A Report on Major project

**DEEPFAKE DETECTION OF AUDIO MANIPULATIONS**

*SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS*

*FOR THE DEGREE OF*

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER ENGINEERING**

**OF**

**VISHWAKARMA INSTITUTE OF TECHNOLOGY**

**Savitribai Phule Pune University**

*BY*

**Suchit Meshram (GR. No. 161577)**

**Raghav Singhania (GR. No. 161147)**

**Sachin Chaudhari (GR. No. 161593)**

**Rushikesh Khardekar (GR. No. 161450)**

*UNDER THE GUIDANCE OF*

Prof. Dr. Ashwini Shringare



DEPARTMENT OF COMPUTER ENGINEERING

BANSILAL RAMNATH AGARWAL CHARITABLE TRUST’S

VISHWAKARMA INSTITUTE OF TECHNOLOGY

(An Autonomous Institute affiliated to Savitribai Phule Pune University) PUNE - 411037

2019 - 2020

**BANSILAL RAMNATH AGARWAL CHARITABLE TRUST’S**

**VISHWAKARMA INSTITUTE OF TECHNOLOGY**

**(An Autonomous Institute affiliated to Savitribai Phule Pune University)**

**PUNE – 411037**



**C E R T I F I C A T E**

This is to certify that the Major Project titled **DeepFake Detection of Audio Manipulation** submitted by **Suchit Meshram (GR No. 161577), Raghav Singhania (GR No. 161147), Sachin Chaudhari (GR No. 161593) , Rushikesh Khardekar (GR No. 161450)** is in partial fulfillment for the award of Degree of Bachelor of Technology in Computer Engineering of Vishwakarma Institute of Technology, Savitribai Phule Pune University. This project report is a record of bonafide work carried out by him under my guidance during the academic year 2019-20.

**Guide Head of Computer Dept.**

**[Prof. Ashwini Shringare] [Prof. Dr. Sandip Shinde]**

Dept. of Computer Engg. VIT, Pune.

**Sign of External Examiner** **Date:19/09/2020**

**ACKNOWLEDGEMENT**

It gives us immense pleasure and satisfaction in presenting this report on “**DeepFake Detection of Audio Manipulations**”. This report work has opened up new vistas of knowledge for us. We can now justifiably claim that this experience will stand us in good stead in the years to come. There are a large number of people without whose help this Unique learning experience would be a nonstarter.

We wish to express my deep sense of gratitude to our Internal Guide, **Prof. Ashwini Shringare** Computer Engineering, for his valuable guidance and useful suggestions, which helped us in completing this report work, in time. Without his cooperation, it would have been extremely difficult for us to complete this report work.

We would like to take the opportunity to thank **Prof. Ashwini Shringare** for extending his kind cooperation and timely help during our report work, whose guidance was like a driving force behind the success for our seminar.

Words are inadequate in offering our thanks to **Prof. Dr. Sandip Shinde**, Head of the Department, Computer Engineering, who has been a source of inspiration and for his timely guidance in the conduct of this report.

We are grateful to other staff members of the Department for giving important suggestions. We also like to thank Director **Mr. Jalnekar** for giving us an opportunity to make our work a success.

Needless to mention, we thank the management and Lab assistants of the department for their encouragement and cooperation in carrying out the report work.

**Date 19/09/2020**

**Raghav Singhania Rushikesh Kharderkar**

**Gr no- 161147 Gr no - 161450**

**Roll No - 69 Roll No - 27**

[**raghav.singhania16@vit.edu**](mailto:raghav.singhania16@vit.edu) **rushikesh.khardekar16@vit.edu**

**Suchit Meshram Sachin Chaudhari**

**Gr No- 161577 Gr No- 161593**

**Roll No -38 Roll No - 09**

[**suchit.meshram16@vit.edu**](mailto:suchit.meshram16@vit.edu) **sachin.chaudhari16@vit.edu**

Table of Contents

[**APPROVALS SIGNATURE BLOCK 9**](#_37m2jsg)

[**1.**](#_1mrcu09) **CONTEXT 10**

[**2.**](#_46r0co2) **PROBLEM 10**

[**3.**](#_2lwamvv) **SOLUTION 10**

Software Project Synopsis

[**4.**](#_3l18frh) **INTRODUCTION 12**

[4.1](#_206ipza) Purpose 13

[4.2](#_4k668n3) Scope 13

[4.3](#_3ygebqi) Overview 13

[**5.**](#_2dlolyb) **OVERALL DESCRIPTION 16**

[5.1](#_3cqmetx) Product Perspective 16

[Product Position Statement 17](#_1rvwp1q)

[5.1.1](#_1664s55) Memory Constraints 19

[5.1.2](#_3q5sasy) Operations 20

[5.2](#_25b2l0r) Product Functions 20

[5.3](#_kgcv8k) User Characteristics 20

[5.4](#_34g0dwd) Constraints 20

[**6.**](#_1jlao46) **SPECIFIC REQUIREMENTS 20**

[6.1](#_43ky6rz) External Interfaces 20

[6.2](#_2iq8gzs) Functions 21

[6.3](#_xvir7l) Performance Requirements 21

[6.4](#_3hv69ve) Logical Database Requirements 21

[6.5](#_1x0gk37) Design Constraints 21

**Feasibility Study Report**

[**7.**](#_39kk8xu) **INTRODUCTION 23**

[7.1](#_1opuj5n) Purpose 24

[7.2](#_48pi1tg) Methodology 24

[7.3](#_2nusc19) References 24

[**8.**](#_1302m92) **GENERAL INFORMATION 25**

[8.1](#_3mzq4wv) Current Systems and Processes 25

[**9.**](#_319y80a) **RECOMMENDATIONS AND CONCLUSION 26**

**Software Project Plan**

[**10.**](#_40ew0vw) **OVERVIEW 28**

[**11.**](#_2fk6b3p) **GOALS AND SCOPE 28**

[11.1](#_upglbi) Project Goals 28

[11.2](#_3ep43zb) Project Scope 29

[11.2.1](#_4du1wux) Excluded 29

[**12.**](#_2szc72q) **SCHEDULE AND MILESTONES 29**

[**13.**](#_184mhaj) **DELIVERABLES 29**

**System Implementation Document**

[**14.**](#_279ka65) **GENERAL INFORMATION 31**

[14.1](#_meukdy) Version Control 31

[14.2](#_36ei31r) Information Details 31

[**15.**](#_1ljsd9k) **COMPONENT DIAGRAM DESCRIPTION 31**

15.1 Classification

15.2 Definition

15.3 Responsibilities

[**16.**](#_45jfvxd) **DEPLOYMENT DIAGRAM DESCRIPTION 34**

[**17.**](#_zu0gcz) **STATE CHART DIAGRAM OVERVIEW 36**

[17.1](#_3jtnz0s) Object State Description 37

[17.2](#_1yyy98l) States Description 37

[**18.**](#_4iylrwe) **SPECIFIC DESCRIPTION 37**

**System Design Document**

[**19.**](#_1d96cc0) **GENERAL INFORMATION 39**

[19.1](#_3x8tuzt) Version Control 39

[19.2](#_2ce457m) Information Details 39

[**20.**](#_rjefff) **CLASS DIAGRAM DESCRIPTION 39**

[20.1](#_3bj1y38) CRC Cards Overview 39

[**1.**](#_1qoc8b1) **CLASS 40**

[**2.**](#_4anzqyu) **RESPONSIBILITY 40**

[**3.**](#_2pta16n) **COLLABORATOR 40**

[Responsibility 40](#_14ykbeg)

[Collaborator 40](#_3oy7u29)

[20.1.1](#_243i4a2) Relationship Guidelines 40

**System Design Document**

[**21.**](#_338fx5o) **GENERAL INFORMATION 43**

[21.1](#_1idq7dh) Version Control 43

[21.2](#_42ddq1a) Information Details 43

[22.](#_2hio093) Conclusion

[23.](#_wnyagw) Future Scope

**List of figures**

|  |  |  |
| --- | --- | --- |
| **Sr No.** | **Figures** | **Page No** |
| 1 | Slover snippet | 14 |
| 2 | Slover snippet 2 | 15 |
| 3 | Bulletins | 16 |
| 4 | Formula | 16 |
| 5 | User Interface | 17 |
| 6 | Tool window | 19 |
| 7 | State chart | 34 |
| 8 | Relationship guidelines | 40 |
| 9 | Testing Process | 44 |

**ABSTRACT**

Audio Deepfakes, technically known as logical-access voice spoofing techniques, have become an increased threat on voice interfaces due to the recent breakthroughs in speech synthesis and voice conversion technologies. Effectively detecting these attacks is critical to many speech applications including automatic speaker verification systems. As new types of speech synthesis and voice conversion techniques are emerging rapidly, the generalization ability of spoofing countermeasures is becoming an increasingly critical challenge. This paper focuses on overcoming this issue by using large margin cosine loss function (LMCL) and online frequency masking augmentation to force the neural network to learn more robust feature embeddings. We evaluate the performance of the proposed system on the ASVspoof 2019 logical access (LA) dataset. Additionally, we evaluate it on a noisy version of the ASVspoof 2019 dataset using publicly available noises to simulate more realistic scenarios. Finally, we evaluate the proposed system on a copy of the dataset that is logically replayed through the telephony channel to simulate spoofing attacks in the call center scenario.

Bansilal Ramnath Agarwal Charitable Trust’s

**Vishwakarma Institute Of Technology, Pune-37**

**Department Of Computer Engineering**

Project Synopsis

Group No : C-20

Group Members:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Roll No** | **Name** | **Class** | **Contact No** | **Email-ID** |
| 38 | Suchit Meshram | B.Tech | 8484911358 | suchit.meshram16@vit.edu |
| 69 | Raghav Singhania | B.Tech | 9421679454 | raghav.singhania16@vit.edu |
| 09 | Sachin Chaudhari | B.Tech | 7020275492 | sachin.chaudhari16@vit.edu |
| 27 | Rushikesh Khardekar | B.Tech | 7588390289 | rushikesh.khardekar16@vit.edu |

Academic Year : 2019-20

Project Title : DeepFake Detection of Audio Manipulations

Project Area : Machine Learning, Deep Learning0

Internal Guide : Prof. Ashwini Shringare

Name of the External Guide:

Contact No:

**Signature of Internal Guide**

**DeepFake Detection of Audio Manipulations**

**Software Project Synopsis**

***19/09/2020***



**Approvals Signature Block**

|  |  |  |
| --- | --- | --- |
| **Project Responsibility** | **Signature** | **Date** |
| *Project Guide (Internal)* | Prof. Ashwini Shringare | 19/09/2020 |
| *Project Guide (External)* |  |  |
| *Documentation Leader* | Suchit Meshram |  |

**Table of Contents**

[**1.**](#_1ci93xb) **CONTEXT 3**

[**2.**](#_3whwml4) **PROBLEM 3**

[**3.**](#_49x2ik5) **SOLUTION 5**

**Context**

In today's world, technology has gone too far away and with that technology some people are trying to get the disadvantage of other people. They are fraudsters. Today's world is more dependent on virtual conversations (Voice or video) rather than actual face to face conversations. So innocent people can be easily fooled by these fraudsters by adding the voice of known people who are getting frauded by fraudsters. They think that the voice of the person on the other side is known to you and can do any mentally harm or access any private information of anyone.

**Problem**

The publication reported in last week of August 2019 that a UK energy company’s chief executive was tricked into wiring €200,000 (or about $220,000 USD) to a Hungarian supplier because he believed his boss was instructing him to do so. But the energy company’s insurance firm, Euler Hermes Group SA, told the WSJ that a clever AI-equipped fraudster was using deepfake software to mimic the voice of the executive and demand his underling pay him within the hour.

**Solution**

To avoid such incidents, we can propose a system to check whether the voice coming from the other side is original or having manipulations in it.

**DeepFake Detection of Audio Manipulations**

**Software Requirements Specification**

**Table of Contents**

[**1.**](#_3j2qqm3) **INTRODUCTION 4**

[1.1](#_1y810tw) Purpose 4

[1.2](#_qsh70q) Scope 4

[1.3](#_1vsw3ci) Definitions, acronyms, and abbreviations 4

[1.4](#_4i7ojhp) References 4

[1.5](#_30j0zll) Overview 4

[**2.**](#_1fob9te) **OVERALL DESCRIPTION 5**

[Problem Statement 5](#_3znysh7)

[2.1](#_2et92p0) Product Perspective 5

[Product Position Statement 5](#_4fsjm0b)

[2.1.1](#_3dy6vkm) System Interfaces 5

[2.1.2](#_2uxtw84) User Interfaces 6

[2.1.3](#_1a346fx) Hardware Interfaces 6

[2.1.4](#_3u2rp3q) Software Interfaces 6

[2.1.5](#_2981zbj) Communications Interfaces 6

[2.1.6](#_odc9jc) Memory Constraints 6

[2.1.7](#_38czs75) Operations 6

[2.1.8](#_1nia2ey) Site Adaptation Requirements 6

[2.2](#_17dp8vu) Product Functions 7

[2.3](#_3rdcrjn) User Characteristics 7

[2.4](#_26in1rg) Constraints 7

[2.5](#_47hxl2r) Assumptions and Dependencies 7

[2.6](#_2mn7vak) Apportioning of Requirements 7

[**3.**](#_lnxbz9) **SPECIFIC REQUIREMENTS 7**

[3.1](#_35nkun2) External Interfaces 8

[3.2](#_1ksv4uv) Functions 8

[3.3](#_44sinio) Performance Requirements 9

[3.4](#_2jxsxqh) Logical Database Requirements 9

[3.5](#_z337ya) Design Constraints 9

[3.5.1](#_11si5id) Standards Compliance. 9

[3.6](#_3ls5o66) Software System Attributes 10

[3.6.1](#_20xfydz) Reliability 10

[3.6.2](#_4kx3h1s) Availability 10

[3.6.3](#_302dr9l) Security 10

[3.6.4](#_1f7o1he) Portability 10

[3.7](#_3z7bk57) Organizing the Specific Requirements 11

[3.7.1](#_2eclud0) System Mode 11

[3.7.2](#_thw4kt) User Class 11

[3.7.3](#_3dhjn8m) Feature 11

[3.7.4](#_1smtxgf) Stimulus 12

[3.7.5](#_4cmhg48) Response 12

[3.7.6](#_2rrrqc1) Functional Hierarchy 12

[3.8](#_16x20ju) Additional Comments 12

[**4.**](#_3qwpj7n) **SUPPORTING INFORMATION. 12**

[**DOCUMENT CONTROL 13**](#_261ztfg)

[Change History 13](#_l7a3n9)

[Document Storage 13](#_356xmb2)

[Document Owner 13](#_1kc7wiv)

[**APPENDICES 14**](#_44bvf6o)

**INTRODUCTION**

1. Spyder IDE:

Spyder is an open source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with a number of prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy and Cython, as well as other open source software. It is released under the MIT license.

Initially created and developed by Pierre Raybaut in 2009, since 2012 Spyder has been maintained and continuously improved by a team of scientific Python developers and the community.

Spyder is extensible with first- and third-party plugins,includes support for interactive tools for data inspection and embeds Python-specific code quality assurance and introspection instruments, such as Pyflakes, Pylint[7] and Rope. It is available cross-platform through Anaconda, on Windows, on macOS through MacPorts, and on major Linux distributions such as Arch Linux, Debian, Fedora, Gentoo Linux, openSUSE and Ubuntu.

Spyder uses Qt for its GUI, and is designed to use either of the PyQt or PySide Python bindings. QtPy, a thin abstraction layer developed by the Spyder project and later adopted by multiple other packages, provides the flexibility to use either backend.

1. Anaconda

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012. As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, which are both not free.

1. Python 3.0

Required python 3.0 for using the functions of deep learning.

**Purpose**

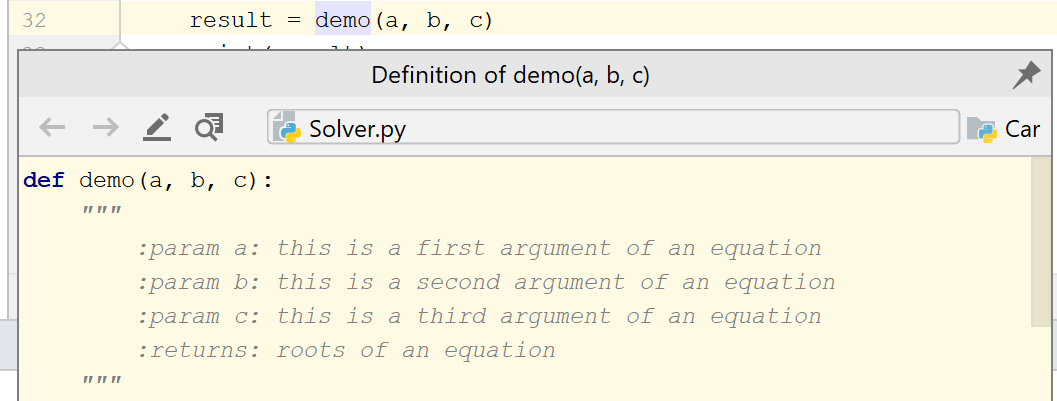
1. Spyder provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as Data Science with Anaconda. PyCharm is cross-platform, with Windows, macOS and Linux versions.
2. Coding assistance and analysis, with code completion, syntax and error highlighting, linter integration, and quick fixes.
3. Project and code navigation: specialized project views, file structure views and quick jumping between files, classes, methods and usages
4. Python refactoring: includes rename, extract method, introduce variable, introduce constant, pull up, push down and others.
5. Support for web framework: Django, web2py and Flask
6. Integrated Python debugger.
7. 4.2 SCOPE
8. Spyder provides a special language that enables you to flexibly define the sets of entities included in a scope. See Scope Language Syntax Reference for details.

**Scope**

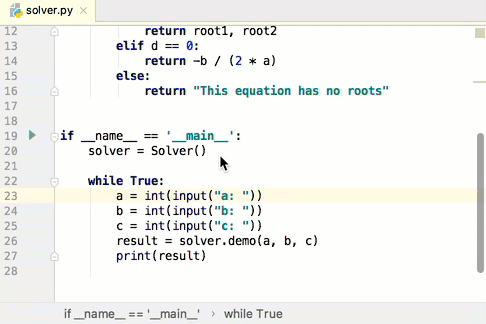
PyCharm provides a special language that enables you to flexibly define the sets of entities included in a scope. See Scope Language Syntax Reference for details.

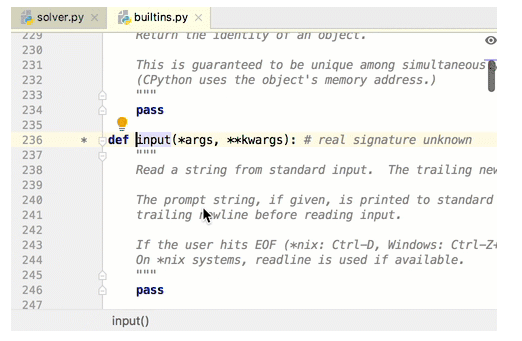
**Overview**

In Spyder, you can see where and how symbols, such as tags, classes, fields, methods, or functions are defined in your project. For this purpose, the IDE features the Quick Definition popup.



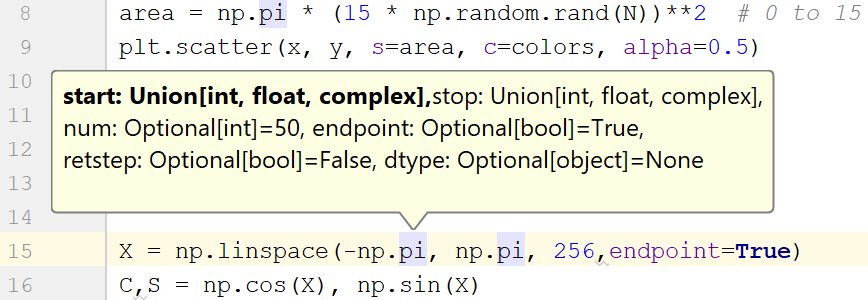
Alternatively, with the ctrl key pressed, hover the cursor over any symbol. PyCharm displays the symbol as a link and shows its definition in a tooltip. Click this link to jump to the definition of the symbol.





The Parameter Info popup shows the names of parameters in method and function calls. PyCharm automatically shows a popup with all available method signatures within 1 second (1000 milliseconds) after you type an opening bracket in the editor, or select a method from the suggestions list.

You can explicitly invoke the popup if it has closed or if your IDE is [configured not to show the popup automatically](https://www.jetbrains.com/help/pycharm/viewing-reference-information.html#configure-parameter-info-popup). To do so, press Cntr+p (or click View | Parameter Info).



**Overall Description**

Problem Statement

[Provide a statement summarizing the problem being solved by this project. The following format may be used:]

|  |  |
| --- | --- |
| The problem of | *Memory storage for large dataset* |
| Affects | *Deepfake voice* |
| The impact of which is | *Saving Storage* |
| A successful solution would | *Successfully save memory of device* |

**Product Perspective**

Deepfake detection of Audio manipulation systems is for protecting cyber-crimes or to stop getting people manipulated in the name of others. This system will detect whether the voice on the other side is legit or not. If yes then show real and if manipulated using machines then fake.

Product Position Statement

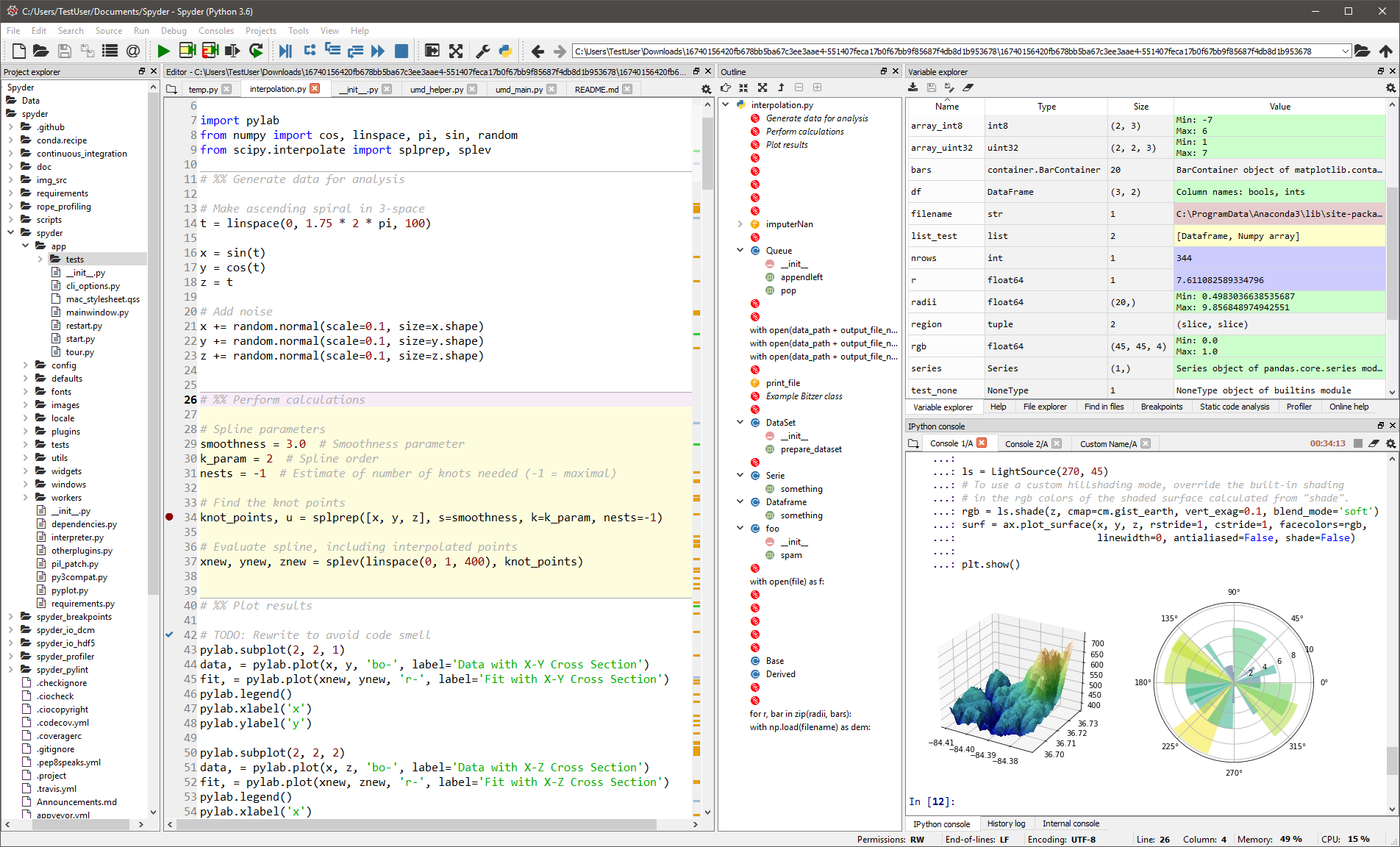
[Provide an overall statement summarizing at the highest level, the unique position the product intends to fill in the marketplace. The following format may be used:]

|  |  |
| --- | --- |
| For | *Normal people* |
| Who | *Anyone* |
| The (product name) | *Deepfake detection* |
| That | *Detects the manipulations in voice if manipulated* |
| Unlike | *AI generated voice detection* |
| Our product | *Accurately detects manipulated human voice with machine generated voice.* |

**System Interfaces**

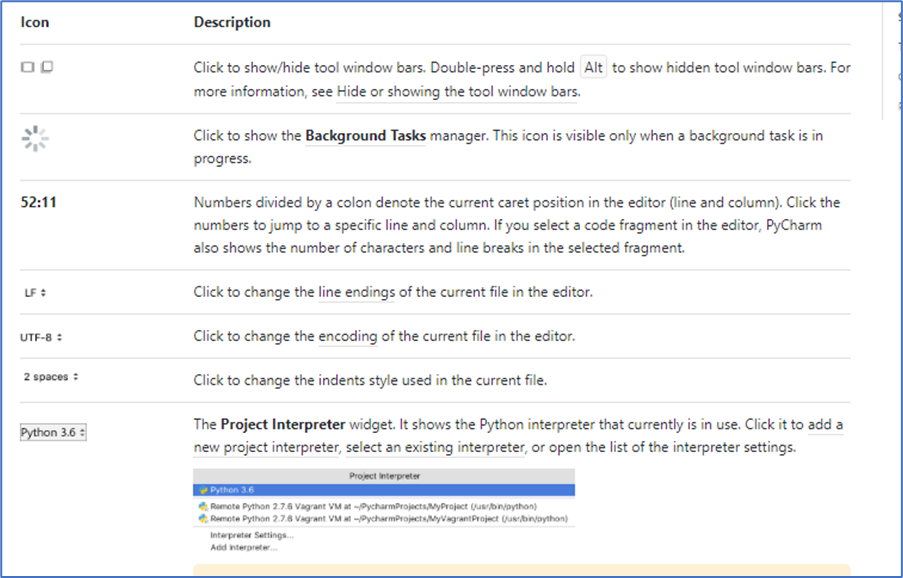
**User Interfaces**

When you open a project in PyCharm, the default user interface looks as follows:



**Status bar:**

The status bar at the bottom shows event messages and descriptions of actions when you hover over them with the mouse pointer. It also indicates the overall project and IDE status and provides access to various settings. The table below lists default icons and elements shown on the status bar. Depending on the set of [plugins](https://www.jetbrains.com/help/pycharm/managing-plugins.html) and configuration settings, there can be many other elements in addition to the default ones.



Tool windows:

Tool windows provide functionality that supplements editing code. For example, the Project tool window shows you the structure of your project, and the Run tool window displays the output of your application when you run it.

By default, tool windows are docked to the sides and bottom of the main window. You can arrange them as necessary, undock, resize, hide, and so on. Right-click the title of the tool window or click setting in the title for its arrangement options.

You can assign shortcuts to quickly access the tool windows that you frequently use. Some of them have shortcuts by default. For example, to open the Project tool window, press alt , and to open the Terminal tool window, press Alt+F12. To jump from the editor to the last active tool window, press F12

Popup menus:

Popup menus provide quick access for actions related to the current context. For example, press Alt+Insert in the editor to open the Generate popup for generating code constructs based on the context. Pressing Alt+Insert in the Project tool window will open the New popup for adding new files and directories to your project.

You can create custom popup menus using quick lists of actions that you often use. For example, you can press cntr+shft+alt+t to open the Refactor This popup with a predefined quick list of actions related to refactoring.

**Memory Constraints**

Deepfake detection of audio manipulation requires following specifications.

The system must have enough memory to store the dataset required for the training and testing of this project.

The system should support the platform on which project is made. All algorithms like algorithms of Deep learning should be supported.

Primary memory should be at least 8GB with a high end processor and secondary memory should be 30GB.

**Operations**

1. Conversion of Voice to its spectrogram image
2. Applying deep learning algorithms on them
3. Training A particular model for detecting test image
4. Detecting random voice fake or real

**Product Functions**

In recent days fraudsters are finding new ways to do fraud with people. One of the ways is phone scamming or providing the false information through others' voices which is not said by that person. That voice is called deepfake voice. And that is very difficult to detect normally with human ears. So this product detects if any manipulations are done on incoming voice or voice clip. First we took the large dataset of real and fake audio samples, male and female in every age group then we trained those samples using neural nets. We created a model out of those samples. And now we can detect any input voice clip given to this model is real or fake.

**User Characteristics**

Any user can use this application but users should know how to give inputs to the system.

**Constraints**

* Space: The solution needs to be implemented in a space-efficient manner. It must not interfere with the existing controls of the car.
* Power: There will be a limited power source so the solution needs to be designed so that it can operate properly on limited power requirements.

**Specific Requirements**

These are the requirements for the project:

1. PC/Laptop with RAM>8GB

2. Intel Quad core processor and better.

3. System with high end GPU

4. Monitor

5. Installed Spyder IDE

**External Interfaces**

a) Dataset is provided in audio format

b) User voice clip is first provided to the system and after performing operations the result is detected.

**Functions**

a) Importing the libraries.

a) Importing the dataset

b) Preprocessing the dataset.

c) Training the dataset.

d) Initializing ML algorithm.

e) Computing the output

**Performance Requirements**

For the success of this project quite good accuracy is needed as our project lies slightly in the field of deep learning. We have been able to get quite good accuracy on the given dataset and even on appending our records.

**Logical Database Requirements**

We require large number of audio clips real and fake which are labelled for training and testing purpose.

**Design Constraints**

No design constraints

**DeepFake Detection of Audio Manipulations**

**Feasibility Study Report**

***19/09/2020***



**Approvals Signature Block**

|  |  |  |
| --- | --- | --- |
| **Project Responsibility** | **Signature** | **Date** |
| *Project Guide (Internal)* | Prof. Ashwini Shringare | 19/09/2020 |
| *Project Guide (External)* |  |  |
| *Documentation Leader* |  |  |

**Table of Contents**

**1. INTRODUCTION 4 - 1**

1.1 Purpose 4 - 1

1.3 Methodology 4 - 2

1.4 References 4 - 3

**2. GENERAL INFORMATION 4 - 1**

2.1 Current Systems and Processes 4 - 1

2.2 System Objectives 5 - 1

2.3 Issues.. 5 - 2

2.4 Assumptions and Constraints 5 - 3

**3. ALTERNATIVES 5 - 1**

3.1 Comparison of Alternatives 6 - 1

**4. RECOMMENDATIONS AND CONCLUSIONS 6 - 1**

**INTRODUCTION**

The word Deepfake is a combination of "Deep Learning" and "Fake." Deep Fakes are used to replace a person in an existing image or video with someone else's likeness by using artificial neural networks. Even sounds such as voices can be trained to sound like a particular person. For example, your phone rings, you pick up. It’s your spouse asking you for details about your savings account — they don’t have the account information on hand, but want to deposit money there this afternoon. Later, you realize a bunch of money has gone missing! After investigating, you find out that the person masquerading as them on the other line was a voice 100% generated with AI. You’ve just been scammed, and on top of that, can’t believe the voice you thought belonged to your spouse was actually a fake.

**Purpose**

As mentioned above, malicious uses of deepfakes are not only terrifying, but actually beginning to happen. Building tools that can accurately discern between real and fake media is an increasingly urgent matter. As machine learning practitioners, we have the capabilities to do this, and can help mitigate a real-world problem with drastic consequences.

**Methodology**

We needed very big data then we downloaded the ASVSpoof 2019 challenge data.

The method was simple. Extract the real and fake voice samples from already labelled data. Train data till we get a good accuracy to detect real and fake voices of any voice sample.

**References**

1) Zhizheng Wu, Tomi Kinnunen, Nicholas Evans, Junichi Yamagishi, Cemal Hanilc¸i, Md Sahidullah, and Aleksandr Sizov, “Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge,” in Sixteenth Annual Conference of the International Speech Communication Association, 2015.

2) A¨aron Van Den Oord, Sander Dieleman, Heiga Zen,Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner,Andrew W Senior, and Koray Kavukcuoglu, “Wavenet: A generative model for raw audio.,” SSW, vol. 125, 2016.

3) Sercan ¨O Arik, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, et al., “Deep voice: Real-time neural text-to-speech,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 195–204.

4) Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Zongheng Yang Jaitly, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, et al., “Tacotron: Towards end-to-end speech synthesis,” in Interspeech, 2017

**General Information**

The goal of this software is to detect the audio manipulations in any voice clip. These audio manipulations can be used to extract sensitive information from targeted people.

The idea is to train and test various real and fake voice samples and after applying deep learning algorithms, develop a model such that any voice clip can be labelled as real or fake.

**Current Systems and Processes**

*.*

Current system is present that detects the AI generated voice which is different from deepfake voice. Deepfake voice is created by taking sample of human voice and manipulated according to that. We can use that voice pitch to say anything similar to that person. Deepfake voice is more realistic than AI generated voice.

**Recommendations and Conclusion**

1.This project can be implemented on low end devices as well as mobile devices. We just need to install required platforms on those devices. We can use only trained models to detect.

2. This project can be integrated with any voice chatting application so that users can be anytime informed to alert if a voice on their side is found suspicious*.*

**Deepfake Detection of Audio Manipulations**

**Software Project Plan**

***19/09/2020***



**Approvals Signature Block**

|  |  |  |
| --- | --- | --- |
| **Project Responsibility** | **Signature** | **Date** |
| *Project Guide (Internal)* | Prof. Ashwini Shringare | 19/09/2020 |
| *Project Guide (External)* |  |  |
| *Documentation Leader* |  |  |

**Table of Contents**

[**1.** **OVERVIEW**](about:blank) **3**

[**2.** **GOALS AND SCOPE**](about:blank) **3**

[2.1 Project Goals](about:blank) 3

[2.2 Project Scope](about:blank) 5

[2.2.1 Included](about:blank) 5

[2.2.2 Excluded](about:blank) 5

[**3.** **SCHEDULE AND MILESTONES**](about:blank) **5**

[**4.** **DELIVERABLES**](about:blank) **6**

**OVERVIEW**

The product is to develop a system which can detect audio manipulations and timely warnings to users if users during voice calls or in a given voice or video clip.

If video is input then wee can easily extract audio from it and perform operations which are normally performed on any audio file.

Audio sample parameters can be Amplitude and Frequency. Amplitude is just show that how loud sound is or progress of pitch of sound wave but frequency is deciding factor in detecting the features in audio samples

The set of large data is trained in the system. By producing frequency spectrograms of each labelled audio file and stored into real and fake classes. After that deep learning image classification models are applied on them to classify the test images.

Once the model is trained and getting a good percentage of accuracy then we can check any audio file for the result of the project.

**GOALS AND SCOPE**

**Project Goals**

|  |  |  |
| --- | --- | --- |
| **Project Goal** | **Priority** | **Comment/Description/Reference** |
| **Functional Goals:** | 2 | For details see the Project Requirements Specification **Error! Reference source not found.** |
| To detect deepfake audio | 1 | Detects the manipulated voice by machines |
| To detect deepfake audios from videos | 2 | Detects if video contains deepfake voice |
|  |  |  |
| **Business Goals:** |  |  |
| Avoid Fraudsters | 1 | To make safe voice conversation |
| Cost efficiency | 2 | Make the software available at low cost |
| **Technological Goals:** |  |  |
| Data usage | 1 | Study how the training data can be used for rating purposes. |
| Shell Architecture | 2 | Design of a shell-based architecture for this project. |
| **Quality Goals:** | 2 |  |
| User Interface | 1 | To improve the quality of the user interface and make it easy so that any type of user can use this product. |
|  |  |  |
| **Constraints:** |  |  |
| Low end devices | 2 | Can be used on any device present in the market |
| Space and time | 1 | To use faster and with smooth experience. |

**Project Scope**

**Included**

Deepfake Audio detection is a potentially significant research problem to its application in various fields which require visual experience. This can also detect deepfake voice in videos too. We extract the audio from video source and perform the same operations done on voice dataset.

**Excluded**

This project will exclude detection of AI generated voices. We are not so sure that this software will label AI generated voices as real or fake.

**Schedule and Milestones**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Milestones** | | **Description** | **Milestone Criteria** | **Planned Date** |
| M0 |  | Start Project | Budget Release | <10/1/2020> |
|  |  | e.g.: Project goals and scope defined | PRS or SRS reviewed  Stakeholders identified Impl. Proposal reviewed | <12/1/2020> |
| M1 |  | Start Planning |  | <18/01/2020> |
|  |  | <milestone description,  e.g. Life Cycle Objectives LCO defined> | Scope and concept described | <24/01/2020> |
| M2 |  | Start Execution |  | <14/02/2020> |
|  |  | <milestone description,  e.g. Life Cycle Architecture LCA defined> | Requirements agreed, project plan reviewed, resources committed | <16/02/2020> |
| M3 |  | Confirm Execution |  | <21/02/2020> |
|  |  | <milestone description,  e.g. alpha version> | Architecture reviewed and stable | <222/02/2020> |
| M4 |  | Start Introduction |  | <01/03/2020> |
|  |  | <milestone description,  e.g. system test passed> | Coding of new functionality finished,  Draft documentation | <05/03/2020> |
| M5 |  | Release Product |  | <15/07/2020> |
|  |  | <milestone description> | Product system tested, documentation reviewed | <15/07/2020> |
| M6 |  | Close Project |  | <23/08/2020> |

**Deliverables**

|  |  |  |  |
| --- | --- | --- | --- |
| **Identifier** | **Deliverable** | **Planned Date** | **Receiver** |
| D1 | Code Executable | 19/09/2020 | Product Owner |
| D2 | Training for the user | 20/09/2020 | End user |
| D3 | Dataset Training Guide | 21/09/2020 | Client Organization |
| D4 | Project Report | 19/09/2020 | Product Owner |

**DeepFake Detection of Audio Manipulations**

**System Implementation Document**

***19/02/2020***



**Approvals Signature Block**

|  |  |  |
| --- | --- | --- |
| **Project Responsibility** | **Signature** | **Date** |
| *Project Guide (Internal)* | Prof. Ashwini Shringare | 19/09/2020 |
| *Project Guide (External)* |  |  |
| *Documentation Leader* |  |  |

**Table of Contents**

[**1.**](#_3o7alnk) **GENERAL INFORMATION 4**

[1.1](#_23ckvvd) Version Control 4

[1.2](#_ihv636) Information Details 4

[**2.**](#_32hioqz) **COMPONENT DIAGRAM DESCRIPTION 4**

[**3.**](#_1hmsyys) **DEPLOYMENT DIAGRAM DESCRIPTION 7**

**General Information**

*Fill in the following details to keep trace of Project Phases and Iterations.*

|  |  |
| --- | --- |
| **Informational Item** | **Information** |
| Document Title | Deepfake detection of audio manipulations |
| Version | 1.0 |
| Author | Suchit Meshram |
| Project Name | Deepfake Detection of Audio manipulations |
| Project Phase | Phase 1 |
| Project Iteration | 1 |

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| Aug-02-2006 | 1.0 | Created |  |

**Information Details**

|  |  |
| --- | --- |
| **Informational Item** | **Information** |
| Filename | C:\\Desktop\Generic Class Template.doc |
| Last Saved On | Saturday, September 19, 2020, 1:38:00 PM |
| Last Saved By | Suchit Meshram |
| Number of Pages |  |

**Component diagram description**

Each subsection of this section will refer to or contain a detailed description of a system software component. The discussion provided should cover the following software component attributes:

***Classification***

Libraries Used: Pandas, Librosa, Numpy, Matplotlib, OS, soundfile, Keras, Tensorflow

Functions Used: mel\_spectrogram(), vid2Aud()

***Definition***

The specific purpose and semantic meaning of the component. This may need to refer back to the requirements specification.

1. Pandas :

pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language. It is already well on its way toward this goal.

pandas is well suited for many different kinds of data:

Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet

Ordered and unordered (not necessarily fixed-frequency) time series data.

Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels

Any other form of observational / statistical data sets. The data actually need not be labeled at all to be placed into a pandas data structure

The two primary data structures of pandas, Series (1-dimensional) and DataFrame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. For R users, DataFrame provides everything that R’s data.frame provides and much more. pandas is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries.

Here are just a few of the things that pandas does well:

Easy handling of missing data (represented as NaN) in floating point as well as non-floating point data

Size mutability: columns can be inserted and deleted from DataFrame and higher dimensional objects

Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, DataFrame, etc. automatically align the data for you in computations

Powerful, flexible group by functionality to perform split-apply-combine operations on data sets, for both aggregating and transforming data

Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects

Intelligent label-based slicing, fancy indexing, and subsetting of large data sets

Intuitive merging and joining data sets

Flexible reshaping and pivoting of data sets

Hierarchical labeling of axes (possible to have multiple labels per tick)

Robust IO tools for loading data from flat files (CSV and delimited), Excel files, databases, and saving / loading data from the ultrafast HDF5 format

Time series-specific functionality: date range generation and frequency conversion, moving window statistics, date shifting and lagging.

Many of these principles are here to address the shortcomings frequently experienced using other languages / scientific research environments. For data scientists, working with data is typically divided into multiple stages: munging and cleaning data, analyzing / modeling it, then organizing the results of the analysis into a form suitable for plotting or tabular display. pandas is the ideal tool for all of these tasks.

(2)Librosa:

is a python package for music and audio analysis. It provides the building blocks necessary to create music information retrieval systems.

(3) NumPy:

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python.

(4)matplotlib:

Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits.

(5)OS :

The OS module in python provides functions for interacting with the operating system. OS, comes under Python’s standard utility modules. This module provides a portable way of using operating system dependent functionality. The \*os\* and \*os.path\* modules include many functions to interact with the file system.

(6)soundfile:

SoundFile can read and write sound files. File reading/writing is supported through libsndfile, which is a free, cross-platform, open-source (LGPL) library for reading and writing many different sampled sound file formats that runs on many platforms including Windows, OS X, and Unix. It is accessed through CFFI, which is a foreign function interface for Python calling C code. CFFI is supported for CPython 2.6+, 3.x and PyPy 2.0+. SoundFile represents audio data as NumPy arrays.

SoundFile is BSD licensed (BSD 3-Clause License).

(7)Keras:

Keras leverages various optimization techniques to make high level neural network API easier and more performant. It supports the following features −

Consistent, simple and extensible API.

Minimal structure - easy to achieve the result without any frills.

It supports multiple platforms and backends.

It is a user friendly framework which runs on both CPU and GPU.

Highly scalability of computation.

* Benefits

Keras is highly powerful and dynamic framework and comes up with the following advantages −

Larger community support.

Easy to test.

Keras neural networks are written in Python which makes things simpler.

Keras supports both convolution and recurrent networks.

Deep learning models are discrete components, so that you can combine into many ways.

(8)Tensorflow:

TensorFlow is an open source machine learning framework for all developers. It is used for implementing machine learning and deep learning applications. To develop and research fascinating ideas on artificial intelligence, the Google team created TensorFlow. TensorFlow is designed in Python programming language, hence it is considered an easy to understand framework.

***Responsibilities***

The primary responsibilities and/or behavior of this component. What does this component accomplish? What roles does it play? What kinds of services does it provide to its clients? For some components, this may need to refer back to the requirement specification.

(1) mel\_spectogram():

This function has the responsibility to convert mel spectrogram image of any given audio input file and save with the previous name with image extension.

(2) vid2Aud():

This function has responsibility to extract the audio from a video file and save with the previous name with audio extension.

**deployment diagram description**

To determine whether you need to create a deployment model, ask yourself this: if you knew nothing about the system and someone asked you to install it and/or maintain and support it, would you want a description of how the parts of the system fit together? Deployment models force you to think about important deployment issues long before you must deliver the actual system.

(1) First of all the system should have python installed in the system

(2) Since our project uses deep learning concepts then the system must have installed its platforms such as tensorflow, Theano, Keras, etc.

(3) To plot the graphs we need matplotlib, to access audios and videos we need libraries such as Librosa, sound file, moviepy etc.

(4) We also need time library to detect the time required to keep the track while processing

**DeepFake Detection of Audio manipulations**

**Behavior: State Chart Diagram**

***19/09/2020***



**Approvals Signature Block**

|  |  |  |
| --- | --- | --- |
| **Project Responsibility** | **Signature** | **Date** |
| *Project Guide (Internal)* | Prof. Ashwini Shringare | 19/09/2020 |
| *Project Guide (External)* |  |  |
| *Documentation Leader* |  |  |

**Table of Contents**

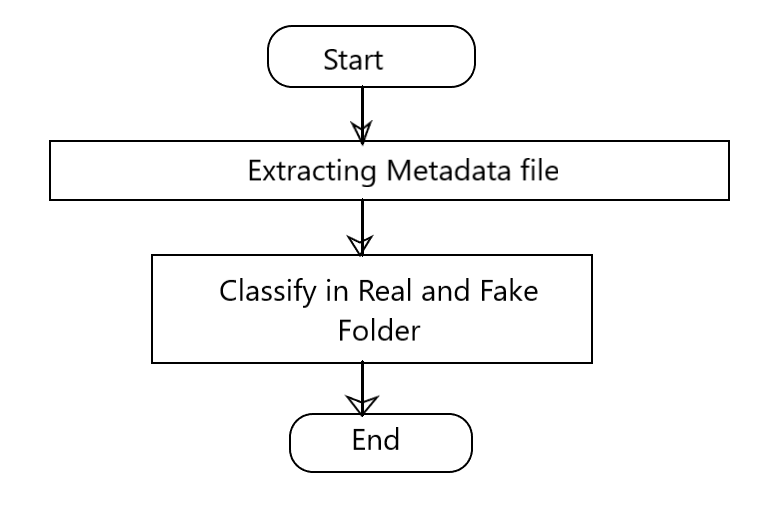
[**1.** **STATE CHART DIAGRAM OVERVIEW**](about:blank) **4**

[1.1 Object State Description](about:blank) 4

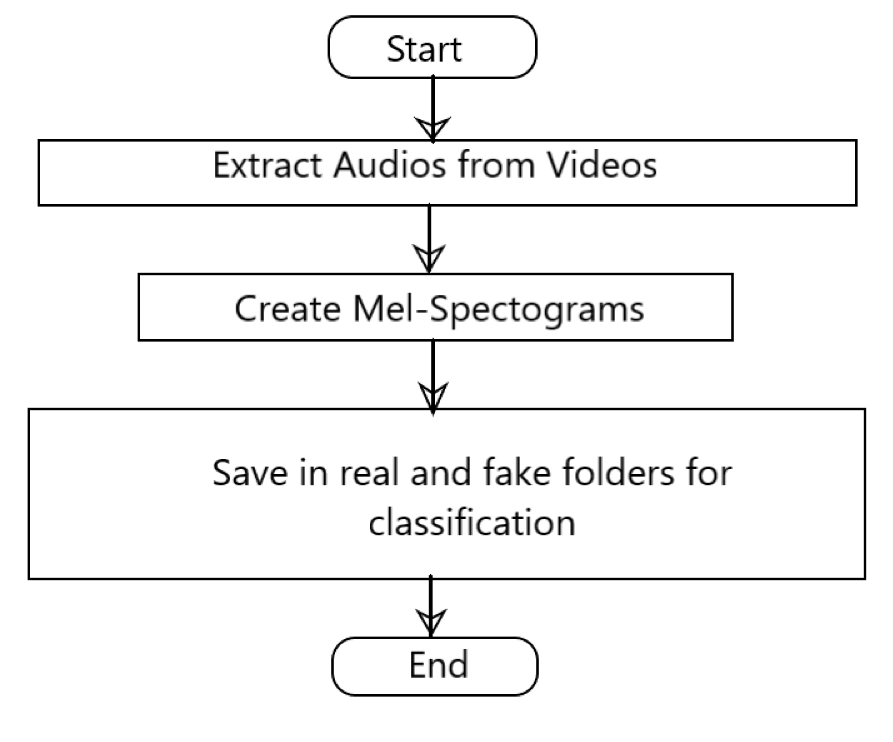
[1.2 States Description](about:blank) 4

[**2.** **SPECIFIC DESCRIPTION**](about:blank) **4**

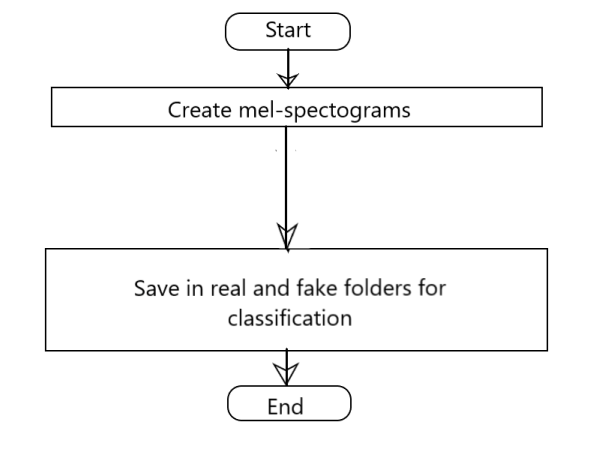
**State Chart Diagram Overview**



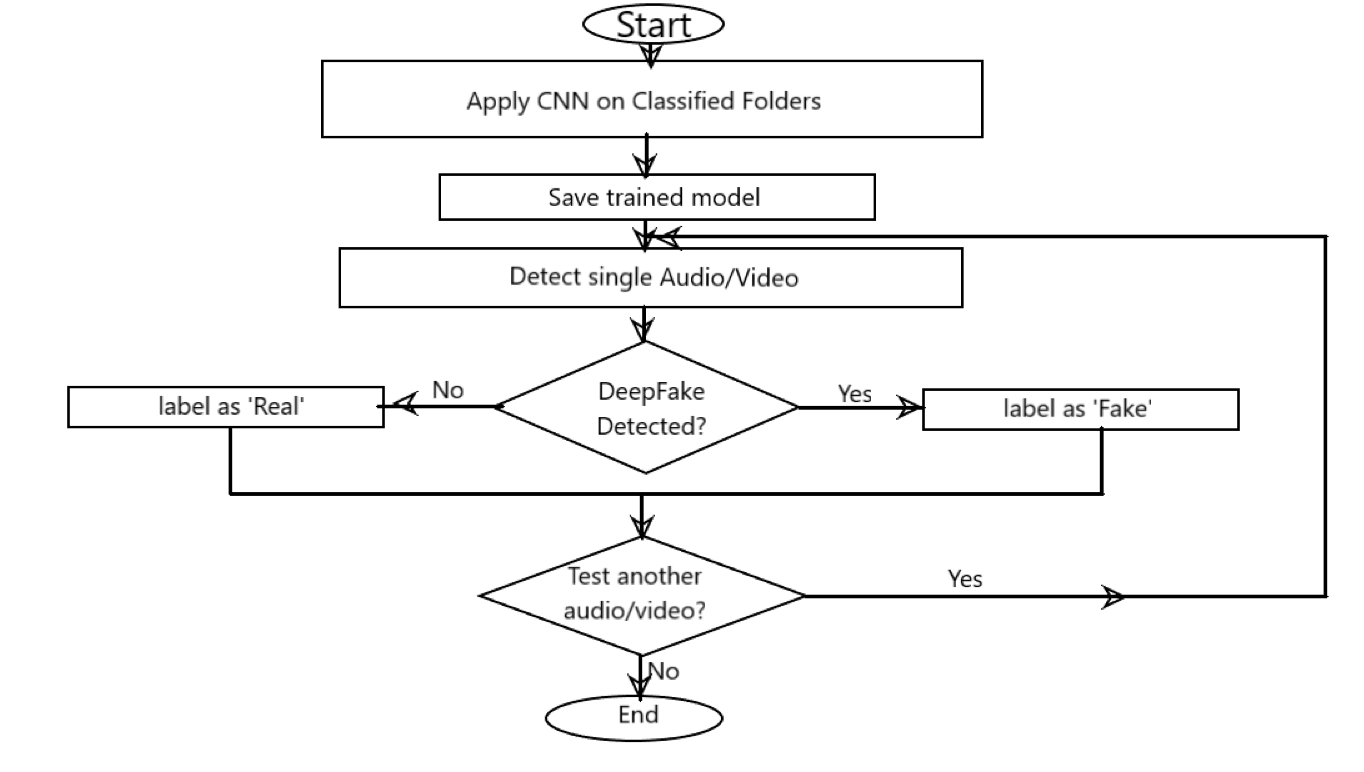
*Fig: Flow chart of Seperating real and fake files*



*Fig. Flow Chart for Creating mel spectograms of video dataset*



*Fig. Flow Chart for Creating mel spectograms of audio dataset*



*Fig. Flow chart for detecting deepfake*

**Object State Description**

|  |  |
| --- | --- |
| **Object Name** | **States** |
| Input File | Audio or video file |
| Model | Detection of Deepfake |
| Output | Detecting whether voice real or fake |

**States Description**

|  |  |  |
| --- | --- | --- |
| **State Name** | **Type** | **Action Set** |
| Input | Audio/Video | Input to model for training |
| Spectrogram | Image | To find features in Audio |
| Output | Binary output | Shows real or fake |

**DeepFake Detection of Audio Manipulations**

**System Design Document**

***19/09/2020***



**Approvals Signature Block**

|  |  |  |
| --- | --- | --- |
| **Project Responsibility** | **Signature** | **Date** |
| *Project Guide (Internal)* | Prof. Ashwini Shringare | 19/09/2020 |
| *Project Guide (External)* |  |  |
| *Documentation Leader* |  |  |

**Table of Contents**

[**1.** **GENERAL INFORMATION**](about:blank) **4**

[1.1 Version Control](about:blank) 4

[1.2 Information Details](about:blank) 4

[**2.** **CLASS DIAGRAM DESCRIPTION**](about:blank) **4**

[2.1 CRC Template](about:blank) 4

[2.2 CRC Cards Overview](about:blank) 5

[Responsibility](about:blank) 5

[Collaborator](about:blank) 5

[Domain Users (3-5 people)](about:blank) 5

[OO Design Analyst (1-2 people)](about:blank) 5

[Facilitator (1 person)](about:blank) 6

[Scribe (1-2 people)](about:blank) 6

[Observers (0-N people)](about:blank) 6

[The CRC Technique](about:blank) 6

[Selecting a Scenario](about:blank) 6

[Creating Initial CRC Card(s)](about:blank) 7

[Arranging the CRC Cards](about:blank) 7

[The CRC Interview](about:blank) 7

[Finding Classes](about:blank) 7

[Finding Collaborators](about:blank) 7

[2.2.1 General Guidelines](about:blank) 8

[2.2.2 Class Style Guidelines](about:blank) 9

[2.2.3 Interfaces](about:blank) 10

[2.2.4 Relationship Guidelines](about:blank) 10

[2.2.5 Inheritance Guidelines](about:blank) 11

[2.2.6 Aggregation and Composition Guidelines](about:blank) 12

[2.3 Class Diagram](about:blank) 12

**General Information**

|  |  |
| --- | --- |
| **Informational Item** | **Information** |
| Document Title | System Design Document |
| Version | 1.0 |
| Author | Suchit Meshram |
| Project Name | Deepfake detection of audio manipulations |
| Project Phase | Phase 1 |
| Project Iteration | 1 |

**Version Control**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| Aug-02-2006 | 1.0 | Created |  |

**Information Details**

|  |  |
| --- | --- |
| Filename | C:\Documents |
| Last Saved On | Saturday, September 19, 2020, 1:38:00 PM |
| Last Saved By | Suchit Meshram |
| Number of Pages |  |

**Class diagram description**

**CRC Cards Overview**

CRC (Class-Responsibility-Collaborator) Card Modeling is a simple yet powerful object-oriented analysis technique. CRC modeling often includes the users, analysts, and developers in a modeling and design process, bringing together the entire development team to form a common understanding of an OO development project. It is one of many tools that should be used in the collaborative design of a system.

A CRC Model is a collection of cards (usually standard index cards or larger) that are divided into three sections.

1. Class
2. Responsibility
3. Collaborator

Class

A Class represents a collection of similar objects. Objects are things of interest in the system being modeled. They can be a person, place, thing, or any other concept important to the system at hand. The Class name appears across the top of the CRC card.

Responsibility

A Responsibility is anything that the class knows or does. These responsibilities are things that the class has knowledge about itself, or things the class can do with the knowledge it has.

For example, a person class might have knowledge (and responsibility) for its name, address, and phone number. In another example an automobile class might have knowledge of its size, its number of doors, or it might be able to do things like stop and go. The Responsibilities of a class appear along the left side of the CRC card.

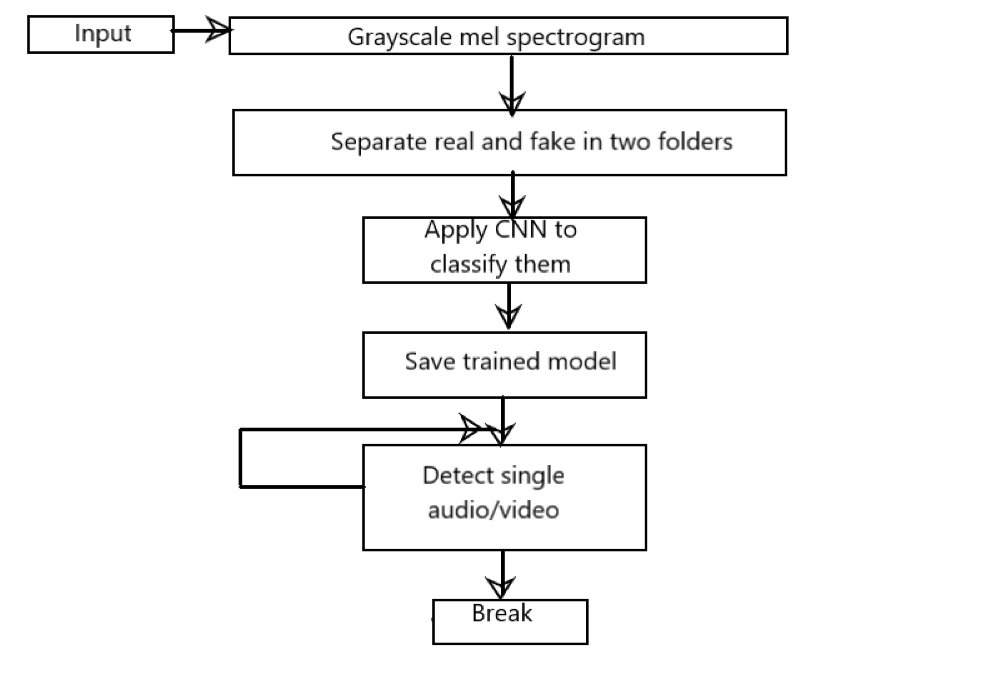
Collaborator

A Collaborator is another class that is used to get information for, or perform actions for the class at hand. It often works with a particular class to complete a step (or steps) in a scenario. The Collaborators of a class appear along the right side of the CRC card. The CRC team is comprised of different types of people involved in the project. The overall size of the CRC team is important, and active participants should be limited to 6.

Interaction between active participants grows experientially with the number candidates.

**Relationship Guidelines**

For ease of discussion the term relationships shall include all UML concepts such as associations, aggregation, composition, dependencies, inheritance, and realizations – in other words, if it’s a line on a UML class diagram we’ll consider it a relationship.



*Fig. Relational Diagram of project*

**Conclusion**

It is not the first time the world encounters forgery, it has been there for eras. Every technology has its own pros and cons. Though in this concern, the negativity outweighs the positive aspects, if people are enlightened and if proper solutions are attained such that, if another AI could possibly differentiate a legitimate audio and fake one, then that should be called a true development.

From the study conducted it is found out that the majority of the audio files were differentiated between original and duplicate copies. The wave differences could be identified between the original and duplicate copies. The alterations were identified using spectrograph of the original files. WAV files are used in all the areas of audio, from portable players, and handheld recorders, to audio interfaces and more. The WAV files are a more accurate and lossless format. They are of simple format, so we feed this wav files to CNN and got a Model which gives a great accuracy to find the Fake audios

**Future Scope**

* Can be done on videos
* UI interface for the Detection
* A App for mobile to detect incoming calls are genuine or not
* Fake News on all Platforms like YouTube, Radio, etc.
* Can be used in voice-controlled Passwords and Locke

**References**

1] Zhizheng Wu, Tomi Kinnunen, Nicholas Evans, Junichi Yamagishi, Cemal Hanilc¸i, Md Sahidullah, and Aleksandr Sizov, “Asvspoof 2015: the first automatic speaker verification spoofing and countermeasures challenge,” in Sixteenth Annual Conference of the International Speech Communication Association, 2015.

[2] A¨aron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W Senior, and Koray Kavukcuoglu, “Wavenet: A generative model for raw audio.,” SSW, vol. 125, 2016.

[3] Sercan ¨O Arik, Mike Chrzanowski, Adam Coates, Gregory Diamos, Andrew Gibiansky, Yongguo Kang, Xian Li, John Miller, Andrew Ng, Jonathan Raiman, et al., “Deep voice: Real-time neural text-to-speech,” in Proceedings of the 34th International Conference on Machine Learning-Volume 70. JMLR. org, 2017, pp. 195–204.

[4] Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Zongheng Yang Jaitly, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, et al., “Tacotron: Towards end-to-end speech synthesis,” in Interspeech, 2017.

[5] Jaime Lorenzo-Trueba, Junichi Yamagishi, Tomoki Toda, Daisuke Saito, Fernando Villavicencio, Tomi Kinnunen, and Zhenhua Ling, “The voice conversion challenge 2018: Promoting development of parallel and nonparallel methods,” in Proc. Odyssey 2018 The Speaker and Language Recognition Workshop, 2018, pp. 195–202.

[6] Tomi Kinnunen, Jaime Lorenzo-Trueba, Junichi Yamagishi, Tomoki Toda, Daisuke Saito, Fernando Villavicencio, and Zhenhua Ling, “A spoofing benchmark for the 2018 voice conversion challenge: Leveraging from spoofing countermeasures for speech artifact assessment,” in Proc. Odyssey 2018 The Speaker and Language Recognition Workshop, 2018, pp. 187–194.

[7] Massimiliano Todisco, Xin Wang, Ville Vestman, Md Sahidullah, Hector Delgado, Andreas Nautsch, JunichiYamagishi, Nicholas Evans, Tomi Kinnunen, and Kong Aik Lee, “Asvspoof 2019: Future horizons in spoofed and fake audio detection,” arXiv preprint arXiv:1904.05441, 2019.

[8] Massimiliano Todisco, H´ector Delgado, and Nicholas Evans, “A new feature for automatic speaker verification anti-spoofing: Constant q cepstral coefficients,” in Odyssey 2016, The Speaker and Language Recognition Workshop, 2016.

[9] ZhizhengWu, Eng Siong Chng, and Haizhou Li, “Detecting converted speech and natural speech for anti-spoofing attack in speaker recognition,” 2012.

[10] ZhizhengWu, Tomi Kinnunen, Eng Siong Chng, Haizhou Li, and Eliathamby Ambikairajah, “A study on spoofingattack in state-of-the-art speaker verification: the telephone speech case,” 2012.

[11] Rohan Kumar Das, Jichen Yang, and Haizhou Li, “Long range acoustic and deep features perspective on asvspoof 2019,” in IEEE Autom. Speech Recognit. Understanding Workshop, 2019.

[12] Yanmin Qian, Nanxin Chen, Heinrich Dinkel, and Zhizheng Wu, “Deep feature engineering for noise robust spoofing detection,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no. 10, pp. 1942–1955, 2017.

[13] Hossein Zeinali, Themos Stafylakis, Georgia Athanasopoulou, Johan Rohdin, Ioannis Gkinis, Luk´aˇs Burget, Jan Cˇ ernocky`, et al., “Detecting spoofing attacks using vgg and sincnet: But-omilia submission to asvspoof 2019 challenge,” arXiv preprint arXiv:1907.12908, 2019.

[14] Yexin Yang, Hongji Wang, Heinrich Dinkel, Zhengyang Chen, Shuai Wang, Yanmin Qian, and Kai Yu, “The sjtu robust anti-spoofing system for the asvspoof 2019 challenge,” Proc. Interspeech 2019, pp. 1038–1042, 2019.

[15] Balamurali BT, Kin Wah Edward Lin, Simon Lui, Jer-Ming Chen, and Dorien Herremans, “Towards robust audio spoofing detection: a detailed comparison of traditional and learned features,” arXiv preprint arXiv:1905.12439, 2019.

[16] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, andWei Liu, “Cosface: Large margin cosine loss for deep face recognition,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5265–5274.

[17] Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le, “Specaugment: A simple data augmentation method for automatic speech recognition,” arXiv preprint arXiv:1904.08779, 2019.

[18] Satoshi Nakamura, Kazuo Hiyane, Futoshi Asano, and Takashi Endo, “Sound scene data collection in real acoustical environments,” Journal of the Acoustical Society of Japan, vol. 20, no. 3, pp. 225–231, 1999.

[19] Marco Jeub, Magnus Schafer, and Peter Vary, “A binaural room impulse response database for the evaluation of dereverberation algorithms,” in 2009 16th International Conference on Digital Signal Processing, 2009, pp. 1–5.

[20] Tom Ko, Vijayaditya Peddinti, Daniel Povey, Michael L. Seltzer, and Sanjeev Khudanpur, “A study on data augmentation of reverberant speech for robust speech recognition,” in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2017, pp. 5220–5224.

[21] David Snyder, Guoguo Chen, and Daniel Povey, “MUSAN: A music, speech, and noise corpus,” CoRR, vol. abs/1510.08484, 2015.

[22] Md Sahidullah, Tomi Kinnunen, and Cemal Hanilc¸i, “A comparison of features for synthetic speech detection,” in Interspeech. 2015, ISCA (the International Speech Communication Association).

[23] Xinhui Zhou, Daniel Garcia-Romero, Ramani Duraiswami, Carol Espy-Wilson, and Shihab Shamma, “Linear versus mel frequency cepstral coefficients for speaker recognition,” in 2011 IEEE Workshop on Automatic Speech Recognition & Understanding. IEEE, 2011, pp. 559–564.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Identity mappings in deep residual networks,” in European conference on computer vision. Springer, 2016, pp. 630–645.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, “Deep residual learning for image recognition,” in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.

[26] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, “X-vectors: Robust dnn embeddings for speaker recognition,” ICASSP, 2018.

[27] G¨unter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter, “Self-normalizing neural networks,” in Advances in neural information processing systems, 2017, pp. 971–980.

[28] Diederik P Kingma and Jimmy Ba, “Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[29] Tomi Kinnunen, Kong Aik Lee, H´ector Delgado, Nicholas Evans, Massimiliano Todisco, Md Sahidullah, Junichi Yamagishi, and Douglas A Reynolds, “t-dcf: a detection cost function for the tandem assessment of spoofing countermeasures measuresand automatic speaker verification,” arXiv preprint arXiv:1804.09618, 2018.