Implementation of a System for Describing Images using AI Models

Introduction – Statement of Problem

Artificial Intelligence or AI was once limited to science fiction stories, where writers would write about its potential benefits as utopias or its possible dangers as dystopias. Since decades ago, these stories have introduced different effects of creating human-like intelligence and as many of them were focused on the rebellions by robots, a huge range of people become resistant to accepting the positive effects of this wide technology. However, scientists who believed in the benefits of this science tried hard to fight with this attitude and in conjunction with the rapid growth of other technologies, people are using it consciously or even unconsciously in many aspects of their lives.

Actually, the improvement of computational techniques as well as unlimited access to the Internet and the enhancement of Hardware and Software platforms have made it possible for many conventional tasks to be involved with AI, and examples of this are seen in a wide range from which the most accessible ones are probably the ones that are deployed on mobile phones, ranging from face recognition for security, object detection in cameras, spell checkers, intelligent assistants and many more. These facilities have changed the way people use technology so smoothly that many of them are still unaware of the real power of AI.

With this said, allowing ordinary users to use AI models in their basic forms and gradually making them familiar with scientific and daily applications of these models can be an enjoyable and also educational experience and this is the main concept of this research and project. The definition of different sections of this mobile phone application and the purposes of each of them are explained in detail in advance, and most of the work is based on Computer Vision models, as they can make it possible for users to see the world from the view of a computer and this makes them more understandable.

This application is programmed to allow the users experience the use of three Computer Vision models in four levels all of which begin with capturing a photo or choosing an image from the gallery. The first level is Object Recognition and it automatically makes the user aware of the objects that are seen on his desired image. The next level is Object Detection, and it makes the user aware of the objects that are seen on his desired image, and also draws a box around each of them to specify their exact location. The third level is about making a descriptive caption for the image based on the objects of the image and their logical relations and the last level is for recommending a music playlist to the user considering the general concept of the image and its objects.

This sequence of levels represents a range of possible applications of AI models in this area and gives the user a sense of what happens in more complicated systems by introducing more basic elements and features. In addition to this, four other usages are employed in this application.

First of all, users will have the right to give or not to give permission to the system to save their images and the produced results in a database. In case of permission, the users can participate in the process of making datasets by providing pictures that are labeled by already tested models. This is done indirectly and with no extra effort, so the users can simply enjoy using the application and at the same time participate in a scientific process.

The second profit that the approval of this permission offers is that an album is made by their used images and the produced results, and this can be used as a technique for saving memories and moments with the participation of AI.

The third and most important purpose of this application is that the Image Captioning section can be used by visually impaired people, with the help of an intelligent voice assistant that is accessible on mobile phones, to provide them with a description of a photo from their surroundings or a selected image, hopefully facilitating their independence process. By using more complicated models, or models that are trained for specific detections, this process can be enhanced and the results will more acceptable.

The final characteristic of this application is that it is implemented to function as a platform for Computer Vision models, so adding models that are not already installed, removing them, or changing the way they produce results is simply possible for programmers. Therefore, this application can be seen as a generalizable system with minimum limits.

In the next sections, the possible solutions for deploying models, as the core of this application, are discussed and the best solution is selected. Next, different dimensions of the implemented system, used tools and technologies, and the reasons for using them are comprehensively described. Finally, results and possible future works are introduced.

Possible Solutions

The basic element of this project is about using AI models on mobile phones. Considering the limitations of the main platform, choosing the adequate approach depends on different challenges and should be analyzed carefully. For this reason, first an introduction to the used models is provided. Then, two main solutions are introduced and compared from different aspects, and finally the final decision is made.

Three main models are selected for this project and used at different levels. For keeping the results coherent and similar, all of the selected models are trained with a unit dataset, the COCO dataset[1].

[1] https://cocodataset.org/#home

The first model used in the first, second, and last level, namely Object Recognition, Object Detection, and music playlist recommender is an Object Detection saved model from TensorFlow Detection Model Zoo[2]. The selected model uses a single-shot detection model while other models use R-CNN and Faster R-CNN models, which means instead of having two steps for first finding the regions with possible objects and then locating them, all objects are located in one step, so they provide accuracy and speed at the same time.

The second model, is an Image Captioning model based on the "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention"[3] paper's method[4] from Kelvin Xu and others. In this method, an attention-based model is introduced with inspirations from Machine Translation and Object Detection methods and in combination with encoder-decoder methods.

The third and last model is a model implemented by a previous graduate student at the Shahid Beheshti University, which is basically similar to the previous models and combines two models for Object Detection and Caption Generation. The only reason for using this model is to show the simple generalization of the application.

Using AI models on mobile phones can be done in two main methods. In this project, our focus is on Android mobile phones, but possible solutions are also accessible for mobile phones with other operating systems.

The two methods are discriminated in the inference process, where an already trained will be used to perform a specific task. This inference process can be run on the device itself, or on a cloud service that is accessed remotely using the internet.

For using the model on a device, different tools including ML Kit[5] by Google and TensorFlow Lite[6] are accessible, and for using the model on a server, server-client architectures can be used. In advance, both methods will be compared in terms of five issues[7].

Latency is the first issue and from a general point of view, it is the time delay between the cause and the effect of an action. When the model is on server, latency will increase as a result of asynchronous communication between the device and the server, and available bandwidth can also affect latency. But, in cases where the model is on the device, latency will decrease and this enhances real time experiences.

[2] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md
[3] https://arxiv.org/pdf/1502.03044.pdf
[4] https://github.com/deeprnn/image_captioning
[5] https://developers.google.com/ml-kit/
[6] https://www.tensorflow.org/lite
[7] https://developer.android.com/ml

Resources are the next effective factor. By placing the model on the server, better and unlimited access to Hardware and Software resources is provided. However, when the model is on a device like a mobile phone, its resources including power and storage can make limits for performance and possibility of usage.

Another factor is the requirement of a network connection. While using a model which is placed on a server needs constant access to the network, by putting the model on the device it is possible to use them with no or poor access to network infrastructures.

Cost is another factor, which is seen in server-based models in terms of bandwidth for transferring data, and computing charge and server cost for developers, while it is seen in terms of battery usage and model download time for users in on device methods.

The last factor is privacy, as the data may leave the device to reach the server and additional precautions will be necessary in this case, but when the model is placed on the device this concern disappears as the data never leaves the device.

Considering all the mentioned factors, it can be concluded that in terms of latency, needing network connection, and privacy, keeping the model on the phone can be a better solution. However, in this application, plenty of models are already needed to be used and as it is being introduced as a generalizable platform, more models may be also added. As a result, the cost and resources factors play a significant role in the decision process, and not considering them will result in an application that is too heavy for normal users to install.

Consequently, considering the fact that network connection is now accessible most of the time, the transferred data is not sensitive, and the mentioned delay is low enough to be ignored, placing the model on the server is selected as the best approach.

Methodology and Implementation

Considering the chosen approach for using models on this application, the main architecture of the system is the client-server architecture. Both sides will be introduced in advance.

Server Side

The main language used for server-side is Python, which has favorable features and libraries both for the implementation of the server and the use of AI models which are usually made with Python, and is also simple, widely used, adaptive, and platform-independent.
For solving the issue of managing all necessary libraries and packages, Anaconda[8] which consists of a virtual environment, packages, and Conda is used. Conda is used to make the desirable virtual environment and allows the packages to be installed, removed, or updated for a project, separated from other projects.

[8] https://www.anaconda.com

For making connections with the client-side and Flask[9] as a micro framework is used as it does not require specific tools or libraries and allows the developer to extend a simple core by his desirable choices. Using this tool, RSET Application Programming Interfaces are implemented to make the connections, using the rules of RESTful webserver. An alternative could be SOAP which needs more bandwidth and is not a proper choice for this project.

The server side consists of five components that are specifically designed, implemented, tested, and deployed for different features of the program and can have connections with each other, and as a result, a microservice architecture[10] that maps each component a microservice is designed. Microservices are highly maintainable and testable, loosely coupled, and independently deployable in small teams and considering business capabilities. These features made them the best choice for this system, as otherwise, updating, removing, or adding another component which is a necessity for generalization would have been impossible or bothersome, problems of a component would have impacts on all other components, and all of them had to be implemented with similar tools. Of course, increasing the number of microservice might finally result in problems in maintaining the system, but this is not the case for this application considering its scale. Each component is described in advance.

The first microservice is for using the database which is used for login and signup procedures, and saving the sent images and their produced results with the permission of the user. The used database is PostgreSQL[11] and consists of four main tables, one for users and others for images and results.

The second, third, and fourth microservices are for using the Object Detection and the two Image captioning models respectively. They include interfaces for receiving an image from the client-side, running the model on the received image, producing results and processing them, making connections with the database, and sending the processed result back to the client-side. The process that is done on the results is related to what will be shown to the user. So, for the Object Recognition case, it is a sentence made by the name of the objects on the image, for Object Detection, it is the location of each object and its name, for image captioning it is the caption with no specific change, and for music playlist recommender, it is a link to a playlist from SoundCloud which its subject is related to the detected objects based on word similarity using word2vec methods from Natural Language Processing techniques. It simply means that for example, if a bed is seen on the image, at is close in meaning to sleep, a playlist suitable for sleeping will be recommended to the user. Of course, the number of playlists is limited and not all objects are exactly related to a subjec5 in this case, but this option is featured to show the possibility of unlimited tasks done by AI models and its results are acceptable at this level.

[9] https://flask.palletsprojects.com/en/2.0.x/
[10] https://microservices.io/
[11] https://www.postgresql.org/

The last component is used for login and signup and works by receiving username and password, evaluating them, and returning results to the client for other steps of authorization. Having profiles and the possibility of login and signup makes it possible for users to have an exclusive album and keep their data safe.

Client Side

The client-side is implemented as an Android application using Java and Android Studio which is an Integrated Development Environment and makes it possible for developers to handle desired packages and modules, implement, and deploy the applications easily.

As the server is independent of the client, implementing this application for mobile phones with other operating systems is also possible with no needed change on the server-side, and also the main components will be similar in different platforms. In other words, the only difference is the tools and way of implementation. For all systems, this application consists of four main sections that will be introduced in advance.

The first and main section is the interface that is implemented to allow users to use models by taking pictures, sending them to the server, receiving back the result, and watching an album of results. As mentioned before, the results that are received in the client are based on what will be shown to the user. For Object Recognition, a sentence including names of the objects will be shown, for Object Detection a box around each object which is placed on the image using the received locations in different random colors in addition to the name of the object at the top will be shown, for Image Captioning models a caption will be shown, and for music playlist recommender a clickable link and a directive sentence will be shown. Also, the album is made with images on one side and results on the other side.

The second section is designed for login and signup and receives a username and password from the user, if the provided data was correct according to simple points including length and language, the data will be sent to the server and after receiving a result back the process continuous.

These two sections are in direct connection with interfaces that are implemented on the server and this becomes possible by OKHttp which provides functions for making connections and sending or receiving data.

The next section is a setting section and its main feature is a toggle button with which the user may approve permission for saving images or not. This and the username and password feature are the minimum designed methods for privacy.

The last section is an information section that provides the user with information about the application and methods and guides him in using different features. As mentioned before this application can have educational purposes for introducing AI models to ordinary people, and this section is deployed to be a help for this process.

These components are all implemented in a simple design and consequently, using the application with help from intelligent voice assistants, such as TalkBack for android and VoiceOver for IOS, is possible for visually impaired users. Also, simple design refers to a minimal and standard design that allows locating components and using them to be done conveniently.

Testing and Results

The introduced system is developed using at least three different models for five levels using client-server architecture. The server-side application can be deployed on a server using Docker[12] or by installing all requirements on the server itself, and the client-side application can be deployed on a mobile phone. Using the Internet, connections are made and image or user data can be sent to the server, models can be run, and the produced results can be sent back to the client to be shown to the user.

As the used models are previously tested using common metrics in related papers, the testing procedure for this application is limited to unit tests for the Java application, periodic practical tests during the implementation process, and final practical tests done by different users.

The two first methods help in finding errors, fixing them, and enhancing the quality of the application. The third test which includes observing the user while using the application can help in maximizing the usability of the application, and enhancing the experience.

In test cases with no problems with the network connection, all the features work as expected with no errors, and the connection between two sides of the system works properly. However, more users are needed for an exact test in order to evaluate the functionality of the system while facing a huge number of users. An example of produced results in different levels and provided options can be seen in figure 1.

Future Work

As this research is about an application, many methods are possible for improving by adding extra features. Among them, possible approaches that can increase the quality of this application without necessarily changing its main aspects are described in advance. These methods are both about server and client-side, and the models that are used for different components.

[12] https://www.docker.com/

About the models, as AI methods are growing rapidly, more efficient models which need less storage and can be run faster and more accurately are possible to be made, and the recent ones can be replaced by the new ones. If the models become light enough, even placing the models on devices might become possible and this brings the benefits of on-device inference into practice.

One example of other models that can be used is using Dense Captioning models instead of the Whole Image Captioning models that are used in this application. The main difference between them is that Dense Captioning models make as many as possible results according to the detected objects while Whole Image Captioning models produce only one sentence which best describes the model, and this transfer can be useful especially for visually impaired people.

Also, models that are made in languages other than English can be used and in this case, the language of the application can also become selective. These changes are all possible with no specific change in other sections of the application as it is implemented in a way to be generalizable.

As mentioned before, the client-side application is implemented for Android mobile phones, so implementing it for mobile phones with other operating systems is also a possible future work.

In addition, due to the purposes of this application some usages were designed and introduced to the user, but other applications, even using the same models can be added to the application by changing the way results are shown to users.

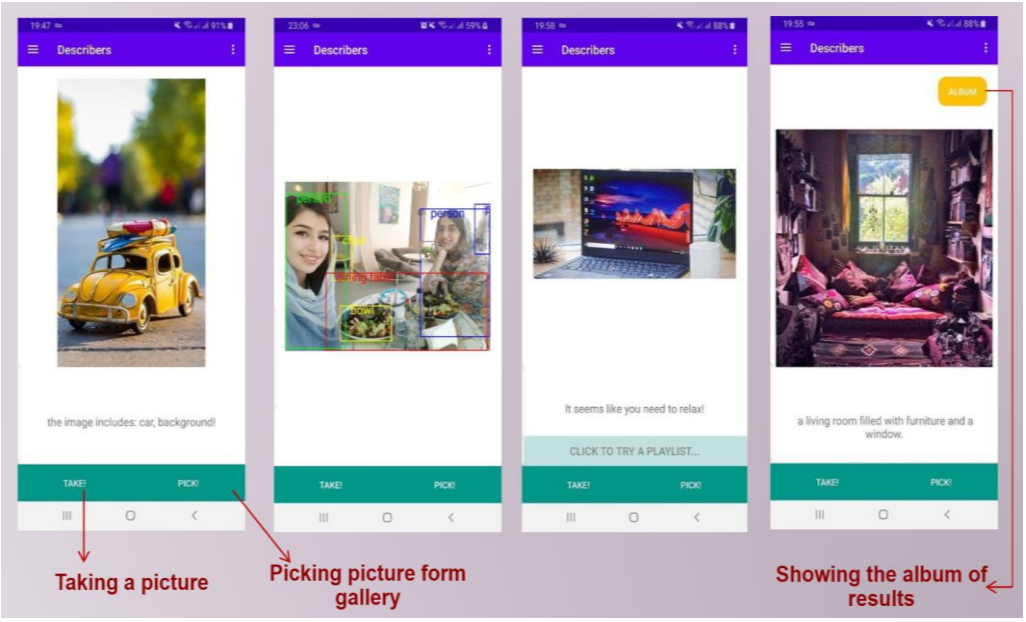

Figure 1: Results produced by the application for Object Recognition, Object Detection, Image Captioning, and Music Playlist Recommender in addition to introduction of options