System Fusion with Deep Ensembles

Liviu-Daniel Ştefan
liviu_daniel.stefan@upb.ro
University Politehnica of Bucharest
Bucharest, Romania

Mihai Gabriel Constantin
mgconstantin@imag.pub.ro
University Politehnica of Bucharest
Bucharest, Romania

Bogdan Ionescu
bogdan.ionescu@upb.ro
University Politehnica of Bucharest
Bucharest, Romania

ABSTRACT

Deep neural networks (DNNs) are universal estimators that have achieved state-of-the-art performance in a broad spectrum of classification tasks, opening new perspectives for many applications. One of them is addressing ensemble learning. In this paper, we introduce a set of deep learning techniques for ensemble learning with dense, attention, and convolutional neural network layers. Our approach automatically discovers patterns and correlations between the decisions of individual classifiers, therefore, alleviating the difficulty of building such architectures. To assess its robustness, we evaluate our approach on two complex data sets that target different perspectives of predicting the user perception of multimedia data, i.e., interestingness and violence. The proposed approach outperforms the existing state-of-the-art algorithms by a large margin.

CCS CONCEPTS

• **Computing methodologies → Ensemble methods**; **Neural networks**; **Visual content-based indexing and retrieval**.

KEYWORDS

ensemble learning; late fusion; deep neural networks; deep ensembles; interestingness; violence; subjective perception.

ACM Reference Format:
Liviu-Daniel Ştefan, Mihai Gabriel Constantin, and Bogdan Ionescu. 2020. System Fusion with Deep Ensembles. In *Proceedings of the 2020 International Conference on Multimedia Retrieval (ICMR '20), June 8–11, 2020, Dublin, Ireland.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3372278.3390720

1 INTRODUCTION

Ensemble learning has a successful history in the research community in general and in machine learning in particular [18, 33, 37]. An ensemble is composed of a set of individual component classifiers characterized by *high diversity*, with an inherent domain of competence, and a *learning or combination algorithm*, that creates a new output based on the individual outputs of classifiers. It provides a late fusion scheme based on a collection of systems. The goal of *ensembling* is to obtain a strong learner based on the experience of the myriad of classifiers it incorporates. This approach has been empirically demonstrated to outperform most state-of-the-art learners in many benchmark campaigns, both for general tasks, e.g., classification problems [3, 30], and for domain specific applications, e.g., memorability, violence and interestingness prediction [2, 6, 7, 26].

Recently, deep neural networks have become vital tools for enabling effective learning in a broad array of applications [14, 22, 23, 36]. One of the consolidated findings of state-of-the-art deep learning architectures [13, 20, 28, 32] is that they are able to discover intricate structures in vast data sets, due to their superior representation capabilities for high-dimensional data, and jointly, with their classifying capabilities, dramatically outperforming conventional descriptors and classifiers. Addressing *ensembling* with deep learning approaches is an open research problem that has not seen too much progress in the research community.

Ensemble systems have more recently attained state-of-the-art results in several tasks linked with the prediction of subjective human perception of visual data, such as the prediction of visual interest [9, 34], the prediction of media memorability [2, 5], classification of violent videos [7, 29] or emotional content analysis [8, 31]. For a more comprehensive view of ensembling methods and their applications, the reader is referred to the following survey papers [12, 19, 27]. Given a probability on a hypothesis, the ensemble mechanism could be a majority voting, weighting or statistical scheme, or more intricate approaches such as the use of hierarchical or network architectures.

In this article, we introduce a *deep ensembling architecture*—a deep learning-based approach, designed to discover patterns and correlations between the decisions of individual classifiers. With the help of dense, attention, and convolutional layers, we aim to model the bias learned by each classifier and the correlations between biases to improve the overall performance of the ensembling system. The contribution beyond state of the art can be summarized with the following: (i) We develop and implement an ensembling method that uses deep neural network models with dense, convolutional and attention layers to achieve system fusion; (ii) We propose a novel input decoration scheme that transforms the input into a matrix representation that takes into account the correlations between inducers, specifically designed for convolutional layers; (iii) We solve with our approach two difficult scenarios that target the prediction of user perception of multimedia data, i.e., interestingness and violence prediction. To the best of our knowledge, these approaches are not explored in the literature, current systems focusing on statistical or traditional fusion architectures.

To show the benefits of the ensemble approach and evaluate our methodology, we provide comprehensive experiments on a series of intricate tasks that target subjective user perception of visual data, i.e., interestingness and violence prediction. We compare our proposed ensemble system with classical approaches and state-of-the-art methods that incorporate these approaches. Experiments show that our approach provides a significant boost in

ICMR '20, June 8–11, 2020, Dublin, Ireland

ACM ISBN 978-1-4503-7087-5/20/06…$15.00
https://doi.org/10.1145/3372278.3390720

performance, outperforming the state-of-the-art approaches by a large margin. Distantly related approaches of ensembling via deep learning center around developing an ensemble of deep neural network features [17, 35]. Despite sharing some similarities with our method, these approaches do not use the scores yielded by inducers, rather they perform an early fusion at features level.

The remainder of the paper proceeds as follows. We first present our proposed deep ensembling architectures in Section 2. Then, we present the experimental setup and the results and analysis in Section 3 and Section 4, respectively. Lastly, Section 5 presents our conclusions and discusses future work.

2 PROPOSED METHOD

The standard ensembling problem can be formulated as following. Given a set of n instances and m representations $D = \{(x_i, y_i)\}$, $|D| = n, x_i \in \mathbb{R}^m, y_i \in \mathbb{R}$, where x_i and y_i represent the input vector for the sample i, and the output vector of classifier i, respectively. An ensemble setting uses an aggregation function that aggregates k classifiers $\{f_0, ..., f_{k-1}\}$ toward providing a single prediction: $\widehat{y_i} = \varphi(x_i) = G(f_0, ..., f_{k-1})$, where $\widehat{y_i} \in \mathbb{Z}$ and $G(.)$ represents the aggregation function.

In this context, we propose several deep learning-based ensembling architectures for data retrieval. That is, given multiple video or image inputs, our goal is to perform retrieval while combining the information from multiple learners in a beneficial way. Our assumption is that, by performing the aggregation via a deep neural network architecture, we can model the bias learned by each system and the correlations between the biases more efficiently, thus allowing us to perform retrieval robustly. To this end, we employ simple, yet efficient deep neural network architectures, based on dense (Section 2.1), attention (Section 2.2), and convolutional layers (Section 2.3), respectively.

2.1 Dense Architecture

Considering dense architectures are universal approximators, capable of learning any function, we build the first ensemble network by stacking such dense layers. The diagram of the implemented dense architecture is presented in Figure 1a. Firstly, we start by defining a set of rules to build network architectures, namely: (i) varying the numbers of dense layers, i.e., {5, 10, 15, 20, 25}; (ii) varying the numbers of neurons for each dense layer, i.e., {25, 50, 500, 1000, 2000, 50-00}; (iii) including or excluding batch normalization layers. In this context, we start with a minimum of 5 dense layers and 25 neurons and we create end-to-end architectures in a progressive order, simplest models first. Each combination of parameters is trained and evaluated until the most effective architecture is found.

2.2 Dense Architecture with Attention

To further improve the precision of the ensemble architecture, we further include soft attention maps with values between 0 and 1, to our baseline ensemble policies, by learning the attention parameters in an end-to-end manner, helping the networks to focus on key elements of the input. A diagram of the proposed implementation is presented in Figure 1b. Let $x_i \in \mathbb{R}^k$ be the input vector, $z \in \mathbb{R}^k$ a feature map vector, $a \in [0, 1]^k$ a soft attention vector, $g \in \mathbb{R}^k$ an attention estimator, and $f_\phi(x_i)$ and attention network with parameters ϕ. The attention is implemented as $a = f_\phi(x_i), g = a \odot z$ where $\odot$ is element-wise multiplication, while z is an output of another neural network $f_\phi(x_i)$ with parameters ϕ.

2.3 Dense Architecture with Convolutions

The final proposed architecture includes the addition of convolutional layers and a technique for input decoration, allowing us to disclose local relationships between locally adjacent scores yielded by the inducers incorporated in the ensemble setting. Convolutional layers are able to accurately explore local relationships in different mathematical spaces. However, given the intrinsic randomness of the order of the input vector x_i and, therefore, the lack of any obvious localized correlation between adjacent elements of the vector, we chose to deploy an input decoration scheme that would allow the use of convolutional layers.

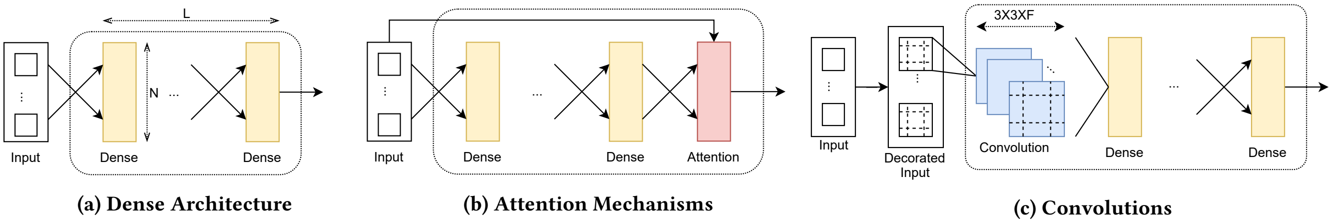
A diagram of the proposed convolutional architecture is presented in Figure 1c. Given the $x_i = [s_{0,i}...s_{k-1,i}]$ input vector for a sample i, representing the output scores of each classifier, we choose to decorate each element of this vector with output scores and correlation scores from the most similar systems with respect to output. Therefore, given the matrix $Y = [y_0...y_{k-1}]$ where each of the y_i vectors represents the scores given by the classifier f_i for all the samples, we calculate the similarity between each classifier via the Pearson's correlation coefficient [25]. The final decorated version of the input is represented in equation 1, where, for each sample i, the pairs $(c_{0,j}, r_{0,j})$ represent the output score (c) and Pearson's correlation score (r) for the most similar system with any given system score s_j, the pairs $(c_{1,j}, s_{1,j})$ represent the second most similar system, and so on.

$$xd_i = \begin{bmatrix} r_{3,0} & c_{0,0} & r_{0,0} & & r_{3,k-1} & c_{0,k-1} & r_{0,k-1} \\ c_{3,0} & s_0 & c_{1,0} & \dots & c_{3,k-1} & s_{k-1} & c_{1,k-1} \\ r_{2,0} & c_{2,0} & r_{1,0} & & r_{2,k-1} & c_{2,k-1} & r_{1,k-1} \end{bmatrix} \quad (1)$$

We use the newly created xd_i vector as input for our dense architecture with one convolutional layer. The convolutional layer has 3×3 size filters, therefore having 10 trainable parameters for each filter in the layer, and a stride of 3, followed by an average pooling layer. We tested three filter configurations for the convolutional layer: 1, 5, or 10 filters, thus allowing the network to perform a more extensive array of similarity analysis.

2.4 Ensembling

To perform the actual ensembling, we employ the following steps. Firstly, for the individual x_i vectors, we create the decorated xd_i vectors, and normalize the two collections of vectors. Then, using the x_i vectors as input, we start the processing of dense architectures with 5 layers, 25 neurons per layer, followed by a search for the best performing dense architecture by varying the network model size, according to the values presented in Section 2.1. The best performing dense model is augmented with an attention layer, while keeping the x_i vectors as input. The final step involves the deployment of convolutional layers. Using the best performing dense architecture and the xd_i vectors as input, we search for the best performing convolutional architecture by varying the number of filters applied to the layers, according to the values presented in Section 2.3. The fusion results correspond to the output of the networks. During the dense network search (see Section 2.1), we evaluate 60



(a) Dense Architecture

(b) Attention Mechanisms

(c) Convolutions

Figure 1: Illustration of the proposed deep ensembling methods: variable number of dense layers (L), variable number of neurons per dense layer (N), and variable number of filters for the convolutional layer (F).

dense networks for each data set. Each network is trained for 200 epochs, with a batch size of 64, using an initial learning rate of 0.01 optimized via the Adam optimizer [15], with $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 1e - 08$. The loss function for all tested models is binary cross-entropy.

3 EXPERIMENTAL SETUP

3.1 Data Sets

To validate the solutions, we conducted extensive experiments on two intricate benchmark data sets, namely the *Predicting Media Interestingness*, and *Affective Impact of Movies*. These data were validated during the yearly MediaEval benchmarking initiative for multimedia evaluation campaign. All the data comes with high-quality annotations provided by experts.

The MediaEval 2017 Predicting Media Interestingness data set [9] (*INT2017*) provides data for two scenarios: (i) prediction of image visual interestingness (*INT2017.Image*), and (ii) prediction of video visual interestingness (*INT2017.Video*), each of them containing 9,831 individual key-frames and video segments respectively.

The MediaEval 2015 Affective Impact of Movies (*VSD2015.Video*) data set [29] is composed of 31 full movies, 86 YouTube videos and 10,900 short clips extracted from 199 movies of various genres (up to 96 hours), with the benchamrking scenario of automatically classifying video content as violent or non-violent, using the definition of violence provided by the authors [29].

We have considered all the systems participating in the benchmarking campaigns, namely 33 systems for the INT2017.Image, 42 systems for the INT2017.Video, and 48 systems for the VSD2015.Video. Inducers ranged from SVMs, Naive Bayes, LDA, ensembles, to DNNs and CNN representations. For a detailed description, the reader may access the participants working notes here[1].

3.2 Evaluation

Ensembling requires typically tens of systems to be able to boost the performance. In practice, is basically impossible to implement or retrieve such a large number of systems from the authors, considering also re-running them in the very same conditions. There are also no best practices in this respect in the literature. The only approaches that do so use a very reduced number of inducers, e.g., less than 10 [21]. We therefore adopted a compromise that uses all the system runs submitted to the respective benchmarking competitions, and experimenting solely on the test data, as we do not have access to the development systems. These test data system runs were provided by the task organizers.

The evaluation is carried out using the following test data split scenarios: (i) 75% training and 25% testing (RSKF75), and (ii) 50% training and 50% testing (RSKF50). Split samples are randomized, and this action is performed multiple times to obtain a thorough coverage. In the end, 100 partitions are generated. The metrics are computed as average values over these partitions. We stress that this approach is even more disadvantageous than training the systems on the entire development data, because, the number of items is significantly lower. An empiric test performed on the systems our team submitted to the Interestingness task [4] shows a dramatic drop in performance, i.e., 46.85% for the INT2017.Image task and 57.1% on the INT2017.Video task, supporting the assumption that training in this setup is disadvantageous.

For assessing performance, we use the official metrics released by the authors of the data (on which the inducers were optimized), namely: (i) for the INT2017 data set, we use the Mean Average Precision over the 10 highest ranked items (mAP@10), and (ii) for the VSD2015.Video data set, the Mean Average Precision (mAP).

4 RESULTS AND DISCUSSION

This section presents the results of the best-performing architectures: we present the baseline and state-of-the-art systems that will be used as comparison (Section 4.1), and we present our approaches (Section 4.2). Overall results are summarized in Table 1.

4.1 Baseline Systems

Best performers at MediaEval. For reference, we present the best-performing systems for each of the three data/scenarios, as released during the MediaEval benchmarking. They will represent, both one of the inducers for our algorithms, and a baseline for comparing the results we achieved. The three systems are the following: Permadi et al. [26] for the INT2017.Image data, with a mAP@10 of 0.1385, Ben-Ahmed et al. [1] for the INT2017.Video data, with a mAP@10 of 0.0827 and Dai et al. [7] for the VSD2015.Video data, with a mAP of 0.296.

Best systems from the literature. Another class of systems that will be used as comparison baselines is represented by approaches published outside the MediaEval benchmark competition, but that use the same data sets and provide state-of-the-art results. The three methods used are the following: Parekh et al. [24] for the INT2017.Image data, with a mAP@10 of 0.156, Wang et al. [34] for

[1] http://ceur-ws.org/Vol-1984/, http://ceur-ws.org/Vol-1436/.

Table 1: Results on the three scenarios: INT2017.Image, INT2017.Video and VSD2015.Video. We present the best results for the baseline systems (b), baseline ensembling systems (e), and the three proposed architectures (proposed), for each test data split scenario (the standard dev/test used by individual systems, RSKF75 or RSKF50 used in this evaluation).

INT2017.Image

System	Split	mAP@10
Permadi et al. [26] (b)	dev/test	0.1385
Parekh et al. [24] (b)	dev/test	0.156
BAda [10] (e)	RSKF50	0.1523
	RSKF75	0.1674
Dense (proposed)	RSKF50	0.2316
	RSKF75	0.3355
Attention (proposed)	RSKF50	**0.2399**
	RSKF75	0.3389
Convolutional (proposed)	RSKF50	0.2293
	RSKF75	**0.3436**

INT2017.Video

System	Split	mAP@10
Ben-Ahmed et al. [1] (b)	dev/test	0.0827
Wang et al. [34] (b)	dev/test	0.093
BAda [10] (e)	RSKF50	0.0961
	RSKF75	0.1129
Dense (proposed)	RSKF50	0.1563
	RSKF75	0.2677
Attention (proposed)	RSKF50	0.1668
	RSKF75	0.2750
Convolutional (proposed)	RSKF50	**0.1692**
	RSKF75	**0.2799**

VSD2015.Video

System	Split	mAP
Dai et al. [7] (b)	dev/test	0.296
Li et al. [21] (b)	dev/test	0.303
BGrad [11] (e)	RSKF50	0.3521
	RSKF75	0.392
Dense (proposed)	RSKF50	0.6192
	RSKF75	0.6341
Attention (proposed)	RSKF50	0.6228
	RSKF75	**0.6486**
Convolutional (proposed)	RSKF50	**0.6281**
	RSKF75	0.6471

* Please note that a direct comparison between the individual systems and the ensembling should be carried out cautiously, as the data are different. However, as presented in the article, this gives a clear indicator of the boosting performance.

the INT2017.Video data, with a mAP@10 of 0.093 and Li et al. [21] fot the VSD2015.Video data, with a mAP of 0.303.

Ensembling systems. The final class of baseline approaches consists of classical ensembling methods, that use fusion strategies such as: fusion of the scores by taking the minimum (*LFMin*), maximum (*LFMax*), combination of min-max (*LFMinMax*), average (*LFAvg*), median (*LFMed*), and weighted (*LFWeight*) of the scores of all the individual systems [16] and two boosting approaches: AdaBoost [10] (*BAda*) and Gradient Boosting [11] (*BGrad*). In creating these ensembles, we used as inducers the same systems presented in Section 3.1 and the same evaluation methods presented in Section 3.2 therefore, creating an accurate base for comparing our methods.

4.2 Proposed Systems

The results are summarized in Table 1. The proposed approaches surpass both the two best performers from the literature and the baseline fusion methods. As expected, results for the RSKF75 split are better than the ones achieved on the RSKF50 split, due to the more training data. The results show that the proposed approaches clearly outperform all three baseline runs, under all three proposed architectures. While the dense architectures outperformed the baselines, the best performance is achieved with the attention and convolutional architectures.

For the INT2017.Image data, the best performing dense architecture uses 10 dense layers and 1,000 neurons per layer, without batch normalization, achieving a mAP@10 of 0.3355 for the RSKF75 split and of 0.2316 for RSKF50. The addition of attention layer further increased the results, the best performing architecture with the RSKF50 split achieving a mAP@10 of 0.2399. The same is true for the convolutional layers, where an architecture with 5 filters achieved the best results for the RSKF75 split, namely 0.3436. For the INT2017.Video data, the best performance with a dense architecture is achieved using 25 layers and 2,000 neurons per layer, with batch normalization, yielding a mAP@10 of 0.2677 and 0.1562, for RSKF75 and RSKF50, respectively. While the introduction of an attention layer does improve these scores, the best results overall are obtained with the convolutional layers. A different setup performs best for each of the two splits, i.e., a layer with 5 filters for the RSKF75 split, achieving a mAP@10 of 0.2799, and a layer with 10 filters for the RSKF50 split, with a mAP@10 of 0.1692.

Finally, for the VSD2015.Video data, an architecture with 5 dense layers and 500 neurons per layer achieved the highest performance for the dense architecture, with a mAP of 0.6341 for the RSKF75 split and of 0.6192 for the RSKF50. The best results for the RSKF75 split are obtained with an attention architecture, mAP of 0.6486, while the best results for the RSKF50 split are obtained with a convolutional architecture with 10 filters, mAP of 0.6281. While the increase in performance brought by the addition of attention and convolutional layers over the dense architecture is not very big, it is worth noting that except for the INT2017.Image data and using the convolutional architecture on the RSKF50 setup, the results are constantly better when these layers are added.

5 CONCLUSIONS

In this paper, we proposed a deep ensembling approach that employs architectures based on dense, attention, and convolutional layers. The main advantage of the proposed architecture is in its ability to automatically discover correlations between the inducer systems. Tested in two difficult scenarios, i.e., for interestingness and violence prediction, results proved a great improvement compared to the inducer systems and state-of-the-art approaches, with many instances reaching at least double the mAP performance, e.g., from 0.156 to 0.3436, and from 0.09 to 0.2799 on the image and video prediction of interestingness, and from 0.303 to to 0.6486 on the violence prediction.

As future work, we propose to further experiment with the impact of the number and diversity of inducers on the ensembling results and with combining attention and convolution approaches.

ACKNOWLEDGMENTS

The work has been funded by the Ministry of Innovation and Research, UEFISCDI, project SPIA-VA, agreement 2SOL/2017, and by the Operational Programme Human Capital of the Ministry of Europe Funds through the Financial Agreement 51675/09.07.2019, SMIS code 125125.

REFERENCES

[1] Olfa Ben Ahmed, Jonas Wacker, Alessandro Gaballo, and Benoit Huet. 2017. EURECOM@MediaEval 2017: Media Genre Inference for Predicting Media Interestingness. In *MediaEval Workshop, Dublin, Ireland, September 13-15.*

[2] David Azcona, Enric Moreu, Feiyan Hu, Tomás Ward, and Alan Smeaton. 2019. Predicting Media Memorability Using Ensemble Models. *In Proc. of MediaEval 2019 Workshop, Sophia Antipolis, France, Oct. 27-29, 2019* (2019).

[3] Robert E Banfield, Lawrence O Hall, Kevin W Bowyer, and W Philip Kegelmeyer. 2006. A comparison of decision tree ensemble creation techniques. *IEEE transactions on pattern analysis and machine intelligence* 29, 1 (2006), 173–180.

[4] Mihai Gabriel Constantin, Bogdan Andrei Boteanu, and Bogdan Ionescu. 2017. LAPI at MediaEval 2017-Predicting Media Interestingness. *In Proc. of MediaEval 2017 Workshop.*

[5] Mihai Gabriel Constantin, Bogdan Ionescu, Claire-Hélène Demarty, Ngoc Q. K. Duong, Xavier Alameda-Pineda, and Mats Sjöberg. 2019. Predicting Media Memorability Task at MediaEval 2019. *In Proc. of MediaEval 2019 Workshop, Sophia Antipolis, France, Oct. 27-29, 2019* (2019).

[6] Qi Dai, Zuxuan Wu, Yu-Gang Jiang, Xiangyang Xue, and Jinhui Tang. 2014. Fudan-NJUST at MediaEval 2014: Violent Scenes Detection Using Deep Neural Networks.. In *MediaEval.*

[7] Qi Dai, Rui-Wei Zhao, Zuxuan Wu, Xi Wang, Zichen Gu, Wenhai Wu, and Yu-Gang Jiang. 2015. Fudan-Huawei at MediaEval 2015: Detecting Violent Scenes and Affective Impact in Movies with Deep Learning. In *Working Notes Proceedings of the MediaEval 2015 Workshop.*

[8] Emmanuel Dellandréa, Martijn Huigsloot, Liming Chen, Yoann Baveye, Zhongzhe Xiao, and Mats Sjöberg. 2018. The MediaEval 2018 Emotional Impact of Movies Task. *In Proc. of MediaEval 2018 Workshop* (2018).

[9] Claire-Hélène Demarty, Mats Sjöberg, Bogdan Ionescu, Thanh-Toan Do, Michael Gygli, and Ngoc Duong. 2017. Mediaeval 2017 predicting media interestingness task.

[10] Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences* 55, 1 (1997), 119–139.

[11] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[12] Heitor Murilo Gomes, Jean Paul Barddal, Fabrício Enembreck, and Albert Bifet. 2017. A survey on ensemble learning for data stream classification. *ACM Computing Surveys (CSUR)* 50, 2 (2017), 23.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 770–778.

[14] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2017. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196* (2017).

[15] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[16] Josef Kittler, Mohamad Hatef, Robert PW Duin, and Jiri Matas. 1998. On combining classifiers. *IEEE transactions on pattern analysis and machine intelligence* 20, 3 (1998), 226–239.

[17] Peter Kontschieder, Madalina Fiterau, Antonio Criminisi, and Samuel Rota Bulo. 2015. Deep neural decision forests. In *Proceedings of the IEEE international conference on computer vision.* 1467–1475.

[18] Kamran Kowsari, Mojtaba Heidarysafa, Donald E Brown, Kiana Jafari Meimandi, and Laura E Barnes. 2018. Rmdl: Random multimodel deep learning for classification. In *Proceedings of the 2nd International Conference on Information System and Data Mining.* ACM, 19–28.

[19] Bartosz Krawczyk, Leandro L Minku, João Gama, Jerzy Stefanowski, and Michał Woźniak. 2017. Ensemble learning for data stream analysis: A survey. *Information Fusion* 37 (2017), 132–156.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems.* 1097–1105.

[21] Xirong Li, Yujia Huo, Qin Jin, and Jieping Xu. 2016. Detecting Violence in Video using Subclasses. In *Proceedings of the 2016 ACM Conference on Multimedia Conference, MM 2016.* ACM, 586–590.

[22] Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. 2017. Plug & play generative networks: Conditional iterative generation of images in latent space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.* 4467–4477.

[23] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499* (2016).

[24] Jayneel Parekh, Harshvardhan Tibrewal, and Sanjeel Parekh. 2018. Deep Pairwise Classification and Ranking for Predicting Media Interestingness. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval, ICMR, Yokohama, Japan, June 11-14.* ACM, 428–433.

[25] Karl Pearson. 1896. VII. Mathematical contributions to the theory of evolution.—III. Regression, heredity, and panmixia. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* 187 (1896), 253–318.

[26] Reza Aditya Permadi, Septian Gilang Permana Putra, Helmiriawan, and Cynthia C. S. Liem. 2017. DUT-MMSR at MediaEval 2017: Predicting Media Interestingness Task. In *MediaEval Workshop, Dublin, Ireland, September 13-15.*

[27] Omer Sagi and Lior Rokach. 2018. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 8, 4 (2018), e1249.

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

[29] Mats Sjöberg, Yoann Baveye, Hanli Wang, Vu Lam Quang, Bogdan Ionescu, Emmanuel Dellandréa, Markus Schedl, Claire-Hélène Demarty, and Liming Chen. 2015. The MediaEval 2015 Affective Impact of Movies Task.. In *MediaEval.*

[30] So Young Sohn and HW Shin. 2007. Experimental study for the comparison of classifier combination methods. *Pattern Recognition* 40, 1 (2007), 33–40.

[31] Jennifer J Sun, Ting Liu, and Gautam Prasad. 2018. Gla in mediaeval 2018 emotional impact of movies task. *In Proc. of MediaEval 2018 Workshop* (2018).

[32] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 1–9.

[33] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. 2017. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204* (2017).

[34] Shuai Wang, Shizhe Chen, Jinming Zhao, and Qin Jin. 2018. Video Interestingness Prediction Based on Ranking Model. In *Proceedings of the Joint Workshop of the 4th Workshop on Affective Social Multimedia Computing and First Multi-Modal Affective Computing of Large-Scale Multimedia Data (ASMMC-MMAC'18).* ACM, 55–61.

[35] Yun Yi, Hanli Wang, and Qinyu Li. 2019. Affective Video Content Analysis with Adaptive Fusion Recurrent Network. *IEEE Transactions on Multimedia* (2019).

[36] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. 2017. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision.* 5907–5915.

[37] Shangtong Zhang and Hengshuai Yao. 2019. Ace: An actor ensemble algorithm for continuous control with tree search. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 5789–5796.