Title	Image Preference Estimation with Word Embedding Model and Convolutional Neural Network
Author(s)	万, 樺
Citation	
Issue Date	2021-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/17149
Rights	
Description	Supervisor: 長谷川 忍, 先端科学技術研究科, 修士(情報科学)

JAIST

JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

Master's Thesis

Image Preference Estimation with Word Embedding Model and Convolutional Neural Network

WAN HUA

Supervisor HASEGAWA SHINOBU

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March 2021

Abstract

To improve the traditional image recommendation system and classification methods, we propose measuring a distance between two vectorized representations: **User Preference Vector (UPV)** and **Image Classification Vector (ICV)**. Both vectors are obtained by our designed procedures together with the calculation formula. For the purpose of individual preferences toward images into consideration, instead of a conventional approach which focus on explore features and attributes only appear within particular contents, along with whole processing procedure lack of combination with contemporary techniques, such as Natural Language Processing (NLP) for embed user keywords, as well as probabilities draw from the deep neural network concerning each label also fulling into deliberation.

Regarding our experiment procedures, the creation of **UPV** can be divide into several steps:

i. Construct an original website based on Python programming language and FLASK framework, combining with an original database to collect and save user inputs as the plain natural language recourses.

ii. Derive the data from the website and pass them into the data cleaning section, which is the text preprocessing for raw text data, such as lower the capital letters, removing punctuations and whitespaces, tokenization, etc.

iii. Pass the data being cleaned into the word embedding model to output each vectorized representation for user preferences.

iv. Calculate the converted vectors based on assigned weights as well as designed formula to procure the **UPV**.

On the other hand, Convolutional Neural Network (CNN) takes place in the creation of **ICV**:

i. Apply the Cifar-10 image dataset (a total of 60000 images and 10 labels) to train the neural network, which is constructed by the Python language and Keras framework in a cloud environment.

ii. Feed the model by an original image dataset (Kawaii dataset, a total of 30 images, divided into 3 branches), records the probabilities of each image come

by the output layer (output of SoftMax function), which is an array telling us the probabilities concerning each image being categorized into all labels.

iii.   Pass the probabilities into our secondary designed formula to procure the **ICV**.

After obtaining both **UPV** and **ICV**, we calculate the distance between those vectors to digitize individual preference toward images. Based on our hypothesis, a small distance as the output draw from our proposed model represent a higher priority for our respondents. Contemplation of justification as well as evaluation for the purposed model, we conduct the Single User Estimation together with Multiuser Estimation as our preliminary experiment at the beginning of the evaluation section. In Single User Estimation, our proposed model works well and distinguished image priority toward User ID.01. However, in the case of Multiuser Estimation, an partial prediction for User ID.03 appeared.

Conducive to evaluate our proposed model properly, we determined a further investigation in the sense of both scientific viewpoints together with user-oriented viewpoint. Specifically, we designed an inquiry form with the same content (10 image sets, each including 3 images pick up from the Kawaii dataset, along with a table of keywords) for all respondents. Later, we introduce d the statistic approach toward User ID.04 and understood that the assigned weights for images and system output (distance) were statistically insignificant (R-value equals 0.24 but P-value ≈ 0.19 which is large than 0.05). Besides, we also apply other correlation analysis methods to calculate the results and draw the regression line for the User ID.04.

Subsequently, we continue to distribute the inquire form and collect information until User ID.07 and apply several data analysis methods to the output data raised by our experiment, such as correlation (statistic approach notifies that the assigned weight and output distance is statistically insignificant, P-value equals to $0.27 > 0.05$), logistic regression (75% accuracy), draw confusion matrix, F-measure, etc. As the results of this research, a total of 43 samples being tested (3 samples draw from the preliminary experiment and 40 samples draw form inquire from). 8 of the samples are perfectly hit (18.6% accuracy), and 21 of the sample identified that our proposed model affected somehow (Partially correct, floating from 18.6% ~ 67.4%). Nevertheless, we also have a total of 14 samples of image sets that could not properly distinguish by our proposed model. Overall, after we checked the assigned weights from our respondents, comparing with the calculation results (distance), we understand that the user performance in the

sense of image priority assignment gives a significant influence on the results.

Furthermore, we also understand several weaknesses regarding our proposed model. Such as we should update the Kawaii dataset closer to the training dataset instead of 2/3 branches are not affected, in the sense of images can be learned and distinguished by the convolutional neural network. We should also update our policy regarding the inquiry form, which notifies our respondent keywords and image priorities should be related. Lastly, scientific approaches such as correlation and linear/logistic regression could not make much sense for experiment results. We should also consider other approaches.

At last, this research investigated a digitalized representation between **UPV** and **ICV**. Which gives inspiration to whose works in image recommendation and classification problems. We believe that all procedures combined in this experiment, such as website and database for data collection. Word embedding and data cleaning for text processing, neural network, distance calculation following our design formula, etc. The flow that appears in this experiment will be a ignite.

Keywords: **User Preference Vector (UPV), Image Classification Vector (ICV)**, Distance Calculation, Word Embedding, Convolutional Neural Network, Kawaii Dataset.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Problem Statement

With the widespread use of smart devices, taking and sharing pictures has become an imperative component for us to delight our daily lives. On the other hand, an enormous amount of unclassified images inside smart devices quickly press storage capacity. Contemporary electronic merchants also face a situation that displays the most proper image to a particular user, in the sense of commercial advertising. The image classification and recommendation technique are the fundamental convention approach to resolve this series of problems. However, the existing image classification methods mainly highlighted the accuracy of the classification performance for not an individual viewpoint but a whole dataset [4][5]. Besides, the previous image recommendation system focused on image-based exploration for user preference, such as photo distribution [3]. None of them dealt with text (Natural Language) representation as to the elements of user preference. Furthermore, accompanied by the thirdly rising of artificial intelligence, although similarities of the words are considered in recent years, proper parameters and distance calculation methods between user and images remain unexplored in the image recommendation system.

1.2 Research Objects

The purpose of this research is to propose an approach aiming to retrieve an understandable distance between human oriented inner priority and computer output. Essentially, this research expects to be designed to serve the purpose of the reinforcement, toward a traditional image classification task and photographic recommendation task, under the circumstance of individual being considered. We believe that it is feasible to embed several keywords as the representation for a particular user, as the parameter passing into the mathematic formula, combining with another parameter, represents the image vector extracted from the neural network. Our hypothesis is that we can calculate

the distance between a user-oriented vector and an image-oriented vector. The digital number as the outcome should be the observation, revealing how close this particular image to this particular user. Therefore, it is not difficult to implement sorting methods toward distance sequence afterward, as the improvement of the image recommendation system.

1.3 Research Approach/Research Questions

This research is mainly divided into three different phases. In the first phase, we focus on collecting and preprocessing natural language resources (keywords) and drawing from a certain number of respondents (target user) through a designed website. In the second phase, we proposed a deep learning model trained by the Cifar-10 dataset for the image classification task. In the final phase, we apply a statistical approach as well as a data analysis procedure for the experiment results as the evaluation section. In detail, we describe each different phase as below, which also corresponds to our research questions.

RQ1: How to prepare suitable datasets with proper pre-processing methods?
Dataset preparation is divided into two different steps. In the first step, we collect the user preferences through the website, respondents are expecting to have different culture background, which means the recorded keywords express a diverse priority of concerning as the statement of their own favor. Subsequently, we apply a conventional text cleaning section to convert multiple keywords into vector representation **User Preference Vector (UPV)** as the first parameter for conducting the distance calculation. Regarding the specific techniques in this step, we carefully record all the keywords and sentences as natural language resources via a built dataset, apply normalization, lemmatization, etc..., to dealing with raw text data. Also, we remove meaningless whitespace, symbols, as well as stop words, which convenience us to apply word2vec (mainly GloVe) for vectorization. In the second step, we build and train the convolutional neural network, passing an original image dataset called Kawaii dataset (gathering of an appropriate number of images and intentionally divide by several branches) into the model, output the possibilities of each image being categorized through the SoftMax function in the output layer of the neural network. We calculate the **Image Classification Vector (ICV)** as the second parameter for distance calculation with a designed formula based on the possibilities. Due to the Convolutional Neural Network (CNN) model trained by the

prestigious image dataset (Cifar-10), we do not need a preprocessing section for the training images as it has already been done.

RQ2: What is the role **UPV** and **ICV** take in this research?

UPV and **ICV** are the essential parameters for calculating the distance, in the order word, measuring the interval between those two parameters is our key concept as well as the motivation of the experiment. Our hypothesis assumes that it is possible to create a believable interspace between an individual user and computer output. It allows us to clarify how close a particular image toward a particular user, making sense for improving traditional recommendation systems with formal sorting algorithms. Details regarding our design, the form of **UPV** and **ICV** is a 50-size vector representation. The former was created from the user preferences through natural language processing (NLP) method, and the latter was created from the CNN model as the classification results. If defining **UPV** and **ICV** in a single sentence**,** digitalized representation of the user prefers and image contents. Following this concept, we can also calculate the breadth between **UPV** and **ICV** through a standardized mathematic approach.

RQ3: How to evaluate the proposed model?

The evaluation section for the proposed model operates through both the scientific viewpoint and human interaction viewpoint. The former considers existing formulas or axiom, aiming to make sense toward **UPV** and **ICV**. For instance, we calculate the correlation between the **UPV** and **ICV** and make efforts to figure out the P-value, which leads us to understand whether experiment results randomly occur. Also, we draw the linear regression line to predict the following numbers and logistic regression line for binary classification, under the idea of supervised learning in the sense of model output the correct distance. All the scientific approach gives us the evidence to contemplate regarding the efficiency of the proposed model deeply. On the other hand, we also conduct a user-oriented survey through interview and inquiry form as the second part of the evaluation section. Unlike the early part of **the UPV** collecting phase (Website), respondents who join the evaluation section are notified regarding the results of distance calculation from the proposed model to justify and confirm user satisfaction. Under a condition of output makes sense as an internal representation of their preferences or not. Besides, we would like to mention here we are selecting the respondents from the university and a varying background covered by multiple nationalities and age distribution in our evaluation section, due to a consideration of the sample respondents' independency.

1.4 Programming Environment

Due to the programming as the most potent tool through all the phases in this research, we need to emphasize the environment and procedures. Python is in charge of most of the tasks because of generalization in the artificial intelligence field and convenience for powerful importable libraries. All tasks include but are not limited to website, database, text-preprocessing, word embedding, convolutional neural network, distance calculation, correlation, linear/logistic regression, etc. Most of the code runs in the cloud environment (Google Colaboratory) because of the comfortable interface and enjoyable coding experience, together with the quality of computing resources. For instance, GPU (Graphics Processing Unit) acceleration for image processing in a deep neural network. On the other hand, the website's construction runs on a local PC owing to the command prompt take the role of a web server for launching. Furthermore, multiple Python flameworks cropped up in this research, such as Keras for deep learning algorithms, Flask for web structure, etc. Along with bunch of the libraries imported mainly under an operating system of Windows 10. Lastly, because of the immature programming skills as well as the understanding of the background theories, the flow of the codes is not optimized.

1.5 Thesis Outline

Following the introduction section, Chapter 2, the background section is mainly divided into two parts. The first part described relative works as the literature review in this research and highlighted the experiment's originalities. On the other hand, the second part of Chapter 2 focuses on the sketch of the background techniques in our experiment.

The method section, Chapter3, digs into detail regarding the curial parameters **User Preference Vector (UPV)** and **Image Classification Vector (ICV)**. We explicitly narrate the whole steps to procure **UPV** and **ICV**, for the distance calculation afterward, together with the comparison of embedding models and classification models. We also introduce the original image dataset (Kawaii dataset), etc.

Chapter4, the evaluation section, includes a scientific approach and a user-oriented approach toward our proposed image preference estimation procedures, predominantly describing the data analysis results through correlation and regression. On the other hand, we describe the distance in detail, aiming to clarify our hypothesis.

Chapter5 is the summary section of this research. We mainly describe the observation and considerations regarding our proposed model as the conclusion part of this section. Subsequently, based on our investigation, we discuss the contribution of our experiment and the future works as the reinforcement for the extent of this topic.

Chapter 2

Background

2.1 Related Works/Originalities

Nguen et al. proposed an optimized feature representation model in an unlabeled dataset combines with a latent factor model (LFM), weighted matrix factorization (WMF), and convolutional neural network (CNN) as a personalized photo recommendation system [1]. Savchenko et al. developed a user preference prediction engine based on scene understanding, object detection, and face recognition, suggesting that text recognition techniques make preference prediction more reliable as future work [2]. Díez et al. extracted personalized information from photos uploaded by individual users to understand users' main attraction and mentioned that synonymy of images would match the user preference in the future [3]. Wang et al. explored web meta-data problems by investigating the numerical representation of web text data, combining with CNN for images [8]. Yao et al. trace an image modeling and classification task using textual and visual features [9]. Chen et al. introduce a new application capable of social media platforms to recommend related images from both local and global image pools [14]. From these discussions, we understand that natural language stands for human expression for elaborating individual preferences, and embedding methods play a significant role in the sense of vectorization for updating image classification tasks. [2][3] reveal the image classification, and recommendation task should combine with contemporary techniques such as artificial intelligence, following this idea, even though the previous research [8] considers merging the text data from the web with the CNN model and proposing a useful framework improve image classification task. Still, individual existence is not appearing or considered a strong position for the conventional engagement of image classification.

Therefore, originalities in our research reflecting in serval aspect which previous work could not cover:

1. We are exploring the text data from the website and constructing an original homepage for the respondents.
2. We created an original image dataset (Kawaii dataset) for the CNN classification procedure with three branches drawn from our own consideration.
3. We calculate the distance between an individual and system output following mathematical formula.
4. We apply serval data analysis techniques for our initial experiment results as a different approach to the image classification and recommendation task compares with the previous work.

2.2 Technical Background

2.2.1 Text Preprocessing

Data preprocessing is an essential step for reducing irrelevant information for an unreliable dataset, aiming to grab a predictable and analyzable form. Under this concept, text preprocessing in natural language processing is a preparation procedure for cleaning the raw text data for further analysis. Nowadays, Python libraries such as NLTK (Steven Bird, 2001) and re (built-in package) are in charge of the text preprocessing task.

1. Converting capital letters to lowercase.
This step is usually recognized as the first step for text preprocessing. During this step, all capital letters in a string are expecting to convert into lower case. In python3.7, we use the lower function to achieve this goal.

2. Removing punctuations and whitespaces.
Symbols, such as [!"#$%&'()*+,-./:;<=>?@[¥]^_`{|}~]: have no meaning for most of the text analysis task, string punctuation function will help us remove all of the symbols to lighten the string's volume. As the same reason for moving punctuation in our text dataset, whitespaces also take no place for our analysis.

3. Tokenization

Tokenization is the procedure to split the raw text data into smaller pieces. All words, numbers, punctuation marks can be considered as a single token. There are many tools under Python programing language that can serve this purpose, such as Natural language Toolkit (The University of Pennsylvania 2001), Genism (RaRe Technologies 2009), OpenNMT (Yoon kim, harvardnlp, 2016), and Memory-Based Shallow (Vincent Van Asch, Tom De Smedt, 2010).

4. Removing stop words

Stop words are the words that appear in a language with a high frequency. In the case of English, words such as "the," "a," "on," "is," "all," etc. are belong to this categorization. Usually, these words do not carry a significant meaning and do not influence the contexts themselves. In the text preprocessing procedure, it is better to remove stop words as well as symbols and whitespaces. With python3.7, we can import scikit-learn or spacy to check the list and apply NLTK to remove them.

2.2.2 Word Embedding

The term Word Embedding represents a series of language modeling and feature learning techniques in natural language processing. It is the method to convert words from an enormous vocabulary into vectors composed of consecutive digital numbers. For the fulfillment of passing words into calculable mathematic formulas. There are many methods to generate vectors from words, including but not limited to neural networks, dimension reduction for co-occurrence matrix of words, and probabilistic models. Through all the above techniques, the vectors obtained are expecting to explicitly represent the meaning and the contexts carried by the original words or sentences.

Cosine similarity (measuring the cosine angle within two vectors, whether pointing to the same direction or not) is one of the primary methods for calculating the similarities. For instance, assume the giving document1 = "Final exam of natural language processing course." and document2 = "leaving exam of machine learning course." By observing the d1 and d2, record the frequency of each word appear in the corpus, in this case, d1 and d2 have two overlaps (exam, course), the similarity of d1 and d2 is 2. However, this method will disable when we encounter a long document case, for example, d3 = "Elizana

entry the Crystalsong Forest, defeat the monster called a chief machine, before the leaving, she discovers the scrawled note which described a fairy tale regarding the failure of learning the way of Druid transformation." In this case, overlap (similarity) between d2 and d3 is 3. Which is large than the one between d2 and d1, regardless of the apparent fact that d2 and d3 nearly have no similarities. Therefore, an algebra approach takes place.

Calculate by the cosine of two non-zero vectors (Euclidean dot product formula):

$$\mathbf{A} \cdot \mathbf{B} = \|\mathbf{A}\| \, \|\mathbf{B}\| \cos \theta$$

Cosine similarity is represented using a dot product and magnitude as:

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

2.2.3 Global Vectors for Word Representation

GloVe (Jeffrey, Richard, and Christopher, 2014) is an open-source project at Stanford University base on an unsupervised learning algorithm for obtaining vector representation. Trained on non-zero entries of the global word-word co-occurrence matrix, which calculate the frequency of words co-occurs with one another. Mapping by calculating the distance between words in the sense of semantic similarities, resulting in a linear substructure (an enlarged set of discriminative numbers, representing the vector difference)

Co-occurrence can be defined as:
F_ik: Frequency of word *i* and *k* appeared together.
F_i: The total Frequency word *i* appeared in the corpus.
P_ik = F_ik/F_i: The probability of word *i* and *k* together.

Given two words, student (i) and salaryman (j), if the third word(k):
is very similar to student (i) but irrelevant to salaryman (j), For instance, study (k),
P_ik/P_jk will be very high (>1);

is very similar to salaryman (j) but irrelevant to student (i), For instance, work(k), P_ik/P_jk will be very small (<1);
is related or unrelated to either word, then P_ik/P_jk will be close to 1.

*See details regarding the consecutive mathematical formulations listed in the original article of GloVe (Jeffrey, Richard, and Christopher, 2014).

2.2.4 Convolutional Neural Network

Convolutional neural network (CNN) is a deep neural network class commonly used for visual dataset. Construction by multilayer perceptron (MLP), which brings an agreement about each neuron in one layer, is connected to all the other neurons in the next layer.

1. Neuron

Neuron is a mathematical computation, receive the inputs, multiply the weights, passing the summation via activation function into other neurons, for each neuron in neural network, output y can be calculated as: $y = \sigma(z)$, $z = x1w1+x2w2+b$, which $x1 = input1$, $x2 = input2$, $w1 = $ weight for input1, $w2 = $ weight for input2, $b = $ bias, $z = $ weighted input (summation of all inputs and bias), $\sigma = $ activation function. Details in the processing section, x1 and x2 passing into neural networks are usually normalized or standardized for better performance. For instance, when we normalize a picture in range 0~255, we divide the numbers with 255 to obtain a range between 0 and 1. Weights usually initiate (Gaussian Distribution or so) in a small number, bias closes to 0 at the beginning for a better learning accuracy.

2. Pooling

Pooling is used to reduce the dimensions of the feature map, the parameters to learn, and the number of calculations to be performed (intricacy) in the neural network in conjunction with filter (define the size of the region for filter out from the original image) and stride (define the size of the matrix, which means the number of pixels). There are two common methods to conduct the pooling, max pooling output the max number from the determined grid, instead of outputting the maximum number, average polling output the average number from the determined grid.

3. Padding

As a matter of fact, if the filter does not fit the scale to get the information in the sense of each pixel from the original image only one time, we need a padding region to cover the whole image. For instance, if the stride equal to 1 and the size of zero-padding equal to (K-1)/2, then the input/output will always have the same dimensions. *K is the filter size.

4. Activation Function

Activation Functions are crucial components for determining the output and accuracy of neural network base on mathematic equations. In a deep learning model, each neuron's activation function strongly influences the model's outcome. This thesis will mainly discuss ReLU and SoftMax activation function those applied in our neural network.

（1）ReLU (Rectified Linear Unit), Relu(z) = {0 if z < 0; z if z >=0}

Rectified linear in the neural network applies stochastic gradient descent with a backpropagation algorithm to train the model. Compared with a traditional sigmoid function, the ReLU activation function acts as a linear function, but it is a nonlinear function that allows the model to learn a complex dataset. ReLU function has the advantage of converging the model in a comparatively short period, but it also has a disadvantage, which is not differentiable at z = 0. In this research, we apply the ReLU activation function in all neurons except the output layer.

（2）SoftMax (Normalized Exponential Function)

The SoftMax function is also well known as the normalized exponential function. It usually appears as the neural network's last activation function to normalize the model's output to a probability distribution toward each label (class). The final summation of the probability distribution will be unlimited close to 1.

2.2.5 Euclidean Distance

Euclidean distance is one of the methods to calculate the distance in the sense of a mathematical perspective. Nowadays, it is mainly used for calculating the length of two points (or objects) in a Euclidean space, whose result represents a number. Euclidean distance can also be calculated from the Cartesian coordinate, which specifies each point in a plane by a series of numerical coordinates. Euclidean Distance equations come from

the Pythagorean theorem and vary for calculating distance in a different dimensional circumstance.

One dimension:

$$d(p, q) = \sqrt{(p - q)^2}$$

Two dimensions:

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}$$

N dimensions:

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

p and q stand for two points in Euclidean n-space, qi and pi stands for the vectors, starting from the (0,0) and n stands for the n-spaces. Given line1 (coordinate p1, p2, p3) and line2 (coordinate q1, q2, q3) in a 3-dimensional space, the Euclidean distance can be calculated as $\sqrt{(q1-p1)^2 +(q2-p2)^2+(q3-p3)^2}$. For instance, the distance between P (0.9,0.6,0.3) and Q (1.2,1.4,1.6) is $\sqrt{(0.9-1.2)^2+(0.6-1.4)^2+(0.3-1.6)^2} = \sqrt{(0.09+0.64+1.69)} \approx 1.497$.

2.2.6 Correlation Coefficient

Correlation usually identifies the relationships between two or more variables in a statistic and data science aspect. For example, as we have two variables represent in two arrays, both belong to the same dataset. The title of the array is called a feature, and each number or data point they contain called observation. Features are the properties or attributes for the observations. We can analyze any two features of a dataset, and it is possible to find some types of correlation between them.

If we have two arrays x and y represent as points in the coordinate system, the line describes as left up to right down will indicates the strong negative correlation between them, on the other hand, the line describes as left down to right up will indicates the strong

positive correlation between them, which is the reverse line compare to the previous one. Furthermore, a stable horizontal line crosses the coordinate system indicate that there is no obvious trend between x and y which usually represent a weak correlation or hard to observed.

Pearson correlation coefficient

In this research, we mainly use Pearson's r to quantify the correlation, a linear correlation method to measure the linear correlation between two variables x and y between -1 (very strong negative correlation) to 1 (very strong positive correlation).

The formula can be referred to as below:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2 \sum (y_i - \bar{y})^2}}$$

r stands for the correlation coefficient, xi stands for the values of the x-variable in the dataset, x bar stands for the mean value of the x-variable, yi stands for the values of the y-variables in the dataset, and y bar stands for the mean value of the y-variable.

Chapter 3

Method

3.1 Approaches

To proceed with our research, we need to obtain two core variables to conduct this experiment. This section describes the overviews of the variables, namely, **User Preference Vector (UPV)** and **Image Classification Vector (ICV)**.

UPV shows user preferences based on their own personality as well as character traits, mainly created by the following steps from natural language resources, as shown in Figure 3.1.

1. Raw data is collected from the website and stored in the database.
2. Text preprocessing is applied to clean the data.
3. Word embedding model is imported to convert words into a vector representation.
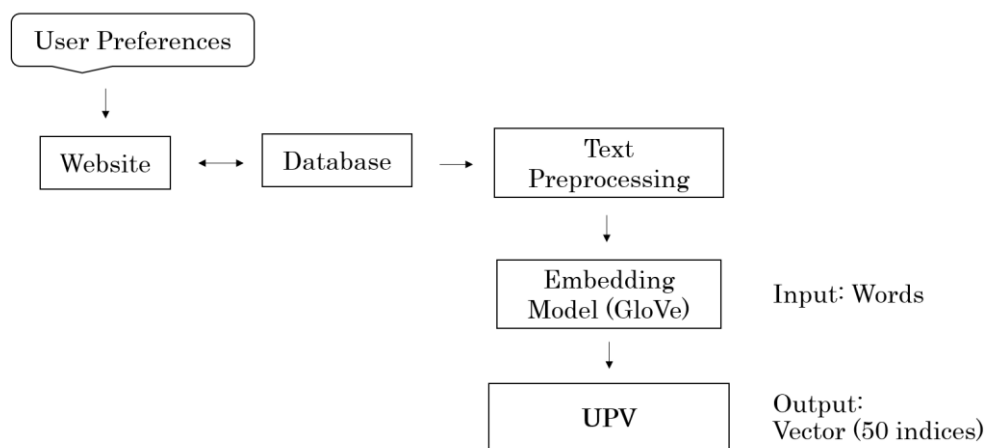

Figure 3.1 Creation of UPV

ICV is initially created by the following steps from our proposed image classification model, regarding the vector representation of each different image in a whole image dataset, as shown in Figure 3.2.

1. We build and train our CNN model fed by the cifar10 image dataset and apply a word embedding model for all the labels of the categories of the cifar10 dataset.
2. We create an original image dataset called the Kawaii dataset, passing all the images into our model and producing all the possibilities of the images classified as the output in our model.
3. We build a formula to calculate the results based on our proposed method to obtain the ICV.
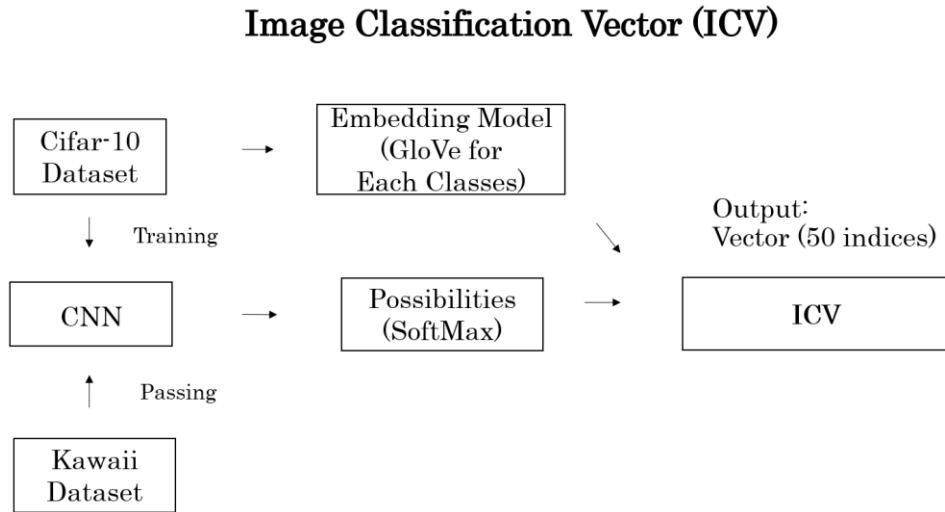

Figure 3.2 Creation of ICV

After obtaining both core variables, we assume to have an insight toward user preference individually by calculating the distance between those variables. The following sections describe the detailed procedures of the proposed vectors.

3.2 User Preference Vector (UPV)

3.2.1 Website

Building the website is the first step for us to obtain the User Preference Vector. After considering online survey services (SurveyMonkey, Google Forms, etc.), we decide to build our own website to collect user preferences for the following reasons. Firstly, a website, designed and performed through web pages, is a powerful tool to bring a holistic view to our respondents, confirming their understanding of the research purpose and procedure. Secondly, the website is also a convenient tool to give the advantages of transparency regarding all the collected information. Lastly, the website has a high-quality performance for the user interface, including user input interface as well as management permission. We decide to give our respondents full permission to edit and check their responses after a normal submission session to update and delete their submissions to the database.

User preferences collected through the website are stored in our database and continued to proceed as raw NLP (Natural language processing) resources. Python is a primary programming language in charge of constructing our website. We chose FLASK (A micro web framework written in Python, Miguel Grinberg 2014) as our programming framework. Besides, we decide to use HTML to design the front web pages, CSS to modify the font size and background color as well as a bootstrap for the decoration in the sense of slot and blocks for user input. Our desktop PC is the server to run the website. Details about the environment setting are as shown in Figure 3.3.


```
C:\Users\mayn\Desktop\website>set FLASK_APP=website.py

C:\Users\mayn\Desktop\website>set FLASK_ENV=development

C:\Users\mayn\Desktop\website>set FLASK_DEBUG=1

C:\Users\mayn\Desktop\website>python -m flask run --host=0.0.0.0
 * Serving Flask app "website.py " (lazy loading)
 * Environment: development
 * Debug mode: on
 * Restarting with stat
C:\Users\mayn\AppData\Local\Packages\PythonSoftwareFoundation.Pyt
ant overhead and will be disabled by default in the future.  Set
  'SQLALCHEMY_TRACK_MODIFICATIONS adds significant overhead and '
 * Debugger is active!
 * Debugger PIN: 943-662-679
 * Running on http://0.0.0.0:5000/ (Press CTRL+C to quit)
```

Figure 3.3 Environment Settings (Web)

Here is the index page of our website (Figure 3.4). We have a Navbar in the top front of our main page (keywords "About our experiment," "Join the Survey," "Quick Response," and "Back to index" which guide our respondents to other subpages respectively to display the extra message of our research or conduct the user preference collecting section.). The image dataset shown below on our index page is a glimpse of the cifar10 dataset, a prestigious image classification dataset. We apply this dataset to train our model described in the image classification section.
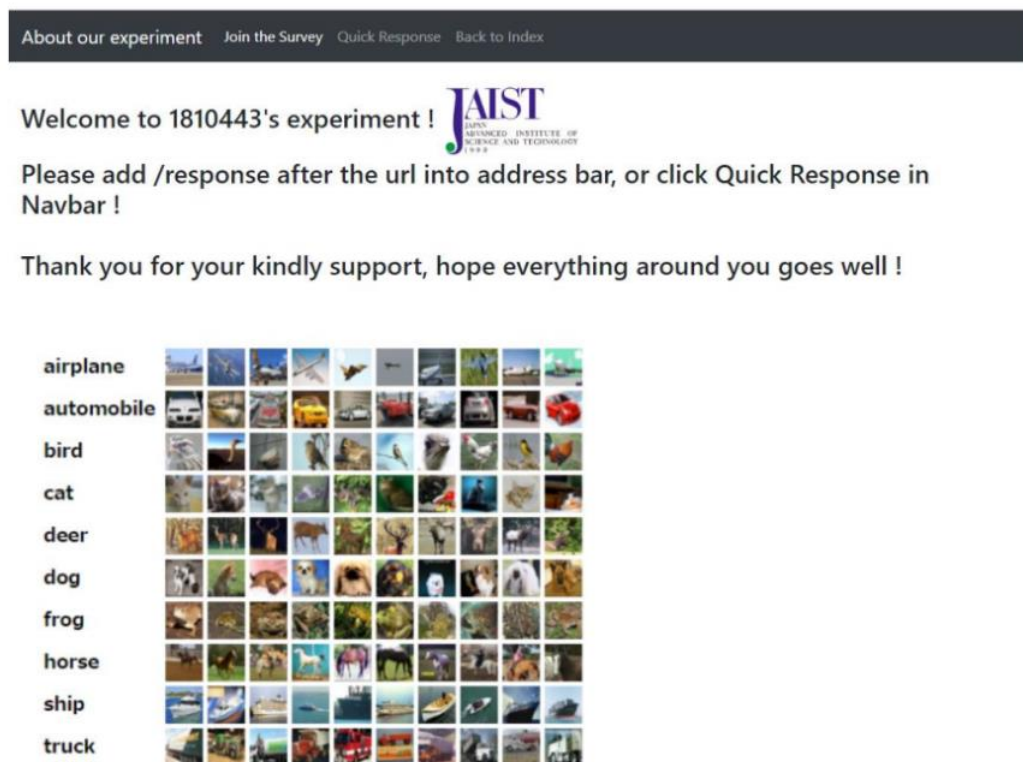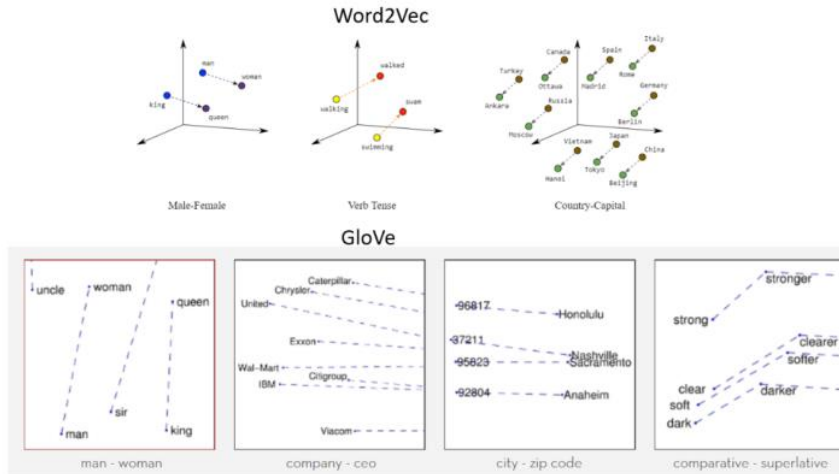

Figure 3.4 Index (Web)

When respondents click the "about our experiment" in Navbar, the linked page will display a brief explanation about our research procedure step by step. (Figure 3.5, Figure 3.6). Our purpose here mainly aims to give the respondents a holistic view of the experiment. Some background figures also appear to serve the purpose of reinforcement regarding technical backgrounds - all the figures citing in our website collected from google images and original articles in our reference. In addition, introductions are shown in Figure 3.5 and Figure 3.6 end up respondent interactive phase, which do not include the evaluation part of the whole experiment (co-relation analysis, linear/logistic regression, etc. toward experiment result).

Figure 3.5 Regarding Experiment 01 (Web)

Figure 3.6 Regarding Experiment 02 (Web)

The "Join the survey" page includes serval blocks, mainly following a bootstrap format for our respondents to input their preferences (Figure 3.7). On this page, we only temporarily track user input in case of users' personal information (name, student number, etc.). We will not record such inputs on this page because we did not link our database to these data. The inputs will automatically disappear after we shut down the server. Regardless of that, we will send an email to our respondents to show our gratitude. We invite them for further collaboration, which is the evaluation part of our experiment after calculating the distance.

Figure 3.7 Survey (Web)

Here is the "Thanks" page returned for our respondents after filling the form (Figure 3.8). This page will also automatically send an email from our default email address to our respondents' email addresses based on our settings. Temporary information displays on this page to confirm our respondents, based on their inputs regarding their preferences.

Thank you for your participanting ! We will pass your preferences into our model and contact with you later.

Calculation formula for preferences : prefer01 (embedding) * weight + prefer02 (embedding) * weight + prefer03 (embedding) * weight

If you are interesting in our research, you can also check the results we collected...

- WanHua : 1810443 ---- cat 30% , dog 70% , AI 0%

Figure 3.8 Thanks (Web)

The "Quick response" page records user inputs as we linked our database with this page (Figure 3.9). Therefore, we have a declaration of policies to make sure the respondents understand the purpose of our information collection procedure. As we mentioned before, our respondents have the full permission to manipulate the database to change or delete their inputs, reflecting in green and red buttons.

Here is the quick response page :
Notice : This page linked to our database which permanently save your input.

Please confirm we will only use your input information for our experiment, you can also update and delete your input message.

☐ Yes, I confrimed

An input format "nickname(Email) - prefer01(weight01), prefer02(weight02)..."
Example : "Elizana (Wayne30443@gmail.com) - Rouge(70%), DemonHunter(20%), Priest(10%)"

Name (Prefer)

Add prefer

WAN - THE ANIMAL (70%), A GAME (20%), SPORTS (10%) Update Delete
Elizana - rouge(70%), demonhunter(20%), priest(10%) Update Delete
quarantine - home(100%) Update Delete
wayne30443 - airplane (100%) Update Delete
Vedovelli - heroine(50%), story(50%) Update Delete
ahlberg - october(70%), december(30%) Update Delete
GRANDPA AI80 AutoMobile12 Tourism8 Update Delete
Euclidean - distance, python, correlation Update Delete

Figure 3.9 Quick Response (Web)

3.2.2 Database

As the main reason for constructing the website is to collect user preferences, preserve and proceed with all information received for word embedding phrases, we also need to build our database capable of storing and maintaining the inputs from our respondents. Otherwise, after we close the server at once, all information will be eliminated. To accomplish this task, we install flask-sqlalchemy through the command prompt, an extension for Flask to add the support for SQLAlchemy to our website. Subsequently, we import SQLAlchemy, which gives us the power and flexibilities to communicate between Python and Database (Specifically, the library used as an object relational mapper (ORM) to translates Python class to a table in the sense of the different type of codes). Later, we apply SQLite databases, swapping out the settings, assign the relative path for the database file. We add a database class inside our source code's main file for constructing the website in our programming section (Figure 3.10).
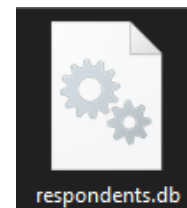
Figure 3.10 Database
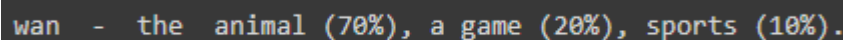
After we got our model created, we need to embody our database file through the command prompt. Firstly, we simply type python3 into the interactive python shell. After that, we import DB and use the commend "db.create_all()" to create the database. As we mention in the website section, we create two different pages to offer an alternative way for respondents. "Quick Response" page will return the user preferences based on the user ids. Our database permanently saves all input as raw natural language resources. The database file reflects a respondents.db file saved into the same path as our website file. Ultimately, as we aim to privilege our respondents to manipulate their inputs, we create the green and red buttons based on bootstrap, allowing them to update and delete their inputs.

22

3.2.3 Text Preprocessing

After we collected respondents' preferences through the website and saved the raw information into the database, text-preprocessing in charge of the data cleaning section, we apply several text-preprocessing techniques to deal with the raw data embedding model. In detail, we mainly apply a standard procedure to clean the strings based on Python3. We have five steps in our text-preprocessing procedure, as we also mention in the background for each of the different steps. As a preparation, we sign up for the account and setting up the environment base on Google Co-laboratory. Which is a well-known data science platform (editor) integrated with both convenience and computing resources. Besides, Google Co-laboratory allows us to run multiple blocks and check the result simultaneously, efficiently, and clarify our programming work with a great leap compared with a run in local PC.

Assuming we have an example user called "Wan", input the preference1, preference2, preference3 with identical weights as THE   ANIMAL (70%), A GAME (20%), SPORTS (10%), the whole sentences containing some random whitespaces and stop words.
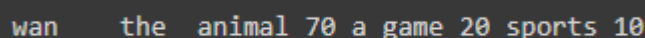
Firstly, we convert the capital letters to lowercase operated by the lower function, create a list to contain all the strings from respondents, and lower each element (strings) in this list. The output of the program for the respondent "Wan" refers, as shown in Figure 3.11:

```
wan  -  the  animal (70%), a game (20%), sports (10%).
```

Figure 3.11 Lower (Text)

Secondly, we are removing the punctuation (symbols) with method translate, method maketrans, etc., as they take no place but confuse our analysis (Figure 3.12). We keep the numbers here compared to a default text-preprocessing section because the numbers assigned by our respondents take an imperative meaning in our experiment.

```
wan    the  animal 70 a game 20 sports 10
```

Figure 3.12 Punctuation (Text)

23

Thirdly, as a similar reason on step 2, we remove the white space with strip, split or replace function due to the different results we want (removing the head and end whitespace, removing duplicate whitespace, or removing all whitespace, etc.) All the functions and library such as re allows us to manipulate the strings to return in an ideal shape. Figure 3.13 shows the result of removing whitespace with re.

```
'wan the animal 70 a game 20 sports 10'
```

Figure 3.13 Remove Whitespace (Text) - re

Fourthly, we tokenize each different word in the sentences aiming to remove the stop words. Most commonly appeared words in sentences but carried no important meanings, Natural Language Toolkit (NLTK) offers the power we need to accomplish this task and scikit-learn or spaCy. Figure 3.14 shows the result of tokenizing and removing stop words from the previous step.

```
['wan', 'animal', '70', 'game', '20', 'sports', '10']
```

Figure 3.14 Stop words (Text) - NLTK.

In addition, we also apply stemming (reducing words to word stem) and lemmatization (find a correct base forms of words) for each different sentence from our respondents (Figure 3.15), as well as Part of Speech tagging (POS) (Figure 3.16), which assign the grammatical tagging to the giving word (nouns, verbs, adjectives, etc.) In the case of our example user "Wan," the stemming and lemmatization results toward preferences (animal, game, and sports) and POS results (Particularly, "NN" stands for noun singular, "NNS" stands for noun plural and "CD" stands for cardinal digit).

```
wan          wan
anim         animal
70           70
game         game
20           20
sport        sport
10           10
```

Figure 3.15 Stemming and lemmatization (Text) - NLTK.

```
[('wan', 'NN'), ('animal', 'NN'), ('70', 'CD'),
```

```
('game', 'NN'), ('20', 'CD'), ('sports', 'NNS'), ('10', 'CD')]
```

Figure 3.16 POS (Text) - TextBlob, NLTK

3.2.4 Word Embedding

Word embedding takes on converting user preferences into a digital format that allows us to pick up the trail to obtain the **UPV.** Specifically, we apply Gensim (an open-source library that collects enormous documents as a corpus, prestigious with parallelized implementations of word2vec as well as doc2vec algorithms) for processing text data in a mathematical convenience representation. Details regarding the models, we tested glove-wiki-gigaword-50 (400000 records), word2vec-google-news-300 (3000000 records), etc., towards the respondents' preferences. The former includes 50 vector size, pre-trained vectors based on Wikipedia 2014 + Gigaword tokens, and the latter includes 300 vector size, trained on the part of the Google News dataset (around 100 billion words). As a description of the example output, we embed the preference "animal" from user "Wan."

```
array([ 0.49652 , -0.65143 , -1.0869  , -0.10205 ,  0.81724 ,
        0.923   , -0.56206 , -1.3801  ,  1.8115  ,  0.068438 ,
        0.63906 ,  0.24468 ,  1.0308  ,  0.10202 ,  0.48498 ,
       -0.08387 ,  0.61688 ,  0.35812 , -0.75196 , -0.3548  ,
       -0.14173 ,  0.042311,  0.42242 , -0.21013 ,  0.28935 ,
       -1.1214  , -0.5278  , -0.046298,  0.064643, -0.43924 ,
        2.4004  , -0.29715 , -0.19765 , -0.88725 , -0.62955 ,
        0.64092 ,  0.14741 , -0.0089431,  0.39569 ,  0.060899,
       -0.33917 , -0.15897 ,  0.22115 ,  0.83813 ,  1.6032  ,
       -0.010252, -0.36843 , -0.32005 ,  0.4658  , -0.10813 ],
      dtype=float32)
```

Figure 3.17 glove-wiki-gigaword-50 ("animal")

Figure 3.18 word2vec-google-news-300 ("animal")

As reviewed above, compare with word2vec-google-news-300 (Figure 3.18), embedding output from glove-wiki-gigaword-50 (Figure 3.17) have the advantage in the sense of simplicity, but a disadvantage in lower precision (less vocabulary for searching and calculate similarities). Apparently, in the word embedding section, if we value the understanding and directness regarding the outputs, we might inevitably lose fidelity and exactitude. Nevertheless, suppose we only focus on accuracy. In that case, we will increase the calculation complexities and have some difficulties to handle the balance, in the sense of the whole progress in our experiment. After testing several embedding models, we choose glove-wiki-gigaword-50 as the primary embedding model to deal with user preference (words) as well as image (labels) in the later CNN phase, due to a consideration of the time cost (download speed, result checking, etc.) and computing complexity (size of the vector, digits after the decimal point)."

3.2.5 Preferences Calculation

We asked the respondents to fill the form with serval preferences during the previous user preference collection phase and attach with a percentage (weight mark) behind the words. In addition, we also deliver the message to the respondents to confirm the summation of total numbers being 100% in our policy paragraph, due to our designed formula to obtain the **UPV:**

$$\sum_{i=1}^{n} Embed(Pi) * Wi$$

As the output vector from the glove-wiki-gigaword-50 model, P stands for user preferences, W stands for the weights toward preference. For each respondent, the whole embedding vector's summation will be recorded as the **UPV**. As an output example, we embed the word "animal," "game," and "sport" from user "Wan" to attach with designed weight "70," "20," and "10," respectively. The final output of **UPV** for user "Wan" calculate as Embedding (animal) * 0.7 + Embedding (game) * 0.2 + Embedding (sport) *0.1, with the summation of weights (0.7+0.2+0.1=1) (Figure 3.19). We need to emphasize here which is we are not limit the numbers of preferences can be filled within quick response page, the total number can be a single idea or multiple consideration, but he final summation of the weights for all preferences needs to be 1. Lastly, in our justification section, we convert the output vector into a list representation and record all indices numbers. For each of them, we calculate the summation of original output from word embedding version of "animal," "game," and "sport" from index 0 to index 49. Compare with an index with a direct output, two of the results have perfectly corresponded.

```
array([ 7.96810091e-02, -4.28549021e-01, -8.98943961e-01,  2.31881022e-01,
        5.77975988e-01,  6.29171968e-01, -7.09243000e-01, -1.03162622e+00,
        1.13244390e+00,  1.71486214e-01,  5.68367004e-01,  1.77790403e-01,
        3.49309981e-01,  9.24420059e-02,  6.48148000e-01, -1.80252999e-01,
        4.84958977e-01,  3.93252015e-01, -9.66466010e-01, -3.73700976e-01,
       -3.58864993e-01, -2.05830112e-03,  2.29584739e-01, -2.31653973e-02,
        2.73483992e-01, -1.11224008e+00, -4.34197038e-01, -4.11139801e-03,
        2.45097339e-01, -5.92326999e-01,  2.59396982e+00,  1.22015014e-01,
       -1.50191993e-01, -7.30292976e-01, -4.33066994e-01,  6.40172005e-01,
        2.12521002e-01,  1.61178842e-01,  1.52527004e-01, -2.01157704e-01,
       -2.30892614e-01, -2.54326999e-01, -3.88901681e-03,  7.43831992e-01,
        9.98484015e-01,  1.76704705e-01, -1.46448985e-01, -1.10783979e-01,
        3.95956606e-01, -6.28059953e-02], dtype=float32)
```

Figure 3.19 UPV for "Wan"

3.3 Image Classification Vector (ICV)

3.3.1 Classification

Until here, we procure the **UPV** as the first variable, in contemplation distance calculation in the later section, we still need the second variable, the **ICV**. Regarding securing this crucial component, we need to build and train the machine learning model capable of image classification.

Firstly, we prepare two different datasets those are Cifar-10 dataset (total amount of images being 60000, 32*32 color pixel, divided by 10 classes and each class take in 6000 images, 5 training batch, and 1 test batch, each includes 10000 images) and Cifar-100 dataset (same with Cifar-10 dataset but divided into 100 ordinary classes and 20 superclasses, each ordinary class includes 600 images). Subsequently, we attempt two different machine learning frameworks for the Cifar-10 and Cifar-100 datasets. Figure 3.20, Figure 3.21 describe the results of the first framework: Core ML (unified machine learning models in Apple products operating system).
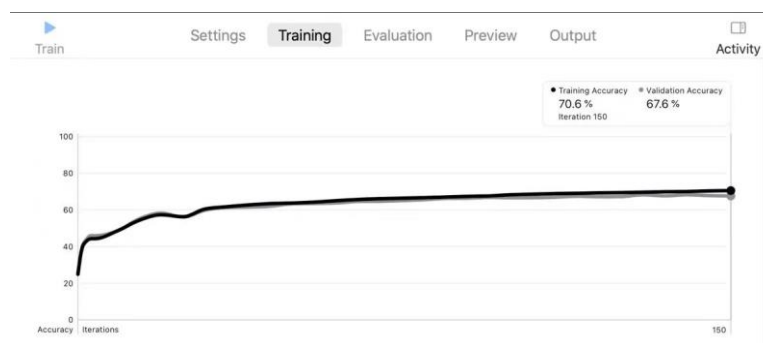
Figure 3.20 Cifar-10 Core ML (Iteration150)

Figure 3.21 Cifar-100 Core ML (Iteration150)

28

As Figures 3.20 and 3.21 shown above, when we train the models through Core ML, Cifar-10 dataset maintains an approximately 70.6% classification accuracy after 150 iteration, which is slightly lower than our expectation (Compare with a common model for distinguishing Cifar-10 dataset). On the other hand, Cifar-100 achieves an 80.5% training accuracy (black line) marvelously after 150 iterations. However, if we pay attention to the validation accuracy (grey line, 49.1% as the outcome), the model has undergone an overfitting circumstance, starting from around 25 iterations (a patent divergence regarding the training accuracy line and validation accuracy line). Due to the consideration of generalization of the model, we subtract the meaningless iteration and record an approximately 49% accuracy run by Core ML in the sense of Cifar-100 dataset. CNN (Convolutional Neural Network) constructed by Keras (an open-source library offering artificial neural networks with Python) in a traditional Windows operating system under Google Colab environment is another endeavor. As Figure 3.22 and Figure 3.23 shown below, the results for Cifar-10 image dataset are 88% training accuracy and 79% validation accuracy after running a 25 epoch. On the other hand, Cifar-100 dataset only performed a 54% training accuracy and 44% validation accuracy due to a 100 categorize distribution toward classes of the images. Both CNN models under a hyperparameter as ReLU activation function for the input and hidden layer and SoftMax function for output layer, 25 epoch, and 32 batch size, etc.

```
Epoch 25/25
1407/1407 [==============================] - 7s 5ms/step - loss: 0.3322 - accuracy: 0.8841
```

```
val_loss: 0.7449 - val_accuracy: 0.7896
```

Figure 3.22 Cifar-10 CNN (Epoch 25)

```
Epoch 25/25
1563/1563 [==============================] - 7s 4ms/step - loss: 1.6283 - accuracy: 0.5438
```

```
val loss: 2.2364604473114014
val accuracy: 0.4377000033855438
```

Figure 3.23 Cifar-100 CNN (Epoch 25)

In conclusion, eventually, we decide to import Cifar-10 dataset as the primary dataset to train the model and Keras for constructing the convolutional neural network. Because Cifar-10 dataset has a higher accuracy comparing with Cifar-100 under a certain generalized rate. On the other hand, Keras performs better transparency and compatibilities in the sense of fine tuning toward hyperparameters, in contrast with Core

ML operated in a mac environment. All the advantages from Keras neural network structure led us to proceed with the subsequent experiments in a later section.

3.3.2 Probabilities

As we understand that the output of the SoftMax function in a neural network (output layer) being the probabilities toward each labels the images being categorized. It is a common event to output the probabilities (total summation will be 1) to investigate further. As an example, Figure 3.24 shown an original image (long-range fighter aircraft Mitsubishi A6M "Zero") (referring from google image), Figure 3.25 and Figure 3.26 shown the probabilities image "zero" being categorized under the model, trained by Cifar-10 and Cifar-100, respectively.

Figure 3.24 zero.jpg (original)
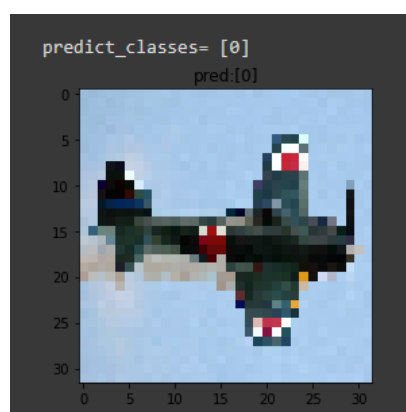

```
array([[9.9997699e-01, 3.7834656e-08, 2.0620482e-07, 3.2060100e-08,
        2.1827734e-10, 1.0871416e-10, 6.2414700e-13, 4.2519183e-08,
        2.8805960e-10, 2.2771503e-05]], dtype=float32)
```

Figure 3.25 Probabilities (Cifar-10)

```
array([[7.79733682e-07, 1.41781529e-05, 8.14446776e-06, 3.83789938e-06,
        6.88698833e-07, 9.84830112e-05, 3.78936849e-04, 3.89519054e-03,
        3.06124017e-02, 7.49016763e-04, 1.52373395e-04, 2.54845563e-05,
        1.15756900e-03, 3.11002019e-03, 5.80678927e-04, 1.41206488e-04,
        7.46928854e-04, 1.90386840e-04, 6.74047551e-05, 7.47259415e-04,
        3.07977432e-04, 1.53712577e-07, 1.02767348e-03, 2.92546681e-10,
        9.29007016e-04, 3.87987122e-04, 2.06337543e-03, 1.12052358e-05,
        3.43368100e-07, 3.69788351e-04, 6.90722061e-07, 2.33074047e-06,
        1.24027749e-04, 3.32079821e-06, 1.84839500e-05, 2.11862953e-05,
        4.70160657e-08, 9.85329898e-05, 5.79990738e-05, 1.04276685e-03,
        1.67445478e-03, 4.42722673e-03, 7.07006620e-06, 1.72172903e-07,
        1.84623976e-04, 4.74074877e-05, 1.30136068e-05, 2.69360607e-05,
        9.09132063e-01, 1.47831054e-06, 1.16574415e-06, 2.32960374e-05,
        1.63675895e-06, 1.30320103e-07, 2.01803836e-04, 5.19156401e-06,
        1.44782331e-04, 6.03920489e-06, 1.49710579e-02, 2.68011104e-06,
        3.68117981e-08, 7.39253301e-06, 4.87361722e-05, 7.72470921e-10,
        4.11366756e-07, 2.35133484e-05, 4.21559438e-04, 2.79028001e-07,
        1.28847805e-05, 8.50984088e-06, 1.26429019e-04, 2.73306378e-09,
        1.77940237e-05, 2.66301413e-07, 3.60073773e-06, 4.44254576e-04,
        1.72552211e-06, 4.67789368e-06, 4.32862667e-04, 1.00255660e-04,
        4.61214267e-06, 4.05951636e-03, 1.93214670e-04, 1.24109338e-05,
        6.98927150e-04, 1.48414183e-04, 1.37241834e-04, 6.95115654e-03,
        8.78219580e-05, 2.72962730e-03, 9.69585439e-04, 1.52658706e-03,
        6.85979496e-04, 5.24534407e-05, 4.50527438e-07, 5.34695528e-06,
        3.12649718e-07, 2.03195705e-05, 3.05093977e-07, 4.03026461e-05]],
      dtype=float32)
```

Figure 3.26 Probabilities (Cifar-100)

Because we trained the model with different datasets (Cifar-10, Cifar100, etc.), the size of the possibilities toward a passing image will also vary if we pass the image "zero" into the models, even though the preprocessing procedure is the same (compress the image into 32*32 pixels, converting the image into array aiming to conduct normalization, read the trained model, and load the weights, etc.). In Cifar-10 case, the image is categorized into the 1st label (Index 0) among 10 labels with a possibility of 9.9997699e-01 (approximately 99%). As we understand that the first label from Cifar-10 dataset pointing to the "airplane" class, the model returns a perfect hit. However, if we pass the image "zero" into the second model (trained by Cifar-100), "zero" is categorized into the 49th label (Index 48), which is the "road" class under the superclass "large man-made outdoor things," with a possibility of 9.09132063e-01 (approximately 9.1%). Obviously, this

31

distinguishes failed. Although we do not have a specific airplane class in Cifar-100 dataset as the labels, still, we have the superclass (vehicles1, vehicles2) and subclasses (lawnmower, rocket, streetcar, tank, tractor, etc.), which are somehow similar to the label "airplane" or "aircraft" in a common sense of adding fuel or movable. Nevertheless, the expecting result could not come true, due to the reason of in our training dataset (Cifar-100) for this classification model, we do not have the collections of the images those representing an object "airplane".

Following the comparison of two models (trained by Cifar-10 and trained by Cifar-100), we decided to apply the former as the primary model serves for processing images. Due to a similar consideration regarding deciding the word embedding model in the previous section, we need to reduce the calculation complexities and handle the whole experiment's time cost. In addition, another point we need to highlight here, in this procuring possibility phase, as the image "zero" is the first time for the Cifar-100 model to classify, which leads us to conclude that we can also pass the images that do not have an obvious attribute (objects belong to the labels of the training dataset) into the model to obtain the possibilities. For instance, we can pass a JAIST mascot into the model and return the possibilities of each class judged by the neural network (result as approximately 43% JAIST mascot belong to the "dog" category, which is somehow understandable). (Figure 3.27). The variables we obtain for each different image are constant if we maintain the same training dataset and perpetuate the parameters in the neural network. This output from the SoftMax function plays a significant role for us to procure the **ICV.**


```
array([[2.2643004e-02, 3.9538874e-05, 2.5017563e-01, 4.0050149e-02,
        8.8153757e-02, 4.2774644e-01, 8.6303707e-03, 8.7130284e-05,
        1.6147232e-01, 1.0017044e-03]], dtype=float32)
```

Figure 3.27 JAIST mascot (Model trained by Cifar-10)

3.3.3 Kawaii Dataset

Kawaii dataset is an original image dataset that includes a total of 30 images representing various content. The whole image dataset is divided into three different branches, "Belong," "Ambiguous," and "No Related." Each of the branches includes 10 different images. All the images are collected through google image and passing into the model, which is trained by the Cifar-10 dataset to obtain the **ICV**, respectively for a further investigation.

1. Belong (Figure 3.28): The first branch of Kawaii image dataset, this branch of images can be classified into the same class of Cifar-10 dataset without further discussion, or the images in this branch have a single main theme to represent, which is also fit the label of Cifar-10 dataset. For instance, we can classify the image "bird.jpg" into the class bird, which also appears as the label in Cifar-10 dataset. We can also classify the image dog01.jpg into the class dog for the same reason above.
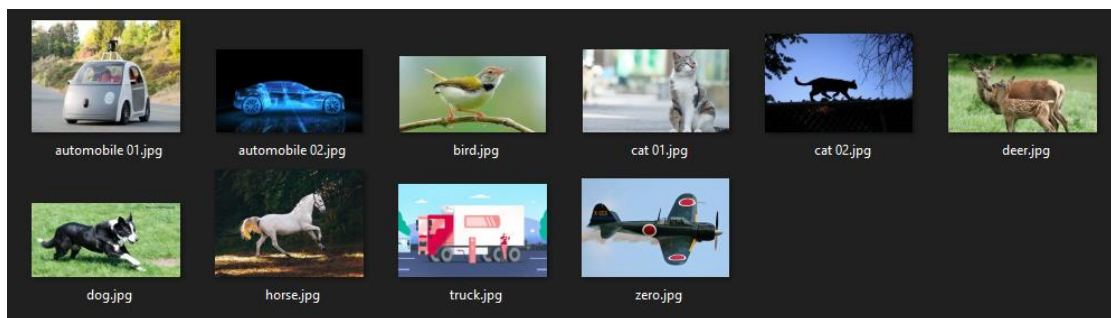
Figure 3.28 Belong (Kawaii Dataset)

2. Ambiguous (Figure 3.29): The second branch of the Kawaii image dataset, this branch of images can be classified into the same class of Cifar-10 dataset. Nevertheless, all the images have multiple main themes or have ambiguities. For instance, we have the hesitation to labeling the image "airplane alarm.jpg" into the class airplane because of "airplane alarm.jpg" representing an essential massage somehow relative to "COVID-19", as the same reason, we cannot easily classify the image "anime.jpg" into the class frog due to the contents appear in this image, which is a fiction animation character far away from a real frog.

33
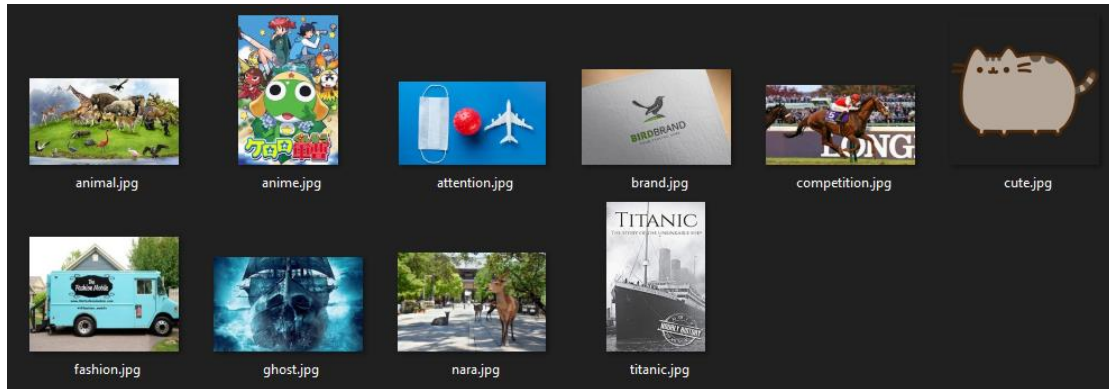
Figure 3.29 Ambiguous (Kawaii Dataset)

3. No Related (Figure 3.30): The third branch of the Kawaii image dataset, this branch of images cannot be classified into the same label as Cifar-10 dataset. All the images have different or irrelevant main themes or tend to another label or labels. A common observation is that we do not have the contents in our trained model for this image branch. For instance, we cannot classify the image "Crocodile eyes.jpg" into any class to fit our model, as well as we cannot classify the image "skyrim.jpg" for the same reason. All the images in this branch are stirring the model for further observation.
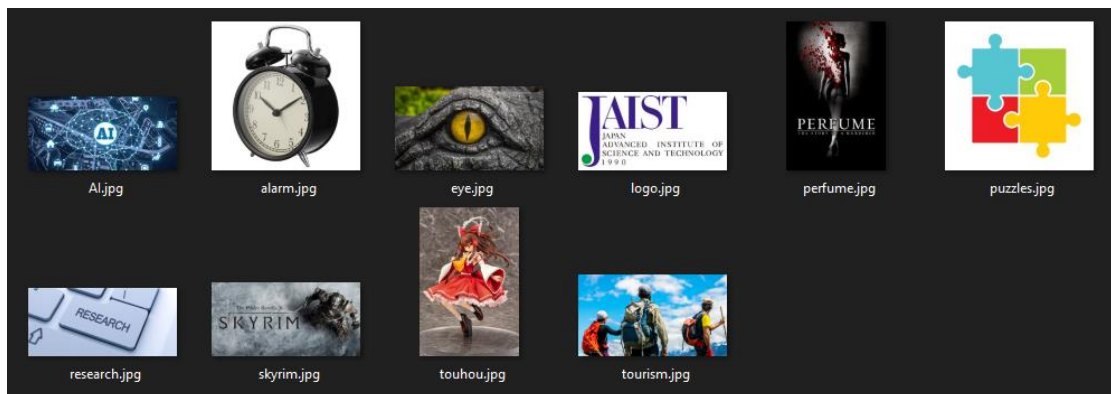
Figure 3.30 No Related (Kawaii Dataset)

3.3.4 Image Calculation

As the main purpose, we extract the possibilities toward each different image from a convolutional neural network. Subsequently, the second crucial component for experimenting, the **ICV** is designed as the following formula:

$$\sum_{i=1}^{n} Embed(label(i)) * pos(i)$$

The label stands for the whole dataset labels, which is applied to train the classification model. For instance, in Cifar-10, we have 10 labels (airplane, automobile, bird, etc.). The variable i will start from 0 and end by 9. In the case of Cifar-100, i start from 0 and continue until 99. All the labels are described in a natural language style (word). Therefore, it allows us to embed all the labels into a vector representation as we have done the same procedure for the user preferences. Pos here stands for the possibilities toward each label (class) the passing image being categorized. For a single image, if we train the model with n classes, the size of the possibilities will always remain the same with the total amount of labels, which is n. Later, we convert and save the variable "label" and variable "pos" into a list, respectively, multiple these two lists for each index, take the summation of n vector as the **ICV. ICV** is expected to integrate the meaning extracted from the image and a classification result from the neural network. For example, we pass the image touhou.jpg (Figure 3.31) from the No related branch (Kawaii dataset) into the model trained by the Cifar-10 image dataset (Figure 3.32).

Figure 3.31 touhou.jpg (Google)


```
[ 0.58490354 -0.20417431 -0.6825989   0.81585073  0.32450637  0.73656404
  0.08869052 -0.39216894  0.8084183  -0.33925828  0.01584052  0.43878478
  1.5443571   0.00893568 -0.06830634  0.22767363  0.28672674  1.0754249
 -1.1152244  -0.3919975  -1.1105156  -0.8702948   0.56588054 -0.02273211
  0.9517082  -0.43879354 -0.4337988   0.72223455 -0.47302982 -0.9510132
  0.7146859  -0.52895874  0.10339836  0.5362669  -0.31213877  0.0360037
 -0.22526056 -0.7837352   0.0708976  -0.8984749  -0.6722524  -0.13740921
 -0.8314806   0.33067182  1.2706206  -0.17437229  0.5944264  -1.4614776
  0.1936162  -0.3085715 ]
```

Figure 3.32 ICV for touhou.jpg

35

Chapter 4

Evaluation

4.1 Distance Calculation

4.1.1 Preliminary Experiment (Single User Estimation)

Once we obtain the **User Preference Vector (UPV)** for each respondent and **Image Classification Vector (ICV)** for every single image passing into the model, distance calculation allows us to evaluate whether the system output matches the user preference or not. In detail, we mainly apply a Euclidean distance to calculate the interval between **UPV** and **ICV**. In this section, it happens to be intricate when the **UPV** and **ICV** both variables. Therefore, we consider **UPV** as constant, which means an individual respondent being located, distance calculation will tell us the length of each image comparing with this identical respondent's preference. The lowest number has less distance and vice versa.

In the previous method section, we obtain the **UPV** pointing to the user "Wan," (User ID.01) which embedded from multiple preferences, "animal," "game," and "sport" with a determined weight 70, 20, 10, respectively, later, we are showing "dog.jpg" and "AI.jpg" to the user "Wan" (Figure 4.1), procuring an image priority represent as dog > AI, specifically, user "Wan" prefer "dog.jpg" more than "AI.jpg." Subsequently, we pass the image "dog.jpg" and "AI.jpg" into the model, compress them into 32*32 pixels and predict the labels being categorized (Figure 4.2), calculated **ICV** for both images based on predict results from the neural network (Figure 4.3 and Figure 4.4).

Figure 4.1 dog.jpg and ai.jpg
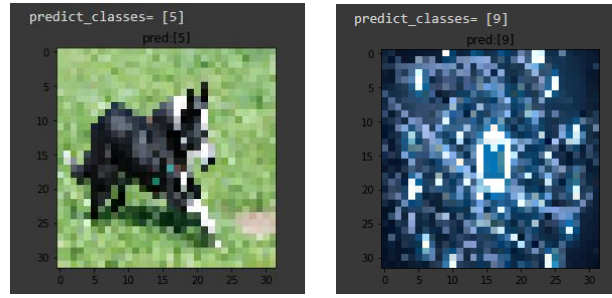
Figure 4.2 Predictions

```
[ 0.079452    -0.18018584 -0.56749153 -0.14972119  0.52327824  0.48888397
 -0.9144386   -0.35711497  1.0629215   -0.6874811  -0.00434914  0.3973705
  0.58873284   0.1434996    0.33573982 -0.11327112  0.3529217   0.97037464
 -1.3235319   -0.63535976 -0.47318769 -0.18532903  0.5084187   0.5450897
  0.5029429   -1.5349712   -0.996383    0.36899236  0.2576209  -0.5483999
  1.4407467    0.18684179 -0.43067747  0.70803463  0.05142963  0.2685303
  0.04932332  -0.22032823 -0.02764383 -0.66956335 -0.24391721  0.02221985
 -0.7180467    0.78386796  1.113106    -0.6324798   0.08800104 -1.0976168
  0.65203065   0.06439348]
```

Figure 4.3 ICV for dog.jpg

```
[ 0.40721685 -0.30968162  0.8061926   0.20937891  0.2895352   0.5639808
 -0.98706806 -0.06757575  0.8009534  -0.58428097  0.17275801 -0.4664548
 -0.1561554   0.2376391    0.2733396  -0.2641144  -0.27900422  1.4549868
 -0.6896564  -1.2935286    0.14471814 -0.22190216 -0.33108807  0.43710762
  0.21466614 -1.1447489   -0.38195974  0.8836399   0.9627742  -0.34597522
  1.5204972  -0.28706053  -0.29911384  0.7369812   0.80522346  0.21978685
  0.10214268 -0.4735214    0.22229815  0.32472828 -0.39949733 -0.06418443
 -0.04040132 -0.71077466   0.7472847  -0.26451674 -0.04642011 -0.65324557
  0.7269966  -0.4535722 ]
```

Figure 4.4 ICV for AI.jpg

Finally, we convert the vector representation of **ICV** into a type list as we have done the same procedure for **UPV** and implement Euclidean distance by Python to calculate the interval for both images. Explicitly, after the calculation, the distance between **UPV** (Wan) and **ICV** (dog) is approximate 3.23. On the other hand, the distance between **UPV** (Wan) and **ICV** (AI) is approximate 4.42. Due to the distance "dog.jpg" less than distance "AI.jpg." which matches priorities. In addition, the results of the dot product between **UPV** (Wan) and **ICV** (dog) is approximately 16, large than the dot product between **UPV** (Wan) and **ICV** (AI), which is approximately 9. Such a positive number tells us the two vectors are in the same direction, but **the ICV** (dog) angle is less than the angle of **ICV** (AI).

4.1.2 Preliminary Experiment (Multiuser Estimation)

Continue as the second part of the preliminary experiment, even though we have an intuition regarding the effectiveness of the proposed model, we also have a strong skeptical coexist because we only understand a few about the results and could not have an observation in the sense of the errors. Therefore, we invited the other two respondents to join the experiment and obtain two sets of preferences (words) and image priorities draw from the Kawaii dataset.

Table 1 ID.02/ID.03

User ID	Keywords	Image Priority	Observed Results
ID.02	Animation (40)	Competition (1)	Competition (4.0)
	Horse (30)	Research (2)	Research (4.8)
	Puzzle (30)	AI (3)	AI (5.1)
ID.03	Horse (40)	Nara (1)	Nara (2.8)
	Tourism (35)	Dog (2)	**Skyrim (2.9)**
	Animal (25)	**Skyrim (3)**	Dog (3.1)

In User ID.02 's case, our proposed model succeeds in estimating user preferences based on multiple giving keywords. Observed results match a fact of image priority. Nevertheless, in User ID.03's case, the model could not have a correct estimation. Specifically, "skyrim.jpg" has a lower preference level, but the result of distance calculation reflects as User ID.03 prefer "skyrim.jpg" than "dog.jpg." However, if we consider all three images, we also understand that the model output the correct predict as "Nara.jpg" close to User ID.03's preference the most, the results partially fit our expect, on the other words, an equivocal (Not a binary represents as true or false due to the correct and wrong estimation coexist) appears as distinguish outcome. Overall, the preliminary experiment results reveal an unknowing part of the model, bring us the inspiration, and confusion. Current circumstance delivers a message which is we need a further investigation, regarding the reducible and irreducible errors, keyword scales, image scales, optimization, and explainable AI, etc. All the clues lead us to determine a further investigation, which describes in the later section.

4.2 Inquire Form

To evaluate and observe the proposed model precisely, we designed an inquiry form for multiple respondents, under a baseline of limitation scale of keywords (represent as keyword list), and a limitation scale of images (represent as Kawaii dataset). The first part of the inquiring form can be referring as below (Figure 4.5):

*This Inquire Form will cost around 3mins.

Question 1: Keywords

Please choose your favorite words from the keyword list (The keyword list is selected based on our training and testing dataset). Also, please add a certain weight for each word being chosen. (Summation of weights are expecting to be 100).

*Example: Airplane (50), Dog (30), Puzzle (20).
*Weights: 50+30+20 =100

Airplane	Automobile	Bird	Cat	Deer
Dog	Frog	Horse	Ship	Truck
Attention	World	Anime	Brand	Competition
Cute	Fashion	Ghost	Nara	Titanic
AI	Alarm	Eye	Logo	Perfume
Puzzle	Research	Skyrim	Touhou	Tourism

Select your keywords:

Figure 4.5 Question1 - Keywords

Subsequently, the second part inquiries an image priority as we have done in the previous section (Figure 4.6). Only this time we set up the goal for 10 sets of the images (each includes 3 images) draw from Kawaii dataset (each different branch). Respondents are expected to fill the parenthesis with weights (summation being 100) for all 10 sets, which means for 1 single respondent, we can have 10 sets of sample data (Figure 4.7).

Question 2: Weight distribution for Images

Please fill the 10 set of brackets corresponding to your preferences toward images and distribute weights for each set of images:

*Example: If you prefer image "Controller" the most, follow by the image "Sport", and image "Phone" the less, you can distribute a totally 100 weights for 1 set (3 images).

(60)   (30)   (10)

Figure 4.6 Question2 - Image Priority (Google Image)
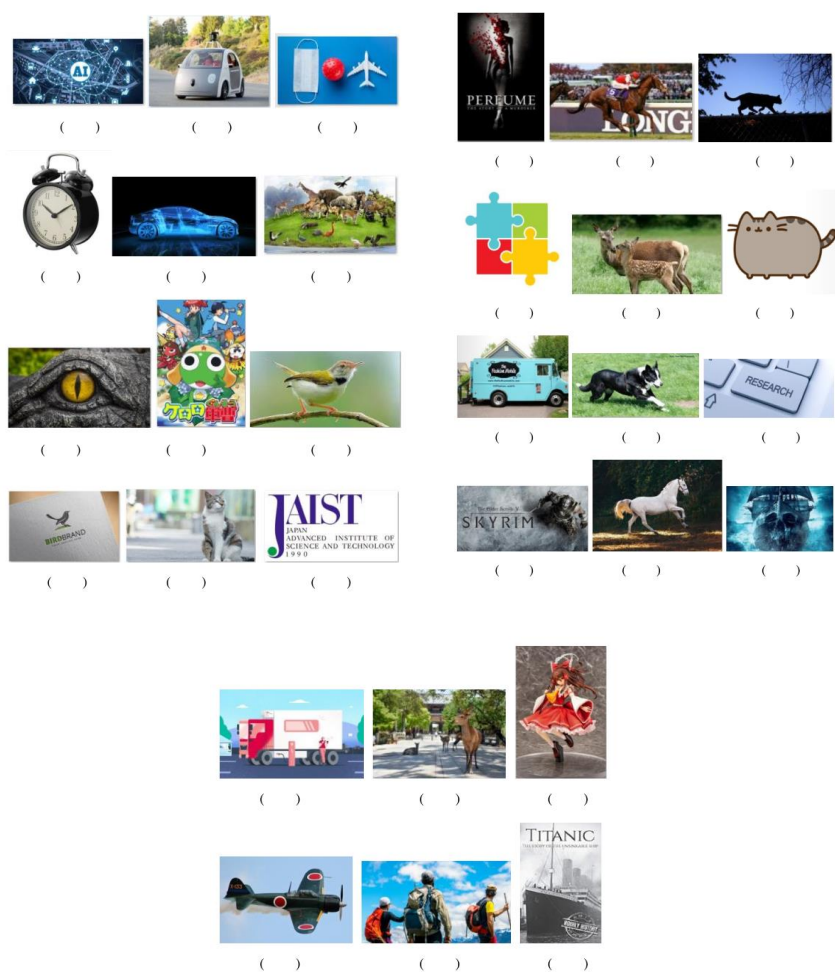
Figure 4.7 10 image sets

4.3 Data Analysis

4.3.1 Correlation

User ID.04 was the first respondent offering the information for this explicit experiment as follows:

1. We embedded the keywords into **UPV** (ID.04) and outputted the **ICV** (i1, i2, i3, … i30) from the neural network.
2. We calculated the distance for all the **ICV** (i1, i2, i3, …, i30) with **UPV** (ID.04).
3. We compared the results with the inquiry form to evaluate our model, as shown in Table 4.2.

In detail, the Image Priority column stands for assigned weight filling by our respondents. Summation of the weight is 100 and the respondents are freely distributing all the weight to 3 images in 1 set. Also, if replace the weight as ranking, we use number "1" represents the largest preference (assigned the largest weight), and number "3" represents the smallest (In case of "0", we apply an "-" mark as the user has no preference toward a certain image at all). On the other hand, the Observed Results column stands for whether the system output matches the inquiry form or not, "Incorrect" represents the image with the largest weight could not have the smallest distance through the system output and vice versa. "Correct" represent a perfect hit, and "Partial" represents the correct and incorrect output co-exist within 3 images. For instance, the image with the highest priority output as the second distance, and the image with the lowest priority have the largest distance.

Table 2 Set1-10 (ID.04)

User ID	Keywords	Image Priority	Observed Results
ID.04			
	AI (80)	80,20, 0 (Set01)	Partial (Set01)
	Automobile (12)	60,10,30 (Set02)	Correct (Set02)
	Tourism (8)	10,20,70 (Set03)	Partial (Set03)
		10,10,80 (Set04)	Incorrect (Set04)
		40,50,10 (Set05)	Partial (Set05)
		40,30,30 (Set06)	Incorrect (Set06)
		10,20,70 (Set07)	Partial (Set07)
		50,30,20 (Set08)	Partial (Set08)
		60,20,20 (Set09)	Incorrect (Set09)
		0, 100, 0 (Set10)	*Incorrect (Set10)

Also, we apply the correlation analysis for all image sets (01-10), between the weights assigned to the images (x array) and the distance as the system output from our proposed model (y array). Shown in Table 4.3. Nevertheless, P-value and R-value have a huge floating bandwidth from minimum to maximum. In our consideration, it is because of x and y contain too less indices (3 images within the sets) to make sense for observe the correlation. As a conclusion, the only image set statistic significant (P<0.05) for User ID.04 is the image set10, which is an incorrect predict from the model, with the assigned weight (0, 100, 0). We used an asterisk mark for highlighting this result shown in Table 4.2.

Table 3 R-value and P-value (ID.04)

Image Set (ID.04)	R-value	P-value
Set01	-0.04099720860271485	0.9738930496045457
Set02	-0.9770876913384728	0.13654074744394323
Set03	-0.6081635059495852	0.5838129930961687
Set04	0.9937530431743646	0.07119602015019756
Set05	0.1759663393798107	0.8873900223953001
Set06	0.981798914670341	0.12164792126501946
Set07	0.4944349144322373	0.6707500571395224
Set08	-0.7259119017761976	0.48283599711613706
Set09	0.9561111004818128	0.1893101049523075
Set10	0.9970337848200698	0.0490460296894045

As a further investigation for pursuing a reasonable explanation, we change the analysis method to obtain more indices. In detail, we combine all sets of the images. For instance, in User ID.04's case, we remain x array represents the assigned weight toward images and y array represents the distance which identifies the interval between **UPV** (ID.04) and **ICV** (1-30). Only this time, both x and y array include 30 indices, as shown in Figure 4.8.

```
30
[ 80  20   0  60  10  30  10  20  70  10  10  80  40  50  10  40  30  30
  10  20  70  50  30  20  60  20  20   0 100   0]


30
[5.60322009 6.12764811 5.43062503 4.43941726 5.34125806 5.15736563
 5.26664791 6.04820371 4.9465394  4.93789682 4.7469632  6.31481565
 5.63876035 5.18092941 5.23304281 5.29830155 5.15141574 5.18092941
 5.35788203 5.1396738  5.39952997 4.65729068 5.18822509 4.96825815
 6.317532   5.09440073 4.56853825 5.70244884 6.21429582 5.74612626]
```

Figure 4.8 Weight and Distance

Pearson's r correlation analysis between x and y shows a slight positive correlation (Approximately 0.24) between assigned weight and distance (Figure 4.9). Nevertheless, if we continue calculating the P-value for this correlation result, P-value $\approx$ 0.19, which is large than 0.05 (Figure 4.10), the statistical approach announces that the correlation between them is statistically insignificant (Considering a random occur). We also apply Spearman's rho (Figure 4.11) and Kendall's tau (Figure 4.12) to analyze the giving x and y. In detail, Spearman's rho output an approximately 0.073 as the correlation result between the giving x and y as well as a P-value approximately 0.70. On the other hand, Kendall's tau output is approximately 0.044 as the correlation between x and y, and a P-value is approximately 0.75. Both analysis methods announce that we should consider that the correlation between the assigned weight and output distance being randomly occurred.

```
[[1.          0.24386093]
 [0.24386093 1.         ]]
```

Figure 4.9 Correlation Matrix (NumPy)

```
(0.24386092970830198, 0.19406911944076907)
```

Figure 4.10 Pearson's r (SciPy)

```
SpearmanrResult(correlation=0.07303727821302367, pvalue=0.7013074005267406)
```

Figure 4.11 Spearman's rho (SciPy)

```
KendalltauResult(correlation=0.04369575640002634, pvalue=0.7451534903832047)
```

Figure 4.12 Kendall's tau (SciPy)

4.3.2 Regression

1. Linear Regression

Once we derive the correlations between assigned weight and distance, we can also make the efforts to draw a linear regression line. Specifically, we calculate the slope of the regression line (0.004486416004423194) by the mathematic formula represented as the change of y divide by the change of x, as well as the intercept of the regression line (5.197058859026602) by calculating the y-intercept when x = 0. Furthermore, we procure the standard error of the estimated gradient (0.0033718252079896443), for this linear regression line (Figure 4.13).
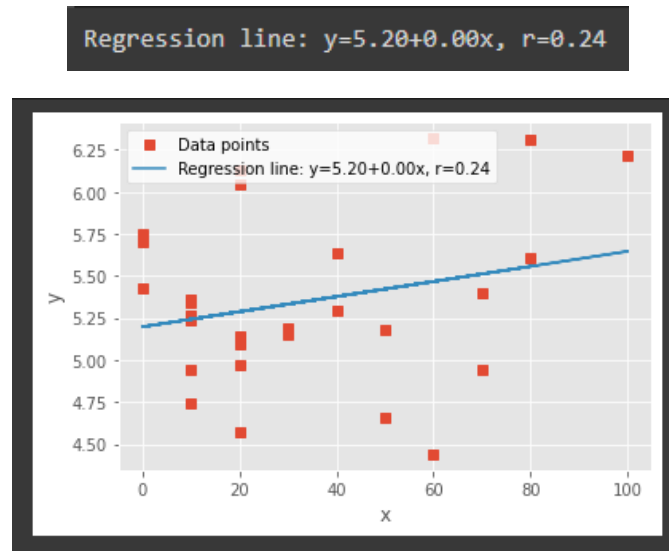
Figure 4.13 Linear Regression Line (Matplotlib)

2. Logistic Regression

Logistic regression also one of our analysis methods to capture the relationship between designed weight and output distance. In this case, because logistic regression is mainly applied for binary classification to interpret the results, we assign 0 and 1 to fill the y array instead of distance. In detail, we assign the number 0 represent as invalid output (the proposed model does not match the user preferences), and 1 represent as valid output (purposed model match the user prefer somehow). For instance, in User ID. 04's case, the "incorrect" image set being recorded as 0, and "partial," "correct" image set being recorded as 1. Therefore, y array can be described as y = [1,1,1,0,1,0,1,1,0,0]. On the

other hand, for conducting the logistic regression analysis, x array needs to be 2-dimension. We applied the reshape function to modify the x array with the argument (-1,3). Later, we fit the model and print the string representation of the model shown in Figure 4.14.

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=0, solver='liblinear', tol=0.0001, verbose=0,
                   warm_start=False)
```

Figure 4.14 Fit model (scikit-learn)

Subsequently, we continue the analysis by output the coefficients (each x variable's effect on the y variable) of the model (Figure 4.15) and evaluate the model's score. In User ID. 04's case, we have a score equals to 0.7 because we have 7 correct predictions. Nevertheless, if we output the probability array, the categorical results are all 1 for x except the image set10. These categorical results intrigue our suspicious. We output the probability matrix for further observation (Figure 4.16). Even though we have several categorical results for image sets that are close, the 1-9 image sets are categorized into class 1 due to the probability of class 0.

```
array([[ 0.02049054, -0.014735  ,  0.00421679]])
```

Figure 4.15 Coefficient (scikit-learn)

```
array([[0.20674759, 0.79325241],
       [0.22993367, 0.77006633],
       [0.44880985, 0.55119015],
       [0.40251461, 0.59748539],
       [0.46875594, 0.53124406],
       [0.37656002, 0.62343998],
       [0.44880985, 0.55119015],
       [0.33918852, 0.66081148],
       [0.26518952, 0.73481048],
       [0.81357362, 0.18642638]])
```

Figure 4.16 Probability matrix (scikit-learn)

Finally, we plot the confusion matrix by Python library Matplotlib, as shown in Figure 4.17. For User ID.04, predictions are distributed into 3 different slots, described as predicted 0, match the actual 0 (upper-left position, totally 1), and predicted 1 match the

actual 1 (lower-right position, totally 6). Also, we have serval observations classified incorrectly. Those are the predicted 1 mismatch the actual 0 (upper-right position, totally 3). In contrast to the original y array, which describe as [1,1,1,0,1,0,1,1,0,0]. We can also apply F-measure for a further calculation based on the plotted confusion matrix (Figure 4.18). On the other hand, if we change the original definition of 0 and 1 for y array as "partial" mark also equals to 0, then the y array becomes to [0,1,0,0,0,0,0,0,0,0] correspondingly, which output a prefect prediction (Figure 4.19).
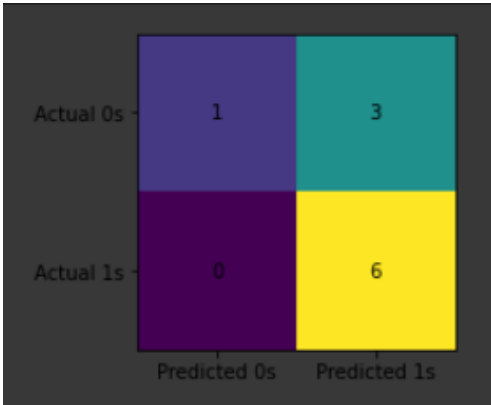
Figure 4.17 Confusion Matrix 01 (ID.04)

	precision	recall	f1-score	support
0	1.00	0.25	0.40	4
1	0.67	1.00	0.80	6
accuracy			0.70	10
macro avg	0.83	0.62	0.60	10
weighted avg	0.80	0.70	0.64	10

Figure 4.18 F-measure (ID.04)
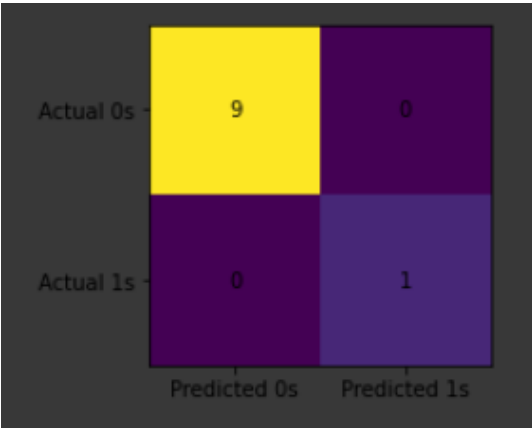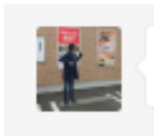
Figure 4.19 Confusion Matrix 02 (ID.04)

4.3.3 All Samples
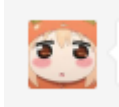
As the inquiry form in charge of setting up the baseline for all respondents, we continue to calculate the distance between **UPV** (Individual) and **ICV** (constant vector) as the final evaluation part for the proposed model. Eventually, we finished the calculation until the User ID.07. For details regarding each respondent's keywords along with image distance, we plot table 4.4, 4.5, and 4.6, as shown below:

Table 4 ID.05

Keywords (ID.05)	Image Set (1-10)		Distance
ID.05		SET 01	Partial
	(01)	ai.jpg	4.008618709749134
	(02)	automobile01.jpg	4.350099923338944
	(03)	attention.jpg	2.7176146523692024
		SET 02	Incorrect
Keywords:	(04)	alarm.jpg	3.527526491582579
Airplane (30)	(05)	automobile02.jpg	4.467894787980907
Research (30)	(06)	animal.jpg	4.815952433561925
Dog (10)			
Ship (20)		SET 03	Correct
Titanic (10)	(07)	eye	5.038894782454962
	(08)	anime	4.345007628555461
	(09)	bird	3.2824820540615383
		SET 04	Partial
	(10)	brand	3.6996785428404753
	(11)	cat01	3.375649016943949
	(12)	logo	4.612264204391921
		SET 05	Incorrect
	(13)	perfume	4.03720754580832
	(14)	competition	3.802883422190393
	(15)	cat02	3.489269255047014

		SET 06	Incorrect
	(16)	puzzle	3.525018066837102
	(17)	deer	4.794432297450001
	(18)	cute	3.802883422190393
		SET 07	Partial
	(19)	fashion	4.5068978555726815
	(20)	dog	4.268444289825992
	(21)	research	2.4595496473077825
		SET 08	Incorrect
	(22)	skyrim	2.544605175703283
	(23)	horse	4.174132718773923
	(24)	ghost	2.859989940282542
		SET 09	Correct
	(25)	truck	4.615824115317389
	(26)	nara	2.349860118686501
	(27)	touhou	3.002424906672341
		SET 10	Incorrect
	(28)	zero	2.8377574506216074
	(29)	tourism	4.452907444018788
	(30)	titanic	3.006035606507105

Table 5 ID.06

Keywords (ID.06)	Image Set (1-10)		Distance
ID.06		SET 01	Partial
	(31)	ai.jpg	5.51668076180391
	(32)	automobile01.jpg	5.911023838179149
	(33)	attention.jpg	4.932169555870007
		SET 02	Partial
	(34)	alarm.jpg	2.349789253187381
	(35)	automobile02.jpg	5.553668844224715

	(36)	animal.jpg	2.5968285191292133
Keywords:			
Anime (30)		SET 03	Partial
Frog (30)	(37)	eye	2.698713898633426
Cat (20)	(38)	anime	5.858402710730769
Research (10)	(39)	bird	3.112259972234278
Tourism (10)			
		SET 04	Incorrect
	(40)	brand	2.882591043119047
	(41)	cat01	2.8405013319558265
	(42)	logo	6.111210626439508
		SET 05	Incorrect
	(43)	perfume	5.533370226330422
	(44)	competition	4.211938706500924
	(45)	cat02	5.137152805970019
		SET 06	Partial
	(46)	puzzle	5.165455674710944
	(47)	deer	2.5467071137797097
	(48)	cute	4.211938706500924
		SET 07	Partial
	(49)	fashion	5.577682432438246
	(50)	dog	3.0623810281663055
	(51)	research	4.837226405040576
		SET 08	Incorrect
	(52)	skyrim	3.104267548001169
	(53)	horse	3.254802138573072
	(54)	ghost	4.31133047409912
		SET 09	Partial
	(55)	truck	6.113995582565312
	(56)	nara	3.831717067825309
	(57)	touhou	3.181918910227415

		SET 10	Incorrect
	(58)	zero	5.159714908755754
	(59)	tourism	5.994533135895172
	(60)	titanic	5.213776863598985

Table 6 ID.07

Keywords (ID.07)	Image Set (1-10)		Distance
ID.07		SET 01	Partial
	(61)	ai.jpg	3.9920485552116656
	(62)	automobile01.jpg	4.111163332312878
	(63)	attention.jpg	2.528598286068179
Keywords:		SET 02	Incorrect
Ship (50)	(64)	alarm.jpg	2.9971208678682713
Dog (30)	(65)	automobile02.jpg	4.891886918067863
Tourism (20)	(66)	animal.jpg	4.454551161616393
		SET 03	Correct
	(67)	eye	4.6960845969615
	(68)	anime	4.160891939291193
	(69)	bird	3.0481557233894674
		SET 04	Correct
	(70)	brand	3.192344873960161
	(71)	cat01	2.682761720644159
	(72)	logo	4.372922751484492
		SET 05	Partial
	(73)	perfume	3.9957616831423954
	(74)	competition	3.1704017471781807
	(75)	cat02	3.7336479819431325
		SET 06	Partial
	(76)	puzzle	3.6014797678280694

(77)	deer	4.396125131626435
(78)	cute	3.1704017471781807
	SET 07	Partial
(79)	fashion	4.927216046538552
(80)	dog	3.6718028643749707
(81)	research	2.061359141795893
	SET 08	Partial
(82)	skyrim	1.6625201452370233
(83)	horse	3.6429185581616186
(84)	ghost	2.4965688231467276
	SET 09	Correct
(85)	truck	4.375790560751759
(86)	nara	1.1885626827931701
(87)	touhou	2.552548610331052
	SET 10	Partial
(88)	zero	3.4735286295075443
(89)	tourism	4.240438334755673
(90)	titanic	2.3769581036427443

At last, because of we procured a total of 40 samples of image sets (each set include 3 images, totally 120 assigned weight and distance), collected from the inquire form responded by User ID.04 to User ID.07 (Figure 4.20), we apply the correlation analysis and logistic regression for all sample data. The correlation between the assigned weight (x) and output distance (y) is 0.09803027927664261 (slightly positive correlation). Nevertheless, P-value is for the giving x and y is 0.2867873062094666 (large than 0.05, statistically insignificant). On the other hand, in case of logistic regression, after we pass all distance those attached with a "partial" label belong to the the class "1" for y array, final score (accuracy) as the outcome is equal to 0.75 (75% accuracy). In addition, the confusion matrix shows in Figure 4.21 and F-measure shows in Figure 4.22.

```
120
[ 80  20   0  60  10  30  10  20  70  10  10  80  40  50  10  40  30  30
  10  20  70  50  30  20  60  20  20   0 100   0  50  40  10  10  70  20
   0  40  60  30  60  10  35  20  45  10  75  15  35  60   5  30  50  20
  30  60  10  10  50  40  20  50  30  10  60  30  20  30  50  70  10  20
  15  80   5  30  60  10  35   5  60  10  90   0   5  80  15   8  90   2
  50  20  30  10  20  70   0  30  70  10  40  50  70  10  20  10  20  70
  30  30  40  20  40  40  30  40  30  20  50  30]


120
[5.60322009 6.12764811 5.43062503 4.43941726 5.34125806 5.15736563
 5.26664791 6.04820371 4.9465394  4.93789682 4.7469632  6.31481565
 5.63876035 5.18092941 5.23304281 5.29830155 5.15141574 5.18092941
 5.35788203 5.1396738  5.39952997 4.65729068 5.18822509 4.96825815
 6.317532   5.09440073 4.56853825 5.70244884 6.21429582 5.74612626
 3.99204856 4.11116333 2.52859829 2.99712087 4.89188692 4.45455116
 4.6960846  4.16089194 3.04815572 3.19234487 2.68276172 4.37292275
 3.99576168 3.17040175 3.73364798 3.60147977 4.39612513 3.17040175
 4.92721605 3.67180286 2.06135914 1.66252015 3.64291856 2.49656882
 4.37579056 1.18856268 2.55254861 3.47352863 4.24043833 2.3769581
 4.00861871 4.35009992 2.71761465 3.52752649 4.46789479 4.81595243
 5.03889478 4.34500763 3.28248205 3.69967854 3.37564902 4.6122642
 4.03720755 3.80288342 3.48926926 3.52501807 4.7944323  3.80288342
 4.50689786 4.26844429 2.45954965 2.54460518 4.17413272 2.85998994
 4.61582412 2.34986012 3.00242491 2.83775745 4.45290744 3.00603561
 5.51668076 5.91102384 4.93216956 2.34978925 5.55366884 2.59682852
 2.6987139  5.85840271 3.11225997 2.88259104 2.84050133 6.11121063
 5.53337023 4.21193871 5.13715281 5.16545567 2.54670711 4.21193871
 5.57768243 3.06238103 4.83722641 3.10426755 3.25480214 4.31133047
 6.11399558 3.83171707 3.18191891 5.15971491 5.99453314 5.21377686]


[[1.         0.09803028]
 [0.09803028 1.         ]]
```

Figure 4.20 40 Samples
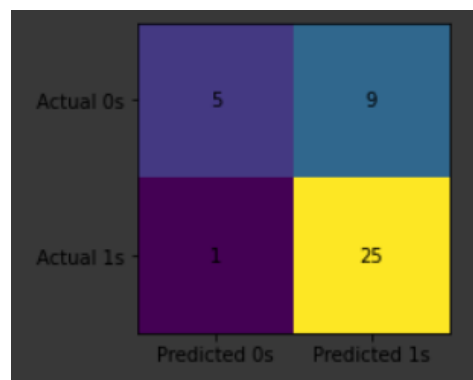
Figure 4.21 Confusion Matrix 03 (ALL)

	precision	recall	f1-score	support
0	0.83	0.36	0.50	14
1	0.74	0.96	0.83	26
accuracy			0.75	40
macro avg	0.78	0.66	0.67	40
weighted avg	0.77	0.75	0.72	40

Figure 4.22 F-Measure (ALL)

Chapter 5

Summary

5.1 Conclusion

In retrospect of this research, we obtain the UPV and ICV through a series of procedures. Later, we calculate the distance between them, obtained a digitalized representation of user preference as the system output. Finally, we compare the calculation results with the user priorities, justify our purposed model through the user-oriented interview as well as a scientific approach. In conclusion, our proposed model could not precisely estimate user preference within our evaluation section (User ID.01 ~ ID.07). In detail, we have 8 correct hits in a total of 43 samples (approximately 18.6% integrated accuracy), and 21 "partial" estimation that identifies our proposed model effected somehow (floating from 18.6% ~ 67.4% within the image sets). On the other hand, we also encountered 14 "Incorrect" image preference estimation (around 32.6% of the samples could not match our expectation) recorded as the experiment result. Nevertheless, we strongly believe that our proposed model could be a significant idea and choose not to reject our hypothesis arbitrarily after we checked all the inquiry form and experiment results carefully.

5.2 Discussion

There are serval reasons lead us to this conclusion. Among them, most cannot afford to ignore, the selections toward image set made by our respondents. For instance, In User ID. 05's case, we notice that although User ID.05 selected "Airplane," "Ship," "Titanic," etc., as the keywords. However, User ID.05 did not choose the images with the above contents (features, characteristics learn by the neural network) that related to "airplane" or "ship," etc., as the highest priority within image sets. In other words, the proposed model embedding the User ID. 05's keywords such as "airplane," "ship" into vector and output a small distance with the images somehow contain the correct objects, but our respondents easily select other images as their priority which does not match the submitted keywords (It happens that keywords and image priority irrelevant or even

53

reverse direction). This phenomenon arises accompany with the inevitable problems such as we could not evaluate the purposed model accurately and obscurities bringing into the data analysis. Regarding this problem, we consider that because the keywords can only represent a narrow aspect of the user preferences, the other elements such as the user's current mood, color reflections, image style, along with latent messages contained also influence user priority. For instance, image "zero" shapes into airplane but can also associate with intangible attributes such as the Empire of Japan, WorldWar2, etc. Those attributes take a negative impact on the user's consideration. Therefore, even though our respondents preferring the keyword "airplane", but image "zero" cannot match the highest user priority within the image sets. In addition, image "attention" from the ambiguous branch of the Kawaii dataset not only represents the idea of the airplane but also carries the message toward COVID-19, which might give the respondents a negative impact resulting in the user's image priorities unparalleled with the keywords. Therefore, the keywords table with a limit scale is not sufficient to fulfill the investigation together with supporting experiment results. Because the objects appear within the image only a finite section when our idea devoted to digitalizing user preferences.

5.3 Future Works

Oppositely, regardless of the altercation of user performance, we also understand several weaknesses of our proposed procedures:

1. We should link the Kawaii dataset closer to the training dataset instead of 2/3 images from the Kawaii dataset are not affected, in the sense of images can be learned and distinguished by the neural network.
2. We should make a statement within our policy section regarding the inquiry form, which notes that keywords and image priorities should be related somehow in common sense.
3. The scientific approach, such as correlation and linear/logistic regression, could not make much sense for our experiment results and bring us difficulties in understanding or making a reasonable explanation.

To cope with this series of problems, we should also consider the other evaluation methods for our proposed model. Overall, we could not prove that our proposed model

works effectively for image estimation through distance calculation. However, our hypothesis should not be rejected in a hurry. A more well-constructed procedure (high accuracy neural network, more appropriately image dataset, numbers of respondents, a standardized and automagical flow, etc.) are needed.

5.4 Contributions

This research investigated a digitalized representation between the user prefer (**UPV**) and a particular image (**ICV**). As an attempt to improve the traditional image recommendation system, those lack of attention with the combination of artificial intelligence, and less appropriately solutions for the image classification/recommendation problems under a circumstance of individual being considered. As our contributions to this experiment, we believe that our research flow regarding the creation of **UPV** and **ICV** gives inspiration to others whose works in this field, as well as we ignite the distance calculation approach together with machine learning algorithms and natural language representation. Specifically, although we only apply word embedding (GloVe) for the vectorization of keywords in this experiment due to a limited research period, we place the clue to the later such as they can also make efforts to introduce the document to vector techniques to embed the whole sentences or paragraphs as the representation of individual, etc. The experience and source code for website construction, database, data cleaning, neural network, data analysis, etc. Such procedures in this experiment can be independent but also associate with other techniques under other ideas aiming to solve the problems not mentioned in this research. Overall, as a holistic view for our proposed steps, each with a rationale for its inclusion. The exploration and confirmation (feasibility evaluation) in this research strengthen our understanding.

Acknowledgement

I would like to express my appreciation to my supervisor, Professor Hasegawa Shinobu for picking me up when I struggle, his patient guidance helps me go through the master's course, without his help, I have no confidence to finish the study.

I also take this opportunity to records my gratitude for the other professors who deliver the knowledge to me, especially Professor Shirai for Natural Language Processing, Professor Beuran for Operating System, and Professor Ikeda for Data Structure and Algorithm, etc. All the courses I attend gives me the idea regarding science and technology, avail me entry the avenue of engineering. I believe that a real academy never confuses the student, instead of that, it is an inspiration, a heartfelt admiration for the wise of the human being.

Lastly, I would like to thank my dear friends Sai, Kwon Kham, Khun, Aung Thura Phyo, together with Lee - Kun, etc. All the encouragement and assistance I received cheer me up not only in my study but also mentally when I was weak and fragile.

<div align="right">

Wan Hua

2021.03

</div>

Bibliography

[1] Hanh T.H. Nguyen (B), Martin Wistuba, Josif Grabocka, Lucas Rego Drumond, and Lars Schmidt-Thieme, Personalized Recommendation of Photography Based on Deep Learning, Information Sciences - Volume 520, May 2020, Pages 416-430

[2] A.V. Savchenko, K.V. Demochkin, I.S. Grechikhin, User Preference Prediction in Visual Data on Mobile Devices, Computer Vision and Pattern Recognition 68T10 arXiv:1907.04519 [cs.CV] Jul 2019

[3] JorgeDíez, PabloPérez-Núñez, OscarLuaces, BeatrizRemeseiro, AntonioBahamonde, Towards explainable personalized recommendations by learning from users' photos, Information Sciences Volume 520, May 2020, Pages 416-430

[4] Tianjun Xiao, Yichong Xu, Kuiyuan Yang, Jiaxing Zhang, Yuxin Peng, Zheng Zhang, CNN-RNN: A Unified Framework for Multi-Label Image Classification, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 842-850

[5] Jiang Wang, Yi Yang, Junhua Mao, Zhiheng Huang, Chang Huang, Wei Xu, The Application of Two-Level Attention Models in Deep Convolutional Neural Network for Fine-Grained Image Classification, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 2285-2294

[6] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, Stefan Carlsson, CNN Features Off-the-Shelf: An Astounding Baseline for Recognition, The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, 2014, pp. 806-813

[7] Andreas C. Mueller, Sarah Guido, Introduction to Machine Learning with Python: A Guide for Data Scientists O'Reilly Media, 2016/10/21 ISBN-13: 978-1449369415

[8] D. Wang, K. Mao and G. Ng, "Convolutional neural networks and multimodal fusion for text aided image classification," 2017 20th International Conference on Information Fusion (Fusion), Xi'an, 2017, pp. 1-7, doi: 10.23919/ICIF.2017.8009768.

[9] Yazhou Yao, Wankou Yang, Pu Huang, Qiong Wang, Yunfei Cai, Zhenmin Tang, Exploiting textual and visual features for image categorization, Pattern Recognition Letters, Volume 117, 2019, Pages 140-145, ISSN 0167-8655.

[10] X. He and Y. Peng, "Fine-Grained Image Classification via Combining Vision and Language," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 7332-7340, doi: 10.1109/CVPR.2017.775.

[11] Fangyi Zhu, Zhanyu Ma, Xiaoxu Li, Guang Chen, Jen-Tzung Chien, Jing-Hao Xue, Jun Guo, Image-text dual neural network with decision strategy for small-sample image classification, Neurocomputing, Volume 328, 2019, Pages 182-188, ISSN 0925-2312.

[12] D. Wang and K. Mao, "Learning Semantic Text Features for Web Text-Aided Image Classification," in IEEE Transactions on Multimedia, vol. 21, no. 12, pp. 2985-2996, Dec. 2019, doi: 10.1109/TMM.2019.2920620.

[13] Dongzhe Wang, Kezhi Mao, Task-generic semantic convolutional neural network for web text-aided image classification, Neurocomputing, Volume 329, 2019, Pages 103-115, ISSN 0925-2312.

[14] T. Chen, Y. Chen, H. Guo and J. Luo, "You Type a Few Words and We Do the Rest: Image Recommendation for Social Multimedia Posts," 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018, pp. 2124-2133, doi: 10.1109/BigData.2018.8622513.

[15] Géron A. (2017). Hands on Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O'Reilly Media.

[16] Domingos P (2012) A few useful things to know about machine learning. Commun ACM 55(10).

[17] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pretraining of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[18] Bermingham, Adam and Smeaton, Alan F. (2010) Classifying sentiment in microblogs: is brevity an advantage? In: CIKM 2010 - 19th International Conference on Information and Knowledge Management, 26-30 October 2010, Toronto, Canada. ISBN 978-1-4503-0099-5. 10.1145/1871437.1871741.

[19] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In EMNLP, 2014.

[20] Thushan Ganegedara, Intuitive Guide to Understanding GloVe Embeddings, Towards Data Science. 2019

[21] A. Grover and J. Leskovec. node2vec: Scalable feature learning for networks. In KDD, 2016.

[22] J. w. Tan, S. Chang, S. Abdul-Kareem, H. J. Yap and K. Yong, "Deep Learning for Plant Species Classification Using Leaf Vein Morphometric," in IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 17, no. 1, pp. 82-90, 1 Jan.-Feb. 2020, doi: 10.1109/TCBB.2018.2848653.

[23] Y. Li, Y. Zhang, P. Li and X. Hu, "Unsupervised Cross-Lingual Word Embeddings Learning with Adversarial Training," 2019 IEEE International Conference on Big Knowledge (ICBK), Beijing, China, 2019, pp. 160-166, doi: 10.1109/ICBK.2019.00029.

[24] A. Sugiura and Y. Yukizaki, "Investigation of calculation/distance measurement method using spread-spectrum communications system," in IEEE Transactions on Intelligent Transportation Systems, vol. 3, no. 2, pp. 130-135, June 2002, doi: 10.1109/TITS.2002.801423.

[25] Pier Paolo Ippolito, Roadmap to Natural Language Processing (NLP), University of Southampton, 2020.

[26] Bird, Steven. (2006). NLTK: The natural language toolkit. 10.3115/1225403.1225421.

[27] Miguel Grinberg. 2018. Flask Web Development: Developing Web Applications with Python. " O'Reilly Media, Inc.".

[28] Dang, Quyen & Jeong, Sungmoon & Chong, Nak. (2017). Personalized robot emotion representation through retrieval of memories. 65-70. 10.1109/ICCAR.2017.7942662.

[29] J. Liang, L. Ma, X. Xiong, D. Shao, Y. Xiang and X. Wang, "Mutual Information-Based Word Embedding for Unsupervised Cross-Domain Sentiment Classification," 2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA), Chengdu, China, 2019, pp. 625-628, doi: 10.1109/ICCCBDA.2019.8725662.

[30] Krizhevsky, Alex & Sutskever, Ilya & Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

[31] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

[32] Cohen, Jonathan & Servan-Schreiber, David & Mcclelland, James. (1992). A Parallel Distributed Processing Approach to Automaticity. The American journal of psychology. 105. 239-69. 10.2307/1423029.

[33] M. Li, H. Wang and J. Li, "Mining conditional functional dependency rules on big data," in Big Data Mining and Analytics, vol. 3, no. 1, pp. 68-84, March 2020, doi: 10.26599/BDMA.2019.9020019.

[34] https://realpython.com/

[35] https://keras.io/

[36] https://flask.palletsprojects.com/en/1.1.x/