KidsGUARD: Fine Grained Approach for Child Unsafe Video Representation and Detection

Shubham Singh
Indraprastha Institute of Information Technology, Delhi
shubham12101@iiitd.ac.in

Rishabh Kaushal
Indraprastha Institute of Information Technology, Delhi
rishabhk@iiitd.ac.in

Arun Balaji Buduru
Indraprastha Institute of Information Technology, Delhi
arunb@iiitd.ac.in

Ponnurangam Kumaraguru
Indraprastha Institute of Information Technology, Delhi
pk@iiitd.ac.in

ABSTRACT

Increasingly more and more videos are being uploaded on video sharing platforms, and a significant number of viewers on these platforms are children. At times, these videos have violent or sexually explicit scenes (referred as child unsafe) to catch children's attention. To evade moderation, malicious video uploaders typically limit the child unsafe content to only a few frames in the video. Hence, a fine-grained approach, referred as KidsGUARD[1], to detect sparsely present child unsafe content is required. Prior approaches to content moderation either flag the entire video as inappropriate or use hand-crafted features derived from video frames. In this work, we leverage Long Short Term Memory (LSTM) based autoencoder to learn effective video representations of video descriptors obtained from using VGG16 Convolutional Neural Network (CNN). Encoded video representations are fed into LSTM classifier for detection of sparse child unsafe video content. To evaluate this approach, we create a dataset of 109,835 video clips curated specifically for child unsafe content. We find that deep learning approach (1) detects fine-grained child unsafe video content with the granularity of 1 second, (2) identifies even sparsely location child unsafe video content by achieving a high recall of 81% at high precision of 80%, and (3) outperforms baseline video encoding approaches based on like Fisher Vector (FV) and Vector of Locally Aggregated Descriptors (VLAD).

CCS CONCEPTS

• **Security and privacy** → **Social network security and privacy**; • **Information systems** → *Social networks*; • **Social and professional topics** → *Children*; • **Computing methodologies** → *Supervised learning by classification*; Semi-supervised learning settings;

KEYWORDS

Social Media Analysis, Video Analysis, Child Safety

ACM Reference Format:
Shubham Singh, Rishabh Kaushal, Arun Balaji Buduru, and Ponnurangam Kumaraguru. 2019. KidsGUARD: Fine Grained Approach for Child Unsafe Video Representation and Detection. In *The 34th ACM/SIGAPP Symposium on Applied Computing (SAC '19), April 8–12, 2019, Limassol, Cyprus.* ACM, Limassol, Cyprus, 8 pages. https://doi.org/10.1145/3297280.3297487

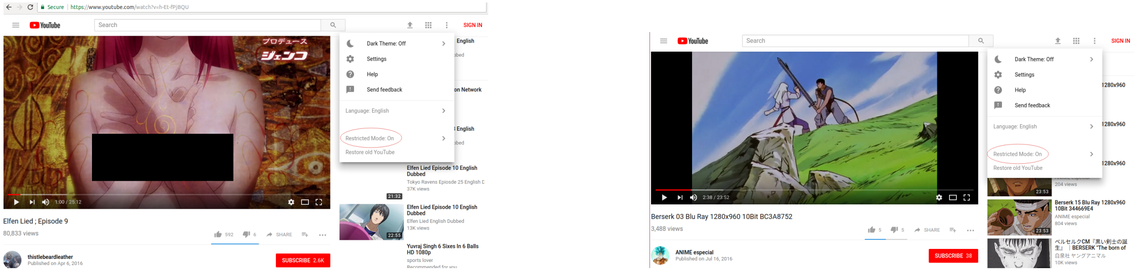
1 INTRODUCTION

The creation and the consumption of videos on the web have increased a lot over the last decade. Popular video sharing platforms, like YouTube, receive one billion hours of video views. Given the large-scale, content moderation and regulation as per the platform's guidelines becomes extremely challenging. From the user's perspective, it becomes extremely critical when viewers are children, for whom these video platforms have virtually become the television [5]. Livingstone et. al. [22] highlight 'kids in the online world getting exposed to pornography' as among the most prominent threats to children. There are significant concerns that videos targeted for children have violence or sexually explicit content [27] which we refer as *child unsafe content.* There are laws, for instance, Children's Online Privacy Protection Act (COPPA), which expect video sharing platforms to adopt and enforce mechanisms for online safety for children. Besides these norms, the presence of child unsafe content on a video sharing platform decreases its reputation and viewership as it could potentially encourage parents to dissuade their children from watching videos on such platforms. Consequently, video sharing platforms employ a dedicated group of experts, use automated mechanisms and rely on crowd-sourcing to perform content moderation. Once detected, a video is typically made as age-restricted or completely removed. Age-restricted videos have reduced visibility and are ineligible for monetization on most video platforms. As a result, putting child unsafe content *sparsely* (located in only a few scenes in a video) is one common strategy adopted by malicious video uploaders. There are many examples of such videos (Figure 1a and Figure 1b) on YouTube which can be viewed even with *restricted mode*[2] enabled, thereby indicating that the existing detection mechanisms are not working well. Further, it may be observed that the number of subscriptions and number of views for the video depicting sexually explicit content in Figure 1(a) is 2.6K and 81K since its upload on 6 April, 2016 whereas, for another video

[1] Code and Dataset at https://github.com/precog-iiitd/kidsguard-sac

[2] Restricted mode is a configuration setting on YouTube using which viewers can avoid getting inappropriate content

SAC '19, April 8–12, 2019, Limassol, Cyprus
© 2019 Association for Computing Machinery.
ACM ISBN 978-1-4503-5933-7/19/04…$15.00
https://doi.org/10.1145/3297280.3297487



(a) Video depicting nudity, uploaded since April 6, 2016 by channel with 2.6K subscribers and viewed 81K times.

(b) Video depicting violence uploaded since July 16, 2016 by channel with 38 subscribers and viewed 3.5K times.

Figure 1: Videos (get played with restricted mode ON) depicting the presence of nudity and violence which have evaded detection.

depicting violence, the number of subscribers and views are quite less (38 subscribers and 3.5K views). It is evident that the problem persists irrespective of the uploader's popularity and the video's popularity.

We address the problem by leveraging deep learning based video representation that would help in the fine-grained detection of child unsafe video content. We refer our proposed solution as *KidsGUARD*. Two types of child unsafe content are studied, one, which is sexually explicit and second, which contains violence. Put together, and we have three unsafe classes namely *sexual*, *violent* and *both* while all the rest is considered *safe*. Prior approaches [3, 7, 8, 10, 21, 30, 40] have addressed this problem using hand-crafted features at the frame level. In recent years, deep neural network based approaches [13, 25, 32, 44, 47] have emerged to solve problems in domain of image and video processing. Recurrent Neural Network (RNN) based on Long Short-Term Memory (LSTM) [9] memory units have been found to be very useful for capturing the context in a given sequence [36, 42]. However, their application to the critical problem of detection of child unsafe content has been underexplored. Our methodology is inspired by the work of Srivastava et. al. [35] which proposes an unsupervised video representation model. We build on top of their work by adapting their model for detecting child unsafe video content. We leverage LSTM autoencoders to learn video representations by taking video descriptors obtained from VGG16 Convolutional Neural Network (CNN). After conducting exhaustive experiments on 109,835 video clips, we find that fine-tuning of LSTM autoencoder works best at video clip size of one second and hence, it is apt for fine-grained level for detection of sparse child unsafe video content. Furthermore, to address the issue of sparsely located child unsafe video content, we find that the approach achieves a high recall of 81%, while at the same time, maintaining a high precision of 80%. The approach outperforms other baseline approaches of video representations based on Fisher Vector (FV) [31] and Vector of Locally Aggregated Descriptors (VLAD) [11].

Significant findings from work are as below:

- We create, first of its kind, a video dataset comprising of 109,835 video clips specifically targeted towards children, each of length one second annotated for violent and/or sexually explicit content. We plan to make this dataset publicly available for future research.
- We perform fine-grained detection of video content so that *only* portions of child unsafe content are pruned from long duration videos, rather than flagging the entire video as inappropriate for children.
- We propose a deep learning based video representation technique for fine-grained detection of child unsafe video content and perform an extensive evaluation.

Our work is helpful for video sharing platforms to weed out sparsely present child unsafe content without taking down the entire video or putting age-restrictions on videos. It also helps in building parental control solutions through the browser extensions wherein child unsafe portions are blurred so that a child can watch video safely.

2 RELATED WORK

The problem of detecting child unsafe content in videos can be *cast* into video classification or event detection problems. Most of the earlier approaches extract hand-crafted features at the frame or video level to identify discriminative patterns that aid in the detection of child unsafe content. Detection of motion features [8, 10] and skin color [7] have been used to flag indecent and pornographic content. Multi-modal approaches which fuse audio with video along with motion and skin color features have also been explored. Liu et. al. [20] propose fusing of audio signals (words) in addition to features constructed from video frames. Ulges et. al. [40] propose a multi-modal approach that combines features from audio, video and skin color. Liu et. al. [21] extend their earlier work of fusion by developing a framework that fuses audio features along with features obtained from video content for improved detection of indecent video content. Visual feature extraction and features derived from the periodicity of the audio stream are used to detect illicit content in video frames in the work by Rea et. al. [30]. Caetano et. al. [3] propose an approach using a novel video descriptor which comprises of low level local features along with *BossaNova* which is a mid-level representation of video.

Data-driven approaches which extract spatio-temporal features from video frames followed by application of conventional machine learning algorithms have been tried. Ochoa et. al. [26, 39] propose a machine learning based approach for classifying video genre for

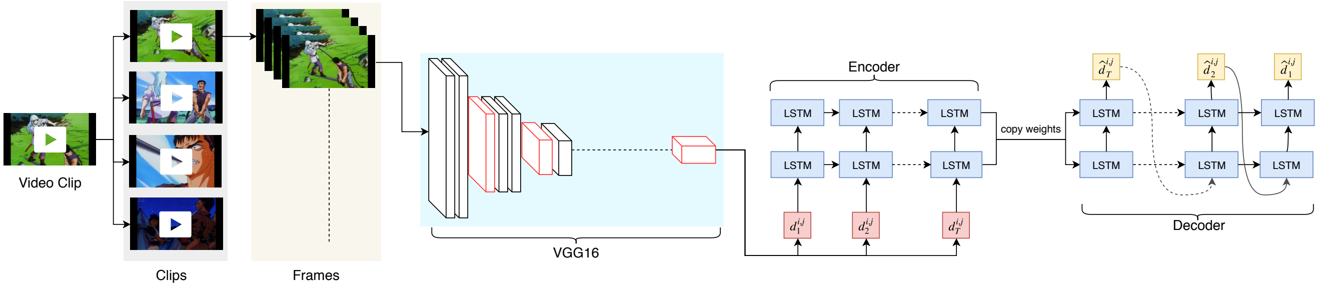


Figure 2: The left half of the figure shows the video preprocessing stages, where a video is divided into clips and a clip is divided into frames. Those frames are fed into pretrained VGG16 CNN to get feature vector. The right half shows the architecture of the LSTM Autoencoder, comprising of the encoder and the decoder. The encoder takes a sequence of video descriptors, $d_t^{i,j}$ up to T time steps and learns a *video representation*. The decoder learns to reproduce the input sequence using the learned weights from the encoder and produces the reconstructed sequence as $\hat{d}_t^{i,j}$ in the reverse order of time.

adult content. Jung et. al. [12] use the spatio-temporal motion patterns which are converted into one-dimensional signal followed by color-based region segmentation as a feature for real-time detection of indecent videos. Tang et. al. [38] propose *Pornprob*, a framework to detect pornography in videos. The framework is a combination of unsupervised clustering approach of LDA along with SVM based supervision to detect pornographic content in videos. Kaushal et. al. [14] use the meta-data of YouTube videos to construct features which were fed to machine learning classifiers to identify child unsafe content. Three kinds of meta-data features used were based on video, comments, and uploaders. After content detection, the authors performed a detailed characterization study of their uploaders and found closely linked communities of unsafe and safe content promoters. Lopes et. al. [23] use visual features (BOVF) based approach to detect indecency in videos. Authors claimed that their approach works well for even low sampling rates. Lee et. al. [19] propose a novel multi-level hierarchical system for detection of objectionable videos comprising of three phases involving hash signatures, features based on single frame and set of frames.

Significant advances in deep neural networks are being made. Karpathy et. al. [13] performe a large-scale video classification of over one million YouTube videos among 487 classes using CNNs. Multiple connectivity approaches of CNN were explored to exploit spatio-temporal features across video frames. This work was followed by Yue-Hei Ng et al. [47] which explores information across longer time periods of a video for classification. They use recurrent neural network based on LSTM which are placed at the output of CNNs to classify videos better. Ngiam et. al. [25] in their work apply deep learning over multiple modalities namely image, video, text, and audio. Wu et. al. [44] propose a deep learning framework in a hybrid configuration for classification of video which can capture static as well as short-term motion in videos using two different CNNs. The outcome of these two CNNs are fed into LSTM pipeline to capture the temporal clues. Simonyan et. al. [32] build a deep learning pipeline to detect action among videos. More specifically, they train CNNs to distinguish still scenes from those having motion. For capturing the context in a given sequence of frames in a video, RNNs based on LSTM memory units have been explored [36, 42]. Most recently, Wehrmann et. al. [43] have achieved best results using RNN-LSTM approach on Pornographic Database.[3] Application of RNN-LSTM approach to the critical problem of fine-grained detection of child unsafe video content has not been explored yet, which is our focus in this work.

3 METHODOLOGY

We divide the problem of detecting child unsafe content into the following two subproblems:

Subproblem 1. ***Video Representation*** *Given a video V comprising of a sequence of small contiguous portions, referred to as clips <c_1,c_2, ..., c_n>, the goal is to construct a function that learns an effective representation of video clips <r_1,r_2, ..., r_n>.*

Subproblem 2. ***Video Classification*** *Given a video clip representations <r_1,r_2, ..., r_n>, the goal is to build a classification function f which assigns annotations to each of the i^{th} clip c_i from among the four labels namely violent, sexual, both or safe.*

We solve these subproblems in two phases namely *video preprocessing phase* and *video representation-cum-classification phase.*

3.1 Video Preprocessing

Given a video V, it is converted into a compressed feature vector, referred as *video descriptor.* The left half of Figure 2 explains the various stages of preprocessing.

Splitting: Given that our goal is to perform fine-grained video classification, we split the i^{th} input video V_i into fixed, small sized portions of videos referred to as *video clips* (c_1^i,c_2^i,... and so on) each of say X seconds. During the evaluation of our approach, we conduct experiments with different video clip durations X = 10, 5, 3 and 1 seconds to evaluate the performance of our approach with varying degrees of granularity.

Sampling: Each of the j^{th} video clip c_j^i belonging to video V_i is sampled at the rate of 6 frames per second (FPS), thereby generating 6*X frames for each video clip, where X is the clip size in seconds.

[3] Pornographic Database is a dataset of pornographic videos, not necessarily targeted for children

Notationally, we represent these frames as $f_1^{i,j}, f_2^{i,j}, ... f_6^{i,j}$, where $f_t^{i,j}$ means t^{th} frame of clip c_j^i belonging to video V_i.

The average frame rate of the video was 23-24 FPS[4], and sampling roughly one-fourth of the frames gives us 6 FPS. Out of all the frame rates that were tried, best results are produced by sampling at 6 FPS, as higher frame rates resulted in frame redundancy and high computational times.

Resizing: Each of these frames are resized to 224 × 224 and each clip represented in form of 6^*X frames with frame size 224 × 224 × 3 forms input of next stage.

Extracting Video Descriptor: As depicted in Figure 2, we employ VGG16 [33] CNN, pretrained on around 1.3 million ImageNet data as the model learns to classify each image into one of the 1,000 classes. We use VGG16 to extract a feature vector from the frame, which we refer as *video descriptor*. Use of VGG16 for describing frames in a given video has been used in prior works [25, 35, 47]. Every 6^*X frames of a clip, each of dimensions 224 × 224 × 3, is passed as input to the VGG16 model. The frame goes across a stack of convolutional layers, each containing a sequence of filters, of size $3x3$ and stride of 1 and max-pooling is used as spatial pooling. In our approach, we discard the fully connected layers in final three stages and instead perform a softmax at last stage from the VGG16 pipeline. The output from the last layer of convolutional network which is of length 512 is considered as the final video descriptor $d_t^{i,j}$ of frame $f_t^{i,j}$ which is used as input in the next phase. To sum up, we obtain video descriptor of size 512 real values for each input frame image of size 224 × 224 × 3.

3.2 Video Representation & Classification

In this second phase of the pipeline, we first train an LSTM autoencoder so that it can learn an effective video representation for our video data. We use the autoencoder training as a *semi-supervised learning* technique which allows the model to update the weights based on unlabeled data. Subsequently, we use the trained encoder module from the autoencoder, add a fully connected layer and use it as an LSTM video classifier.

3.2.1 LSTM Autoencoder. Recurrent Neural Networks (RNN) are efficient with temporal sequential information, which make them ideal for our fine-grained, context-aware video analysis. One of the variants of RNNs, Long Short-Term Memory (LSTM) networks [9] are useful for learning long-term time dependencies. These LSTMs are used to form an autoencoder, that consists of two components: *encoder* and *decoder*. As shown in the right half of Figure 2, the autoencoder is trained upon a sequence of input signals, and it learns to recreate those input signals in the reverse sequence, with some loss. LSTM units take input at each time step t and update their cell state c_t and hidden state h_t. These internal states are maintained by three gates inside the LSTM structure, which are known as input gate n_t, forget gate f_t and output gate o_t. Given a time sequence of video descriptors as $d_t^{i,j}$ of length T corresponding to $j^{th}clip$ of i^{th} video and previous hidden state h_{t-1} at next time instance $t-1$, the gate values are updated as:

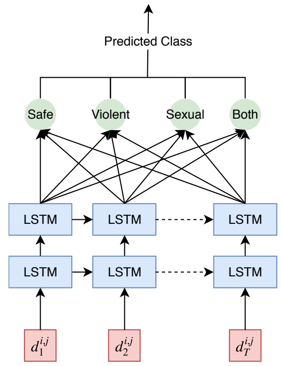


Figure 3: LSTM Classifier – The encoder module from Figure 2 is connected to layer which is fully connected at each time-step, containing four nodes, one for each class.

$$n_t = \sigma(W_{dn}d_t + W_{hn}h_{(t-1)} + b_n) \tag{1}$$
$$f_t = \sigma(W_{df}d_t + W_{hf}h_{(t-1)} + b_f) \tag{2}$$
$$o_t = \sigma(W_{do}d_t + W_{ho}h_{(t-1)} + b_o) \tag{3}$$

where W represents the weight matrices and b are the bias vectors. The cell state c_t and the hidden state h_t are finally updated at time step t by:

$$c_t = f_t c_{(t-1)} + n_t \tanh(W_{dc}d_t + W_{hc}h_{(t-1)} + b_c) \tag{4}$$
$$h_t = o_t \tanh(c_t) \tag{5}$$

After preprocessing, the video descriptors derived from VGG16 are used for input in this phase. Given that we sample video clips at 6 FPS, we obtain six video descriptors for each video clip which are passed sequentially to six LSTM units to *learn* video representation for the input video clip, which is referred to as the *encoder* of LSTM autoencoder as depicted in Figure 2. The encoder comprises of two layers of LSTM with each layer having 2,048 hidden units. Given a sequence of input features, the encoder's job is to learn the representation of that sequence and output a *context vector*, or in our case, we call it a *video representative vector*. The process is reversed in the *decoder* part of LSTM autoencoder depicted in Figure 2. Like the encoder, it is made up of two LSTM layers with 2,048 hidden states in each layer. The aim of the decoder is to learn to create back the input sequence using only the video representative vector as the input. The LSTM autoencoder is fed with few hundred thousand sequences of video descriptors corresponding to video clips to learn an effective representation for each video clip. We train this model using backpropagation and measure the loss between the input and the reconstructed sequence using mean squared error.

3.2.2 LSTM Classifier. For classification, we take the encoder module of the LSTM autoencoder and add a fully connected linear layer at each time step of LSTM output, as illustrated in Figure 3. The encoder module, as described in Section 3.2.1, comprises of two layers of LSTM with each layer having 2,048 hidden units. Given that we want to classify each of the video clips into one of the four classes, namely *Safe*, *Violent*, *Sexual* and *Both*, the final fully connected layer contains four output nodes, one for each class. For an output vector

[4] YouTube allows videos to be uploaded at 60 FPS also.

Table 1: Details of Videos (Anime Episodes). Source: https://www.anime-planet.com

Anime Series	Relevant Tags	#Episodes	#Clips
Kill La Kill	Swordplay	22	32,259
Shingeki No Kyojin	Explicit Violence	15	21,693
Elfen Lied	Sexual Abuse, Nudity, Explicit Violence	14	20,967
Berserk	Sexual Abuse, Explicit Violence	25	34,916

x_j of length C for C classes, we obtain log probabilities for each class by using the log softmax activation function, denoted as:

$$log\_softmax(x_j) = \log\left(\frac{\exp(x_j)}{\sum_{c=0}^{c-1}\exp(x_c)}\right) \tag{6}$$

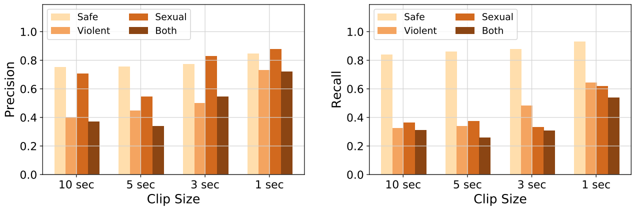
We measure the negative log-likelihood loss between the output vector x_j and one-hot vector for true label y_j as:

$$loss(x_j, y_j) = -\sum_{c=0}^{C-1} y_j log\_softmax(x_j) \tag{7}$$

The negative log-likelihood loss together with the log softmax activation function is also termed as cross entropy loss function. While training the LSTM classifier, we start with the weights learned by the encoder module during the LSTM autoencoder training and fine-tune those weights via back propagation.

4 DATASET & GROUND TRUTH

There are numerous video datasets available for research. Google has released Youtube-8M[5] video dataset that comprises of millions of video IDs along with their labels drawn from 4,716 classes. While Youtube-8M has videos which are quite generic, there are video datasets that target specific categories like action recognition (HMDB51 [17], Kinetics [15]), sports (UCF101 [34], Sports-1M [13]) and captions (ActivityNet captions [16], MSR-VTT [45]). However, none of the existing datasets focuses on child unsafe content, the closest we can find is the Pornographic Dataset [1] which has long duration videos and represent videos which are not typically watched by children. Therefore, there was a need to build a video dataset that can act as a benchmark dataset for research in the area of identifying child unsafe video content. Given that our focus is to identify the content unsafe for kids, we decide to curate animated videos containing relatively sparse and short snippets of violence and nudity. To this end, we identify four *anime* series which are Japanese animation cartoons containing explicit sexual and violent content. Anime videos make a good candidate dataset for our experiments as they are animated videos containing interspersed indecent content. Each anime series comprises of varying number of episodes. Each episode is typically about 20 - 25 minutes of length. Since we are focusing on fine-grained detection, the dataset has to comprise of small duration video clips. To this end, we split each episode into one-second duration *clips* resulting in 109,835 video clips, taking into account all episodes in that series as depicted in Table 1. To construct ground truth, a video annotation portal was developed. Annotators were ten undergraduate and graduate students belonging to the age group of 20 - 25 years, both male and female, who were given detailed instructions about the annotation task. Once they log in, they are provided with a list of videos to be watched. While watching, they were expected to mark portions of the video (thereby recording the start and end time at the back-end) as belonging to either *violent* or *sexual* or *both*. Once annotators finish watching a video, they were asked to explicitly mark the video as *watched* so that unmarked portions could be treated as the *safe* class. Labels on marked portions of the video were subsequently mapped on to each video clip of size one second. Of the 109,835 clips, 4,865 (4.4%) clips were annotated by a single annotator and hence, were ignored. At least two annotators annotated all the remaining video clips, and for more than 50% of these clips, there were more than two annotators. Clips where annotator agreement could not be achieved were not provided as input in our proposed pipeline. 68,038 clips were labeled as belonging to *safe* class, 9,730 clips to *both* class while 20,368 clips and 6,834 clips belonged to *violent* and *sexual* classes, respectively. Cohen's kappa inter-annotator agreement turned out to be 0.63 which is considered substantial agreement [18].



(a) Bar plot of precision values. **(b) Bar plot of recall values.**

Figure 4: Precision and Recall increase as we reduce clip size from *10, 5, 3 to 1* seconds in experiments with LSTM autoencoder and classifier, thereby indicating that LSTMs are able to maintain context for shorter (fine-grained) duration clip sizes.

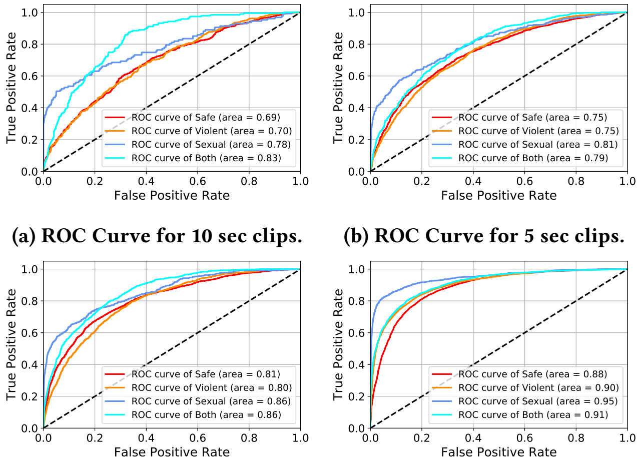
5 EXPERIMENTS AND RESULTS

In this section, we present approaches to evaluation of our methodology which uses VGG16 based video descriptors followed by LSTM autoencoder and LSTM classifier. In particular, we design experiments for the following:

- Quantitative assessment of our methodology by varying video clip size, class distribution and, label information.
- Comparison of our methodology with other baselines for video descriptors and classifiers.

All the experiments were implemented using PyTorch deep learning framework and trained on a single NVIDIA Tesla K40c GPU. The models are trained until the loss stops decreasing and the process takes around 38-40 hours for LSTM autoencoder and around 28-30 hours for LSTM classifier to converge. We perform an 80:20 training and testing data split in all the experiments. To facilitate reproducibility, we plan to release dataset and code for all the experiments upon acceptance of work.

[5] https://research.google.com/youtube8m/index.html



(a) ROC Curve for 10 sec clips.

(b) ROC Curve for 5 sec clips.

(c) ROC Curve for 3 sec clips.

(d) ROC Curve for 1 sec clips.

Figure 5: ROC Curves for imbalanced dataset of clips at X = 10, 5, 3 and 1 seconds. The ROC values improve with the decrease in clip size.

5.1 Quantitative Assessment

5.1.1 Effect of Clip Size. LSTM autoencoders are expected to keep track of *context* across the frames within the same clip. In this experiment, we evaluate the effect of change in clip size on the performance of the LSTM classifier for fine-grained detection of child unsafe content. We separately train LSTM autoencoder model on different video clip sizes, each clip of X seconds, where $X = 10, 5, 3, 1$. Learned weights from each of these separately trained encoder models are used as initial weights for the LSTM classifier which is trained on annotated video clips. Figure 4 shows that with the decrease in clip size, precision, recall and AUC values increase, thereby, implying that LSTM autoencoder is an effective approach for fine-grained detection of child unsafe video content. ROC curve for each of the clip sizes, as depicted in Figure 5 also presents similar trend as observed for precision and recall. The curves improve for shorter duration of clips and we see significant improvement for the *sexual* class. We also limit the minimum value of clip size, $X = 1$ because at 1 second, the clip contains six frames, and reducing the clip size further would have meant that we have little to no *context*. Also, annotating clips below 1 second would be quite hard.

5.1.2 Effect of Balanced Sampling. From section 4, it can be observed that the class distribution, that is, number of clips belonging to each of the four classes, is highly imbalanced. This is expected because the occurrence of a scene depicting sexual or violent content would be less frequent in proportion to the *safe* content in an episode, given the behavior of malicious uploader as alluded to in section 1. However, this disproportion leads the classifier to see more of the *safe* samples compared to the other class samples, thus introducing a bias. Therefore, in this experiment, we study the performance of the classifier in two scenarios, one in which the classifier is trained with imbalanced class distribution and second when it is trained with balanced class distribution. To balance the class distribution, we undersample the clips belonging to the abundant *safe* class, such that the number of clips belonging to the *safe* class is almost equal to the sum of clips belonging to the *violent*, *sexual* and *both* class. As observed from Table 2, when we move from imbalanced to balanced distribution, the recall for all the *unsafe* classes (*violent*, *sexual* and *both*) improves. Given that we target children, it is highly desirable to flag as many unsafe portions of the video as possible.

Table 2: Precision, Recall and AUC values for experiments with balanced vs imbalanced class distribution keeping clip size constant as 1 second. Recall values improve when we move from imbalanced to balanced sampling. The Samples column shows the number of testing samples from our 80:20 train-test split.

Data set	Class	Samples	Precision	Recall	AUC
Imbalanced	Safe	13607	0.85	0.93	0.88
	Violent	4074	0.73	0.64	0.90
	Sexual	1367	0.88	0.62	0.95
	Both	1946	0.72	0.54	0.91
Balanced	Safe	7401	0.81	0.84	0.89
	Violent	4074	0.75	**0.71**	0.90
	Sexual	1367	0.73	**0.76**	0.95
	Both	1945	0.70	**0.64**	0.92

Table 3: Precision, Recall and AUC values for experiment with Safe vs Unsafe sample set, keeping clip size constant as 1 second. The recall value for Unsafe class grows more than the average recall value of Unsafe samples in the Balanced sample set.

Class	Samples	Precision	Recall	AUC
Safe	7402	0.81	0.80	0.88
Unsafe	7385	0.80	**0.81**	0.88

Table 4: Precision, Recall and AUC values for experiment with the Unsafe sample set, keeping clip size constant as 1 second. The recall values improve for all the Unsafe classes.

Class	Samples	Precision	Recall	AUC
Violent	4074	0.86	**0.91**	0.94
Sexual	1367	0.89	**0.85**	0.97
Both	1945	0.80	**0.72**	0.92

5.1.3 Binary vs. Multi-class Classification. Our experiments in subsubsection 5.1.2 show that we achieve an increase in recall values for the *unsafe* class by having a balanced sample set. Now, in this experiment, we transform this balanced dataset into a binary class dataset by labeling all samples from classes *violent*, *sexual* and *both* as *unsafe*. We train a *binary classifier* exclusively with *safe* and *unsafe* classes, which learns to distinguish between these two classes. We see an increase in classification performance of *unsafe* class up to 80% for precision and 81% for recall, as shown in Table 3. Additionally, we take the *violent*, *sexual* and *both* samples and train a *secondary multi-class classifier* that only learns to classify these subsamples. This would be helpful in a typical deployment scenario wherein we first flag clip as *safe* or *unsafe* and then subsequently, provide specific labels to the *unsafe* clip. We observe precision and recall values of 91%, 85% and 72% for *violent*, *sexual* and *both*, which is a significant increase when compared to the results obtained by simply balancing out the class distribution. We report these numbers in Table 4.

5.2 Comparison with Baselines

In this section, we compare the performance of LSTM autoencoder and LSTM classifier with other baselines and report our observations in Table 5.

5.2.1 VGG16 based CNN Variants. To begin with, we determine the performance of VGG16 based CNN component as the sole classifier for our problem. To evaluate, we discard the final three layers from the VGG16 pipeline and add a fully connected layer with four output nodes instead. The weights of VGG16 pipeline pretrained on ImageNet are used as initial weights and subsequently, fine-tuned with our labeled dataset. We perform two variations in this approach. In the first, we freeze the weights of all the layers, except for the last one (refer this as *VGG16+FC*), and for the second model, we let the weights in all the layers to be updated (refer this as *Fine-Tuned VGG16+FC*). We find that the model with the fine-tuned weights performs slightly poorly, compared to the VGG16 model with weights pretrained on ImageNet. This is because we use our child unsafe video dataset for fine-tuning which is a specific dataset targeted for children but is relatively of smaller size as compared to the ImageNet dataset. Therefore, for the rest of the experiments, we use VGG16 based CNN pretrained on ImageNet.

5.2.2 Video Encoders and Classifier Variants. In these experiments, we compare the *effectiveness* of our LSTM autoencoder based video encoding with the baseline video encoding techniques and how it affects the performance of the classifier. Previous experiments done by Xu et al. [46] show that if the video encodings are distinctive enough, then even a shallow machine learning model can be effective in classification. Therefore, we first directly run the features obtained from the VGG16 CNN, without any video encoding technique, through a Support Vector Machine (SVM) [4, 28] classifier (refer this as *VGG16+SVM*) and get a recall of 71% for *safe* and 63% for *unsafe* class. Next, we take the VGG16 features and apply Fisher Vector (FC) encoding [31, 41] and feed it to SVM classifier (refer this as *VGG16+FV+SVM*), which results in a 58% and 71% recall for the *safe* and *unsafe* class. We are emphasizing recall values because, in the context of our problem of detecting child unsafe video content, it is extremely crucial to detect as much unsafe content sparsely located in a video as possible. Going forward, we use Vector of Locally Aggregated Descriptors (VLAD) [11, 41] as another popular video encoding technique, to represent video features and run an SVM model over them. The recall values are increased to 70% for *safe* but decrease to 62% for *unsafe* classes. Finally, to validate whether LSTM autoencoder has learned effective representation for the video clips, we take the LSTM autoencoder based encoded vector and run it through SVM. We get 79% recall for *safe* and 74% for *unsafe* class. The *increase in recall values of both safe and unsafe video content by SVM classifier indicates that the learned representation through LSTM autoencoder is more effective than the other techniques.* In fact, LSTM autoencoder based encoded representations, when passed through a fully connected layer (FC) achieve best recall rates of 80% and 81% for both *safe* and *unsafe* class, thereby reinforcing that our approach of LSTM autoencoder based encoding is a superior approach for fine grained detection of child unsafe content.

Table 5: Precision, recall and AUC values for various baselines. Performance of only VGG16 based CNN pipeline is measured. Next, video encodings using FV and VLAD are compared with LSTM Autoencoder (LA) based video representation.

Classifier	Class	Precision	Recall	AUC
VGG16 based CNN Variants				
VGG16+FC	Safe	0.55	0.76	0.75
	Unsafe	0.79	0.69	0.87
Fine-tuned VGG16+FC	Safe	0.58	0.7	0.75
	Unsafe	0.75	0.69	0.81
Video Encoder and Classifier Variants				
VGG16+SVM	Safe	0.66	0.71	0.70
	Unsafe	0.68	0.63	0.70
VGG16+FV+SVM	Safe	0.67	0.58	0.72
	Unsafe	0.63	0.71	0.72
VGG16+VLAD+SVM	Safe	0.65	0.70	0.28
	Unsafe	0.68	0.62	0.28
VGG16+LA+SVM	Safe	0.76	**0.79**	0.86
	Unsafe	0.78	**0.75**	0.86
VGG16+LA+FC	Safe	0.81	**0.80**	0.88
	Unsafe	0.80	**0.81**	0.88

6 CONCLUSION & FUTURE WORK

In this work, we propose LSTM autoencoder based video representations as an effective approach for fine-grained detection of child unsafe video content. Key takeaways are (1) LSTM autoencoder based video representation are the most suitable for capturing context at a *fine-grained* level. (2) End-to-end training of LSTM autoencoder-cum-classifier with almost equally balanced safe and unsafe content results in improvement in recall rates. (3) LSTM autoencoder-cum-classifier outperforms other baselines and conventional approaches for video encodings (FV and VLAD). Our work would help the video sharing platforms to prune child unsafe video content automatically. Only the relevant portions of child unsafe content can be blurred rather than taking down the entire video. Our work would also help build parental control solutions in the form of browser extensions which would display video safely to children by weeding out child unsafe portions. Furthermore, our methodology to flag child unsafe video content is entirely dependent on video analysis and is independent of video meta-data features like number of views, subscription count of the uploader, which can easily be manipulated by malicious users [2, 24]. As a follow-up, more specifically, the recent approaches which leverage spatio-temporal attention [6, 29] for video classification and inception-v4 and inception-resnet models [37] needs to be further explored .

REFERENCES

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

[2] Vlad Bulakh, Christopher W Dunn, and Minaxi Gupta. 2014. Identifying fraudulently promoted online videos. In *Proceedings of the 23rd International Conference on World Wide Web.* ACM, 1111–1116.

[3] Carlos Caetano, Sandra Avila, Silvio Guimaraes, and Arnaldo de A Araújo. 2014. Pornography detection using bossanova video descriptor. In *Signal Processing Conference (EUSIPCO), 2014 Proceedings of the 22nd European.* IEEE, 1681–1685.

[4] Chih-Chung Chang and Chih-Jen Lin. 2011. LIBSVM: a library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)* 2, 3 (2011), 27.

[5] Stuart Dredge. 2015. Why YouTube is the new children's TV... and why it matters. *theguardian* (November 2015). https://www.theguardian.com/technology/2015/nov/19/youtube-is-the-new-childrens-tv-heres-why-that-matters [Online; posted 19-November-2015].

[6] Wenbin Du, Yali Wang, and Yu Qiao. 2018. Recurrent Spatial-Temporal Attention Network for Action Recognition in Videos. *IEEE Transactions on Image Processing* 27, 3 (2018), 1347–1360.

[7] Lijuan Duan, Guoqin Cui, Wen Gao, and Hongming Zhang. 2002. Adult image detection method base-on skin color model and support vector machine. In *Asian conference on computer vision.* 797–800.

[8] Tadilo Endeshaw, Johan Garcia, and Andreas Jakobsson. 2008. Classification of indecent videos by low complexity repetitive motion detection. In *Applied Imagery Pattern Recognition Workshop, 2008. AIPR'08. 37th IEEE.* IEEE, 1–7.

[9] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[10] Christian Jansohn, Adrian Ulges, and Thomas M Breuel. 2009. Detecting pornographic video content by combining image features with motion information. In *Proceedings of the 17th ACM international conference on Multimedia.* ACM, 601–604.

[11] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. 2010. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.* IEEE, 3304–3311.

[12] Soonhong Jung, Junsic Youn, and Sanghoon Sull. 2014. A real-time system for detecting indecent videos based on spatiotemporal patterns. *IEEE Transactions on Consumer Electronics* 60, 4 (2014), 696–701.

[13] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition.* 1725–1732.

[14] Rishabh Kaushal, Srishty Saha, Payal Bajaj, and Ponnurangam Kumaraguru. 2016. KidsTube: Detection, characterization and analysis of child unsafe content & promoters on YouTube. In *Privacy, Security and Trust (PST), 2016 14th Annual Conference on.* IEEE, 157–164.

[15] Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. 2017. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950* (2017).

[16] Ranjay Krishna, Kenji Hata, Frederic Ren, Li Fei-Fei, and Juan Carlos Niebles. 2017. Dense-Captioning Events in Videos. In *International Conference on Computer Vision (ICCV).*

[17] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. 2011. HMDB: a large video database for human motion recognition. In *Proceedings of the International Conference on Computer Vision (ICCV).*

[18] J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *biometrics* (1977), 159–174.

[19] Seungmin Lee, Woochul Shim, and Sehun Kim. 2009. Hierarchical system for objectionable video detection. *IEEE Transactions on Consumer Electronics* 55, 2 (2009).

[20] Yizhi Liu, Xiangdong Wang, Yongdong Zhang, and Sheng Tang. 2011. Fusing audio-words with visual features for pornographic video detection. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on.* IEEE, 1488–1493.

[21] Yizhi Liu, Ying Yang, Hongtao Xie, and Sheng Tang. 2014. Fusing audio vocabulary with visual features for pornographic video detection. *Future Generation Computer Systems* 31 (2014), 69–76.

[22] Sonia Livingstone, Leslie Haddon, Anke Görzig, and Kjartan Ólafsson. 2010. Risks and safety on the internet: the perspective of European children: key findings from the EU Kids Online survey of 9-16 year olds and their parents in 25 countries. (2010).

[23] Ana Paula B Lopes, Sandra EF de Avila, Anderson NA Peixoto, Rodrigo S Oliveira, Marcelo de M Coelho, and Arnaldo de A Araújo. 2009. Nude detection in video using bag-of-visual-features. In *Computer Graphics and Image Processing (SIBGRAPI), 2009 XXII Brazilian Symposium on.* IEEE, 224–231.

[24] Miriam Marciel, Rubén Cuevas, Albert Banchs, Roberto González, Stefano Traverso, Mohamed Ahmed, and Arturo Azcorra. 2016. Understanding the detection of view fraud in video content portals. In *Proceedings of the 25th International Conference on World Wide Web.* International World Wide Web Conferences Steering Committee, 357–368.

[25] Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. 2011. Multimodal deep learning. In *Proceedings of the 28th international conference on machine learning (ICML-11).* 689–696.

[26] Victor M Torres Ochoa, Sule Yildirim Yayilgan, and Faouzi Alaya Cheikh. 2012. Adult video content detection using machine learning techniques. In *Signal Image Technology and Internet Based Systems (SITIS), 2012 Eighth International Conference on.* IEEE, 967–974.

[27] Joanne Orlando. 2017. The way your children watch YouTube is not that surprising – but it is a concern. Here are some tips. https://theconversation.com/the-way-your-children-watch-youtube-is-not-that-surprising-but-it-is-a-concern-here-are-some-tips-87597. *theconversation* (December 2017). [Online; posted 1-December-2017].

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[29] Yuxin Peng, Yunzhen Zhao, and Junchao Zhang. 2018. Two-stream Collaborative Learning with Spatial-temporal Attention for Video Classification. *IEEE Transactions on Circuits and Systems for Video Technology* (2018).

[30] Niall Rea, Gerard Lacey, R Dahyot, and C Lambe. 2006. Multimodal periodicity analysis for illicit content detection in videos. (2006).

[31] Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. 2013. Image classification with the fisher vector: Theory and practice. *International journal of computer vision* 105, 3 (2013), 222–245.

[32] Karen Simonyan and Andrew Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems.* 568–576.

[33] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[34] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. 2012. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012).

[35] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. 2015. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning.* 843–852.

[36] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems.* 3104–3112.

[37] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. 2017. Inception-v4, inception-resnet and the impact of residual connections on learning.. In *AAAI*, Vol. 4. 12.

[38] Sheng Tang, Jintao Li, Yongdong Zhang, Cheng Xie, Ming Li, Yizhi Liu, Xiufeng Hua, Yan-Tao Zheng, Jinhui Tang, and Tat-Seng Chua. 2009. Pornprobe: an lda-svm based pornography detection system. In *Proceedings of the 17th ACM international conference on Multimedia.* ACM, 1003–1004.

[39] Victor Manuel Torres Ochoa. 2012. *Adult video content detection using Machine Learning Techniques.* Master's thesis.

[40] Adrian Ulges, Christian Schulze, Damian Borth, and Armin Stahl. 2012. Pornography detection in video benefits (a lot) from a multi-modal approach. In *Proceedings of the 2012 ACM international workshop on Audio and multimedia methods for large-scale video analysis.* ACM, 21–26.

[41] A. Vedaldi and B. Fulkerson. 2008. VLFeat: An Open and Portable Library of Computer Vision Algorithms. http://www.vlfeat.org/. (2008).

[42] Subhashini Venugopalan, Marcus Rohrbach, Jeffrey Donahue, Raymond Mooney, Trevor Darrell, and Kate Saenko. 2015. Sequence to sequence-video to text. In *Proceedings of the IEEE international conference on computer vision.* 4534–4542.

[43] Jônatas Wehrmann, Gabriel S Simões, Rodrigo C Barros, and Victor F Cavalcante. 2018. Adult content detection in videos with convolutional and recurrent neural networks. *Neurocomputing* 272 (2018), 432–438.

[44] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, and Xiangyang Xue. 2015. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd ACM international conference on Multimedia.* ACM, 461–470.

[45] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. 2016. Msr-vtt: A large video description dataset for bridging video and language. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on.* IEEE, 5288–5296.

[46] Zhongwen Xu, Yi Yang, and Alexander G Hauptmann. 2015. A discriminative CNN video representation for event detection. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on.* IEEE, 1798–1807.

[47] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. 2015. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 4694–4702.