PRS-PM-ISY5002
PROJECT REPORT

Imantics.io – Finding Patterns Within Images

TEAM MEMBERS

KUEK YONG JIE ADRIEL (A0229985H) - 1
CHUA HAO ZI (A0229960W) - 2
ANG JENN NING (A0229970U) - 3

For the fulfilment of the Practice Module under the Pattern Recognition Systems (ISY5002) Graduate Certificate as part of the requirements for the Master of Technology in Intelligent Systems

EXECUTIVE SUMMARY & ABSTRACT

In the current modern age, smartphones have become part of a necessity in everyone's daily life. Equipped with high quality cameras, smartphones serve as a viable alternative to high-end cameras for people to take down images of beautiful sceneries or unforgettable moments or even for work-related matters. As we utilise our phones more, these smartphones become a massive trove of images which are disordered, messy and fragmented. There has been introduction of AI to aid in the organisation of images in recent years, but the inbuilt image filtering algorithm within the smartphone has not been completely successful in segregating the images according to their similarities.

Therefore, in this Pattern Recognition Project Module, our team of three would like to tackle this challenge through the creation of an intelligent image archival framework. The envisaged system would be able to discover semantic clusters and perform image retrieval functions for users to effectively filter out the images for storage or deletion through either a text or image query. The framework aims to provide an ease-of-life and intelligent means to organising images/data using different deep learning and pattern recognition techniques.

By adopting some of the techniques taught in the course, and additional research on current State-of-the-art frameworks, we developed an interactive website to showcase the pattern discovery pipeline, where the various feature extraction models were applied on a custom dataset specifically curated as a proof-of-concept. The pipeline performs visual clustering that assigns images into different categories based on their similarities and eventually to produce classification outputs of a set of images that corresponds to the user's text or image queries.

We believe that this system has great value, pushing the boundaries of computer vision and visual understanding. As mobile devices have become the replacement for traditional cameras, and with the surge in device ownership, we project a demand in scale for such a requirement for an intelligent system to handle large amounts of unlabelled data and to make sense of the information. Through this project, we aim to provide a preliminary baseline for such a tool that can benefit and provide a richer user experience on mobile devices.

CONTENTS

1. INTRODUCTION

1.1. PROBLEM DESCRIPTION & BACKGROUND

According to a Comtech study done in 2014, the **third most important consideration** for phone buyers is the quality of smartphones' cameras. This suggests that many phone buyers use smartphones as alternatives to the traditional digital camera. Being slender and slim, smartphones allow users to easily bring to places and take photos as and when they like to. But coupled with that great convenience, most users do not properly segregate these images according to different groups or categories.

As time passes by, the users will accumulate a large number of images in their mobile phone galleries. But the mobile phone does not have the function to effectively categorize the images into different groups, e.g. work-related, birthdays, travels or other groups that users wish to classify into.

With the lack of such capability, users may have to spend hours to categorize the images for archival or deletion. And it will also be difficult for users to find some particular images in the massive gallery of images.

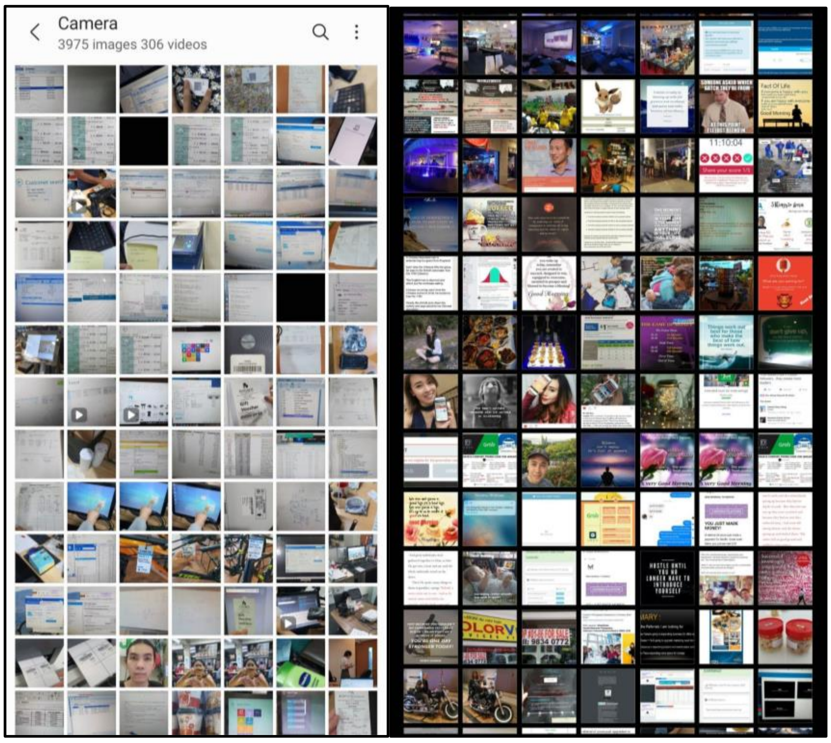


Figure 1: Screenshots of image gallery in mobile phones

1.2. PROJECT OBJECTIVE

In Android phones, the gallery app includes a search function for users to input certain texts to query images that semantically corresponds to the text input. This involves interpreting the text and its semantic relationship to the various images within the gallery and exhibiting relevant images to users. For locations-related text, the image retrieval algorithms are highly relevant and this could be attributed to the location metadata that is tagged to the images when the photos are taken.

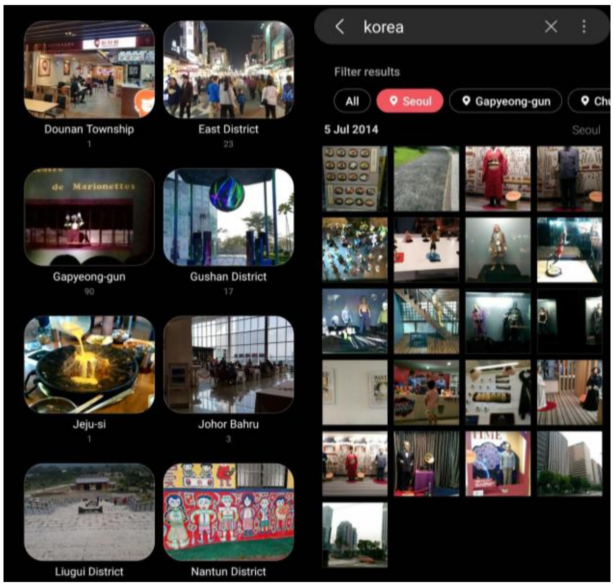


Figure 2: Image tag through geo-location information

However, the method is not as precise when other text queries, like birthday, are processed by the algorithms. The method returns only a few relevant images or no images at all for some cases. In addition, it is not equipped with the capability to do image retrieval based on an image input, unlike Google Images where it allows users to upload images and search the web for websites that contain similar images.

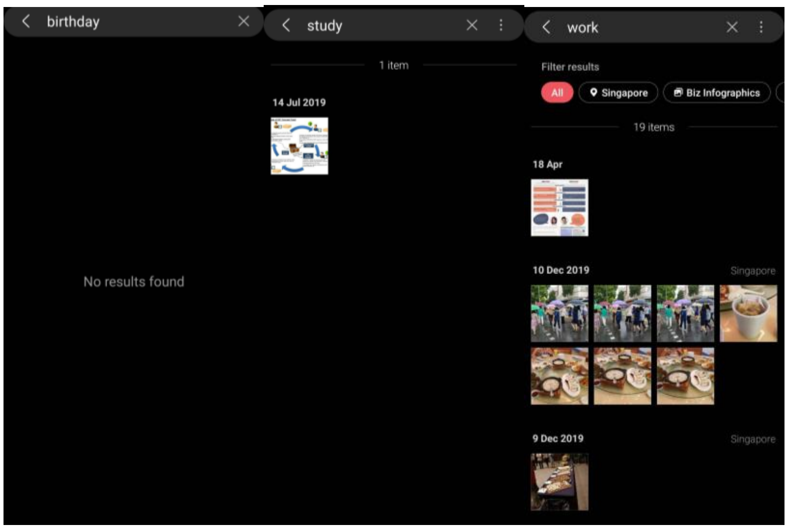


Figure 3: Semantic retrieval on mobile phone gallery images

The team's project objective is therefore to create an image clustering and image retrieval framework that enables users to efficiently search for images that they want in their mobile phones, which have an exponentially increasing number of photos as time goes by.

1.3. REAL-WORLD USE CASE

According to Statista report, the current number of smartphone users in the world today is approximately 6.3 million, and this means 80.76% of the world's population owns a smartphone out of 7.9 million in the year 2021. On the other hand, there is an investigation report done by Keypoint Intelligence that estimated 1.43 trillion photo images in the year 2020. The number will only increase exponentially. In machine perspective, a digital image is fundamentally a 2-dimensional array stored with a given filename. Given these facts, only the human can interpret and grant them true information of an image on what it visually resembles of physical objects, events, and relationships amongst them. It's collectively impossible to label and associate the ever-growing digital images with class information. Hence, most mobile captured images are merely stored with datetime-stamped and running numbers as their respective filenames.

BYOD – Bring your own devices, has been trending and become a workplace norm by default when enterprises do not offer a for-work-only phone. It implicitly stipulates that the workforce be allowed to use one's personally own devices for working purposes such as image taking. It became inevitable when work-life integration came to our personal devices without a good discipline to keep separated or removed as it no longer needed to exist in the device. Thus, over time, the workplaces' images are visually fragmented in one's smartphone gallery. Archiving and deleting these images becomes a routine job and an annoying task to many working adults when the segregation is unsubtly upheld.

Given this real-life scenario, it motivates us to have this project with an initiative to identify and discover patterns within images and to develop an intelligent image search and archival framework. By mapping our motivation, our project idea conceives to enable mobile captured images to be clustered without labelled data annotated and allows users to perform actions on the clustered images which is work-related. Although some similar state-of-art technologies are available in our day-to-day smartphone such as facial detection (to identify friends' faces and label as people), location-detected (to classify with labelled GPS information), and object detection (to identify and classify them with classes information), contemporary smartphones app are not supported to cluster semantical information as example given in following table:

Object	**Semantic**
Laptop, Office, Desk, Chair	Work
Balloon, Cake, Candles	Birthday
National Flag, Red Shirt, People	Singapore National Day
Sky, Mountain, Trees, Rocks	Trekking

Figure 4: Semantic information derived from various objects present within an image

With this work-related segregation, it also extends our exploration to enable the cluster to yield results by querying the above-mentioned keywords "birthday" and "Singapore National Day" instead of the objects that are required to be presented on images (which might not be necessary).

In order to tackle these real-world problems, our project aims to utilise a hybrid learning approach to employ various computer vision (CV) and machine learning (ML) techniques to perform feature extraction, pattern recognition, dimensionality reduction and image clustering to achieve semantic image retrieval.

1.4. DATA REQUIREMENTS

In this project exploration, we will be specifically focusing on image datasets to apply the learning frameworks for discovery. We will be using 2 image datasets in this project:

- One is the ImageNet-1K image subset from the ImageNet Large Scale Visual Recognition Challenge (ILSVRC 2012) which comprises 45,000 images for training, and 5,000 images for testing.

Figure 5: ImageNet 2012 challenge subset dataset with 1000 class categories

- Two is a custom mobile phone gallery dataset jointly contributed by the team members to satisfy the following conditions:
  - The images dataset that is used to train the model need to be curated from the actual mobile phone galleries of human users to simulate an environment of real-life image gallery
  - The images are renamed in chronological order and are saved in JPG file to prevent duplication of image names during collection of data
  - The size of image dataset needs to be sufficiently large enough (>1,000) in order for the model training to be effective
  - When curating the image dataset, images are stored in the database in the raw form, i.e. no resizing of images to standard dimensions.

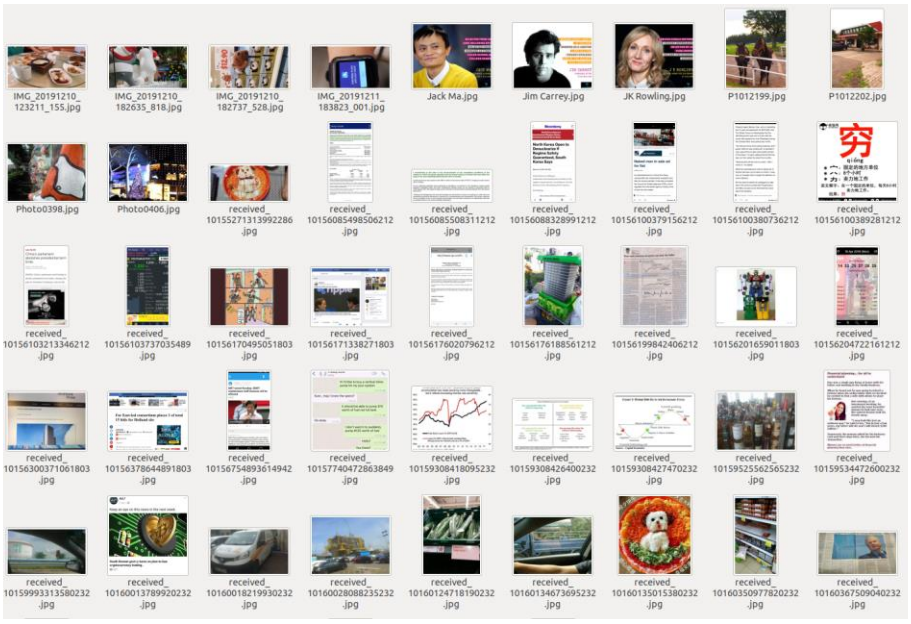

IMG_20191210_123211_155.jpg
IMG_20191210_182635_818.jpg
IMG_20191210_182737_528.jpg
IMG_20191211_183823_001.jpg
Jack Ma.jpg
Jim Carrey.jpg
JK Rowling.jpg
P1012199.jpg
P1012202.jpg
Photo0398.jpg
Photo0406.jpg
received_10155271313992286.jpg
received_10156085498506212.jpg
received_10156085508311212.jpg
received_10156088328991212.jpg
received_10156100379156212.jpg
received_10156100380736212.jpg
received_10156100389281212.jpg
received_10156103213346212.jpg
received_10156103737035489.jpg
received_10156170495051803.jpg
received_10156171338271803.jpg
received_10156176020796212.jpg
received_10156176188561212.jpg
received_10156199842406212.jpg
received_10156201659011803.jpg
received_10156204722161212.jpg
received_10156300371061803.jpg
received_10156378644891803.jpg
received_10156754893614942.jpg
received_10157740472863849.jpg
received_10159308418095232.jpg
received_10159308426400232.jpg
received_10159308427470232.jpg
received_10159525562565232.jpg
received_10159534472600232.jpg
received_10159993313580232.jpg
received_10160013789920232.jpg
received_10160018219930232.jpg
received_10160028088235232.jpg
received_10160124718190232.jpg
received_10160134673695232.jpg
received_10160135015380232.jpg
received_10160350977820232.jpg
received_10160367509040232.jpg

Figure 6: Custom mobile phone gallery dataset with 2000 unlabelled examples across a diverse range of images

2. SYSTEM SOLUTION

2.1. OVERVIEW

Figure 7: Finding Patterns in Images - Imantics.io

Through brainstorming, the team came up with the idea of building a web-based image archival and search framework which we named *Imantics.io*. The core concept behind *Imantics.io* is to provide users with an intelligent image pattern discovery framework to derive semantic understanding for archival and organisation.

The framework is designed as a discovery pipeline beginning with an image dataset input, followed by image feature extraction and feature selection blocks to obtain signal representations for downstream tasks of image clustering and image retrieval respectively. The corresponding image outputs enable the system to discover novel semantics or genres of images through clustering, as well as performing zero-shot classification through retrieval. The discovery pipeline for *Imantics.io* is illustrated in the following figure.

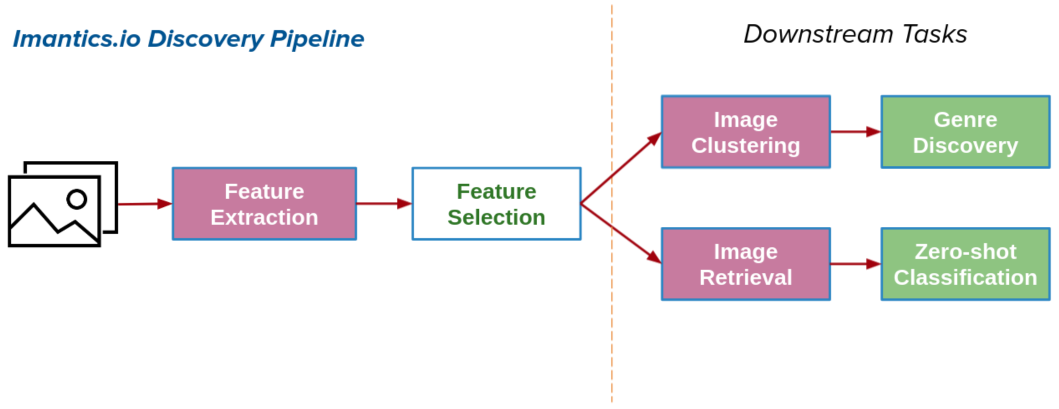


Figure 8: Imantics.io Discovery Pipeline

2.2. FEATURE EXTRACTION

Beginning with the feature extraction block, we explored various deep learning frameworks that had been developed and introduced to the computer vision community. The Very Deep Convolutional Networks for Large-Scale Image Recognition (Simonyan, K. and Zisserman, A. 2015), a.k.a. VGG16, is one of the famous convolutional neural networks (CNN) that achieved 92.7 top-5 test accuracy in the 2014 ImageNet Large Scale Visual Recognition Challenge (*ILSVRC2014*). In this project, this convolution neural network (*CNN*) model is adopted as one of the baseline feature extractors because of its uniform architecture and only consists of 16 convolutional layers as compared with its competing *CNN* architectures – *LeNet*, *AlexNet*, *VGGNet*, *GoogLeNet*, *ResNet* and *ZFNet*. Despite its infamously huge parameters size (138 million) at that point in time, it retained its competitiveness and was widely used to tackle a multitude of vision tasks over the years.

As often regarded and suggested by its name, *CNN* is a deep learning (*DL*) neural network that is embedded with both feature extraction and classification. In simple words, *CNN* provides automatic feature extraction by applying a series of convolution operators over image pixels. In the conventional machine learning feature extraction method, a 2D dimensional RGB-channel image can be pre-processed by computing its x and y pixels with three matrices of red, green, and blue which range from 0 to 255. Handcrafted feature extraction methods often must deal with a large dimension and domain expert knowledge is also required. Deep learning methods like *CNN* on the other hand, allows for the feature extraction process to be learned from the data available through the minimisation of a loss objective function. These types of architecture, in combination with a large image dataset, has proven to be very successful in domain-specific tasks where they have been applied on.

An image feature is a piece of information about content, especially important information for humans to interpret objects and their semantic relationship. However, machines rely on extracting features to identify and discover patterns. The representations of the image can be formulated and indexed in the form of a vector array or embedding tensor for our project's downstream tasks – image clustering and image retrieval. Otherwise, the performance of the downstream tasks would be poor without good features being extracted for clustering and retrieval. While *DL* is still considered a 'Blackbox' on interpretability and explainability especially on the learned *CNN's* features, the table below illustrates how human, and machines interpret features.

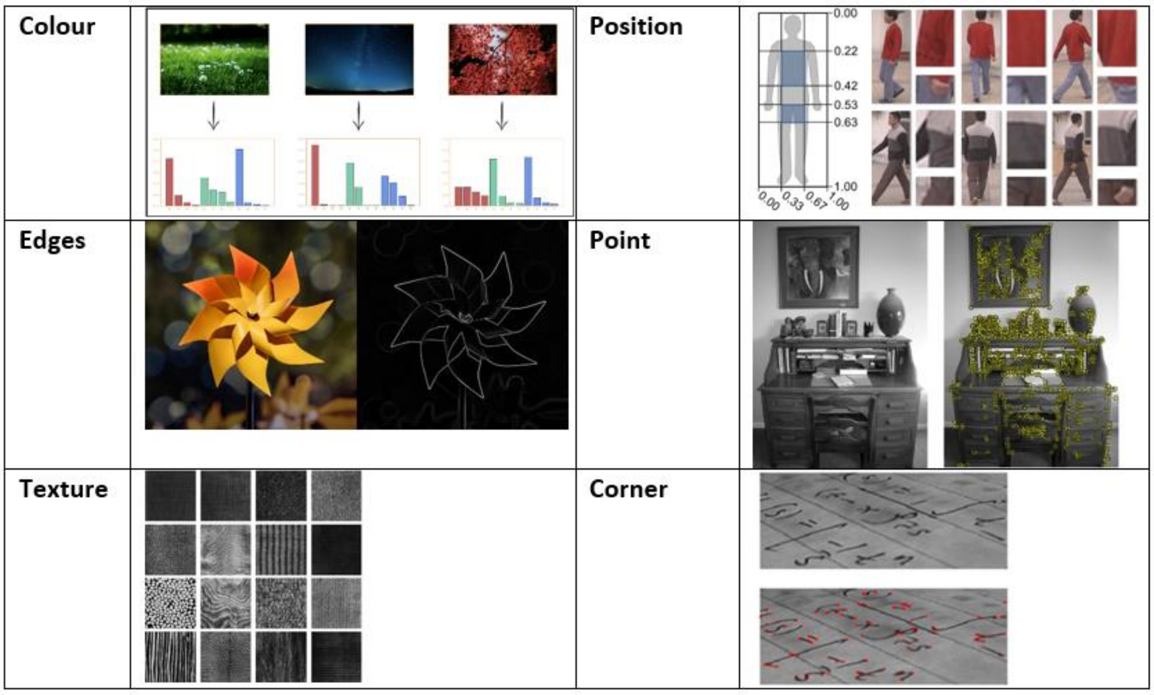


Figure 9: Low level receptive fields feature extraction

2.3. CHALLENGES

In dealing with large image datasets, the team has identified some of the associated challenges:

1. Firstly, a massive dataset size makes the class labelling and annotations on such a custom dataset to be tedious and painful. Without a proper class label, the efficacy of the supervised learning model is greatly impacted. Even with available but noisy labels, it would also have a detrimental effect on the model's performance. Also, when considering the factor of real-world deployment, it is impractical to have mobile phone users to annotate and label the photo after capturing.
2. Secondly, although the pre-trained model on *ImageNet* may provide certain level of generalisability to similar liked datasets i.e. CIFAR10, the performance dips significantly for out-of-domain images or images that the dataset has completely not seen before (for e.g. chat messages, memes etc.). Therefore, there exist an issue of generalisability to our mobile phone gallery kind of images. Given the fact that the photos are taken by mobile phone users across the world from different sources, backgrounds and cultures, the model must be able to be generalized with a strong representation in order to realise and develop such a semantic engine.

3. Finally, while recent deep learning architectures have evolved to enforce learning of high-level features together with low-level features within the images. We found that traditional *CNN* still tends to latch on to low-level feature cues to perform prediction tasks. The idea of a semantic feature representation has not been adequately captured and baked into the feature extraction architecture. In the project exploration, we aim to discover both low-level and high-level visual features to identify semantic patterns through richer feature representations. We believe that strong feature representations are the key to unlocking generalisability in deep learning and crucial step towards building artificial general intelligence.

2.4. SELF-SUPERVISED LEARNING FRAMEWORK

In the approach to dealing with large amounts of unlabelled data, recent works in unsupervised learning have been gaining much attention and traction. In particular, the emergence of self-supervised learning (*SSL*) paradigms has achieved significant breakthroughs in the computer vision domain. Supervised learning has brought about the deep learning revolution with specialist models such as *AlexNet* and *VGG-16*. These deep models have outperformed human-level performance on the ImageNet dataset challenge in which they have been trained. However, these specialist models do not generalise well to novel data that are out-of-domain (*OOD*). In essence, supervised learning presents itself as a bottleneck in moving towards general intelligence as it is simply impractical to label everything we know in this world.

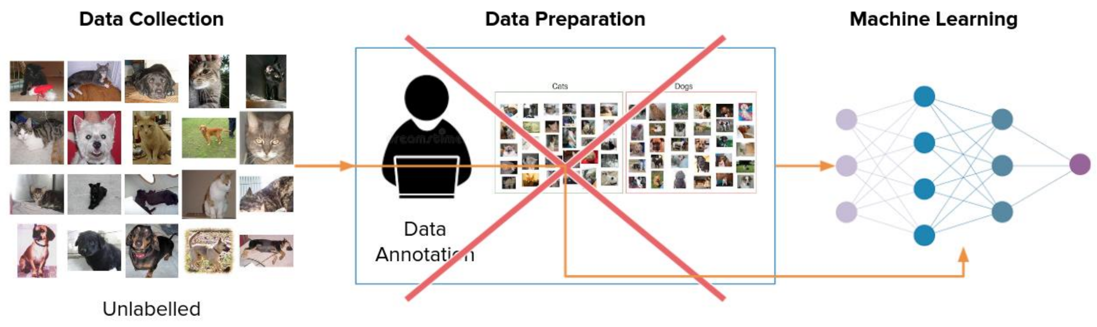


Figure 10: Utilising unsupervised methods to extract features as guiding signals for self-supervised frameworks

At the core of its intuition, *SSL* is based on the hypothesis that generalised knowledge, otherwise known as "common sense", forms the basic building blocks of biological intelligence in both humans and animals. Similar to babies, humans learn about the world largely through observations, forming generalised predictive models and hypotheses. Subsequently, we can apply such background knowledge to accomplish novel tasks with great accuracy. One good example would be the ability to drive reasonably well and safely in a brand new foreign country just by utilising the knowledge of driving in our own cities.

SSL aims to obtain supervisory signals from the data itself, learning strong and important representations that are deemed critical. This is usually done with orders of magnitude of data to be able to generalise well. Typically, the generic framework for *SSL* is to predict any unobserved or hidden part of the input from the observed or unhidden portions in the input. One example that is used to a good extent in training language models is the random masking of parts of a sentence to predict missing words from the remaining unmasked words.

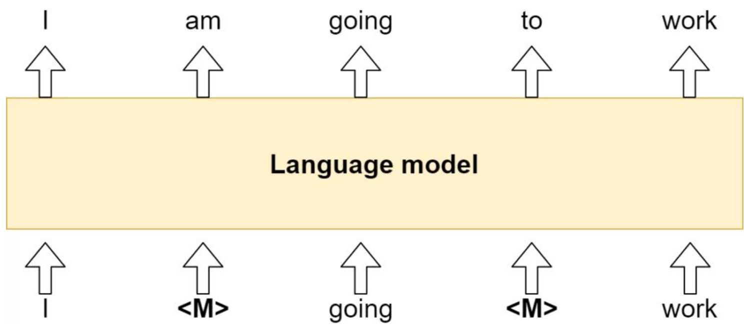


Figure 11: Masked language modelling for token representation learning

With Masked Language Modelling (*MLM*), *SSL* methods have long seen successes in Natural Language Processing (*NLP*) domains with large-scale pretraining that produced State-of-the-Art (*SOTA*) frameworks such as BERT, RoBERTa and OpenAI's *GPT* models. However, this still remains largely a challenge on the computer vision domain front. Primarily due to the fact that visual images represent a continuous distribution with higher dimensions as opposed to text tokens, which can be discretized based on a finite set of vocabulary. When performing masking on small patches within images, and then tasking the model to accurately predict it would be a non-trivial task due to the fact that there could be an infinite amount of possibilities which would be difficult to model.

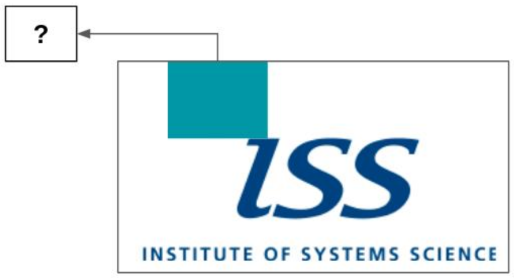


Figure 12: An infinite number of possibilities when predicting masked patches from image data which represents a high-dimensional continuous data distribution

To address the problem of working with image data, Yann LeCun (LeCun, 2021) introduces *SSL* training as an Energy-Based Model (*EBM*), where the system is given 2 inputs (x and y) and trained to tell how compatible of incompatible they are with each other. To indicate compatibility, the system will produce a value, also known as the energy value, which will be high for poor compatibility and low for good compatibility. The model is then iteratively shown random examples of x and y, and trained to produce low energy for positive samples and high energy for negative samples. This framework of training therefore introduces the idea of *Contrastive Learning* to tackle *SSL* challenges with image data.

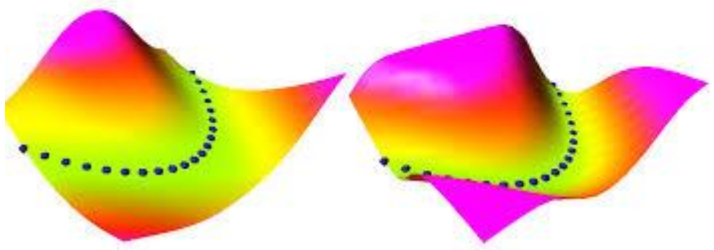


Figure 13: EBM training through a contrastive loss regime. Pushing down the energy along the contour for compatible pairs while pushing up the energy at all other places along the manifold (incompatible pairs)

The notion of *SSL* training through some form of contrastive learning can be framed as a *pretext task*, whereby the aim is to enable the model to learn good data representations rather than solving the task at hand. There are a wide range of pretext tasks for images, from recovering input from corruption e.g. cross-channel auto-encoders (colourisation) (Zhang et al., 2017), to the forming of pseudo-labels through transformation of a single exemplar image through patch orderings otherwise known as "*solving jigsaw puzzles*" (Noroozi & Favaro, 2016). In the next segment, we will introduce the pretext task of instance discrimination through image augmentation to train a simple model through contrastive learning.

2.4.1. PRETEXT TASK - INSTANCE DISCRIMINATION

In instance discrimination, data augmentations such as crop, resize, flip, colour distortion, rotate, and noise addition that are performed on the whole image or portions of the image to constraint the contrastive learning methodology. The idea is that two augmented versions of the same image (positive pairs) should derive similar representations (or lower energy when viewed through the *EBM* lens), while two other augmented versions of different images (negative pairs) should produce very different representations (high energy).

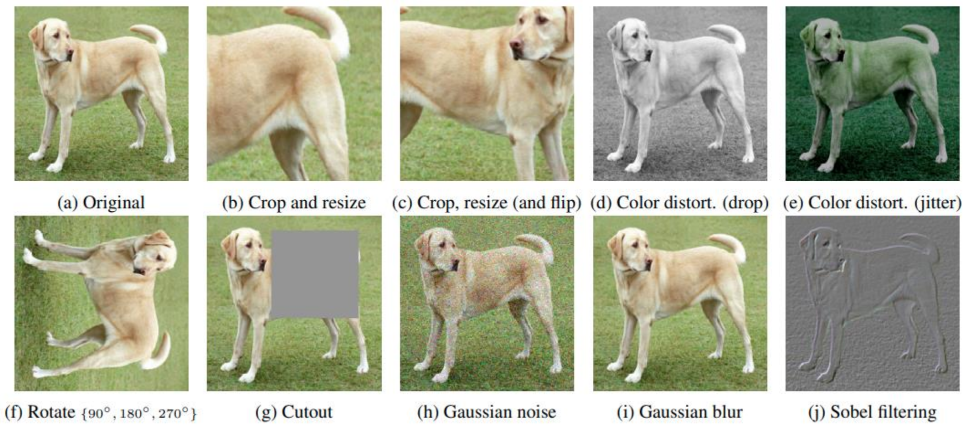
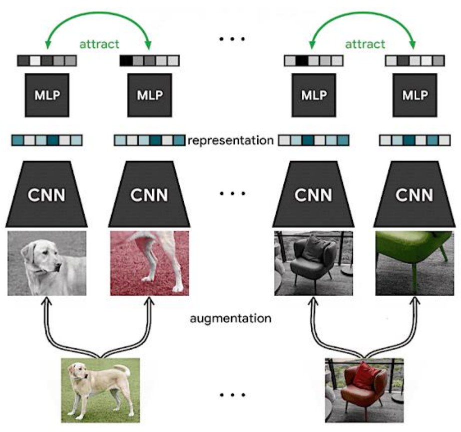

(a) Original
(b) Crop and resize
(c) Crop, resize (and flip)
(d) Color distort. (drop)
(e) Color distort. (jitter)
(f) Rotate {90°, 180°, 270°}
(g) Cutout
(h) Gaussian noise
(i) Gaussian blur
(j) Sobel filtering

Figure 14: Common data augmentation operators used in SimCLR for ablation studies. Final augmentation policy for SimCLR only includes random crop, colour distortion and Gaussian blur.

The level and composition of data augmentation is important in *SSL* frameworks as it is directly correlated to the strength of the learned representations as compared to data augmentation for feature extraction through supervised methods. This framework is introduced in Geoffrey Hinton's Google Brain group in their seminal paper which will be discussed in the next segment.

2.4.2. SIMPLE FRAMEWORK FOR CONTRASTIVE LEARNING OF VISUAL REPRESENTATIONS (SIMCLR)

A Simple Framework for Contrastive Learning of Visual Representations (Chen et al., 2020), otherwise known as *SimCLR*, is a widely popular paper which has received more than 2.3k citations since its release in 2020. *SimCLR* learns representation by maximising agreement between differently augmented views of the same data example via a contrastive loss in the latent space.



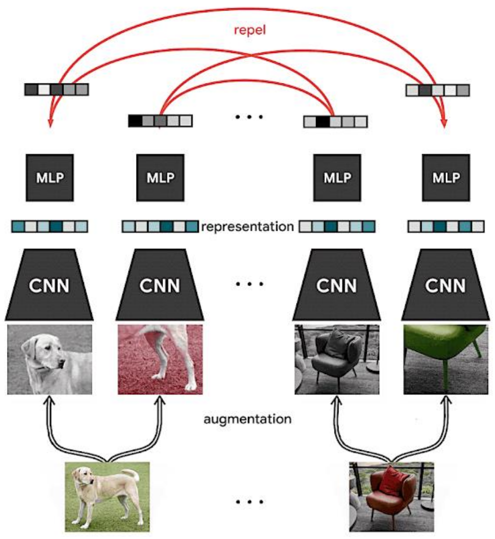


Figure 15: SimCLR framework for contrastive learning to enforce similarities within augmented forms of the same image and repulsion for augmented forms of different images

A random sample of a mini-batch of *N* examples is taken to mine both the positive and negative samples. Stochastic data augmentation transforms given data through a series of image manipulations to produce augmented data examples. The *CNN* base encoder (*ResNet-50*) is used to do feature extraction from the augmented data examples. This is then passed into a small multi-layer perceptron (*MLP*) projection head that maps the representations into vector space where a contrastive loss, known as the normalised temperaturescaled cross entropy loss (*NT-Xent*), is applied.

$$\ell_{i,j} = -\log \frac{\exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\mathrm{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

The base model is trained on the ImageNet *ILSVRC-2012* dataset and evaluated with a linear classifier trained on top of the frozen base model, which is the de-facto linear evaluation protocol on the dataset. The evaluation results showed that *SimCLR* was able to match a fully supervised *ResNet-50* model's accuracy at the expense of longer training duration, wider and deeper networks as shown in the figure below. The paper postulates that *SimCLR* benefits from both a strong data augmentation composition and a larger model to derive robust visual representations.

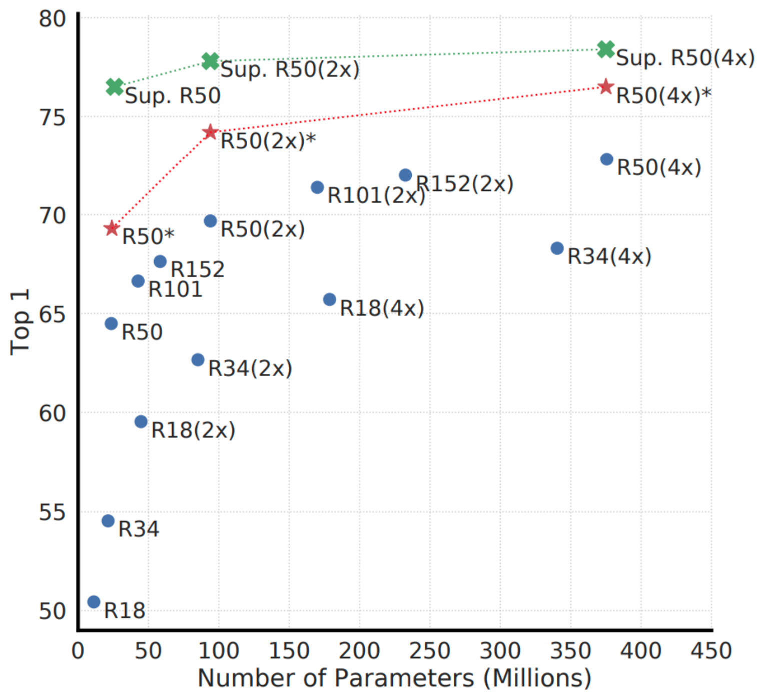


Figure 16: Linear evaluation comparison between fully supervised methods (green dots - ResNet) trained for 100 epochs and simCLR (blue dots - ResNet base)

We utilise *SimCLR* pre-training for feature representation together with Momentum Contrast (*MoCo*), which would be introduced in the next segment, in the Semantic Clustering Through Adopting Nearest Neighbours (*SCAN*) framework. The *SCAN* architecture and its summary will also be highlighted in the following segment.

2.4.3. MOMENTUM CONTRAST FOR UNSUPERVISED VISUAL REPRESENTATION LEARNING (MOCO)

Also based on contrastive learning, Momentum Contrast (*MoCo*) for Unsupervised Visual Representation Learning (He et al., 2020) was designed as a way of building large and consistent dictionaries for unsupervised learning with a contrastive loss. *MoCo* reframes the sample pair mining problem as building dynamic dictionaries, where the "keys" in the dictionary are sampled from data (Image patches) and an encoded "query" would perform a dictionary *look-up*. The matching key and query should be similar, and dissimilar to others.

MoCo builds on the desired basis that the dictionaries should be large and consistent as they evolved during training, in order to better sample the underlying high-dimensional and continuous visual space. Unlike *SimCLR* where only one single feature encoder is used, *MoCo* uses two encoders, which are known as the encoder for the query input and the momentum-based moving average encoder for the dictionary keys respectively. Both encoders are fed augmented versions of the same input image to produce a query and corresponding keys. The keys are then queued for subsequent usage. During the training phase, positive pairs are constructed from queries of keys from the current mini-batch, while negative pairs are formed from queries of current mini-batch and keys from previous mini-batches. The dynamic evolving nature of the dictionary ensures that good features can be learned from a large dictionary which covers a rich set of negative samples.

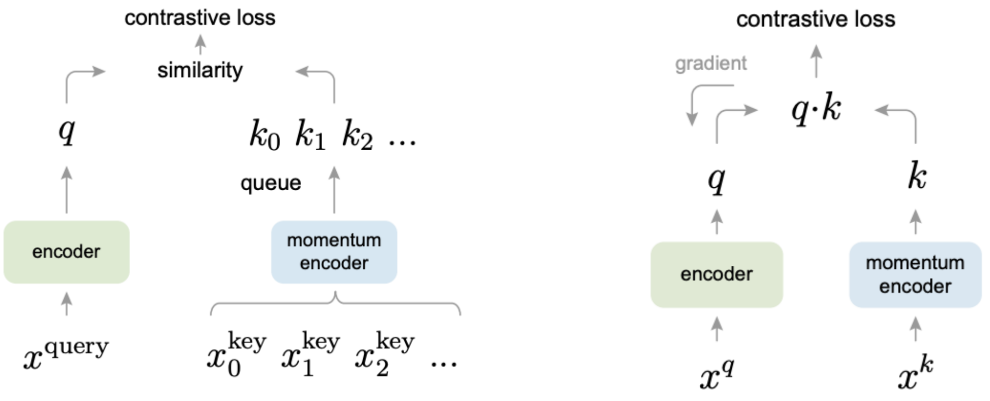
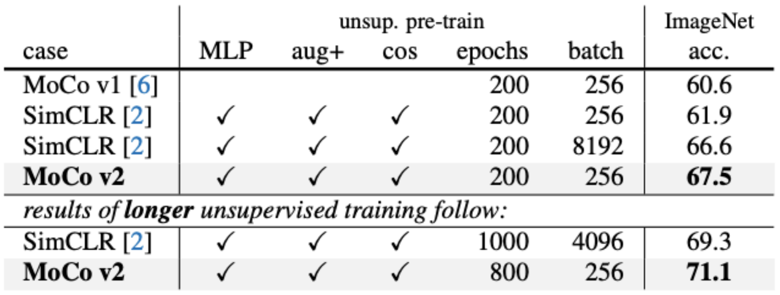


Figure 17: MoCo's query and momentum encoders on 2 branches working on augmented versions of the same input images. During training, query encoder is updated via back-propagation while the momentum encoder is updated by linear interpolation

The parameters of the momentum encoder are updated in consistency with the encoder despite its evolution, and this momentum contrast update enables strong performance in the learned representations. *m* denotes the momentum coefficient taking a value between 0 and 1, while *θκ* and *θρ* are the momentum and query encoders respectively.

$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}$$

MoCo also introduced *shuffling BatchNorm* to tackle the issue of information leak when utilising mini-batch sampling. It was found that the model was able to "cheat" the pretext task to find a low-loss solution due to possible communication among samples during the *BatchNorm* update. Therefore, the authors proposed to shuffle *BatchNorm* and train on multiple GPUs to ensure independent sampling. In the study, *MoCo* experimented with *ImageNet-1M* dataset with ~ 1.28 million images, and *Instagram-1B* dataset with ~ 940 million public images from Instagram to represent a long-tailed and unbalanced distribution of real-world data.The training objective loss function was closely similar to that of *SimCLR*, utilising the Information Noise Contrastive Estimation (*InfoNCE*) loss to enforce contrastive learning. With these innovations, *MoCo* reports competitive results on linear evaluation with *SimCLR*, with the newer version update *MoCo v2* pushing by a wide performance margin.



	unsup. pre-train					ImageNet
case	MLP	aug+	cos	epochs	batch	acc.
MoCo v1 [6]				200	256	60.6
SimCLR [2]	✓	✓	✓	200	256	61.9
SimCLR [2]	✓	✓	✓	200	8192	66.6
MoCo v2	✓	✓	✓	200	256	**67.5**
results of ***longer*** *unsupervised training follow:*						
SimCLR [2]	✓	✓	✓	1000	4096	69.3
MoCo v2	✓	✓	✓	800	256	**71.1**

Figure 18: MoCo vs SimCLR ImageNet linear classifier accuracy

2.4.4. SELF-DISTILLATION WITH NO LABELS (DINO)

In *SimCLR* and *MoCo*, the feature encoders are largely still based on convolutional neural networks to extract meaningful representations during the pretext task. In a recent paper by *Facebook AI Research* titled Emerging Properties in Self-Supervised Vision Transformers (Caron et al., 2021), the authors explored an innovative approach for *SSL* on Vision Transformers. Transformers have gained huge popularity in recent years and have produced many *SOTA* models, in particular within the *NLP* and speech domain. At the heart, Transformers use a correlation computation architecture, also known as the *Attention Mechanism*, to mimic how a human would determine the notion of saliency and focus on areas that are of importance.

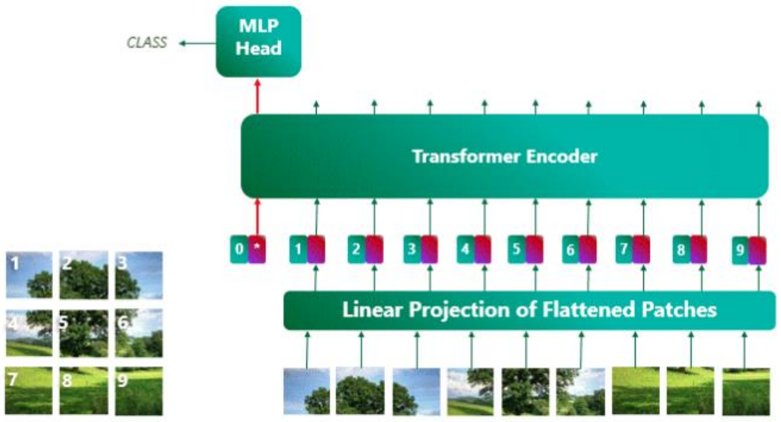


Figure 19: Vision Transformer segments an image into non-overlapping patches and transformed into vectors similar to that in word sentence

The authors introduced a framework to training a Vision Transformer through a method known as *Self-Distillation*. The model, named Self-Distillation with No Labels (*DINO*), employs 2 networks - a teacher branch and a student branch. The student network is then trained through the process of knowledge distillation to simply match the output of the teacher network over different augmented views of the same image. In knowledge distillation (Hinton et al., 2015), a model is typically trained with a small network to mimic the output of a larger network to compress the model. Self-training then aims to improve the quality of features through the propagation of an initial set of annotations to a larger set of unlabelled instances.

In *DINO*, the 2 networks share an identical architecture, the Vision Transformer (*ViT)* (Dosovitskiy et al., 2020). The instance discrimination task follows a multi-crop data augmentation strategy, where 2 patches of great dimensions and partial overlap are obtained from the same input image. The idea is to provide a *"global view"* for the teacher network, and the smaller crop view goes to the student network to encourage a *"Local-to-Global"* correspondence for contextual analysis.

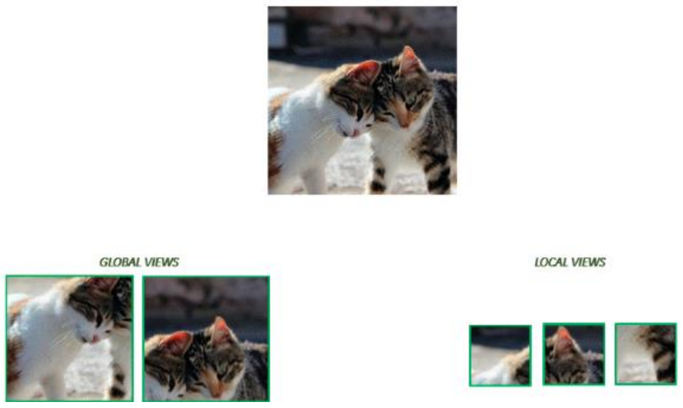


Figure 20: Multi-Crop strategy to enforce local-to-global correspondence for strong feature representation learning.

Following in the footsteps of MoCo, *DINO* utilises a momentum update rule where the teacher network's weights are updated via an *Exponential Moving Average* (*EMA*) from the student branch. When both the networks output the same embeddings regardless of the input, model collapse during training can be prevented. During the student training, information learned from the student network can then be shared to the teacher network through this method to perform classification based solely on the global views given to it.

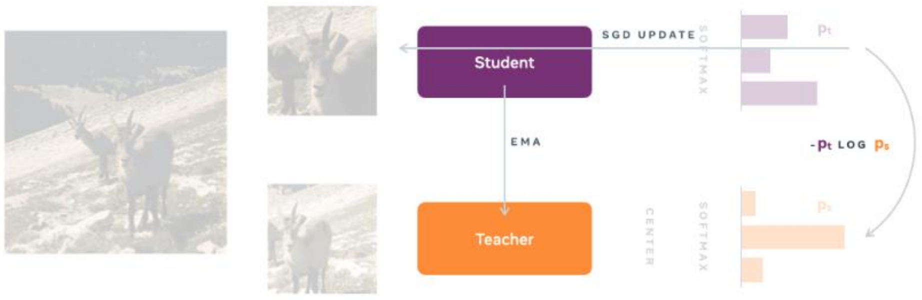


Figure 21: Self-Distillation through augmented views of the same image.

Each of the networks outputs a *K* dimensional feature that is normalised with a temperature softmax over the feature dimensions. Feature similarities are then measured through a cross-entropy loss, and a stop-gradient operator initiated on the teacher branch to allow gradients to be propagated only through the student network. Finally, the trained network of the teacher or student branch can be utilised to perform inference to produce an image feature vector with *K* dimensions to be passed on to the next block to perform other downstream tasks.

Similarly, *DINO* was pre-trained on the ImageNet dataset to benchmark with existing *SSL* frameworks for representation learning. As a result of the training regime, *DINO* showed strong performance in both linear evaluation and K-Nearest Neighbours accuracy, surpassing both *MoCo* and *SimCLR* by more than a margin of 5%. *DINO* also can show its learned representations through the self-attention [cls] token on the heads (6 heads) of the last layer.

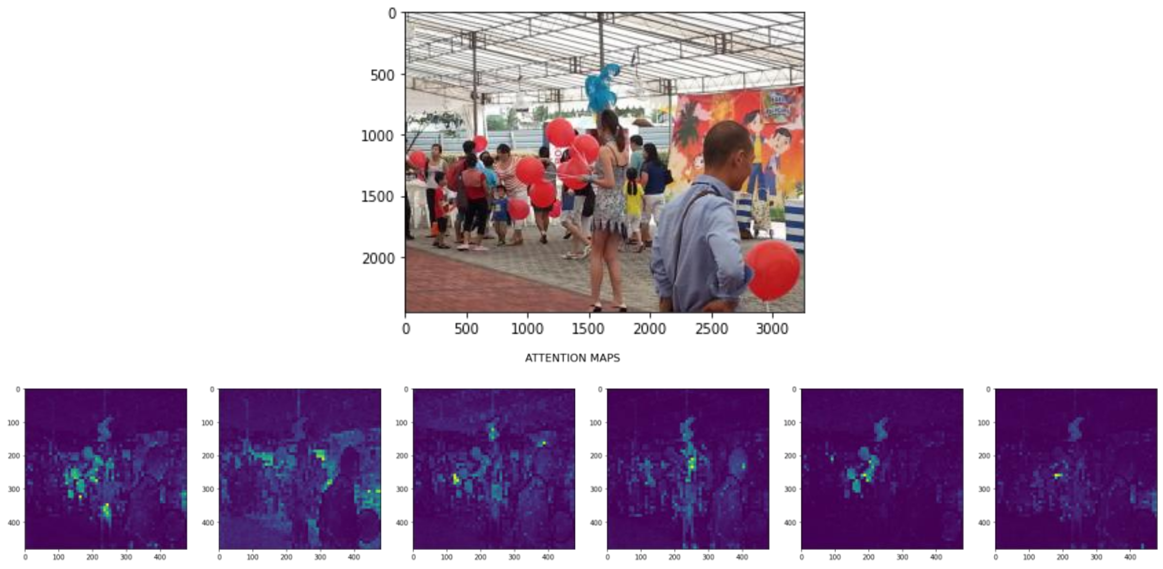


Figure 1: Attention maps of 6 heads at last layer of a sample image from Imantics.io custom mobile phone gallery dataset showcases model's ability to focus on salient portions of the image through SSL

From this, we can see that *DINO* is able to understand object semantics within images and this feature will be important in our adaptation to derive strong and meaningful clustering and retrieval for Imantics.io. In addition to feature representation learning, *DINO* also was able to perform remarkably well in subsequent downstream tasks such as *Copy Detection, Image Retrieval* and *Image Segmentation* which further displays the strength of *SSL* of Vision Transformers on large-scale and unlabelled datasets. One of the strategies that we had adopted would be to apply the strong feature representations from *DINO* along with the different clustering algorithms to retrieve semantic clusters for an unlabelled dataset.

2.4.5. SEMANTIC CLUSTERING THROUGH ADOPTING NEAREST NEIGHBOURS (SCAN)

In clustering, one of the pertinent challenges often faced is in deciding on the optimal number of clusters when presented with the data at hand. In real-world conditions, there may not be access to ground-truth labels or that the number of semantic classes are not *a priori* known. Most of the time, domain expert knowledge may be applied to heuristically determine an approximate number of clusters. Other times, trial and error through a grid-search method for a range of cluster values would provide some insights.

When we perform image clustering through *k-means* using the features derived from supervised learning methods such as *VGG-16* as highlighted earlier, we observed problems with *cluster degeneracy* from the cluster results. The definition when a cluster solution is degenerated is when there are 1 or cluster centres with no entities allocated. Usually for degenerate clustering outputs, some of the traits observable are a lesser than ideal number of clusters derived, or multiple clusters collapsing into 1 big cluster.

Semantic Clustering by Adopting Nearest Neighbours (*SCAN*) (Gansbeke et al., 2020) aims to tackle this issue through a 2-step framework combining *SSL* representation learning and mining nearest neighbours.

In the first step, *SCAN* uses features extracted through one of the *SSL* methods *MoCo* or *SimCLR* to mine nearest neighbours of each image based on feature similarity. It was empirically found that most entities within the nearest neighbours belong to the same semantic class and would provide a good baseline for semantic clustering.

In the second step, the semantically meaningful nearest neighbours obtained are integrated as a prior into a learnable approach. A neural network (*nn*) classifier is defined to group each image and its nearest neighbours through a clustering loss to produce both consistent and discriminative (one-hot) predictions. The *nn* $\Phi\eta$ with weights η terminates in a softmax function to perform a soft label assignment over the cluster $C = \{1,...,C\}$. The probability of sample X_i being assigned to cluster c is denoted as $\Phi^c\eta(X_i)$. The objective function, termed semantic clustering loss, is minimised through training.

$$\Lambda = -\frac{1}{|\mathcal{D}|}\sum_{X\in\mathcal{D}}\sum_{k\in\mathcal{N}_X}\log\langle\Phi_\eta(X),\Phi_\eta(k)\rangle + \lambda\sum_{c\in\mathcal{C}}\Phi_\eta'^c\log\Phi_\eta'^c,$$

$$\text{with } \Phi_\eta'^c = \frac{1}{|\mathcal{D}|}\sum_{X\in\mathcal{D}}\Phi_\eta^c(X).$$

Where 〈.〉 denotes the dot product operator, and λ is an entropy term which spreads the predictions uniformly across the cluster C. In our mobile phone gallery dataset, we experimented with a range of entropy values with 4.0, 4.5 and 5.0 to evaluate the clustering results. We found that for a limited diverse set of images such as our mobile phone gallery, an entropy value of between 4.0 to 4.5 and below is required to be able to train the *nn* while the ImageNet-2012 dataset (50k images) was able to be trained with an entropy value of 5.0.

SCAN also provides an additional step called fine-tuning through self-labelling after training semantic clustering loss to exploit well-classified examples in step 2 and correct for mistakes due to noisy nearest neighbours. The step involves the selection of confident samples through a probability thresholding as "prototypes". Data augmentation is then done on the confident samples and a cross-entropy loss is applied to update the weights for the obtained pseudo-labels. In our experimentation, it was found that due to the lack of confident samples from the output from Step 2, our custom mobile phone gallery dataset was not able to be fine-tuned through self-labelling. The ImageNet-2012 dataset however, was able to perform self-labelling with a confidence threshold set at 0.96 with the number of clusters set to 1000 to match the original ImageNet classification labels. Self-labelling was reported to be able to boost classification accuracy across the different *k* values.

ImageNet	**50 Classes**				**100 Classes**				**200 Classes**			
Metric	Top-1	Top-5	NMI	ARI	Top-1	Top-5	NMI	ARI	Top-1	Top-5	NMI	ARI
K-means	65.9	-	77.5	57.9	59.7	-	76.1	50.8	52.5	-	75.5	43.2
SCAN*	75.1	91.9	80.5	63.5	66.2	88.1	78.7	54.4	56.3	80.3	75.7	44.1
SCAN†	76.8	91.4	82.2	66.1	68.9	86.1	80.8	57.6	58.1	80.6	77.2	47.0

Figure 22: Validation set results for k = 50, 100 and 200 respectively on the ImageNet-2012 dataset. Feature extraction was based on MoCo. SCAN denotes results after step 2 semantic clustering loss and SCAN† denotes results after step 3 self-labelling.*

We will utilise *SCAN's* clustering framework (with *MoCo* and *SimCLR* feature representation extraction), in conjunction with *DINO* representations and the various clustering methodology to study and compare the various clustering results on both ImageNet-2012 and our custom mobile phone gallery datasets.

2.5. PROPOSED CLUSTERING MEASUREMENT METRICS

The clustering quality of the different models are evaluated based on two main metrics: Normalized Mutual Information (*NMI*) and Adjusted Rand Index (*ARI*).

- **Normalized Mutual Information (NMI)**: *NMI* gives an indication of the clustering quality. *NMI* measures reduction in uncertainty and reduction in entropy of class labels when given the cluster labels (i.e. entropy decreases with decrease in uncertainty). *NMI* is useful as a comparison of different clustering models that have different numbers of clusters since *NMI* is normalized. NMI ranges from 0 to 1, with 1 representing perfectly complete labelling. Generally, *NMI* > 0.7 indicates relatively good clustering.
- **Adjusted Rand Index (ARI)**: The Rand Index computes a similarity measure between two clusters by considering all pairs of samples and counting pairs that are assigned in the same or different clusters in the predicted and true clusters. This can be used to compare actual class labels and predicted cluster labels. The adjusted Rand index is the corrected-for-chance version of the Rand index. The adjusted Rand index (*ARI*) ranges from 0 to 1, with 0 representing random labelling independent of the number of clusters and samples, and 1 representing identical clusters.

2.6. CLUSTERING FRAMEWORK

In our approach to an intelligent image archival system, we will explore various image clustering algorithms including the frameworks that were introduced during the course of the study and more. The following table summarises the different clustering frameworks explored:

Hierarchical Clustering	Partition Clustering	Density-Based Clustering
Agglomerative	K-means	DBSCAN
	k-means++	OPTICS

Figure 23: Clustering Frameworks

For k-means (random and k-means++ initialisation), the challenge when tackling a novel unlabelled dataset is in determining the optimal value for k. One of the common methods used in helping with the selection of k-value is the *Elbow Method* where the goal is to select the value of k just before the Sum of Squared-Errors (*SSE*) starts to display diminishing returns. However, for our case of the custom mobile phone gallery dataset, this method did not prove to be useful as the elbow was not clearly identified as seen in the figure below.

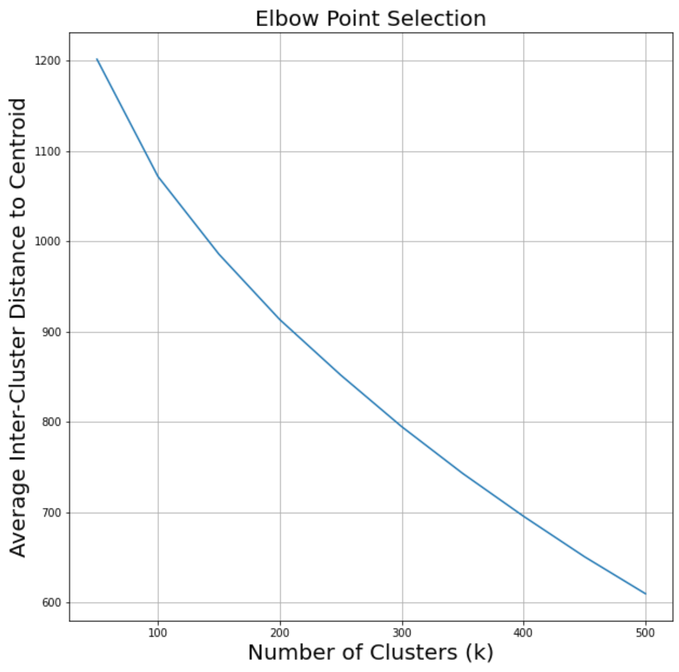


Figure 24: Elbow Point selection based on SSE vs k clusters on custom mobile phone gallery dataset.

To overcome this, our team selected to perform a grid-search iteration across different k-values and select the optimal clustering parameters based on the number of images within cluster distribution, presence of cluster degeneracy (please refer to segment 2.4.5 *SCAN* algorithm for elaboration) and visual inspection.

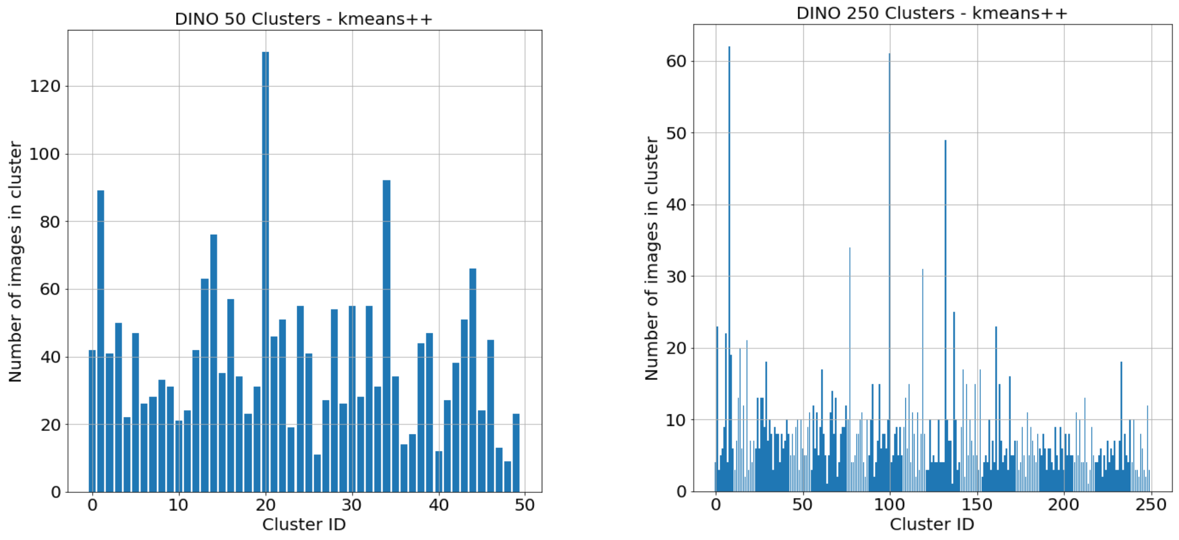


Figure 25: k-means++ clustering utilising DINO features with 50 clusters vs 250 clusters.

Due to the random initialisation of cluster centroids for k-means and k-means++, the resulting clusters retrieved are not repeatable with every iteration producing different *SSE* values depending on where the start point is sampled. In comparison, we also explored the use of bottom-up Agglomerative Clustering that iteratively computes cluster memberships through a distance measurement metric. Agglomerative clustering removes the uncertainty in a priori determining the optimum number of clusters at the start, with the algorithm automatically evaluating the cluster numbers based on a distance measure and threshold value. Based on our findings, we observed that although Agglomerative Clustering is slightly more computationally intensive due to the algorithm having to go through each cluster point to build up the hierarchical dendrogram, it provides more parametric control for fine-tuning and the results are repeatable. In our results, we found that the Agglomerative approach returns the highest *NMI* and *ARI* scores on the *ImageNet-2012* validation data. In addition, we observe that the agglomerative method returns a more even distribution spread of cluster entities across the dataset as compared to the k-means method.

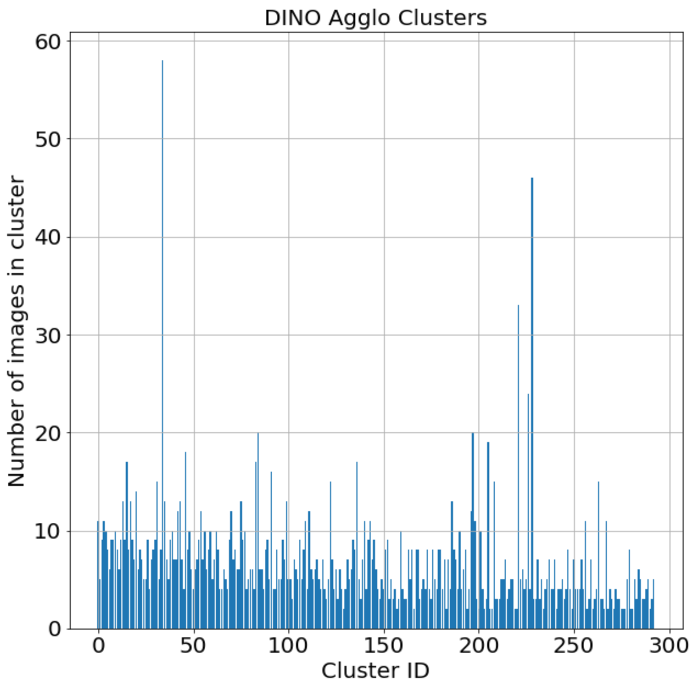
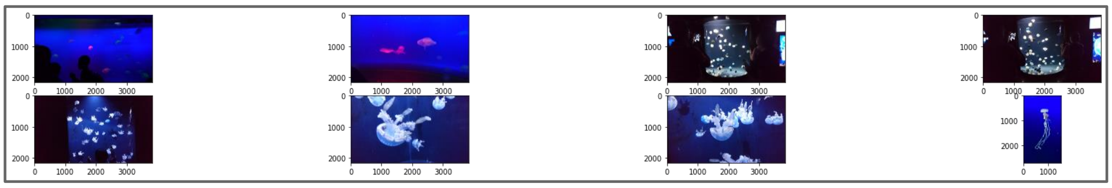


Figure 26: Agglomerative Clustering of DINO features on our custom mobile phone gallery dataset. Clusters discovered = 293.

Cluster 5



Cluster 12

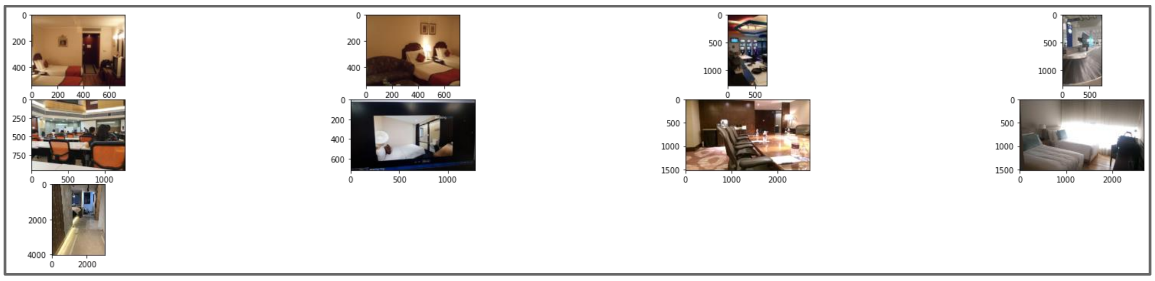


Figure 27: Sample clusters 5 and 12 from Agglomerative clustering showing semantic clustering based on DINO features.

One of the other prior assumptions when performing k-means is that the data that we have follows a spherical-shape distribution. This may not be true for real-world image gallery data, as different people may tend to capture certain scenes more than others. In dealing with clusters of arbitrary shapes, density-based frameworks have been proven to be more efficient in obtaining clusters while maintaining robustness against noise and outliers. The two density-based algorithms explored in the project are Density-Based Spatial Clustering of Applications with Noise (*DBSCAN*) and Ordering Points to Identify Cluster Structure (*OPTICS*).

DBSCAN estimates the clustering density by counting the number of points in a fixed-radius neighbourhood or epsilon value. It then determines whether 2 points are connected only if they lie within each other's neighbourhood. Parameters of *DBSCAN* include the Eps-neighbourhood of a point and MinPts, which denotes the minimum number of points in the neighbourhood. *DBSCAN* introduces the concept of *core points*, *border points* and *noise points*. Should the algorithm be unable to determine whether a specific point comes under the core or border category, it would be discarded under the noise point bin. In our experiments, we observed that there are a large number of images that fall into the noise bin, and therefore only effectively clustering a small subset of the whole image dataset.

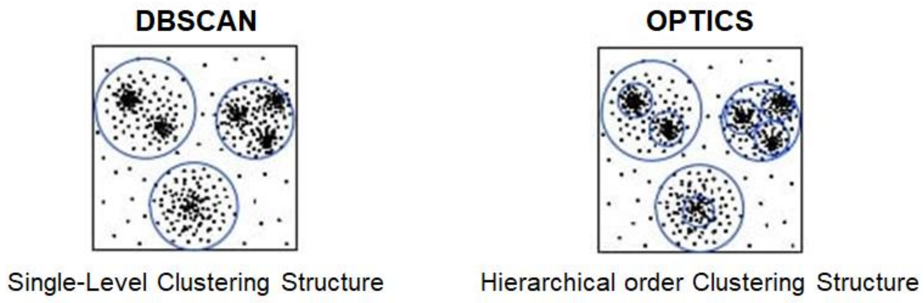

DBSCAN
OPTICS
Single-Level Clustering Structure
Hierarchical order Clustering Structure

Figure 28: Single level vs Hierarchical order for density-based clustering frameworks

OPTICS can be seen as an extension to *DBSCAN* where the only difference is that it does not assign cluster memberships, but instead, stores the order in which points are processed. *OPTICS* introduces the concept of *core distance* and *reachability distance*. For each point randomly selected, *OPTICS* iteratively computes and updates a reachability distance within the neighbourhood. On a high-level, *OPTICS* can be viewed as a hierarchical ordering of clustering points within the data as compared to the single-level clustering structure seen in *DBSCAN*. Also due to the algorithm process, *OPTICS* is more computationally intensive as compared to *DBSCAN*. However, we observed in our results that *OPTICS* discards more data points into the noise bin as opposed to *DBSCAN* while enforcing a more even distribution of memberships within the clusters. *DBSCAN* on the other hand, is observed to tend to collapse multiple clusters together in one big cluster, thereby producing similar results when cluster with *SCAN* framework.

More details of the clustering analysis and results will be further highlighted in the performance and results segment in the later part of this report.

2.7. RETRIEVAL FRAMEWORK

For the image retrieval functions in our project, the function will allow users to either input a text or upload an image which will then be used to retrieve images from the dataset that corresponds to the user's queries. And to implement that, this requires two phases: one is to get feature extractions from the query, and second to implement the image retrieval model.

For image query, we utilized the same *DINO* model as mentioned in Section 2.4.4. to extract the features from the image uploaded. Using the visual representations created by the model, we pass these into the *K-Nearest Neighbours* model to extract semantically similar images from the image dataset based on the k-values that the user selects. This will return a series of images that are similar to the uploaded image.

For the text query, we utilized a large scale contrastive pretrained model *CLIP* (Contrastive Language-Image Pretraining) to extract textual representations from the text inputs. And similarly, we pass these into the *K-Nearest Neighbours* model to retrieve images from the dataset that corresponds to this textual representation.

The *CLIP* model is a result of the embodiment of the various works in zero-shot transfer, natural language supervision and multimodal learning. *CLIP* is a neural network model that is trained on 400 million image-text pairs and can associate the visual representations with the textual representations of the image-text pairs. This association algorithm can be used for visual classification tasks, which in our case can be used to retrieve images from the dataset that semantically corresponds to the textual query input by the user.

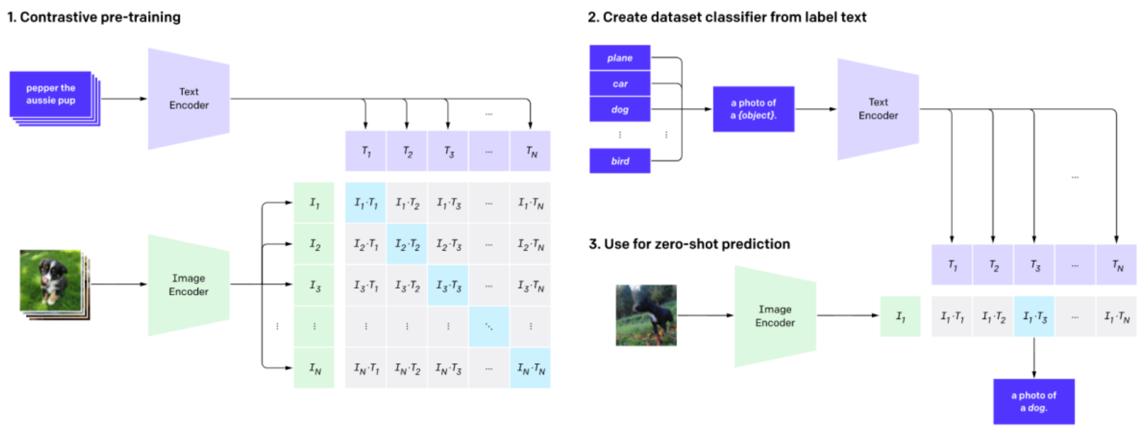


Figure 29: CLIP 3-steps framework

CLIP model first trains the image and text encoders to establish the association between the image and texts in the dataset. This algorithm is then used as a zero-shot classifier. This classifier is then further used to predict the textual representations of the input image in the form of a caption. For our use case, the reverse was implemented instead where the textual inputs from the user are passed through this classifier to produce the associated visual representation, which is then further used to retrieve semantically similar images from the dataset through *k-NN*.

2.8. SYSTEM ARCHITECTURE

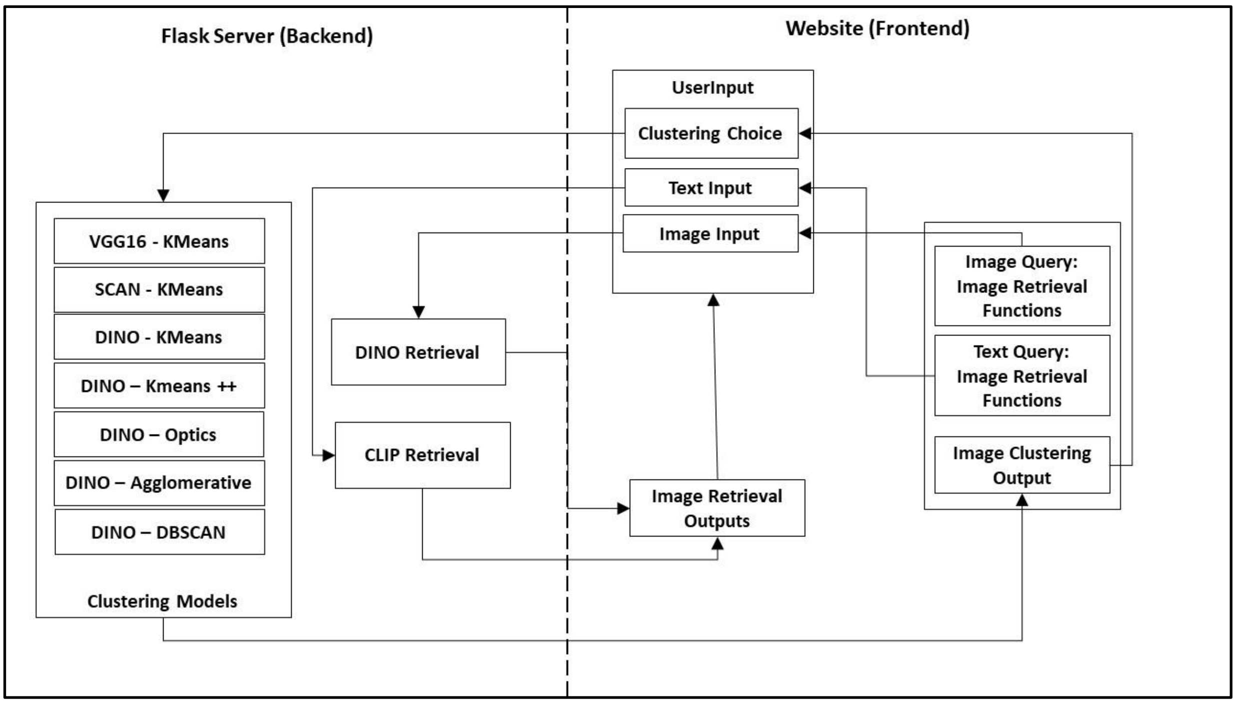


Figure 30: Overview of Imantics.io System Architecture

The system architecture is developed on a Flask server as the backend which provides the serving of the query displays to the web frontend. User queries serves as input to the 2 retrieval functions (text-based and image-based) or the selection of display for the clustering outputs from the different models.

2.8.1. WEB-UI ARCHITECTURE

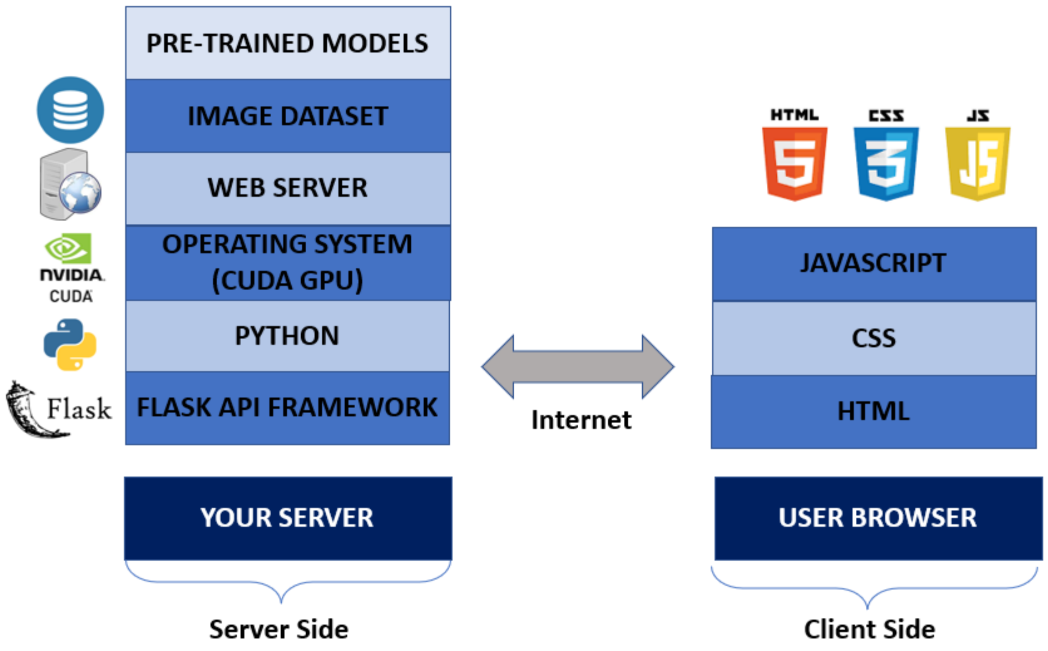


Figure 31: Imantics.io Software Stack

The server-side software stack is built on using Python using the Flask API framework, while the client-side stack comprises of the usual HTML/CSS/JS structure for display and interactions. The system is currently hosted on local machine as a proof-of-concept but is scalable to an online web-based UI with the addition of a cloud hosting infrastructure. In this project, we focus primarily on the evaluation of the pattern recognition framework and would not be developing an online web UI for demonstration.

The web architecture below shows the top level data flow sequence in which users interact with the web-page to display the image clusters or perform an image retrieval task.

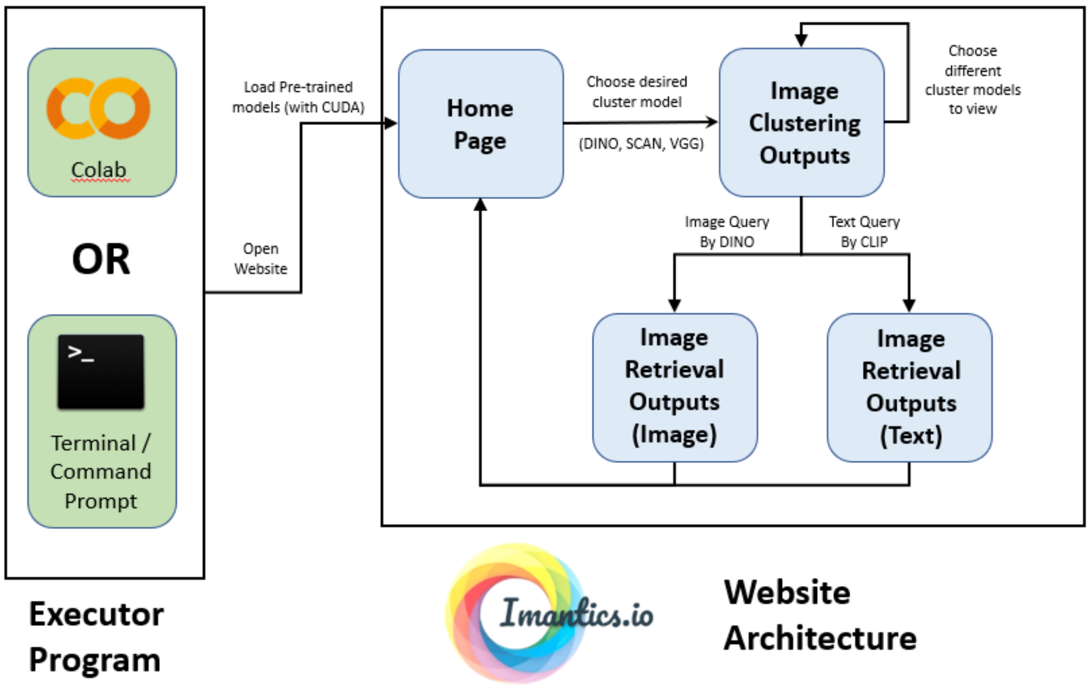

Colab
OR
Terminal / Command Prompt
Executor Program
Load Pre-trained models (with CUDA)
Open Website
Home Page
Choose desired cluster model
(DINO, SCAN, VGG)
Image Clustering Outputs
Choose different cluster models to view
Image Query By DINO
Text Query By CLIP
Image Retrieval Outputs (Image)
Image Retrieval Outputs (Text)
Imantics.io
Website Architecture

Figure 32: Imantics.io Website Architecture

2.9. SYSTEM IMPLEMENTATION

The system is fully developed on Python (3.7) utilizing various feature extraction models, i.e. VGG16, DINO, SCAN, CLIP, via PyTorch (1.9.1) library for model training. The system is implemented together with Flask API, and website building blocks of HTML/Javascript/CSS to form the core for backend and frontend interaction framework for the website architecture.

S/N	Software Lib/Models	Usage	Version/url
1.	PyTorch	Model Training	1.10.0+cu102
2.	Torchvision	Model Training	0.11.1+cu102
3.	CUDAToolkit	Model Training	11.0
4.	Flask	Web Framework	2.0.1
5.	Werkzeug	Web Application	2.0.1
6.	DINO	Feature Extraction, Image Retrieval	https://github.com/facebookresearch/dino
7.	SCAN	Feature Extraction	https://github.com/wvangansbeke/Unsupervised-Classification
8.	VGG16	Feature Extraction	Keras.application library
9.	CLIP	Image Retrieval	https://github.com/openai/CLIP.git -> With additional modifications
10.	Numpy	Misc Computation and Plotting	1.20.1
11.	Pandas	Misc Computation and Plotting	1.2.3
12.	Scikit-learn	Misc Computation and Plotting	0.24.1
13.	Matplotlib	Misc Computation and Plotting	3.3.4
14.	Seaborn	Misc Computation and Plotting	0.11.1
15.	Statistics	Misc Computation and Plotting	1.0.3.5
16.	Openpyxl	Misc Computation and Plotting	3.0.9
17.	ftfy	Misc Computation and Plotting	6.0.3
18.	regex	Misc Computation and Plotting	2021.10.8
19.	tqdm	Misc Computation and Plotting	4.62.3
20	natsort	Misc Computation and Plotting	7.1.1

Figure 33: Software libraries and model lists

2.9.1. CLASS DIAGRAM

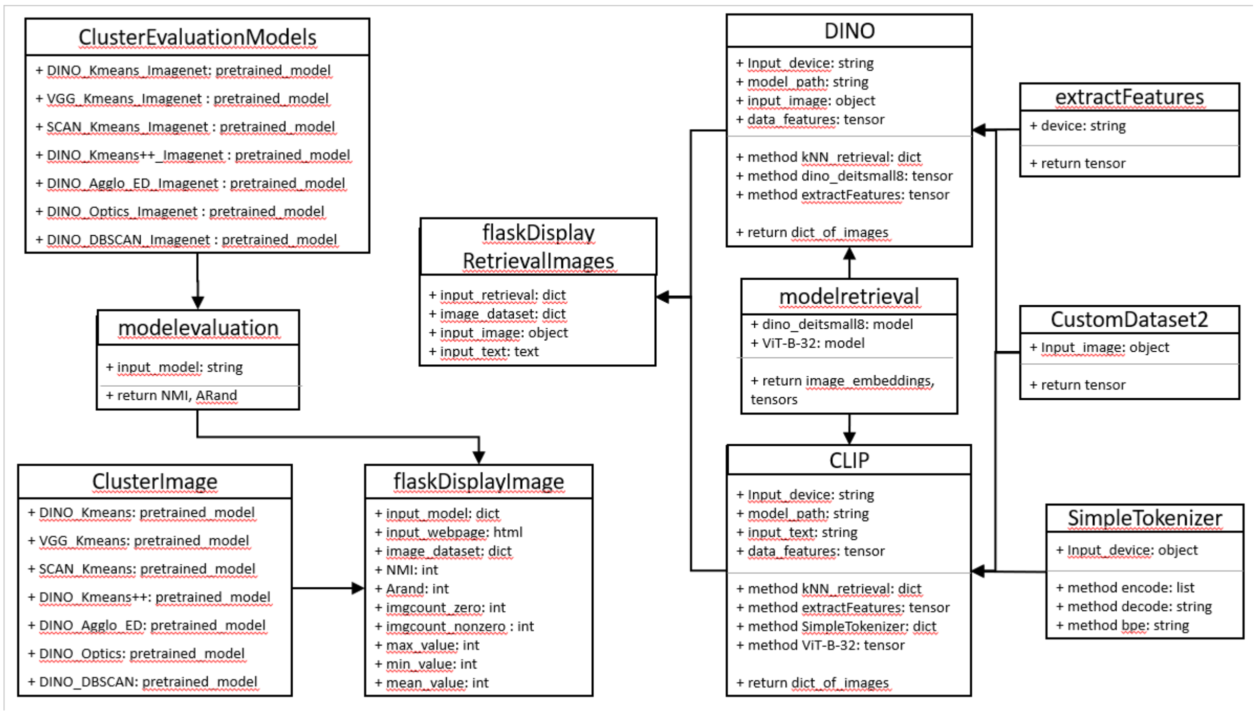


Figure 34: Imantics.io Class Diagram

2.10. ASSUMPTIONS

In the project evaluation, here are the list of assumptions that we have made:

1. The custom mobile phone gallery dataset is sufficiently diverse enough to allow for semantic clusters to be formed. Should the dataset be sparse, or having insufficient diversity, the clustering results would suffer degradation and exhibit degeneracy.

2. The custom mobile phone gallery dataset only has static images. In this framework, our team would not be handling dynamic content such as videos, gifs etc, as it is beyond the scope of the project.

3. PERFORMANCE AND RESULTS

3.1. MODEL RESULTS ON IMAGENET-2012

The evaluation of the model is conducted with 5,000 images from the *ImageNet* dataset. As the customized 2,000 image dataset that is curated from the mobile phone galleries does not come with pre-defined class labels, it would be very difficult to do the evaluation unless the clusters are pre-labelled manually. Due to the high difficulty of the task in a short period of time, the evaluation was conducted using the *ImageNet* dataset instead.

Different models with different parameters are trained with the ImageNet Dataset. The clustering output is then evaluated using the *NMI* and *ARI* metrics as introduced earlier. The table below captures the clustering summary for the different model frameworks:

Name of Model (evaluated with 5,000 images from Imagenet Dataset)	Cluster size	Normalized Mutual Information (NMI)	Adjusted Rand Index (ARI)
DINO Agglomerative Clustering -- Euclidean Distance of 1.2	1449	0.90	0.45219
DINO Agglomerative Clustering -- Euclidean Distance of 1.5	813	0.88	0.42122
DINO Agglomerative Clustering -- Euclidean Distance of 2.0	402	0.83	0.27819
DINO DBSCAN Clustering -- Epsilon 08, Minimum Samples 2	470	0.55	0.00179
DINO DBSCAN Clustering -- Epsilon 08, Minimum Samples 3	297	0.48	0.00141
DINO DBSCAN Clustering -- Epsilon 09, Minimum Samples 2	442	0.64	0.00586
DINO KMeans++ Clustering	1000	0.88	0.41066
DINO KMeans Clustering	1000	0.86	0.32650
DINO Optics Clustering -- Minimum Samples 3	460	0.60	0.00260
DINO Optics Clustering -- Minimum Samples 4	311	0.55	0.00215
DINO Optics Clustering -- Minimum Samples 5	211	0.48	0.00163
SCAN KMeans Clustering	149	0.50	0.00022
VGG16 KMeans Clustering	1000	0.78	0.10787

Figure 35: Performance Table

Sample images from the different clustering models with good clustering quality and bad clustering quality can be observed in Appendix C.

3.2. DISCUSSION AND OBSERVATIONS

From the clustering results, we can observe that agglomerative hierarchical clustering gives the best *NMI* and *ARI* scores amongst the rest of the models. In the agglomerative framework, we fixed and utilised the linkage criterion as ward linkage which minimises the Error-Sum-of-Squares (*ESS*) when fusing the clusters. We perform a grid-search across distance threshold parameters from 0.5 to 3.0 with a step size of 0.1. As agglomerative clustering traverses the entire dataset to compute and build a full dendrogram for cluster assignment, it is also the most computationally heavy amongst the cluster analysis frameworks that we have tested.

For both *DBSCAN* and *OPTICS* density-based clustering frameworks, we observe that a substantial number of images were thrown into the noise bins (Cluster -1). Approximately more than half of the dataset was determined to be noise points and not taken into consideration in the final clustering output. As a result, the density-based frameworks only managed to cluster a small subset of the entire dataset as compared to the other clustering methods. While baseline methods like *k-means* endeavour to assign a cluster id to every image present in the dataset, resulting in some of the noise points being grouped together, the density-based frameworks enforce a “stricter” view on cluster memberships, thus producing cluster assignments with lesser noise. Therefore, through the process, we found that it is highly heuristic when determining the optimal cluster parameters to find a balance between selecting noisier clusters or discarding away more data.

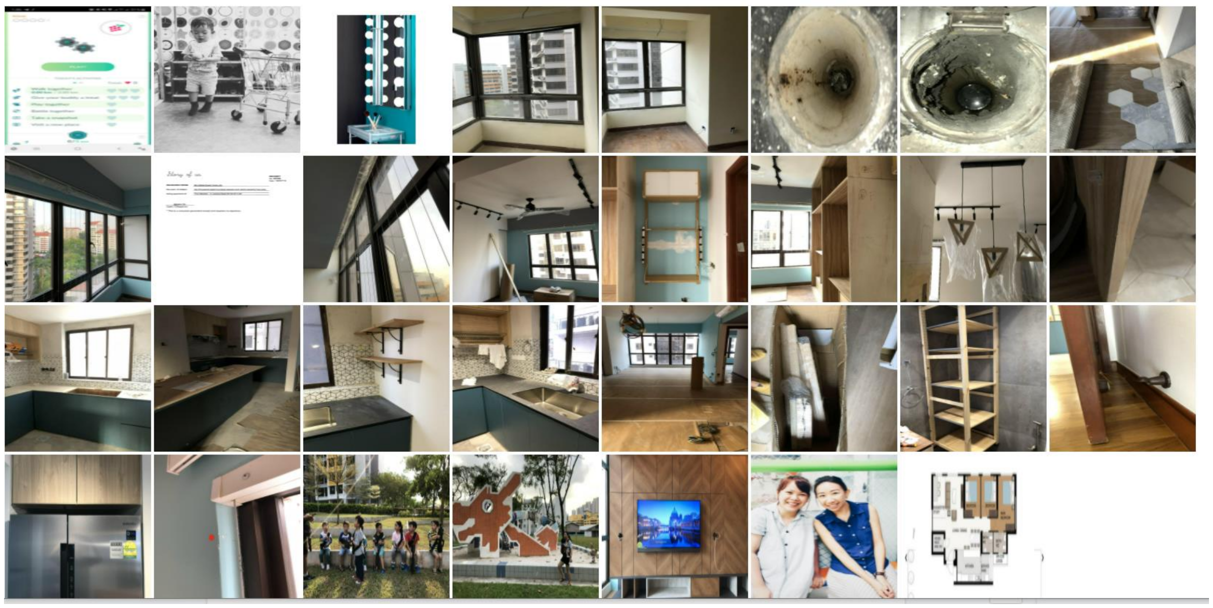


Figure 36: Cluster output (DINO + K-means++), where k=100. Positive semantic cluster with some noise.

Overall, *DINO* with *K-means* clustering returns largely positive clusters with the inclusion of noise data points as highlighted. From the results, we were able to inspect and determine that while some of the images may be visually diverse, they still generally retain a common cluster semantic space. In the above figure, we can identify the cluster being related to home renovation/home furniture.

For the case of *SCAN*, the cluster distribution within the dataset is automatically discovered through the fine-tuning step with a semantic clustering loss. Similar to *K-means*, we were able to retrieve even cluster distributions across the dataset. However, we observe that several the cluster membership assignments show multiple smaller sub-clusters within a larger cluster. This phenomenon can be observed in the figure below where we see semantic ideas such as inspirational posters, advertisements and infographics, and graphs all collapsing into one big cluster. One of the primary reasons is due to the relatively small-sized quantity for the custom mobile phone gallery dataset, where through the mining of the nearest neighbours, the model was not able to train and converge with a high entropy regularisation value. Therefore, in order for the model to achieve convergence, we had to relax the regularisation term, and this had a slightly negative impact on the final clusters derived. We postulate the *SCAN* would work better with a much larger and diverse dataset, like that of *ImageNet,* to be able to reduce the occurrence of such noisy clusters at the output of the training.

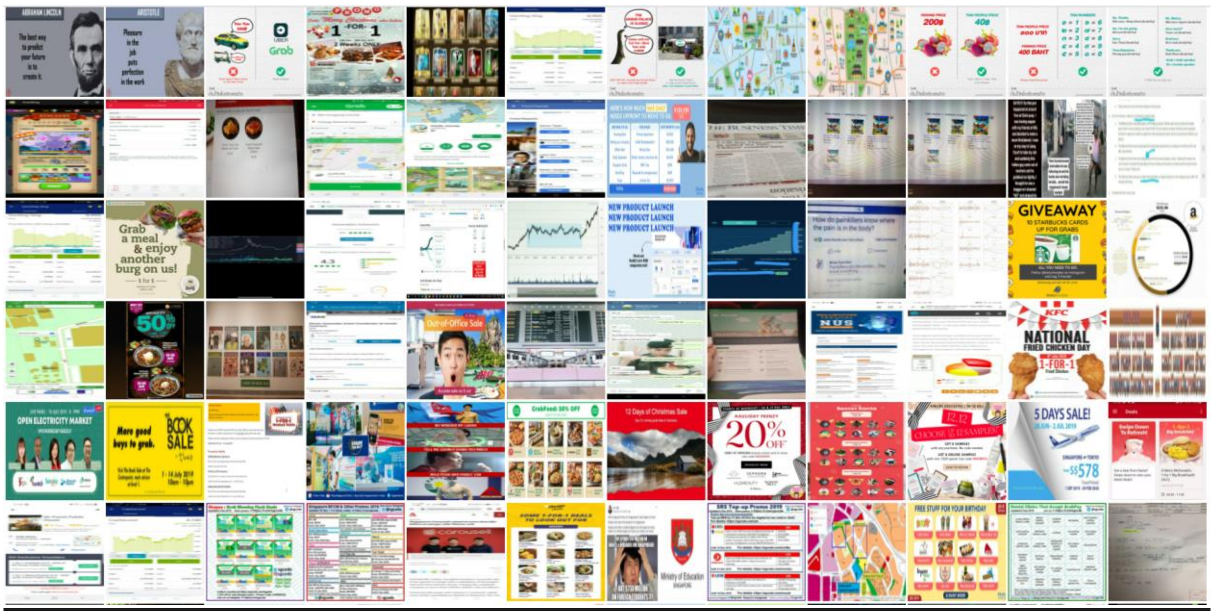


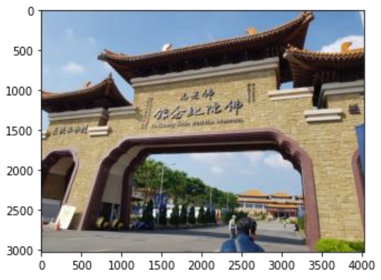
Figure 37: Cluster output (SCAN + K-means), cluster_size=149. A Noisy cluster with multiple smaller sub-clusters collapsing into one big cluster.

Based on the models shown, each of the frameworks has its strengths and weaknesses. While currently the output does not perform any form of ensembling or fusion to improve the overall results, this would be one of the proposed future work that will be discussed slightly more in detail under the future work section subsequently.

3.3. MULTIMODAL RETRIEVAL

In the multimodal retrieval framework, we employ *DINO* feature representations with top *K-nearest neighbours* using the Cosine distance measure for image-based retrieval. With its strong visual attention, the retrieval results through *DINO* were observed to be semantically similar and contextually accurate. One example of a retrieved neighbourhood can be viewed in the figure below:

***DINO* Input Image Query**



***DINO* Top 10 Neighbours**

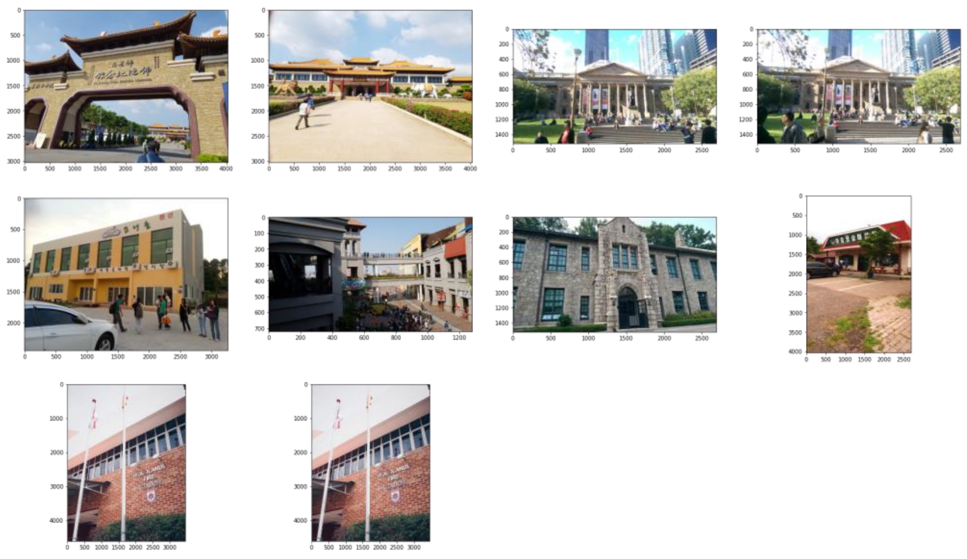


Figure 38: DINO Top 10 neighbours for Image Retrieval with kNN

The model was able to search and discover closely related themes of old historical architecture buildings that were of close match to the original image query. The results highlight *DINO*'s capabilities in retaining semantic information that can be propagated to visually diverse images for discovery and retrieval.

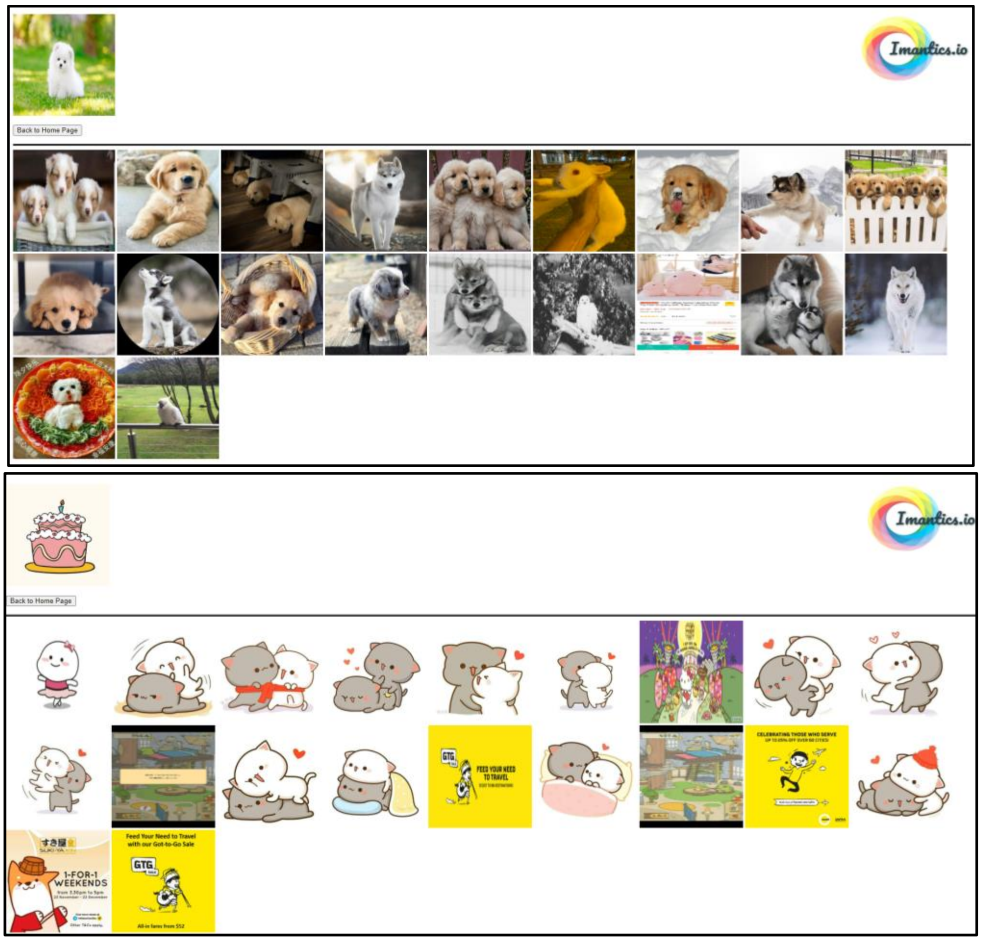
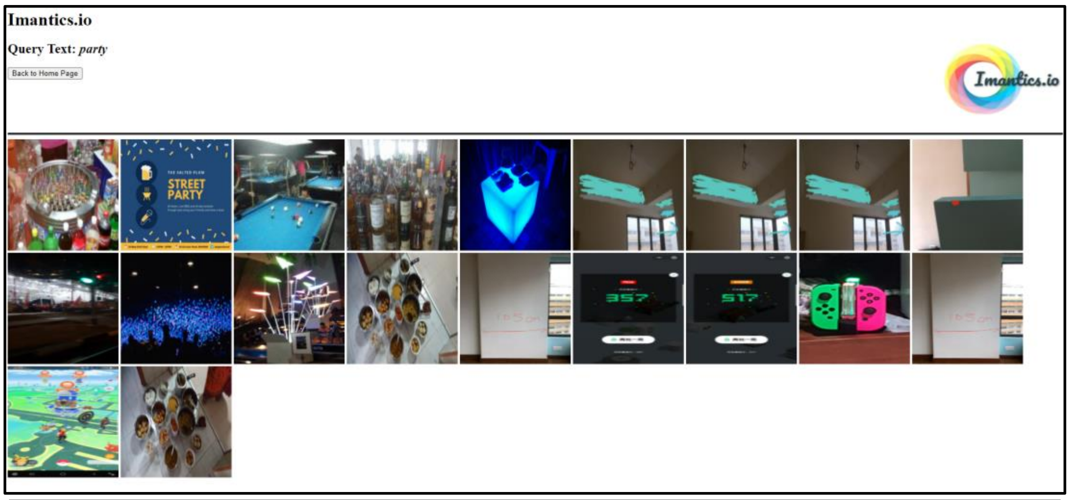


Figure 39: Image Query based on dogs and cartoons

For textual-based retrieval, we utilise *CLIP* to query in the projected shared image-text embedding space to retrieve the top *K-nearest neighbours* through the Euclidean distance measure. *CLIP* leverages on an interesting framework otherwise known as *prompt engineering* where the input model is guided through the textual input on what are the semantics of interest to discover. While still largely trial and error, the arrangements of the text prompts presented to *CLIP* often yield interesting discoveries within the dataset. To follow the input sequence for text prompts to the model, we concatenate a base prompt "This is a photo of a {}" before each input token.



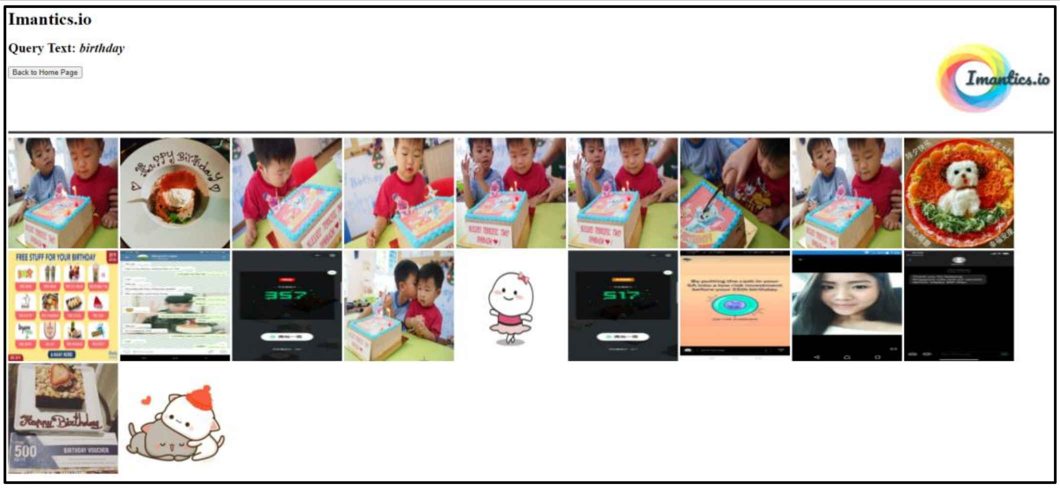


Figure 40: Text Query based results for Party and Birthday respectively

CLIP was able to identify elements related to "birthday" such as the birthday cake and similar images of games and alcohol for "party". It was also interesting to note that the model was able to detect textual content within images as seen in the image containing the words "Street Party", which highlights its ability to recognise both text and visual content to enforce alignment.

While *CLIP* provides a huge baseline for cross-modal retrieval for both images and texts, it is worth noting that such a large pre-trained model comes with its own inherent biases and risks. There have been reports on usage of the model displaying cases of young males being labelled as gangsters or thieves, while others have also reported on the perennial issue of mis-associating black-skin people with apes and gorillas. Although much effort has been put into identifying such inherent biases and tackling it at the onset, the challenge of working with such large-scale pre-trained corpus still remains a hurdle today.

3.4. LIMITATIONS AND IMPROVEMENTS

S/N	LIMITATIONS	IMPROVEMENTS
1.	Results are not extensive due to the limited size of curated dataset (i.e. 2000 mobile gallery image	To further add more images to the dataset to improve the clustering and image retrieval accuracy and

	dataset)	precision.
2.	The system currently works as a web UI page, where the usefulness of the system is limited	To deploy the system in the mobile phone where it can work based on the actual images within the user's phone gallery and user can easily query their gallery to extract the images they want
3.	Currently the models are pretrained on an existing dataset, deployed and utilised as inference with the weights frozen. While there is a certain level of generalisability using some of the *SSL* methods, significant data shifts would adversely impact the eventual model accuracy and efficacy.	One of the interesting framework that can be explored is the idea of continual learning, where increment training with addition of novel unseen data could be performed over the baseline model. Continual learning of online models while maintaining robustness against catastrophic forgetting is another emerging area of research in representation learning. This would be highly applicable for online model servings that require periodic updates to maintain performance.
4.	The framework currently does not allow users to compute clustering on-the-fly as it would take a fair bit of time due to the high computation workload. The clusters are pre-computed beforehand before displaying on the web UI.	To be able to explore incremental cluster discovery would be a bonus (i.e. users do not need to re-cluster the entire dataset with the addition of a handful of images). One possible approach is to utilise soft-labels from the clustering output to train a classifier and perform predictions on novel images that arrive in batches.

Figure 41: Table of limitations & improvements

3.5. FUTURE WORK

The existing architecture consists of the application of various clustering models on the datasets to generate different groupings of the images according to their similarities based on the model parameters. The next phase of this project will be to explore various ensemble techniques to create a hybrid ensemble of these models where the algorithms can create better clustering outputs. These ensemble classifiers may include majority voting, weighted voting, simple averaging, weighted averaging or stacking.

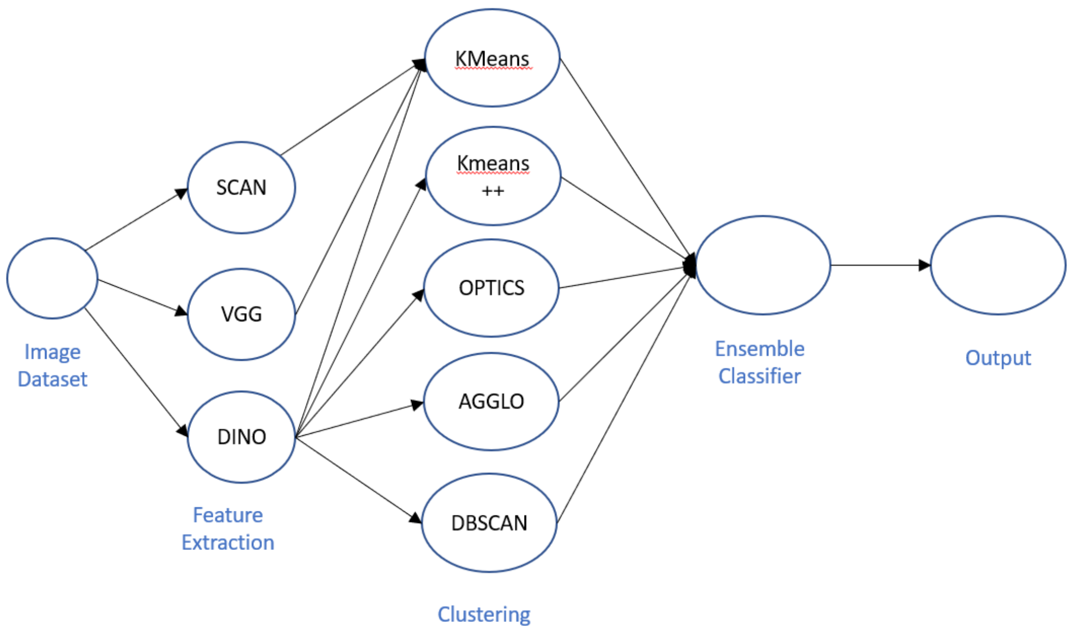


Figure 42: Table of limitations & improvements

Also, the architecture will be more applicable if it is used in a mobile phone. Future work will include incorporating this architecture into the mobile phone as an application for easy image retrieval and image categorization by the users.

4. CONCLUSION

In this project, our team has successfully developed and implemented a semantics image archival system that is able to generalise to a novel unseen dataset through the use of *Self-Supervised Learning*. The strong visual representations are derived with no labels using the instance discrimination pretext task of maximising similarities of heavily augmented pairs of images and minimising dissimilarities of different images through contrastive learning. We explored various pretext tasks such as *MoCo* and *SimCLR* which are *Convolution Neural Network* based, and *DINO*, which is a *Transformer*-based architecture for the feature extraction, followed by a suite of clustering techniques. In addition, we also explored another self-supervised clustering framework (*SCAN*) that fine-tunes and clusters the image dataset via supervision from the nearest neighbours.

The testing and comparison of the different frameworks on both the labelled ImageNet dataset and the custom mobile phone gallery dataset showcases the models' abilities in unsupervised pattern discovery, with *DINO* and agglomerative clustering returning the highest Normalised Mutual Information (*NMI)* and Adjusted Rand Index (*ARI)* scores. Through the visual inspection of the final clusters, we were able to identify the dominant semantic ideas in several the clusters derived on the unlabelled mobile phone gallery dataset. This further affirms the strong feature representations that the model has produced were able to generalise across domains to handle such Out-Of-Domain (*OOD)* scenarios.

With the extension of contrastive learning, the team also explored using a large-scale image-text cross modality retrieval framework (*CLIP*) to perform semantic text and image retrieval through *k-nearest neighbours*. In the experimentation, we were able to observe that *CLIP* was able to extract corresponding visual images with the input text prompts that is akin to zero-shot classification on the custom mobile phone gallery dataset. However, as *CLIP* was originally pre-trained on a huge corpus of image-text pairs (400 million), there exist inherent biases within the model that would be needed to be addressed, such as retrieving images with young males when input text query is “gangster” or “thief”.

Finally, we have developed and deployed the archival framework as an interactive Web-based UI to demonstrate the components explored within the project timeline. The team has definitely gained deeper knowledge into generalising pattern recognition through self-supervision and have understood first-hand the practical application of techniques that were taught and shared during the course of the certificate such as *K-means* clustering and deep learning with *Keras*/*Tensorflow*.

We wish to thank the course coordinator and all the respective lecturers for the amazing learning journey and the wonderful opportunity for this exploration where the knowledge gained will be useful for real-world applications.

5. REFERENCES

- Caron, M., Touvron, H., Misra, I., Jegou, H., Mairal, J., Bojanowski, P., & Joulin, A. (2021). Emerging Properties in Self-Supervised Vision Transformers. *ArXiv*, *abs/2104.14294*.
- Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A Simple Framework for Contrastive Learning of Visual Representations. *ArVix*, *abs/2002.05709*.
- Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ArXiv*, *abs/2010.11929*.
- Gansbeke, W. V., Vandenhende, S., Georgoulis, S., Proesmans, M., & Gool, L. V. (2020). SCAN: Learning to Classify Images without Labels. *ECCV*.
- He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum Contrast for Unsupervised Visual Representation Learning. *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9726-9735.
- Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. *arXiv*, *abs/1503.02531*.
- LeCun, Y. (2021, March 4). *Self-Supervised Learning*. Self-Supervised Learning: The Dark Matter of Intelligence. https://ai.facebook.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/
- Noroozi, M., & Favaro, P. (2016). Unsupervised learning of visual representations by solving jigsaw puzzles. *ECCV*.
- Zhang, R., Isola, P., & Efros, A. (2017). Split-brain autoencoders: Unsupervised learning by cross-channel prediction. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 645-654.
- Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

6. APPENDIX

APPENDIX A: Project Proposal

GRADUATE CERTIFICATE: Pattern Recognition System (PRS)

a) PRACTICE MODULE: Project Proposal

Date of proposal: 15 September 2021
Project Title: Discovering Patterns Within Images
Group ID (As Enrolled in LumiNUS Class Groups): 7 **Group Members (name , Student ID):** Adriel Kuek, A0229985H Ang Jenn Ning, A0229970U Chua Hao Zi, A0229960W
Sponsor/Client: *(Company Name, Address and Contact Name, Email, if any)* *NA*
Background/Aims/Objectives: Since ILSVRC 2012, deep learning has revolutionised computer vision in providing the machine with vision recognition capabilities surpassing human accuracy. However, the core of the issue still remains where image datasets are costly and labour-intensive to curate and annotate. Additionally, the visual concepts typically learned are only as good as the given dataset. They excel well within the domain that they have been trained on but perform poorly when generalised. In this project, the team endeavour to explore pattern recognition in images utilising Unsupervised or Self-Supervised Learning (SSL) methodologies with an aim to learn robust feature representations that can be well generalised and employed on various downstream tasks. In particular, the team will focus on image clustering and image retrieval based on the latent features extracted through representation learning. We compare the results to classical frameworks on the ImageNet-1000 dataset used in ILSVRC 2012 and extend this work to an unlabelled custom dataset that we aim to curate. The dataset comprises random images in an image gallery collection typically found in a person's mobile phone. The main idea is to perform semantic clustering and retrieval for intelligent filtering and categorisation. Finally, the team would seek to deepen our knowledge and understanding in deep learning applications in computer vision and also to explore current State-of-the-Art (SOTA) methodologies in Self-Supervised Image Representation Learning.
Project Descriptions:

- To explore standard image analysis framework for image clustering and image retrieval, starting with data cleaning and preparation, feature extraction, feature selection and cluster analysis (Kmeans, agglomerative etc.).
- To explore self-supervision representation learning frameworks for image feature extraction through a pretext task of instance discrimination using contrastive learning methods and knowledge distillation frameworks.
- To explore various feature selection processes (PCA, expert knowledge etc.) and its effect in downstream results through various ablation studies.
- To develop quality evaluation metrics to assess the performance of the model
- To compare and analyse self-supervised learning methods with classical deep learning feature extraction algorithms.
- To curate a custom mobile phone gallery dataset that is representative of typical everyday users and perform the identified image clustering and image retrieval frameworks on it.
- To develop a simple user interface display showcasing the different techniques utilised in the project.
- To compile findings and learning experiences gleaned from the course of the project work into a final report and presentation material for sharing and submission.

b) Team Formation & Registration

Team Name: Group 7
Project Title (repeated): Imantics.io – Finding Patterns Within Images
System Name (if decided): Imantics.io
Team Member 1 Name: KUEK YONG JIE ADRIEL
Team Member 1 Matriculation Number: A0229985H
Team Member 1 Contact (Mobile/Email): 91888750 / e0687393@u.nus.edu.sg
Team Member 2 Name: CHUA HAO ZI
Team Member 2 Matriculation Number: A0229960W
Team Member 2 Contact (Mobile/Email): 83829287/e0687368@u.nus.edu

Team Member 3 Name: ANG JENN NING
Team Member 3 Matriculation Number: A0229970U
Team Member 3 Contact (Mobile/Email): 84699239 / e0687378@u.nus.edu

APPENDIX B: Installation & User Guide

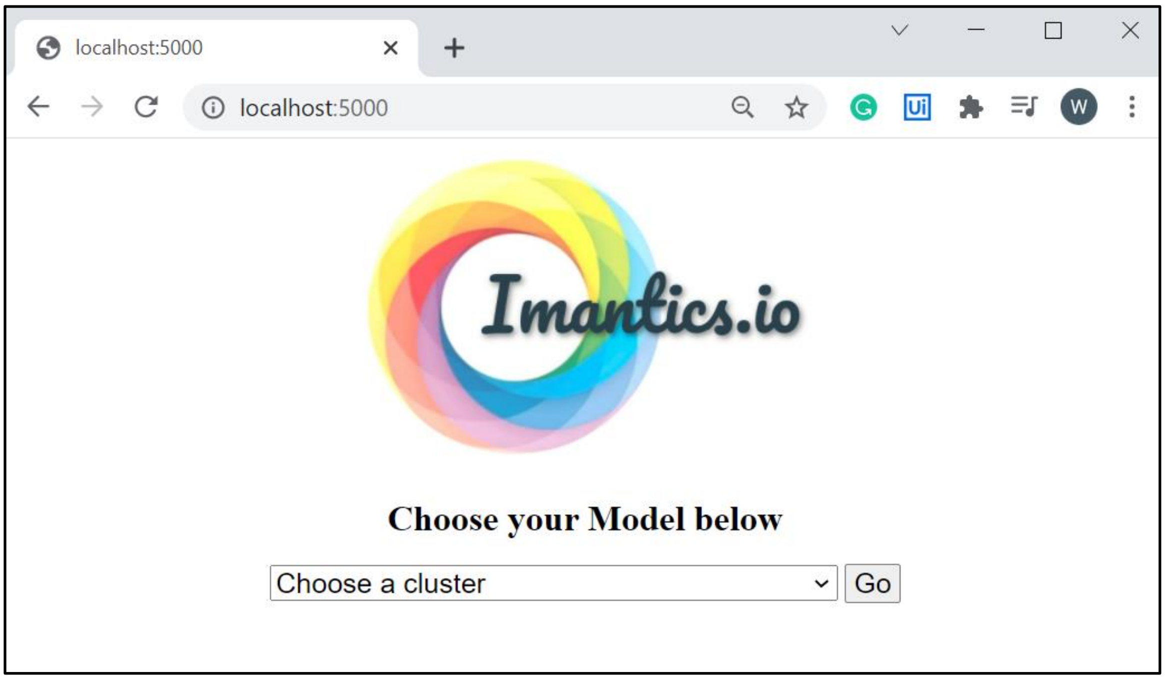
a) Web-UI deployment

To run this architecture, the user is required to have a computer / laptop device that is equipped with CUDA GPU. For users who have CUDA GPU in their local device, they can run the architecture with the steps in (b). For users who do not have CUDA GPU in their local device, they can run the architecture using Google Colab with the steps in (c).

b) Imantics.io (with CUDA GPU in local device)

1. Download the whole repository with the link below:
   https://drive.google.com/drive/folders/1SZRIZVE0if58kbfxXlZ79B4ZIwjjOU6u?usp=sharing
2. Downloading will take some time as it contains large-size models, as well as a 2,000 images worth of dataset
3. Open command terminal and change directory to the project folder.
4. Create virtual environment with the following command
   ```
   python -m venv virtual\
   ```
5. Then run the below command to activate the virtual environment
   ```
   virtual\Scripts\activate
   ```
6. Run the below commands to download the relevant libraries.
   ```
   pip3 install torch==1.10.0+cu102 torchvision==0.11.1+cu102
   torchaudio===0.10.0+cu102 -f
   https://download.pytorch.org/whl/cu102/torch_stable.html

   pip install -r requirements.txt
   ```
7. Then run the below command to run the architecture:
   ```
   python program.py
   ```
8. Wait for the codes to finish running. Below statement will be shown to indicate the codes have finished running:
   ```
   * Running on http://localhost:5000/ (Press CTRL+C to quit)
   ```
9. Go to Google Chrome browser and key in `http://localhost:5000/`
10. This will open up the website browser for users to see the different clustering outputs, as well as to use the image retrieval functions embedded inside the architecture.



Choose your Model below

c) Imantics.io (using Google Colab GPU)

1. Open the “Main.ipynb” in the shared link below. Alternatively, the user can also download a copy of the whole project folder from the link below into their own Google Drive and open up “Main.ipynb” file.
   https://drive.google.com/drive/folders/1SZRIZVE0if58kbfxXlZ79B4ZlwjjOU6u?usp=sharing

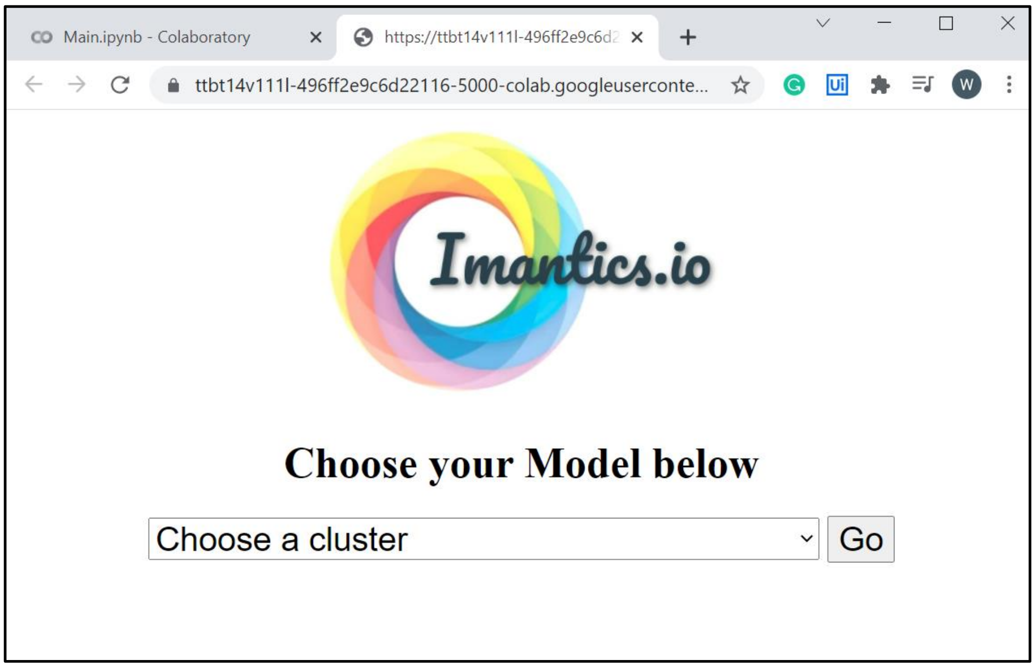
2. Downloading will take some time as it contains large-size models, as well as a 2,000 images worth of dataset.
3. The “Main.ipynb” will be opened automatically with Google Colab. Go to “Edit” >> “Notebook Settings” and ensure the “Hardware Accelerator” is set to <GPU>.
4. Run all cells in the “Main.ipynb”. Wait for the codes to run finish and below statements will be shown which indicates the website is ready:

```
 * Debugger is active!
 * Debugger PIN: 123-123-123
```

5. The output for the cell below will show the URL path for the website. Click on the weblink to open up the website.

```
from google.colab.output import eval_js
urlhost = eval_js("google.colab.kernel.proxyPort(5000)")
```

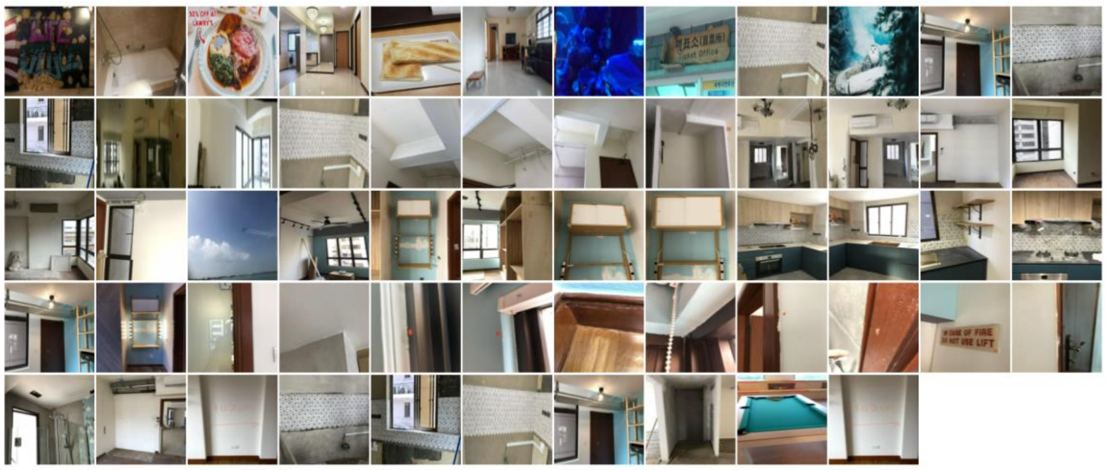
6. This will open up the website browser for users to see the different clustering outputs, as well as to use the image retrieval functions embedded inside the architecture.


Main.ipynb - Colaboratory
https://ttbt14v111l-496ff2e9c6d2
ttbt14v111l-496ff2e9c6d22116-5000-colab.googleuserconte...
Imantics.io
Choose your Model below
Choose a cluster
Go

Note: Using Google Colab to run the website architecture will not be as fast and smooth as compared to running on local devices. Google colab serves as an alternative to users who do not have access to CUDA GPU. Browsing from webpage to webpage using Google Colab may have a delay time of around 1 to 2minutes.

APPENDIX C: Images of Clustering Outputs

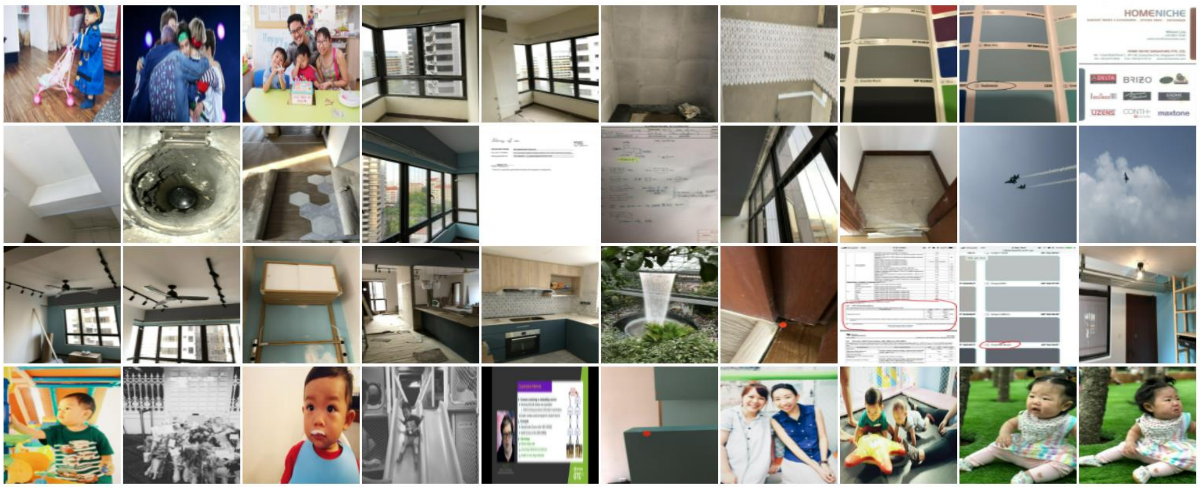
Cluster output from DINO model with KMeans Clustering of 250 cluster size



Cluster output from SCAN model with KMeans Clustering of 200 cluster size

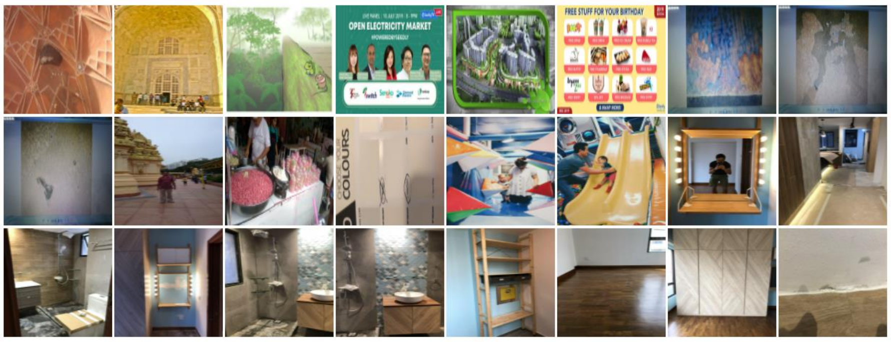
UBS
Malindo air
Google
Booking.com
Singtel
5 DAYS SALE!
QATAR AIRWAYS
AIR NEW ZEALAND
PLANNING TO START A STARTUP?
Electrolux
Expedia
中国南方航空 CHINA SOUTHERN AIRLINES
Jetstar
CATHAY PACIFIC
I HAVE $10K TO INVEST OVER 10 YEARS. SHOULD I PUT MY MONEY IN REITS, BONDS OR INSURANCE?
QATAR AIRWAYS
POWERBEATS PRO
INVESTMENT 101: BASICS OF IPO
GET RICH OR DIE TRYIN'
Grab
cebu pacific
Emirates
MINISTRY OF MANPOWER
Australian International School
sea
TEMASEK HOLDINGS

Cluster output from VGG model with KMeans Clustering of 200 cluster size



Cluster output from DINO model with DBSCAN Clustering of Epsilon 8 and Min Samples 3

Cluster output from DINO model with Optics Clustering of Min Samples 5



Cluster output from DINO model with Agglomerative Clustering of Euclidean Distance 20

APPENDIX D: Mapped System Functionalities

Mapped System Functionalities against knowledge, techniques, and skills of modular courses

Modular Courses	System Functionalities / Technique Applied
Problem Solving Using Pattern Recognition (PSUPR)	● **Cluster Analysis** K-means, DBSCAN, Agglomerative Hierarchical. ● **Retrieval** k-Nearest Neighbours
Pattern Recognition and Machine Learning Systems (PRMLS)	● **Deep Learning** Image Feature Extraction (DINO, MoCo, SimCLR etc.)
Intelligent Sensing and Sense-Making (ISSM)	● **Multi-modal Processing** CLIP ● **Data Augmentation** Pretext Tasks, Instance Discrimination ● **Sense-making pipeline** Imantics.io discovery pipeline