RE-SEARCH

ECE324 Final Report

Annie Wang & Yanisa Khambanonda

November 29th, 2019

Word Count: 2000

Penalty: NULL

Introduction

Online shopping has boomed in the last decade because of its convenience and flexibility, but we see an opportunity to further improve the shopping experience for owners and customers alike by implementing a machine learned classifier that automatically labels inventory images. For store managers, we can replace the tedious process of manually labelling thousands of images with a neural net classifier that can achieve a similar accuracy while producing more detailed and standardized labels. For the consumers, we anticipate that these standardized labels will yield more effective search engines to improve their shopping experience.

Classification of clothing features is a great way to apply machine learning because features like collars and buttons are translationally invariant in images. Furthermore, traditional methods of comparing with templates to classify is difficult because catalog images from different stores are stylistically different. Fortunately, the kernels of a CNN can learn these features and generalize to classify images it hasn't seen before. Furthermore, the abundant number of online clothing stores provide a large source of images to obtain data samples. For the scope of this project, we focused on classifying tops for 4 types of clothing features: color, neckline, sleeve length and the presence of buttons.

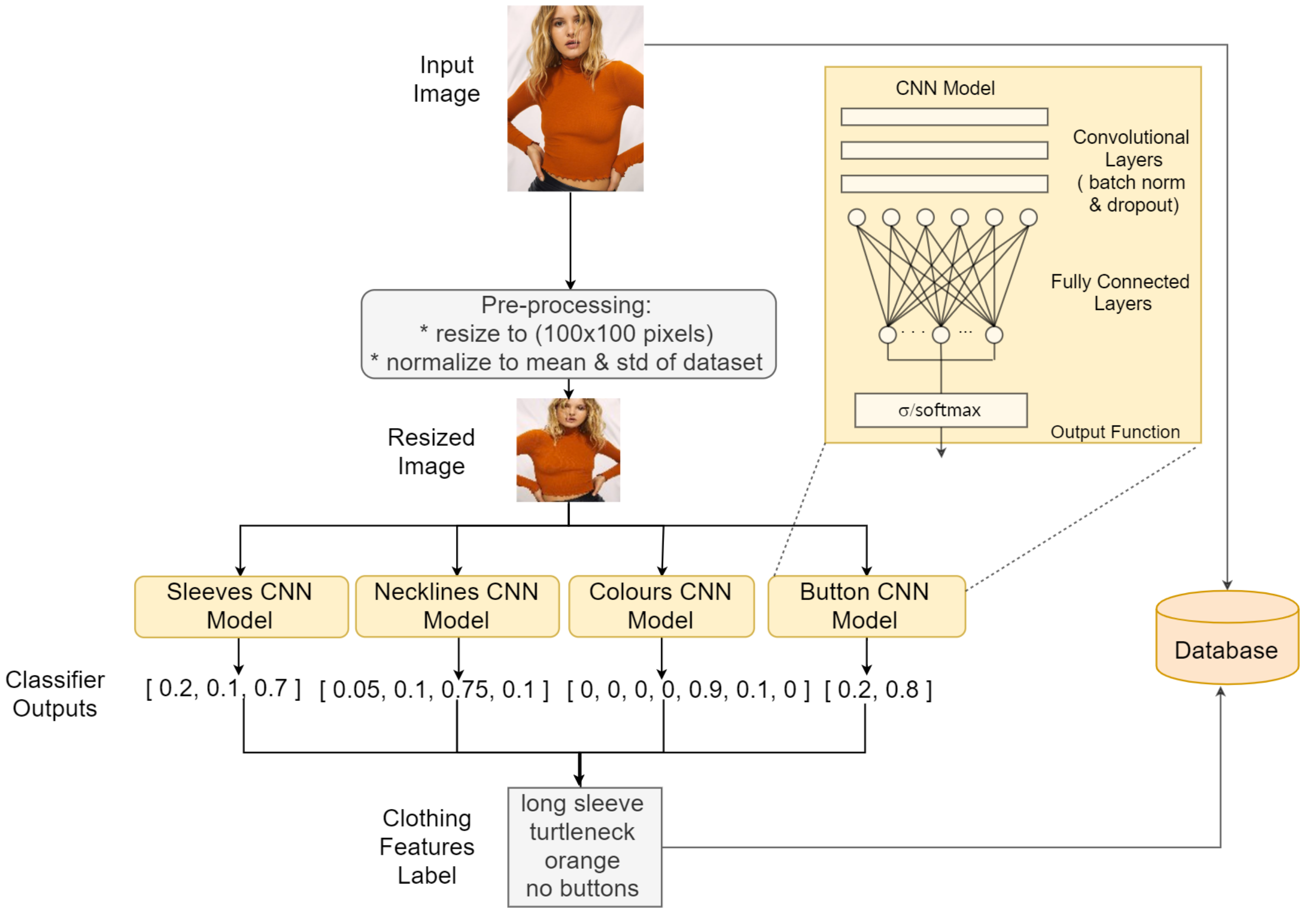
Background

Upon researching similar applications of machine learning, we concluded that a CNN model would be a good starting point. Based on the documented literature and open source projects we found online, clothing image classification, and specifically clothing feature classification appears to be a somewhat unique yet still possible machine learning application. For reference, James Le created a model that identifies articles of clothing using convolutional neural networks[1]. While this problem appears simpler than our goal, it could in the future be used in conjunction with our tops-classifier, as an initial process to categorize clothing items before trying to apply feature classification.

A generalized approach applied differently is Davis King's facial feature detection model used in python's *face_recognition* library[2]. This CNN model takes in images of people and extracts information about their facial features as an intermediate step to a facial recognition project. This model is similar to ours because they are not just trying to detect faces, but to classify their features, like the eyes or nose. While neither of these works exactly relate to our intended application, they are relevant motivations and background that convince us that the task can be achieved through machine learning.

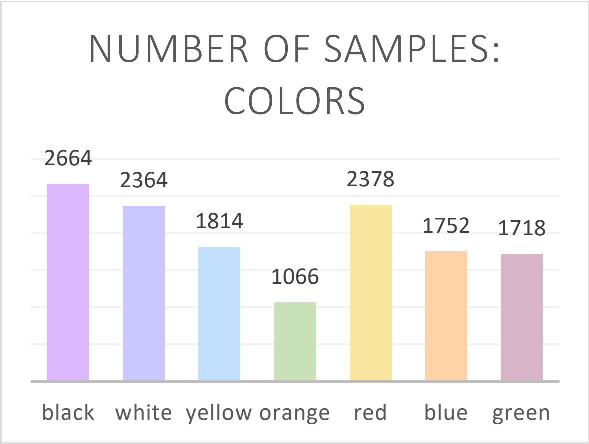
Illustration

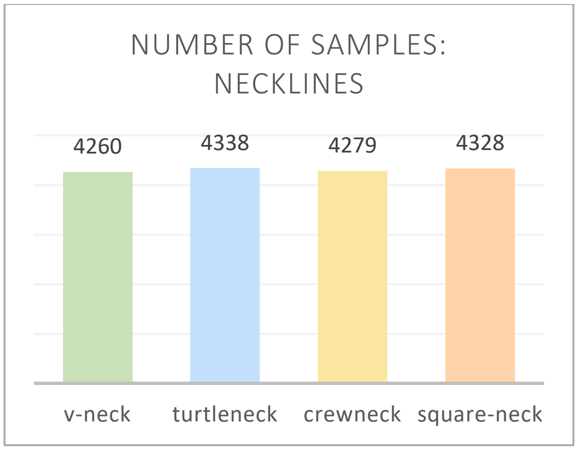
The following is a summative image of the basic design of our project.

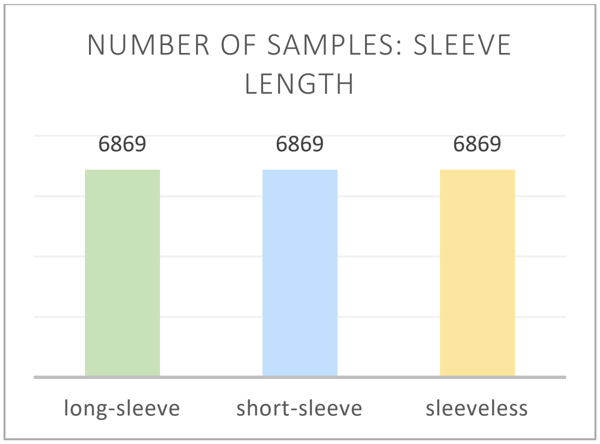


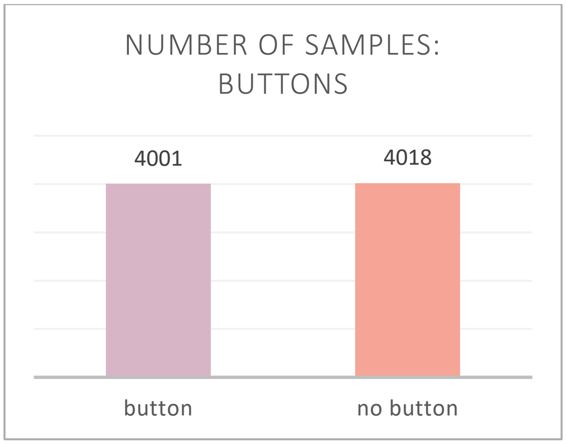
Data Collection and Processing

The team underwent three stages to build our dataset. In stage one, we scraped images of clothing tops from 20 online clothing stores using the searched terms as the image label. For example, image results from searching "orange tops" would be labeled "orange". Through this method, we collected over 10,000 images for each of the 4 clothing features: colors, necklines, sleeve lengths and buttons, and tried to maintain a balanced distribution of data samples in each class. Some data distribution graphs are shown below.



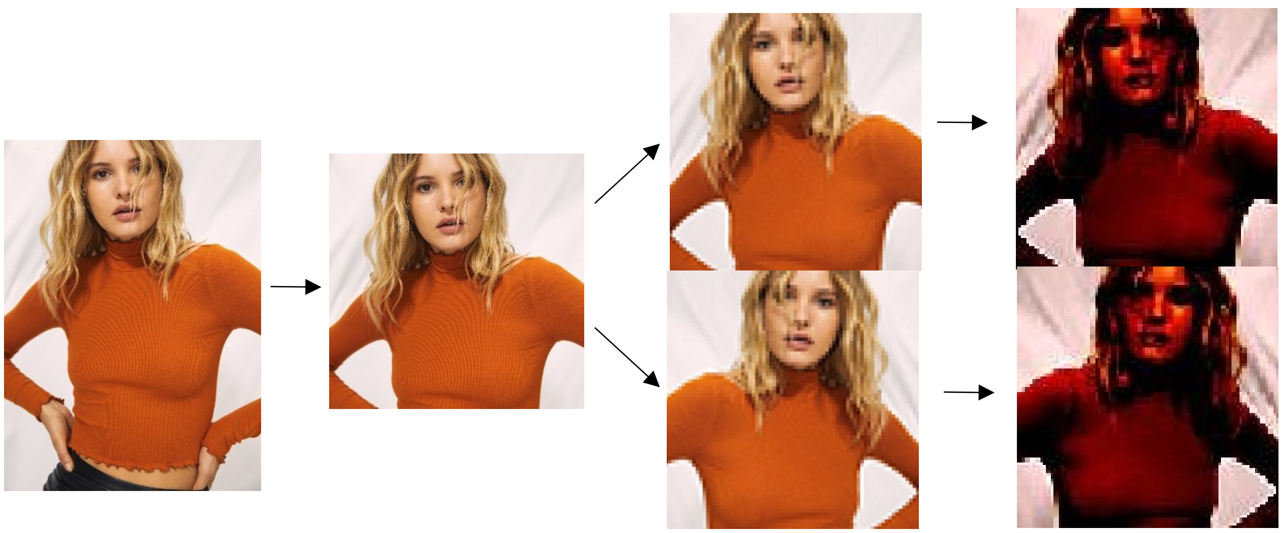






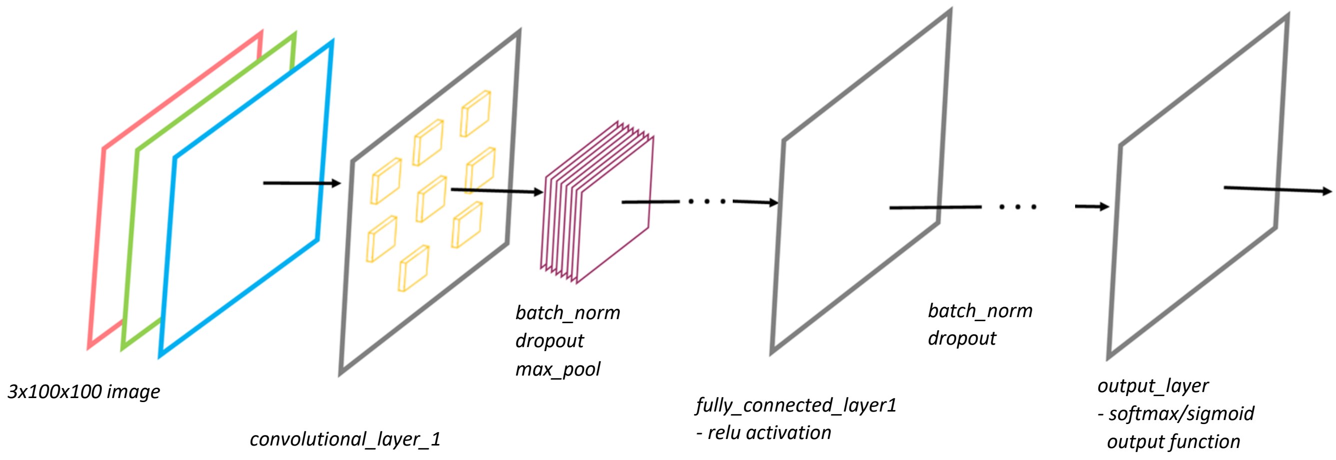
In stage two, we cleaned the data with consideration to some constraints and qualifications. These included images having correct labels, having the top fully visible, and having a discernable contrast between the top and the background. Our constraints also led to removal of full-body images, images of clothing without models, and non-top articles of clothing. Some example exclusions are shown below.

In stage three, data preprocessing, we first resized all the cleaned images to 100x100 pixels to standardize the samples and to reduce training time. Furthermore, we horizontally mirrored each image for data augmentation. Next, we normalized the entire dataset before dividing them into train, validation and test sets. Finally, we one-hot encoded each image label. This process and a data example are shown below.



Turtleneck → Tensor([0, 0, 1, 0, 0])

Architecture



To build each model, we began with a basic CNN and optimized hyperparameters differently to classify for each clothing feature. Each CNN takes in images of 100x100 pixels with RGB channels which pass through multiple convolutional layers as well as batch normalization, dropout (sometimes) and pooling layers. Afterwards, the inputs are passed through 2 fully connected layers with batch normalization and sometimes dropout layers followed by a softmax or sigmoid output function. The key optimized hyperparameters are shown below for each clothing feature CNN.

	# Convolutional Layers	# Model Parameters	Learning Rate	Loss Function	Batch Size	Dropout Rate
Buttons	2	212,721	0.0022	MSE	32	0.3
Colors	2	156,475	0.005	MSE	32	NA
Necklines	3	11,072	0.002	Cross Entropy	16	0.17
Sleeves	4	3,862	0.003	Cross Entropy	32	NA

Baseline

In our colors baseline model, the input RGB image is flattened by averaging to a 2D image then converted into a tensor array. The tensor is then passed through a fully connected layer with 40*40 hidden units, then a ReLU activation function. Following, the result is passed through a second fully connected layer with an output dimension of 7, equal to the number of color classes. Finally, we applied a softmax output function.

We adapted this model to generate baseline models for sleeves, necklines and buttons. We changed the output dimension to equal the clothing feature's number of classes and used a sigmoid output function instead of softmax for classifying buttons.

Qualitative Results

After examining individual input images and their corresponding predicted labels, we found the following common cases.

Input Image_1:

Predicted Label_1: [white, v-neck, short sleeves, buttons]

The image was classified completely correctly. We found that our classifier performed very well with images where the model's body was directly facing the camera and all edges of the clothing top were clearly visible.

Input Image_2:

Predicted Label_2: [red, square neck, sleeveless, no buttons]

Here, the image was classified partially correctly. This neckline type was not technically encompassed in our classes and the sleeve length was predicted incorrectly. This top is "off the shoulder", which creates a flat neckline type we did not train for, and therefore is an example of a limitation of our model. We deduce that the classifier predicted sleeve length incorrectly because an important characteristic of sleeveless tops is that the shoulder is exposed. While the shoulder is exposed here, there are still sleeves, possibly causing the classifier to get confused and predict incorrectly.

Input Image_3:

Predicted Label_3: [green, v-neck, long sleeves, buttons]

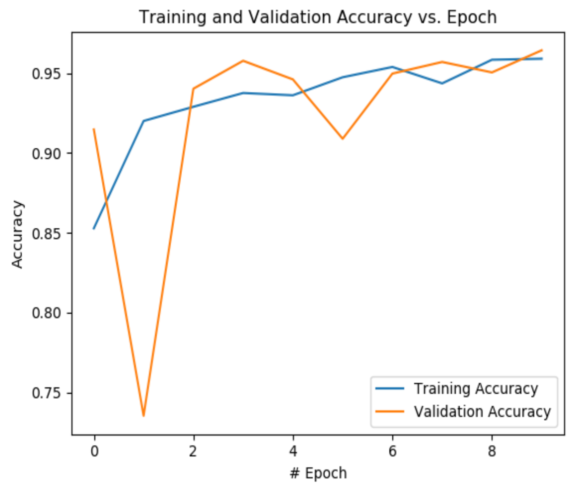
Here, the image was classified mostly correctly. Buttons were found on the bottoms worn by the model, but not on the top, possibly causing the classifier to make a false positive prediction.

Quantitative Results

Below are the confusion matrices and accuracy plots for our final models. We chose these quantitative measures since confusion matrices provide valuable information on which classes the model performs best/worst with, which helps us analyze ways to improve it; furthermore, accuracy plots give insight on the progress of our model throughout the training, and on whether our model is overfit.

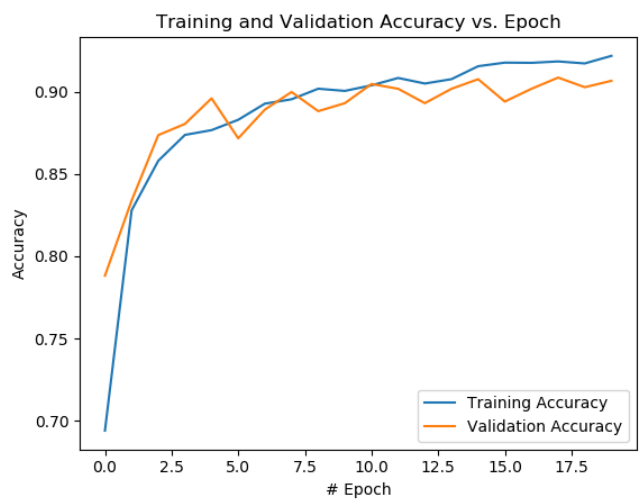
COLOURS

	Black	Blue	Green	Orange	Red	White	Yellow
Black	265	0	1	0	0	1	0
Blue	5	160	5	0	0	6	0
Green	0	1	149	0	0	2	16
Orange	0	1	0	103	1	1	2
Red	1	0	0	19	219	0	0
White	1	0	2	0	0	234	0
Yellow	0	0	2	1	0	5	174



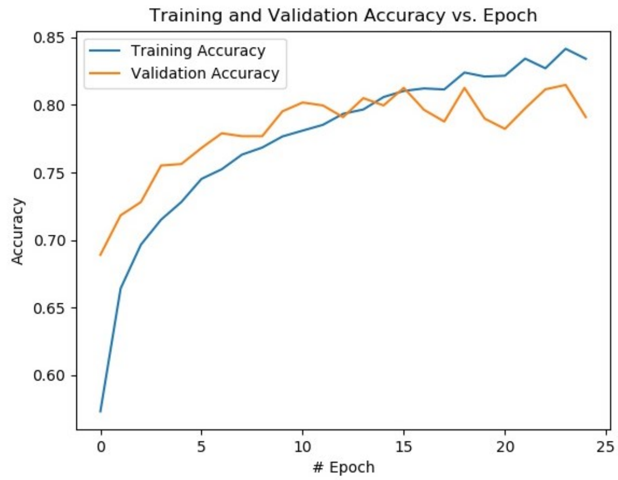
SLEEVES

	Long	Short	Sleeveless
Long	658	21	7
Short	52	604	30
Sleeveless	19	44	623



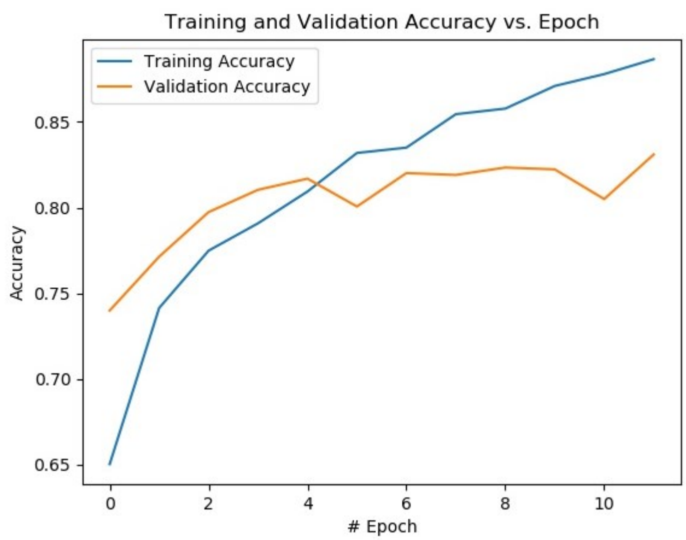
NECKLINES

	Crew	Square	Turtle	V
Crew	293	31	66	47
Square	35	353	7	37
Turtle	51	19	347	16
V	35	36	18	337



BUTTONS

	Button	NoButton
Button	363	73
NoButton	117	372



Discussion & Learning

Based on the results above, we were genuinely surprised with how well our model performed. The confusion matrices and sample outputs revealed that often, incorrect results are reasonably justified, or the source of confusion makes sense. Comparisons to results of the baseline and a random predictor reveal that our models perform significantly better.

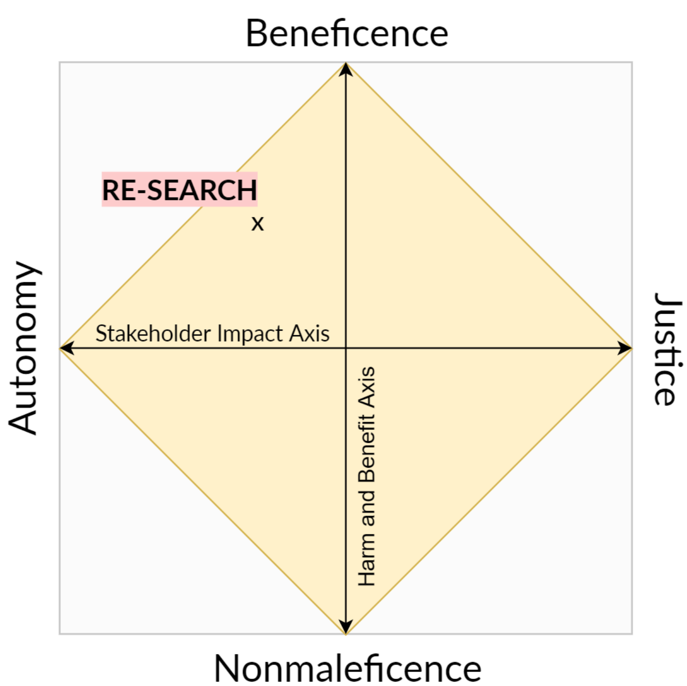
Through this experience we learnt a lot about the best practices for collaborative engineering work – especially about collaboration on code, the end-to-end processes of a machine learning engineer. Given a future opportunity, there are a few things we would change. For one, we realized a market survey or additional research on the target market and the topic would be useful to gain additional insights. In our case, this would include doing more research on popular clothing features to ensure our chosen classes and categories are relevant and useful. We would have also wanted to automate some of our processes from the start including the use of scripts to perform our hyperparameter grid searches. We were also amazed by how much faster our training process was after switching to using CUDA on gpu and wish we had realized this sooner.

As an extension to the project we would have also wanted to expand our scope to include other clothing types or styles. While we think the project is important and useful, it appears we would need to put in more work to widen the scope enough to handle integration into a real clothing store.

Ethical Framework

The main stakeholders to this project include shopping customers, store managers/workers, fashion designers and clothing models. Given our project is implemented by real clothing companies there are some ethical concerns to be aware of.

When placing our project on the reflexive principlism plane we considered it to be more towards the autonomy and beneficence axes. A clothing image classifier can produce more accurate and detailed image tagging without the efforts and costs of manual labelling, resulting in a better search engine experience for the end user. This is mostly beneficial to customers and the clothing store since there is a better customer experience for limited costs. Thus, we placed RE-SEARCH more towards beneficence than nonmaleficence. However, another consequence of our standardized tagging is some autonomy within the search results since our search engine would return results in which our model's confidence is maximized for the classification. This means clothing images that are easier for our model to classify will be prioritized more often. This may also possibly encourage/influence fashion designers to create designs in a more standardize way causing autonomy in the fashion industry. Furthermore, its arguable that by automating the clothing feature tagging process, we are taking away manual labour jobs from store workers.



Reflection

One of the biggest challenges for this project was data acquisition, since we created our entire labelled dataset from scratch. This was a tough task since it involved scraping 20 online stores to gather over 60,000 image samples. Furthermore, to ensure confidence in the quality of our dataset, we had to manually verify the labels to give our models the best shot at a success. In addition to the data collection, we performed data augmentation techniques to generate more data for lacking classes in attempts to create an evenly distributed dataset. In addition to the data, we decided to create 4 separate ML models, for which hyperparameters were individually optimized for.

Reflecting on the outcome of our project, we were pleasantly surprised with the ~80%+ accuracies that we achieved on each model, and their performances in comparison to a random classifier. Beyond the requirements, we also created a database that automatically stores classified labels and images, as part of our functional web application that demonstrates the use case of our project. Through building our database, customer “search engine” prototype and real-time image classification web interface, we also learnt practical skills in SQL, and web development. Lastly, we would like to thank Professor Rose and his team for all their guidance and insights during this project.

Permissions:

Both members: Yanisa Khambanonda and Annie Zhuoer Wang grant permission to post the following on a course website:

- Video
- Final report
- Source Code (master branch)

Resources:

[1] J. Le, “How to Classify Fashion Images Easily Using ConvNets”, *Nanonets*, 2018. [Online], Available: https://nanonets.com/blog/how-to-classify-fashion-images-easily-using-convnets/ [Accessed Oct 25, 2019].

[2] A. Geitgey, “face_recognition”, *Github,* Nov 15, 2019. [Online], Available: https://github.com/ageitgey/face_recognition [Accessed Nov 27, 2019].