IJCCS (Indonesian Journal of Computing and Cybernetics Systems)
Vol.17, No.3, July 2023, pp. 291~300
ISSN (print): 1978-1520, ISSN (online): 2460-7258
DOI: https://doi.org/10.22146/ijccs.76058

Audio-Visual CNN using Transfer Learning for TV Commercial Break Detection

Muhammad Zha'farudin Pudya Wardana*[1], Moh. Edi Wibowo[2]
[1]Master Program in Computer Science, FMIPA UGM, Yogyakarta, Indonesia
[2]Department of Computer Science and Electronics, FMIPA UGM, Yogyakarta, Indonesia
e-mail: ***[1]muhammadzpw@mail.ugm.ac.id**, [2]mediw@ugm.ac.id

Abstrak

Permasalahan deteksi iklan pada media TV memiliki tantangan yang cukup sulit karena keragamanan jenis acara dan saluran TV. Penggunaan metode deep learning untuk deteksi iklan telah menunjukkan hasil yang cukup baik. Namun, deep learning membutuhkan waktu komputasi pelatihan yang lama dengan epoch pelatihan yang besar untuk mendapatkan akurasi tinggi.

Penelitian ini memanfaatkan transfer learning untuk mengurangi waktu pelatihan dengan membatasi jumlah epoch sebesar 20. Fitur data video yang digunakan yaitu dari aspek audio berupa Mel-spektogram dan aspek visual berupa frame. Dataset dikumpulkan dengan merekam beberapa program yang disiarkan di beberapa saluran TV nasional. Pre-trained model CNN MobileNetV2, InceptionV3, dan DenseNet169 dilatih kembali dan digunakan untuk deteksi pada level shot. Kemudian dilakukan pasca pemrosesan untuk mengelompokkan shot pada segmen iklan dan non-iklan.

Hasil deteksi shot terbaik diperoleh pada model Audio Visual CNN menggunakan transfer learning dengan akurasi 93,26% pada 20 epoch, melampaui hasil CNN tanpa transfer learning dengan akurasi 88,17% pada 77 epoch. Ditambah perbaikan pada pasca pemrosesan, hasil akhir Audio Visual CNN menggunakan transfer learning meningkat dengan akurasi 96,42%.

Kata kunci*—Iklan, TV, CNN, Transfer Learning, InceptionV3, MobileNetV2, DenseNet169*

Abstract

The TV commercial detection problem is a hard challenge due to the variety of programs and TV channels. The usage of deep learning methods to solve this problem has shown good results. However, it takes a long time with many training epochs to get high accuracy.

This research uses transfer learning techniques to reduce training time and limits the number of training epochs to 20. From video data, the audio feature is extracted with Mel-spectrogram representation, and the visual features are picked from a video frame. The datasets were gathered by recording programs from various TV channels in Indonesia. Pre-trained CNN models such as MobileNetV2, InceptionV3, and DenseNet169 are re-trained and are used to detect commercials at the shot level. We do post-processing to cluster the shots into segments of commercials and non-commercials.

The best result is shown by Audio-Visual CNN using transfer learning with an accuracy of 93.26% with only 20 training epochs. It is faster and better than the CNN model without using transfer learning with an accuracy of 88.17% and 77 training epochs. The result by adding post-processing increases the accuracy of Audio-Visual CNN using transfer learning to 96.42%.

Keywords*—Commercial, TV, CNN, Transfer Learning, InceptionV3, MobileNetV2, DenseNet169*

Received July 6th,2022; Revised July 30th, 2023; Accepted July 31th, 2023

1. INTRODUCTION

According to [1], TV is still the most effective media to promote products. Many companies spend billions of dollars to put their commercials on TV. On the other side, TV viewers may feel disturbed by the commercials because they periodically interrupted when we enjoy our favorite TV programs. Thus, an automatic commercial break detection system is developed.

Detecting commercials among various TV programs and channels is a challenging task. Because of many varieties of programs and commercials, it is hard to learn the pattern to recognize whether it is a commercial scene. Three methods have been done to make a commercial detection system: retrieval-based approach, hand-engineering the feature, and deep learning approach.

Retrieval-based approaches which have been done in [2] depend on the commercial dataset to make a detection. This method works very well for known commercials included in the dataset but will fail to detect if the new commercial brand appears. Thus, another method is introduced in [3], [4], and [5] by hand engineering the features and learning based on the extracted feature. This method does not depend on the commercial dataset, but it needs some predefined assumptions such as limiting the detection among homogeneous TV programs such as news, sport, etc.

Some studies in [6] and [7] use deep learning to create their commercial detection system. This method shows a good result for heterogeneous TV programs, and it does not depend on the commercial dataset to predict whether a block of video is a commercial. We provide some commercial examples for the model to learn, and it will generalize the unseen commercial from the dataset. However, detection methodology using deep learning takes a long time to train the model with a vast number of training epochs. As seen in [7], the model takes 55 epochs to make an accuracy of 82%.

Training data is an important part of training a deep learning model. So, picking the right features to be fed into a CNN architecture is also a key to successful work using deep learning. It is common to cut the video into shots using shot boundary detection which has also been done in previous works [3], [2], [4], [6], and [7]. Video has visual and audio aspects which we can be used to train deep learning models. As seen in [7], using a combination of the audio and visual parts will increase the accuracy. In this study, we use both audio and visual parts as the data fed into our proposed architectures.

Transfer learning is a concept to reuse existing deep learning models for another domain. In this study, we used some state-of-the-art CNN architectures like InceptionV3 [8], DenseNet169 [9], and MobileNetV2 [10] to detect commercial. We propose to design audio visual CNN which combines data from extracted Mel-spectrogram from audio data and picks a frame from the visual part. We use those CNN architectures as feature extractors for both audio and visual parts. Those pre-trained CNN models have a good basis for extracting image features. This motivates us to use it for commercial detection by re-training those CNN models with our commercial dataset. Hopefully, by having an initially good basis for extracting images, the retraining process will take a small number of epochs to achieve better accuracy.

2. METHODS

There are two phases of detection in this study: detection models at the shot level and post-processing to get segments of commercial and non-commercial. Pre-trained CNN architecture is used for detection models at the shot level. To build and train the audio-visual CNN model, **Figure 1** shows the overall methods used in this study. Firstly, the video data are collected by recording some programs on various Indonesia's TV channels. And then label the video as commercial and non-commercial. After that, we use shot boundary detection (SBD) to slice the videos into shots. After labeling and slicing, visual and audio are extracted to build an audio-

visual dataset. This audio-visual dataset is then fed to the pre-trained CNN architecture to detect commercials. Evaluation is done by collecting accuracy, precision, recall, and f1-score metrics.

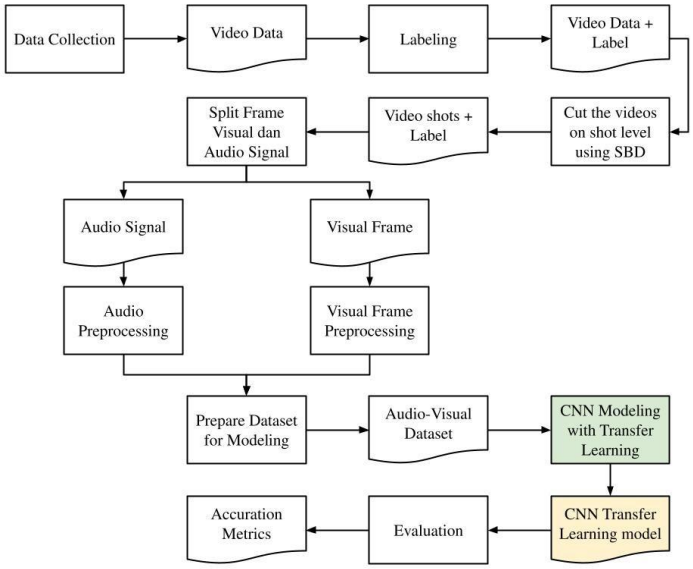


Figure 1 Overview of commercial detection method at shot level

After detecting commercial in shot level, post-processing is done to cluster commercial and non-commercial shots into commercial and non-commercial segments. **Figure 2** illustrates the overall framework for commercial detection proposed in this paper. In the next section, we will discuss the detailed methodology for each process mentioned in the block diagrams shown in **Figure 1** and **Figure 2**.

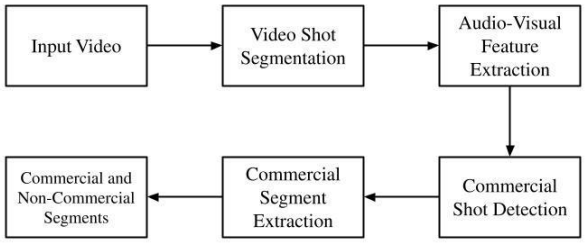


Figure 2 Framework for commercial detection at segment level

2.1. Data Collection and Labeling

Data collection is done by recording programs on various TV channels in Indonesia. There are five videos in total with various programs and commercials. The total video duration that we collect is **14 hours 47 minutes 33 seconds**. After collecting all videos, we label each segment inside each video file as commercial and non-commercial.

2.2. Shot Segmentation using Shot Boundary Detection

We use shot boundary detection to segment the videos into shots. Shot boundary detection is done by calculating the dissimilarity of each adjacent frame. First, we obtain the current frame and next frame and then calculate the histogram of each frame. We compute the histogram difference using Manhattan distance. By setting a certain threshold, we can obtain the boundary of a shot. The adjacent frame which has dissimilarity above the threshold will be the new boundary. This boundary information is used to slice the videos into shots.

2.3. Audio and Visual Data Pre-processing

In this phase, we have obtained the shots and its label. In this section, we will explain a detailed process to extract an audio and visual representation of a shot so that it can be fed into our CNN model.

2.3.1. Audio data pre-processing

The audio signal is 1-dimensional data that cannot be fed directly to a CNN model. We choose the Mel-spectrogram representation of the audio as the input of our CNN model. A spectrogram is a time-frequency dimensional representation of audio which is common to represent audio to be fed into a CNN architecture. First, we extract the audio signal from the shot and then transform the whole audio shot into a spectrogram using a short-time fourier transform (STFT). We apply a Mel scale transformation to create Mel-spectrogram. The spectrogram of different shot duration will have different sizes of Mel-spectrogram. After that, we resize the Mel-spectrogram into 160x160 pixel images.

In the previous work such as in [7], the audio is resampled before transforming into a spectrogram so it will result in the same size of spectrogram despite different shot durations. In this paper, we propose a different approach to handle different shot durations. We analyzed our data that generally, the duration of commercial shots is relatively shorter than non-commercial ones. By using our method, we can embed the duration information in the Mel-spectrogram representation. **Figure 3** shows that a shot with a long duration has a tighter representation than a shot with a short duration.

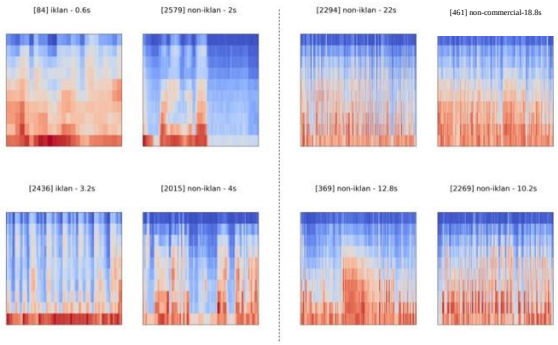


Figure 3 Mel-spectrogram representations of different shot durations

2.3.2. Visual data pre-processing

For the visual part, we do two things to make a visual representation of the shot: pick one frame for each shot and resize the frame into 160x160 pixels. We pick the center frame of a shot to represent the shot. This approach will take smaller computation complexity than in previous work [6] which needs four frames fed into CNN for shot level detection. In this stage, we have a visual representation of a shot: a frame with its label which is ready to be fed into a CNN model.

2.3.3. Training, validation, and test dataset splitting

We have five videos mentioned before. We divide four videos as training and validation dataset and one video for testing. We choose one video that the model has not seen before to test the generalization ability of our CNN model. We do a stratified random sampling for training and validation so that we can maintain the balance between the number of commercial and non-commercial shots. And then split 2/3 for training and the rest for validation. The total shot from the whole dataset is 15,187 shots. Figure 4 shows training, validation, and testing dataset distribution. It is balanced between the number of commercial and non-commercial shots

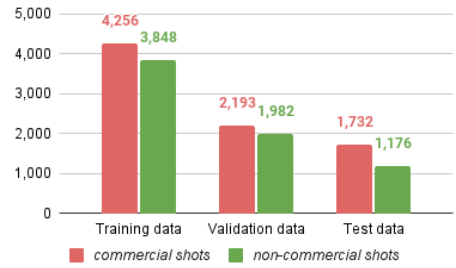
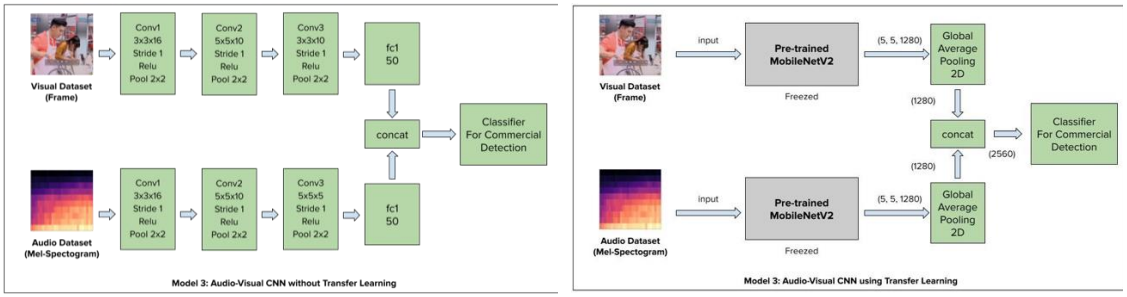


Figure 4 Training, validation, and test dataset distribution

2.4. Audio Visual CNN Architecture

In this paper we propose two branches of CNN processing both audio and visual features. We use the transfer learning method by using existing pre-trained CNN architecture such as MobileNetV2, InceptionV3, and DenseNet169 as a feature extractor for both branches. Those CNN are pre-trained in a large ImageNet dataset. To compare with preliminary work without using transfer learning, we also re-implement the Audio-Visual CNN architecture proposed by Minaee *et al.* [7].



(a) (b)

Figure 5 (a) Audio-Visual CNN architecture without transfer learning [7] (b) Audio-Visual CNN architecture using transfer learning proposed in this paper

Figure 5 shows Audio-Visual CNN architectures both using transfer learning and not using transfer learning. We remove the fully connected layer of the pre-trained model; the last layer will be a feature map in a certain dimension. After visual and audio data are fed into the CNN branch, the last convolutional layer of the pre-trained model produces a feature map regarding what CNN architecture we use. For example, if we use MobileNetV2, the last feature maps will be in a dimension of (5,5,1280). We use Global Average Pooling 2D to reduce the dimension to 1280. Those 1280 features represent an audio or visual feature. To combine features from both audio and visual, we concatenate the features so that now we have 2560 audio-visual features to be fed into a fully connected layer for running a binary classification.

In the example above, 2560 neurons are fed into an input of the fully connected layers. We add only one hidden layer with 100 neurons. We also use the ReLU activation function on each hidden layer's neurons and a dropout of 0.2 to prevent overfitting. To further prevent overfitting, we also add L2-regularization on the hidden layer. After that, we have one neuron for the output layer to output a binary number: 1 for commercial and 0 for non-commercial. To train the whole model, we use binary cross-entropy as the loss function shown in **equation (1)** and Adam optimizer [11] to minimize the loss for training.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log\big(p(y_i)\big) + (1-y_i) \cdot log\big(1-p(y_i)\big) \quad (1)$$

Besides Audio-Visual CNN, we also implement and test the detection model using Audio-only and Visual-only CNN. The architecture difference from **Figure 5** is that Audio or Visual only CNN only has one branch so there will be no concatenation happening. The audio or visual features resulting from the convolutional layers will be directly fed into the fully connected layers for classification. We test whether a combination of audio and visual will increase the accuracy of the commercial detection.

2.5. Post-Processing to Extract Segments

Using a heuristic method, post-processing results segments of commercial and noncommercial. The algorithm starts with setting a minimum consecutive shots threshold to cluster a segment. It will look up for subsequence with the same label consecutively and build a segment if the consecutive subsequence meets the minimum consecutive shots threshold. We also use the transition between a segment to another segment as a segment boundary to know the start time and the end time of a segment. The post-processing algorithm can also refine false detection from audio-visual CNN at shot level, thus increasing the accuracy of the entire system. **Figure 6** shows the refinement for false predicted shots in the middle of a segment using post-processing.

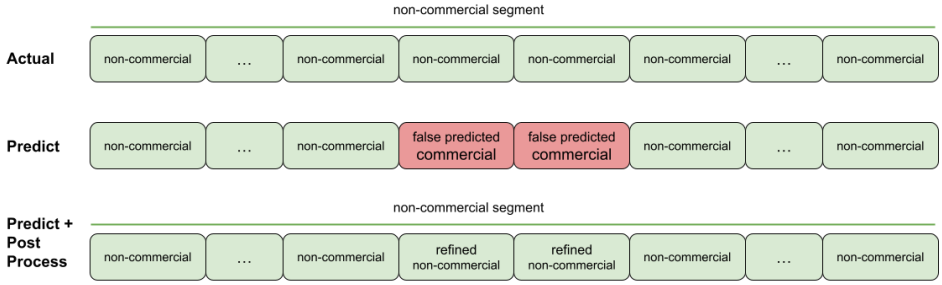


Figure 6 Refinement for false detected shots in the middle of a segment

2.6. Testing and Evaluation

Model evaluation is done by testing through the testing dataset and calculating the accuracy metrics. First, we calculate the confusion matrix as ground truth for calculating other metrics. From the confusion matrix shown in **Table 1**, we can obtain accuracy as shown in equation (2), precision as shown in equation (3), recall as shown in equation (4), and f1 score as shown in equation (5).

Table 1 Confusion matrix

	Prediction: Commercial	**Prediction: non-commercial**
Actual: commercial	*TP*	*FN*
Actual: non-commercial	*FP*	*TN*

$$Acc = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{2}$$

$$\text{Precision} = \frac{(TP)}{(TP + FP)} \tag{3}$$

$$Recall = \frac{(TP)}{(TP + FN)} \tag{4}$$

$$\text{F1} = 2 \times \frac{(Recall \times Precision)}{(Recall + Precision)} \tag{5}$$

3. RESULTS AND DISCUSSION

The parameters tested in this paper are shown in **Table 2**. First, from a feature aspect, we want to know if fusing audio and visual features will increase accuracy. Secondly, we've implemented two types of CNN model architectures: without transfer learning and with transfer learning. We want to know if the transfer learning method can increase accuracy and reduce training epoch at the same time. And the last parameter to be tested is the retraining mechanism for the transfer learning model. We want to know the impact of retraining some portions of the pre-trained CNN versus the whole CNN layers.

Table 2 Testing parameters

No.	Parameter	Testing candidate
1.	Features	a. Audio only b. Visual only c. Audio + Visual
2.	CNN architecture	a. Ad-Net [7] (without transfer learning) b. MobileNetV2 [10] (transfer learning) c. InceptionV3 [8] (transfer learning) d. DenseNet169 [9] (transfer learning)
3.	Retraining mechanism	a. Without fine-tuning b. Fine-tune at the last CNN block c. Fine-tune the whole CNN layers

3.1. Non-Transfer Learning Model Experimental Results

The experiments for non-transfer learning CNN models are done by testing the model trained in 20 epochs and testing again with the model trained until the model converges. **Table 3** shows the best accuracy for 20 training epochs is obtained with visual CNN with 85.59% accuracy. However, the best accuracy is obtained with audio-visual CNN after training the models until they converge as shown in **Table 4**. We have the best result for the non-transfer learning model with 88.17% of accuracy with 77 training epochs.

Table 3 Non-transfer learning CNN model performance after 20 epochs

Model	Accuracy	Precision	Recall	F1-score	Total Epoch
Visual	**0.8559**	0.8806	**0.8770**	**0.8788**	20
Audio	0.7246	0.8221	0.6859	0.7479	20
Audio + Visual	0.8267	**0.9367**	0.7604	0.8394	20

Table 4 Non-transfer learning CNN model performance after converging

Model	Accuracy	Precision	Recall	F1-score	Total Epoch
Visual	0.8528	0.8890	0.8603	0.8744	93
Audio	0.7077	0.8331	0.6368	0.7219	46
Audio + Visual	**0.8817**	**0.9166**	**0.8816**	**0.8988**	77

3.2. Transfer Learning Model Experimental Results

We run our experiment for the transfer learning model with 20 training epochs. Then we freeze the pre-trained CNN models and train new fully connected layers for commercial detection on the first 10 epochs. And then, we experiment with the next 10 epochs for the re-training mechanism. In the first experiment, we continue 10 epochs without fine-tuning the pre-trained CNN. In the second experiment, we continue the next 10 epochs by fine-tuning only the last pre-trained CNN block. In the third experiment, we unfreeze and re-train all pre-trained CNN layers. We do all those three experiments in MobileNetV2, InceptionV3, and DenseNet169. **Table 5** shows one of the experimental results done with MobileNetV2, the rest CNN models show a similar result. We can conclude that the best accuracy for transfer learning can be achieved by re-training all layers in the pre-trained CNN.

Table 5 Experimental results on retraining mechanism for transfer learning

No.	CNN Architecture	Fine-tuning	Accuracy	Precision	Recall	F1-score
1	Audio-Visual MobileNetV2	All	**0.9326**	**0.9321**	**0.9276**	**0.9297**
2	Audio-Visual MobileNetV2	Last Block	0.8649	0.8863	0.8868	0.8866
3	Audio-Visual MobileNetV2	None	0.8432	0.8688	0.8678	0.8683

We compare the performance results of the transfer learning model with retraining all layers as shown in **Table 6**. From these experiments, we also see the pattern that combines both audio and visual features can increase the accuracy of the transfer learning models. MobileNetV2 shows the best performance in terms of accuracy. In another metric, both InceptionV3 and MobileNetV2 got the same highest F1 score. DenseNet169 on the other hand, despite the lowest overall accuracy and f1, shows the best precision score among the three.

Table 6 Transfer learning CNN model performance after 20 epochs by retraining all pre-trained CNN layers

No.	CNN Architecture	Accuracy	Precision	Recall	F1-score
I.	**MobileNetV2**				
a.	Audio	0.8566	0.8704	0.8920	0.8811
b.	Video	0.9154	0.9231	**0.9359**	0.9295
c.	Audio + Video	**0.9326**	**0.9321**	0.9276	**0.9297**
II.	**InceptionV3**				
a.	Audio	0.8184	0.8199	0.8909	0.8539
b.	Video	0.9054	0.9288	0.9111	0.9198
c.	Audio + Video	**0.9164**	**0.9321**	**0.9273**	**0.9297**
III.	**DenseNet169**				
a.	Audio	0.8404	0.8743	0.8551	0.8646
b.	Video	0.9099	0.9200	**0.9296**	**0.9248**
c.	Audio + Video	**0.9127**	**0.9562**	0.8943	0.9242

Audio-Visual CNN models in **Table 6** use the same architecture for both audio and visual branches. We also mix some architecture on the audio and visual branches with different CNN architecture. The results are shown in **Table 7**. MobileNetV2-MobileNetV2 architecture still got the best accuracy, but DenseNet169-InceptionV3 architecture got the highest recall and F1 score.

Table 7 Result of mixed audio-visual branches of the CNN architectures

No	Audio	Visual	Accuracy	Precision	Recall	F1-score
1	MobileNet	MobileNet	**0.9326**	0.9321	0.9276	0.9297
2	DenseNet	Inception	0.9298	0.9463	**0.9353**	**0.9408**
3	MobileNet	Inception	0.9192	0.9469	0.9157	0.9310
4	Inception	Inception	0.9164	0.9321	0.9273	0.9297
5	DenseNet	MobileNet	0.9144	0.9390	0.9157	0.9272
6	DenseNet	DenseNet	0.9127	**0.9562**	0.8943	0.9242

3.3. Post Processing Results

As mentioned earlier, in addition to extracting the segments of commercial and non-commercial, the post-processing stage also refines the false detection at the shot level. Thus, we compare the result before post-processing and after post-processing as shown in **Table 8**. The overall performance metrics are increased. Overall, our proposed algorithm obtains the best accuracy for commercial detection with an accuracy of 96,42%.

Table 8 Accuracy increased after applying post-processing

No	Model	Accuracy	Precision	Recall	F1-score
1	Audio-Visual CNN	0.8817	0.9166	0.8816	0.8988

2	Audio-Visual CNN + Post Process	0.9340	**0.9771**	0.9105	0.9426
3	Audio-Visual CNN Transfer Learning	0.9326	0.9321	0.9276	0.9297
4	Audio-Visual CNN Transfer Learning + Post Process	**0.9642**	0.9537	**0.9879**	**0.9705**

3.4. Inaccurate Detection Cases Evaluation

We investigate more about inaccurate detection cases produced by our system. From 96.42% of accuracy, we found some errors appear on the edge of a segment and the other is purely from shot level detection which detect inaccurate consecutive shots more than the minimum consecutive threshold for building a segment. **Figure 7** shows a case from a TV program called "Ini Talkshow". The opening and the closing of a segment in this program show a graphic with a typography design that says "Ini Talkshow" with a music background. In the shot level detection, our Audio-Visual CNN model detects this shot as commercial. When post-processing is done, these inaccuracies cannot be refined because the previous consecutive shots are also inside a segment of the commercial. Thus, these false predicted shots will still be detected as commercial. This also resulting a little inaccuracy in the start timestamp and the end timestamp of the segment. This illustrates the short coming of our post-processing algorithm which cannot refine shot detection inaccuracies at the edge of a segment.

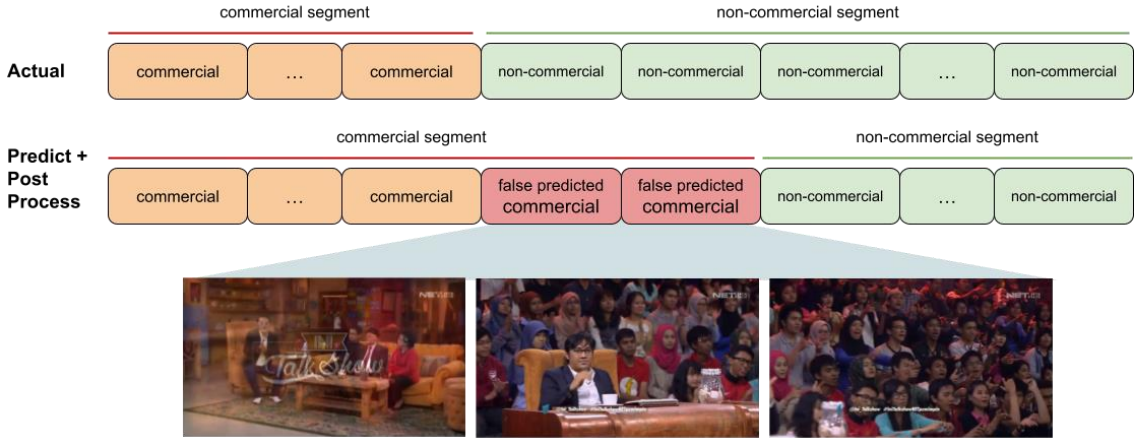


Figure 7 Inaccurate shot level detection at the edge of a segment

4. CONCLUSIONS

In this paper, we propose Audio Visual CNN using transfer learning for a commercial detection system. Our system consists of three modules: shot boundary detection to automatically slice the video into shots, detection module at shot level, and post-processing to cluster the classified shots into segments of commercial and non-commercial. We show that fusing both audio and visual features can increase accuracy. The shortcoming of fusing the two features is that it doubled the model complexity. The usage of the transfer learning method increases the detection accuracy at the shot level and decreases the training epoch. By using transfer learning, we obtain an accuracy of 93.26% with only 20 training epochs, while the accuracy by not using transfer learning is 88.17% trained in 77 training epochs. We test our system using the pre-trained CNN MobileNetV2, InceptionV3, and DenseNet169. The best accuracy is obtained by retraining all layers of the pre-trained model. Our post-processing module can also refine the false detected shots in the middle of a segment. Thus, by using the post-processing module, the accuracy of our proposed algorithm for commercial detection increased to 96.42%.

REFERENCES

[1] S. Li Yujuns and Luo, "A TV Commercial Detection System," in *Web Information Systems and Mining*, 2011, pp. 35–43.

[2] X. Wu and S. Satoh, "Ultrahigh-Speed TV Commercial Detection, Extraction, and Matching," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 6, pp. 1054–1069, 2013, doi: 10.1109/TCSVT.2013.2248991.
[3] Z. Feng and C. Lab, "Real Time Commercial Detection in Videos," 2013.
[4] A. Vyas, R. Kannao, V. Bhargava, and P. Guha, "Commercial Block Detection in Broadcast News Videos," 2014. doi: 10.1145/2683483.2683546.
[5] A. Gomes, M. P. Queluz, and F. Pereira, "Automatic detection of TV commercial blocks: A new approach based on digital on-screen graphics classification," in *2017 11th International Conference on Signal Processing and Communication Systems (ICSPCS)*, 2017, pp. 1–6.
[6] M. Li, Y. Guo, and Y. Chen, "CNN-Based Commercial Detection in TV Broadcasting," in *Proceedings of the 2017 VI International Conference on Network, Communication and Computing*, 2017, pp. 48–53. doi: 10.1145/3171592.3171619.
[7] S. Minaee, I. Bouazizi, P. Kolan, and H. Najafzadeh, "Ad-Net: Audio-Visual Convolutional Neural Network for Advertisement Detection In Videos," *ArXiv*, vol. abs/1806.08612, 2018.
[8] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.
[9] G. Huang, Z. Liu, and K. Q. Weinberger, "Densely Connected Convolutional Networks," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, 2017.
[10] M. Sandler, A. G. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.
[11] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, 2015.