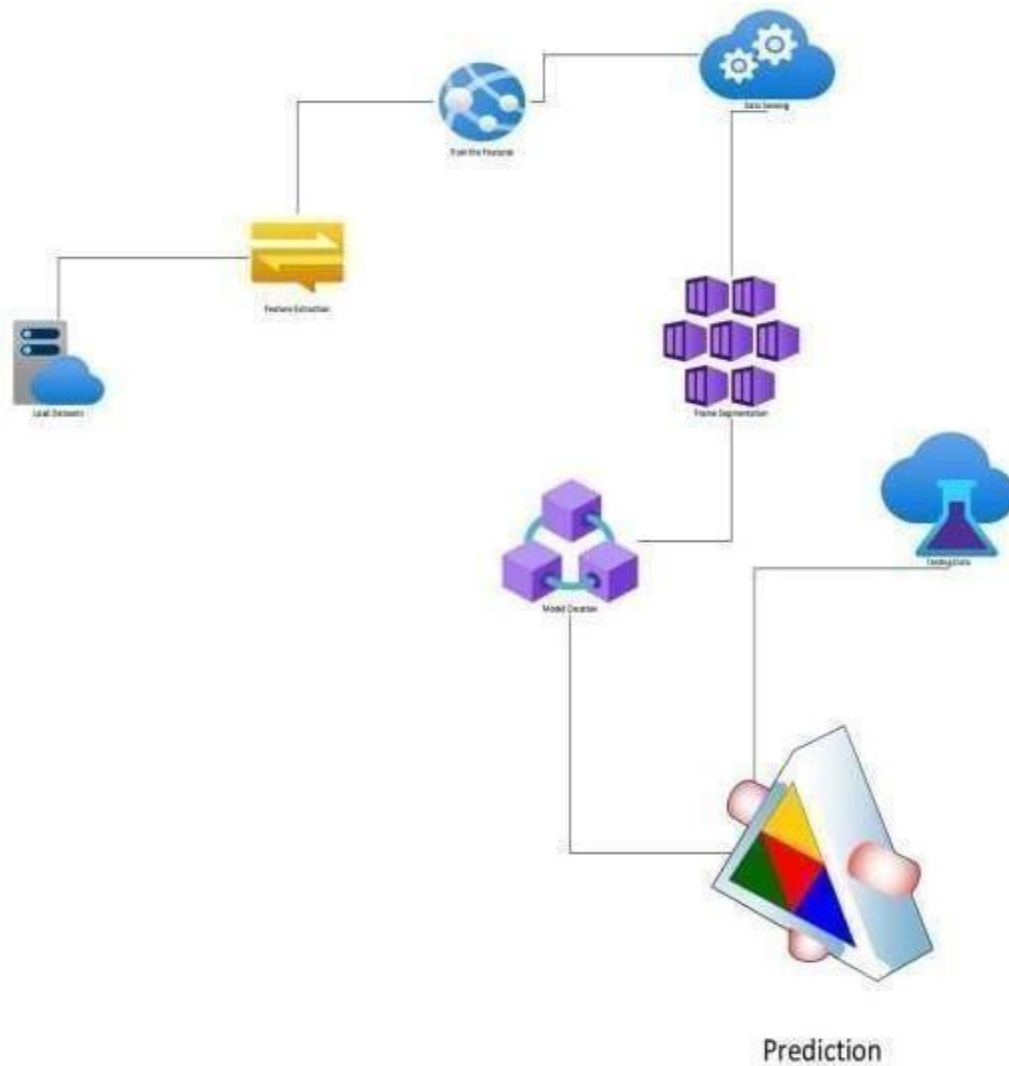


Fig 1.3. Architecture

4.6 SYSTEM IMPLEMENTATION

4.6.1. Product data Evaluation

- The primary purpose of Data Evaluation is to examine a dataset without making any assumptions about what it might contain. By leaving assumptions at the door, researchers and data analysts can recognize patterns and potential causes for observed behaviors.
- This ultimately helps to answer a particular question of interest or to inform decisions about which statistical model would be best to use in later stages of data analysis. Exploratory data analysis is used to validate technical and business assumptions, and to identify patterns.
- The assumptions that analysts tend to make about raw datasets can be placed in one of two categories — technical assumptions and business assumptions.
- In order to maintain confidence that the most optimal analytical models and algorithms are used in data analysis, and that the resulting findings are indeed accurate, specific technical assumptions about the data must be correct.

4.6.2. Pre-processing the dataset

- A dataset can be viewed as a collection of data objects, which are often also called as a records, points, vectors, patterns, events, cases, samples, observations, or entities.
- Data objects are described by a number of features, that capture the basic characteristics of an object, such as the mass of a physical object or the time at which an event occurred, etc.. Features are often called as variables, characteristics, fields, attributes, or dimensions.
- Features can be
 - a) Categorical : Features whose values are taken from a defined set of values. For instance, days in a week :
{Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday} is a category because its value is

always taken from this set.

- b) Numerical :** Features whose values are continuous or integer-valued. They are represented by numbers and possess most of the properties of numbers. For instance, number of steps you walk in a day, or the speed at which you are driving your car at image.

4.6.3. Image Pre processing

- Image pre-processing is the term for operations on images at the lowest level of abstraction. These operations do not increase image information content but they decrease it if entropy is an information measure. The aim of pre-processing is an improvement of the image data that suppresses undesired distortions or enhances some image features relevant for further processing and analysis task.
- Brightness transformations modify pixel brightness and the transformation depends on the properties of a pixel itself. In PBT, output pixel's value depends only on the corresponding input pixel value. Examples of such operators include brightness and contrast adjustments as well as color correction and transformations.
- Contrast enhancement is an important area in image processing for both human and computer vision. It is widely used for medical image processing and as a pre-processing step in speech recognition, texture synthesis, and many other image/video processing applications.
- Image augmentation was performed to the training dataset. Specifically, augmentations were performed in memory and the results were not saved, the so-called just-in-time augmentation that is now the standard way for using the approach.

4.6.4. Feature Extraction

- The main idea behind the HOG method is to express the image as a group of local histograms.

- This histogram consists of gradient orientation and magnitude.
- This technique calculates the gradient value in a particular area of an image. Each image has the characteristics indicated by the gradient distribution. This characteristic is obtained by dividing the image into a small area called a cell.
- Each cell is arranged a histogram of a gradient. The combination of this histogram is used as a descriptor that represents an object.
- To use a sobel filter in one or two directions, either horizontally or vertically. Next is to create parts called cells. Each pixel in a cell has its own histogram value based on the value generated in the gradient calculation.

4.6.5. Training the model

- Feature selection methods are intended to reduce the number of input variables to those that are believed to be most useful to a model in order to predict the target variable.
- Some predictive modeling problems have a large number of variables that can slow the development and training of models and require a large amount of system memory. Additionally, the performance of some models can degrade when including input variables that are not relevant to the target variable.
- Wrapper feature selection methods create many models with different subsets of input features and select those features that result in the best performing model according to a performance metric. These methods are unconcerned with the variable types, although they can be computationally expensive.
- It is common to use correlation type statistical measures between input and output variables as the basis for filter feature selection. As such, the choice of statistical measures is highly dependent upon the variable data types.
- Common data types include numerical (such as height) and

categorical (such as a label), although each may be further subdivided such as integer and floating point for numerical variables, and boolean, ordinal, or nominal for categorical variables.

4.6.6. Product category Prediction

- One of the most important problems when considering the training of models is the tension between optimization and generalization. Optimization is the process of adjusting a model to get the best performance possible on training data (the learning process).
- Generalization is how well the model performs on unseen data. The goal is to obtain the best generalization ability.
- At the beginning of training, those two issues are correlated, the lower the loss on training data, the lower the loss on test data. This happens while the model is still underfitted.
- But, after a number of iterations on the training data, generalization stops to improve and the validation metrics freeze first, and then start to degrade.
- The model is starting to overfit: it has learned so well the training data that has learned patterns that are too specific to training data and irrelevant to new data.
- Cells have a size of 4×4 pixels in an image while the block has a size of 2×2 cell or 8×8 pixels. Gabor filter is one filter that can simulate the characteristics of the human visual system in isolating the frequency and specific orientation of the image.
- This characteristic makes Gabor Filter appropriate for texture recognition applications in computer vision. Spatially, a Gabor function is a sinusoid that is modulated by the Gauss function.
- We also choose our base networks among these networks with a notable consideration: since we aim at using our

method for real-time vehicle applications, the most important criterion to choose our base network is the accuracy/speed trade-off.

- Given an input image I , a channel C of I is a feature map, where the output pixels are computed from corresponding pixels of the input image.
- Aggregated channel features are extracted from multiple image channels using pixel lookup method. Many image channels are available for extracting features.
- For example, a trivial channel of a grayscale image is the image itself. For a color image, each color channel can be used as a channel. Other channels can be computed using various transformations of I .
- In order to accelerate the speed of feature extraction, all transformations are required to be translational invariant. It means that the transformation needs only to be evaluated once on the entire image rather than separately for each overlapping detection window.
- Spatial pooling is used to combine multiple visual descriptors obtained at nearby locations into a lower dimensional descriptor over the pooling region. We follow the work of which is shown that pooling can enhance.
- The robustness of two hand-crafted low-level features, covariance features.
- A covariance matrix is a positive semidefinite matrix which provides a measure of the relationship between multiple sets of variates.
- The diagonal elements of a covariance matrix represent the variance of each feature and non-diagonal elements represent the correlation between different features.
- The spatial invariance and robustness of the covariance descriptors can be improved by applying pooling method.

There are two common pooling methods in this context: average pooling and max pooling.

- Max pooling is used in our framework as it has been shown to outperform average pooling in image classification. Max pooling uses the maximum value of a pooling region to represent the pooled features in the region.
- It aims to retain the most salient information and discard irrelevant details and noises over the pooling region. The image window is divided into multiple dense patches. Covariance features are computed over pixels within each patch. Then, we perform max pooling over a fixed-size pooling region and use the pooled features to represent the covariance features in the pooling region.
- In fact, multiple covariance matrices within each pooling region are summarized into a single matrix which has better invariance to image deformation and translation. The pooled features extracted from each pooling region is called the spatially pooled covariance.

CHAPTER 5

RESULTS AND DISCUSSION

we use two CNN models to classify the HS codes according to text and image. Then we use the proposed adaptive fusion method to improve the performance of customs classification. Finally, we conduct a comparison experiment to show the advantage of our model. In this study, the overall accuracy refers to the classification accuracy of all the products in test set; precision is the correct ratio among products classified into a certain HS code; recall is the correct ratio among products belonging to a certain HS code; and F-measure considers precision and recall by calculating their harmonic average. In practice, we only check the correctness of the products classified into a certain class. However, this does not mean that we do not care about the products misplaced into other classes. Thus, we use the overall accuracy and the F-measure as performance metrics in this study. This project is mainly useful for the e-commerce companies to predict whether a product is getting churned or not. So that it makes easy for them to predict the reason behind the chums and improve the quality of products. We implemented the deep learning algorithms where that is CNN

. We will get various accuracies for each epoch. There are 10 epochs for each epoch there are 100 items and also consists of batch size for an epoch and it train the model and well get various accuracies for each attributes and also model loss and model accuracy for each attribute. Therefore, the highest accuracy for model loss is 80% and for the highest accuracy for model accuracy id 99.8%.

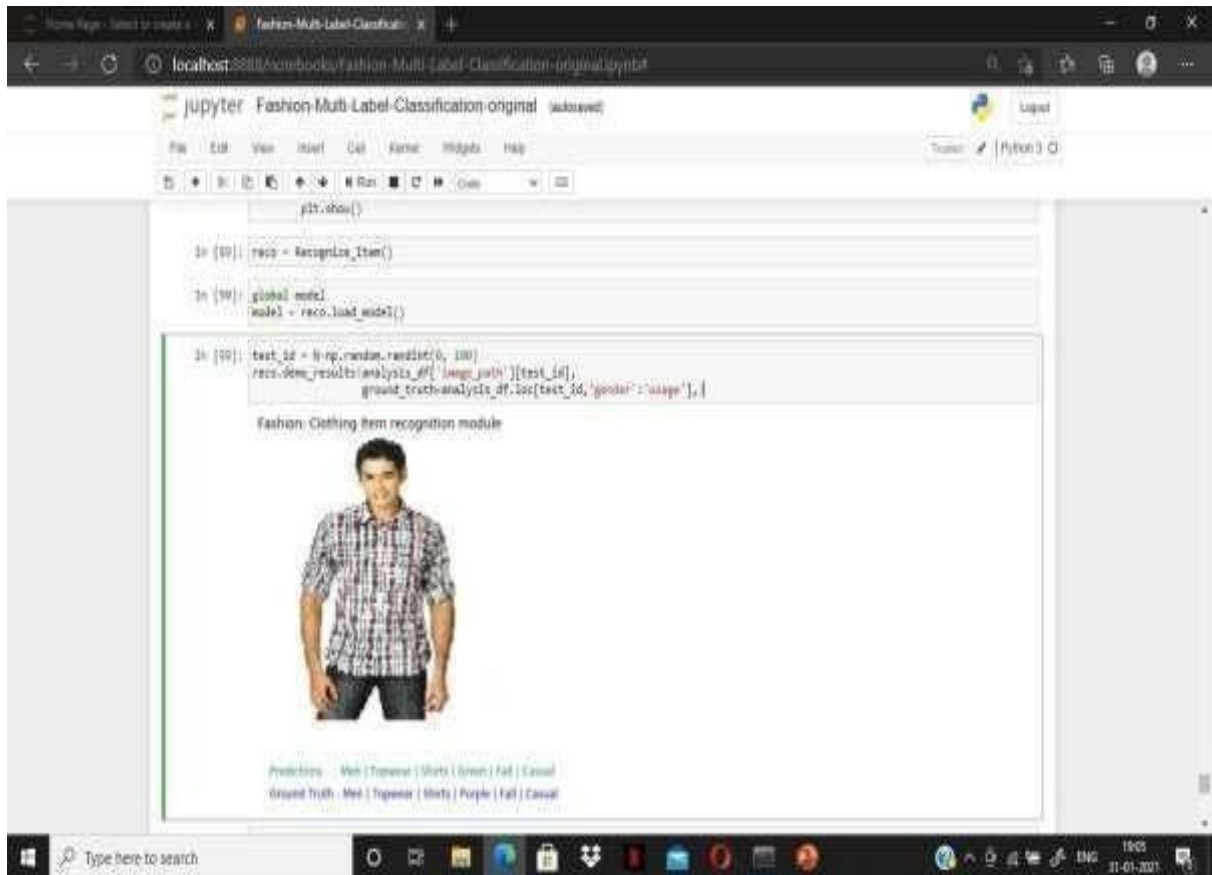


fig 1.4: output for Men's clothing

```

test_id = N-np.random.randint(0, 100) reco.demo_results(analysis_df[ 'image_path' ][test_id],
ground_truth=analysis_df.loc[test_id, 'articleType':'usage'],)

```

From the above figure 1.5 the output, we get to know about the prediction of an image. That is: Gender of the image, product id ,color, season to wear the product and etc., For above output :

Predictions : Men | Top-wear | shirts | white | Fall | Formal.

Ground Truth : Shirts | white | Fall | Formal.

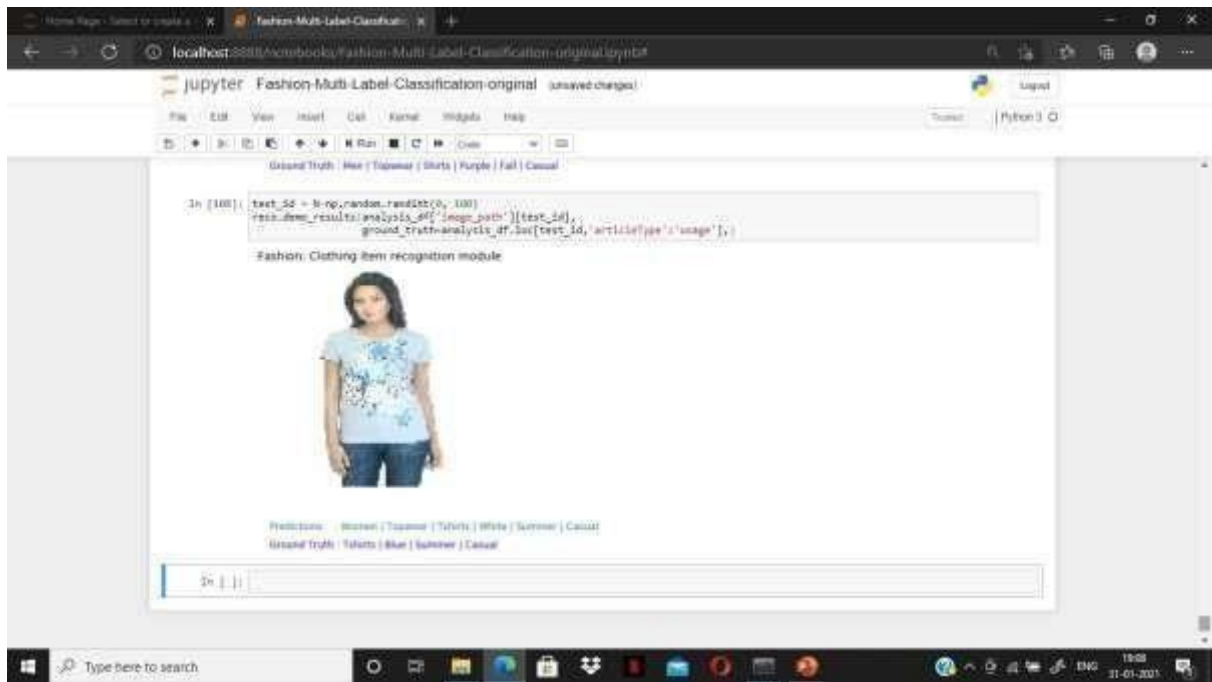


fig 1.5 :output for women's clothing

From this above figure 1.5 output, we get to know about the prediction of an image. That is: Gender of the image, product id ,color, season to wear the product and etc., For above output :

Displays the Women's clothing type that means the prediction of image.

Prediction : Women | Top-wear | T-shirt | sky blue |Fall |

Casual.

Ground Truth : T-shirt | sky-blue | Fall | Casual.

CHAPTER 6

CONCLUSION AND FUTURE ENHANCEMENT

6.1. CONCLUSION

In this research, we present a two-stream multi-task network for fashion recognition. Our model integrated the proposed knowledge sharing methods and outperforms previous methods on fashion landmark detection, category prediction, and attribute estimation.

By studying the current situation of cross-border e-commerce and analyzing its characteristics and shortcomings, this article proposes a new type of cross-border ecommerce product classification model. This technology can provide convenient and efficient use efficiency for e-commerce business. E-commerce platform is the carrier of cross-border e-commerce, and product classification plays a very important role in the E-commerce platform.

Aiming at the shortcomings of existing product classification, this paper develops a new classification technology. A product classification model was developed by combining image recognition and deep learning technology. The applicability and efficiency of related models are verified through performance and function tests. This technology can effectively improve the rationality of the functional design of the ecommerce platform, thereby improving the efficiency of cross-border commerce business. The advantages of cross-border e-commerce cannot be demonstrated either. At the same time, the development of cross-border e-commerce has vigorously promoted the transformation and development of e-commerce platforms, and the development of e-commerce platforms has enhanced the advantages of cross-border e-commerce. At the same time, limited to the shortcomings of the current level, the recognition rate and packet loss rate of this technology will be further studied. We will be committed to the in-depth development of cross-border e-commerce platform technology, and provide scientific reference for the development of cross-border ecommerce business.

6.2. FUTURE ENHANCEMENT

Current growth of DL research and industry applications demonstrate its “ubiquitous” presence in every facet of AI be it NLP or computer vision applications. With time and research opportunities, unsupervised learning methods may deliver models that will closely mimic human behavior. Deep Learning technology’s limitations in being able to reason is a hindrance to automated, decision – support tools. The apparent conflict between consumer data protection laws and research needs of high volumes of consumer data will continue. Google’s acquisition of deep mind technologies hold promise for global marketers . The future ML and DL technologies must demonstrate learning from limited training materials, and transfer between contexts, continuous learning, and adaptive capabilities to remain useful.

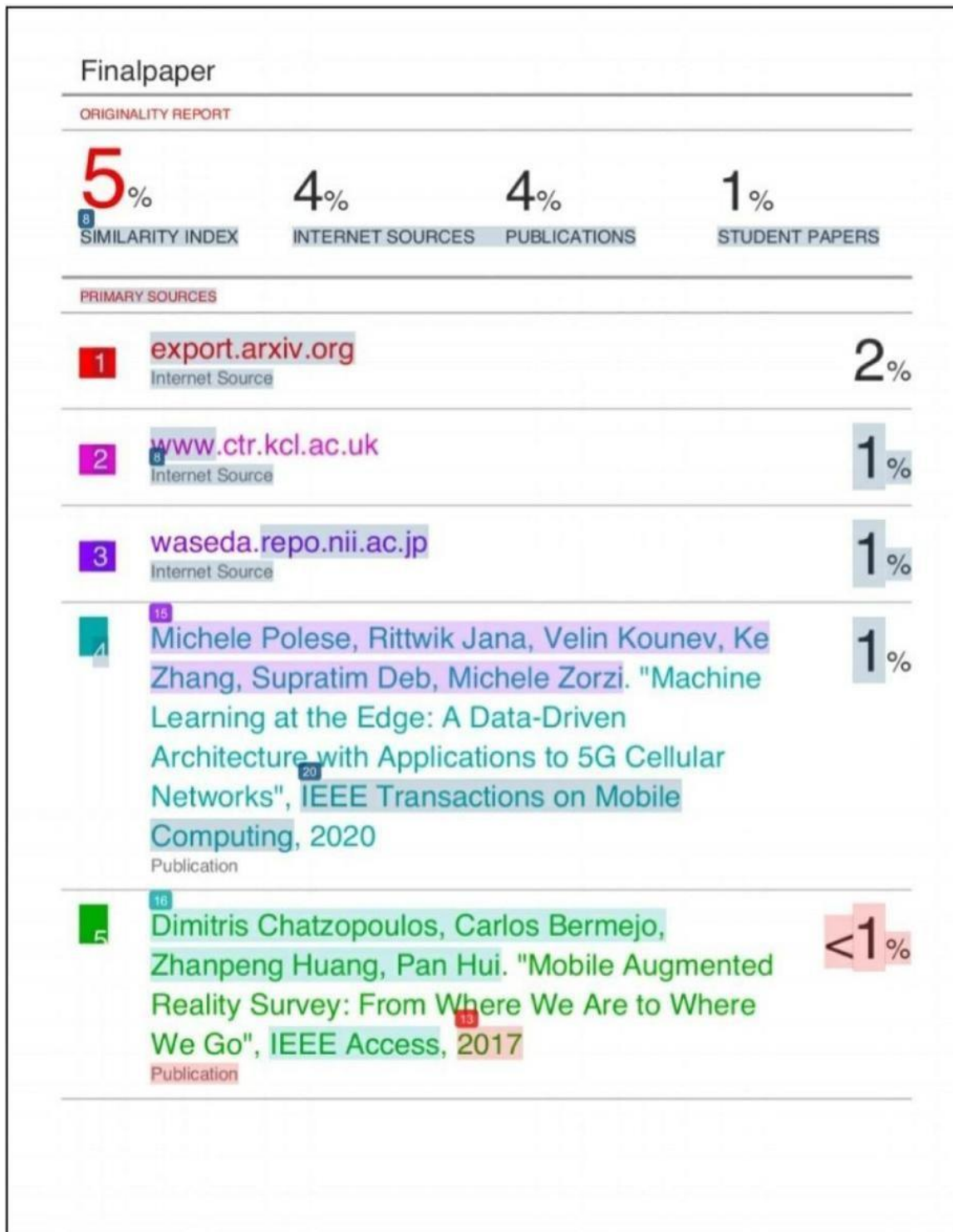
Through globally popular, DL may not be the only savior of AI solutions. If DL technology research progress in the current pace, developers may soon find themselves outpaced and will be forced to take intensive training. In this project, we present a two-stream multi- task network for fashion recognition. Our model integrated the proposed knowledge sharing methods and outperforms previous methods on fashion landmark detection, category prediction, and attribute estimation. The method can also be extended to other challenging multi – tasking problems, and is also applicable to many computer vision related tasks. However, how to improve the accuracy of classification more greatly needs to be further studied in the aspect.

REFERNECES

- [1] Evan Shelhamer, Jonathan Long, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," TPAMI, vol. 39, no. 4, pp. 640–651, 2017.
- [2] Ke Gong, Xiaodan Liang, Dongyu Zhang, Xiaohui Shen, and Liang Lin, "Look into person: Selfsupervised structure-sensitive learning and a new benchmark for human parsing," in CVPR, 2017, pp. 932– 940.
- [3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," TPAMI, 2017.
- [4] N. Shabairou, et al., "Color image identification and reconstruction using artificial neural networks on multimode fiber images: towards an all- optical design," Optics Letters, vol. 43,no. 22, pp. 5603-5606, 2018.
- [5] Singhrao, et al., "An Image Processing Technique for Identification of Interstitial Catheters in High- and Low-Field MR-Guided HDR Brachytherapy," Medical Physics, vol. 6,no. 7, pp. 423-436, 2018.
- [6] Valarezo, et al., "Drivers and barriers to cross-border e-commerce: Evidence from Spanish individual behavior," Telecommunications Policy, vol. 42,no. 6, pp. 464-473, 2018.
- [7] Valarezo, et al., "Drivers and barriers to cross-border e-commerce: Evidence from Spanish individual behavior," Telecommunications Policy, vol. 17,no. 8, pp. 133-144, 2018.
- [8] Wang, et al., "Study of traffic characteristics based on internet web real-time traffic conditions by image identification technology," Imaging Science Journal, vol. 2,no. 11, pp. 223-234,2018.
- [9] X. Wang, J. Xie, and Z.-P. Fan, "B2C cross-border E-commerce logistics mode selection considering product returns," International Journal of Production Research, no. 1, pp. 1-20, 2020.

APPENDIX

A. PLAGIARISM REPORT



B. JOURNAL PAPER

Cross Border E-Commerce Product Based On Image Recognition And Deep Learning

N. Meghana
Department of Computer Science and Engineering
Sathyabama Institute of science and Technology, Chennai, India
meghananuli123@gmail.com

V. Sai Mounika
Department of Computer Science and Engineering
Sathyabama Institute of Science and Technology, Chennai, India
Saimounikavattumalli@gmail.com

Dr. Lakshmanan
Department of Computer Science and Engineering
Sathyabama Institute of Science and Technology, Chennai, India
lakshmanan.cse@gmail.com

meghananuli123@gmail.com

Abstract –

We focus on the problem of parsing fashion images for detecting various types of clothing and styles. The current state of the art techniques for the problem are mostly based on variations of the seg net model. The techniques formulate the problem as segmentation and typically rely on geometrical shapes and position to segment the image. However, specifically for fashion images, each clothing items is made up of specific type of materials with characteristics visual texture patterns. Exploiting the texture for recognizing the clothing type is an another important clue which has been ignored so far by the state of the art. In this paper, we propose a two – stream deep neural network architecture for fashion image parsing.

Keywords – Data analytics , Predict the category of product.

I. INTRODUCTION

We do have conducted some experiments on the bidirectional message passing between the parallel network. And we observed that the nature sharing to the e-commerce product. Deep Learning has evolved hand in hand with the digital era, which has brought about an explosion of data in all forms and every region of the world. This data, simply known as big data.

Deep Learning is a machine learning technique that teaches computers to do what comes naturally to humans. A computer model learns to perform classification tasks directly from images, text, or sound. It achieves state of art accuracy, sometimes exceeding human level performance.

Deep Learning is a computer software that mimics the network of neurons in a brain. It is a subset of machine learning and it is called deep learning because it makes use of deep neural networks. Deep Learning algorithms are conducted with connected layers.

They are:

- The first layer is called input layer.
- The second layer is called output layer.
- All layers in between are called hidden layers.



Fig 1.1 Deep Learning algorithm

Deep Learning mainly consists of 5 steps. They are:

- Understand the problem
- Data Identification
- Select algorithm
- Train the model
- Test the model.

By using all these methods or steps we can predict or classify the given problem. And also we will obtain maximum accuracy of the products like image or text or sound.

Here we will use CNN algorithm. That means Convolutional Neural Network also called as CNN or ConVet, it is a deep learning algorithm which can take input images and assign importance to various aspects in the image and be able to differentiate one from the other.

i. Aim of Project :

The aim of project is to build intuition and understanding of how neural networks can be trained to identify the micro category of a product using its images. And also for identifying an unified and robust pre-segmentation processing for fashion image identification using deep learning algorithms.

ii. Scope of Project :

The main purpose of doing this project is:

- Building user friendly application where user can easier interact easily.
- User can get all the reviews of search query.
- It is also used to build a real time application.

IV. METHODOLOGY

Studying the literature is the most important step in the software development process.

With all these things, the next step is to figure

development methodology, the designer needs to make a contention.

Consumer to consumer (C2C) e-commerce

exchange. It concludes that seller generated product descriptions can mitigate mistrust.

C2C e-commerce and TPC researchers agree that mistrust can be mitigated by detailed content, and they call for platform designers to help improve platform content, and they call for platform designers to improve platforms and sellers reputations.

Researchers methodology : Four platforms were chosen using specific criteria. Product descriptions were collected once per week of six weeks, generating 1900 product descriptions. These descriptions were utilized and given reliable content categories, a methodology called quantitative analysis. This study invites technical and professional communicators to provide more guidelines for users about the kinds of content they may include, and designers to explore the content entry process using usability and user-experience research.

These customer engagement techniques might encourage a seller that facilitates higher quality content among these users.

III. SYSTEM IMPLEMENTATION

A. Existing

Cross-border e-commerce is a process

which is conducive to improving the people's material living standards and quality, and is also

product classification plays a very important role in the e-commerce platform.

to be deemed by the e-commerce platform as

product classification plays a very important role in the e-commerce platform.

To achieve the object of knowledge sharing at the semantic level, we propose a feature map sharing method called structure awareness between the two parallel streams.

Deep learning have some standard algorithms that are used mostly for classifications both in alphanumerical forms. In this project we considered 40k+ entries from different attributes of gender, id, master-category, sub-category, season, year, usage, article type, base color and product display name.

The project architecture has been articulated below;

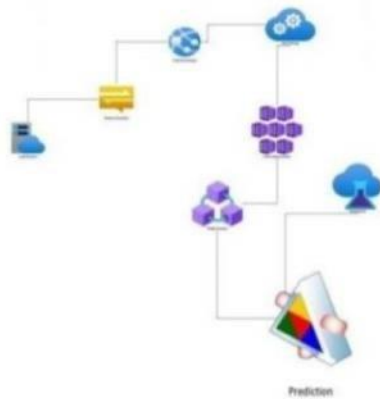


Fig 1.2 System Architecture

C. Advantages of Proposed system :

- Computationally efficient.
- Captures the special features from an image.
- Have the ability to learn by themselves and product the output that is not limited to the input provided to them.

D. Implementation :

The modules included in the project are:

• Data Pre-processing:

Performance is based on the analysis contained in this section of the sample dataset. In this module we will filter the data by removing all the null values and performing several activities like encoding to make the data set clean and machine understandable format.

• Data Analytics:

Analytical data with simple data analysis to pronounce the words. Plant data manufacturing extraction process carrying extracted data for examination. Data analysis helps improve network system performance, capacity and availability . The Business-to-Business process may vary depending on the requirements.

• Model Selection:

In this module we will implement three algorithms named:

1. CNN Model

After creating CNN model we will get total number of data points and labels.
Data Points : 10269

Labels : gender, sub-category, article-Type, base-color, usage and season

After creating we will train the model then we will get sample product id and image shape. Then we will train all labels.

For example : If we train gender label

Gender :: [men, women, boys]

Target shape :: (9242, 3)

Similarly, we will do for all labels. After training for predicting the data we will use classification method in that we will use epochs and batch size for classification the data and to obtain an accurate accuracy of the data.

• Product Category Prediction:

In product category prediction we will batch normalization and pooling layers. And we will predict the image by using sample image which we trained in training part.

• Testing the data:

For testing we have to use reco() method, which means we have to recognize the model which we used in prediction one, then we will get output.

```
test_id=N-np.random.randint(0,100)
reco.demo_results=( )
```

This is the method we will use for testing the data.

M. RESULT

This project is mainly useful for the e-commerce companies to predict whether a product is getting churned or not. So that it makes easy for improve the quality of products.

We implemented the deep learning algorithms where that is CNN. We will get various accuracies for each epoch. There are 10 epochs for each epoch there are 100 items and it train

Prediction : Women | Top-wear | T-shirt | sky
bin fidl|Ce*sJ.

Qex*d Tnab : T*s*t | •kt-bin IFsy t Cw4.

V. CONCLUSION

In this research, we present a two – stream multi task network for fashion recognition. Our model integrated the proposed knowledge sharing the methods and outperforms previous methods on



FIG 1.1 Men's clothing type

Predsc*ai* : hgea | Tag+-web' | +tdnc | ahñe |

Go rdl Tnidt : Shhs* | mjcYr | FeN | Fxxtaal.



Fig 1.3 Displays the Women's clothing type

[14] Trevor Darrell, "Fully convolutional networks for semantic segmentation," TPAMI, vol. 39, no. 4, pp.640-651, 2017.

[18]
[2]

deep convolutional nets, atrous convolutional and fully connected crfs,

[3] N. shabairou, et al., " An image

and reconstructing using artificial

no.22, pp. 5603-5606, 2018.

[4]
[5]

C. SOURCE CODE

```
import warnings
warnings.filterwarnings("ignore")
import os
import json
import ipywidgets as widgets
from IPython.core.display import HTML
import numpy as np
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
from collections import Counter, defaultdict
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.layers import Input, Dense, Dropout, Conv2D,
MaxPooling2D
from tensorflow.keras.layers import BatchNormalization, add,
GlobalAveragePooling2D
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.utils import plot_model
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
images_folder = "images/"
fashion_info = "fashion_info.csv"
df = pd.read_csv(fashion_info, error_bad_lines=False, warn_bad_lines=False)
print("fashion_info file contains {0} rows with {1} attributes".format(len(df),
```

```

len(df.keys()))
print("The Attributes are ")
print(df.keys().values)
df.info()
df.head() image_path
= []
for img_id in df["id"]:
    tmp = images_folder + str(img_id) + ".jpg" if
    os.path.exists(tmp):
        image_path.append(tmp)
    else:
        df.drop(df[df.id == img_id].index, inplace=True)
        print("Image {0} does not exist in the {1} folder ".format(img_id,
images_folder))
df["image_path"] = image_path
df.dropna(inplace=True)
df.reset_index(drop=True, inplace=True)
df.head()
df.tail()
df.sample(frac=1, random_state=10)
HTML("<h1      style='color:green;text-decoration:underline;text-
align:center;margin:35pt;'>Exploratory  Data  Analysis</h1>")
df.nunique()
fig = px.pie(df, names="gender", title="Number of men's Apparals vs Number of
women's Apparals",)
fig.show()
category_counts = pd.value_counts(df["masterCategory"])
fig = go.Figure([go.Bar(x=category_counts.index, y=category_counts.values ,
text=category_counts.values,marker_color='darkGreen')])
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.show()
seasons=pd.value_counts(df['season']) fig
= go.Figure(data=[go.Scatter(
    x=seasons.index, y=seasons.values,

```

```

mode='markers',
marker=dict(
    color=['rgb(93, 164, 214)', 'rgb(255, 144, 14)',
           'rgb(44, 160, 101)', 'rgb(255, 65, 54)'],
    opacity=[1, 0.8, 0.6, 0.4],
    size=[40, 60, 80, 100])
)]
    )
fig.show()
articles=pd.value_counts(df['articleType'])
fig = go.Figure([go.Bar(x=articles.index, y=articles.values ,
text=articles.values,marker_color='indianred')])
fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.show()
df = df[df['masterCategory'] == 'Apparel'] df.drop_duplicates(subset='image_path',
inplace=True) df.drop(columns=['masterCategory', 'year', 'productDisplayName'],
inplace=True) df.reset_index(inplace=True, drop=True)
df.tail(5)
df.head()
def reduce_df(df, category, class_): print('Analysing
    {} as feature'.format(category)) print('Prior
    Number of datapoints: {} :: Classes:
    {}'.format(len(df),set(df[category])))
    df = df[df[category].isin(class_)]
    df.reset_index(drop=True, inplace=True) print('Posterior
    Number of datapoints: {} :: Classes:
    {} \n'.format(len(df),set(df[category])))
    return df
def disp_category(category, df, min_items=500): dff
    = df.groupby(category)
    class_ = dff.count().sort_values(by='id')['id'].reset_index()
    class_.columns=[category,'count']
    class_ = class_[class_['count']>=min_items][category]

```

```

df = df[df[category].isin(class_)]
labels = df[category]
counts = defaultdict(int)
for l in labels:
    counts[l] += 1
counts_df = pd.DataFrame.from_dict(counts, orient='index')
counts_df.columns = ['count']
counts_df.sort_values('count', ascending=False, inplace=True) fig, ax
= plt.subplots()
ax = sns.barplot(x=counts_df.index, y=counts_df['count'], ax=ax)
fig.set_size_inches(10,5)
ax.set_xticklabels(ax.xaxis.get_majorticklabels(), rotation=-90);
ax.set_title('Label: '+category.upper())
plt.show() return
class_
for cate in df.keys()[1:-1]:
    class_ = disp_category(cate, df, min_items=700) df =
    reduce_df(df, cate, class_)
df.nunique()
def grid_plot(title, grouped_df, samples=2):
    samples= len(grouped_df)
    item_id, img_path = grouped_df['id'].values, grouped_df['image_path'].values
    plt.figure(figsize=(7,15))
    for i in range(samples):
        plt.subplot(len(item_id) / samples + 1, 3, i + 1, title=title)
        plt.axis('off')
        plt.imshow(mpimg.imread(img_path[i]))
for cat in df.keys()[1:-1]:
    tmp_df = df.groupby(cat)
    for group_objects in tmp_df:
        title = '{}:\n {}'.format(cat,group_objects[0])
        grouped_df = group_objects[1].sample(n=3)
        grid_plot(title, grouped_df)
df.head()

```

```

HTML("<h1 style='color:green;text-decoration:underline;text-align:center;margin:35pt;'>Create CNN Model</h1>")
analysis_df = df.sample(frac=0.95, random_state=10)
analysis_df.reset_index(drop=True, inplace=True)
labels = analysis_df.keys()[1:-1].values
N = len(analysis_df)
print("Total number of Data_points {} \n Labels {}".format(N, labels))
randm = np.random.randint(0, N)
img = mpimg.imread(analysis_df['image_path'][randm])
plt.figure(figsize=(7,7))
plt.imshow(img)
plt.title(str(analysis_df[labels].values[randm]))
plt.xlabel('Product_id :: {} \n Image shape :: {}'.format(str(analysis_df['id'][randm]),
img.shape))
plt.show()
def load_image(path, shape=(112,112,3)):
    image_list = np.zeros((len(path), shape[0], shape[1], shape[2]))
    for i, fig in enumerate(path):
        img = image.load_img(fig, color_mode='rgb', target_size=shape)
        x = image.img_to_array(img).astype('float16')
        x = x / 255.0
        image_list[i] = x
    return image_list
def load_attr(df, attr, N=None, det=False, one_hot=True):
    le = LabelEncoder()
    le.fit(df[attr])
    target = le.transform(df[attr])
    if N is None:
        N = len(set(target))
    if one_hot:
        target = to_categorical(target, num_classes=N)
    if det:
        print('\n {}: {}'.format(attr, le.classes_))
        print('Target shape', target.shape)

```

```

    return le.classes_, N, target t =
int(0.9*N)
x_train = load_image(analysis_df['image_path'][:t])
gen_names, num_gen, gender_tr = load_attr(analysis_df[:t], 'gender', det=True)
sea_names, num_sea, season_tr = load_attr(analysis_df[:t], 'season', det=True)
use_names, num_use, usage_tr = load_attr(analysis_df[:t], 'usage', det=True)
col_names, num_col, baseColour_tr = load_attr(analysis_df[:t], 'baseColour', det=True)
art_names, num_art, articleType_tr = load_attr(analysis_df[:t], 'articleType',
det=True)
sub_names, num_sub, subCategory_tr = load_attr(analysis_df[:t], 'subCategory',
det=True)
x_val = load_image(analysis_df['image_path'][t:-100])
_, _, gender_val = load_attr(analysis_df[t:-100], 'gender', N=num_gen)
_, _, subCategory_val = load_attr(analysis_df[t:-100], 'subCategory', N=num_sub)
_, _, articleType_val = load_attr(analysis_df[t:-100], 'articleType', N=num_art)
_, _, baseColour_val = load_attr(analysis_df[t:-100], 'baseColour', N=num_col)
_, _, season_val = load_attr(analysis_df[t:-100], 'season', N=num_sea)
_, _, usage_val = load_attr(analysis_df[t:-100], 'usage', N=num_use) dict_
= {'gen_names' : gen_names.tolist(),
    'sub_names' : sub_names.tolist(),
    'art_names' : art_names.tolist(),
    'col_names' : col_names.tolist(),
    'sea_names' : sea_names.tolist(),
    'use_names' : use_names.tolist()}
json.dump(dict_, open('label_map.json', 'w'))
print("\n Distinct classes (Per label):',num_gen, num_sub, num_art, num_col,
num_sea, num_use)
print('Shape:: Train: {}, Val: {}'.format(x_train.shape, x_val.shape))
class Classifier():
    def init (self, input_shape=(112,112,3), pre_model=None):
        self.img_shape = input_shape
        if pre_model is None:
            self.cnn_model = self.build_model()

```

```

else:
    try:
        self.cnn_model = load_model(pre_model)
    except OSError:
        print('Unable to load {}'.format(pre_model))
self.cnn_model.compile(loss='categorical_crossentropy',
                        optimizer='adam',
                        metrics=['accuracy'])
plot_model(self.cnn_model, to_file='consolidated_model.png') self.cnn_model.summary()
def custom_residual_block(self, r, channels): r =
    BatchNormalization()(r)
    r = Conv2D(channels, (3, 3), activation='relu')(r)
    h = Conv2D(channels, (3, 3), padding='same', activation='relu')(r) h =
    Conv2D(channels, (3, 3), padding='same', activation='relu')(h) h =
    Conv2D(channels, (3, 3), padding='same', activation='relu')(h) return
    add([h, r])
def build_model(self):
    input_layer = Input(shape=self.img_shape)
    en_in = Conv2D(16, (5, 5), activation='relu')(input_layer) h =
    MaxPooling2D((3, 3))(en_in)
    h = self.custom_residual_block(h, 32) h =
    self.custom_residual_block(h, 32) h =
    self.custom_residual_block(h, 48) h =
    self.custom_residual_block(h, 48) h =
    self.custom_residual_block(h, 48) h =
    self.custom_residual_block(h, 48) h =
    self.custom_residual_block(h, 54) h =
    self.custom_residual_block(h, 54)
    en_out = self.custom_residual_block(h, 54) gen =
    self.custom_residual_block(en_out, 48) gen =
    GlobalAveragePooling2D()(gen)
    gen = Dense(100, activation='relu')(gen) gen
    = Dropout(0.1)(gen)

```

```

gen = Dense(50, activation='relu')(gen)
gen_out = Dense(num_gen, activation='softmax', name= 'gen_out')(gen) sub =
self.custom_residual_block(en_out, 48)
sub = GlobalAveragePooling2D()(sub) sub =
Dense(100, activation='relu')(sub) sub =
Dropout(0.1)(sub)
sub = Dense(50, activation='relu')(sub)
sub_out = Dense(num_sub, activation='softmax', name= 'sub_out')(sub) art =
self.custom_residual_block(en_out, 48)
art = GlobalAveragePooling2D()(art) art =
Dense(100, activation='relu')(art) art =
Dropout(0.1)(art)
art = Dense(50, activation='relu')(art)
art_out = Dense(num_art, activation='softmax', name= 'art_out')(art) col =
self.custom_residual_block(en_out, 48)
col = GlobalAveragePooling2D()(col) col =
Dense(100, activation='relu')(col) col =
Dropout(0.1)(col)
col = Dense(50, activation='relu')(col)
col_out = Dense(num_col, activation='softmax', name= 'col_out')(col) sea =
self.custom_residual_block(en_out, 48)
sea = GlobalAveragePooling2D()(sea) sea =
Dense(100, activation='relu')(sea) sea =
Dropout(0.1)(sea)
sea = Dense(50, activation='relu')(sea)
sea_out = Dense(num_sea, activation='softmax', name= 'sea_out')(sea) use =
self.custom_residual_block(en_out, 48)
use = GlobalAveragePooling2D()(use) use =
Dense(100, activation='relu')(use) use =
Dropout(0.1)(use)
use = Dense(50, activation='relu')(use)
use_out = Dense(num_use, activation='softmax', name='use_out')(use) return
Model(input_layer, [gen_out, sub_out, art_out, col_out, sea_out,
use_out])

```

```

def train_model(self, x_train, y_train, x_val, y_val, save_path, epochs=25,
batch_size=16):
    early_stopping = EarlyStopping(monitor='val_loss', min_delta=0,
                                   patience=5, verbose=1,
                                   mode='auto', restore_best_weights=True)
    check_pointer = ModelCheckpoint(save_path, monitor='val_loss',
                                   verbose=1, save_best_only=True,
                                   save_weights_only=False,
                                   mode='auto', period=1)
    history = self.cnn_model.fit(x_train, y_train,
                                batch_size=batch_size,
                                epochs=epochs,
                                validation_data=(x_val, y_val),
                                callbacks=[early_stopping, check_pointer])

    return history

def eval_model(self, x_test):
    preds = self.cnn_model.predict(x_test)
    return preds

ce = Classifier(pre_model=None)
history = ce.train_model(x_train, [gender_tr, subCategory_tr, articleType_tr,
                                   baseColour_tr, season_tr, usage_tr],
                        x_val, [gender_val, subCategory_val, articleType_val,
                                baseColour_val, season_val, usage_val],
                        save_path='consolidated_model.h5',
                        epochs=10,
                        batch_size=128)

def learning_curves(hist):
    plt.style.use('ggplot')
    epochs = range(1, len(hist['loss'])+1)
    colors = ['b', 'g', 'r', 'c', 'm', 'y']
    loss_ = [s for s in hist.keys() if 'loss' in s and 'val' not in s]
    val_loss_ = [s for s in hist.keys() if 'loss' in s and 'val' in s]
    acc_ = [s for s in hist.keys() if 'acc' in s and 'val' not in s]
    val_acc_ = [s for s in hist.keys() if 'acc' in s and 'val' in s]

```

```

plt.figure(1, figsize=(20,10))
for tr, val ,c in zip(loss_, val_loss_, colors):
    plt.plot(epochs, hist[tr], c,
              label='train_{ } : {}'.format(tr, str(format(hist[tr][-1],'.3f'))))
    plt.plot(epochs, hist[val], '-.'+c,
              label='{ } : {}'.format(val, str(format(hist[val][-1],'.3f'))))
plt.title('Model Loss (Distinct labels)')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend(loc='upper right')
plt.show()

plt.figure(2, figsize=(20,10))
for tr, val, c in zip(acc_, val_acc_, colors):
    plt.plot(epochs, hist[tr], c,
              label='train_{ } : {}'.format(tr, str(format(hist[tr][-1],'.3f'))))
    plt.plot(epochs, hist[val], '-.'+c,
              label='{ } : {}'.format(val, str(format(hist[val][-1],'.3f'))))
plt.title('Model Accuracy (Distinct labels)') plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend(loc='lower right')
plt.show()

learning_curves(history.history)

class Recognize_Item(object):
    def __init__(self, model_path='consolidated_model.h5'):
        self.model_path = model_path
    def load_model(self):
        try:
            item_reco_model = load_model(self.model_path)
            return item_reco_model
        except Exception:
            raise Exception('#-----Failed to load model file          #')
    def process_img(self, image_path):
        img = image.load_img(image_path, color_mode='rgb',

```

```

target_size=(112,112,3))

    x = image.img_to_array(img).astype('float32')
    return x / 255.0

def tmp_fn(self, one_hot_labels):
    flatten_labels = []
    for i in range(len(one_hot_labels)): flatten_labels.append(np.argmax(one_hot_labels[i],
        axis=-1)[0])
    return self.class_map(flatten_labels) def
class_map(self, e):
    label_map = json.load(open('label_map.json'))
    dict_ = {'gen_names' : gen_names.tolist(),
'sub_names' : sub_names.tolist(),
'art_names' : art_names.tolist(),
'col_names' : col_names.tolist(),
'sea_names' : sea_names.tolist(),
'use_names' : use_names.tolist()} gender =
label_map['gen_names'][e[0]]
subCategory = label_map['sub_names'][e[1]]
articleType = label_map['art_names'][e[2]]
baseColour = label_map['col_names'][e[3]]
season = label_map['sea_names'][e[4]] usage =
label_map['use_names'][e[5]]
    return [gender, subCategory, articleType, baseColour, season, usage] def
predict_all(self, model, image):
    x_image = np.expand_dims(image, axis=0)
    return self.tmp_fn(model.predict(x_image))
def demo_results(self, image_path, ground_truth=None):
    image = self.process_img(image_path) predicted_labels =
self.predict_all(model, image) plt.figure(figsize=(5,7))
plt.title('Fashion: Clothing Item recognition module')
plt.text(0, 133, str('Predictions      : '+' | '.join(predicted_labels)),
        fontsize=12, color='teal')
    if ground_truth is None:

```

```
        pass
    else:
        plt.text(0, 142, str('Ground Truth : '+' | '.join(ground_truth)),
                 fontsize=12, color='b')
    plt.axis('off')
    plt.imshow(image)
    plt.show()
reco = Recognize_Item()
global model
model = reco.load_model()
test_id = N-np.random.randint(0, 100) reco.demo_results(analysis_df['image_path'][test_id],
                 ground_truth=analysis_df.loc[test_id,'gender':'usage'],)
```