
# Project Report for CSE376

*Interface for AI-generated-image Detector*

Shahrab Khan Sami
2018331003
shahrab03@student.sust.edu

Department of Computer Science & Engineering

December 30, 2022

**Abstract**

In this report, I present the development of an interface for an AI-generated-image detector using a Convolutional Neural Network (CNN) based on the Xception model [1] modified to work on the Fast Fourier Transform [2] of the input image. The goal of the interface is to enable users to easily input and classify images using the CNN model, without requiring any technical knowledge of machine learning. To create the interface, we designed a user-friendly android application and integrated the CNN model into the backend.

The result is an interface that is easy to use and which produces high accuracy in detecting AI-generated images. This work demonstrates the potential of using interfaces to democratize access to advanced machine learning models, and I believe that it has the potential to benefit a wide range of users.

# Contents

# 1 Introduction

Artificial intelligence (AI) has the potential to revolutionize many aspects of our lives, including the way we create and consume media. One area where AI has made significant strides in recent times is in the generation of images, with deep learning models capable of generating highly realistic images that are indistinguishable from those created by humans whether it be art pieces or camera photography. However, sometimes these AI-generated images can be difficult to identify, even for experts, and this can have significant implications for fields such as digital media, where the authenticity of images is important. The use of AI-generated images has the potential to blur the lines between reality and fiction, and this raises important questions about the authenticity of images and the potential for misinformation.

To address this problem, I present an interface for an AI-generated-image detector using a CNN based on the Xception model [1] modified to work on the FFT of the input image. The interface is developed by me and my project partner *Md Adith Mollah* [3]. By providing an easy-to-use interface for detecting AI-generated images, we aim to help users make informed decisions about the content they consume. In addition, the development of such an interface could have broader applications in fields such as digital forensics, where the authenticity of images is critical. The model is developed by *Thomas Oakley Browne* [4], a postgraduate student of La Trobe University, Melbourne, Australia. In this report, I describe the design and implementation of the interface and the evaluation of its performance.

In the following sections of this report, I describe the problem of detecting AI-generated images and building an interface for it in more detail and review the related work in this area. I then describe the methodology used to develop and evaluate the interface, including the design of the android application and the training and evaluation of the CNN model. Finally, I present a detailed system description of our project. I conclude by discussing the implications of our work and suggesting areas for future work.

Overall, this report aims to provide a comprehensive overview of the development and evaluation of our interface for detecting AI-generated images. I believe that our work has the potential to benefit a wide range of users by providing a simple and effective way to detect and classify AI-generated images. I hope that the app helps users make more informed decisions about the content they consume and helps ensure the authenticity of images in the digital world.

# 2 Problem Definition

As AI system develops, it is becoming more and more incumbent that its downsides are addressed and steps be taken to prevent them. One Such issue is the prevalence of misinformation generated by advances AI systems which are increasingly difficult to distinguish from true information. Many researchers have started to tackle such problems often with the help of other AI systems and Neural Network models. But these systems

are still at a point that they are almost inaccessible without proper setup and relevant knowledge in the subject.

At the very least one needs to know a programming language like python to even begin to use the models developed by the researchers. Then correctly loading and setting up the model, ensuring proper format of input, proper interpretation of the results and their entailment etc are challenging tasks for a person not of technical background.

Furthermore, the problem of detecting AI-generated images is an active area of research, and there are many different approaches and techniques that have been proposed. In order to effectively detect misinformation in a field, one has to choose the right model.

To comprehend the relevance of the problem, it will be important to review the relevant literature and to understand the strengths and limitations of different approaches.

# 3 Related work

There are broadly two types of works that are related to our project:

## 3.1 Studies on the problem of detecting AI-generated images

There are many research papers and articles that have addressed the problem of detecting AI-generated images and the various approaches that have been proposed for solving this problem.

1. GAN image detector based on a limited sub-sampling architecture and suitable contrastive learning paradigm [5]
2. Transferable framework for detecting GAN-generated images using a teacher and student model that iteratively teach and evaluate each other to improve performance [6]
3. Fake image detector that uses Self-attention mechanism to exploit structural defects in GAN with priority to detecting up-sampling [7]
4. Comparative study on forgery detectors of images from social media [8]

## 3.2 Studies on user interfaces and user experience (UX)

As the interface we are developing is intended to be user-friendly, it is important to consider the user experience of the interface. There is a large body of research on user interfaces and UX, and these can provide valuable guidance on designing interfaces that are easy to use and enjoyable for users.

1. Paper introducing core concepts from UI/UX design important to cartography and visualization, including the distinction between UI design and UX design, Norman's stages of interaction framework, and three dimensions of UI design [9]

2. Study to definitions of UX and its elements through a literature survey, user interviews, and indirect observations, and proposes definitions of UX, usability, affect, and user value to help design products or services with greater levels of UX [10]
3. Paper about a systematic literature review of existing studies on mobile UI design patterns, including their strengths and areas that require further research [11]

# 4 Methodology

## 4.1 Research and Planning

First, we conducted a literature review to understand the current state of the art in using CNN models for image classification and the benefits of using Flutter for mobile app development. This helped us to understand the challenges and opportunities involved in using these technologies for our project.

Next, we identified the specific requirements for our app, including the type of images that the CNN model would need to classify and the features that the app should have for a user-friendly experience. Based on these requirements, we selected an appropriate pre-built CNN model that had been trained on a large dataset and had a high accuracy rate for image classification.

We then began the planning process for the app development, including designing the user interface and user experience, determining the necessary functionalities and features, and creating a project timeline. We also identified any potential challenges or issues that we might encounter during the development process and planned strategies to address them.

Overall, the research and planning phase was crucial for ensuring the success of our project and allowed us to develop a clear roadmap for the development process.

## 4.2 Setting up the development environment

WE decided to develop the app using Flutter [12]. For this, it was necessary to set up the development environment on the computer. This included installing the flutter SDK, configuring the Android Studio IDE, and setting up an emulator to test the app on. The flutter documentation provided detailed instructions on how to set up the development environment for both Windows and macOS operating systems. Once the development environment was set up, it was possible to create a new flutter project and begin building the app.

## 4.3 Integration of the model into the app

For this, we developed an API backend for the app using FastAPI [13]. The model was loaded using the Keras [14] library. The input and output layers of the model were

identified and mapped to the appropriate widgets in the Flutter interface. All necessary pre-processing steps were implemented in the Flutter interface to ensure that the input data was compatible with the model and that the output data was in a usable format.

The Flutter interface was then tested with a small sample of input data to ensure that the CNN model was being correctly integrated and was producing the expected output. All necessary adjustments were made to the Flutter interface or the CNN model to ensure that they were working together smoothly.

## 4.4 Development of the app interface

The first step in the development of the app interface was to design the overall layout and structure of the app. This included deciding on the placement of input fields, buttons, and output display areas on the screen and determining the overall aesthetic and user experience.

Once the overall layout was developed, the input fields were implemented for users to input images for the CNN model to analyze. These input fields were designed to be user-friendly and intuitive, allowing users to easily select and upload images from their device.

Next, buttons were implemented to trigger the model and display the output. These buttons were designed to be clearly visible and easy to press, allowing users to quickly and easily use the app.

Finally, output display areas were implemented to show the results of the model. These display areas were designed to be visually appealing and easy to understand, allowing users to quickly and easily interpret the results of the model.

## 4.5 Testing and debugging

Testing and debugging was a critical step in the development process, as it allowed us to identify and fix any issues with the app's functionality. To test the app, we used a range of different devices, including smartphones and tablets, to ensure it was compatible with a variety of different hardware and operating systems. During testing, we focused on verifying that all features of the app were functioning correctly, including the input fields for images, the button for triggering the model, and the display areas for the model output. If any issues were discovered, we used debugging tools and techniques to identify the root cause of the problem and implement a solution. This process was repeated until the app was stable and free of any significant issues, ensuring that it would be reliable and user-friendly for our intended audience.

## 4.6 Finalization

As our project is still ongoing, this phase of development is not yet complete. We are in the phase of optimizing the core functionality and planning to add a number of

additional ones like User account system, a history system for the users to save their recent searches and results, options for users to select from multiple models which might be good for specific cases etc. We hope that with the additional feathers the app would be a valuable tool for users of any background in combating misinformation.

# 5 System Description

## 5.1 Overview & FrontEnd

Our system is primarily an android application that allows the user to upload an image which is then fed into a CNN model to predict weather the image is generated by an AI system or not.

When the user first opens the app, they are greeted with a page as displayed in Figure-1a. The UI design prompts the user to weather upload an image or use the device camera to take a photo to be evaluated by the model.

## 5.2 The CNN model

The model is designed using Tensorflow [15] and trained on real images from Flickr [16] and fake images generated by StyleGAN2 [17]. Some evalutaion metrics of the model as compared to the Xception [1] model is given below:

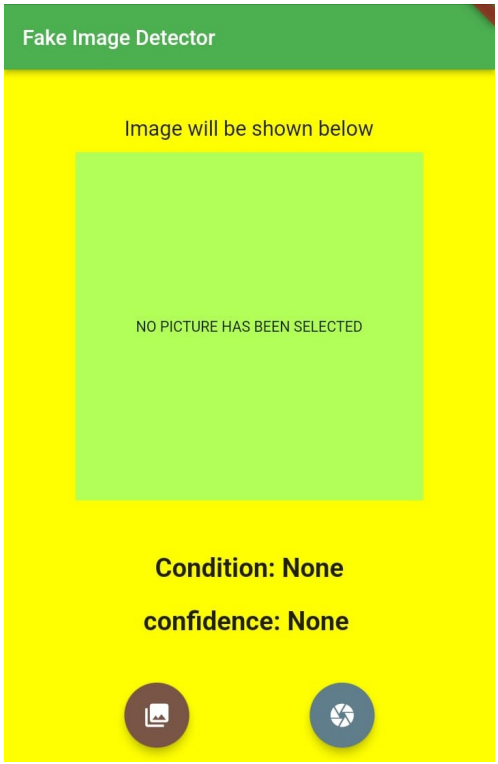
| Metrics | Xception with Augmentations | Fourier Xception |
|---|---|---|
| Loss | 0.0842 | 0.0354 |
| Accuracy | 0.9775 | 0.9937 |
| Area Under Curve (AUC) | 0.9948 | 0.9987 |
| Precision | 0.9823 | 0.9975 |
| Recall | 0.9725 | 0.9900 |
| F1 Score | 0.9765 | 0.9929 |

Table 1: Evaluation Metrics Comparison

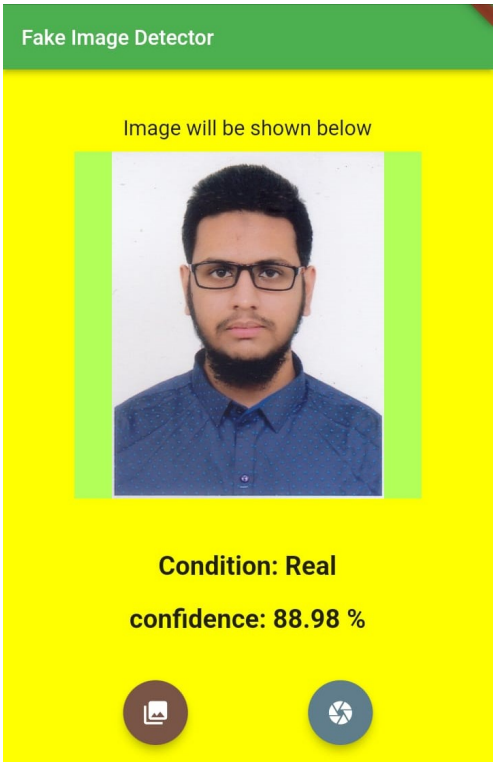
## 5.3 BackEnd & Processing

Once the user chooses an image, the image is displayed on the view frame. Simultaneously the image is sent via a *POST* request to the backend API hosted at `hugginface.co` [18]. In the backend, the image is processed as:

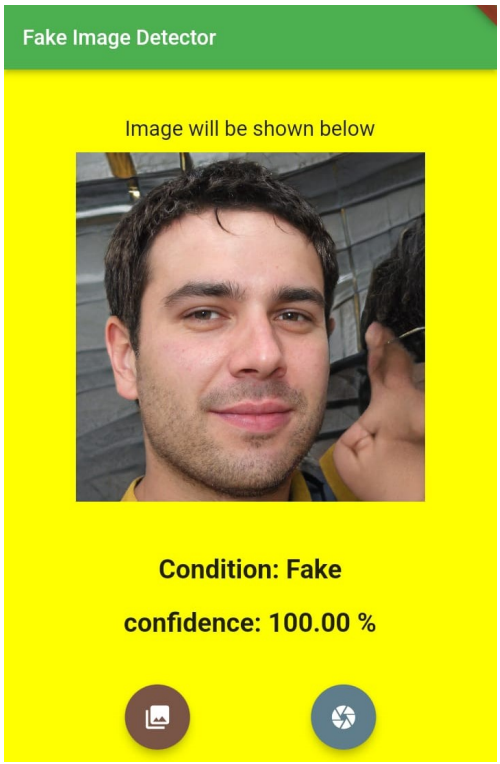
- The color image is converted into gray-scale according to ITU-R BT.601-7 recommendations [19].
- FFT algorithm is applied followed by FFT shift
- Magnitude spectrum of the resultant image is calculated




(a) Homepage




(b) Real




(c) Fake1


(d) Fake2

Figure 1: UI demonstration

- The Magnitude spectrum is color-mapped into Viridis color-map

The processed image is then given as input of the model and the resultant output is sent as a response to the frontend flutter application.

### 5.4 Result & Code

The verdict of the model is then determined from the response and displayed along with the confidence level weather the model says the image is real (Figure-1b) or fake (Figure-1d & Figure-1c).

The source code for both frontend and backend of our project can be found on GitHub [20] [21]

## 6 Conclusion

In this report, I presented the development of an interface for an AI-generated-image detector using a Convolutional Neural Network (CNN) model based on the Xception model modified to work with the Fast Fourier Transform of the input image. Our goal was to enable users to easily input and classify images using the CNN model without requiring any technical knowledge of machine learning. To create the interface, we designed a user-friendly android application and integrated the CNN model into the backend.

The resulting interface is easy to use and produces high accuracy in detecting AI-generated images. This work demonstrates the potential of using interfaces to democratize access to advanced machine learning models and has the potential to benefit a wide range of users. The development of such an interface could have broader applications in fields such as digital forensics, where the authenticity of images is critical.

In the future, we hope to continue improving the accuracy and performance of our AI-generated image detector, as well as exploring potential applications in other areas. I believe that our work has the potential to have a significant impact on the way we consume and verify the authenticity of images in the digital world.

# List of Figures



# List of Tables

# References


[1] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1251–1258, 2017.

[2] Wikipedia contributors, "Fast fourier transform — Wikipedia, the free encyclopedia." `https://en.wikipedia.org/w/index.php?title=Fast_Fourier_transform&oldid=1129063361`, 2022. [Online; accessed 29-December-2022].

[3] Md Adith Mollah, "Linkedin profile." `https://www.linkedin.com/in/md-adith-m-36aa70127`, 2022. [Online; accessed 30-December-2022].

[4] Thomas Oakley Browne, "Linkedin profile." `https://www.linkedin.com/in/thomas-oakley-browne`, 2022. [Online; accessed 30-December-2022].

[5] D. Cozzolino, D. Gragnaniello, G. Poggi, and L. Verdoliva, "Towards universal gan image detection," in *2021 International Conference on Visual Communications and Image Processing (VCIP)*, pp. 1–5, IEEE, 2021.

[6] H. Jeon, Y. Bang, J. Kim, and S. S. Woo, "T-gd: Transferable gan-generated images detection framework," *arXiv preprint arXiv:2008.04115*, 2020.

[7] Z. Mi, X. Jiang, T. Sun, and K. Xu, "Gan-generated image detection with self-attention mechanism against gan generator defect," *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 5, pp. 969–981, 2020.

[8] F. Marra, D. Gragnaniello, D. Cozzolino, and L. Verdoliva, "Detection of gan-generated fake images over social networks," in *2018 IEEE conference on multimedia information processing and retrieval (MIPR)*, pp. 384–389, IEEE, 2018.

[9] R. E. Roth, "User interface and user experience (ui/ux) design," *Geographic Information Science & Technology Body of Knowledge*, vol. 2, pp. 1–11, 2017.

[10] J. Park, S. H. Han, H. K. Kim, Y. Cho, and W. Park, "Developing elements of user experience for mobile phones and services: survey, interview, and observation approaches," *Human Factors and Ergonomics in Manufacturing & Service Industries*, vol. 23, no. 4, pp. 279–293, 2013.

[11] L. Punchoojit and N. Hongwarittorrn, "Usability studies on mobile user interface design patterns: a systematic literature review," *Advances in Human-Computer Interaction*, vol. 2017, 2017.

[12] Google, "Flutter: An open-source ui software development kit." `https://flutter.dev`, 2017. [Online; accessed 30-December-2022].

[13] Sebastián Ramírez, "Fastapi: A web framework for developing restful apis in python." `https://fastapi.tiangolo.com`, 2018. [Online; accessed 30-December-2022].

[14] François Chollet, "Keras: A software library that provides a python interface for artificial neural networks." https://fastapi.tiangolo.com, 2015. [Online; accessed 30-December-2022].

[15] Google, "Tensorflow: A free and open-source software library for machine learning and artificial intelligence." https://www.tensorflow.org, 2015. [Online; accessed 30-December-2022].

[16] Yahoo, "Flickr: An american image hosting and video hosting service." https://www.flickr.com, 2004. [Online; accessed 30-December-2022].

[17] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119, 2020.

[18] Thomas Wolf, Lysandre Debut, and Victor Sanh, "Hugging face: A company that develops tools for building applications using machine learning." https://huggingface.co, 2016. [Online; accessed 30-December-2022].

[19] International Telecommunication Union, "Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios." https://www.itu.int/dms_pubrec/itu-r/rec/bt/R-REC-BT.601-7-201103-I!!PDF-E.pdf, 2011. [Online; accessed 30-December-2022].

[20] S. Khan, "Project 350 FastAPI (BackEnd)." https://github.com/S-K-Sami/P350_FastAPI, 2022.

[21] M. A. Mollah, "Project 350 Flutter (FrontEnd)." https://github.com/Adith082/Project-350, 2022.

**Acknowledgement**

The development of this project would not have been possible without the support and guidance of several individuals and organizations.

First and foremost, I would like to express my gratitude to my project partner, Md Adith Mollah, for his valuable contributions and for sharing his knowledge and expertise in the development of the android application.

I would also like to thank Thomas Oakley Browne for providing the CNN model that forms the core of our interface.

Most especially, I would like to thank Mohammad Shahidur Rahman sir and Md Masum sir for supervising this project, Dr. Farida Chowdhury mam for sharing her knowledge and her guidance in this course and Dr. Jabed Chowdhury for his valuable advice regarding the development of the interface.

Finally, I would like to express my appreciation to my University for providing the resources and support needed to complete this project.