ABSTRACT

With the rapid progress of deepfake techniques in recent years, facial video forgery can generate highly deceptive video contents and bring severe security threats. And detection of such forgery videos is much more urgent and challenging. Most existing detection methods treat the problem as a vanilla binary classification problem. It is observed that most existing face forgery methods left some common artifacts in the spatial domain and time domain, including generative defects in the spatial domain and inter-frame inconsistencies in the time domain. And a spatial-temporal model is proposed which has two components for capturing spatial and temporal forgery traces in global perspective respectively. The two components are designed using a novel long distance attention mechanism. The one component of the spatial domain is used to capture artifacts in a single frame, and the other component of the time domain is used to capture artifacts in consecutive frames.

TABLE OF CONTENTS

1. INTRODUCTION

The deep fake videos are designed to replace the face of one person with another's. The advancement of generative models [1]–[4] makes deep fake videos become very realistic. In the meantime, the emergence of some face forgery application [5]–[7] enables everyone to produce highly deceptive forged videos. Now, the deep fake videos are flooding the Internet. In the internet era, such technology can be easily used to spread rumors and hatred, which brings great harm to society. Thus the high quality deep fake videos that cannot be distinguished by human eyes directly have aroused interest among researchers. An effective detection method is urgently needed.

The general process of generating deep fake videos is shown in Fig. 1. Firstly, the video is divided into frames and the face in each frame is located and cropped. Then, the original face is converted into the target face by using a generative model and spliced into the corresponding frame. Finally, all frames are serialized to compose the deep fake video. In these processes, two kinds of defects are inevitably introduced. In the process of generating forged faces, the visual artifacts in the spatial domain are introduced by the imperfect generation model. In the process of combining frame sequences into videos, the inconsistencies between frames are caused by the lack of global constraints.

Many detection methods are proposed [8]–[10] based on the defects in the spatial domain. Some of the methods take advantage of the defects of face semantics in deep fake videos, because the generative models lack global constraints in the process of fake face generation, which introduces some abnormal face parts and mismatched details in the face from a global perspective. For example, face parts with abnormal positions [10], asymmetric faces [11], and eyes with different colors [8]. However, it's fragile to rely entirely on these semantics. Once the deep fake videos do not contain the specific semantic defects that the method depends on, the performance will be significantly degraded.

There are also some "deep" approaches [9], [12], [13], which attempt to excavate spatial defects according to the characteristics of the deep fake generators. However, compared with image contents, the forgery traces in the spatial domain are very weak, and the convolutional networks tend to extract image content features rather than the traces [14]. So blindly utilizing deep learning is not very effective in catching fake contents [15].

Since the deep fake video is synthesized frame by frame, and there is no precise constraint between the frame sequences, the inconsistencies in the time domain will be introduced. Some methods exploit these defects of the time domain. The movements of eyes are exploited in [16]. Li et al. [17] use the human blink frequency in the video to detect the deep fake videos. The movement of lip [18] and the heart rate [19] are also exploited as the identification basis between authentic videos and deep fake videos in the time domain. The optical flows and the movement patterns of the real face and fake face are classified in [20] and [21], respectively.

All of the methods mentioned above take the deep fake detection as a vanilla binary classification problem. However, as the counterfeits become more and more realistic, the differences between real and fake ones will become more and more subtle and local which making such global feature-based vanilla solutions work not well [22].

Similar problems have been studied in the field of fine grained classification. Fine-grained classification aims to classify very similar categories, such as species of the bird, models of the car, and types of the aircraft [23]. Since the deep fake detection and fine-grained classification share the same spirit, that learning subtle and discriminative features, in [22], the deep fake detection is reformulated as a fine-grained classification task. And a convolutional attention module with 1×1 is adopted to make a network focus on the subtle but critical regions.

However, combining global semantics is just as important as focusing on local areas. Because some defects are normal from a local or isolated perspective, but abnormal from a global perspective. For example, uncoordinated head postures [24], mismatched facial expressions and head movements [25], and mismatched eye details [26]. These kinds of defects exist between different parts of the face at a long distance. In other words, the local areas of focus should be determined according to the global semantics [27], and modeling long distance dependencies in both spatial domain and time domain is important. But it is not directly for the convolutional attention mechanism, especially when the kernel is small. The global pooling may be a choice to assembling global information, however, the weak forgery clues will be averaged by this operation, and resulting in a loss of distinguish ability [22].

Vision Transformer (VIT) [28] is a widely used model, which can draw global dependencies and assemble global information relying entirely on a self-attention mechanism. However, according to our experiments, as well as some existing works [29] [30], the effect of applying VIT directly to the deep fake detection task is general. Thus, we draw lessons from the fine-grained classification and propose a novel long distance attention mechanism according to the characteristics of deep fake videos. The long distance attention mechanism is designed to determine the pivotal parts of forgery by assembling information from a global perspective. We adopt the long distance attention in our spatial-temporal model to exploit the defects in the spatial domain and time domain. The spatial-temporal model is used to generate attention maps in the form of patches and guides the network to focus on pivotal local parts of the face. An example of our attention maps is shown in Fig. 2, the pivotal parts of the face are emphasized as highlights.

The contributions of this paper are summarized as follows:

• The experience of the fine-grained classification field is introduced, and a novel long distance attention mechanism is proposed which can generate guidance by assembling global information.

• It confirms that the attention mechanism with a longer attention span is more effective for assembling global information and highlighting local regions. And in the process of generating attention maps, the non-convolution module is also feasible.

• A spatial-temporal model is proposed to capture the defects in the spatial domain and time domain, according to the characteristics of deep fake videos, the model adopts the long distance attention as the main mechanism to construct a multi-level semantic guidance. The experimental results show that it achieves the state-of-the-art performance. The remainder of this paper is organized as follows. In Section II, we first discuss the related work in the field of fine grained classification. Then, the classical Vision Transformer is introduced briefly. In Section III, we analyze the defect characteristics of deep fake videos. In Section IV, the proposed method is introduced in details. Section V discusses the experimental results. The ablation analysis is given in Section VI. The conclusion is presented in Section VII.

2. LITERATURE SURVEY

1. Introduction to Deepfakes and Their Impact

Deepfake Technology: The term "deepfake" is derived from "deep learning" and "fake," representing AI-generated synthetic media where a person's likeness is superimposed onto another person’s body in a seamless manner.

Impact and Challenges: The rise of deepfakes has brought about significant challenges in various fields, including politics, entertainment, and cybersecurity, due to their potential to spread misinformation and breach privacy.

2. Traditional Methods for Deepfake Detection

Feature-Based Approaches: Early methods relied on detecting inconsistencies in facial features, lighting, and shadows. These methods often fell short due to the increasing sophistication of deepfake algorithms.

Machine Learning Techniques: Techniques such as Support Vector Machines (SVM) and Random Forests were used to identify manipulated content based on extracted features, but they struggled with high false-positive rates.

3. Deep Learning Techniques

Convolutional Neural Networks (CNNs): CNNs have been extensively used for image-based deepfake detection due to their ability to capture spatial hierarchies. However, they often fail to consider temporal dependencies in videos.

Recurrent Neural Networks (RNNs): RNNs and their variants, such as Long Short-Term Memory (LSTM) networks, have been employed to capture temporal patterns in video sequences. While effective, they can struggle with long-term dependencies due to gradient issues.

4. Attention Mechanisms in Deepfake Detection

Self-Attention Mechanism: Introduced in the Transformer model, self-attention mechanisms allow models to weigh the importance of different parts of the input sequence, enhancing the capture of dependencies across frames in a video.

Vision Transformers (ViTs): Recently, ViTs have shown promising results in image classification tasks by applying Transformer models directly to image patches. Their application to deepfake detection is an emerging area of research.

5. Long-Distance Attention for Deepfake Detection

Concept of Long-Distance Attention: Long-distance attention mechanisms extend the capability of self-attention by focusing on long-range dependencies across the entire video sequence. This approach helps in identifying subtle manipulations that occur over time.

Current Research and Findings: Recent studies have demonstrated the efficacy of long-distance attention mechanisms in improving the accuracy of deepfake detection. These models can capture intricate relationships across frames, leading to more reliable detection outcomes.

6. Comparative Studies

Performance Evaluation: Comparative studies show that models incorporating long-distance attention outperform traditional CNNs, RNNs, and even standard Transformer models in various deepfake detection benchmarks.

Challenges and Limitations: While promising, the implementation of long-distance attention models faces challenges such as computational complexity and the need for large datasets for training.

3. ANALYSIS AND DESIGN

Overview

The proliferation of deepfake videos, which manipulate or generate synthetic media using advanced AI techniques, poses significant threats to information integrity, privacy, and security. Detecting these deepfake videos has become an imperative task in the domain of digital forensics and cybersecurity. This project report delves into the analysis and design of a robust deepfake video detection system leveraging the concept of Long-Distance Attention (LDA).

Existing System

In the past few years, the performance of general image classification tasks has been significantly improved. From the amazing start of Alexnet [31] in Imagenet [32], the method based on deep learning almost dominate the Imagenet competition. However, for fine-grained object recognition [33]–[37], there are still great challenges. The main reason is that the two objects are almost the same from the global and apparent point of visual. Therefore, how to recognize the subtle differences in some key parts is a central theme for fine-grained recognition.

Earlier works [38], [39] leverage human-annotated bounding box of key parts and achieve good results. But the disadvantage is that it needs expensive manual annotation, and the location of manual annotation is not always the best distinguishing area [40], [41], which completely depends on the cognitive level of the annotator.

Since the key step of fine-grained classification is focusing on more discriminative local areas [42], many weakly supervised learning methods [23], [40], [43] have been proposed. Most of them use kinds of convolutional attention mechanisms to find the pivotal parts for detection. Fu et al. [43] use a recurrent attention convolutional neural network (RA-CNN) to learn discriminative region attention. Hu et al. [44] propose a channel-wise attention method to model interdependencies between channels. In [40], a multi-attention convolutional neural network is adopted and more fine-grained features can be learned. Hu et al. [23] propose a weakly supervised data augmentation network using attention cropping and attention dropping.

Deepfake detection and fine-grained classification are similar, that attempt to classify very similar things. Thus we learn from the experience in this field and leverage the attention maps generated with long range information to make the networks focus on pivotal regions.

Disadvantages

The spatial attention model is not designed to capture the artifacts that existed in the spatial domain with a single frame.

The system not implemented Effectiveness of spatial-temporal model which leads the system less effective.

Proposed System

- The experience of the fine-grained classification field is introduced, and a novel long distance attention mechanism is proposed which can generate guidance by assembling global information.
- It confirms that the attention mechanism with a longer attention span is more effective for assembling global information and highlighting local regions. And in the process of generating attention maps, the non-convolution module is also feasible.
- A spatial-temporal model is proposed to capture the defects in the spatial domain and time domain, according to the characteristics of deepfake videos, the model adopts the long-distance attention as the main mechanism to construct a multi-level semantic guidance. The experimental results show that it achieves the state-of-the-art performance.

Advantages

In the proposed system, the motivation to use long distance attention is given first and then the proposed model is described briefly. As aforementioned, there is no precise global constraint in the deepfake generation model, which always introduces disharmony between local regions in the face forgery from a global perspective.

In addition to the artifacts that exist in each forgery frame itself, there are also inconsistencies (e.g., unsmooth lip movement) between frame sequences because the deepfake videos are generated frame by frame. To capture these defects, a spatial-temporal model is proposed, which has two components for capturing spatial and temporal defects respectively. Each component has a novel long distance attention mechanism which can be used to assembling the global information to highlight local regions.

SYSTEM REQUIREMENTS

- ➢ H/W System Configuration: -
- ➢ Processor - Pentium –IV
- ➢ RAM - 4 GB (min)
- ➢ Hard Disk - 20 GB
- ➢ Key Board - Standard Windows Keyboard
- ➢ Mouse - Two or Three Button Mouse
- ➢ Monitor - SVGA
- SOFTWARE REQUIREMENTS:
- Operating system : Windows 7 Ultimate.
- Coding Language : Python.
- Front-End : Python.
- Back-End : Django-ORM
- Designing : Html, css, javascript.
- Data Base : MySQL (WAMP Server).

Python

Python is a **high-level, interpreted**, **interactive** and **object-oriented scriptinglanguage**. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

- **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive:** You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented:** Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language:** Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

1.3 Python Features

Python's features include:

- **Easy-to-learn:** Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read:** Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain:** Python's source code is fairly easy-to-maintain.
- **A broad standard library:** Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode:** Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable:** Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable:** You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases:** Python provides interfaces to all major commercial databases.

- **GUI Programming:** Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable:** Python provides a better structure and support for large programs than shell scripting.

Python has a big list of good features:

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identifyBusiness process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

SYSTEM TESTING

The following are the Testing Methodologies:

- **Unit Testing.**
- **Integration Testing.**
- **User Acceptance Testing.**
- **Output Testing.**
- **Validation Testing.**

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module’s control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All-important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.

The following are the types of Integration Testing:

1. Top-Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

2. Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

- The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
- A driver (i.e.) the control program for testing is written to coordinate test case input and output.
- The cluster is tested.
- Drivers are removed and clusters are combined moving upward in the program structure

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

Other Testing Methodologies

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field:

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data:

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data:

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY:

A strategy for system testing integrates system test cases and design techniques into a well-planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation. A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

4. IMPLEMENTATION

```
from django.db.models import Count

from django.db.models import Q

from django.shortcuts import render, redirect, get_object_or_404

import pandas as pd

from sklearn.feature_extraction.text import CountVectorizer

from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

from sklearn.metrics import accuracy_score

from sklearn.tree import DecisionTreeClassifier

from sklearn.ensemble import VotingClassifier

# Create your views here.

from                    Remote_User.models                    import
ClientRegister_Model,deepfake_video_detection,detection_ratio,detection_accuracy

def login(request):

if request.method == "POST" and 'submit1' in request.POST:

username = request.POST.get('username')
```

```
password = request.POST.get('password')

try:

enter = ClientRegister_Model.objects.get(username=username,password=password)

request.session["userid"] = enter.id

return redirect('ViewYourProfile')

except:

pass

return render(request,'RUser/login.html')

def index(request):

return render(request, 'RUser/index.html')

def Add_DataSet_Details(request):

return render(request, 'RUser/Add_DataSet_Details.html', {"excel_data": ''})

def Register1(request):

if request.method == "POST":

username = request.POST.get('username')

email = request.POST.get('email')
```

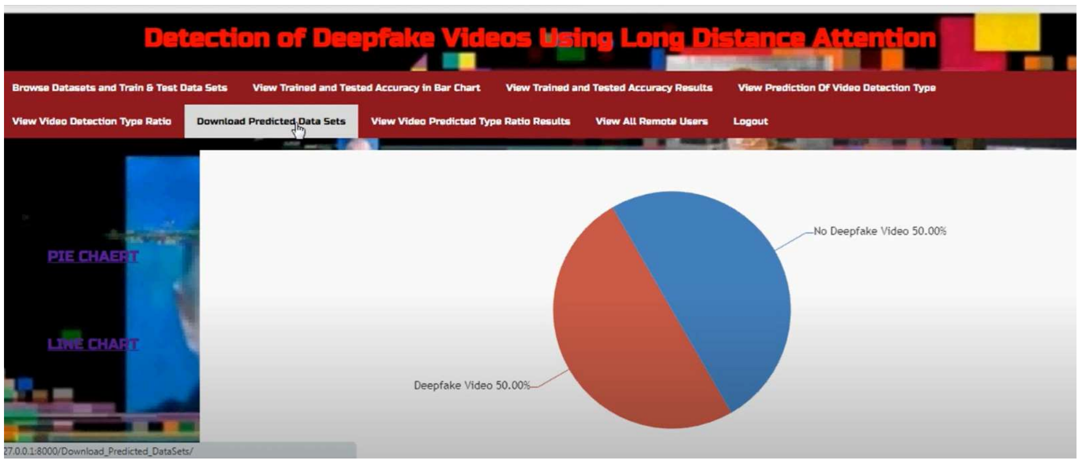
```
password = request.POST.get('password')

phoneno = request.POST.get('phoneno')

country = request.POST.get('country')

state = request.POST.get('state')

city = request.POST.get('city')

address = request.POST.get('address')

gender = request.POST.get('gender')

ClientRegister_Model.objects.create(username=username,  email=email,  password=password,
phoneno=phoneno,

country=country, state=state, city=city,address=address,gender=gender)

obj = "Registered Successfully"

return render(request, 'RUser/Register1.html',{'object':obj})

else:

return render(request,'RUser/Register1.html')

def ViewYourProfile(request):

userid = request.session['userid']
```

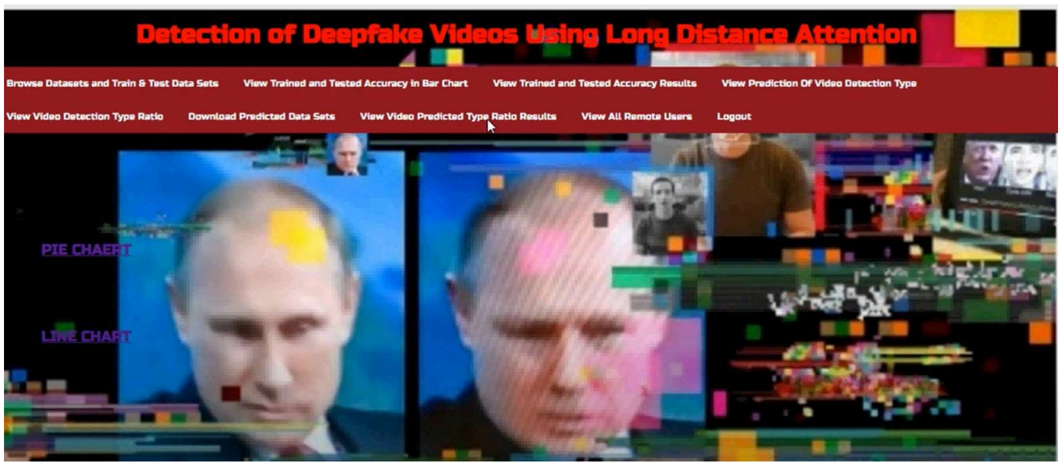
```
obj = ClientRegister_Model.objects.get(id= userid)

return render(request,'RUser/ViewYourProfile.html',{'object':obj})

def Predict_Video_Type(request):

if request.method == "POST":

if request.method == "POST":

Fid=request.POST.get('Fid')

video_id=request.POST.get('video_id')

title=request.POST.get('title')

channel_title=request.POST.get('channel_title')

publish_time=request.POST.get('publish_time')

tags=request.POST.get('tags')

views=request.POST.get('views')

likes=request.POST.get('likes')

dislikes=request.POST.get('dislikes')

thumbnail_link=request.POST.get('thumbnail_link')

description=request.POST.get('description')
```

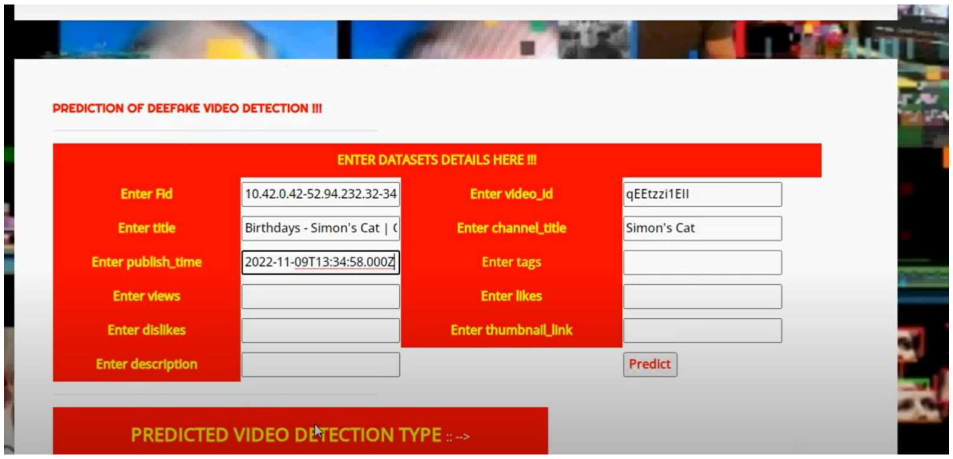
```
df = pd.read_csv('Datasets.csv',encoding='latin-1')

def apply_response(Label):

if (Label == 0):

return 0 # No Deepfake Video

elif (Label == 1):

return 1 # Deepfake Video

df['results'] = df['Label'].apply(apply_response)

cv = CountVectorizer()

X = df['Fid']

y = df['results']

print("Fid")

print(X)

print("Results")

print("CLASSIFICATION REPORT")

print(classification_report(y_test, y_pred))

print("CONFUSION MATRIX")
print(confusion_matrix(y_test, y_pred))

models.append(('MLPClassifier', mlpc))

# SVM Model

print("SVM")

from sklearn import svm

lin_clf = svm.LinearSVC()

lin_clf.fit(X_train, y_train)

predict_svm = lin_clf.predict(X_test)

svm_acc = accuracy_score(y_test, predict_svm) * 100

print(svm_acc)

print("CLASSIFICATION REPORT")

print(classification_report(y_test, predict_svm))

print(confusion_matrix(y_test, clfpredict))

models.append(('GradientBoostingClassifier', clf))

classifier = VotingClassifier(models)

classifier.fit(X_train, y_train)
```

```
y_pred = classifier.predict(X_test)

Fid1 = [Fid]

vector1 = cv.transform(Fid1).toarray()

predict_text = classifier.predict(vector1)

pred = str(predict_text).replace("[", "")

pred1 = pred.replace("]", "")

prediction = int(pred1)

if (prediction == 0):

val = 'No Deepfake Video'

elif (prediction == 1):

val = 'Deepfake Video'

print(val)

print(pred1)

deepfake_video_detection.objects.create(

Fid=Fid,

video_id=video_id,
title=title,

channel_title=channel_title,

publish_time=publish_time,

tags=tags,

views=views,

likes=likes,

dislikes=dislikes,

thumbnail_link=thumbnail_link,

description=description,

Prediction=val)
```

6. TESTING/DEBUGGING RESULTS


Detection of Deepfake Videos Using Long Distance Attention
Browse Datasets and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Prediction Of Video Detection Type
View Video Detection Type Ratio
Download Predicted Data Sets
View Video Predicted Type Ratio Results
View All Remote Users
Logout
PIE CHAERT
LINE CHART
No Deepfake Video 50.00%
Deepfake Video 50.00%
27.0.0.1:8000/Download_Predicted_DataSets/


Detection of Deepfake Videos Using Long Distance Attention
Browse Datasets and Train & Test Data Sets
View Trained and Tested Accuracy in Bar Chart
View Trained and Tested Accuracy Results
View Prediction Of Video Detection Type
View Video Detection Type Ratio
Download Predicted Data Sets
View Video Predicted Type Ratio Results
View All Remote Users
Logout
PIE CHAERT
LINE CHART

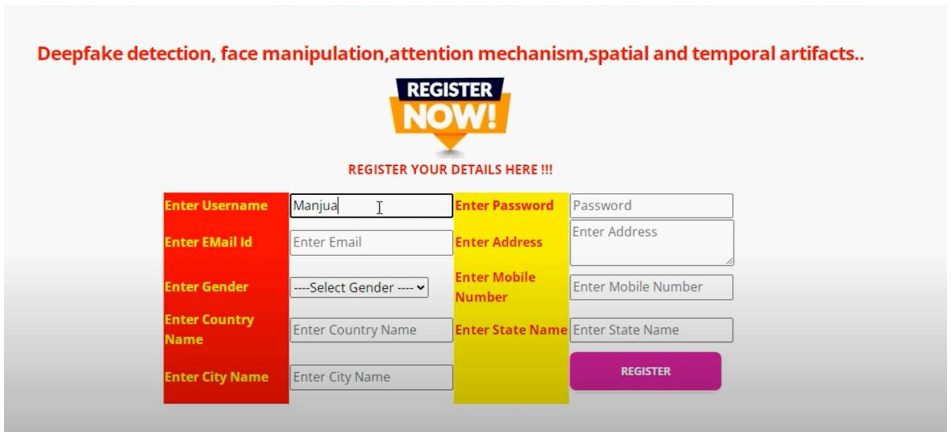


PREDICTION OF DEEFAKE VIDEO DETECTION !!!

ENTER DATASETS DETAILS HERE !!!

Enter Fid	10.42.0.42-52.94.232.32-34	Enter video_id	qEEtzzi1EIl
Enter title	Birthdays - Simon's Cat | (	Enter channel_title	Simon's Cat
Enter publish_time	2022-11-09T13:34:58.000Z	Enter tags	
Enter views		Enter likes	
Enter dislikes		Enter thumbnail_link	
Enter description			Predict

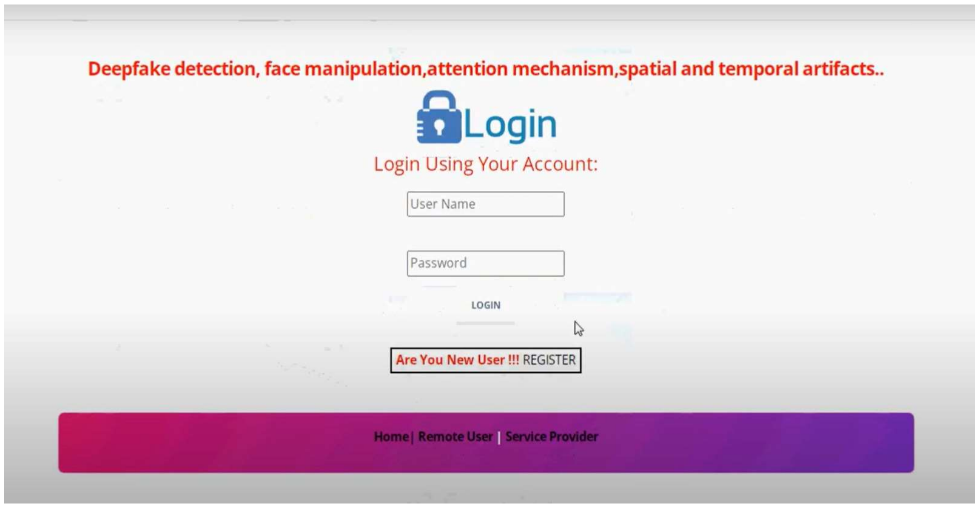
PREDICTED VIDEO DETECTION TYPE :: -->



Deepfake detection, face manipulation,attention mechanism,spatial and temporal artifacts..

REGISTER YOUR DETAILS HERE !!!

Enter Username	Manjua	Enter Password	Password
Enter EMail Id	Enter Email	Enter Address	Enter Address
Enter Gender	----Select Gender ----	Enter Mobile Number	Enter Mobile Number
Enter Country Name	Enter Country Name	Enter State Name	Enter State Name
Enter City Name	Enter City Name		REGISTER



Deepfake detection, face manipulation,attention mechanism,spatial and temporal artifacts..

Login Using Your Account:

User Name

Password

LOGIN

Are You New User !!! REGISTER

Home | Remote User | Service Provider

7. CONCLUSION

In this paper, we detect deep fake video from the perspective of fine-grained classification since the difference between fake and real faces is very subtle. According to the generation defects of the deep fake generation model in the spatial domain and the inconsistencies in the time domain, a spatial temporal attention model is designed to make the network focus on the pivotal local regions. And a novel long distance attention mechanism is proposed to capture the global semantic inconsistency in deep fake. In order to better extract the texture information and statistical information of the image, we divide the image into small patches, and recalibrate the importance between them. Extensive experiments have been performed to demonstrate that our method achieves state-of the- art performance, showing that the proposed long distance attention mechanism is capable of generating guidance from a global perspective. Apart from the spatial-temporal model and the long distance attention mechanism, we think a main contribution of this paper is that we confirm not only focusing on pivotal areas is important, but combining global semantics is also critical. This is a noteworthy point, which can be a strategy to improve current models.

8. REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative Adversarial Nets," in Advances in Neural Information Processing Systems, vol. 27, Montreal, CANADA, 2014.

[2] D. P. Kingma and M.Welling, "Auto-Encoding Variational Bayes," 2014.

[3] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive Growing of GANs for Improved Quality, Stability, and Variation," in International Conference on Learning Representations, Vancouver, Canada, 2018.

[4] Q. Duan and L. Zhang, "Look More Into Occlusion: Realistic Face Frontalization and Recognition With BoostGAN," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 1, pp. 214–228, 2021.

[5] "deepfake," http://www.github.com/deepfakes/ Accessed September 18, 2019.

[6] "fakeapp," http://www.fakeapp.com/ Accessed February 20, 2020.

[7] “faceswap,” http://www.github.com/MarekKowalski/ Accessed September 30, 2019.

[8] F. Matern, C. Riess, and M. Stamminger, “Exploiting Visual Artifacts to Expose Deepfakes and Face Manipulations,” in IEEE Winter Applications of Computer Vision Workshops, Waikoloa, USA, 2019, pp. 83–92.

[9] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, “Mesonet: a Compact Facial Video Forgery Detection Network,” in IEEE International Workshop on Information Forensics and Security, Hong Kong, China, 2018, pp. 1–7.

[10] X. Yang, Y. Li, H. Qi, and S. Lyu, “Exposing GAN-Synthesized Faces Using Landmark Locations,” in Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, Paris, France, 2019, p. 113–118.

9. APPENDICES

Appendix A: Glossary of Terms

Deepfake: A synthetic media where a person in an existing image or video is replaced with someone else's likeness using artificial intelligence.

Long Distance Attention (LDA): A mechanism in neural networks designed to capture dependencies over extended sequences, particularly effective in analyzing video frames.

Convolutional Neural Network (CNN): A type of deep learning algorithm particularly effective for image and video recognition tasks.

Recurrent Neural Network (RNN): A type of neural network designed to recognize patterns in sequences of data such as time series or video frames.

Transformer: A deep learning model architecture that relies on attention mechanisms to process sequential data, often outperforming RNNs and CNNs in tasks involving sequences.

Appendix B: Dataset Description

FaceForensics++: A large-scale dataset for detecting facial manipulations in videos. It contains manipulated videos created with various techniques, along with the original, unaltered videos.

DeepFake Detection Challenge Dataset: A dataset provided by Facebook as part of the DeepFake Detection Challenge, consisting of numerous deepfake and real videos to foster the development of detection systems.

Celeb-DF: A high-quality dataset containing deepfake videos generated with improved deepfake generation techniques, offering a diverse set of manipulated videos for training and testing.

Appendix C: Software and Tools

TensorFlow: An open-source machine learning library used for developing and training deep learning models.

PyTorch: An open-source deep learning platform that provides a flexible and efficient way to build neural networks.

OpenCV: An open-source computer vision and machine learning software library used for video processing and manipulation.

CUDA: A parallel computing platform and application programming interface model created by Nvidia, allowing software to use GPUs for general-purpose processing.

Appendix D: Hyperparameters and Model Configuration

Attention Mechanism Hyperparameters:

Number of attention heads: 8

Attention window size: 16 frames

Dropout rate: 0.1

CNN Configuration: