Detecting DeepFakes Using Deep Learning

"Meet Team DeepTRUTH" by Team 5, Image created for Senior Design I

Jouri Ghazi, Jahtega Djukpen, , Ashton Bryant, Zacary Louis

Capstone Senior Design II

Fall 2025

TABLE OF CONTENTS

Problem Statement **4**
Overall Objectives *4*
Background & Historical Perspective *4*
Needs Statement *5*
Major Design and Implementation Challenges *5*
Dataset Selection & Management 5
Machine Learning Training 6
Website Development 6
Implications of Project Success *6*
Requirements and Constraints **8**
Algorithmic Performance Criteria *8*
Accuracy 8
Precision 9
Recall 9
F1 Score 9
Solution Efficiency *10*
Algorithm Process Time 10
Website Boot Time 10
Resource Usage 10
Dataset Generalizability (Scalability) 10
Demonstration Usability *11*
Interpretability 11
Accessibility 11
User Interface 11
Code Maintainability *11*
ISIP Guideline Compliance 11
Documentation 12
Ethics & Security Criteria *12*
Algorithmic Transparency 12
Website Security 13
Ethical Dataset Development 13
Potential Solutions **14**
Face Detection 14
Demonstration 15
Solution 1: Random Forest *15*
Discrete Cosine Transform 15
Random Forest Model 15
Solution 2: Convolutional Neural Network (CNN) *17*
Parallel Dataset 17
CNN 18

Solution 3: Xception Model *20*
Model 20
Next Step *21*
Appendix **22**
Abbreviations Used *22*
Citations **23**

Problem Statement

Overall Objectives

With the ongoing rise of AI generated media, detecting such content has become a critical challenge to maintain digital authenticity and prevent fraud. DeepFakes are a form of AI generated content that typically mimics one's facial features or voice to replace or alter a person's identity [1]. DeepFakes pose a great risk to the spread of misinformation, our detection tool would work towards enhancing enterprise security against DeepFake attacks and can be leveraged within an educational scenario to enhance media literacy education [2]. This project aims to develop a web-based DeepFake detection tool, trained on a machine learning (ML) algorithm.

Our tool currently processes all image file forms supported by the Python Image Library, recognizing faces using the Haar Cascade method, and determining the authenticity of the image using a random forest algorithm, producing a Real/Fake result with a probabilistic confidence score. Moving forward, our aim is further improving the accuracy and accessibility of our tool. Among the upgrades we aim to make, we will move our web tool to a publicly hosted server. The next critical upgrade is our algorithm; within this endeavor we aim to reduce overfitting our algorithm to the training data so our tool is generalizable to a wide variety of image conditions. To achieve this, we aim to implement a deep learning (DL) convolutional neural network (CNN). Additionally, we hope to improve the front-end design and user interface (UI) of the web tool and add additional functionalities, such as the support of audio and video content.

Background & Historical Perspective

DeepFake content can take multiple forms, such as images, audio, video. DeepFake images can take the form of either generated depictions of nonexistent people or manipulated photographs of real individuals. Modern generative AI (gen-AI) content can instantaneously create photorealistic content that is almost indistinguishable from authentic images, heightening the risk of misuse, deception, and reputational harm [3]. Social media has allowed for the instant spread of generated content, amplifying its harmful effects, individuals are increasingly vulnerable to targeted DeepFake attacks which may result in psychological and financial distress. To mitigate these risks, a detection tool is required to accurately distinguish between genuine and manipulated content.

DeepFakes emerged as photo editing tools have become widely accessible with the development of generative modeling during the 2010s. The term "DeepFake" was first coined on Reddit in 2017, as availability of open source editing software and large image datasets, enabled by the rise of social media, helped advance gen-AI technology [4].

The development of generative adversarial networks (GAN) made it possible to create of photorealistic DeepFake images. GAN are a type of neural network based DL model that works by having two networks, the generator and discriminator, compete with one another in an iterative process for improved results. The generator network forges new data with the goal

of making it indistinguishable from real data in attempt to fool the discriminator network. The rapid evolution of GAN models oftentimes outpaces the development of new algorithms, outdating earlier generations of detection methods. Generated DeepFakes increasingly captures natural human attributes, such as small changes in lighting, texture, and facial expressions [5].

Current detection models face a moving target as GAN generated images are continuously evolving to bypass detection, where each generation of forgeries may potentially nullify existing detection methods, requiring a continuous cycle detection retraining stay current with deepfake advances. This cycle is referred to as an "arms race" where detection and generation technology develop along one another, each motivating the other to improve [6].

The United States National Security Agency, Federal Bureau of investigation and the Army Criminal Investigation Command have publicly addressed the threat posed by the emergence of Gen-AI content, stating that "The tools and techniques for manipulating authentic multimedia are not new, but the ease and scale with which cyber actors are using these techniques are. This creates a new set of challenges to national security" [7]. Several laws within the United States have been introduced to mediate the risk posed by DeepFakes, Act 35 passed in Pennsylvania makes it a crime to create or distribute deepfakes for fraudulent purposes or to cause harm, this act was signed on July 7th, 2025, and will be made effective September 5th, 2025 [8]. As Gen-AI continues to improve the development of effective detection tools, it remains a critical research area. These initiatives demonstrate the need for accurate and reliable DeepFake detections method to mitigate the potential harm of Gen-AI content.

Needs Statement

This project highlights the need for the implementation of a reliable easy-to- access AI content detector to distinguish DeepFake content. This tool can mitigate the ongoing risk of fraudulent activity pertaining to identity theft, defamation, and psychological harm potentially caused by the spread of DeepFake content.

Major Design and Implementation Challenges

Dataset Selection & Management

Our DeepFake Detector depends on a machine learning model; whose accuracy is correlated to the quality and quantity of the training dataset [9]. The dataset we choose should contain a diverse selection of high-quality images, capturing variations in lighting, resolution, facial expression and demographics to ensure that the model generalizes well [10]. Although several datasets are available, many lack samples generated with the latest techniques, limiting the generalizability and accuracy of our model [11]. Table 1 in the Appendix presents the DeepFake datasets considered, sourced from a variety of companies and initiatives.

Gen-AI content increasingly mimics natural attributes, such as subtle variations in lighting, texture and facial expressions, which is rapidly outpacing the development of new

algorithms, outdating earlier generations of detection methods. As the quality of GAN-generated content improves, an artifact based detection approach becomes less effective. This increases the need for a diverse and comprehensive dataset to reliably distinguish between real and fake content [10]. Data diversity and recency poses a design constraint for our detection method.

Machine Learning Training

The risk of overfitting poses another design constraint, which occurs when a model learns the training data too closely, leading it to capture noise and specific patterns, hindering its ability to perform onto new, unseen data. This is caused by diversity and bias found within the data which prevents the model to reliably perform on real-world inputs.

Deep learning (DL) relies on multi-layered neural networks to discover patterns from large datasets. A Convolutional Neural Network (CNN) is a form of DL algorithm that works with grid-like data and learns the spatial hierarchies of features [12]. Although CNNs are an effective in processing images, this method lacks interpretability, making it difficult to understand which patterns and cues the model uses to distinguish real from fake images. This poses a barrier of trust and accountability and hindering potential improvements and parameter tuning.

Website Development

A priority for the website development is to allow users to easily upload images and interact with the UI. The result of the detection should be clearly displayed to encourage user confidence within our tool, this can be accomplished through visual cues or confidence scores, allowing users to understand the model's decision [13]. The website should be accessible and well-designed to provide a straightforward and intuitive experience with uploading and interpreting results.

The site should support real-time processing, as the CNN detection models are computationally intensive and time consuming, the tool should deliver fast results to ensure a seamless demonstration and improve the user's experience [14]. The interface should be easy to use and intuitive, allowing the user to seamlessly engage with the site's tools.

Implications of Project Success

This project aims to develop an algorithm that accurately differentiates between real and generated images. The final result would be a functional web application where users can upload an image and instantly determine its authenticity.

The broader implications of success extend to several United Nations Sustainable Development Goals (SDGs):

SDG 3: Good Health and Well-Being – By limiting the spread of harmful and exploitative deepfake content, individuals are better protected from psychological distress, harassment, and identity misuse. For instance, 67% of victims of image-

based sexual abuse experience negative mental health effects, including anxiety and long-lasting distress [15].

SDG 4: Quality Education – The tool would reduce misinformation and promote digital literacy, helping learners and educators access trustworthy information in an increasingly digital world.

SDG 9: Industry, Innovation, and Infrastructure – Developing an advanced AI system contributes to technological innovation and strengthens the security of digital infrastructures.

SDG 10: Reduced Inequalities – Vulnerable groups, including women and minorities, are disproportionately affected by deepfakes. Women make up 96% of all deepfake pornography victims, and in some regions, as many as 40% of women have experienced online harassment. These statistics show that this project can help safeguard vulnerable groups and reduce digital exploitation and abuse [15].

SDG 12: Responsible Consumption and Production – Encourages ethical use and creation of media by making manipulations easier to identify and discouraging irresponsible content production.

SDG 16: Peace, Justice, and Strong Institutions – By preventing deepfakes from undermining trust in institutions, media, and democratic processes, the project reinforces accountability, justice, and societal trust. Research has shown that exposure to deepfakes significantly increases distrust in government and erodes public confidence in democratic systems and the rule of law [16].

The success of this project would not only demonstrate the power of machine learning in addressing modern digital challenges but also contribute meaningfully to the global goals of ensuring well-being, reducing inequality, promoting innovation, and protecting the integrity of information.

Requirements and Constraints

Our solution consists of a DeepFake Detection model, and a demonstration interface that is deployed on the ISIP Cluster and is publicly available. The requirements and constraints criteria include algorithmic performance, which is a measure of our detection model's ability to reliably detect the generated content. Demonstration usability, which determines the quality of our demonstration website, ensuring its accessibility to all users. Another criteria is the efficiency of our solution, measuring the time and computational resources required to reliably classify the DeepFake. Code maintainability is used to make sure that our code complies with the ISIP standards and is usable in the future. The final criteria is about the ethics and safety of our training and deployment process, ensuring that the results provided by the detection model are explainable, and adheres to privacy restrictions. Table 1 references the requirements and constraints.

Table 1: Requirements & Constrains

	Criteria	Requirement /Constraint	Unit	Goal Value	Negotiable /Non-Negotiable	Standard
Performance	Accuracy	requirement	%	95%	Negotiable	N/A
	Precision	requirement	%	90%	Negotiable	N/A
	Recall	requirement	%	85%	Negotiable	N/A
	F1 Score	requirement	%	85%	Negotiable	N/A
Efficiency	Algorithm Process time	requirement	ms/pixel	1500	Negotiable	N/A
	Website boot time	requirement	Seconds	5	Negotiable	N/A
	Resource Usage	constraint	gb	<128	Non-negotiable	ISIP
	Dataset Generalizability	requirement	%	60%	Negotiable	N/A
Code Maintainability	ISIP Guideline compliance	constraint	Pass/Fail	Pass	Non-negotiable	ISIP
	Documentation	requirement	Pass/Fail	Pass	Non-negotiable	ISO 12207
Usability	Interpretability	requirement	Pass/Fail	Pass	Negotiable	WACG
	Accessibility	requirement	Pass/Fail	Pass	Negotiable	WACG
	User Interface	requirement	Pass/Fail	Pass	Negotiable	WACG
Ethics & Security	Algorithmic Transparency	constraint	Pass/Fail	Pass	Non-negotiable	IEEE 7003-2024 standard
	Website Security	requirement	Pass/Fail	Pass	Non-negotiable	ISO 27001

Algorithmic Performance Criteria

Accuracy

The accuracy of an AI model measures the correctness, calculated by the percentage of correctly classified instances in relation to the number of total classifications. A model developed by VGG11 reports up to 94.46% accuracy on a gen-AI detection model [17]. Based on these results, our detection model is required to reach at least 95% in performance

accuracy, this would prove the feasibility the implementation of a DeepFake detection method [17]. Our previously conducted surveys have resulted in at 60%. We will consider this value as negotiable due to the restricted computational time and resources required for algorithmic training.

Table 2: Confusion Matrix

Actual/Predicted	Predicted Real	Predicted Generated
Actual Real	True Negative (TN) Correctly classified real images	False Positive (FP) Real images wrongly classified as fake
Actual Generated	False Negative (FN) False images wrongly classified as Real	True Positive (TP) Correctly classified generated images

Precision

Precision is the proportion of all the model's positive classification to those that are actually positive. Within this project, this metric would measure the fraction of images that are correctly classified as fake to all images classified as fake. A perfect model would have no false positive classifications and result in a precision score of 1.0. Within the scope of this project, we aim for a score of 90% and consider this value as negotiable as it is correlated to our model's accuracy [18].

$$Precison = \frac{True\ Positive}{True\ Positive + False\ Positives}$$

Recall

Recall measures how often a model correctly classifies positive instances in comparison to all true positives. In this project, this metric measures images classified as fake to all fake images. A perfect model would have a recall of 1.0 meaning all fake images are classified correctly. Based off comparative models, we should be able to achieve a negotiable recall value of 85% [18].

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negatives}$$

F1 Score

The F1 score is the harmonic mean of precision and recall. It is especially important when working with imbalanced datasets, where one a class (e.g. real or fake) contains significantly more data than the other. For our model, a target of at least 85% is required. The F-1 score is calculated from precision and recall. Table 2 presents a confusion matrix, illustrating the categories for our model performance [18].

$$F1 = 2 \times \frac{Precison \times Recall}{Precison + Recall}$$

Solution Efficiency

Algorithm Process Time

An algorithm with a short execution time would improve its usability and would allow for its integration into software or web services that utilizes real-time user interaction. Achieving a runtime under 1500 ms per 1000×1000 pixels ensures that our model remains competitive in the rapidly evolving field of DeepFake detection, while also leaving room to increase computational complexity. Faster execution additionally allows for a greater number of tests to be conducted, accelerating model refinement and improvement during training. This metric is negotiable [19].

Website Boot Time

Considering that the website speeds is dependent on the internet connection, and we have limited control of our server, a requirement for our demonstration is a short website boot time. This website should load within 5 seconds. As this website serves as a feasibility test, it must support real-time demonstration.

Resource Usage

The demonstration application will be available on the ISIP website, and will operate within the constraints of the Neuronix Cluster resources. This application is allocated a limitation 128GB and the processing capacity of the cluster [20]. For the purposes of this project, a single node of CPU should be sufficient for the machine learning system.

Dataset Generalizability (Scalability)

Scalability within this project refers to how a model handles larger, more diverse data sets [21]. Scalability makes sure that the model developed can perform well, keeping up with the increasing scale of the datasets. For our project, scalability isn't as critical of a concern. Since we are mainly focused on proving the feasibility of developing this type of machine learning powered system, we are not as focused on preparing a system for ultra-large datasets. What is important is that our system can still perform on unseen data and data that is significantly different from the training data. This is what the negotiable requirement under this criterion deals with.

We have one negotiable requirement under this criterion and that is dataset generalizability on unseen data. With this project, we are trying to prove that developing a system of this nature is feasible. To do that, the system we are currently developing needs to be able to perform on unseen data.

Demonstration Usability

Interpretability

Once the classification results are provided, the user must be able to interpret the results and the algorithm that was utilized. To achieve this, our interface will display a clear textual result ("real" or "fake") for each prediction, with corresponding color allocations to reinforce understanding. To further enhance transparency, a confidence meter will indicate how certain the model is about its decision, based on the internal probability or certainty score for the provided data. These interpretability features help ensure that users can not only see outcomes, but also understand the reasoning behind them, by improving and facilitating informed responses [22].

Accessibility

Accessibility is a negotiable requirement to ensure inclusive use of the system for all users, regardless of disability or device limitations. To meet this requirement, the web application must be compatible with screen readers and support full keyboard navigation for individuals with visual impairments. High-contrast display modes and alternative text descriptions for all non-text content will be implemented to comply with accessibility standards. Wherever possible, the design will follow the Web Content Accessibility Guidelines (WACG) [23], which provide internationally recognized criteria for making web technology more inclusive. Incorporating these practices not only ensures legal and ethical compliance but also improves the overall usability of the platform.

User Interface

By using a responsive design framework, the user interface must be able to adjust to a variety of screen sizes and devices, including desktops, tablets, and smartphones. Relying on legible typography and an easy-to-understand structure, the layout will minimize visual clutter while maintaining a polished yet straightforward appearance. Classification results will be displayed using consistent color coding, making sure that the color selections follow accessibility standards for contrast. The WCAG provides design principals, stating that the website's content must be perceivable, and the interface components be operable, with the website's usage instructions be understandable, and the website content can be robust enough to be widely interpreted. The principles of perceivability, operability, understandability and robustness are defined as POUR [24]. These principles support the professionalism, responsiveness, and clarity of our demonstration, ensuring its ease of use and accessibility for all users.

Code Maintainability

ISIP Guideline Compliance

Our first constraint is compliance with the ISIP standards. This non-negotiable constraint will cover the coding standards that we will follow. We chose to use the ISIP standards because

the software we are producing will reside in the ISIP environment. Additionally, our web tool will be hosted on ISIP's web server. Our non-negotiable requirement is documentation, this includes a user guide, in-line comments in our code, and a project overview page. This documentation will facilitate upgrades and edits we make throughout this semester and will aid future developers who need to understand and/or upgrade our software [20].

Documentation

Code maintainability refers to how easily code can be modified and updated over time [25]. More specifically, maintainable code makes it easier and more efficient for current and future developers to fix bugs, add features, and update the code to keep up with modern technology. Writing and organizing code that is maintainable is crucial to the longevity of the codebase, especially with how rapidly software is evolving nowadays. The ISO 12207 standard defines the software lifecycle process, providing a structured framework to systematically develop software [26]. For this project, writing clean, maintainable code is important for both our current and future progress. As we write our code, making it maintainable will ensure that we can easily make updates as we progress through the semester. Furthermore, this will allow any future students to easily understand how our code works in case they decide to further our project or are simply learning from our work. There are many strategies for writing maintainable code, including following coding standards, writing meaningful comments, using descriptive naming conventions, and maintaining up-to-date documentation [25]. Our requirement and constraint under this criterion reflect some of these strategies.

Ethics & Security Criteria

Algorithmic Transparency

Algorithmic transparency is a non-negotiable constraint of our system to ensure that the processes behind classification remain interpretable and accountable. CNNs and other deep learning models are considered to be "black-box" systems, making it hard for stakeholders and users to comprehend the reasoning behind a particular decision [27]. To make it clear which patterns affect classification results, our proposed solution prioritizes explainability, such as feature visualization and additional interpretability methods, to clarify which patterns influence classification outcomes. By prioritizing explainability alongside accuracy, we ensure ethical alignment with the broader AI community's emphasis on fairness, accountability, and transparency [27]. This requirement reduces the risk of misuse, builds user trust, and provides auditors the ability to assess whether the system behaves as intended.

When developing our algorithm, we need to be aware of unintended bias, where a model may mistakenly classify against a group of individuals based on characteristics such as race or gender. This stems from the underrepresentation of a group within the training dataset, when unmonitored, these biases could result in systematic discrimination. The IEEE 7003-2024 standard provides a framework to address these risks [28]. This standard calls for the establishment of a bias profile, where all the considerations regarding bias are documented.

Website Security

Website security is a non-negotiable requirement, evaluated as pass/fail. Since the trained detection model and user-submitted data are accessible through a web interface, secure handling of inputs and outputs is mandatory. Basic security measures such as input validation and HTTPS encryption must be implemented to protect sensitive interactions [29]. The ISO 27001 is referred to as the international standard for Information Security Management Systems, providing a framework for managing sensitive information securely. In addition, no user information or uploaded data will be retained by the system after classification is complete; all user input is immediately deleted to ensure privacy and prevent data misuse [30]. A breach would compromise both user privacy and model integrity, thereby disqualifying the system for deployment. Successful implementation of these standard security protocols and privacy protections upholds user trust in the platform.

Ethical Dataset Development

When handling our data, we need to maintain ethical dataset practices. This constraint determines where we source our datasets and images from. We are only sourcing images from databases that are open source and pull their images from sources like Google and Facebook which are licensed. ISO 27001 outlines the requirements for the proper classification, handling, and protection of information [31]. Following these guidelines ensures data is handled properly. Following these guidelines and only sourcing our data from the sources name earlier, we ensure that we aren't collecting and using individual's private data without their consent or knowledge [32].

Potential Solutions

Machine learning models need large amounts of representative high-quality data. Our model required that we source thousands of real and DeepFake images. To fulfill our needs, we chose the OpenForensics dataset. This open-source dataset contains 45,473 real images and 70,325 fake images taken from Google's Open Images Dataset. Each image is richly annotated, with segmentation masks describing outlines of faces in pixel values, and bounding boxes referencing rectangular areas around faces. This means that DeepFake pixels are known. This dataset is challenging due to sophisticated GAN models deployed onto high resolution images. Furthermore, most images contain multiple faces, with a mixture of real and fake faces.

Our solution transformed this data for use with NEDC tools. Annotations were reformatted from large "json" files containing all annotations into numerous "csv" files. Each '"csv" file contained annotations for a single image. The annotations categorized each row as a separate face with columns specifying classification, bounding box, and segmentation mask data. The original dataset subdivided images into "test-challenge", "test-dev", "train", and "val" directories. Our modifications further subdivided each category into directories containing 50 images. Lastly, 288 images were removed from the dataset for lacking annotations [33].

Face Detection

Face detection is a crucial preprocessing stage in the workflow of our proposed solutions. Precisely identifying and cropping faces guarantees that later models work on the regions of interest, specifically the individual faces, instead of entire images, which frequently have several faces or unrelated background material. This focused approach improves computational efficiency, enhances model accuracy, and minimizes false positives. In particular, the Random Forest method benefits from this step, as it requires well-aligned inputs and explicit feature vectors for optimal classification performance. Deep learning models such as CNNs and Xception can work with uncropped images, but dedicated face detection generally strengthens their robustness and consistency, especially in mixed, high-resolution datasets.

The Haar Cascade classifier was chosen for this preprocessing step. This technique leverages Haar-like features, simple edge and line patterns measured across windows of the image, and applies a boosting approach (AdaBoost) to combine these weak features into a strong classifier [34]. The cascade architecture allows for rapid rejection of non-face regions and efficient focusing of computation on probable faces. Despite advances in deep learning, the Haar Cascade remains a strong choice for real-world, high-resolution images due to its speed and effectiveness.

In our system, the Haar Cascade detector preprocesses each image and outputs annotations containing information about detected faces only. These face-focused annotations are then passed to the classification models, ensuring that the models operate exclusively on relevant facial data.

Demonstration

To demonstrate the feasibility of our DeepFake detector, a website will be developed. This application will be deployed for public use and hosted on the ISIP site. Within this demonstration, the face detection method will be utilized, in addition to all the machine learning models implemented.

Solution 1: Random Forest

Discrete Cosine Transform

A Discrete Cosine Transform (DCT) is a mathematical transformation similar in concept to the more familiar Fourier transform. A DCT converts a series of data points into a sum of cosine functions, each with different frequencies [35]. While similar in concept to a Fourier transform, a DCT uses only real numbers. DCT has many uses, but it is widely used in image and video compression because of its efficiency in representing data [36]. For this solution, we are making use of the DCT to create our input data for the Random Forest Model. For this solution, we feed the RGB values of a given image through a DCT and store the resulting values (cosine coefficients). Each color channel has its own feature vector, and these vectors are what we use as input for the Random Forest Model.

We chose to use the DCT for this solution so that we could be efficient with our data. Originally, we were extracting the RGB values and storing them into csv files to use as input data. These files were very large and were slow to generate. Thus, with advice from our advisor, we decided to perform a DCT on those RGB values and store the first 100 coefficients instead. This approach allows us to represent the image's qualities with only 100 values as opposed to three values per pixel in the image, which would be approximately 196,608 values for a 256x256 image. This is for one image, and our dataset partitions contain hundreds of images of varying sizes. Using the DCT allows us to more efficiently represent our data.

Random Forest Model

Our first classification model will utilize the extracted DCT values to train a Random Forest model. The random forest is a machine learning algorithm that classifies by combining the output of multiple decision trees. This is an ensemble learning method, which combines multiple models to achieve higher accuracy, rather than a single model. The main component of a Random Forest model are the decision trees, each tree is built by the splitting of data,

based on its structure, leading to the creation of the leaves. Each tree in the forest is trained on a random subset of training data, which is bootstrapped, meaning that some datapoints may appear within multiple trees learning data. Figure 1 visualizes the structure of a Random Forest algorithm [38].

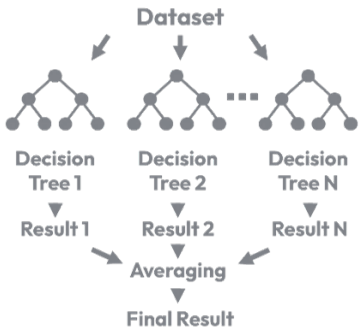

Figure 1: Random Forest Visualization

When designing the Random Forest model, there are several hyperparameters that may be alternated to tune the model's accuracy and generalization. The number of estimators, or number of trees within the forest. The maximum depth of each tree, the minimum number of samples needed to split a trees node, and the minimum number of samples required within a leaf node. These hyperparameters are summarized in Table 3 [39].

Table 3: Random Forest Hyperparameter Considerations

Hyperparameter (Python Variable)	Purpose	Typical Test Values
N estimators (n_esimators)	Number of trees in a forest More trees, higher accuracy, increased computational power	50, 100, 200
Maximum depth (max_depth)	Maximum depth of each tree, controlling model complexity and preventing overfitting	5, 10, 20, None
Minimum Samples Split (min_samples_split)	minimum number of samples required to split a node, more values result in simple trees	2, 5, 10,
Minimum Samples Leaf (min_samples_leaf)	Minimum numbers of samples needed for a leaf node to prevent overfitting	1, 2, 5, 10
Maximum Features (max_features)	Number of features to consider when looking for best split	'sqrt', 'log2', 0.2–1.0 fraction of total features
Bootstrap (bootstrap)	Using bootstrap samples	True/False
Criterion (criterion)	Function to measure quality of split	'gini', 'entropy'
Random State (random_state)	Seed for reproducibility	Any value

In addition to alternating the hyperparameters of the model, the DCT values may be alternated as well. As the upper left corner of the DCT quantifies low frequencies, while the lower right corner summarizes the high frequency values. High frequency values describe the sudden change between two textures, and low frequency is used to describe gradients. Certain portions of these values may be selected to train on, as shown in Figure 2. Code 1 demonstrates how the Random Forest models is initiated, trained and evaluated [36].

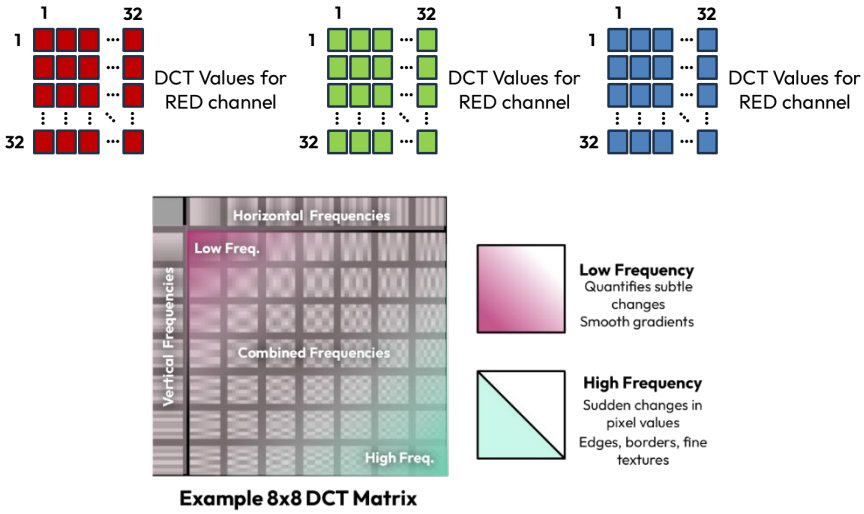

Figure 2: DCT Visualization

Code 1: Example CNN Implementation

```
rf = RandomForestClassifier( # initialize random forest with hyperparameters
    n_estimators=100,        # number of trees
    max_depth=None,          # maximum depth of each tree
    min_samples_split=2,     # minimum samples required to split a node
    min_samples_leaf=1,      # minimum samples required at a leaf
    max_features='auto',     # number of features considered for best split
    bootstrap=True,          # use bootstrap samples when building trees
    criterion='gini',        # function to measure quality of split
    random_state=42)         # random seed for reproducibility
rf.fit(X_train, y_train)     # train the model
y_pred = rf.predict(X_test)  # predict on test set
```

Solution 2: Convolutional Neural Network (CNN)

Parallel Dataset

This new parallel dataset is essentially the same as the dataset we used for Solution 1. We will be extracting the faces from the images in our original dataset and store them as their own image. We are making this modification to the dataset so that we can train the CNN on one image at a time in hopes that this will allow the model to learn more effectively.

CNN

Convolutional Neural Networks is a type of deep learning model that processes grid like data, such as images and is commonly used within image classification. CNNs are made up of several layers including the convolutional, pooling and fully connected layers. The convolutional layers apply filters, also known as kernels, to the input image to allow the detection of various features, such as the edges, textures and patterns. These features would then be learned within the training process [40].

The pooling layers would reduce the spatial dimension of the image, helping retain the most important features, reducing computational complexity. The fully connected layers would take in the high level features that were extracted by the previous layers and use them to make a classification. When trained, these layers would learn the patterns differentiated between real and generated content. Figure 3 visualizes the layers of the CNN, and how they are utilized to classify an image.

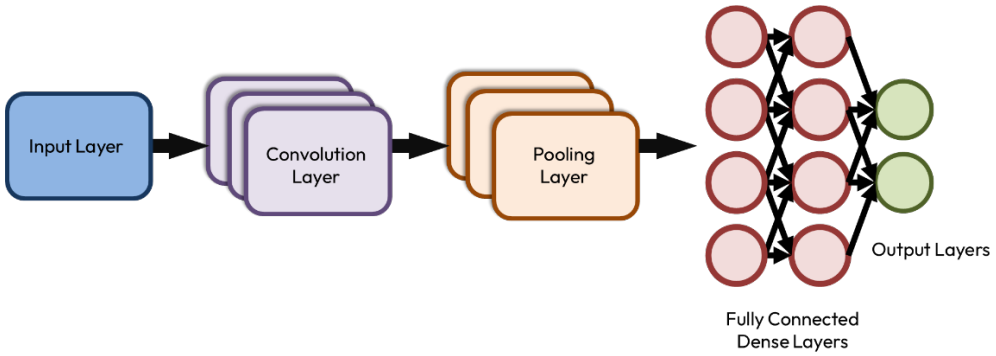

Figure 3: Layers of CNN

As the CNN processes images, not matrix features of vectors, a parallel dataset of just the extracted faces would need to be utilized for training and evaluation. Once a CNN model is initiated, the convolutional layers would be added, the filter number and the size across the image would need to be defined, as well as the actuation model. These layers would be able to learn the low-level features such as the edges of the image. A pooling layer is then introduced which would reduce the spatial dimension of the data and the computational cost to process [41]. Table 4 provides a brief explanation of the purpose of each layer and their parameters.

Table 4: CNN Layers

Layer	Description	Parameters
Input Layer	Takes in image information	Input size of image
Convolutional Layer	Feature Extraction using kernels (filters)	Number of filters, Filter size (pixel x pixel), Stride, Padding

Activation Layer	introduces non-linearity	Activation function choice
Pooling Layer	Reduces spatial dimensions	Pooling type (max, average, global), pool size (n×n), stride
Fully Connected Layer	Connects features to output classes	Number of neurons, activation function
Output Layer	Produces final prediction	Number of classes (2: Real/Fale), activation function

The 2-dimentional features are then flattened into a 1-dimenstional vector and is passed to the fully connected layers. The output layer is made up of 2 neurons, where would represent 0-real or 1-fake, initiated with an activation function, which would output a probability distribution for each class, allowing the model to classify the image [41]. Code 2 demonstrates an example of how a CNN and its layers are initiated.

Code 2: Example CNN Implementation

```
X_train=X_train/255 # normalizing the pixel values
X_test=X_test/255   # normalizing the pixel values
model=Sequential()  # defining model
model.add(Conv2D(32,(3,3),activation='relu',input_shape=(28,28,1))) # adding
convolution layer
model.add(MaxPool2D(2,2)) # adding pooling layer
model.add(Flatten())      # adding fully connected layer
model.add(Dense(100,activation='relu'))
model.add(Dense(10,activation='softmax')) # adding output layer
model.compile(loss='sparse_categorical_crossentropy',optimizer='adam',metrics
=['accuracy']) # compiling the model
```

Once the model is implemented, it would be trained on the data, and the number of Epochs would be varied. Epochs are passthrough on the entire training data, where the model is able to learn and update its internal weights based on the error within tis predictions, allowing the model to gradually improve its accuracy. Table 5 provides a brief explanation of the parameters found in each layer and their purpose [40].

Table 5: CNN Parameter Definition

Layer	Parameter	Definition
Convolutional Layer	Stride	Step size taken by the filter has it moves across the input image.
Convolutional Layer	Padding	Adds extra pixels around the input of the image prior to convolution, helping control the size of output.
Pooling Layer	Pooling Type	Reduces the spatial size of the feature maps while keeping important information.
Pooling Layer	Pooling size	Dimension of the pooling region.
Fully connected Layer	Activation function	Adds nonlinearity and allows the network model to form complicated decision boundaries.
Output Layer	Activation Function	Transforms probability calculations to classification.

Solution 3: Xception Model

Model

Xception is a deep learning model used for image classification, and stands for Extreme Inception, proposed in 2017. While normal CNNs utilize convolutions to learn image patterns, Xception models look at the spatial patterns within each channel individually and combine the information within the color channels. This process is called depth wise separable convolution and makes the network more efficient due to the usage of less parameters and are oftentimes more accurate than traditional CNNs. Xception models have become a benchmark for DeepFake detection, due to their ability to detect subtle artifacts such as blending errors, color mismatches and or texture inconsistencies. This solution will utilize the same parallel dataset developed for Solution 2 for training, development and evaluation [42].

The Xception model is made of three portions, an entry flow, a middle flow, and an exit flow. The entry flow is the first layer that processes the input image, to extract low-level features such as edges, colors and textures. The middle flow is considered the core of the network, where its purpose is to extract deeper and more abstract features. The exit flow is the final layer prior to classification, where the features are compressed. The exit flow considers all the features learned and provides a classification of real or fake. Between each convolutional layer and flow, a residual skip connection is included, this adds the original input to the layers output, preventing the degradation of information as its passed through the system [44]. Figure 4 demonstrates how the model layers are organized. Table 5 demonstrates the various parameters of the Xception model layers and which portions can be adjusted [43].

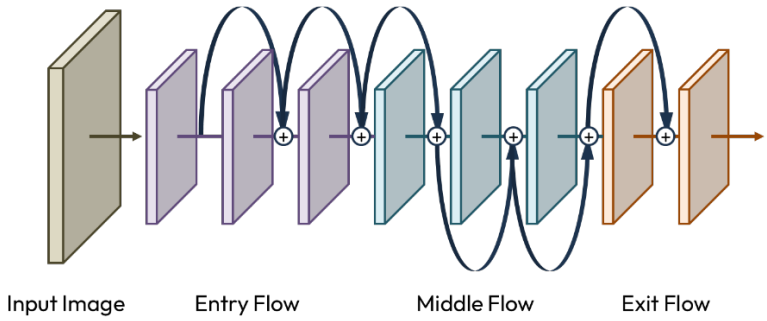

Figure 4: Xception Model Flow

Table 6: Parameters of Xception Model

Layer	Purpose	Parameter
Entry Flow	Extracts low level features	Filter size, Filters, Stride, Padding, Activation
Middle Flow	Learns higher level features	Filter size, Filters, Stride, Padding, Activation

Exit Flow	Produces classification results	Filter size, Filters, Stride, Padding, Activation, Global Average Pooling, Dense Units

Next Step

We plan to begin our evaluation using the Random Forest model because prior research has demonstrated its effectiveness as a baseline for DeepFake detection, particularly when combined with handcrafted feature extraction methods such as Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP) [45], [46]. Random Forest is also quick to train, computationally efficient, and relatively easy to interpret compared to deep learning models, making it an ideal first choice for establishing baseline performance. After validating the Random Forest results, we will transition to a Convolutional Neural Network (CNN), which is specifically designed for image analysis tasks and better suited to learning spatial and texture-based features directly from the data. Finally, we will evaluate the Xception model as our third approach due to its demonstrated state-of-the-art performance in DeepFake detection benchmarks [47]. Although our initial focus will be on Random Forest, as optimization continues, we will move on to develop and test the CNN and Xception models. Our long-term objective is to make all three models accessible for users to evaluate and contrast, offering a balance between computational demands, speed, and accuracy.

Appendix

Table 7: DeepFake Datasets

Name	Real Photos	Fake Photos	Media Form
DFFD	58,703	240,336	Images
ForgeryNet	1,438,201	1,457,861	Images
Generated Photos	Not Available	10,000	Images
CelebA	Not Available	202,599	Images
FaceForensics	Not Available	500,000 frames containing faces from 1004 videos	Video Frames
Celeb-DF	590 original videos	5639 corresponding DeepFake videos	Videos
OpenForensics: Multi-Face Forgery Detection And Segmentation In-The-Wild Dataset	45473	70325	Photos
Deepfake Detection Challenge Dataset	Not Available	100,000 videos	Videos
Flickr-Faces-HQ	70,000		Photos
Deepfake Synthetic-20K Dataset	Not Available	20K synthetically generated face images	Photos
Individualized Deepfake Detection Dataset	23k authentic	22k deepfake	Photos
UADFV	49 videos	49 videos	Videos

Abbreviations Used

- Machine Learning (ML)
- Deep Learning (DL)
- General Adversarial Network (GAN)
- Convolutional Neural Network (CNN)
- User interface (UI)

Citations

[1] F. Juefei-Xu, R. Wang, Y. Huang, Q. Guo, L. Ma, and Y. Liu, "Countering Malicious DeepFakes: Survey, Battleground, and Horizon," *arXiv.org*, 2021. https://arxiv.org/abs/2103.00218

[2] Tenali Anusha and A. Srinagesh, "Deepfake Video Detection: A Comprehensive Survey of Advanced Machine Learning and Deep Learning Techniques to Combat Synthetic Video Manipulation," *IEEE Xplore*, pp. 1033–1041, Jan. 2025, doi: https://doi.org/10.1109/icmsci62561.2025.10894187.

[3] F.-A. Croitoru *et al.*, "Deepfake Media Generation and Detection in the Generative AI Era: A Survey and Outlook," *Arxiv.org*, 2024. https://arxiv.org/html/2411.19537 (accessed Aug. 31, 2025).

[4] D. Gamage, P. Ghasiya, V. Bonagiri, M. Whiting, and K. Sasahara, "Are Deepfakes Concerning? Analyzing Conversations of Deepfakes on Reddit and Exploring Societal Implications," *Are Deepfakes Concerning? Analyzing Conversations of Deepfakes on Reddit and Exploring Societal Implications*, 2022, doi: https://doi.org/10.1145/3491102.3517446.

[5] T. Say, M. Alkan, and A. Kocak, "Advancing GAN Deepfake Detection: Mixed Datasets and Comprehensive Artifact Analysis," *Applied Sciences*, vol. 15, no. 2, pp. 923–923, Jan. 2025, doi: https://doi.org/10.3390/app15020923.

[6] A. Dehghani and H. Saberi, "Generating and Detecting Various Types of Fake Image and Audio Content: A Review of Modern Deep Learning Technologies and Tools," *arXiv.org*, 2025. https://arxiv.org/abs/2501.06227

[7] National Security Agency, "NSA, U.S. Federal Agencies Advise on Deepfake Threats," *National Security Agency/Central Security Service*, Sep. 12, 2023. https://www.nsa.gov/Press-Room/Press-Releases-Statements/Press-Release-View/Article/3523329/nsa-us-federal-agencies-advise-on-deepfake-threats/ (accessed Aug. 31, 2025).

[8] J. SHAPIRO, *Act No. 35*. 2025. Available: https://www.palegis.us/statutes/unconsolidated/law-information?sessYr=2025&sessInd=0&actNum=35

[9] H. Song, S. Huang, Y. Dong, and W.-W. Tu, "Robustness and Generalizability of Deepfake Detection: A Study with Diffusion Models," *arXiv.org*, 2023. https://arxiv.org/abs/2309.02218

[10] S. M. Qureshi, A. Saeed, S. H. Almotiri, F. Ahmad, and M. A. Al Ghamdi, "Deepfake forensics: a survey of digital forensic methods for multimodal deepfake identification on social media," *PeerJ Computer Science*, vol. 10, p. e2037, May 2024, doi: https://doi.org/10.7717/peerj-cs.2037.

[11] C. Bhattacharyya, H. Wang, F. Zhang, S. Kim, and X. Zhu, "Diffusion Deepfake," *arXiv.org*, 2024. https://arxiv.org/abs/2404.01579 (accessed Aug. 31, 2025).

[12] U. R. Acharya, Y. Zhang, J. Wang, and W. Ding, "Convolutional Neural Network - an overview | ScienceDirect Topics," *www.sciencedirect.com*. https://www.sciencedirect.com/topics/engineering/convolutional-neural-network

[13] Google PAIR, "People + AI Guidebook," *Withgoogle.com*, 2017. https://pair.withgoogle.com/chapter/explainability-trust/

[14] A. Rajesh, S. Tiwari, Vivek Gotecha, and N. Y. Kapadnis, "Deepfake Detection using CNN," *ResearchGate*, Mar. 2024. https://www.researchgate.net/publication/378609583_Deepfake_Detection_using_CNN

[15] J. Laffier and A. Rehman, "Deepfakes and Harm to Women," *Journal of Digital Life and Learning*, vol. 3, no. 1, pp. 1–21, Jun. 2023, doi: https://doi.org/10.51357/jdll.v3i1.218.

[16] S. Ahmed, M. Masood, A. Wei, and K. Ichikawa, "False failures, real distrust: the impact of an infrastructure failure deepfake on government trust," *Frontiers in Psychology*, vol. 16, May 2025, doi: https://doi.org/10.3389/fpsyg.2025.1574840.

[17] Chai, L., Bau, D., Lim, SN., Isola, P. (2020). What Makes Fake Images Detectable? Understanding Properties that Generalize. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science(), vol 12371. Springer, Cham. https://doi.org/10.1007/978-3-030-58574-7_7

[18] Google, "Classification: Accuracy, recall, precision, and related metrics," *Google for Developers*, 2024. https://developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall

[19] M. S. Saealal, M. Z. Ibrahim, M. I. Shapiai, and N. Fadilah, "In-The-Wild Deepfake Detection using Adaptable CNN Models with Visual Class Activation Mapping for Improved Accuracy," 2023 5th International Conference on Computer Communication and the Internet (ICCCI), pp. 9–14, Jun. 2023, doi: https://doi.org/10.1109/ICCCI59363.2023.10210096.

[20] J. Picone, "Neuronix: A Low-Cost High Performance Cluster," *Piconepress.com*, 2018. https://isip.piconepress.com/projects/neuronix/html/about.shtml (accessed Sep. 22, 2025).

[21] "DEEPCHECKS GLOSSARY," [Online]. Available: https://www.deepchecks.com/glossary/ml-scalability/. [Accessed 15 September 2025].

[22] A. Jha, "ML Model Interpretation Tools - What, Why, and How to Interpret," *neptune.ai*, May 08, 2021. https://neptune.ai/blog/ml-model-interpretation-tools

[23] W3C, "Web Content Accessibility Guidelines (WCAG) 2.1," *W3.org*, May 06, 2025. https://www.w3.org/TR/WCAG21/

[24] W3C, *Web Content Accessibility Guidelines (WCAG) 2.1*. Jun. 2018. [Online]. Available: https://www.w3.org/TR/WCAG21/

[25] M. Y. Arif, "LinkedIn," 27 June 2024. [Online]. Available: https://www.linkedin.com/pulse/importance-code-readability-maintainability-software-development-s9ckf/. [Accessed 14 September 2025].

[26] ISO, "ISO/IEC/IEEE 12207:2017," *ISO*, 2017. https://www.iso.org/standard/63712.html
[27] F. Doshi-Velez and B. Kim, "Towards A Rigorous Science of Interpretable Machine Learning," *arXiv:1702.08608 [cs, stat]*, vol. 2, no. 2, Mar. 2017, Available: https://arxiv.org/abs/1702.08608
[28] "IEEE Standard for Algorithmic Bias Considerations," *IEEE*, Dec. 2024, doi: https://doi.org/10.1109/ieeestd.2025.10851955.
[29] OWASP, "OWASP Top 10: 2021," *OWASP*, 2021. https://owasp.org/Top10/
[30] International Organization for Standardization, "ISO/IEC 27001 standard – information security management systems," *ISO*, 2022. https://www.iso.org/standard/27001
[31] S. Baker, "ISO 27001 Information Classification and Handling Policy Beginner's Guide," 17 July 2025. [Online]. Available: https://hightable.io/information-classification-and-handling-policy/. [Accessed 20 September 2025].
[32] J. Jayan, "Importance of Ethical Data Collection," PromptCLoud, 1 August 2024. [Online]. Available: https://www.promptcloud.com/blog/importance-of-ethical-data-collection/. [Accessed 15 September 2025].
[33] T.-N. Le, H. Nguyen, J. Yamagishi, and I. Echizen, "OpenForensics: Large-Scale Challenging Dataset For Multi-Face Forgery Detection And Segmentation In-The-Wild." Available: https://openaccess.thecvf.com/content/ICCV2021/papers/Le_OpenForensics_Large-Scale_Challenging_Dataset_for_Multi-Face_Forgery_Detection_and_Segmentation_ICCV_2021_paper.pdf
[34] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Kauai, HI, USA, 2001, pp. I-511–I-518. doi:10.1109/CVPR.2001.990517.
[35] "Discrete Cosine Transform - MATLAB & Simulink," *www.mathworks.com*. https://www.mathworks.com/help/images/discrete-cosine-transform.html
[36] K. R. Vijayanagar, "Discrete Cosine Transform in Video Compression - Explain Like I'm 5 - OTTVerse," *ottverse.com*, Aug. 20, 2020. https://ottverse.com/discrete-cosine-transform-dct-video-compression/
[37] T.-N. Le, H. Nguyen, J. Yamagishi, and I. Echizen, "OpenForensics: Large-Scale Challenging Dataset For Multi-Face Forgery Detection And Segmentation In-The-Wild." Available: https://openaccess.thecvf.com/content/ICCV2021/papers/Le_OpenForensics_Large-Scale_Challenging_Dataset_for_Multi-Face_Forgery_Detection_and_Segmentation_ICCV_2021_paper.pdf
[38] IBM, "What is random forest?," *Ibm.com*, Oct. 20, 2021. https://www.ibm.com/think/topics/random-forest
[39] Scikit-Learn, "sklearn.ensemble.RandomForestClassifier — scikit-learn 0.20.3 Documentation," *Scikit-learn.org*, 2025. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
[40] IBM, "Convolutional Neural Networks," *Ibm.com*, Oct. 06, 2021. https://www.ibm.com/think/topics/convolutional-neural-networks

[41] Jorgecardete, "Convolutional Neural Networks: A Comprehensive Guide," *The Deep Hub*, Feb. 17, 2024. https://medium.com/thedeephub/convolutional-neural-networks-a-comprehensive-guide-5cc0b5eae175

[42] G. Boesch, "Xception Model: Analyzing Depthwise Separable Convolutions - viso.ai," *viso.ai*, May 16, 2024. https://viso.ai/deep-learning/xception-model/

[43] F. Chollet, "Xception: Deep Learning with Depthwise Separable Convolutions," *arXiv.org*, 2016. https://arxiv.org/abs/1610.02357

[44] M. Prokofieva, "Attention in Transformers: residual connection layer, a shortcut that makes it work," *Medium*, Oct. 29, 2024. https://medium.com/@mariaprokofieva/attention-in-transformers-residual-connection-layer-a-shortcut-that-makes-it-work-165b52566167

[45] A. Farid, "Creating and detecting deep fakes," *IEEE Signal Processing Magazine*, vol. 36, no. 1, pp. 18–27, Jan. 2019. doi:10.1109/MSP.2018.2877581

[46] Y. Zhang, X. Luo, and H. Jiang, "Lightweight deepfake detection based on multi-feature fusion and Random Forest," *arXiv preprint* arXiv:2502.11763, 2025. [Online]. Available: https://arxiv.org/abs/2502.11763

[47] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 1251–1258. doi:10.1109/CVPR.2017.195.