Creating a Model to Distinguish Between Real and AI-Generated Pictures

Team Members: Keira James, Farhikhta Farzan, Tesneem Essa

Group Members:

Since the group has just formed, we are still in the process of finding positions. (Positions subject to change).

Group leader Unknown

- Keira James | Brooklyn College
  - Primary role Feature engineer
  - Secondary role Model Developer

- Farhikhta Farzan | Queens College
  - Primary role Unknown
  - Secondary role Unknown

- Tesneem Essa | Hunter College
  - Primary role Unknown
  - Secondary role Unknown

Mentor:

- Harpreet Gaur | CTP Mentor
  - Harpreet Gaur is our instructor in CTP Data Science class, as well as our mentor in this CTP Data Science project.
  - harpreet.gaur97@gmail.com | https://www.linkedin.com/in/harpreetgaur?

Project Outline:

The goal of this project is to develop a deep learning model that can accurately distinguish between real images and AI-generated images. We will collect datasets of real images and fake images. The data will be preprocessed, normalized, and augmented to enhance training. Using TensorFlow and Keras, we will design a Convolutional Neural Network (CNN) for classification, and validating performance through a confusion matrix. Finally, the project will include documentation of the process, findings, and suggestions for future improvements.

Practical Application:

Our project is crucial in today's world,where AI-generated content is increasingly prevalent. This model can be used by social media platforms, news organizations, and even everyday people to verify the authenticity of images, helping to fight against misinformation and ensure the integrity of visual media. Here are 3 use cases:

Case 1: Single Image Classification:

A user wants to classify a single image to determine if it is real or AI-generated. After logging in, the user uploads an image through a simple file uploader interface. The system then preprocesses the image before passing it to the CNN model for prediction. Once the model outputs a label, the user sees the uploaded image displayed alongside the classification result (e.g., “Real” or “AI-generated”). The user can then choose to classify another image.

Case 2: Batch Image Analysis:
A researcher aims to analyze multiple images to see how many are real versus AI-generated. They upload an entire folder containing various images to the batch analysis feature. The system processes each image sequentially, applying the same preprocessing and prediction steps used in the single image classification. After processing, the system could outputs the amount that are real and the amount that are AI generated.

Case 3: Automated Content Moderation:
A social media platform uses the model to moderate images uploaded by users. When users upload images to the platform, the system checks each image using the trained CNN to determine if it is real or AI-generated. If the model flags an image as AI-generated, it triggers a review process for potentially misleading content. This functionality helps the platform maintain content integrity, ensuring that users are not misled by manipulated or AI-generated images.

Overview of Steps/ Tasks to be Completed for Implementation:

1. **Find Dataset (70% training, 15% validation, 15% testing):**
   - Find a training dataset that has images that are created by AI and images that are not AI generated; this will be used to train our model.
   - Find a validation set to test how well the model is learning and to tune it’s settings (hyperparameters like learing rate and layers); (validation set helps check if the model is really learning the right patterns or just memorizing the training data. If the model performs well on the training set but poorly on the validation set, it might be overfitting)
   - Find a test dataset to see how well the model performs on completely new data.
2. **Prepare Dataset:**
   - Make sure that images are all of the same dimension, orientation, brightness level, etc.
3. **Choose Model Architecture:**
   - Deciding a model architecture that will best work with our project.
4. **Implement the Model:**
   - Learn about relevant machine learning libraries that can be used to create the model we decide.
5. **Train the Model:**
   - Use dataset to train the model. Monitor accuracy and loss to prevent overfitting. Use validation set to tune hyperparameters and assess performance.

6. **Evaluate the Model:**
   - Evaluate the model using test dataset to measure its accuracy, precision, recall, and F1 score (a way to measure how well a model is performing);
   - Implement a confusion matrix to help visualize how well the model is doing by showing the number of correct and incorrect predictions it makes.
7. **Test, Fine Tune, and Optimize:**
   - Hyperparameter Tuning: Experiment with learning rates, batch sizes, and number of epochs.

Roles based on Task Overview: (Subject to Change)

Keira James	- Choose model architecture and learn about how to implement it.
Farhikhta Farzan	- Find additional datasets or possibly partition the one into 3 sets.
Tesneem Essa	

Continuation/ More Detailed Task Implementation:

Record/Jot down a summary of what you did/ found here

1.Find Dataset (training, validation, testing):

- https://www.kaggle.com/datasets/birdy654/cifake-real-and-ai-generated-synthetic-images
  Dataset titled "CIFAKE: Real and AI-Generated Synthetic"provides a set of 60,000 real images and AI generated images to use in the training of our model.
  Sources:
  Krizhevsky, A., & Hinton, G. (2009). Learning multiple layers of features from tiny images. Bird, J.J. and Lotfi, A., 2024. CIFAKE: Image Classification and Explainable Identification of AI-Generated Synthetic Images. IEEE Access.

********Question: Can some of the data from this overall set be sectioned off to use for validation and test set?*******

2.Prepare/ Analyze Dataset:

- Check that the photos are unique and diverse.
- Check the dimensions.
- Check which are real and fake.

3.Choose Model Architecture:

Model Architecture: the blueprint for building a machine learning model. It describes how the model is structured and how it processes data to make predictions. It includes layers, neurons, activation functions, loss functions, and optimizers. The best MAs for our project would be:

- *Convolutional Neural Networks*: used primarily for image processing. Examples include **VGG, ResNet,** or **custom CNNs**.
- ~~*Generative Adversarial Networks (GANs)*: Comprises two models: a generator and a discriminator, which are trained simultaneously. The generator creates fake data, while the discriminator tries to distinguish between real and fake data.~~

4. Implement the MA using library:

Implement the Model: Deep learning libraries are tools that make creating a project like this simplier/easier. (Think of MA as the blueprint and deep learning libraries as the tools used to build the model according to the blueprint.) Good deep learning libraries would be:

- **TensorFlow**: a popular library created by Google; it helps with building all kinds of smart models.
- **Keras**: a user-friendly library that works on top of TensorFlow

5. Train Model:

- Begin training model

6. Evaluate the Model:

- Troubleshoot and document anything; report data to teammate to represent via charts and tables on Tableau

7. Test, Fine Tune, Optimize:

- Hyperparameter tuning will occur here

Design Documentation - CTP Project Design Document
Version Control Repository - https://github.com/KeiraJames/CTP-Project-2024
Project Management Board - https://www.notion.so/109d7a94666680359908e166f9320626?v=d61cce51338e479583e42a8bc66425d0&pvs=4

Version Control System: https://github.com/KeiraJames/CTP-Project-2024

- We will use GtiHub for our project. We will branch individual repositories. On a weekly basis, we would set up tasks and either push code, data visualization, or documentation..
- We will discuss on Slack throughout the semester, communicating and discussing problems regarding our project in our Slack channel.

- On every Sunday (Subject to change), we would share our progress and report them to our mentor.

Project Management Board:

https://www.notion.so/109d7a94666680359908e166f9320626?v=d61cce51338e479583e42a8bc66425d0&pvs=4

- Notion will be the platform for our project management board due to its friendly interface and various accessible functionality.

Visualization:

- We will use Tableau for charting and visualization, allowing us to create interactive and insightful visual representations of the data and model performance metrics. Tableau's features will allow us to analyze errors, understand classification results, and effectively communicate findings via dashboards.
- For the development of the application's API and user interface, we will use Streamlit. Streamlit allows users to upload images, view predictions, and explore results.

Role Descriptions:

Primary Role Description

- Data collector and cleaner
    - The primary role of a data collector is to gather data from various sources. We will not be scraping data together ourselves but rather finding csv files or similar sources that best suit our needs via Kaggle or other platforms. This person will section the data into training, validation, and testing; they will also focus on preparing and cleaning the collected data to make it suitable for analysis. This involves identifying and rectifying issues or inconsistencies in the data.
- Feature engineer and Model developer
    - Feature engineering involves transforming raw data into a format that can be effectively used by machine learning algorithms. It aims to highlight relevant information and improve the performance of models. This role is also responsible for building, training, and optimizing machine learning models based on the prepared dataset. This involves selecting appropriate algorithms, tuning hyperparameters, and ensuring the model's robustness.

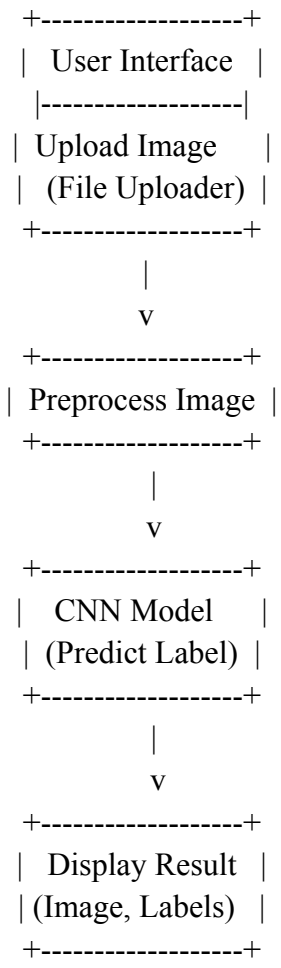
Secondary Role Description

- ~~Model Integration and Testing~~
    - ~~Developing APIs (Application Programming Interfaces) or other interfaces to connect the model with other systems.~~
    - ~~Integrating the model into the existing software infrastructure, whether it's a web application, mobile app, or backend service.~~

    - ~~Collaborating with software developers to embed the model into the production environment.~~
    - ~~Ensuring that the model works smoothly with other components and systems.~~
    - ~~Addressing compatibility issues and dependencies.~~
- Model Interpretation/Visualization and Reporting
    - Visualizing model performance metrics on Tableau, such as precision-recall curves, ROC curves, and confusion matrices.
    - Generating feature importance plots to highlight the impact of different variables on model predictions.
    - Create visualizations for loss and accuracy over epochs, and generate a confusion matrix.
    - Create interactive web application via Streamlit depicting project.
- Model Developer
    - Design and implement the CNN model architecture.
    - Train the model, tune hyperparameters, and monitor performance.
    - Analyze model results and generate performance metrics.
    - Write code for the CNN architecture and compile the model.
    - Train the model using the training set and validate using the validation set.
- ~~Deployment~~
    - ~~Establishing the necessary infrastructure to host and run the model, including servers, databases, and networking configurations.~~
    - ~~Creating a deployable package of the trained model, which may include the model file, dependencies, and any preprocessing steps.~~
    - ~~Developing APIs (Application Programming Interfaces) or endpoints that allow other software applications to send data to the model and receive predictions in return.~~
    - ~~Implementing version control for models to facilitate updates and rollback procedures in case of issues with a new model version.~~
- Documentation and Documentation Review
    - Documenting the specifics of the machine learning model(s) developed, including the choice of algorithms, hyperparameters, and any considerations for model interpretation. This section may also include information on model training, validation, and evaluation.
    - Providing comments and explanations within the codebase to make it more understandable and maintainable. This includes documenting functions, classes, and complex logic.
    - Summarizing the results of model evaluation, including performance metrics, comparisons with baselines, and insights gained from the evaluation process.

- Offering a high-level overview of the entire project, including its goals, scope, and the problem it aims to address. This section may also include information on the project's stakeholders and contributors.
- Clearly stating any assumptions made during the project, as well as the limitations of the data, models, and findings.
- Prepare slides or a report for presenting the findings for Demo Night.

Diagram of how we expect our model to flow:



Description

- A user wants to classify a single image to determine whether it is real or AI-generated.

Input

- An image file uploaded by the user (e.g., a JPEG of a landscape).

Process

- User uploads an image.
- The system preprocesses the image (resizing, normalization).
- The CNN model predicts the label of the image.
- The result is displayed to the user.

Expected Output

- Display of the uploaded image.
- Classification result (e.g., “Real” or “AI-generated”).
- Option to classify another image.

Anticipated Challenges and Solutions

Challenges:

- Data Collection
    - Datasets with poor data quality including missing values, outliers, or inaccuracies, can significantly impact model performance.
- Overfitting
    - When a model learns the training data too well, capturing noise rather than the underlying patterns, leading to poor generalization on new data.
- Uninterpretable Models
    - Complex models like deep neural networks may lack interpretability, making it challenging to explain predictions to stakeholders or ensure compliance with regulatory requirements.
- Implementing Model Architecture Wrong
    - Errors in implementing model architecture can lead to suboptimal performance and wasted resources.

Possible Solutions:

- Data collection Solutions
    - Implement data preprocessing steps, including handling missing values appropriately, outlier detection and treatment, and thorough exploratory data analysis to identify and address data quality issues.
- Overfitting Solutions
    - Use techniques such as cross-validation, regularization, and feature engineering to prevent overfitting. Monitoring model performance on validation sets and adjusting model complexity can also help mitigate overfitting.
- Uninterpretable Models Solutions
    - Consider using interpretable models when transparency is crucial.
- Implementing Model Architecture Solutions
    - Start with small prototypes or projects to validate the architecture before full implementation and involve team members in code reviews to catch potential issues early.

Tentative Schedule: Cross off when finished

Goals:

- Push to Github weekly.
- Meet with teammates via Slack or other and stay updated on each other’s progress.
- Let us know what you’re struggling with or if you’re on track.

Date	Week	Tasks to be Complete	Estimated Time to Complete (hrs)
Sep 22, 2024	Week 04	☑ ~~Create Project Design Document~~ ☑ ~~Create Github~~ ☑ ~~Create Project Design Board~~ ☐ Branch github onto individual computers.	6.5
Sep 29, 2023	Week 05	☐ Finalize Project Choice and Design Documentation. ☐ Finalize Dataset Decision ☐ Look into and experiment with Streamlit. ☐ Establish project roles.	
Oct 6, 2023	Week 06	☐ Clean and analyze dataset; partition if needed (training, validation, testing) ☐ Create prototype for chosen Model Architecture and decide on libraries to be used.	
Oct 13, 2023	Week 07	☐ Begin coding (after coding a prototype/ test of model architecture) ☐ Push to Github	

Oct 20, 2023	Week 08	☐ Start documentation; coding comments, github readme files, etc.	
Oct 27, 2023	Week 09	☐ Continue coding, modify and change as needed. Update teammates on progress. ☐ Begin testing model, checking for overfitting problems using validation dataset. ☐ Document any errors/problems that occur **(confusion matrix).** ☐ **Calculate accuracy, precision, recall, and F1 score to assess model performance.**	
Nov 3, 2023	Week 10	☐ Begin testing the model using test dataset OR continue troubleshooting.	
Nov 3, 2023	Week 11	☐ Summarize findings, challenges faced, and the overall performance of the model thus far.	
Nov 10, 2023	Week 12	☐ Start wrapping up coding? ☐ Begin creating API	

Nov 17, 2023	Week 13	☐ **IF CODING IS DONE:** ☐ Finish documentation. ☐ Start creating elevator pitch and presentation (Prep for presentation night). ☐ Check in with mentors. ☐ Possibly implement a GUI if there's time left over.	
Nov 24, 2023	Week 14	- Finalize work	
Dec 1, 2023	Week 15	- **PROJECT DUE THIS WEEK**	