A Comparative Study of Classifiers for Thumbnail Selection

Kyle Pretorious
Department of Computer Science
University of Pretoria
Pretoria, South Africa
u16234805@tuks.co.za

Nelishia Pillay
Department of Computer Science
University of Pretoria
Pretoria, South Africa
npillay@cs.up.ac.za

***Abstract*—As we move into the fourth industrial revolution video streaming platforms like Netflix are turning to machine learning techniques to maintain a competitive edge in the market. Various problems such as clip creation, network optimization, customer churn prediction, amongst others, have been solved for video streaming platforms using machine learning. This paper focuses on automatic thumbnail selection for movies and series. Classifiers are used to automate the thumbnail selection. The research firstly compares the performance of different convolutional neural networks (CNNs), namely, VGG-19, Inception-v3 and ResNet-50, for solving this problem. The performance of two classifiers, namely, the best performing convolutional neural network and a hybrid approach combining a CNN and genetic programming, are compared for thumbnail selection. The CNN is used for feature extraction and genetic programming for classification. The ResNet-50 CNN outperformed the other CNNs. Both classifiers were successful for thumbnail selection with the convolutional neural network outperforming the hybrid classifier.**

***Index Terms*—automatic thumbnail selection, convolutional neural networks, genetic programming**

I. Introduction

With advances in technology making reliable and inexpensive internet connections available to a larger percentage of the world than ever before, video streaming platforms have become highly popular. A key feature present on every video streaming site is the use of video thumbnails. Video thumbnails are cover images that attempt to capture the content of the video in a visually appealing way. These thumbnails do not only make it easier for users on these platforms to find videos they are looking for [1], but a strong positive correlation exists between the quality of a video thumbnail and the number of views these videos receive [2].

Since the task of manually searching through thousands of frames for thumbnails can become a time consuming task, creating programs that are able to automatically select thumbnails for videos have been studied in recent years [3]–[6]. Automatic thumbnail selection is a complex problem for programs to solve since the characteristics of a good thumbnail cannot be objectively defined. A thumbnail should be able to capture the attention of viewers and interest them in such a way that they decide to view the video.

Automatic thumbnail selection is a fairly new problem and hence not well researched. Existing research into solving the problem of thumbnail selection consists of deterministic approaches focused on extracting key frames from videos, and convolutional neural networks. Given that convolutional neural networks are considered to outperform other techniques, the paper examines a convolutional neural network solution to the problem. The research presented in this paper differs from previous work in that different architectures of convolutional networks are investigated for this problem. Genetic programming [7] is an evolutionary algorithm, that has proven to be successful at classification, that has not previously applied to this problem. The paper also examines hybridizing a convolutional neural network with genetic programming, where the convolutional neural network performs feature extraction and the genetic programming algorithm classification. The performance of the hybrid is compared with that of the best performing convolutional neural network for thumbnail selection. Hence, the main contributions of the research presented in this paper are:

- An investigation of classifiers for solving the thumbnail selection problem which has not been sufficiently researched.
- A comparison of convolutional neural network architectures for thumbnail selection.
- A comparison of performance of convolutional neural networks and a hybrid approach combining a CNN for feature extraction and genetic programming for classification for thumbnail selection.

The following section describes the automatic thumbnail selection problem and section III presents an overview of previous studies attempting to solve this problem. Section IV presents the convolutional neural networks employed for thumbnail selection. The hybrid approach combining a CNN and genetic programming is presented in section V. The experimental setup used to evaluate the CNNs and the hybrid approach are presented in section VI. Section VII describes and compares the performance of the CNNs and the hybrid approach. A summary of the findings of the research and future research is presented in section VIII.

II. The Automatic Thumbnail Selection Problem

Thumbnails are cover images that are used to represent a video or attract viewers, and are commonly used on video

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

streaming platforms where users browse through many possible videos to watch. On competitive streaming sites, such as Youtube, thumbnails have shown to play an important role in the number of views that a video receives, and content creators that are more successful generally have more attractive thumbnails for their videos [2]. Studies such as [1] have shown that thumbnails have a large impact on the rate at which users are able to find content they are searching for on platforms. Popular streaming platforms where there is no competition between content creators, such as Showmax and Netflix, use thumbnails to improve user experience by making it easier for users to search through content. Selecting a good thumbnail for a video becomes more difficult as the length of the video increases, since the person selecting the thumbnail should watch the entire video to be able to consider all the frames. On large platforms, such as Netflix and Showmax, with more than 7000 videos with an average length of more than an hour, manually selecting a thumbnail for each video becomes a task that requires many hours of manual labour.

A solution to this problem is using a program that can automatically select or suggest thumbnails for a video. Optimally, such a program should be able to consider every frame in the video and rank each frame based on how well it meets the criteria of a good thumbnail. A good thumbnail can be seen as one that captures the content of the video while being visually appealing and evoking the interest of the viewer. However, considering every frame in a high-quality video of more than an hour becomes very computationally expensive, and therefore frames are sampled from the video by some sampling techniques. Automatic thumbnail selection can easily be converted into a binary classification problem with the two possible classes being good and bad thumbnails. The good class score of a frame can be used to rank each frame and either the best frame can be automatically set as the thumbnail for the video or a safer approach can be taken where the top set of frames are reviewed by a human and the best is selected.

III. Related Work

The problem of automatic thumbnail selection is in essence highly similar to keyframe selection. Keyframe selection places a higher importance on selecting frames that are representative of a video, in contrast to thumbnail selection that focuses more on attractive frames. The recent work on keyframe selection of Ren et al. [8] uses a Siamese CNN architecture together with a Piecewise ranking loss function to learn the ranking of frames in short videos to select keyframes.

A method for automatic thumbnail selection that focuses on selecting frames that are both representative and attractive is described in [6]. This multi-step method first filters out transition frames and frames of low quality, after which the remaining frames are clustered and frames with the highest stillness are selected from each cluster. The last step selects the video thumbnail from the remaining keyframes by considering both the frame attractiveness and relevance.

A thematic approach to thumbnail selection is proposed in [5]. This is accomplished by using keywords related to the video to retrieve related images from a visual database, after which thumbnails are selected by taking into account the similarity of frames with the retrieved images.

Work has also been done in dynamically selecting video thumbnails or keyframes based on semantic information contained in the video title or the search query used to find the video [3], [4], [9], [10].

The work that is considered to be the most related to the work presented in this paper is that of Arthurs et al. [2]. In this study, two convolutional neural networks were applied to this problem, a variant of AlexNet and a retrained VGGNet model that was pre-trained on ImageNet. Techniques such as dropout, L2 regularization and data augmentation were used to prevent overfitting. The neural networks were tested on a dataset from Youtube, where examples of good thumbnails were obtained from videos with more than 1 million views, and examples of bad thumbnails from videos with less than 100 views. The neural networks obtained accuracies of 80.9% and 86.5% respectively.

The research presented in the paper differs from previous work in terms of :

- The study compares different convolutional neural network architectures for automatic thumbnail selection.
- Investigates the effectiveness of hybridizing a convolutional neural network, for feature extraction, and genetic programming for classification, for automatic thumbnail selection.

The following section describes the convolutional neural networks investigated.

IV. Convolutional Neural Network for Thumbnail Selection

The CNN operates as a classifier that outputs a value between 0 and 1 using the sigmoid activation function on the output neuron. Frames that would make good thumbnails should be assigned a value near 1, while frames that would make bad thumbnails should be assigned a value near 0. After being trained on a data set containing examples of good and bad thumbnails, the CNN is then applied to each frame within a video to assign a score to each frame. Frames are then ranked according to the scores assigned to them using the classifier and the top set of frames can be suggested as thumbnails for the given video. In this work, three different CNN architectures were implemented for thumbnail selection, namely, VGG19 [11], Inception-V3 [12] and ResNet50 [13]. The performance of the neural network architectures for thumbnail classification are compared with the architecture found to perform best described in section VI.

V. Hybrid Approach

This section describes the hybrid approach combining a CNN and genetic programming. While genetic programming has shown to be effective at classifying images, one of the challenges is genetic programming cannot deal with the large number of features [14], [15] . For this reason feature extraction has to be performed before applying the genetic

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

programming classifier. In the proposed approach, a CNN, namely, VGG19 [11] has been employed for feature extraction.

The CNN is firstly trained on an initial set of data. The dense layers of the CNN are removed up until the layer that represents the flattened feature maps produced by the CNN. Given a data set, each image can be run through the convolutional layers to obtain a 1D vector representing the extracted features, creating a new data set where each image is only represented by the 1D feature vector. GP is then trained to produce a classifier that is able to predict the class of the image represented by the feature vector by only considering the feature vector and not the original image itself.

Genetic programming evolves a classifier to classify each frame in a movie or series as a good or bad thumbnail, in a similar manner as that described for CNNs. From the literature surveyed it is evident that arithmetic tree classifiers are most effective for image processing [14]–[17] when employing genetic programming. Furthermore, arithmetic trees are not only able to produce continuous outputs as required for the scoring of frames, they are also able to effectively operate on the continuous inputs produced by the feature extraction phase. Hence, each classifier is an arithmetic tree which produces a numerical value which is transformed to a score between 0 and 1 using a sigmoid function, with a score of 1 indicating that a frame would make a good thumbnail. Each element of the initial population is created using the ramped half-and-half method [7]. The terminal set is comprised of the features in the 1D feature vector produced by the CNN as well as ephemeral constants in the range -1 to 1. The function set includes the following arithmetic operators: addition, subtraction, multiplication, protected division, min and max. The fitness of an individual is calculated as the log loss obtained when using by the individual to classify a sample drawn from the training set. Tournament selection [7] is used to select parents which the mutation and crossover operators are applied to, to create the offspring of each generation. Subtree mutation uniformly selects a node within the parent tree. A random subtree is then generated and the subtree rooted at the selected node is replaced by the generated subtree. Crossover is applied to two parents, where the first parent will act as the main parent and the second parent will act as the donor parent. A node is uniformly selected from each of the two parents. After which, the subtree rooted at the selected node in the main parent is replaced by the subtree rooted at the node selected in the donor parent. The new tree created is then returned as the offspring. The process of fitness evaluation, selection and application of the genetic operators continues for a set number of generations.

VI. Experimental Setup

This section provides an overview of the experimental setup used to evaluate both the classifiers for thumbnail selection.

A. Data Set

The data set used to train the classifiers plays a large role in the ability of the classifiers to select thumbnails. This is because the classifiers will learn the mapping between good and bad thumbnails as it is present in the data set.

It was found that there were not any publicly available data sets for this problem that were of a high quality. For this reason, a data set was created for the purpose of this study. Since the data set has to be relatively large to allow for the two methods being compared to effectively train on the data set without overfitting, the process of the creating the data set was done automatically. This was done using Youtube's developer API. Using this API it was possible to retrieve the video IDs of popular movie and series trailers that have been uploaded to Youtube. Movies and series were chosen as the content to focus on since thumbnails are typically required for them in an commercial environment where automatic thumbnail selection would be applicable. Using the list of video IDs it was possible to once again query the Youtube API for three thumbnails selected by Youtube's automatic thumbnail selection algorithm from the start, middle and end of the video respectively. These thumbnails served as examples of good thumbnails. To obtain examples of bad thumbnails, three frames were randomly selected from the same video. The reason for this is that random frames within a video have a considerably higher chance of making a bad thumbnail than a good one. Hence, using this technique, it was possible to obtain 3 examples of good thumbnails and 3 examples of bad thumbnails from each video.

A total of roughly 2500 thumbnail examples were obtained using this method, where the examples were close to being equally divided between the two classes. To further improve the data set, the examples were manually inspected and any examples that were found to be in the wrong class were moved to the opposite class. After which, the data set still remained roughly balanced between the two classes. Finally, the data set was split into a training set consisting of 75% of the examples and a validation set consisting of the remaining examples.

To allow CNNs to train faster, input images were normalized by dividing each pixel value by 255. Since the data set is relatively small in size, it would be ideal to artificially increase the size of the data by use of data augmentation techniques such as random shearing, scaling, changes in brightness and flipping the images. However, due to the nature of thumbnails, this was not possible. This is because performing data augmentation to good examples of thumbnails would likely impact the visual appearance of the thumbnail negatively. For this reason, data augmentation was not performed.

B. Performance Metrics

The metric used to measure the performance of the binary classifiers produced is the binary cross-entropy error, otherwise known as log loss. Log loss is effectively a measure of the similarity of the probability distribution of the output produced by the classifier with the distribution of the target variables. Hence, a lower log loss indicates that the classifier is able to more accurately predict the output class. This metric will be used as the primary metric for any two classifiers. More specifically, the best validation loss obtained by the

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

classifier during training will be used as the validation loss for comparison. This is because the classifiers are not guaranteed to have reached their best performance at the end of training, but could likely reach their best performance during training. Secondly, the training and validation accuracy will also be provided as a metric that is more interpretable, but will not be used to make final decisions when comparing classifiers.

C. CNN Parameter Tuning

The CNN architecture comparison and parameter tuning performed is discussed in detail in this section, where the architecture and parameters that were found to perform the best are selected for the performance comparison. The CNN architectures chosen for comparison in this study have been chosen due to their diversity in structure and performance in the ILSVRC [18]. The following architectures are compared:

- VGG-19: A 19 convolutional layer deep VGG network [11]
- Inception-v3: The third iteration of the Inception architecture [12]
- ResNet-50: A 50 convolutional layer deep ResNet [13]

For each architecture the following parameter values were chosen based on observations made during training:

- Number of epochs and batch size: To improve training, the batch size was set to the maximum size allowed by the memory capacity of the machine used for training. This was found to be a batch size of 16. The number of epochs trained for were determined by examining the training history of the neural network. It was observed that models generally converge at around 20 epochs, after which they start overfitting. For this reason, the number of epochs were set to 30 with the best validation loss of the model being tracked over the number of epochs, and only saving the version of the model that obtained that loss.
- Regularization: To further prevent overfitting, heavy regularization was applied to the dense layers of the network. This was accomplished by using dropout with a rate of 0.5 on the final flattened feature map of the convolutional layers and the hidden layer.
- Batch Normalization: Batch normalization was applied to all layers within the dense layers of the network. This was done to improve training and for the regularizing effect that batch normalization has shown to have.

The following parameters were empirically tested for each architecture:

- Dense Layers Architecture: The architecture of the dense layers in terms of the number of layers and the number of neurons per layer was tuned. Each CNN was given a single output neuron using a sigmoid activation function, with the other neurons within the dense layers using ReLU activation functions. It was observed that all CNN architectures were able to easily overfit the training data, this is likely because the data set is relatively small and that no data augmentation could be applied. It was found that CNNs using simple dense layers with only a single hidden layer of 64 neurons were able to overfit, due to the fact that the convolutional layers within the CNNs contain such a large number of free parameters. For this reason, the dense layers of all architectures were restricted to a single hidden layer and an output layer with a single neuron. Furthermore, the number of neurons in the hidden layer were tested at 64 neurons to allow slight overfitting and 32 neurons in an attempt to reduce overfitting.
- Optimization Algorithm: Two optimization algorithms used for training were tested and compared for each architecture and dense layer combination. The two methods that were compared are SGD [19] and Adam [20]. Additionally, different values for the hyper parameters of these methods were tested based on previous experience.
- Transfer learning: Lastly, the effect of transfer learning was tested on architecture and parameter combinations that tended to perform well. This was done by comparing the performance of the neural network when randomly initializing weights within the convolutional layers with initializing the weights to that of a neural network trained on ImageNet.

The CNN architecture that was found to consistently perform the best in terms of its best validation loss as recorded over a number of runs was ResNet-50, using 32 neurons within the dense layer and trained using SGD with a learning rate of 0.01 and no momentum. Additionally, it was found that using transfer learning on the convolutional layers of ResNet-50 trained on ImageNet, resulted in the model overfitting towards the end of training while obtaining the best results in the early stages of training. The parameters that were found to result in the best performance are listed in Table I.

TABLE I
BEST CNN ARCHITECTURE AND PARAMETER COMBINATION FOUND

Parameter	**Value**
Convolutional layers architecture	ResNet-50
Number of hidden dense layers	1
Number of neurons per hidden dense layer	32
Number of output neurons	1
Output neuron activation function	Sigmoid
Hidden layers activation function	ReLU
Optimizer	SGD
Learning rate	0.01
Momentum	None
Transfer learning	Convolutional layers only
Regularization	Dropout on fully connected layers
Dropout rate	0.5
Batch Normalization	Applied to all fully connected layers

D. Genetic Programming Parameter Tuning

This section describes the parameter tuning that was performed for the GP algorithm used within the hybrid approach. The parameter tuning performed aimed to minimize the validation loss of the best classifier produced by the

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

algorithm. The following parameter values were selected based on observations made during the training process:

- Population size: It was observed that increasing the population size to a size larger than 500 individuals had no observable positive impact on the training performance achieved. Hence, a population size of 500 was used.
- Number of generations: During most tests performed for different parameter values, it was observed that the best solution found on the training set was found within 500 generations. For this reason, 500 generations were set as the maximum number of generations.

Using the above described population size and number of generations, tests were conducted to empirically determine optimal values for the following parameters:

- Mutation and Crossover probability: The mutation and crossover probabilities determine the probability of performing mutation and crossover to selected individuals to produce offspring. The higher the crossover probability the faster the search will converge, the higher the mutation probability the closer the search becomes to random search. These following mutation and crossover probabilities were tested for the following value pairs (0.1, 0.9), (0.2, 0.8), (0.3, 0.7) and (0.4, 0.6) respectively.
- Initial maximum depth: This parameter determines the maximum depth for trees used in the ramped half and half method to generate the initial population. The minimum depth used was kept fixed at 2, with maximum depths of 3, 5, 10 and 15 being tested.
- Parsimony coefficient: A coefficient that determines the penalty that should be assigned to large trees to prevent bloat. Values were tested in the range of 0.001 - 0.1.
- Tournament size: The tournament size determines the size of the group of individuals selected for comparison during tournament selection. Values of 3, 5 and 7 were tested.

Tests were run where combinations of parameter values from the above described list were chosen non-exhaustively. Parameter value combinations were chosen based on past experience and adjusted based on the effect in validation loss observed. The main focus of the parameter tuning was to prevent overfitting while at the same time allowing trees to be large enough to consider a sufficient subset of the features to make accurate predictions. The parameters that were found to obtain the best results in the experiments conducted are listed in Table II below.

TABLE II
GP PARAMETER VALUES

Parameter	**Value**
Crossover probability	0.8
Mutation probability	0.2
Initial maximum depth	10
Parsimony coefficient	0.001
Tournament size	5
Population size	500

E. Performance Comparison

The comparison of CNNs and the hybrid approach for thumbnail selection will be done using two main comparisons. Firstly, the difference in classification performance of the two methods will be compared and statistical tests will be performed to determine if there is a statistical difference in results. This will be done by training each method 30 times on the training data set and measuring the best validation loss obtained during each run, as well as the training accuracy, training loss and validation accuracy accompanied by the best validation loss for each run. To compare the classification performance of the methods, a two tailed z-test will be performed with $\alpha = 0.05$ to determine if there is a significant difference in results obtained between the two methods.

Secondly, both methods will be applied to select three thumbnails for a set of trailers for movies and series that will act as the test set for this study. These trailers will be introduced in the results section together with the thumbnails selected by both methods for each trailer. The selected thumbnails are then compared and discussed.

VII. RESULTS AND DISCUSSION

This section reports on and compares the performance of ResNet-50 and the hybrid approach for thumbnail selection. The best validation log loss obtained by each method was recorded for each run, together with the training accuracy, training loss and validation accuracy of the classifier at the time that the best validation loss was obtained in the run. The results obtained on the training and validation data sets are presented in Tables 2 and 3 respectively.

TABLE III
TRAINING PERFORMANCE COMPARISON

	Training Accuracy			Training Loss		
Method	**Best**	**Mean**	**Standard Deviation**	**Best**	**Mean**	**Standard Deviation**
CNN	0.93	**0.91****	0.02	0.18	**0.25****	0.05
Hybrid	0.74	0.72	0.02	0.39	0.41	0.02

TABLE IV
VALIDATION PERFORMANCE COMPARISON

	Validation Accuracy			Validation Loss		
Method	**Best**	**Mean**	**Standard Deviation**	**Best**	**Mean**	**Standard Deviation**
CNN	0.83	**0.82****	0.01	0.4	**0.44****	0.02
Hybrid	0.72	0.69	0.02	0.55	0.59	0.02

From these results it can be seen that there was a statistically significant difference in results obtained by the two methods, with the ResNet-50 outperforming the hybrid approach on all metrics recorded. More specifically, the CNN was able to obtain an average validation loss of 0.44 and an average validation accuracy of 82% in comparison to the average validation loss of 0.59 and average validation accuracy of 69%

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

obtained by the hybrid approach . The difference in results could be attributed to two main factors. Firstly, the CNN is able adjust the weights within the convolutional layers to extract features that are likely more suited to the problem at hand, wheres the hybrid approach method was not able to adjust the weights within the convolutional layers and had to work with more general extracted features. However, when using transfer learning with CNNs where the convolutional layers were frozen, CNNs were still found to outperform the hybrid approach. Hence, this is likely not a main factor that led to a large difference in results observed. The second factor considered is the fact that the set of features extracted by convolutional layers are likely still too large for the GP to efficiently operate on. The set of 512 features require large arithmetic trees to effectively consider most features, likely making the program space highly complex to search. Hence, it is likely that neural networks naturally scale to larger sets of features than GP.

It can also be noted that there is a relatively large difference in results obtained by the CNN on the training and validation set, with the CNN obtaining an average training accuracy of 91%, while obtaining an average validation accuracy of 82%. Hence, the CNN was able to heavily overfit the training data set. This could likely be remedied by using a larger data set, since the number of free parameters within the CNN have already been restricted by using few neurons within the dense layers.

The two methods are compared based on their ability to select thumbnails from trailers. Each method is used to select the top 3 frames from each trailer, with frames sampled at a rate of 1 frame per second. Three examples of the top three thumbnails selected by each method are given in figures 1, 2 and 3.

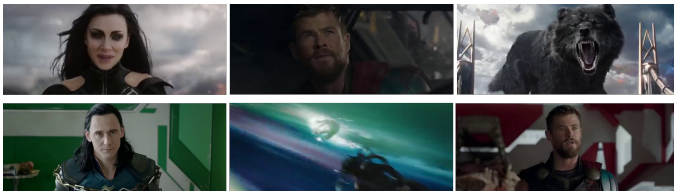

Fig. 1. Selected thumbnails for Thor: by CNN(top) and hybrid (bottom). Thumbnail score decreasing from left to right.

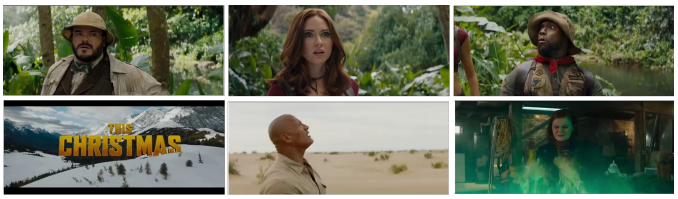

Fig. 2. Selected thumbnails for Jumanji: by CNN(top) and hybrid (bottom). Thumbnail score decreasing from left to right.

From the figures it can be seen that in general, the CNN was able to select appropriate thumbnails more consistently in comparison to the hybrid approach. The thumbnails selected by the CNN are more consistently in focus and well centered on a character present in the trailer. However, the hybrid approach was able to identify two thumbnails that can be considered competitive with those selected by the CNN in Fig1. These results are to be expected as it was shown that the CNN outperformed the hybrid approach on the training and validation data sets.

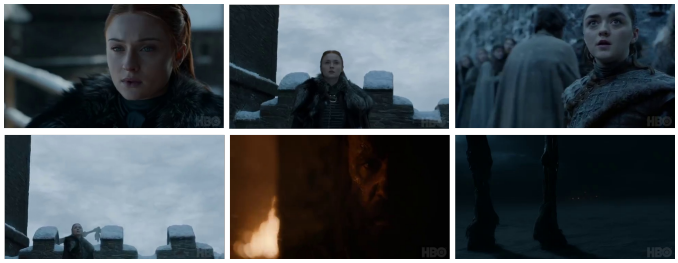

Fig. 3. Selected thumbnails for Game of Thrones: by CNN(top) and hybrid (bottom). Thumbnail score decreasing from left to right.

It can also be noted that the trailer from Fig 3 was likely the most difficult to select thumbnails for since the thumbnails selected by the CNN are not of the same quality as those selected for other trailers. Furthermore, the hybrid approach was not able to select any suitable thumbnails for that specific trailer. This is likely due to the fact that this trailer contains many scenes that are either dark or of a monotone nature when compared to the other trailers.

VIII. Conclusion and Future Research

The main contribution of the research presented is finding a solution to the thumbnail selection problem. The research also compares different convolution neural networks for thumbnail selection and the performance of CNNs with that of a hybrid approach combining a CNN for feature extraction and genetic programming for classification. The ResNet-50 convolutional neural network was found to perform the best for thumbnail selection, outperforming the other CNNs evaluated as well as the hybrid approach.

Future work will focus on improving the performance of both CNN and the hybrid approach in solving this problem. The results obtained by the CNN could likely be improved by increasing the size of the data set, since the data set only contained 2500 examples, which allowed the CNN to easily overfit the training data. Not only will an increased size in the data set prevent the CNN from overfitting, but will also allow larger architectures to be applied to the data set to learn a more complex mapping. The hybrid approach appeared to still struggle with the large number of features present after feature extraction. Hence, this method could possibly be improved by applying further feature reduction before applying GP to the features. Furthermore, other GP representations could be tested such as rule induction trees that have shown to perform well at naturally performing feature selection as part of the evolution process.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

ACKNOWLEDGMENT

The authors would like to thank the Multichoice Group for funding this research as part of the Multichoice Machine Learning Chair.

REFERENCES

[1] M. Christel, "Evaluation and user studies with respect to video summarization and browsing," in *Proc. SPIE 6073, Multimedia Content Analysis, Management, and Retrieval*, 2006, p. https://doi.org/10.1117/12.642841.

[2] N. Aruthurs, S. Birnbaum, and N. Gruver, "Selecting youtube video thumbnails via convolutional neural networks," http://cs231n.stanford.edu/reports/2017/pdfs/710.pdf2017, 2017.

[3] L. W., M. T., Z. Y., C. C., and L. J., "Multi-task deep visual-semantic embedding for video thumbnail selection," in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3707–3715.

[4] L. C., H. Q., and J. S, "Query sensitive dynamic web video thumbnail generation," in *Proceedings of the 2011 18th IEEE International Conference on Image Processing*, vol. 9, 2011, pp. 2449–2452.

[5] G. Y. and a. X. J. Zhang T., "Thematic video thumbnail selection," in *16th IEEE International Conference on Image Processing (ICIP)*, 2009, pp. 4333–4336.

[6] Y. Song, M. Redi, J. Vallmitjana, and A. Jaimes, "To click or not to click: Automatic selection of beautiful thumbnails from videos," in *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, ser. CIKM '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 659–668. [Online]. Available: https://doi.org/10.1145/2983323.2983349

[7] J. R. Koza, *Genetic Programming: On the Programming of Computers by Natural Selection*. MIT Press, 1992.

[8] J. Ren, X. Shen, Z. Lin, and R. Mech, "Best frame selection in a short video," in *Proceedings of the 2020 Winter Conference on Applications of Computer Vision (WACV 2020)*, 2020, pp. 3212–3221.

[9] Y. Yuan, L. Ma, and W. Zhu, "Sentence specified dynamic video thumbnail generation," in *Proceedings of the 27th ACM International Conference on Multimedia*, ser. MM '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 2332–2340. [Online]. Available: https://doi.org/10.1145/3343031.3350985

[10] A. B. Vasudevan, M. Gygli, A. Volokitin, and L. Van Gool, "Query-adaptive video summarization via quality-aware relevance estimation," in *Proceedings of the 25th ACM International Conference on Multimedia*, ser. MM '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 582–590. [Online]. Available: https://doi.org/10.1145/3123266.3123297

[11] K. Simonyan and A. Zisserman. (2015) Very deep convolutional networks for large-scale image recognition. [Online]. Available: Very deep convolutional networks for large-scale image recognition.

[12] C. Szegedy, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016, pp. 770–778.

[14] D. Agnelli, A. Bollini, and L. L., "Image classification: An evolutionary approach," *Pattern Recognition Letters*, vol. 23, no. 1, pp. 303–309, 2002.

[15] R. Nandi, A. Nandi, R. Rangayyan, and S. D., "Classification of breast masses in mammograms using genetic programming and feature selection." *Medical and Biological Engineering and Computing*, vol. 44, no. 8, pp. 683–694, August 2006.

[16] H. Al-Sahaf, A. Song, K. Neshatian, and Z. M., "Two-tier genetic programming: Towards raw pixel-based image classification," *Expert Systems with Applications*, vol. 39, no. 16, pp. 12 291–12 301, 2012.

[17] R. Poli, "Genetic programming for image analysis." in *Proceedings of the 1st Annual Conference on Genetic Programming*, 1996, pp. 363–368.

[18] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[19] J. Kiefer and J. Wolfowitz, "Stochastic estimation of the maximum of a regression function." *Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952.

[20] K. Diederik and B. Jimmy, "Adam: A method for stochastic optimization," in *Proceedings of International Conference on Learning Representations*, vol. 12, 2014.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE