On-Device Image Labelling Photo Management System Using Flutter and ML Kit

Tan Chi Wee(✉) and Ken Ng Chen Kee

Tunku Abdul Rahman University College, 53300 Kuala Lumpur, Malaysia
chiwee@tarc.edu.my

Abstract. Automatic image annotation is the process by which the system automatically assigns relevant labels (metadata) to a digital image. This type of computer vision technique is mainly used in image retrieval systems to organize all the data and seek the interest of images from databases. This technique is also considered as a type of multi-class image classification. Regarding the past related work that had been done by the researchers, annotating digital images have also been used for the Academic Health Care Environment to solve the difficulty of business and graphic arts commercial-off-the-shelf (COTS) software in multi-context authoring and interactive teaching environments. As many pre-trained machine models have been created for the past few years, the requirement for existing models still needs a large set of data to be imported, and the usage of CPU hours is tremendously expensive. Google cloud API can outperform existing models in terms of computational complexity in obtaining image labels. The ML Kit firebase associated with Google Cloud Vision API is idealistically suited in this application, which can be useful in returning a set of labels that comes with a score that indicates confidence the ML model has in its relevance. With all of these labels, assembling all images on related labels is no longer a troublesome issue, and it can be quickly searched by querying on the back-end part.

Keywords: Computer vision · Object recognition · Artificial intelligence · Flutter · Machine learning

1 Introduction

The fundamental basis of image annotation technology is multi-class image classification. Many researchers try to make use of this technique into the mobile application because the trends of using mobile applications are proliferating. Mobile applications are portable to be used everywhere else, so, understandably, making use of mobile applications can effectively increase the living standard. There is a related work which uses the multi-classification approach to detect the malware which was intentionally installed into the local device to steal the user's sensitive information. There are numerous advancements in machine learning that contribute to cybercrime, especially malware. Multi-class classification is useful for assigning multiple labels in the application in order to have classes such as (i.e., spyware, rootkit, ransomware, etc.). This is to contain and cluster

Z. Lv and H. Song (Eds.): INTETAIN 2021, LNICST 429, pp. 208–222, 2022.
https://doi.org/10.1007/978-3-030-99188-3_13

all the malware families and their samples [1]. Moreover, there is an investigation on evaluating whether the performance of Google Cloud Vision API for image labelling is outperformed existing Machine Learning (ML) models in describing labelling. There are three methods proposed to annotate the images. The first method is directly processed by Google cloud vision. This method lacks considerations on the synonyms, so the accuracy is not on the ideal baseline. For the second method, validation is done to indicate whether the labels generated by Google Cloud Vision API were consistent with the instance label, and make a comparison using WordNet. The third method is released for a reason to solve the problem caused by the category labels named by a dataset. In this paper, we try to bring two gulfs, which is multi-class image classification into a mobile app with the help of Flutter and Google Cloud Vision without the needs of internet connection. In other words, the image labelling and recognition process are able to work offline independently. The rest of this paper is organized as follows: Section 2: literature review; Sect. 3: the proposed method and architecture; Sect. 4: all the experimental results and discussion are shown here; and finally, the conclusion is described in Sect. 5.

2 Literature Review and Related Work

2.1 Google Cloud Vision

Google Cloud Vision is an image recognition technology that allows a user to remotely process the content of an image and to retrieve its main features. By using a specialized REST API, called Google Cloud Vision API, developers exploit such a technology within their own applications. Currently, this tool is in limited preview and its services are accessible for trusted tester users only. The drawback of this service is that it requires network connection and cannot work standalone.

2.2 Flutter

Flutter is a framework developed by the Google team, which is built to focus on the development of both android and iOS. By using this framework has relatively some advantages in terms of optimization, Flutter uses widgets and dart programming language to create mobile or web applications from a single code base with high performance, and because the code is simplified, so the maintainability of the software is also high.

Every component in flutter is called a widget. A widget is formed by combining several other widgets, for example, Container widget consists of Align widget, Padding widget as well as `DecoratedBox` widget, etc. Creating a user interface in flutter is as similar to a person building a Lego toy by combining every piece of bricks. When each child widget inherits from its parent widget, it forms a hierarchical tree. If there is a need to respond to events, developers can simply use the framework to replace the necessary widget by looking at the constructed tree without deep diving into the codes.

2.3 Widget Lifecycle Events

In Flutter, there consists of two types of widgets to be used in the application, Stateless and Stateful widget. Stateful widget is used when there is the need for interaction from the user to change the current state of the widget, and the state of the stateless widget is unchangeable. Both of these widgets have a `build()` method which serves the purpose of building the widget itself. In this `build()` method, the parameter called `BuildContext` is an instantiation of the widget to indicate the location of this widget in the tree.

Stateless Widget. A stateless widget does not change dynamically, even the configuration has been modified. For example, the screen displays an image with a predefined name (usually the title does not change frequently). This type of widget has only one class; there consists of three ways of calling the stateless widget to be built. One method is that when the widget is first created, the parent of the widget changes and `InheritedWidget` has affected the widget.

Stateful Widget. The state of the `StatefulWidget` will be changed dynamically based on its configuration. For example, if the screen consists of a button which is to calculate a sum of an equation, in this case, the state(value) will be changed when the user clicks on the button. This type of widget is mutable, which means it can vary based on user interaction. Two classes are stated in this type of widget, State and Stateful widget. `StatefulWidget` class is rebuilt whenever the widget's configuration is different from the previous one. However, state class is still the same unless the `StatefulWidget` is removed from the tree, then it will trigger the State class to be instantiated. New changes value will not repaint on to the screen without calling the `setState()` method. This is to notify the flutter's framework to listen to the changes of the widget and recall back to `build()` method to rebuild a new UI screen.

2.4 Widget Lifecycle Events

In [2], Abdalbasit Mohammed Qadir & Peter Cooper proposed a cargo tracking system that tracks the location of the cargo by integrating the GPS provided by Google and use Flutter framework as its codebase. Their objective is to develop a system which is applicable for cross-platform as well as Web applications. An example of a login screen is provided below (Fig. 1):

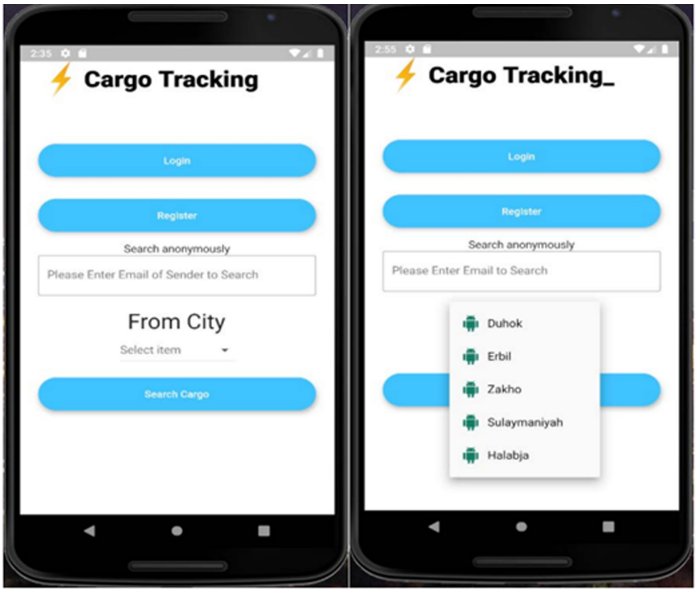

Fig. 1. An example of flutter application

They managed to build the login screen using only a single code base supported by the IOS and Android platforms. They indicate they can write a manageable code and simplify the time-consuming process for working on both platforms.

2.5 Object Detection and Image Labelling

Image annotation is a process of creating a set of data for those computer vision models. This helps the machines to automatically assign metadata into a digital image using relevant keywords and labels. This kind of technique commonly can be found in the image retrieval system, which can easily facilitate the machine to locate particular images from a database.

2.6 Comparison of Image Recognition Tool in the Market

Siham Bacha & Nadjia Benblidia [3] proposed a scheme by combining both content-based and context-based image annotation for automatic image annotation on mobile phones. The combination of these two approaches is to provide a standard feature understanding about the image in different aspects and to improve the description of the image on the high-level concepts. The significant drawbacks of this concept are that the combination brings heavy computation to achieve the final annotation for the AIA system [4].

Zhen Li et al. [5] proposed an image annotation system for media sharing that optimizes the content, context analysis and their integration to label images extracted from the client's mobile (Table 1).

Table 1. Image recognition tools in comparison.

Name	Main Features	Advantages
Talkwalker	• Analyse both text and images, in conjunction • The largest brand logo database - 30,000 logos, objects, scenes • Access to Twitter firehose, 10+ social networks & 150 websites	• Capitalize on user-generated content to boost brand awareness • Protect your brand against trademark abuse • Prove the true Return of investment of your sponsorship
Google Cloud	• Assign labels to images and categorize • Detect objects – where, quantity, facial attributes, landmarks, logos, text OCR • printed and handwritten, explicit content	• Reduce purchase friction with user-friendly mobile UX • Users can upload photo of an item & find similar to purchase
amazon Rekognition	• Easy to use API that doesn't require machine learning expertise • Analyse video and images - objects, people, text, scenes, activities, inappropriate content	• Consistent response times regardless of volume of requests • Facial analysis reveals age range, and sentiment • great for retail
clarifai	• Retrieve images that are similar to query image • Humans used to increase accuracy – for content moderation • Geolocation filter with pre-set parameters – latitude, longitude	• Identify suspicious behaviour happening on your property • Product discoverability based on visual features

(*continued*)

Table 1. (*continued*)

		– product recommendation
LOGOGRAB	• Find logos in infinite number of social media images, videos, GIFs • Extensive library of global and regional brands and marks	• Protect your brand with counterfeit detection • Deliver better customer service, real-time product demos, customer onboarding
IBM Watson	• Train with your own images to achieve unique, powerful results • Build custom classifiers by uploading training images for an application that can detect and tag images	• Computer vision service can learn any new object or person, & attribute – e.g., identifying car type and damage to estimate repair costs • Analyse visual content to optimize processes & ROI, decrease operational costs
imagga	• Auto tagging, auto categorizing, colour extraction, custom training, face recognition available to data sensitive businesses	• Empower product discoverability • Automated adult image content moderation • Analyse users' social media image content
CloudSight™	• Users can expect an average response time of less than 250ms • Recognize objects by scanning phone around room • Filter and categorize images, monitor for inappropriate content	• Make things discoverable on your ecommerce site or through augmented product and image details such as

(*continued*)

Table 1. (*continued*)

		brand, style, type, and more • Let users sell items on your platform by uploading a picture
EyeEm	• AI used to find the best images to license to brands & agencies • Photos automatically tagged and captioned with objects and keywords to make searchable library	• Maximize engagement by determining the discoverability of each photo • Train the tool to maintain a consistent visual language on a global scale • Monitor and measure campaign metrics in real-time to improve content reach

In the content-based analysis, all the previously suggested tags from the media are corporate with the locations using the Gaussian Mixture Model (GMM). The intent of the user community facilitated the tagging process of the annotation system to suggest tag in future tagging. In order to reduce the computational cost for the content analysis process, a bag-of-words (BoW)-based approach with spatial pyramid matching [6] based on dense SIFT is adopted. According to them, by combining the content and context-based information, it brings about 16% and 15% performance improvement to the accuracy and recall rate for the tagging system.

L. Yu, J. Xie and S. Chen [7] proposed a scheme to solve the over-segmentized problem obtained from the segment-based conditional random field (CRF). It makes no sense in terms of statistical information if the extracted segment is too small. In other words, significant segments may not be consistent with the object boundaries, usually would bring error-prone to the process of labelling. The authors of this paper proposed a single-layered segment-based CRF, instead of a multi-layered based CRF, so multi-scale features of pixels, segments and regions can be combined. A modified version of the TextonBoost algorithm is used to determine which object class belongs to which pixel. This brings slightly better results to the recognition accuracy. Although high accuracy is obtained, the authors proposed the segment-based CRF to solve the problem of inaccurate

boundaries caused by the pixel-based CRF. Another problem will also arise when pixels and segments are integrated further. In order to alleviate this problem, a region-based CRF is proposed to the model co-occurrence. The authors take this co-occurrence CRF as a post-processing method which results in the process being very fast and able to overcome some co-occurrence constraints violation errors.

3 Proposed Method

In this proposed method, the architecture in Flutter is divided into three different layers and an API provider, it is relatively significant to understand how the data is processed in the entire system and associated with external plugins (Fig. 2).

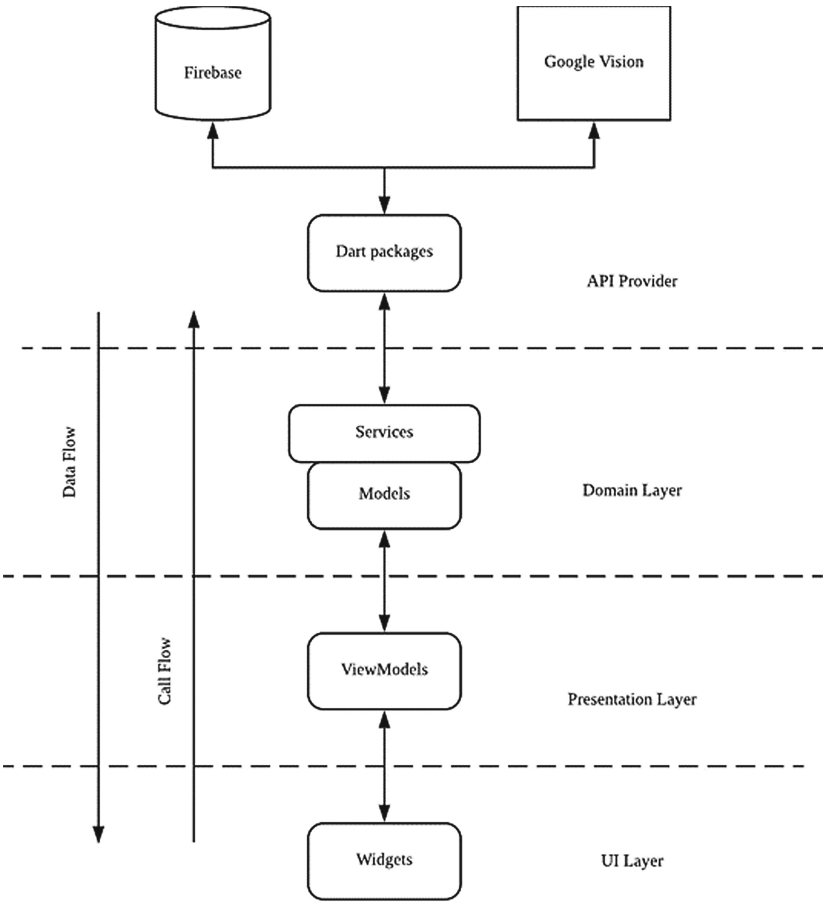

Fig. 2. Proposed architecture for an on-device photo tagging system

The architecture in Flutter is divided into three different layers and an API provider, it is relatively significant to understand how the data is processed in the entire system

and associated with external plugins. The widgets in the user interface (UI) layer are a user control that inherits from the `DiagnosticableTree`. It plays a vital role in how the system layout is supposed to look like. It contains no logic other than displaying data on the screen for the user to view the result. In the presentation layer, the view model responsible for preparing and managing the data for an Activity or a Fragment. The purpose of this view model is to retrieve all the processed or unprocessed data from each of the defined models and combine them into an informative view and then pass it to the UI layer to perform the following rendering process. For the domain layer, models represent the shape of the data. A class in dart is used to describe a model. Particular services in the context class will be provided in order to retrieve stored data from the database. The last stage of the architecture diagram is the application programming interface (API), provider. Each plugin has its dart package. Once the dependencies are implemented into the project, developers are free to use these libraries to call the services provided by firebase or google vision.

The widgets in the user interface (UI) layer are a user control that inherits from the `DiagnosticableTree`. It plays a vital role in how the system layout is supposed to look like. It contains no logic other than displaying data on the screen for the user to view the result. In the presentation layer, the view model responsible for preparing and managing the data for an Activity or a Fragment. The purpose of this view model is to retrieve all the processed or unprocessed data from each of the defined models and combine them into an informative view and then pass it to the UI layer to perform the following rendering process. For the domain layer, models represent the shape of the data. A class in dart is used to describe a model. Particular services in the context class will be provided in order to retrieve stored data from the database. The last stage of the architecture diagram is the application programming interface (API), provider. Each plugin has its dart package. Once the dependencies are implemented into the project, developers are free to use these libraries to call the services provided by Firebase or google vision.

4 Result and Discussion

In this proposed, every image can be categorized with different labels depending on the extracted context information from the image. This can be very useful and convenient for a user to search for the wanted image based on the generated labels. Any inconsistent generated tags can also be modified, which is also a robust feature to enhance the flexibility and correctness of these tags. Processed image will not be re-process again even the user restarts the application. This feature can ensure the time taken for the heavy process in the classifying phase. There are two main key capabilities for this proposed system:

– Production-ready for common use cases: ML Kit comes with a set of ready-to-use APIs for common mobile use cases: recognizing text, detecting faces, identifying landmarks, scanning barcodes, labelling images, and identifying the language of text. Simply pass in data to the ML Kit library, and it gives the information that user need.

– On-device or in the cloud: ML Kit's selection of APIs run on-device or in the cloud. Our on-device APIs can process the data quickly and work even when there's no network connection. Our cloud-based APIs, on the other hand, leverage the power of Google Cloud's machine learning technology to give the user an even higher level of accuracy.

4.1 List of Categories

A total of 400+ categories can be recognized, each of the image can consist of either zero or more than one label(s) (Table 2 and Figs. 3 and 4).

Table 2. Categories that can be used to classify image, i.e., image label

Object and scene recognized				
Team	Monochrome	Speedboat	Penguin	Carnival
Bonfire	Chair	Trunk	Shikoku	Snowboarding
Comics	Poster	Coffee	Palace	Waterskiing
Himalayan	Bar	Soccer	Doily	Wall
Iceberg	Shipwreck	Ragdoll	Polo	Rocket
Bento	Pier	Food	Paper	Countertop
Sink	Community	Standing	Pop music	Beach
Toy	Caving	Fiction	Skiff	Rainbow
Statue	Cave	Fruit	Pizza	Branch
Cheeseburger	Tie	Pho	Pet	Moustache
Tractor	Cabinetry	Sparkler	Quilting	Garden
Sled	Underwater	Presentation	Cage	Gown
Aquarium	Clown	Swing	Skateboard	Field
Circus	Nightclub	Cairn terrier	Surfing	Dog
Sitting	Cycling	Forest	Rugby	Superhero
Beard	Comet	Flag	Lipstick	Flower
Bridge	Mortarboard	Frigate	River	Placemat
Tights	Track	Foot	Race	Subwoofer
Bird	Christmas	Jacket	Rowing	Cathedral
Rafting	Church	Pillow	Road	Building
Park	Clock	Bathing	Running	Airplane
Factory	Dude	Glacier	Room	Fur
Graduation	Cattle	Gymnastics	Roof	Bull
Porcelain	Jungle	Ear	Star	Bench

(*continued*)

Table 2. (*continued*)

Object and scene recognized				
Team	Monochrome	Speedboat	Penguin	Carnival
Twig	Desk	Flora	Sports	Temple
Petal	Curling	Shell	Shoe	Butterfly
Cushion	Cuisine	Grandparent	Tubing	Model
Sunglasses	Cat	Ruins	Space	Marathon
Infrastructure	Juice	Eyelash	Sleep	Needlework
Ferris wheel	Couscous	Bunk bed	Skin	Kitchen
Pomacentridae	Screenshot	Balance	Swimming	Castle
Wetsuit	Crew	Backpacking	School	Aurora
Shetland sheepdog	Skyline	Horse	Sushi	Larva
Brig	Stuffed toy	Glitter	Loveseat	Racing
Watercolor paint	Cookie	Saucer	Superman	Airliner
Competition	Tile	Hair	Cool	Dam
Cliff	Hanukkah	Miniature	Skiing	Textile
Badminton	Crochet	Crowd	Submarine	Groom
Safari	Skateboarder	Curtain	Song	Fun
Bicycle	Clipper	Icon	Class	Steaming
Stadium	Nail	Pixie-bob	Skyscraper	Vegetable
Boat	Cola	Herd	Volcano	Unicycle
Smile	Cutlery	Insect	Television	Jeans
Surfboard	Menu	Ice	Rein	Flowerpot
Fast food	Sari	Bangle	Tattoo	Drawer
Sunset	Plush	Flap	Train	Cake
Hot dog	Pocket	Jewellery	Handrail	Armrest
Shorts	Neon	Knitting	Cup	Aviation
Bus	Icicle	Centrepiece	Vehicle	Fog
Bullfighting	Pasteles	Outerwear	Handbag	Fireworks
Sky	Chain	Love	Lampshade	Farm
Gerbil	Dance	Muscle	Event	Seal
Rock	Dune	Motorcycle	Wine	Shelf
Interaction	Santa claus	Money	Wing	Bangs
Dress	Thanksgiving	Mosque	Wheel	Lightning
Toe	Tuxedo	Tableware	Wakeboarding	Van
Bear	Mouth	Ballroom	Web page	Sphynx

(*continued*)

Table 2. (*continued*)

Object and scene recognized				
Team	Monochrome	Speedboat	Penguin	Carnival
Eating	Desert	Kayak	Ranch	Tire
Tower	Dinosaur	Leisure	Fishing	Denim
Brick	Mufti	Receipt	Heart	Prairie
Junk	Fire	Lake	Cotton	Snorkeling
Person	Bedroom	Lighthouse	Cappuccino	Umbrella
Windsurfing	Goggles	Bridle	Bread	Asphalt
Swimwear	Dragon	Leather	Sand	Sailboat
Roller	Couch	Horn	Basset hound	Bride
Camping	Sledding	Strap	Pattern	Swamp
Playground	Cap	Lego	Supper	Pie
Bathroom	Whiteboard	Scuba diving	Veil	Bag
Laugh	Hat	Leggings	Waterfall	Joker
Balloon	Gelato	Pool	Lunch	News
Concert	Cavalier	Musical instrument	Odometer	Newspaper
Prom	Beanie	Musical	Baby	Piano
Construction	Jersey	Metal	Glasses	Plant
Product	Scarf	Moon	Car	Passport
Reef	Vacation	Blazer	Aircraft	Waterfowl
Picnic	Pitch	Marriage	Hand	Flesh
Wreath	Blackboard	Mobile phone	Rodeo	Net
Wheelbarrow	Deejay	Militia	Canyon	Icing
Boxer	Monument	Tablecloth	Meal	Dalmatian
Necklace	Bumper	Party	Softball	Casino
Bracelet	Longboard	Nebula	Alcohol	Windshield
Cookware and bakeware	Computer	Stairs		

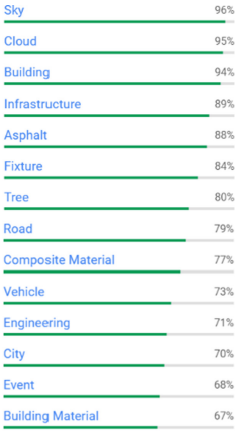

Fig. 3. Sample photo 1, TAR UC main gate with its' image label

Fig. 4. Sample photo 2, fruit in a tableware with its' image label

4.2 Strength and Uniqueness

The classifier supports multiple categories' detection to a single image. In this context, the incorrect label(s) can be modified in order to correct the wrongly labelled categories for the particular image. Generated labels are stored in the client's local disk to ensure the next visit to the application will not re-classify again.

The system is able to generate tags to every image during the initialization stage. All the processed images can also be customized if they were wrongly labelled. And for the last objective stated, the application does not need to internet connection in order to perform classification, so this is also considered a success in achieving the objective written at the beginning. However, the application system still has a lot of room to improvement and the image labelling model have a lot of potential to be further enhanced.

As the number of images stored in a user's local device is increasing tremendously, people are interested to know what the image is all about. Therefore, a smart gallery application associated with the image labelling service is proposed in this project. Label all images with tags during the initializing stage is necessary, so that every image can be categorized into different sections with meaningful context. In order to make it more marketable, application programming interface such as Google Map, Social Share and Google search based on the selected tag are integrated with the system as well. In addition, a part of the system is also highly marketable which is the support of custom tags. The user of the application has the flexibility to edit those tags if those images are wrongly labelled. The first operation they can do is to create a tag and transfer to it. If the selected tag has already existed in the following labels, then it will not recreate another same label to the user. Another alternate operation is that they can remove the image from the labels, so by this way, all those inaccurate labelled images can be eliminated from the particular tag.

Dart is the backend tool attached with the flutter framework [8]. As there is a lack of plugins for monitoring changes in media store (It is a media provider that provides all the collection of images, audio, video from the storage devices). In this case, we need an approach to refresh all the new images captured from the external sources. Along with this project, a plugin for Flutter called `flutter_restart` also being proposed and developed with aims of refreshing all the images captured from external sources. This plugin has made for public can be downloaded from https://pub.dev/packages/flutter_restart [9].

5 Conclusion

The core focus of the project is to create labels for image and photo with and on-device image labelling photo management system using Flutter and machine learning concept which make possible for users to search for related images based on the label on the image. Automatic generation of relevant textual annotations for images can play a crucial role in the performance of image search and image categorization. The effectiveness of browsing an image using annotations can be exclusively faster and even save much time if the system consists of an incredibly large number of images. Traditionally, an image can only exist in one folder, but it could have a decent amount of tags in one image. Although duplication of the same image can be done for both folders, however, this will cause memory wastage problems for the client's phone. By using image annotation technique, saving different key pairs for an image is way more efficient in terms of storage size.

References

1. Alswaina, F., Elleithy, K.: Android malware permission-based multi-class classification using extremely randomized trees. IEEE Access **6**, 76217–76227 (2018). https://doi.org/10.1109/ACCESS.2018.2883975
2. Qadir, A.M., Cooper, P.: GPS-based mobile cross-platform Cargo tracking system with web-based application. In: 2020 8th International Symposium on Digital Forensics and Security (ISDFS), June 2020, pp. 1–7 (2020). https://doi.org/10.1109/ISDFS49300.2020.9116336

3. Bacha, S., Benblidia, N.: Combining context and content for automatic image annotation on mobile phones. In: 2013 International Conference on IT Convergence and Security (ICITCS), December 2013, pp. 1–4 (2013). https://doi.org/10.1109/ICITCS.2013.6717813
4. Zhang, D., Islam, M.M., Lu, G.: A review on automatic image annotation techniques. Pattern Recognit. (2012). https://doi.org/10.1016/j.patcog.2011.05.013
5. Li, Z., Yap, K.-H., Tan, K.-W.: Context-aware mobile image annotation for media search and sharing. Signal Process. Image Commun. **28**(6), 624–641 (2013). https://doi.org/10.1016/j.image.2013.01.003
6. Li, Z.W., Zhang, J., Liu, X., Zhuo, L.: Creating the bag-of-words with spatial context information for image retrieval. Appl. Mech. Mater. **556–562**, 4788–4791 (2014). https://doi.org/10.4028/www.scientific.net/AMM.556-562.4788
7. Yu, L., Xie, J., Chen, S.: Conditional random field-based image labelling combining features of pixels, segments and regions. IET Comput. Vis. **6**(5), 459–467 (2012). https://doi.org/10.1049/iet-cvi.2011.0203
8. Ng, K.: Introducing flutter and getting started. In: Beginning Flutter®, pp. 1–23. Wiley (2019)
9. Tan, C.K., Chi, W., Ng, K.: flutter_restart | Flutter Package (2020). https://pub.dev/packages/flutter_restart