ISSN No. 0976-5697

Volume 11, Special Issue I, May 2020

International Journal of Advanced Research in Computer Science

CONFERENCE PAPER

Available Online at www.ijarcs.info

CENSORSHIP TOOL TO DETECT NSFW CONTENT IN A VIDEO FILE

Vinay Kumar M[1]
Assistant Proffessor
C & IT Department
REVA Univeristy
Bangalore, India
vinaykumarm@reva.edu.in

Idris Shah Hyder[2]
Student,
C & IT Department
REVA Univeristy
Bangalore, India
hyderrstum@gmail.com

Harsh Ranjan[3]
Student
C & IT Department
REVA University
Bangalore, India
harshadvict@gmail.com

Harisha G[4]
Student
C & IT Department
REVA University
Bangalore, India
harishallu12@gmail.com

Kundan Kumar[5]
Student
C & IT Department
REVA University
Bangalore, India
Kd.kundan9572@gmail.com

Abstract – Internet has vast content which is spread like wildfire across different devices, countries, cultures and ages. When so much content is available it is often prone to abuse. The abuse is mainly from NSFW (Not Safe For Work) which reaches inappropriate audiences. Our project focuses on detecting the NSFW content and letting user know if the content that they are viewing has NSFW before viewing the content. Our idea is to work with video files as they possess NSFW frames which popup without user's control over it. We use a pre-trained classifier (nudenet) to classify if a frame has NSFW content or not.

Keywords –NSFW Content, moviepy, nudenet, frames, image classification, ffmpeg engine

1. INTRODUCTION–

Every day peta-bytes of data is generated through various mediums and internet being the major player in this game. In the above given scenario censorship is a major challenge which needs to be addressed with the help of modern technologies. Our project focuses on identifying NSFW content in a video file. A video file is a set of frames which are in a sequence. We process these frames to identify if the video file has NSFW content.

Background- Image classification has rich history in machine learning application. However, The video processing or classifying a video file can be a challenge because of the fact that a video file comprises of multiple frames and these frames are again process which is computational heavy. Building an efficient algorithm to do the same is a challenge.

2. Related Work - This idea was predated by by [1]. They have used the same technique to monitor the dataset. YouTube-8m which is a huge data of YouTube videos and they process these videos[2] Have improvised it by changing certain algorithms which work on optimizing space and time complexity. We took inspiration from multiple algorithms to achieve efficiency in processing the video files.

3. Methodology–The methodology for developing application is as follows.

The figure 1.a shows an overview of the entire project. The entire project is divided into 4 major stages.

- Input stage
- Frames stage
- Classification stage
- Output stage

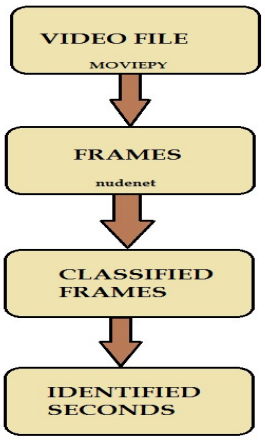

Fig 1.a overview

The first stage comprises of taking the video file from the user. The video as mentioned before comprises of thousands of frames which are in a sequence. We need to extract these frames in order to classify them.

We are using **MoviePy**to achieve this. MoviePy is based on ffmpeg engine which is a powerful tool for editing and processing video and audio. We used this tool to extract frames. But this landed us with a problem of space. On extracting frames from a video clip it often leads to increase in size. In a way, we uncompressed these frames while extraction. Given a scenario where we are trying to extract frames from a 3 minute video. Where it has 40 frames per second. That would give us 120 frames. Each frame is of some size take for example 150kb. That would land us with 150kb X 120 frames = 18000 kb. Which is way bigger than the original video. We had to figure out a way to make sure the problem is addressed. Then we thought of taking frames at an interval 5 second (The resolution can be varied).

Then these frames are fed into image classification algorithm. We are using nudenet which is an open source approach to this issue. We pip installed the module. As shown in the figure 1.b.

```
pip install nudenet
# or
pip install git+https://github.com/bedapudi6788/NudeNet
```

Fig: 1.b pip install

After the classifier has been fired up we can classify the images and search for NSFW content. And the code looks as follows.

```
from nudenet import NudeClassifier
classifier = NudeClassifier()
classifier.classify('path_to_nude_image')
# {'path_to_nude_image': {'safe': 5.8822202e-08, 'unsafe': 1.0}}
```

Fig: 1.c classifier

We feed the classifier with the location of the frames. These frames were retrieved in the initial phase using moviepy. After classification we move the nsfw frames to a different folder. Those frames are later on processed. The name of the frames will be same as that of the second from which it has been extracted from. For example a frame was retrieved at 4^{th} second the name of the file will be 4.jpeg. We use this information to understand which frame has NSFW content.

3.1. Methods to Increase accuracy of the algorithm:

We observed that the frames which were extracted and classified were inaccurate. Because of many reasons like the frame was blur and the classifier mistook it for NSFW. And also due to the resolution. The algorithm checked for frames every 2 seconds as a result it used to miss a lot of frames (for video which has 24fps between 2 seconds we missed 48 frames) and if that frame at that second somehow passed through the classifier that often led to inaccurate detection of frame.

We had to come up with methods to increase accuracy

- **Layered method:**

This method basically works by adding more layers of classification to it. The idea is to get more frames from the previous layers frames. Then calibrate those accordingly by having frame fed into a classifier.

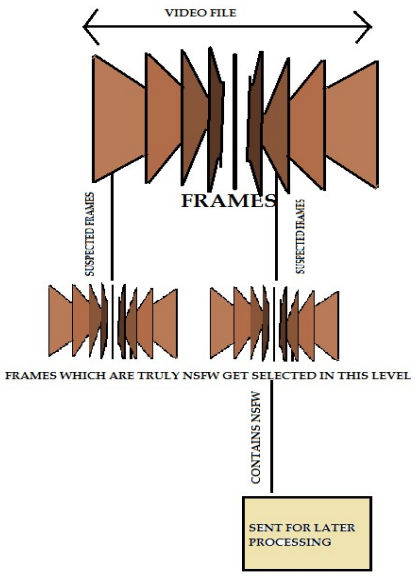

Fig: 2.a multilayer approach

In this approach we feed the classified images into the classifier one more time by taking frames which are adjacent to the suspected frame. Then calculate if those frames also contain NSFW then send those frames to next level for later on processing. This method proved to increase accuracy by 10 – 15 % than the previous single layered approach. This can also compensate for the inaccuracy of the model.

- **Inspiration from floyd's hare and tortoise algorithm:**

The floyd's hare and tortoise algorithm was used initially to detect duplicate elements in an array. The hare being the fast moving pointer and the tortoise being the slow moving pointer. This algorithm proved it's efficiency in both space and time. We took that as an inspiration. During the extraction of the frames from the video clip we initially took frames at a regular interval say at every two second. Now we take two extra pointers (i.e, two extra hares) and along with the initial pointer which acted as a tortoise. Now we could compensate if the classifier missed anything during classification. With this algorithm we can count for inaccuracy of the model. This model gives out more frames compared to the previous method and does not compromise on the space and time complexity.

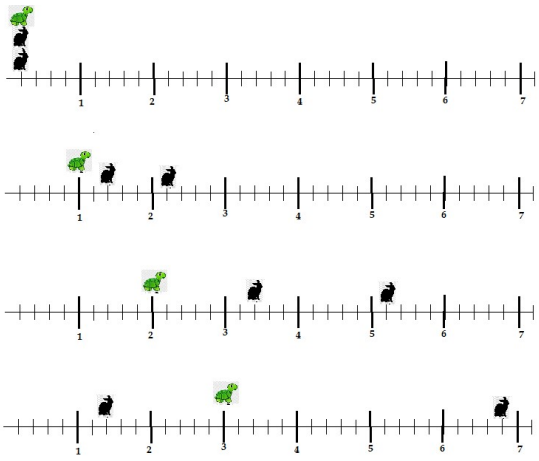

Fig: 2.b Floyd's approach

Consider a scenario where we have a video of 7 seconds and of 5 frames per second. According to this method initially the hares and tortoise are in the same 0^{th} second and in next iteration the tortoise take 1 second as the step. And hare 1 takes some incremental step and hare 2 follows the same but with a different increment. Both the hares and the tortoise will fetch the frames at different increments. The process continues till the tortoise reaches the last second. This approach will give us enough data to work with and at the same time it makes sure that we do not compromise the space.

This approach gave us enough insight on developing an efficient algorithm. The approach is unique in terms of video processing. The later stages are same as that of the fig 1.a .

4. Result and Discussion–The outcome of this research on video processing has provided us with valuable insights on optimizing and implementing an efficient solution for video processing software. Also we discussed and implemented unique ways of extracting frames without compromising on accuracy

and space efficiency. There is still a scope of improvement in this project.

Cons:

- This project completely relies on the ML model (nude net) which we got from the github.
- In case if the quality of the image is compromised the entire model gives out inaccurate output.
- In the algorithm implement one must have extra space on his/her device to accommodate frames which were extracted.
- It is computational heavy.

Pros:

- It is quiet an efficient solution if you are considering image classification in a video file.
- It can be extended to wide variety of applications from using it in your file system to your website.
- The algorithm can used with any other classifier by changing the classifier.

5. Conclusion and Future work- In conclusion of the research we realised that a huge number products can be developed. For instance we can use the classifier and run through your files system to check for NSFW content in your file system. Also an API can be implemented to make it available to wide variety of consumers. The API will be rest-full in nature as a result anyone who can perform http requests and work with ftp will be able to use our API and will be able to avail the service. Furthermore to compensate with computational heavy nature of the current solution is to implement distributed computing where we send the frames to different computational devices at once and these devices will return the information after classification.

6. Acknowledgement-We would like to thank Vinay Kumar M for guiding and motivating us with the project. We would like to thank the director of the school of computing and information technology Dr.SunilkumarManvi for his support.

We would like to thank the faculties of the school of computing and information technology.

We would like to thank our parents and friends.

7. REFERENCES –

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. arXiv preprint arXiv:1609.08675, 2016.

[2] Sandra Avila, Nicolas Thome, Matthieu Cord, Eduardo Valle, and Arnaldo De A AraúJo. Pooling in image representation: The visual codeword point of view. Computer Vision and Image Understanding, 117(5):453–465, 2013.

[3] Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine learning, 20(3):273–297, 1995.

[4] Simon S Haykin et al. Neural networks and learning machines/Simon Haykin. New York: Prentice Hall,, 2009.

[5] Shawn Hershey, Sourish Chaudhuri, Daniel PW Ellis, Jort F Gemmeke, Aren Jansen, R Channing Moore, Manoj Plakal, Devin Platt, Rif A Saurous, Bryan Seybold, et al. Cnn architectures for large-scale audio classification. In 2017 IEEE Int. Conf. on Acoustics, Speech and Signal Processing, pages 131–135. IEEE, 2017.

[6] Berthold KP Horn and Brian G Schunck. Determining optical flow. Artificial intelligence, 17(1-3):185–203, 1981.

[7] Jay Mahadeokar and Gerry Pesavento. Open sourcing a deep learning solution for detecting nsfw images. Retrieved August, 24:2018, 2016.

[8] Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. arXiv preprint arXiv:1706.06905, 2017.

[9] Daniel Moreira, Sandra Avila, Mauricio Perez, Daniel Moraes, Vanessa Testoni, Eduardo Valle, Siome Goldenstein, and Anderson Rocha. Pornography classification: The hidden clues in video space–time. Forensic science international, 268:46–61, 2016.