EasyChair Preprint

№ 7815

Detection of Animated Scenes Among Movie Trailers

Irmak Türköz and Halil Altay Güvenir

April 19, 2022

Detection of Animated Scenes Among Movie Trailers

Irmak Türköz[1] and H. Altay Güvenir[2]

[1] Bilkent University, Dept. of Computer Engineering, Ankara, Turkey
irmak.trkz@gmail.com
iturkoz.wixsite.com/personal
[2] Bilkent University, Dept. of Computer Engineering, Ankara, Turkey
guvenir@cs.bilkent.edu.tr
cs.bilkent.edu.tr/~guvenir

Abstract. This paper presents a method for the detection of animated scenes in movie trailers. Regardless of the studio, artists, and the unique features of diverse animation creation techniques, machine learning tools can provide concise detection methods of animated scenes in movie components. A dataset is prepared by selecting scenes from trailers using shot boundary analysis and removal of scenes containing non-movie contents; e.g. credentials. The dataset is composed of over 1400 movie scenes from 230 trailers of various genres. A Convolutional Neural Network (CNN) architecture followed by Gated Recurrent Unit (GRU) is built by using transfer learning of EfficientNet.

Keywords: Movie Trailer Labeling · Scene Understanding · Deep Learning · Computer Vision.

1 Introduction

Animation refers to a technique of modeling to create an illusion of a photographic event used in cartoons whereas animated movies refer to a sequence of drawings. Fortunately, the wide range of techniques used to create animated movies have been supported by the availability of larger data, more powerful computers, and accessibility to high-level photo-realistic animation creation tools boosted by machine learning.

The problem of distinguishing cartoons from photo-realistic movies can be viewed as a case study of automatic video genre classification as an immensely researched and yet to be improved subject of computer understanding. Moreover, an animation discriminator can be a tool to serve as a discriminator in the Generative Adversarial Networks (GAN), as some studies focus on generating cartoon sequences from photographic sequences. Thanks to improved cartoon discriminators, more realistic cartoons named Computer Generated Images(CGI) can be created. For instance, Chen et al. designed a GAN system to create high-quality cartoon style images from photo-realistic images [4].

Latest technologies in Computer Generated Images (CGI) have been creating more realistic animation styles that become challenging to distinguish by humans. An animation discrimination system can be further enhanced into a system to distinguish even the most photo-realistic yet computer-created movies. As Farid et al. stated, advancements in these realistic computer-generated releases blur the line between reality and fantasy, and they can even cause legal situations, such as engaging in sexually explicit conduct [5].

In this research, we have investigated if a deep learning model can distinguish every animated scene from non-animated ones, despite the diverse cartoon styles created by numerous artists. Our approach is based on training a neural network architecture trained by the scenes from trailers of the animated movie genre and other genres. To construct our dataset, we have employed a part of the MovieLens20M dataset which maps movie genres to YouTube trailers [7]. A trailer is divided into scenes using a shot boundary detection algorithm by automatically splitting the video into separate clips [1]. The term segment is used interchangeably with scene in the rest of the paper.

Once we have extracted the shot boundaries of these trailers, we eliminated segments containing misleading information such as logo and credentials. After the preprocessing is completed, 1447 movie segments exists in the dataset. In order to learn a model using the deep learning techniques, we constructed an architecture containing a Convolutional Neural Network (CNN) followed by a Gated Recurrent Units (GRU). The data set is fed into the deep neural network architecture incorporated CNN-GRU model. The model learned by the system achieved promising results compared to the state-of-the-art research.

In the next section, we will give a brief overview of the related work. Section 2 will describe the dataset used in the experiments. Section 3 introduces the methodology of constructing the model for detecting the animation scenes. The last section concludes the paper and gives directions for future work.

1.1 Related Work

One of the first attempts to classify movie segments as either animated or non-animated is by Roach et al. for the classification of video fragments using motion only [13]. Their dataset consisted of only a few sequences, 8 cartoon sequences simplified as sketched cartoons and animated cartoons, and 20 non-cartoon sequences where they have also classified non-cartoon genre as computer-generated cartoons, sci-fi, and real-life motions to simplify the problem.

A hand-crafted method was suggested by Ianeva et al. [8] with accounting pattern spectrum of parabolic size distributions to map color histograms, texture, color edges, brightness, and various image analysis filters, and trained with Probabilistic Gaussian Mixture Model and SVM. Even though they have accomplished a satisfying accuracy with their trained key-frames, their model suffered from external key-frames with divergent visual content characteristics.

Eventually, Glasberg et al. suggested a method based on MPEG-7 features extracted from 100 representative video sequences of cartoons, and commercials with animated cartoon sequences [6] which accomplished to classify of a sequence

of 50 frames in one minute with 0.8 accuracy of animated scenes and 0.85 of non-animated scenes.

To distinguish not only cartoons created by the human hand but also computer graphics images, from natural photographs Ng et al. suggested a method with accounting wavelet, geometry, and cartoon features of an image with 3200 images belonging to 4 different classes and achieved 0.84 accuracy [11]. Color histograms and SVM were considered to classify computer graphics images from photographic images by Chen et al. which proved that HSV color channels are more informative than RGB color channels with slightly better accuracy and true positive rate [3]. To extract video level features, Chen et al. used color component and color kind based on region segmentation and proved that SVM is better performing than GMM, and Manifold Ranking [2]. They have also considered cartoon segmentation as their features, and they have extracted their frames from the movies with a seemingly primitive shot boundary analyzer as a change detector.

On the other hand, Sankar et al. also considered hybrid images consisting of both cartoon and photo-realistic segments of 557 features which can be reduced 80 without a crucial loss in performance [14]. Moreover, Ionescu et al. worked with two sets of features namely temporal descriptors, like rhythm or action, and color descriptors are determined using color perception to achieve an average global correct classification up to 0.92 [9].

Initial attempts to use neural networks to classify cartoons as movie genre was presented by Montagnuolo et al. with accounting texture information, color histograms and temporal activity information based on the displaced frame difference which acquired average precision of 0.86 and an average recall of 0.85 [10]. Alternatively, Quan et al. accompanied CNN with four convolutional blocks followed by two fully connected layers to achieve 0.94 accuracy on their dataset of images with two labels: natural and computer-generated which did not include human-drawn cartoons [12]. More recently, Zhang et al. attempted to classify computer-generated images from RGB color channel and pixel correlation to acquire 0.94 accuracy on the ScNet dataset with a convolutional neural network architecture [16].

Most of the related work in this area either considered images or motion separately. However, we have assembled a dataset and built a fused model that combines both spatial information using CNN and temporal information using GRU to attempt an improved performance on accuracy.

2 Dataset

We consider one of our contributions to the research is the construction of a dataset composed of animated and non-animated movie trailers since there is not a proper dataset for this specific task as we have observed. As preliminary information, our dataset contains movie trailers with a 30 fps (frame per second) frame rate. Since we assume that a trailer may contain both animated scenes and photo-realistic scenes, we initially split each scene of a trailer by shot boundary

Fig. 1: Eliminated Image Examples

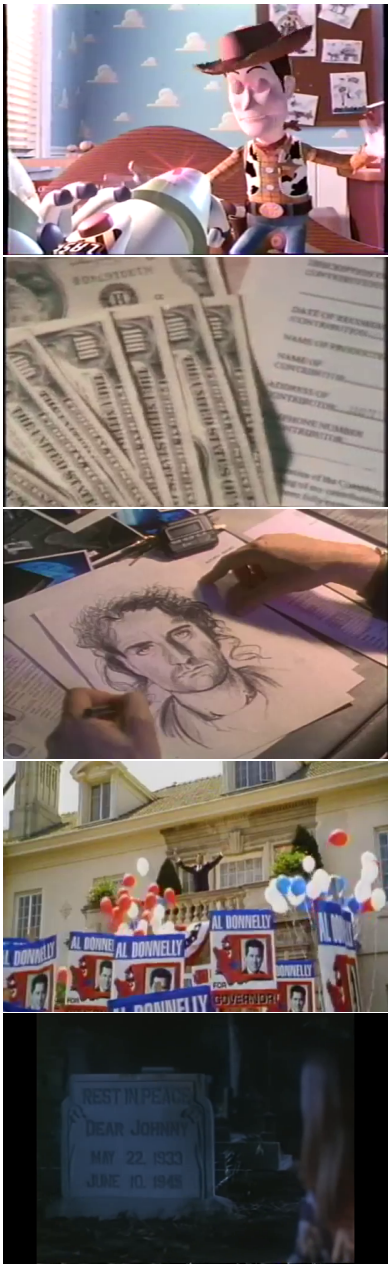

Fig. 2: Kept Image Examples

analysis. In this implementation, we refer to the movie trailer partitions generated by a scene detector as scenes. Each of these scenes is represented by a

pair of frame indices, one for the beginning and the other for the end frame. For example, since every trailer has 30 frames per second, a pair of (360,600) refers to the scene which is a segment of the trailer between 12th and 30th seconds.

The preparation of the dataset is followed by a several steps: (1) The initial links from YouTube trailers are extracted from MovieLens20M with some of the links that are unavailable due to deletion from YouTube [7]. (2) Downloaded trailers are fed into the shot-boundary detection unit of PySceneDetect's[1], ContentDetector with parameters; threshold as 40.0 and minimum scene length as 1, which allows us to discriminate fade-in and fade-out in a faster way [1]. In this step, we have also discovered that black and white trailers were not partitioned due to the implementation procedure of the open-source library. However, since black and white movies are not produced anymore and upcoming animated movies are presumably colored, we have not addressed this issue and removed non-animated black-white trailers from our dataset. (3) The output of shot-boundary detection is later filtered with the removal of scenes with a length less than 100 frames (approximately 3 seconds) and more than 1800 frames (approximately 30 seconds). This allows the later detection algorithm to be more reliable and robust since it is a motion sensitive. (4) The downloaded trailers are pre-processed with our model to eliminate movie trailer segments that are not meaningful such as credential scenes, studio logos, release dates, etc. (see Fig. 1) with a model trained to discriminate the unnecessary components.

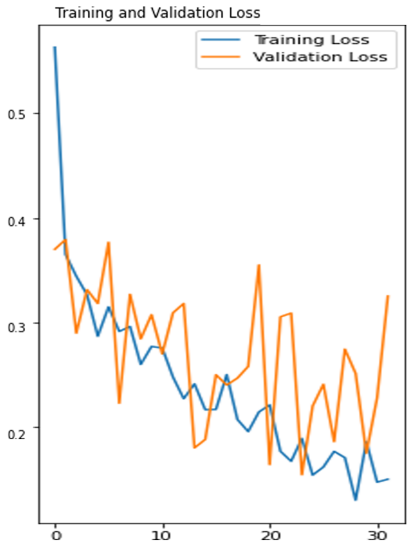

Fig. 3: Training and Validation Loss

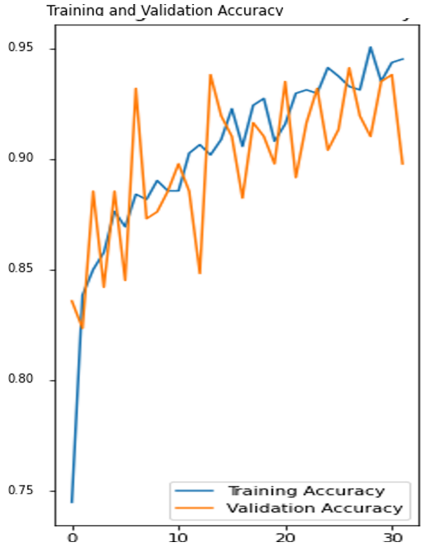

Fig. 4: Training and Validation Acc

[1] http://scenedetect.com/en/latest/

The scene detector allows trailers to be divided into intervals, mapped by the pairs of indices which indicates the beginning and the ending frame of a scene. However, for the fourth step, we consider image level or frame-level information of the scene intervals and we locate the frame in the middle of these intervals.

To eliminate unwanted intervals, we first created an image dataset composed of two classes; scenes to be removed and scenes to keep. Although the system could be created with only one class of scenes to be eliminated; we chose frames from the movies that might incorporate certain writings such as a closed up scene to a newspaper, or a signboard in the background to not eliminate noteworthy in-movie scenes (see Fig. 2). Subsequently, a CNN ResNet with 4 sequential residual blocks of 16, 32, 64 and 128 were used to train the model and acquired 0.9412 binary accuracy on the test data (see Fig. 4). As it can be seen from 3, validation loss decrease rate was significantly reduced after around 30th epoch. Although over-fitting after 30th epoch may seem as fast learning, this task can be considered simple enough to be learned in 30 epochs.

After the model is saved, every scene interval is passed through the model to either keep or remove. Lastly, black-and-white movie trailers are removed from the system and the dataset ended up with 1447 scenes in total that belongs to 234 video trailers. Half of these trailers are labeled as cartoon and the other half includes any other genre that is located in MovieLens20m. 117 movie trailers labeled as adventure, comedy, fantasy, children, romance, drama, action, crime, thriller, horror, mystery, sci-fi, documentary and musical genres are all labelled as non-animated movies. We labeled all of the trailer segments as their ancestors trailers (which they are partitioned from).

3 Methodology

Video processing can be considered as a tedious problem by computer engineers. Every frame is represented by a large number of pixels, which is the product of the height and the with of an image. Further, it is an even slower process considering that every second of a video is represented by 30 frames. For our model, we have prepared our own data generator in which we have optimized the video processing speed during learning period with a library called Decord[2]. The generator generates features for the model with dimensions of number of frames, image width, image height, number of channels (RedGreenBlue or RGB).

Since we split each trailer into several scenes, we considered each of the scenes as a different batch of the same genre. For each scene, we have extracted a set of sequential frames to be used in our model. There are various ways to implement the frame extraction method, however, we used division of each scene to a constant number of frames so that we can record the motion of scenes of different length. Our method involves a simple partitioning of the scene to an equal number of the frames explained in equation 3.

[2] https://github.com/dmlc/decord

$$par_{start} = \frac{scene_{end} + scene_{start} - frame\_rate^2}{2} \quad (1)$$

$$par_{end} = \frac{scene_{end} + scene_{start} + frame\_rate^2}{2} \quad (2)$$

$$frame\_list = (X_{par_{start}}, X_{par_{start}+frame\_rate}, \ldots, X_{par_{end}}) \quad s \quad (3)$$

$scene_{start}$: scene start frame index
$scene_{end}$: scene end frame index
par_{start} : partition start frame index
par_{end} : partition end frame index
$frame\_rate$: how many frames will be produced (given by user)
$frame\_list$: the frame list we end up with

In this equation, *partition start* refers to the starting point of the partition, which is calculated by the scene end point and the scene start points for each of the scene. This equation also allows us to partition the middle section of the scene with an optimum frame rate to reduce any misinformation frames caused by scene changes, such as fade-in or fade-outs. Partition end refers to the where the partition ends, and these frames were chosen with equal distances of frame rate. Frame rate corresponds to the how many frames will be extracted with this algorithm as an end result.

It is crucial to choose how many frames will be fed into the network since the motion is measured by the change between frames of the same scene. Our system allowed us to work with only even numbers because we divided each scene according to equation 3. Thus, we have experimented with a 2, 4, 6, and 8 for frame rates, and concluded that 2 is too short to give information about motion and 8 is too long that the difference between frames did not change much. Thus, a frame rate of 4 is found to be sufficient for capturing the motion and there is a noticeable change between the frames of the same scene.

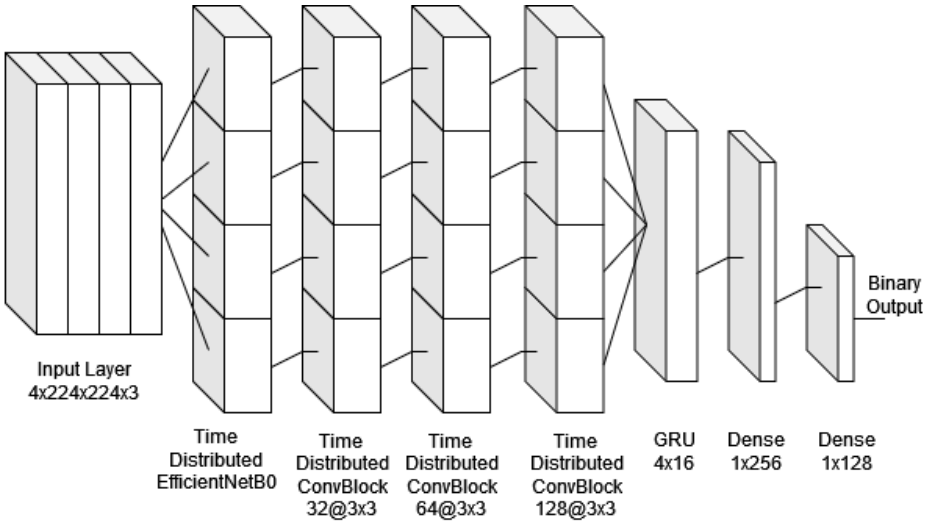

Fig. 5: Fused Model Architecture

To capture both spatial and temporal information, we have used 3 blocks of time distributed CNN with 32, 64 and 128 filters of 3 by 3; followed by a GRU of 16 units (see Fig. 5). We have also experimented with higher number of GRU units which resulted in poor results. To prevent fast over-fitting, we have accounted l1 and l2 kernel regularizers, and we included a batch normalization layer with momentum 0.9 after each convolutional block and a dropout layer. For hidden units, we have used relu activation. For the final Dense layer, we have used sigmoid since we give only one output which calculates the probability of the scene as cartoon with values close to 1.0 or photo-realistic with values close to 0.0. We have used Adam Optimizer with 0.0001 rate, loss function as binary crossentropy and metric of evaluation as binary accuracy.

To accelerate the learning process, we used EfficientNet architectures, which is proved to be very shallow and fast, yet achieves high top-1-accuracy on the ImageNet dataset and a state-of-the-art accuracy on 5 other transfer learning datasets, with an order of magnitude fewer feature size [15]. The main reason to use this transfer learning network is being explanatory for frames and efficient enough to execute in a Time Distributed fashion, since we need to apply it to every time frame extracted from a scene. We have also considered a few other transfer learning techniques, such as Residual Network (ResNet) and Very Deep Convolution Network (VGGNet). However, these models are not meant to do a profound search for objects in animated scenes and they demonstrated fast over-fitting learning due to their depth, without providing significant improvement on the validation set. Thus, fewer convolution layers are found to be helpful to discriminate cartoons from photo-realistic scenes.

4 Results and Evaluation

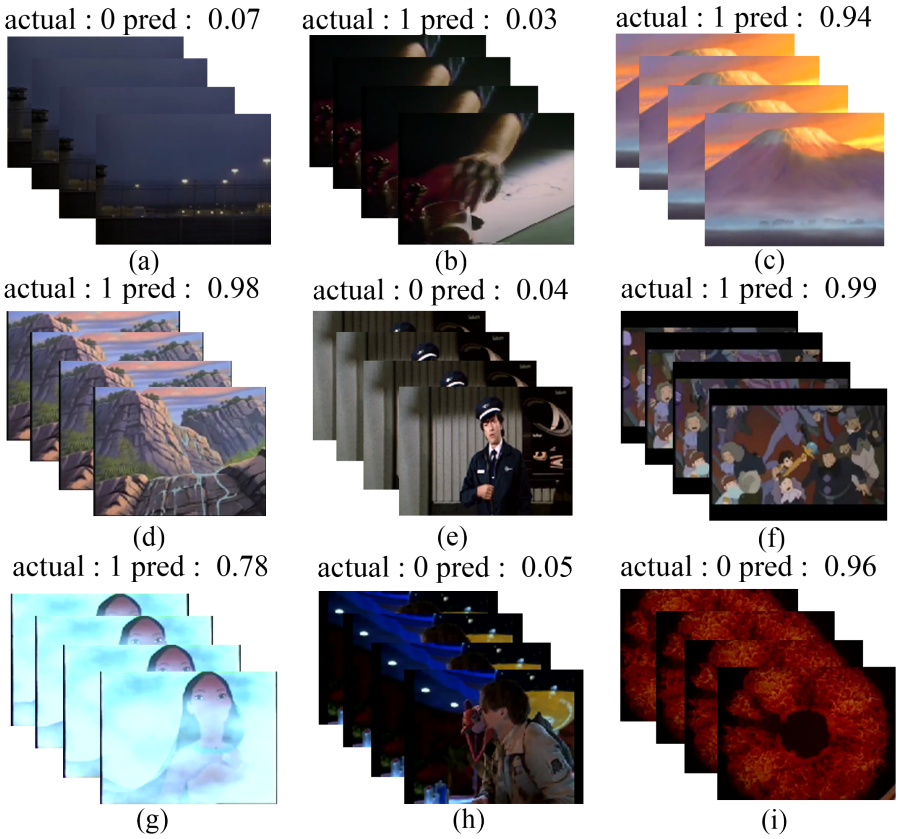

Fig. 6: System Prediction Results

In the training step, 1258 scenes extracted from 200 trailers are used to train and 189 scenes extracted from 34 trailers are used to validate the system. To prevent the testing of the model from a trailer scene that is already in the training data, trailers are separated as training and validation beforehand.

Some prediction examples of our final model is given in Fig. 6. We have labeled the animated scenes as label 1 and the non-animated ones as 0. The prediction result of a scene is an approximation of the probability of being labeled as animation. Our architecture is successful in detecting complex and crowded scenes such as Fig. 6.f. Despite low background information given in some photo-realistic scenes, the system could distinguish cartoon characters from photographic humans as it can be seen in Fig. 6.e.

On the other hand, there might be both animated scenes and photographic ones in the same trailer, and since we have labeled each scene of a trailer as the same label as the trailer, the labels were not provided to the system concisely. Consider the example given in validation set in Fig. 6.b, the scene that the trailer belongs to is a cartoon, but the scene itself is photographic. There are some scenes in the photo-graphic movie trailers that can resemble an animated scene like in Fig. 6.i. On the other hand, our model is successful in discriminating photo-graphic landscape in Fig. 6. a vs animated landscapes in Fig. 6.c-d, and could classify foggy scenes such as Fig. 6.g. Despite the challenging scenes we have given as examples, our model acquired **0.9552 binary accuracy** on our test set in 12 epochs which can be considered an outstanding result compared to the state-of-the-art. After 12 epochs, the over-fitting was observed and validation loss increased, although the training loss kept declining.

5 Conclusion

In conclusion, we have proposed a scene classification method of trailers with labels of either cartoon or photographic. We first built our dataset on top of the open-source dataset MovieLens20M, and segmented each trailer into scenes with shot-boundary detection. After pre-processing each scene with the elimination of inconsequential scenes and identifying several frames, we have trained our model involving CNN followed by GRU units and used transfer learning to advance the training process.

As our future work, we aim to increase the size of our dataset and evaluate the results of our model on other datasets. We also plan to use other transfer learning techniques and other video reading techniques to increase the efficiency of working on video data, since the training takes a long time. Moreover, we plan to use audio features and experiment if these features can further improve the reliability of our system.

Acknowledgements

We would like to express our special thanks to Dr. Sami Arpa, the founder of Largo Films company, for valuable discussions in the research and for sharing his knowledge about movie industry. [1]

References

1. Castellano, B.: Pyscenedetect. http://github.com/Breakthrough/PySceneDetect (07 2012)
2. Chen, F., Lai, M., Ye, Z.: Detecting cartoons: Automatic video genre classification. In: 2010 International Conference on Management and Service Science. pp. 1–4 (2010). https://doi.org/10.1109/ICMSS.2010.5576551

[1] Largo Films SA. EPFL Innovation Park, Building I. 1015, Lausanne/ SWITZERLAND. info [at] largofilms.ch

3. Chen, W., Shi, Y., Xuan, G.: Identifying computer graphics using hsv color model and statistical moments of characteristic functions. pp. 1123 – 1126 (08 2007). https://doi.org/10.1109/ICME.2007.4284852
4. Chen, Y., Lai, Y.K., Liu, Y.J.: Cartoongan: Generative adversarial networks for photo cartoonization. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 9465–9474 (2018). https://doi.org/10.1109/CVPR.2018.00986
5. Farid, H., Bravo, M.J.: Perceptual discrimination of computer generated and photographic faces. Digital Investigation **8**(3), 226–235 (2012). https://doi.org/https://doi.org/10.1016/j.diin.2011.06.003, https://www.sciencedirect.com/science/article/pii/S174228761100051X
6. Glasberg, R., Elazouzi, K., Sikora, T.: Video-genre-classification: recognizing cartoons in real-time using visual-descriptors and a multilayer-percetpron. In: The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005. vol. 2, pp. 1121–1124 (2005). https://doi.org/10.1109/ICACT.2005.246155
7. Harper, F.M., Konstan, J.A.: The movielens datasets: History and context. ACM Trans. Interact. Intell. Syst. **5**(4) (dec 2015). https://doi.org/10.1145/2827872, https://doi.org/10.1145/2827872
8. Ianeva, T., de Vries, A., Rohrig, H.: Detecting cartoons: a case study in automatic video-genre classification. In: 2003 International Conference on Multimedia and Expo. ICME '03. Proceedings (Cat. No.03TH8698). vol. 1, pp. I–449 (2003). https://doi.org/10.1109/ICME.2003.1220951
9. Ionescu, B., Lambert, P.: Classification of animated video genre using color and temporal information (2013)
10. Montagnuolo, M., Messina, A.: Automatic genre classification of tv programmes using gaussian mixture models and neural networks. In: 18th International Workshop on Database and Expert Systems Applications (DEXA 2007). pp. 99–103 (2007). https://doi.org/10.1109/DEXA.2007.92
11. Ng, T.T., Chang, S.F.: An online system for classifying computer graphics images from natural photographs. In: Delp, Edward J., I., Wong, P.W. (eds.) Security, Steganography, and Watermarking of Multimedia Contents VIII. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, vol. 6072, pp. 397–405 (Feb 2006). https://doi.org/10.1117/12.650162
12. Quan, W., Wang, K., Yan, D.M., Zhang, X.: Distinguishing between natural and computer-generated images using convolutional neural networks. IEEE Transactions on Information Forensics and Security **13**(11), 2772–2787 (2018). https://doi.org/10.1109/TIFS.2018.2834147
13. Roach, M., Mason, J., Pawlewski, M.: Motion-based classification of cartoons. In: Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No.01EX489). pp. 146–149 (2001). https://doi.org/10.1109/ISIMP.2001.925353
14. Sankar, G., Zhao, H.V., Yang, Y.H.: Feature based classification of computer graphics and real images. 2009 IEEE International Conference on Acoustics, Speech and Signal Processing pp. 1513–1516 (2009)
15. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks (05 2019)
16. Zhang, R., Quan, W., Fan, L., HU, L.m., Yan, D.: Distinguishing computer-generated images from natural images using channel and pixel correlation. Journal of Computer Science and Technology **35** (02 2020). https://doi.org/10.1007/s11390-020-0216-9