Exploring Deep Fusion Ensembling for Automatic Visual Interestingness Prediction

Mihai Gabriel Constantin and Liviu-Daniel Ştefan and Bogdan Ionescu

Abstract In the context of the ever growing quantity of multimedia content from social, news and educational platforms, generating meaningful recommendations and ratings now requires a more advanced understanding of their impact on the user, such as their subjective perception. One of the important subjective concepts explored by researchers is visual interestingness. While several definitions of this concept are given in the current literature, in a broader sense, this property attempts to measure the ability of audio-visual data to capture and keep the viewer's attention for longer periods of time. While many computer vision and machine learning methods have been tested for predicting media interestingness, overall, due to the heavily subjective nature of interestingness, the precision of the results is relatively low. In this chapter, we investigate several methods that address this problem from a different angle. We first review the literature on interestingness prediction and present an overview of the traditional fusion mechanisms, such as statistical fusion, weighted approaches, boosting, random forests or randomized trees. Further, we explore the possibility of employing a stronger, novel deep learning-based, system fusion for enhancing the performance. We investigate several types of deep networks for creating the fusion systems, including dense, attention, convolutional and cross-space-fusion networks, while also proposing some input decoration methods that help these networks achieve optimal performance. We present the results, as well as an analysis of the correlation between network structure and overall system performance. Experimental validation is carried out on a publicly available data set and on the systems benchmarked during the 2017 MediaEval Predicting Media Interestingness task.

Mihai Gabriel Constantin
University Politehnica of Bucharest, Romania, e-mail: mihai.constantin84@upb.ro

Liviu-Daniel Ştefan
University Politehnica of Bucharest, Romania, e-mail: lstefan@imag.pub.ro

Bogdan Ionescu
University Politehnica of Bucharest, Romania, e-mail: bogdan.ionescu@upb.ro

1 Introduction

Given the prevalence of multimedia data associated with the current online environment and the immense quantity of data uploaded by both amateur and professional content creators, the need for in-depth understanding of the uploaded data has emerged. Automatic classification and recommendation systems are needed in order to help users navigate online platforms that are able to correctly understand both user preferences and the quality of the multimedia content hosted on the platforms. The research and development communities are currently giving increasing attention to the study of subjective content properties, therefore seeking to understand how visual content affects viewers and tune their algorithms accordingly. This represents a shift in research focus from previous directions, such as understanding the content of images and videos via objective properties such as object detection [1] and scene classification [2].

Visual interestingness represents one of the most popular concepts currently being studied, being defined as the capacity of "holding or catching attention" in the Oxford Dictionary of English [3]. Berlyne's initial studies in psychology [4] show that interest heavily influences human behaviour and motivation, while more recent works that study the interestingness of images [5] show that interest and the willingness to view and study a media sample are positively correlated. Many researchers also point out the importance of other factors in creating and maintaining interest [6, 9], like novelty, coping potential, arousal and aesthetic quality. From an emotional perspective, Silvia [6, 7] includes interest among the class of emotions that relate to comprehension, exploration and learning. In this context, it is easy to understand why researchers and developers are starting to focus their efforts on the prediction of multimedia interestingness. An interestingness value assigned to each media item can represent the difference between a video being recommended to users if it fits their viewing profile and being forgotten, and the accurate assessment of this subjective concept can generate more user engagement and satisfaction. On the other hand, it would represent an useful tool for content creators, be they online creators, professors selecting their media samples for classes or advertising agencies, as it could select the most appropriate media samples for distribution out of a large collection of images and videos. Finally, it is important to note that in the current literature the notion of "interestingness" is used to describe two different concepts: *social interestingness* which is usually related to social media concepts like popularity and virality, and *visual interestingness* which is defined as the capacity of media samples to attract and maintain viewer attention. Previous work in this domain have shown these concepts to be both positively [10] and negatively [11] correlated, therefore the link between the concepts is still an opened research direction. However, throughout the rest of this chapter, we will use "interestingness" as a synonym for visual interestingness.

In this chapter we explore the possibility of employing a set of ensembling methods for interestingness prediction, by implementing deep neural networks as the primary ensembling function. To the best of our knowledge, this type of approach presents a high degree of novelty, as deep neural networks are used as inducers in the current state-of-the-art literature, not as the primary ensemble function. Our approach consists of several architectures that include dense, attention, convolutional and the novel cross-space-fusion layers, as well as two input decoration methods that help analyze correlations between similar inducers. Our methods are tested on the publicly available Interestingness10k dataset [19], validated during the 2017 MediaEval[1] Predicting Media Interestingness task [13]. With regards to media interestingness, [8] represents an in-depth literature review of interestingness and covariate concepts, analyzing these concepts and their correlations from psychological, user-centric and computer vision perspectives, while [19] represents a review of the MediaEval Predicting Media Interestingness task, analyzing the best practices, methods, user annotation statistics and the data itself. From an ensembling perspective, three papers introduce some of the deep neural network architectures that we will deploy in this work: [30, 19, 31]. The code corresponding to the proposed methods we will present is available online[2], developed in Python 3 using the Keras 2.2.4 and Tensorflow 1.12 libraries.

The rest of this chapter is organized as follows. Section 2 analyzes the current state-of-the-art, with regards to both interestingness prediction and late fusion systems. In Section 3 we present the methods we propose for media interestingness prediction. Section 4 presents the results and their analysis, pointing out trends and general suggestions with regards to system performance. Finally, Section 5 concludes the paper and discusses future developments.

2 Previous Work

This section discusses and analyzes the current state-of-the-art with regards to two main topics: the advances in the prediction and classification of media interestingness and the most important late fusion methods currently used in the literature, while also presenting some arguments that advocate the deployment of late fusion schemes for interestingness prediction.

2.1 Media Interestingness

From a computer vision perspective, media interestingness prediction, usually referring to prediction in image or video samples, is gaining considerable traction in the community, with a significant increase in the number of papers published on this subject in recent years [19]. However, this is still considered an opened research direction, as methods that improve results are constantly being published. One of the main difficulties in predicting interestingness comes from the subjectivity of interest

[1] https://multimediaeval.github.io/

[2] https://github.com/cmihaigabriel/DeepFusionSystem_v2

among human annotators. Consequently, lower annotator agreement and a lesser degree of separation between interesting and non-interesting samples may be expected when designing a media interestingness dataset or computer vision methods that tackle this issue. Several methods of measuring interest in humans have been used. For example, for the Interestingness10k [19] dataset, annotators are shown pairs of images or videos and are asked to select which of the two samples are more interesting for them, and asked to also consider that "the selected video excerpts/key-frames should be suitable in terms of helping a user to make his/her decision about whether he/she is interested in watching a movie" [52].

Early works in interestingness prediction employ several types of traditional visual features. Gygli et al. [20] use novelty, aesthetics and general preference as cues for image interestingness. Novelty is encoded with the help of a Local Outlier Factor approach, aesthetics via a set of descriptors that encode colorfulness, arousal, complexity, contrast and edge distribution, and general preference is computed by analyzing raw RGB (Red-Green-Blue color space) values, SIFT [47] and GIST [48] features and color histograms. For the prediction of video interestingness Jiang et al. [21] use visual, audio and high-level attributes in a Ranking-SVM (Support-Vector Machine) approach. The authors show that the multi-modal fusion of audio and visual features, consisting of color histograms, SIFT, GIST, MFCC [49], Self-Similarities [50], and Spectrogram SIFT [51], obtains the best result, with a prediction accuracy of 71.4%. Similar methods, that calculate different concepts with the help of traditional descriptors are also used by Grabner et al. [22]. The performance of Sentiment features [23] and C3D models [24] are compared by Gygli and Soleimani [10], and, interestingly sentiment features achieve better results, with a Spearman's correlation rank of $\rho = 0.53$. Another interesting conclusion comes from Fan et al. [25], showing that the fusion of several sources of data improves system performance.

While these studies present interesting approaches, it is difficult to compare them and propose a set of ideas that would increase the chances for a good performance, given their use of different datasets, splits and development conditions. In this context, the MediaEval 2016 and 2017 Predicting Media Interestingness competitions [12, 13] address this problem, by creating a common evaluation framework, consisting of a dataset of images and videos with human-annotated interestingness values, common splits and evaluation metrics for the participating teams and open availability for the data. A large number of systems were submitted to the two editions of the benchmarking competition, 60 systems for the image tasks and 69 for the video tasks, but also outside of the competition, in state-of-the-art papers, 17 image processing systems and 46 video processing systems [19]. While there are many diverse approaches, one noteworthy aspect is that the top results for both tasks can be considered rather low, especially when compared with other more traditional and objective tasks such as object detection or scene classification. For example, the best results achieved during the benchmarking competitions with regards to the official metric, Mean Average Precision (MAP), are $MAP = 0.3075$ in the image prediction task, by Permadi et al. [26], and $MAP = 0.2094$ in the video prediction task, by Ben-Ahmed et al. [27]. These results are further improved outside of the competition, Parekh et al. [28] obtaining a result of $MAP = 0.3125$ for the image task and Wang et al. [29] obtaining a $MAP = 0.2228$. However, a study on the annotation process published by Constantin et al. [19] shows that human annotators also do not achieve near-perfect scores, considering that the best performing annotators never scored above $MAP = 0.7$. This further enforces the idea that the subjectivity of such a task represents one of its main challenges. While the approaches are diverse and a large number of systems are used for image and video predictions in the context of the MediaEval competition, one of the noticeable trends is that many of the top performing systems use some sort of fusion scheme. In general fusion is defined as "a technology to enable combining information from several sources in order to form a unified picture" [53], therefore it involves combining the power of multiple detection systems in order to create a better final system. For the methods analyzed in this context, fusion is applied at feature level (also called *early fusion*), at decision level (also called *late fusion* or *ensemble learning*) or a combination of the two.

2.2 Ensembling Systems

Late fusion, also knows as ensembling systems or decision-level fusion, consist of a set of initial predictors, called *inducers*, that are trained and tested on the dataset, whose prediction outputs are combined in the final step in order to create a new and improved set of predictions. These systems have a long history and are shown to be particularly useful in scenarios where the perfomance of single-system approaches is not considered satisfactory. While their usefulness is proven even in some traditional tasks, such as video action recognition [32], recently there is a noticeable trend of employing such approaches in subjective tasks, that seek to analyze the human perception of multimedia data. Some examples for this trend would include the prediction of media memorability [33], violence detection in videos [34], emotional content analysis [35], and media interestingness prediction [29].

One important theoretical aspect of ensembling systems is formulated by Wolpert [36], stating that, given an ensemble of N inducers, trained in a similar way, it is improbable that the prediction outputs of these inducers are completely uncorrelated. Thus, promoting a high level of diversity in the inducer set may improve the final result of the ensemble. Recently, Liu et al. [37] show that ensemble error may decrease as the inducer error decreases and inducer diversity increases. These aspects and many more are analyzed in depth in several ensembling literature review papers [39, 38].

Regarding the *ensembling functions*, the methods that are used in combining inducer prediction outputs, while there is a high variety among them, deep neural networks still represent a novelty for this domain. To the best of our knowledge, our works in using deep neural networks as the primary ensembling function is one of the first attempts in this direction. So far ensembling functions are dominated by simple statistical methods [46], such as late fusion via weighted arithmetic mean calculation, voting systems, etc. Other more complex approaches employ methods that require an initial learning step, including Boosting approaches such as AdaBoost [40], Gradient Boosting [41] or XGBoost [42], Bagging [43] or Random Forests [44]. While these approaches have been successfully implemented in several tasks, our assumption is that, with the introduction of deep neural networks as the main ensembling function, late fusion results will significantly improve. In our work we will use two approaches as comparison baseline for our proposed prediction method, namely statistical methods and boosting.

One example of a statistical approach is the *weighted late fusion*. Under this scheme, given a set of N inducer methods, $A = [a_1, a_2, ..., a_N]$ that create a set of prediction outputs denoted $Y = [y_1, y_2, ..., y_N]$, the goal of a weighted late fusion approach is to create a set of weights, $W = [w_1, w_2, ..., w_N]$, that, once applied to the prediction outputs Y, represent better predictors for the dataset that is being studied. In other words, weighted late fusion creates a new prediction output denoted y_w, that is calculated as follows:

$$y_w = \frac{y_1 \cdot w_1 + y_2 \cdot w_2 + ... + y_N \cdot w_N}{N} \tag{1}$$

The goal of this approach is to minimize the prediction error ϵ, so that the new prediction output $\epsilon_w < \epsilon_i, i \in [1, N]$. Several types of strategies can be employed in choosing the values of W. The most common strategy involves ordering the Y vector according to inducer performance, i.e., $\epsilon_1 < \epsilon_2 < ... < \epsilon_N$. This would allow systems to assign higher weights for better inducers, thus making sure that the top performing inducers dictate the final result. Working under the assumption that the vector is ordered, some such schemes would be:

$$\begin{cases} w_i = \frac{1}{i}, i \in [1, N] \\ w_i = \frac{1}{e^i}, i \in [0, N-1] \\ w_i = 1 - (\epsilon_i - \epsilon_1), i \in [1, N] \end{cases} \tag{2}$$

Boosting approaches represent another important class of ensemble learning techniques. In general, boosting can be defined as an iterative way of adding inducers into a final ensemble system, while updating the weights assigned to each inducer as more inducers are added in the system. While there are major differences between different boosting approaches, such as AdaBoost and Gradient Boosting, the overarching idea is the sequential training of inducer weights, i.e., trying to adjust the learning process so that it can correct preceding errors.

AdaBoost identifies weaknesses in the inducers in each learning step, represented by miss-classified data points, and assigns higher internal weights for those points, under the assumption that this will allow the next classifiers in the ensembling scheme to correct these errors. Therefore, given a set of data points, $x_i, i \in [1, M]$, initially all the weights for these data points are set to $w_i = 1/M$. The total error can be calculated for each individual inducer $a_j, j \in [1, N]$ as :

$$err_j = \frac{\sum_{i=1}^{M} w_i \cdot \mathcal{I}(C(x_i) \neq y_{j,i})}{\sum_{i=1}^{M} w_i} \tag{3}$$

where $\mathcal{I}$ is a function that outputs 1 for a true positive or negative prediction and 0 for a false positive or negative one and C represents the new classification rule created by the ensembling scheme. Also, given the α factor for each inducer, the system will update the w_i weights accordingly:

$$\alpha_j = \ln \frac{1 - err_j}{err_j} \tag{4}$$

$$w_i = w_i \cdot e^{\alpha_j \cdot \mathcal{I}(C(x_i) \neq y_{j,i})} \tag{5}$$

Thus, considering k as the set of the possible prediction classes associated with the prediction task, the new output can be expressed as:

$$C(x) = \max_k \sum_{i=1}^{M} \alpha_i \cdot \mathcal{I}(y_i(x) = k) \tag{6}$$

Gradient boosting, on the other hand, does not focus on individual data points, but on finding the difference between prediction sets and ground truth data. Therefore, the goal of this method is the minimization of the loss function $L(g, y)$, where y represents the prediction output of the method, while g represents the ground truth values for the given samples. Practically, the goal is to create a new ensembling function $\hat{F}$ that best approximates the ground truth of the dataset:

$$\hat{F} = \min_y \sum_{i=1}^{N} L(g, y) \tag{7}$$

While going through consecutive calls of the training loop, gradient boosting methods seek to apply gradient descent for optimizing the ensembling result. The final version of the ensembling function $\hat{F}$ can therefore be expressed as a weighted sum computed over a set of approximation functions h, starting from the initial version F_0 for this function:

$$\hat{F} = \sum_{i=1}^{M} w_i \cdot h_i + F_0 \tag{8}$$

where M represents the number of training steps. The function is then updated, based on its previous values, as follows:

$$w_m = \min_w \sum_{i=1}^{N} L(g, w \cdot h_m) \tag{9}$$

$$F_m = F_{m-1} + w_m \cdot h_m \tag{10}$$

3 Deep Ensembling

In a general sense, ensembling systems are represented by an algorithm or function $\mathcal{F}$, that, given a set of M dataset samples denoted S and a series of N algorithms denoted A, uses the classification or regression outputs of all the N algorithms, called *inducers*, and by combining them can create a new output for each of the M samples. Individual elements of the sample set can be represented as $s_i, i \in [1, M]$, representing a vector $S = [s_1, s_2, ..., s_M]$, while the series of algorithms can be represented by a set of functions $a_j, j \in [1, N]$, representing a vector $A = [a_1, a_2, ..., a_N]$.

Therefore, a matrix Y (see Equation 11) that contains elements $y_{i,j}, i \in [1, M]$ and $j \in [1, N]$ can be constructed, containing the prediction outputs of each inducer for each individual sample, where each row represents inducer outputs for a certain sample.

$$Y = \begin{bmatrix} y_{1,1} & \cdots & y_{1,N} \\ . & . & . \\ . & . & . \\ . & . & . \\ y_{M,1} & \cdots & y_{M,N} \end{bmatrix} \tag{11}$$

Obtaining the final ensembled prediction output for a single sample i consists of using the $[y_{i,1}, y_{i,2}, ..., y_{i,N}]$ inducer output vector as inputs for the ensembling function $\mathcal{F}$, thus obtaining the final prediction value o_i. This entire process is presented in Figure 1. While some variants of the ensembling methods can be represented by simple mathematical functions, i.e., calculating the average value of the inducer output vector, other functions can be more complex and can require a preliminary learning stage, such as boosting methods, as shown in Section 2.2. We propose a different perspective in which the ensembling function is represented by deep neural networks that will process inducer prediction output values.

It is also interesting to note that, while in more complex cases, such as multi-label regression, the predictions created by the inducers do not represent a single value, as one output probability is assigned to each of the possible labels, in our case inducers output a single value, representing the degree of interestingness assigned to each image or video sample. Therefore the $y_{i,j}$ values are uni-dimensional.

With this general framework in mind, we will present in the following sections some new perspectives, consisting of several types of deep neural networks that are used as ensembling functions for the task of predicting media interestingness. Our assumption in this case is that DNNs are able to better understand the patterns and biases that individual inducers have towards the samples in the dataset. Our proposed DNN models will only use the inducer outputs in determining the final prediction score, so image and video samples will not be fed into the ensemble algorithm.

We investigate four types of DNN architectures as follows: (i) a dense layer-based approach, that is the augmented with (ii) attention layers, (iii) convolutional layers, and finally, (iv) Cross-Space-Fusion layer (CSF), a novel approach designed for

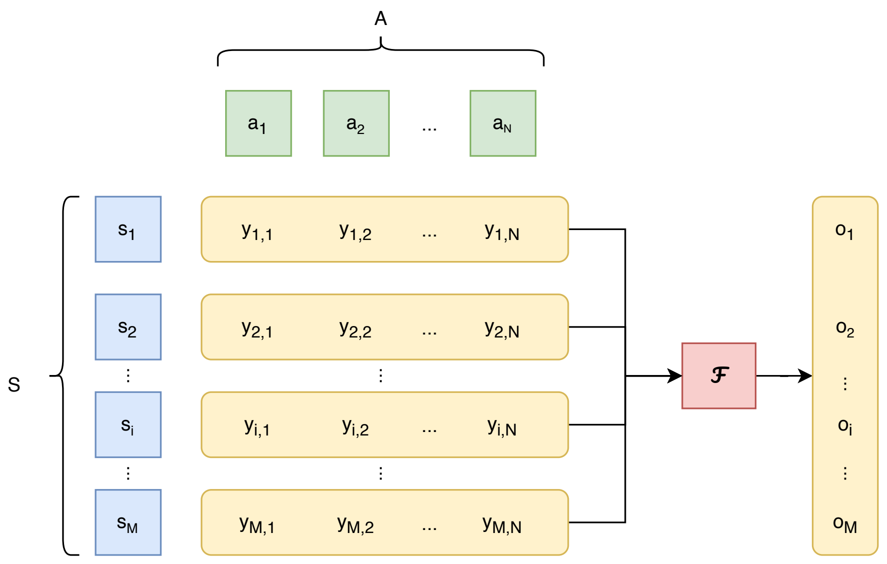

Fig. 1 General presentation of ensembling systems. The sample dataset, denoted S contains M elements, the set of inducers A contains N elements, while the inducer outputs for the samples are denoted as $y_{i,j}, i \in [1, M], j \in [1, N]$. The ensembling algorithm is denoted $\mathcal{F}$ and it produces a set of M prediction outputs, corresponding to each sample, denoted $o_i, i \in [1, M]$. Samples are represented with blue color, inducer algorithms with green, prediction outputs with yellow and the ensembling function with red.

parsing inducer vectors. While the first two types of network do not need any special data pre-processing, the latter two, namely convolutional and CSF, are designed to process data based on the spatial arrangement of data and understand how adjacent elements in a matrix can be interpreted in order to obtain a prediction. While this is heavily exploited in images and videos by convolutional layers, inducer output vectors have no intrinsic spatial arrangement and correlation, and therefore, some data pre-processing and decoration schemes that create spatial information are necessary for these two final types of neural networks, which we will present along with the implementation of the respective DNN models. One of the main reasons we theorize that such structures are able to create better ensembling systems is the ability of neural networks to accurately use various types of input data and classify this data into output predictions. While not directly attempting to model human behaviour and understanding of visual interestingness, we believe these models are able to model inducer behaviour and understanding, thus being able to learn the positive and negative biases of inducers towards visual samples. Thus, while the approaches presented here are centered around the prediction of visual interestingness, they are domain-independent and are useful in other tasks as well [31].

3.1 Dense Networks

Dense networks composed of fully connected (or dense) layers arguably represent one of the most popular DNN implementations. Given the innate ability of dense layers to correctly detect patterns in the input data and accurately classify samples, we theorize that, by using a set of connected dense layers, our proposed method will be able to accurately learn the correlation between inducer biases [14], allowing combinations of inducers to support or dismiss their predictions, based on the patterns the networks learns. Another component of the final network is represented by the addition of batch normalization layers [15], between the individual dense layers, with the role of helping the improving the network's learning process and speeding it up. Several variations of the dense network setup are tested, in order to ensure optimal performance. We present the optimal network architecture search method in Algorithm 1. We therefore change the depth of the network, by testing various numbers of layers in the network (5, 10, 15, 20, 25) and the width of the network by changing the number of neurons per layer (25, 50, 500, 1000, 2000). The third parameter in this search algorithm is represented by the presence or absence of batch normalization layers. Also, in Algorithm 1, the $processDense$ function has the role of both creating the network according to the three variable parameters and the role of training and testing the created network. A schematic view of the dense network architecture is presented in Figure 2.

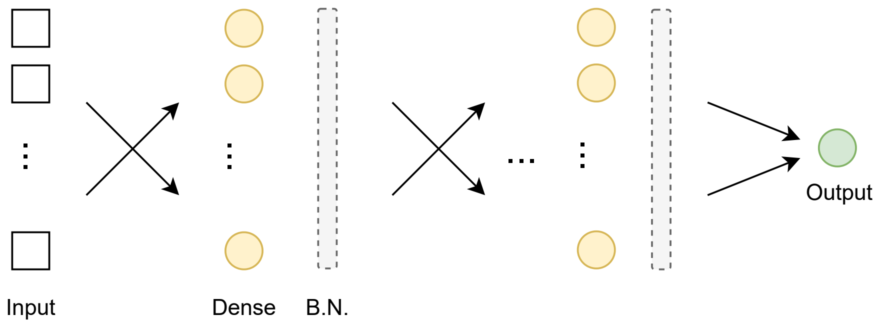

Fig. 2 A schematic presentation of the Dense architecture, presenting the variable width and depth of the network, as well as the presence or absence of the batch normalization layers.

3.2 Attention Augmented Dense Networks

Though computational attention mechanisms [16] were initially predominantly used in works that dealt with text processing and translation, it was quickly adopted in other domains, including computer vision [17]. In a general sense, attention mechanisms

Algorithm 1: Optimal dense network parameter search method.

```
Output : settings for optimal Dense network $best\_neuron, best\_layer, best\_bn$
begin
    //initiate the parameter options for the search algorithm
    $neurons \leftarrow [25, 50, 500, 1000, 2000, 5000]$;
    $layers \leftarrow [5, 10, 15, 20, 25]$;
    $bn \leftarrow [False, True]$ $best\_metric = 0.0$;

    //start searching for the best architecture
    for $i \leftarrow 0$ to 5 do
        for $j \leftarrow 0$ to 4 do
            for $k \leftarrow 0$ to 1 do
                //compute metric for current settings
                $metric \leftarrow processDense(neurons[i], layers[j], bn[k])$;

                //save these settings if they perform better
                if $metric > best\_metric$ then
                    $best\_neuron \leftarrow neurons[i]$;
                    $best\_layer \leftarrow layers[j]$;
                    $best\_bn \leftarrow bn[k]$;
                    $best\_metric \leftarrow metric$;
                end
            end
        end
    end
end
```

have the role of understanding and detecting the parts in the input space that are most important for the final prediction stage and assigning higher weights for the important parts. While in a general computer vision these mechanisms would infer the most important parts in images or videos, the intuition in our ensembling system is that the attention layer will create a set of weights w that will indicate the relevance of each of the values from the inducer output vector $[y_{i,1}, y_{i,2}, ..., y_{i,N}]$. The implementation we choose for our experiments consists of a soft attention layer inserted into the dense architecture presented in Section 3.1, as presented in Figure 3. Using the notation in Equation 12 that represents the network input space for a single sample i, and the soft attention vector as $attn_i$, with values between 0 and 1, the system will create an appropriate attention mask $\widehat{attn_i}$, computed as the element wise product of the input vector and the attention vector, as shown in Equation 13. The learning process for the attention mechanism is based on a supervised back-propagation approach:

$$\overline{y_i} = \left[y_{i,1},\ y_{i,2},\ ...,\ y_{i,N} \right] \tag{12}$$

$$\widehat{attn_i} = attn_i \odot \overline{y_i} \tag{13}$$

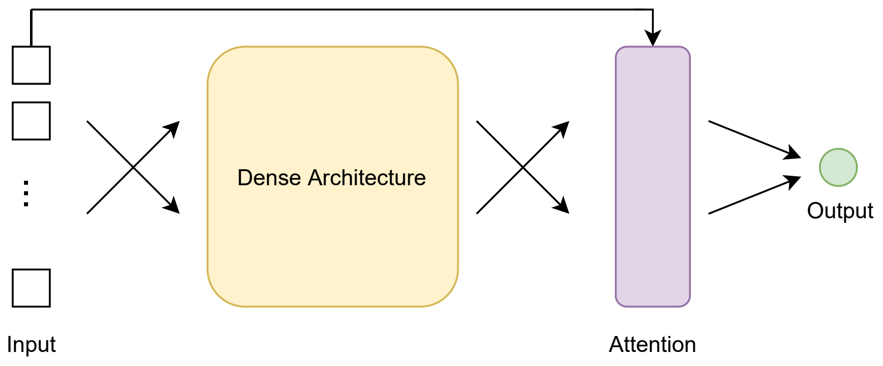

Fig. 3 A schematic presentation of the Attention augmented Dense networks. The attention mechanism, represented by an attention layer, is inserted into a Dense architecture.

3.3 Convolutional Augmented Dense Networks

Convolutional networks represented a big step forward for deep learning in the field of computer vision, aided by the advancement of hardware processing power and software libraries that allow such networks to be easily deployed and lower the processing time, starting with AlexNet's performance at the ILSVRC 2012 benchmarking competition [18]. While the shape of the input space is not important, as one, two or three dimensional convolutional networks have been implemented, they all rely on detecting and learning local correlations between adjacent elements in the input space. More to the point, convolutions can be represented by a set of filters of pre-determined shape that cover and process the entire input space. While this approach performs well for images and videos, that intrinsically have a spatial arrangement and correlation in the input space, in our particular case the order of the inducer prediction outputs in the $\overline{y_i}$ vectors does not have any intrinsic spatial correlation, and, furthermore, at this stage no relationships between individual inducers are calculated. Therefore, we must create these correlations and relationships, via a process we call *input decoration*. Our assumption in this case is that, by creating the decorated input vector for convolutional processing $\overline{dc_i}$ for each sample i and applying convolutional filters to this new input, we would be able create a system where similar inducers can be arranged in close spatial proximity and can support or revoke their prediction decisions based on their spatial relations. Two problems must therefore be solved in order to introduce convolutions into the ensemble networks: (i) find a criterion for detecting similarity between inducers, and (ii) create a spatial arrangement based on the similarity.

For the first problem, similarity between individual inducers can be calculated with the help of the official metric used for measuring system performance in the task. While in the case of interestingness mean average precision at 10 elements is used (mAP@10), in a generalized approach the metric can be expressed as a function $\mathcal{M}$, that takes two vectors as input (either ground truth data and prediction data or two prediction vectors from two separate systems), and outputs a value of similarity between them, denoted r. In other words, given a general form for a prediction vector $p_j = [y_{1,j}, y_{2,j}, ..., y_{M,j}]$, that represents the prediction vector created by inducer j for all the M samples in the dataset, the similarity value between two inducers m and n can be calculated as presented in Equation 14. Finally, by ordering the vector of similarity scores between an inducer m and all other inducers, we can create a list of the most similar inducers for each of the N inducers.

$$r_{m,n} = \mathcal{M}(p_m, p_n) \tag{14}$$

The second problem involves using the similarity values calculated at the previous step, and decorating the predictions for each sample based on the r values. The decorated input vector for a sample i is presented in Equation 15, and is composed of centroids built around the initial inducer prediction output values, denoted $s_1, s_2, ..., s_N$. The elements in each centroid, are as follows: (i) the central element, s_j, represents the initial value, (ii) the similarity scores for the first four most similar inducers, denoted $r_{1,j}$, ... $r_{4,j}$, and (iii) the prediction outputs for sample i extracted from the first four most similar inducers, denoted $c_{1,j}$, ... $c_{4,j}$. This decoration process for a single sample i is presented in Algorithm 2, and can easily be generalized to all the samples in the dataset.

$$\overline{dc_i} = \begin{bmatrix} r_{4,1} & c_{1,1} & r_{1,1} & \cdots & r_{4,N} & c_{1,N} & r_{1,N} \\ c_{4,1} & s_1 & c_{2,1} & \cdots & c_{4,N} & s_N & c_{2,N} \\ r_{3,1} & c_{3,1} & r_{2,1} & \cdots & r_{3,N} & c_{3,N} & r_{2,N} \end{bmatrix} \tag{15}$$

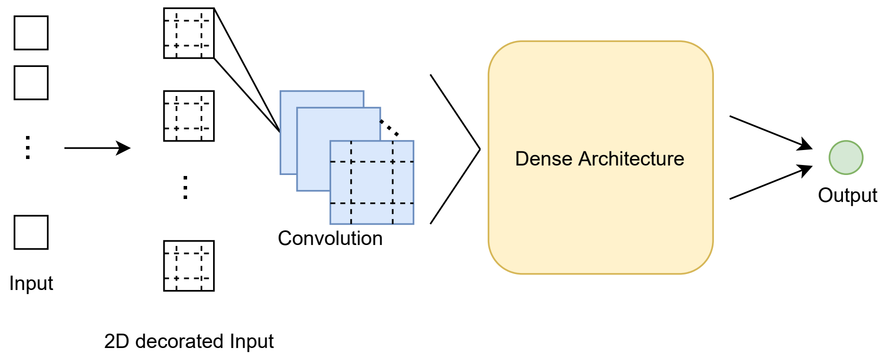

Fig. 4 A schematic presentation of the Convolutional augmented Dense networks. The convolutional layer, presented here with a varying number of filters, is preceded by the input decoration stage and inserted into the dense architecture.

The decorated $\overline{dc_i}$ array will represent the new input for the convolutional ensembling system, as presented in Figure 4. Finally, the $\overline{dc_i}$ array in processed by the convolutional layers, centroid by centroid. Equation 16 shows this process for a single centroid i, where the centroid is element-wise multiplied with the weights in

Algorithm 2: Input decoration algorithm for convolutional networks for sample i.

```
Input  : vector of inducer predictions $\left[p_1\ p_2\ \dots\ p_N\right]$
         vector of predictions for sample $i$ $\left[s_1\ s_2\ \dots\ s_N\right]$
         similarity (and metric) function $\mathcal{M}$
Output : decorated input $\overline{dc_i}$
begin
    //initialize the $\overline{dc_i}$ vector
    $\overline{dc_i} \leftarrow zeros(3 \times N, 3)$;

    //compute the similarity between inducers
    for $j \leftarrow 1$ to $N$ do
        $sim[j] \leftarrow zeros(N)$;
        for $k \leftarrow 1$ to $N$ do
            $sim[j][k] \leftarrow \mathcal{M}(p_j, p_k)$
        end
        //order the $sim_j$ vectors
        $sim[j] \leftarrow OrderDescending(sim[j])$;
    end

    //decorate the input
    for $i \leftarrow 1$ to $N$ do
        //initialize centroid
        $cent_i \leftarrow zeros(3, 3)$;
        for $j \leftarrow 1$ to 4 do
            //insert elements into the centroid according to their proper placement based
              on the similarity measure, as presented in Equation 15
            $cent_i \leftarrow InsertElems(sim[i, j])$
        end

        //insert the centroid in the decorated input vector
        $\overline{dc_i} \leftarrow InsertCentroid(cent_i)$
    end
end
```

the convolutional filter. The final step involves, in our case, an average pooling layer that will output a single element for the convolutional step that represents the average value of the element-wise multiplication result matrix. In a simple case where only one convolutional filter is employed, the input to the dense layers will practically be similar as the initial input, where each inducer output value is basically replaced by the result of the convolution process for the inducer's centroid. Finally, several setups will be tested for the convolutional architecture, that include different number of convolutional filters: 1, 5 or 10 filters. This would allow the network to assess more than one type of correlation between the inducers.

$$\begin{bmatrix} r_{4,i} & c_{1,i} & r_{1,i} \\ c_{4,i} & s_i & c_{2,i} \\ r_{3,i} & c_{3,i} & r_{2,i} \end{bmatrix} \odot \begin{bmatrix} w_1 & w_2 & w_3 \\ w_4 & w_5 & w_6 \\ w_7 & w_8 & w_9 \end{bmatrix} \tag{16}$$

3.4 Cross-Space-Fusion Augmented Dense Networks

With the introduction of convolutional layers in the network a method that can process the similarities between inducers has been created. However, convolutional networks are created with image processing as their main objective and use the same filters for processing the entire image and therefore would, in the case of ensembling systems, share the same weights between different centroids. While this does represent a step forward in processing inducer correlation, our assumption is that correlation between inducers are different for each individual inducer, and therefore weights should not be shared between centroids. Given this assumption, we propose the creation of a novel type of DNN layer, which we name *"Cross-Space-Fusion"*, or *CSF* layer. The implementation of the CSF layer is based on creating a new input decoration method and the creation of the layer itself.

A few architectural decisions must be taken in order to fully exploit the correlation data we generate and overcome the possible limitations of convolutional processing. First of all, as shown in Equation 16, inducer outputs and similarity scores are not processed together, each one of them being multiplied separately with its corespondent convolutional weight. This may break the correlation between the two elements and make it harder to process and learn in the neural network. Secondly, the same possible issue would appear no matter what type of convolutional layer we would use, as three-dimensional convolutional layers do not process correlations inter-dimensionally. Therefore, we propose a novel input decoration method, that would create an additional, third dimension, that would separately memorize similar inducer outputs and similarity scores. Also the CSF layer would need to process these details across the third dimension of the array, processing inducer outputs and corespondent similarity scores together, while using the same $\mathcal{M}$ presented in Equation 14 function for calculating similarity scores. Finally, as previously mentioned, we must take into account that regular convolutional filters may not be the optimal for learning correlations, as they may be different from centroid to centroid. Thus a larger number of parameters must be designed into the CSF layer and, while this may represent a strain on the neural network, the number of added parameters is still small, especially when compared with the depth and width of the dense architecture.

Given the particularities of this approach, Equation 17 presents the new version of the decorated input, where C_i represents the matrix of prediction outputs from the 8 most similar inducers for an inducer i, while R_i represents their respective similarity score, calculated with the help of the $\mathcal{M}$ function. These two matrices create the third dimension of the decorated input, as shown in Figure 5. Similar to the convolutional approach, in this example, the $c_{1,i}$ and $r_{1,i}$ pair represents the prediction output and similarity score of the most similar system with inducer i, $c_{2,i}$ and $r_{2,i}$ the second most similar, and so on. While it is obvious that by using this decoration scheme more similar inducers can be added to the system that in a similar convolutional approach, the question of their utility for this task still remains and will be analyzed, as it may be possible that the new data inserted into the system is noisy or little real correlation exists between the systems.

$$C_i = \begin{bmatrix} c_{1,i} & c_{2,i} & c_{3,i} \\ c_{8,i} & s_i & c_{4,i} \\ c_{7,i} & c_{6,i} & c_{5,i} \end{bmatrix}, R_i = \begin{bmatrix} r_{1,i} & r_{2,i} & r_{3,i} \\ r_{8,i} & 1 & r_{4,i} \\ r_{7,i} & r_{6,i} & r_{5,i} \end{bmatrix} \tag{17}$$

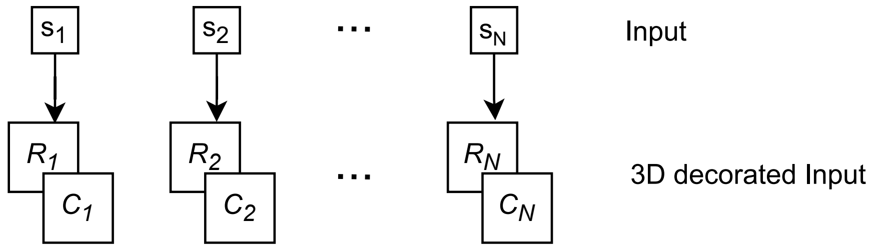

Fig. 5 A schematic presentation of the three-dimensional input decoration result. For each inducer output s_i, we create a pair of centroids C_i and R_i, containing the prediction outputs and similarity scores for similar inducers, as presented in Equation 17.

Algorithm 3 presents this input decoration algorithm. It is worth to note that, in the case of the CSF approach, the shape of the $\overline{dc_i}$ decorated input array changes once more, from $(3 \times N, 3)$ in the convolutional approach to $(3 \times N, 3, 2)$, doubling in size. After the decoration step, the input is fed into the CSF layer. For each (c_i, r_i) group of centroids, the network must create and learn a set of weights that can combine the initial inducer prediction with the prediction outputs and similarity scores grouped in the centroids. Thus, the CSF layer contains a set of α and β parameters that must be learned. Equation 18 describes the operations that are performed by the CSF layer, where α are used for controlling the prediction output of each inducer i and β parameters are used for controlling the prediction outputs and similarity scores for the inducers similar to i.

$$\begin{bmatrix} \frac{\alpha_{1,i} \cdot s_i + \beta_{1,i} \cdot c_{1,i} \cdot r_{1,i}}{2} & \frac{\alpha_{2,i} \cdot s_i + \beta_{2,i} \cdot c_{2,i} \cdot r_{2,i}}{2} & \frac{\alpha_{3,i} \cdot s_i + \beta_{3,i} \cdot c_{3,i} \cdot r_{3,i}}{2} \\ \frac{\alpha_{8,i} \cdot s_i + \beta_{8,i} \cdot c_{8,i} \cdot r_{8,i}}{2} & s_i & \frac{\alpha_{4,i} \cdot s_i + \beta_{4,i} \cdot c_{4,i} \cdot r_{4,i}}{2} \\ \frac{\alpha_{7,i} \cdot s_i + \beta_{7,i} \cdot c_{7,i} \cdot r_{7,i}}{2} & \frac{\alpha_{6,i} \cdot s_i + \beta_{6,i} \cdot c_{6,i} \cdot r_{6,i}}{2} & \frac{\alpha_{5,i} \cdot s_i + \beta_{5,i} \cdot c_{7,i} \cdot r_{5,i}}{2} \end{bmatrix} \tag{18}$$

Figure 6 presents an outline of this approach. As presented, the final step in the CSF augmentation part of the method is represented by the addition of an average pooling layer, thus obtaining an input of equal dimensions as the initial one for the dense architecture. Also, given the number of inducers N, the final number of parameters in the CSF layer is $16 \times N$, with $8 \times N$ α and β parameters. As previously mentioned, we must also take into account the possibility that the addition of so many similar inducers in the centroid could add noise to the input and damage the final result. Thus, we decide to test two different setups for the CSF architecture: $4S$, where we only populate the (c_i, r_i) centroid pairs with the top-4 most similar inducers, and $8S$, where the centroid pairs are completely populated with 8 inducers. It is important to note that, while our experiments may show a preference for one of

Algorithm 3: Input decoration algorithm for Cross-Space-Fusion networks for sample i.

```
Input  : vector of inducer predictions $\left[p_1\ p_2\ \dots\ p_N\right]$
         vector of predictions for sample $i$ $\left[s_1\ s_2\ \dots\ s_N\right]$
         similarity (and metric) function $\mathcal{M}$
Output : decorated input $\overline{dc_i}$
begin
    //initialize the $\overline{dc_i}$ vector
    $\overline{dc_i} \leftarrow zeros(3 \times N, 3, 2)$;

    //compute the similarity between inducers
    for $j \leftarrow 1$ to $N$ do
        $sim[j] \leftarrow zeros(N)$;
        for $k \leftarrow 1$ to $N$ do
            $sim[j][k] \leftarrow \mathcal{M}(p_j, p_k)$
        end
        //order the $sim_j$ vectors
        $sim[j] \leftarrow OrderDescending(sim[j])$;
    end

    //decorate the input
    for $i \leftarrow 1$ to $N$ do
        //initialize centroid pair $c_i, r_i$
        $c_i \leftarrow zeros(3, 3)$;
        $r_i \leftarrow zeros(3, 3)$;
        for $j \leftarrow 1$to4 do
            //insert elements into the centroid pairs according to their proper placement
              based on the similarity measure, as presented in Equation 17
            $(c_i, r_i) \leftarrow InsertElems(sim[i, j])$
        end

        insert the centroid in the decorated input vector
          $\overline{dc_i} \leftarrow InsertCentroid(c_i, r_i)$
    end
end
```

the two setups, in other experiments that may use other datasets or inducers these results may be opposed, or other setups using a different number of populated similar inducers may produce better results.

4 Experimental Setup

This section will present the main components of the experiment and how these components interact. We will describe the training protocol employed for the experiments, the dataset and the evaluation protocol used for obtaining the results.

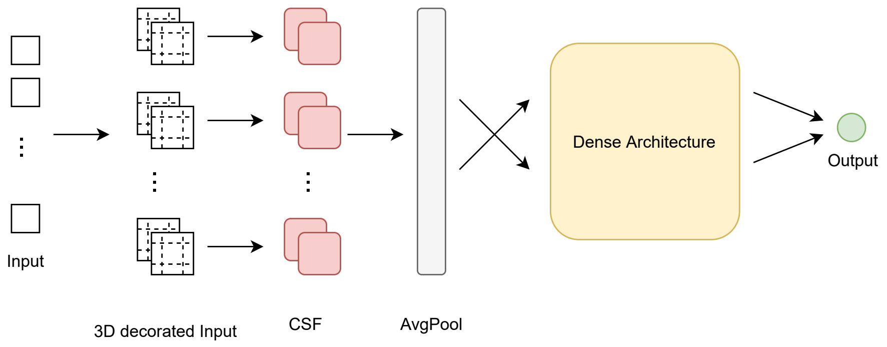

Fig. 6 A schematic presentation of the Cross-Space-Fusion augmented Dense networks. The CSF layer is preceded by the input decoration stage and inserted into the dense architecture.

4.1 Training Protocol

The common component in all the methods presented in Section 3 is represented by the dense architecture deep neural network. Our experiments will therefore start with finding an optimal dense architecture with regards to the depth and width of the network and the positive or negative influence of batch normalization layers, using the values presented in Section 3.1. This is done by collecting the prediction outputs of the entire set of inducers and feeding them into the different variations of the dense architecture networks. This step is described with Algorithm 1. In the following steps, the optimal dense network is augmented with attention, convolutional and CSF layers. As special implementations of the convolutional and CSF layers, the input, consisting of the prediction outputs, is decorated, according to Algorithm 2 for the convolutional approach and Algorithm 3 for the CSF approach.

The training process is performed for 50 epochs, for each variation of the network, using a batch size of 64 samples, mean squared error loss function and an Adam [45] optimizer featuring a learning rate of 0.01. We are interested in pointing out the optimal dense architecture, given the set of search parameters, as well as the effect of augmenting the dense network with the three types of layers: attention, convolutional and CSF.

4.2 Dataset

For our experiments we are using the latest version of the Interestingness10k [19] dataset, validated and used during the MediaEval 2017 Predicting Media Interestingness task [13]. The dataset is composed of 9,831 images and videos, split between 7,396 samples included in the development set (devset) and 2,435 samples in the testing set (testset). Participants to the benchmarking competition were tasked with developing and training their media interestingness prediction methods on the devset, running the systems on the testset samples and submitting their testset predictions to the task organizers for performance calculation.

Given the high number of systems submitted at the benchmarking competition, i.e., 33 for the image task and 42 for the video task, and the considerable amount of research and work that went into creating them, we consider these systems as ideal candidates for being used as inducers in our proposed method. With the help and collaboration of the task organizers, we gathered participant submission files and used them as input into our systems. However, given the fact that participants only submitted predictions for the testset samples and the inherent problems in recreating such a large number of diverse systems, we are bound to only use those predictions and create a new evaluation protocol that will be used in training our systems, based only on the samples that are featured in the testset.

We therefore have to create a new set of data splits, and choose to use two protocols for this: (i) RSKF75, featuring a random stratified k-fold that uses 75% of the samples for training and 25% for testing, and (ii) RSKF50, generating 50% training samples and 50% testing. It is important to note that, in order to avoid any "lucky" data splits that would create an unfair advantage for our approach, the split samples are randomized, and experiments are repeated with different random splits, generating 100 partitions for each network architecture variation. Therefore, the results we present in Section 6 are average values calculated over the 100 partitions. System performance is calculated by using the official metric of the MediaEval benchmarking competition, i.e., MAP@10.

5 Experimental Results

This section presents the experimental results, featuring a comparison with a set of baseline systems, a set of baseline ensembling approaches and identifying the best performing architectures.

5.1 Baseline Systems

In order to correctly position and analyze the results of the proposed methods, we compare them with a few methods from the literature, including (i) the best performers at the MediaEval competition, (ii) the best overall performers on the Interestingness10k dataset, and (iii) a set of traditional ensembling methods.

The best performers from the MediaEval competition also represent inducers for our systems, and an important target for the proposed systems. For the image prediction task we have the system developed by Permadi et al. [26], with a MAP@10 performance of 0.1385, while for video prediction we have the system developed by Ben-Ahmed et al. [27], with a MAP@10 performance of 0.0827. The overall performers consist of methods that are published outside the MediaEval venue, but used the same benchmarking protocol and metrics. For the image task we have the work of Parekh et al. [28], with a perfomance of $MAP@10$ = 0.156, while for the video task, Wang et al. [29] achieve a $MAP@10$ = 0.093.

The final set of baseline systems consists of a set of traditional ensembling methods, that we created using the same protocol and set of inducers as used by our proposed methods. Several types of ensembling methods are tested, starting with simple strategies [46] like taking the maximum value of inducer prediction outputs (*LFMax*), average and mean values (*LFAvg* and *LFMean*), and weighted average (*LFWeight*), but also more complex approaches that involve learning steps, like AdaBoost [40] (*BAda*) and Gradient Boosting [41] (*BGrad*).

5.2 Results

The results are presented in Table 1. At a first glance, it is important to note that the proposed systems surpasses every baseline system, including the best performing baseline ensembling system, which for both images and videos is the AdaBoost approach. Furthermore, the best performing variant of the proposed systems increased performance by a large margin. Taking into account the RSKF75 split, the increase is as follows: for the image subtask an increase of 148.08% over the best MediaEval system, 73.09% over the best overall system and 105.25% over the best traditional ensembling system, while for the video subtask these values are 241.59%, 203.76% and 150.22% respectively.

Table 1 Results on the two interestingness prediction tasks: image and video. Systems are divided into baseline best performers from MediaEval and from the literature (b), best baseline ensembling performance (e) and proposed systems (p), and according to the split the systems employ (original or RSKF50 and RSKF75). The best results with regards to the official metric (MAP@10) are presented in bold.

	Image			Video		
System type	System	Split	MAP@10	System	Split	MAP@10
(b)	Permadi et al. [26]	original	0.1385	Ben-Ahmed et al. [27]	original	0.0827
	Parekh et al. [28]	original	0.1985	Wang et al. [29]	original	0.093
(e)	BAda [40]	RSKF50	0.1523	BAda [40]	RSKF50	0.0961
		RSKF75	0.1674		RSKF75	0.1129
(p)	Dense	RSKF50	0.2316	Dense	RSKF50	0.1563
		RSKF75	0.3355		RSKF75	0.2677
	Attention	RSKF50	0.2399	Attention	RSKF50	0.1668
		RSKF75	0.3389		RSKF75	0.2750
	Convolutional	RSKF50	0.2293	Convolutional	RSKF50	**0.1692**
		RSKF75	**0.3436**		RSKF75	0.2799
	CSF	RSKF50	**0.2403**	CSF	RSKF50	0.1664
		RSKF75	0.3403		RSKF75	**0.2825**

With regards to the overall best performing proposed method, results vary, as the convolutional approach has the best results on the image task using the RSKF75 split (MAP@10 = 0.3436) and in the video task using the RSKF50 split (MAP@10 = 0.1692), while the CSF approach has the best results using the other two variants, obtaining a MAP@10 value of 0.2403 for image prediction under the RSKF50 setup and 0.2825 for video prediction under the RSKF75 setup.

Table 2 Progressive analysis of network setup for the video task, under RSKF75 setup with batch normalization layers activated. The left side of the table shows results when the number of neurons increases, while the right side shows results when the number of layers increases.

Layers	5	5	5	5	5	5	5	10	15	20	25
Neurons	25	50	500	1,000	2,000	5,000	25	25	25	25	25
MAP@10	0.2414	0.2410	0.2493	**0.2529**	0.2506	0.2094	0.2414	0.2529	0.2650	**0.2660**	0.2646

It is also important to note the architecture variations that led to these results, i.e., the optimal dense, convolutional, and CSF architecture setups. For image prediction, the optimal dense architecture uses 10 layers with 1,000 neurons per layer, and no batch normalization, achieving MAP@10 values of 0.2316 for RSKF50 and 0.3355 for RSKF75, while the best performing convolutional architecture uses 5 filters. Also, the best performing CSF setup in this case is $4S$. For video prediction, the optimal dense setup is composed of 25 layers with 2,000 neurons per layer and features batch normalization, achieving MAP@10 values of 0.1563 for RSKF50 and 0.2677 for RSKF75. With regards to the convolutional architecture, the best setup again features 5 convolutional filters, while $4S$ again represents the best setup for the CSF layer. While the dense network performance is very good, the augmentation process with attention and especially convolutional and CSF layers further improves the results.

One final observation with regards to network setup is presented in Table 2. During our experiments, we observed that there are certain points when the network stops learning and achieves saturation. While Table 2 presents a particular setup, for the video task with batch normalization layers and RSKF75 split, the same behaviour is observable regardless of the task, of the presence of batch normalization layers or of the split. In the example presented, increasing the number of neurons past 1,000 while keeping the number of layers constant at 5 only decreases the performance, while the same is true when increasing the number of layers past 20 when maintaining a constant number of 25 neurons per layer. Most importantly, this seems to indicate that the optimal network setup is not outside the set of values we tested in our experiments. Another important point to make here is that the proposed method have a high performance even when looking at the values for the most basic setup (5 layers, 25 neurons per layer), scoring a MAP@10 value of 0.2414, 9.82% lower than the best performing dense architecture, but still significantly better than all the selected baseline methods.

6 Conclusions

This work presents the creation and deployment of a series of deep neural network based ensemble systems, used in the prediction of image and video interestingness. The latest Interestingness10k dataset is used in our experiments, a dataset that was previously used and validated during the MediaEval 2017 Predicting Media Interestingness task. Though a large number of systems use this dataset, both during the MediaEval benchmarking competition and outside it, in different journals and conferences, system performance is generally low when compared with other tasks, i.e., a maximum MAP@10 performance of 0.1985 for image interestingness prediction and 0.093 for video prediction.

While very high, near-perfect performance is not necessarily expected for such tasks, where annotator subjectivity plays an important role, we theorize that the implementation of ensemble systems can increase overall performance. Furthermore, the exploration of deep neural networks as ensembling functions presents a high degree of novelty in the current literature, as current literature shows that they are only employed as inducers and not as ensembling functions. Different network setups are presented and tested, including architectures based on dense, attention, convolutional and CSF layers, presenting the theoretical background of implementing these architectures as ensemble functions and the introduction of input decoration algorithms that allow inducer prediction output data to be used and inducer correlations learned with the help of these architectures.

Experimental results show a significant increase in performance over state-of-the-art systems. Our proposed methods show a 148.08% increase in performance in the image prediction task over the best MediaEval system and 73.09% over the best state-of-the-art system, while in the video prediction task the increase is even higher: 241.59% and 203.76%. Furthermore, the proposed ensemble methods are compared with some traditional ensembling methods implemented under the same conditions, having a significantly better performance, i.e., 105.25% for the image task and 150.22% for the video task. While it is certainly possible that better results could be achieved with other network setups, featuring different number of layers or neurons, or different architectures, we believe the advantages of deep fusion systems to be thoroughly demonstrated. Given the results, it is still unclear which of the two inducer correlation based architectures (convolutional or CSF) perform better for this task, with top results being split between them. However it is important to note that inducer correlation processing did indeed improve the results of both the dense networks and the attention-based networks, thus indicating the validity of inducer correlation calculation, input decoration and correlation processing.

Finally, another important point, not only for our proposed methods, but for ensembling systems in general, is the analysis of the deployability of the proposed systems. While using a late fusion approach can be cost intensive, considering that inducers must be trained, tested and run individually, and a final ensembling step performed before the final prediction is provided, there are cases where developing a late fusion system can become a necessity. Critical infrastructure applications, where very accurate prediction results are a constant need, represent a good example, but, closer to the domain of interestingness prediction, applications where single-method approaches do not perform well, due to inherent multi-modality or complexity of the concept that is being predicted, represent another good example. While deploying an ensembling method may prove to be more costly, it may also be one of the only methods that achieves market-level performance, allowing the introduction of new features that can greatly increase user satisfaction. In this case we also consider the possibility that lowering the number of inducers may not affect system performance to a high degree, therefore trading an insignificant amount of performance for higher execution speed and lower hardware demands. While the creation of an inducer selection method is still an open question for our approach, we propose that future developments could address this problem by analyzing inducer correlations or by testing performance in a recursive leave-one-out scenario.

Acknowledgement

This work was funded under project AI4Media “A European Excellence Centre for Media, Society and Democracy”, grant #951911, H2020 ICT-48-2020.

References

1. He, K., Gkioxari, G., Dollár, P., Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 2961-2969).
2. Yalniz, I. Z., Jégou, H., Chen, K., Paluri, M., Mahajan, D. (2019). Billion-scale semi-supervised learning for image classification. arXiv preprint arXiv:1905.00546.
3. Stevenson, A. (Ed.). (2010). Oxford dictionary of English. Oxford University Press, USA.
4. Berlyne, D. E. (1949). Interest as a psychological concept. British Journal of Psychology, 39(4), 184.
5. Chamaret, C., Demarty, C. H., Demoulin, V., Marquant, G. (2016). Experiencing the interestingness concept within and between pictures. Electronic Imaging, 2016(16), 1-12.
6. Silvia, P. J. (2005). What is interesting? Exploring the appraisal structure of interest. Emotion, 5(1), 89.
7. Silvia, P. J. (2009). Looking past pleasure: anger, confusion, disgust, pride, surprise, and other unusual aesthetic emotions. Psychology of Aesthetics, Creativity, and the Arts, 3(1), 48.
8. Constantin, M. G., Redi, M., Zen, G., Ionescu, B. (2019). Computational understanding of visual interestingness beyond semantics: literature survey and analysis of covariates. ACM Computing Surveys (CSUR), 52(2), 1-37.
9. Hidi, S., Anderson, V. (1992). Situational interest and its impact on reading and expository writing. The role of interest in learning and development, 11, 213-214.
10. Gygli, M., Soleymani, M. (2016, October). Analyzing and predicting GIF interestingness. In Proceedings of the 24th ACM international conference on Multimedia (pp. 122-126).
11. Hsieh, L. C., Hsu, W. H., Wang, H. C. (2014, May). Investigating and predicting social and visual image interestingness on social media by crowdsourcing. In 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) (pp. 4309-4313). IEEE.
12. Demarty, C. H., Sjöberg, M., Ionescu, B., Do, T. T., Wang, H., Duong, N. Q., Lefebvre, F. MediaEval 2016 Predicting Media Interestingness Task. In MediaEval Workshop, Hilversum, The Netherlands, October 20-21, 2016.

13. Demarty, C. H., Sjöberg, M., Ionescu, B., Do, T. T., Gygli, M., Duong, N. (2017, September). Mediaeval 2017 predicting media interestingness task. In MediaEval Workshop, Dublin, Ireland, September 13-15, 2017.
14. Mitchell, T. M. (1980). The need for biases in learning generalizations (pp. 184-191). New Jersey: Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ.
15. Ioffe, S., Szegedy, C. (2015, June). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In International conference on machine learning (pp. 448-456). PMLR.
16. Bahdanau, D., Cho, K., Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.
17. Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R.S., Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning (pp. 2048-2057). PMLR.
18. Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097-1105.
19. Constantin, M. G., Ştefan, L. D., Ionescu, B., Duong, N. Q., Demarty, C. H., Sjöberg, M. (2021). Visual Interestingness Prediction: A Benchmark Framework and Literature Review. International Journal of Computer Vision, 1-25.
20. Gygli, M., Grabner, H., Riemenschneider, H., Nater, F., Van Gool, L. (2013). The interestingness of images. In Proceedings of the IEEE International Conference on Computer Vision (pp. 1633-1640).
21. Jiang, Y. G., Wang, Y., Feng, R., Xue, X., Zheng, Y., Yang, H. (2013, June). Understanding and predicting interestingness of videos. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 27, No. 1).
22. Grabner, H., Nater, F., Druey, M., Van Gool, L. (2013, October). Visual interestingness in image sequences. In Proceedings of the 21st ACM international conference on Multimedia (pp. 1017-1026).
23. Jou, B., Chen, T., Pappas, N., Redi, M., Topkara, M., Chang, S. F. (2015, October). Visual affect around the world: A large-scale multilingual visual sentiment ontology. In Proceedings of the 23rd ACM international conference on Multimedia (pp. 159-168).
24. Tran, D., Bourdev, L., Fergus, R., Torresani, L., Paluri, M. (2015). Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision (pp. 4489-4497).
25. Fan, S., Ng, T. T., Koenig, B. L., Jiang, M., Zhao, Q. (2016). A paradigm for building generalized models of human image perception through data fusion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5762-5771).
26. Permadi, R. A., Putra, S. G. P., Helmiriawan, C., Liem, C. C. (2017). DUT-MMSR at MediaEval 2017: Predicting Media Interestingness Task. In Proceedings of the MediaEval 2017 Workshop, Dublin, Ireland.
27. Ben-Ahmed, O., Wacker, J., Gaballo, A., Huet, B. (2017). Eurecom@ mediaeval 2017: Media genre inference for predicting media interestingnes. In the Proceedings of the MediaEval 2017 Workshop, Dublin, Ireland.
28. Parekh, J., Tibrewal, H., Parekh, S. (2018, June). Deep pairwise classification and ranking for predicting media interestingness. In Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval (pp. 428-433).
29. Wang, S., Chen, S., Zhao, J., Jin, Q. (2018, October). 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 (pp. 55-61).
30. Ştefan, L. D., Constantin, M. G., Ionescu, B. (2020, June). System Fusion with Deep Ensembles. In Proceedings of the 2020 International Conference on Multimedia Retrieval (pp. 256-260).

31. Constantin, M. G., Stefan, L. D., Ionescu, B. (2021, June). DeepFusion: Deep Ensembles for Domain Independent System Fusion. In Proceedings of the 27th International Conference on Multimedia Modeling, Prague, Czech Republic.
32. Sudhakaran, S., Escalera, S., Lanz, O. (2020). Gate-shift networks for video action recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 1102-1111).
33. Azcona, D., Moreu, E., Hu, F., Ward, T. E., Smeaton, A. F. (2020, September). Predicting media memorability using ensemble models. In Proceedings of the MediaEval Workshop.
34. Dai, Q., Zhao, R. W., Wu, Z., Wang, X., Gu, Z., Wu, W., Jiang, Y. G. (2015, September). Fudan-Huawei at MediaEval 2015: Detecting Violent Scenes and Affective Impact in Movies with Deep Learning. In Proceedings of the MediaEval 2015 Workshop, Wurzen, Germany.
35. Sun, J. J., Liu, T., Prasad, G. (2018). Gla in mediaeval 2018 emotional impact of movies task. In Proceedings of the MediaEval 2018 Workshop, Sophia Antipolis, France.
36. Wolpert, D. H. (2002). The supervised learning no-free-lunch theorems. Soft computing and industry, 25-42.
37. Liu, L., Wei, W., Chow, K. H., Loper, M., Gursoy, E., Truex, S., Wu, Y. (2019, November). Deep neural network ensembles against deception: Ensemble diversity, accuracy and robustness. In 2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS) (pp. 274-282). IEEE.
38. Sagi, O., Rokach, L. (2018). Ensemble learning: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 8(4), e1249.
39. Gomes, H. M., Barddal, J. P., Enembreck, F., Bifet, A. (2017). A survey on ensemble learning for data stream classification. ACM Computing Surveys (CSUR), 50(2), 1-36.
40. Freund, Y., Schapire, R., Abe, N. (1999). A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence, 14(771-780), 1612.
41. Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.
42. Chen, T., Guestrin, C. (2016, August). Xgboost: A scalable tree boosting system. In Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining (pp. 785-794).
43. Breiman, L. (1996). Bagging predictors. Machine learning, 24(2), 123-140.
44. Breiman, L. (2001). Random forests. Machine learning, 45(1), 5-32.
45. Kingma, D. P., Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.
46. Kittler, J., Hatef, M., Duin, R. P., Matas, J. (1998). On combining classifiers. IEEE transactions on pattern analysis and machine intelligence, 20(3), 226-239.
47. Lowe, D. G. (1999, September). Object recognition from local scale-invariant features. In Proceedings of the seventh IEEE international conference on computer vision (Vol. 2, pp. 1150-1157).
48. Oliva, A., Torralba, A. (2001). Modeling the shape of the scene: A holistic representation of the spatial envelope. International journal of computer vision, 42(3), 145-175.
49. Stein, B. E., Stanford, T. R. (2008). Multisensory integration: current issues from the perspective of the single neuron. Nature reviews neuroscience, 9(4), 255-266.
50. Shechtman, E., Irani, M. (2007, June). Matching local self-similarities across images and videos. In 2007 IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-8). IEEE.
51. Ke, Y., Hoiem, D., Sukthankar, R. (2005, June). Computer vision for music identification. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) (Vol. 1, pp. 597-604). IEEE.
52. Demarty, C. H., Sjöberg, M., Constantin, M. G., Duong, N. Q., Ionescu, B., Do, T. T., Wang, H. (2017). Predicting interestingness of visual content. In Visual Content Indexing and Retrieval with Psycho-Visual Models (pp. 233-265). Springer, Cham.
53. Khaleghi, B., Khamis, A., Karray, F. O., Razavi, S. N. (2013). Multisensor data fusion: A review of the state-of-the-art. Information fusion, 14(1), 28-44.