Implicit and Explicit Concept Relations in Deep Neural Networks for Multi-Label Video/Image Annotation

Foteini Markatopoulou, Vasileios Mezaris, *Senior Member, IEEE,* and Ioannis Patras, *Senior Member, IEEE*

***Abstract*—In this work we propose a DCNN (Deep Convolutional Neural Network) architecture that addresses the problem of video/image concept annotation by exploiting concept relations at two different levels. At the first level, we build on ideas from multi-task learning, and propose an approach to learn concept-specific representations that are sparse, linear combinations of representations of latent concepts. By enforcing the sharing of the latent concept representations, we exploit the implicit relations between the target concepts. At a second level, we build on ideas from structured output learning, and propose the introduction, at training time, of a new cost term that explicitly models the correlations between the concepts. By doing so, we explicitly model the structure in the output space (i.e., the concept labels). Both of the above are implemented using standard convolutional layers and are incorporated in a single DCNN architecture that can then be trained end-to-end with standard back-propagation. Experiments on four large-scale video and image datasets show that the proposed DCNN improves concept annotation accuracy and outperforms the related state-of-the-art methods.**

***Index Terms*—Video/image concept annotation, deep learning, multi-task learning, structured outputs, multi-label learning, concept correlations, video analysis.**

I. Introduction

Video/image concept annotation, also known as classification [1], is the task of annotating video fragments (e.g., keyframes, video shots) or images with semantic labels referred to as concepts, e.g., object-related concepts (*table, chair*), general terms (*sky*, *sun*), or action-related concepts (*running*). The dominant approach for doing this is training DCNNs in whose architectures the concepts share features up to the very last layer, and then branch off to T different classification branches (using typically one layer), where T is the number of concepts [2]. However, in this way, the implicit feature-level relations between concepts, e.g. the way in which concepts such as a *car* and *motorcycle* share lower-level features modeling things like their wheels, are not directly considered. Also, in such architectures, the relations or interdependencies of the concepts at a semantic level, i.e. the fact that two specific concepts may often appear together or, inversely, the presence of the one may exclude the other, are also not directly taken into consideration. While some methods have been proposed for exploiting in a more elaborate way one of these two different types of concept relations, there is no single method that jointly exploits visual- and semantic-level concept relations in a unified DCNN architecture.

Other related problems that also take advantage of concept relations, include object detection [3], pedestrian detection [4], and attribute and relationship detection [5]. However, these problems are usually addressed using different, problem-specific techniques, and for this reason they are out of the scope of this work.

In this work we propose a DCNN architecture [1] that captures therefore both implicit and explicit concept relations, i.e., both visual-level and semantic-level concept relations. First, implicit concept relations are modeled in a DCNN architecture that learns T concept-specific feature vectors that are themselves linear combinations of $k < T$ latent concept feature vectors. In this way, in the shared representations (i.e., the latent concepts feature vectors), higher-level concepts may share visual features - for example, concepts such as *car*, *motorcycle*, and *airplane* may share features encoding the *wheels* in their depiction [6]. This bears similarities to multi-task learning (MTL) schemes, like GO-MTL [7] and the two-sided network proposed in [8] that factorize the 2D weight matrix that encodes concept specific features. However, in contrast to GO-MTL [7], in our work the factorization is achieved in two standard convolutional network layers, and in contrast to [8], our network does not only verify whether a certain concept that is given as input to the one side of the network is present in the video/image that is given as input to the other side. Instead, it provides scores for all concepts in the output, similar to classical multi-label DCNNs. Second, explicit concept relations are introduced by a new cost term, implemented using a set of standard CNN layers that penalize differences between the matrix encoding the correlations among the ground-truth labels of the concepts, and the correlations between the concept label predictions of our network. In this way, we introduce constraints on the structure

This work was supported by the EU's Horizon 2020 research and innovation programme under grant agreement H2020-687786 InVID, and by Nvidia corporation with the donation of a TitanX GPU.

F. Markatopoulou is with the Information Technologies Institute (ITI), Centre for Research and Technology Hellas (CERTH), Thermi 57001, Greece, and with the Queen Mary University of London, Mile end Campus, UK, E14NS. E-mail: markatopoulou@iti.gr,

V. Mezaris is with the Information Technologies Institute (ITI), Centre for Research and Technology Hellas (CERTH), Thermi 57001, Greece. E-mail: bmezaris@iti.gr,

I. Patras is with the Queen Mary University of London, Mile end Campus, UK, E14NS. E-mail: i.patras@qmul.ac.uk.

[1] Source code available at: https://github.com/markatopoulou/fvmtl-ccelc

of the output space by utilizing the label correlation matrix - this will explicitly capture, for example, the fact that *daytime* and *nighttime* are negatively correlated concepts.

The main contributions of this work are:

1) We propose a new approach, named FV-MTL (Shared Latent Feature Vectors using Multi-task Learning), that learns shared feature vectors corresponding to latent concepts, and expresses the concept-specific feature vectors as their sparse linear combination. This corresponds to a factorization of the weight matrix and is implemented using a set of standard CNN layers.
2) We propose a new cost function, named CCE-LC (Cost Sigmoid Cross-entropy with Label Constraint), which exploits a form of semantic relations, namely correlations between pairs of concepts, in order to predict structured outputs. This is again implemented using a set of standard CNN layers.
3) We incorporate both of the above in a single DCNN architecture that is trained end-to-end to solve the video/image concept annotation problem. We evaluate the trained DCNN both as a standalone classifier, where the direct output of the complete network is evaluated, and as feature generator, where SVM classifiers are trained on DCNN-generated features. In both cases we obtain state of the art results in publicly available datasets and show the benefits of each of the proposed contributions both alone and in tandem.

The remainder of the paper is organized as follows: Section II reviews related work, focusing on MTL and structured output prediction. Section III presents the proposed FV-MTL approach, the proposed CCE-LC cost function, and also the way that FV-MTL can be used jointly with the CCE-LC cost in a unified DCNN architecture. Section IV reports our experiments, results, and comparisons, and finally, Section V summarizes our main conclusions and highlights future work.

II. Related work

Image or video concept annotation is a challenging multi-label classification problem that in recent years is typically addressed using DCNN models that choose a specific DCNN architecture [9], [10] and put a multi-label cost function on the top of it [11]–[13]. As is the case in other multi-label problems, there exist relations between the different concepts, and several methods attempt to utilise/model them so as to improve the performance or reduce the complexity of classification models that treat each concept independently. These methods can be roughly divided in two main categories. In the first category, methods that fall under the framework of multi-task learning (MTL), attempt to learn representations or classification models that, at some level, are shared between the different concepts (tasks) [7], [8], [14]–[24]. In the second category, methods that fall under the framework of structured-output prediction attempt to learn models that make multi-dimensional predictions that respect the structure of the output space using either label constraints or post-processing techniques [8], [25]–[30], [30]–[44]. Label constraints refer to regularizations that are imposed into the learning system in order to exploit label relations (e.g., correlations) [8], [30], [31], [42]–[45]. Post-processing techniques refer to re-calculating the concept prediction results using either meta-learning classifiers or other re-weighting schemes [25]–[29]. In what follows, we first review works in those two broad categories and then highlight their relation and differences with the proposed method.

A. Multi-task Learning

Multi-task Learning (MTL) refers to jointly learning classifiers for many tasks by sharing knowledge across them so as to improve their accuracy, instead of learning individual models for each task. Video/image concept annotation can be treated as a MTL problem, where each task is about recognizing one concept. MTL methods can be divided into two broad categories: i) Shallow MTL methods that focus on shallow linear models and typically require pre-computed features as input, for example local descriptors or DCNN-based pre-computed features and ii) MTL methods that are an integral part of deep network architectures.

The MTL methods belonging to the first category extend typical linear models (e.g., Support vector machines (SVMs)) in order to incorporate task relatedness, i.e., the type of knowledge that should be shared. In a single-task learning (STL) concept annotation scenario, a supervised classifier is trained per concept on positive/negative keyframes/images of this concept. If the classifier is linear (e.g., SVM) its goal is to minimize the empirical cost: $\min_{(\boldsymbol{w}_j)} \mathcal{L}(\boldsymbol{w}_j) + \Theta(\boldsymbol{w}_j)$, where $\boldsymbol{w}_j \in \mathbb{R}^{d_1 \times 1}$ is the task parameter vector to be estimated from the training samples, $\mathcal{L}(\boldsymbol{w}_j)$ is the empirical cost on the training set, $\Theta(\boldsymbol{w}_j)$ is a regularization term and d_1 is the dimensionality of the input feature representation. MTL methods learn the parameters of all of the tasks together at the same time. As a result, assuming T tasks, all the task parameter vectors $\boldsymbol{w}_j$ for $j = 1...T$ are concatenated in a single parameter matrix $\boldsymbol{W} \in \mathbb{R}^{d_1 \times T}$ and the classifier's goal is to minimize the empirical cost: $\min_{(\boldsymbol{W})} \mathcal{L}(\boldsymbol{W}) + \Theta(\boldsymbol{W})$, where $\Theta(\boldsymbol{W})$ now encodes task relatedness. The main difference between MTL methods is the way they define task relatedness. Some methods identify shared feature representations between different tasks and use regularization over $\boldsymbol{W}$ to model task relatedness [14]–[16]. Others identify a shared subspace over the task parameter vectors [17]–[19]. The methods above make the strong assumption that all tasks are related; some newer methods consider the fact that some tasks may be unrelated. For example, the clustered MTL algorithm (CMTL) [20] uses a clustering approach to assign to the same cluster parameters of tasks that lie nearby in terms of their L2 distance. Adaptive MTL (AMTL) [21] decomposes the task parameters into a low-rank structure that captures task relations, and a group-sparse structure that detects outlier tasks. The GO-MTL (Grouping and Overlap in Multi-Task Learning) algorithm [7] and the online version of it [22] use a matrix factorization method, e.g., $\boldsymbol{w}_j = \boldsymbol{V}\boldsymbol{s}_j^\top$, that allows two tasks from different groups to overlap by having one or more bases in common. $\boldsymbol{V}$ corresponds to the parameter vectors of k latent tasks, while $\boldsymbol{s}_j \in \mathbb{R}^{1 \times k}$ is a task-specific weight vector that contains the coefficients of the linear combination of the latent tasks.

With respect to the second category of MTL methods, for start, DCNNs themselves are MTL models that consist of many layers of feature extractors, with the bottom layers learning more generic features that are shared across all of the tasks and the top-most layers being more concept-specific [46]. Typical DCNN architectures follow a *hard* feature/parameter sharing, i.e., each task uses exactly the same weight matrix for the corresponding layer; and similarly a *hard* feature/parameter separation, i.e., the last layer (a.k.a. the classification layer) takes as input the output of the second-last layer and translates it into a set of concept annotation scores learning weight matrices independently for each task [9], [10], [47]. However, more elaborate MTL methods that introduce *soft* feature/parameter sharing, i.e., adjusting how much information and across which tasks should be shared, have been presented. Such methods mainly focus on reformulating existing shallow linear MTL methods in order to be incorporated in DCNNs. For example, [24] proposes a two-sided neural network that unifies several shallow linear MTL methods that use a predictor matrix factorization approach, e.g., $\boldsymbol{w}_j = \boldsymbol{V}\boldsymbol{s}_j^\top$ [7]. MTL in deep learning architectures has also been proposed for facial landmark detection [23] and human pose estimation [48]. In [23] the single task of facial landmark detection is optimized with the assistance of an arbitrary number of related tasks. This is a special case of the conventional MTL that typically aims to maximize the performance of all tasks. In [48], the task of human detection is learned jointly with the task of body locations estimation, which results in improved human pose estimation. In [8] the two-sided neural-network of [24] is modified and extended, for transferring a network that has been originally trained on a source image dataset for concept annotation, to a target video dataset and a corresponding new set of target concepts.

Sometimes the terms MTL and multi-domain learning (MDL) are used interchangeably. However, the problems that each of them aims to solve are different. In MDL shared knowledge is exploited across different domains for the same tasks. For example, in [49] a cross-media (text, image, audio etc.) MDL retrieval method is proposed, where across the domains, the same set of concepts need to be learned. Similarly, an asymmetric MDL approach for person re-identification is proposed in [12] and a multiple-scene surveillance video understanding approach in [50], where in these works different domains refer to different video capture conditions. Asymmetric learning, aka domain adaptation [51], [52], refers to the fact that the method of [12] utilizes information from different source domains in order to improve the performance on the target domain, without considering potential improvement to the source domains as well.

Transfer learning is another related problem that uses the knowledge captured in a source domain in order to learn a target domain without caring about the improvement in the source domain. When a small-sized dataset is available for training a DCNN then a transfer learning technique is followed, where a conventional DCNN, e.g. [10], is firstly trained on a large-scale dataset and then the classification layer is removed, the DCNN is extended by one or more fully-connected layers that are shared across all of the tasks, and a new classification layer is placed on the top of the last extension layer (having size equal to the number of concepts that will be learned in the target domain). Then, the extended network is fine-tuned in the target domain [2].

B. Structured Output Prediction

Structured output prediction refers to methods that exploit semantic relations that may exist between the concept labels, and has received a lot of attention in the deep learning and the broader machine learning field. In contrast to MTL that exploits the common structure that task parameters or low-level features may have across the different tasks, structured output prediction focuses on the semantic relations that exist at the outputs, e.g., concept correlations. Video/image concept annotation is a multi-label learning problem, where given a set of concept labels, each keyframe/image is often associated with more than one labels. In most concept annotation datasets, ground-truth annotation is provided without any accompanying structure information concerning the concept labels; however, in many cases the concept labels are statistically related. For example, in the TRECVID-SIN video annotation dataset [53], which is one of the datasets used in this study, there are several groups of mutually exclusive labels, such as *indoor-outdoor* or *nighttime-sun*. The dataset also includes several positive correlations, such as *car-vehicle* and *dog-animal*. The automated learning of such relationships can incorporate useful knowledge into the model, improving the accuracy of the DCNN. In order to do so, many structured output prediction methods impose some label structural constraints either explicitly, i.e., using predefined rules that are known for the training dataset, or implicitly, i.e., the model is forced to discover existing label relations and considers them as label constraints. Existing methods can again be divided in those that take as input any pre-computed features and those that are tightly integrated with deep learning architectures.

With respect to the the first category, i.e., methods that take as input pre-computed features, two main sub-categories have appeared in the literature: a) Stacking-based approaches that collect the concept annotation scores produced either by a baseline set of concept detectors (e.g., SVMs) or by a DCNN when used as a standalone classifier, and introduce a second learning step in order to refine them, and b) Inner-learning approaches that follow a single-step learning process, which jointly considers extracted features and semantic relations. Stacking approaches aim to detect relations across concepts in the last layer of the stack. In [25] concept annotation scores are obtained from individual concept detectors in the first layer, in order to create a *model vector* for each shot. These vectors form a meta-level training set, which is used to train a second layer of independently trained concept detectors. In [26], a graph-based method is proposed that uses the ground-truth annotation to build decision trees that describe the relations across concepts, separately for each concept, and refines the initial scores by approximating these graphs. Using external knowledge of label relations, Deng et al. [27] proposed a representation, the HEX graph, to express and enforce exclusion, inclusion and overlap relations between labels. This

model was further extended for "soft" label relations using the Ising model by Ding et al. [28]. A different approach that outperforms the above was proposed by [29]. There the authors use model vectors to train multi-label classification algorithms that explicitly exploit label relations, instead of learning a second round of independent concept detectors or graph-models. All the above-mentioned approaches implicitly capture label relations from the meta-level training set of model vectors; as a result, they rely on starting with good concept probability estimates in the model vectors, otherwise the errors are propagated to the next layers.

Inner-learning approaches, on the other hand, use the extracted features and exploit concept relations in a single learning step. For example, the authors of [30] and [31] propose methods that simultaneously learn the relation between visual features and concepts and also the correlations between concepts. In [45] a joint learning-to-rank approach is proposed, which naturally combines the benefits of training a DCNN with a structural SVM model that is used for concept ranking. In [54] the temporal consistency of concept labels across neighboring video shots is exploited. While in [55] an AdaBoost classifier is trained by carefully selecting positive and negative correlated concepts that will be used per iteration. However, inner-learning approaches suffer of computational complexity. For example, [30] has complexity at least quadratic to the number of concepts, making it inapplicable to real video/image concept annotation problems, where the number of concepts is large (e.g. hundreds or thousands). The LMGE algorithm (Label correlation Mining with relaxed Graph Embedding) [31], is a faster approach with linear complexity with respect to the number of concepts; however, the complexity of the training process is about n^3, where n refers to the number of training samples. Many more methods can be found in this category for multi-label image annotation, which explore such label relations to improve the classification accuracy at the expense of increased computational complexity compared to the stacking-based ones, e.g., [32]–[41].

With respect to the second category, i.e., methods that are an integral part of DCNN architectures, structured output prediction techniques have been proposed for application mainly to the pixel-wise semantic segmentation problem. The most popular approach is to combine a DCNN with a graphical model [42], [43], [44]. For example, in [42] a Markov random field is jointly used on top of a DCNN architecture in order to incorporate the spatial relations and label correlations of the assigned labels on the pixels of an image. Similarly, in [44] the conditional random field model is formulated as a recurrent neural network (RNN) and plugged in as part of a DCNN. Structured output prediction for DCNNs has also been proposed for other visual recognition problems, such as group activity recognition [43]. All of these methods employ probabilistic inference to correct the marginal probability of each label. In contrast to the above methods that use graphical models, in [8] an auxiliary cost function that approximates the correlations between the concept labels gets added to the total network's cost. This auxiliary cost function takes the form of a constraint over the task-specific parameters of the network and is shown to improve its accuracy.

C. Related Work and the Proposed Approach

Our method has some similarities with the MTL methods [7] and [8]. However, in contrast to [7] that uses pre-computed features and a shallow linear model to factorize the 2D weight matrix that encodes concept-specific features, our approach learns shared representations as an integral part of a DCNN architecture implementing the factorization in two standard CNN layers. Furthermore, in contrast to [8] that uses a two-sided CNN taking as input in the one side a keyframe/image, and in the other side a semantic descriptor that verifies whether a certain concept is present in the keyframe/image, our method requires only the raw keyframe/image and no additional information regarding the task that should be learned/predicted. Both [7] and [8] can be optimized for only binary or multi-class classification cost functions (e.g., logistic loss), thus ignoring the multi-label nature of the concept-based video/image annotation problem. In contrast, our method works for any multi-label classification cost (e.g., cross-entropy). Our method is also most closely related to [8] and [22], that jointly consider MTL and structured outputs, but differs from them as follows: In [22] pre-computed features are used, specifically, a shallow linear MTL method is proposed that is instantiated with a new cost function that exploits concept correlations and takes as input pre-computed DCNN-based features. In contrast, we propose a single DCNN architecture, trained end-to-end, that incorporates both MTL and structured-output prediction, both of which are implemented using standard convolutional layers, resulting in a more effective classifier. In [8] an auxiliary label-based cost function is proposed that forces the network's output to fit the distribution of a single row of the concept correlation matrix. In contrast, our approach adds a concept correlation cost term to the network's main cost function that forces positively-correlated concepts to receive similar scores and negatively-correlated ones to receive dissimilar scores, again leading to better results.

III. Proposed Approach

A. Problem Formulation and Method Overview

We consider a set of concepts $C = \{c_1, c_2, ..., c_T\}$ and a multi-label training set $\mathcal{P} = \{(\boldsymbol{x}_i, \boldsymbol{y}_i) : \boldsymbol{x}_i \in \mathcal{X}, \boldsymbol{y}_i \in \{0,1\}^{T\times 1}, i = 1...N\}$, where $\boldsymbol{x}_i$ is a 3 channel keyframe/image, and $\boldsymbol{y}_i$ is its ground-truth annotation. A video/image concept annotation system learns T supervised learning tasks, one for each target concept c_j. More specifically, it learns a real-valued function $f : \mathcal{X} \rightarrow \mathcal{Y}$, where $\mathcal{Y} = [0,1]^{T\times N}$ could then be binarized (e.g., thresholded) in order to provide a hard classification result, if needed.

We propose a DCNN architecture that exploits both visual-level and semantic-level concept relations for video/image concept annotation by building on ideas from MTL and structured output prediction, respectively. In Fig. 1 (i) we illustrate a typical $(\Pi + 1)$-layer DCNN architecture, e.g., ResNet, that shares all the layers but the last one (steps (a),(b))) [9], [10]; in Fig. 1 (ii) we illustrate how the typical DCNN architecture of Fig. 1 (i) is extended by one FC extension layer, which was shown to outperform the typical DCNN architecture when used in transfer learning problems [2] (steps (c)-(e));

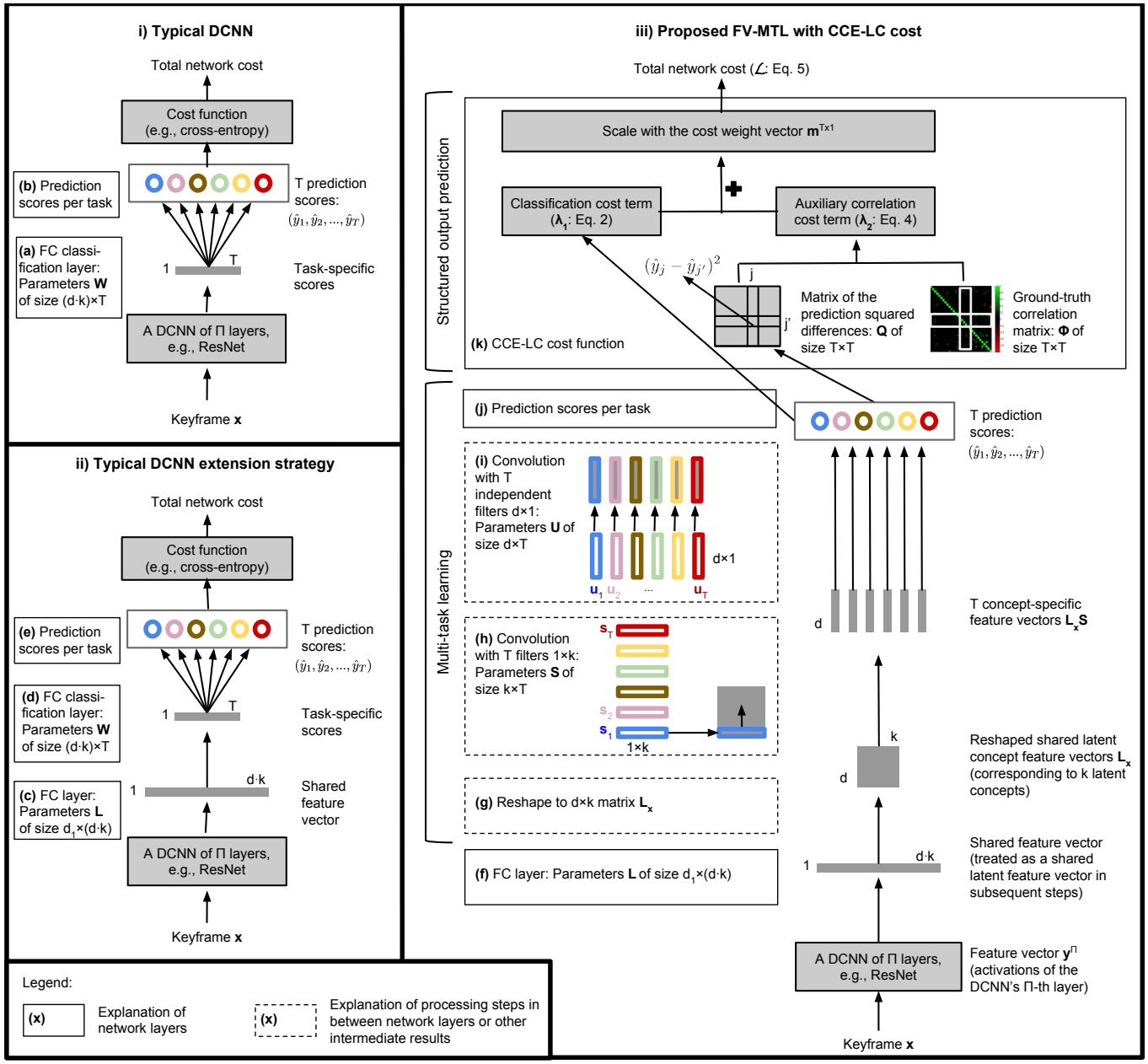

Fig. 1. Sub-figure (i) presents the typical DCNN architecture (e.g., ResNet [10]). Sharing all layers but the last one. Sub-figure (ii) presents the typical DCNN extension strategy proposed in [2]. A shared fully-connected layer, a.k.a the extension layer, and a concept-specific classification layer (a second fully-connected classification layer that maps a common feature representation to concept categories, independently for each concept), are placed on the top of a typical DCNN architecture (e.g., ResNet [10]). Sub-figure (iii) presents the proposed FV-MTL with CCE-LC cost function: FV-MTL is modeled as a stack of standard CNN layers, on the top of which the CCE-LC cost function is placed, which consist of two terms i) the cross-entropy cost term and ii) the auxiliary correlation cost term that integrates structural information. CCE-LC is also modeled as a stack of standard CNN layers.

and finally, in Fig. 1 (iii) we present the proposed DCNN architecture (steps (f)-(k)). In the next subsections we first introduce the new FV-MTL approach for learning implicit visual-level concept relations; this is done using the network layers as shown in Fig. 1 in steps (f) to (i). Second, we introduce the new CCE-LC cost function that learns explicit semantic-level concept relations, which is done in step (k). CCE-LC predicts structured outputs from concept correlations that we can acquire from a training dataset's ground-truth annotations. The source code of our method can be found at https://github.com/markatopoulou/fvmtl-ccelc.

B. Shared Latent Feature Vectors using Multi-task Learning (FV-MTL)

In our approach, similarly to GO-MTL [7], we assume that the parameter vectors of the tasks that present visual-level concept relations (i.e., defined in GO-MTL as belonging to the same group) lie in a low-dimensional subspace, thus sharing information; and, at the same time, dissimilar tasks (i.e., belonging to different groups) may also partially overlap by having one or more bases in common. Allowing the sharing also between dissimilar tasks is more natural than creating

TABLE I
DEFINITION OF THE SYMBOLS

Symbols	Definitions
$\boldsymbol{x}$	A keyframe/image
$\boldsymbol{y}$	A vector containing the ground-truth concept annotations for a keyframe/image
N	The number of training keyframes/images
c	A concept
T	The number of concepts, i.e., number of tasks
i	Keyframe/image index, i.e., $i = 1...N$
j	Concept/task index, i.e., $j = 1...T$
$\hat{\boldsymbol{y}}$	A vector containing the concept prediction scores for a keyframe/image
$\boldsymbol{L_x}$	Latent concept feature vectors of a keyframe/image
$\boldsymbol{S}$	Concept-specific weight matrix, each column corresponds to a task containing the coefficients of the linear combination with $\boldsymbol{L_x}$
$\boldsymbol{L_x S}$	Concept-specific feature vectors incorporating information from k latent concept representations
$\boldsymbol{U}$	Concept-specific parameter matrix for the final classification
k	The number of latent tasks
$\sigma(.)$	The sigmoid function
$\boldsymbol{\Phi}$	The concept correlation matrix calculated from the ground-truth annotated training set
$\boldsymbol{m}$	A cost vector utilized for data balancing
β	Regularization parameter
$\boldsymbol{z}$	Normalization factor vector

disjoint groups of task models. In order to do so, we learn T concept-specific feature vectors that are linear combinations of a small number of latent concept feature vectors that are themselves learned as well. Specifically, our approach uses a shared latent feature vector $\boldsymbol{L_x} \in \mathbb{R}^{d\times k}$ for all task models, where the columns of $\boldsymbol{L_x}$ correspond to d-dimensional feature representations of k latent tasks; and produces T different concept-specific feature vectors $\boldsymbol{L_x s}_j$, for $j = 1...T$, where each of them incorporates information from relevant latent tasks, with $\boldsymbol{s}_j \in \mathbb{R}^{k\times 1}$ being a task-specific weight vector that contains the coefficients of the linear combination. Each linear combination is assumed to be sparse in $\boldsymbol{L_x}$, i.e, $\boldsymbol{s}_j$'s are sparse vectors. In this way we assume that there exist a small number of latent basis tasks and each concept-specific feature vector is a linear combination of them. The overlap in the sparsity patterns of any two tasks, (i.e., how much overlap is observed between two different task-specific weight vectors $\boldsymbol{s}_j$ and $\boldsymbol{s}_{j'}$, where $j \neq j'$) controls the amount of sharing between them.

The above can be implemented in a DCNN architecture by using the network layers depicted in Fig. 1 in steps (f) to (i). Specifically, an input training-set keyframe is processed by a typical DCNN architecture (e.g., ResNet) and a fully-connected layer, to produce a shared representation of the keyframe across all of the tasks (Fig. 1: step (f); this is the same as step (c) in the typical DCNN extension architecture). Subsequently, the output of the fully-connected layer is reshaped to the matrix $\boldsymbol{L_x}$ (Fig. 1: step (g)). Consequently, the reshaped layer outputs k feature vectors that correspond to k latent concepts. Those representations are shared between the T concepts. The subsequent layers calculate T concept-specific feature vectors, where T is the number of the concepts we are interested in detecting. Each of those feature vectors is a combination of k latent concept feature vectors, with coefficients that are specific to the concept in question. This is implemented as a 1D convolutional layer on the k feature masks - in the case that the 1D convolutional layer implements a linear transform, i.e., we do not use a non-linear activation function, then these two layers implement a feature extraction scheme with a bilinear factorization of the weight matrix (Fig. 1 step (h)). Once T feature vectors are extracted, then an additional layer (Fig. 1: step (i)) transforms each of the T feature vectors into T concept annotation scores, one for each of the concepts that we are set to recognize (Fig. 1: step (j)). The above process leads to a *soft* feature sharing, because the latent concept feature vectors adjust how much information and across which tasks should be shared. By contrast, both the typical DCNN and the DCNN extension architecture of [2] output a single feature vector (Fig. 1: step (a) and (d), respectively) that is shared across all of the target concepts and it is subsequently *hard* translated into concept annotation scores independently for each concept (Fig. 1: step (b) and (e), respectively), as was also discussed in Section II.

Formally, the predicted score for the j-th task (concept) and the i-th datapoint (keyframe/image) is given by:

$$\hat{y}_{i,j} = \text{diag}(\boldsymbol{u}_j^\top (\boldsymbol{L}_{\boldsymbol{x}_i} \boldsymbol{s}_j)), \tag{1}$$

where $\boldsymbol{L}_{\boldsymbol{x}_i}$ is the output of the last fully-connected layer of the right part of Fig. 1 (see step (f)), after reshaping the calculated vector of dimension $1 \times (d \cdot k)$, in order to have a matrix of d rows and k columns (Fig. 1: step (g)). Specifically, $\boldsymbol{L}_{\boldsymbol{x}_i} = \text{reshape}(\alpha(\boldsymbol{L}' \boldsymbol{y}_i^{(\Pi)} + \boldsymbol{b}))$, where $\boldsymbol{L} \in \mathbb{R}^{d_1 \times d\cdot k}$ the parameters of the last fully-connected layer, $\boldsymbol{y}_i^{(\Pi)} \in \mathbb{R}^{d_1 \times 1}$ the output of the previous layer, and α the layer's activation functions, e.g. the ReLU. $G = \{g^{(\pi)}\}_{\pi=1}^{\Pi}$ is the set of network parameters for the first Π layers. $\boldsymbol{s}_j, \boldsymbol{u}_j$ are the j-th columns of the parameter matrices $\boldsymbol{S} \in \mathbb{R}^{k\times T}$ and $\boldsymbol{U} \in \mathbb{R}^{d\times T}$, respectively. Each $\boldsymbol{s}_j$ contains a task-specific weight vector of the coefficients of the linear combination with the shared latent feature vector $\boldsymbol{L}_{\boldsymbol{x}_i}$. This linear combination indicates for each concept which latent tasks describe it. Each $\boldsymbol{u}_j$ contains a concept-specific weight vector that transforms the concept-specific feature vectors $\boldsymbol{L}_{\boldsymbol{x}_i} \boldsymbol{S}$ to concept scores.

Similarly to other DCNN works, we optimize the sigmoid cross entropy between the predicted and the ground truth labels that is formed as:

$$\lambda_{1_{i,j}} = y_{i,j} log\sigma(\hat{y}_{i,j}) + (1 - y_{i,j}) log(1 - \sigma(\hat{y}_{i,j})), \tag{2}$$

where $\sigma(\cdot)$ refers to the sigmoid function $\sigma(x) = 1/(1 + exp(-x))$. That is, we optimize Eq. 2 with respect to the parameters of the network. This is the cost of the *classification cost term* branch in Fig. 1 and differs from the GO-MTL cost function [7] in the following ways:

First, while GO-MTL aims to approximate the parameter vector of the j-th observed task $\boldsymbol{w_j}$ by a linear combination of a subset of latent tasks $\boldsymbol{w_j} = \boldsymbol{V s_j}$, where $\boldsymbol{V} \in \mathbb{R}^{d\times T}$ is a shared knowledge basis, our goal is given a keyframe/image i, to learn a new set of concept-specific feature vectors $\boldsymbol{L}_{\boldsymbol{x}_i} \boldsymbol{s}_j$, one per task, that leverage shared properties with all the other tasks. Our assumptions are similar, and we also use a predictor matrix factorization approach $\boldsymbol{L}_{\boldsymbol{x}_i} \boldsymbol{S}$, however, in a different

way: In the proposed approach, given an input keyframe/image our method transforms it into T different concept-specific feature vectors that leverage information from a set of latent concept feature vectors using a bilinear factorization of the weight matrix, as described above. Subsequently, parameter matrix $\boldsymbol{U}$ is used in order to transform these concept-specific representations to concept scores, i.e., $\boldsymbol{U}^\top(\boldsymbol{L}_{\boldsymbol{x}_i}\boldsymbol{S})$. Differently, GO-MTL factorizes the 2D weight matrix that encodes concept-specific features and directly transforms the image/keyframe into concept scores.

Second, GO-MTL [7] uses iterative optimization and shallow linear models to learn the parameters. For example, in each iteration of the GO-MTL [7] method all parameters except for one are kept fixed and the function is optimized towards the non-fixed parameter. In our case a complete DCNN architecture is used, which makes it easy to calculate error differentials per layer w.r.t. its inputs, in order to back-propagate them to previous layers.

Third, the GO-MTL cost function can be optimized with respect either to regression loss (e.g., squared loss) or binary/multi-class classification loss (e.g., logistic loss), thus ignoring the multi-label nature of the problem. In contrast, our method works for any multi-label classification cost (e.g., the sigmoid cross entropy loss, presented above). It should be noted here that we use the sigmoid function on each activation separately and as a result the different outputs do not compete with each other (i.e., their sum does not equal to 1).

To make clear the difference of the proposed architecture from the typical and extension DCNN architectures (Fig. 1 (i) and (ii), respectively) we set $\alpha(\boldsymbol{L}\boldsymbol{y}_i^{(II)}+\boldsymbol{b})=\boldsymbol{\Delta}$ and rewrite Eq. 1 as: $\hat{y}_{i,j}=\text{diag}(\boldsymbol{u}_j^\top(\text{reshape}(\boldsymbol{\Delta})\boldsymbol{s}_j))$.

Similarly, the predicted score for the j-th task and i-th datapoint with respect to the typical and extension DCNN architecture is given by: $\hat{y}_{i,j}^T=\boldsymbol{w}_j^{\top T}(\alpha(\boldsymbol{y}_i^{(II)}+\boldsymbol{b}))$, and $\hat{y}_{i,j}^E=\boldsymbol{w}_j^{\top E}\boldsymbol{\Delta}$, respectively. The task-specific weight vector $\boldsymbol{s}_j$ used in our method contains the coefficients of the latent task feature vectors that will be combined with respect to concept j. This is exactly the way that our method achieves a *soft* feature sharing separately for each concept, i.e., by letting similar concepts to be described by the same latent task feature vectors according to $\boldsymbol{s}_j$. In contrast, the other two architectures of Fig. 1 do not use this linear combination of latent concept feature vectors but let the second-last layer, a single feature vector, to be shared across all of the concepts, thus, a *hard* translation into concept scores is performed independently for each concept.

C. Label Constraints for Structured Output Prediction

Cross-entropy cost is not adequate for capturing semantic concept relations. In this section we present an additional cost term that constitutes an effective way to integrate structural information. By structural information we refer to the inherently available concept correlations in a given ground-truth annotated collection of training videos/images. In order to consider this information we firstly calculate the correlation matrix $\boldsymbol{\Phi}\in[-1,1]^{T\times T}$ from the ground-truth annotated data of the training set. Each position of this matrix corresponds to the ϕ-correlation coefficient between two concepts c_j, $c_{j'}$ calculated as:

$$\phi_{j',j}=\frac{AD-BC}{(A+B)(C+D)(A+C)(B+D)}, \tag{3}$$

where $\phi_{j',j}$ refers to j'-element of the j-th row of the correlation matrix $\boldsymbol{\Phi}$ that contains the correlation between concepts $c_{j'}$ and c_j. $A=p(c_j\wedge c_{j'}|\boldsymbol{y}_i, i=1...N)$, $B=p(c_j\wedge\neg c_{j'}|\boldsymbol{y}_i, i=1...N)$, $C=p(\neg c_j\wedge c_{j'}|\boldsymbol{y}_i, i=1...N)$, $D=p(\neg c_j\wedge\neg c_{j'}|\boldsymbol{y}_i, i=1...N)$, where $p(a|b)$ refers to the probability of a given b. The logical operator $\wedge$ expresses conjunction, e.g., $c_j\wedge c_{j'}$, means that both c_j and c_j appear on the image/keyframe, according to its ground-truth annotations; and $\neg$ expresses negation, e.g., $c_j\wedge\neg c_{j'}$, means that $c_{j'}$ does not appear on the image/keyframe.

The proposed auxiliary concept correlation cost term that uses the correlation matrix $\boldsymbol{\Phi}$ is formed as follows:

$$\lambda_{2_{i,j}}=\frac{1}{T\text{-}1}\sum_{\substack{j'=1,\\ j'\neq j}}^{T}\begin{cases}\phi_{j',j}\left\|\sigma(\hat{y}_{i,j})-\sigma(\hat{y}_{i,j'})\right\|^2, \text{if } \phi_{j',j}\geq 0\\ (-\phi_{j',j})\left\|\sigma(\hat{y}_{i,j})+\sigma(\hat{y}_{i,j'})\right\|^2, \text{otherwise}\end{cases} \tag{4}$$

This term works as a label-based constraint and its role is to add a penalty to concepts that are positively correlated but were assigned with different concept annotation scores. Similarly, it adds a penalty to concepts that are negative-correlated but were not assigned with opposite annotation scores. Contrarily, it does not add a penalty to non-correlated concepts.

We can implement the $\lambda_{2_{i,j}}$ correlation term (Eq. 4) using a set of standard CNN layers, as presented on the top of the right part of Fig. 1. One matrix layer encodes the correlations between the ground-truth labels of the concepts (denoted as $\boldsymbol{\Phi}$), and the other matrix layer contains the correlations between the concept label predictions of our network in the form of squared differences (denoted as $\boldsymbol{Q}\in\mathbb{R}^{T\times T}$, i.e., the matrix $\mathbf{Q}$ contains the differences of activations from the previous layer). Specifically, the matrix $\mathbf{Q}$ gets multiplied, by element-wise multiplication, with the correlation matrix $\boldsymbol{\Phi}$, i.e., $\mathbf{Q}\circ\boldsymbol{\Phi}$. All the rows in the resulting $T\times T$ matrix are added, which leads to a single row vector.

D. FV-MTL with Cost Sigmoid Cross-entropy with Label Constraint (FV-MTL with CCE-LC)

The two cost terms presented in Sections III-B and III-C, i.e., Eq. 2 and Eq. 4, respectively, can be added in a single cost function that forms our total FV-MTL with CCE-LC network's cost as follows:

$$\mathcal{L}=\sum_{i=1}^{N}\frac{1}{T}\sum_{j=1}^{T}\frac{m_{i,j}}{z_j}\left(\lambda_{1_{i,j}}+\beta\lambda_{2_{i,j}}\right) \tag{5}$$

where parameter β controls the importance of concept correlation term.

In the above cost function we introduce the vector $\boldsymbol{m}_i\in\mathbb{R}^{T\times 1}$ that was originally proposed by [13] to address the problem of class imbalance. Class imbalance is a common

problem in concept annotation, where for most datasets the distribution between negative to positive examples per concept is highly imbalanced, with the former outnumbering the latter in most cases. This results in bias of the classifier towards the class (positive or negative) that contains the largest number of samples. Consequently, we introduce the cost vector $\boldsymbol{m}_i$ in our cost function in order to balance the number of positive to negative examples per concept. Let us denote by p_j the number of the positive examples and n_j the number of negative examples for the concept c_j. Then, the ratio r_j of the negative to positive examples is computed as:

$$r_j = \begin{cases} \frac{n_j}{p_j}, & \text{if } n_j \; and \; p_j \neq 0 \\ 1, & \text{otherwise} \end{cases} \quad (6)$$

We create a weight vector $\boldsymbol{m}_i = [m_{i,1}, ..., m_{i,T}]$, for each training example i.e., for $i = 1...N$. Where $m_{i,j} = 1$ if $y_{i,j} = 0$, $m_{i,j} = 0$ if $y_{i,j}$ is unlabeled and $m_{i,j} = r_j$ if $y_{i,j} = 1$, where r_j is given by Eq. 6, and is different for each concept j. This weight vector is multiplied element-wise with the cost function. By doing so we adjust the misclassification cost of positive examples so as to prevent the biasing of the network towards the negative class when only a few positive examples are available. Furthermore, the normalization factor $\boldsymbol{z} \in \mathbb{R}^{T \times 1}$ that is introduced in Eq. 5 is calculated as: $\boldsymbol{z} = \sum_{i=1}^{N} \boldsymbol{m}_i$, where each position of this vector, i.e., z_j, denotes the sum of the weights for concept c_j.

In our overall network architecture, an additional layer is used in order to implement the complete FV-MTL with CCE-LC cost function, adding the two cost terms (λ_1, λ_2) and scaling their sum by the $\boldsymbol{m}$ (Eq., 5). In this way, the complete DCNN architecture learns by considering both the actual ground-truth annotations and also the concept correlations that can be inferred from it (Fig. 1: step (k)). In contrast, a typical DCNN architecture simply incorporates knowledge learned from each individual ground-truth-annotated sample.

IV. Experimental Study

A. Datasets and Experimental Setup

Our experiments were performed on four large multi-label video/image classification datasets, namely the TRECVID-SIN 2013 [53], the PASCAL-VOC 2007 [1], the PASCAL-VOC 2012 [56], and the NUS-WIDE [57], presented in Table II. For assessing concept annotation performance, the indexing problem as defined in [53] was evaluated, i.e., given a concept, the goal was to retrieve the 2000 video shots (or images, depending on the dataset) that are mostly related with it.

The TRECVID-SIN 2013 [53] dataset consists of approximately 600 and 200 hours of internet archive videos for training and testing, respectively. The training set is partially annotated with 346 semantic concepts. The test set is evaluated on 38 concepts for which ground-truth annotations exist, i.e., a subset of the 346 concepts. The PASCAL-VOC 2007 [1] dataset consists of 5011 training and validation images and 4952 test images. The PASCAL-VOC 2012 [56] dataset consists of 22531 images divided into training, validation and test sets (5717, 5823 and 10991 images, respectively). We used the training set to train the various methods of our study, and evaluated them on the validation set. We did not use the original test set because ground-truth annotations are not publicly available for it (the evaluation of a method on the test set is possible only through the evaluation server provided by the PASCAL-VOC competition, submissions to which are restricted to two per week). Both for the PASCAL-VOC 2007 and 2012 the images are annotated with 20 object classes. The NUS-WIDE [57] dataset consists of 269648 Flickr images that have been annotated with 81 semantic concepts. We used a subset of 161789 images for training and the rest of them for testing. Since the available ground-truth annotations for each of the four datasets are not adequate in number in order to train a deep network from scratch without over-fitting its parameters, similarly to other studies [2], we used transfer learning. I.e., we used as a starting point the ResNet-50 network [10], which was originally trained on 1000 ImageNet categories [58], and fine-tuned its parameters towards each of these four datasets.

In order to evaluate the methods' performance in the PASCAL-VOC 2007, 2012 and NUS-WIDE datasets we used the mean average precision (MAP) measure, while, the mean extended inferred average precision (MXinfAP) [59], which is an approximation of MAP, was used for the TRECVID-SIN dataset. MXinfAP is suitable for the partial ground-truth that accompanies the latter dataset.

B. Implementation Details

For the rest of this section, when DCNN training takes place we did it by using the pre-trained ResNet-50 ImageNet network [10] (removing the last classification layer) and fine-tuning it on the target concept annotations. The network's learning rate and momentum was set to 10^{-5} and 0.9, respectively, whereas the mini-batch size was restricted by our hardware resources and set to 32. Multi-label stratification was used in order to ensure similar distribution of positive examples per class on each batch. Stochastic gradient descent (SGD) was used as the network's optimization function. All networks were trained and implemented in Caffe [60]. Regarding the proposed method, the new layers learning rate and momentum were set to 0.1 and $5 \cdot 10^{-4}$, respectively, and β was set to 10. This value for β was chosen based on preliminary experiments on the TRECVID SIN dataset (Fig., 2) that show that this is an appropriate value, and also that the proposed approach is not sensitive to the value of β. The diagonal of the $\boldsymbol{\Phi}$ correlation matrix was set to zero. The model parameter values with respect to the compared methods were either selected experimentally or following the typical heuristics and strategies proposed in the corresponding works. We conducted our experiments on two NVIDIA TitanX GPUs.

Each trained DCNN was used in two different ways to annotate new images/keyframes with semantic concepts: a) As a standalone classifier, where each test image/keyframe was forward-propagated by the network and the network's output was used as the final class distribution that was assigned to the image/keyframe. b) As a feature generator, where the training set was once again forward-propagated by the network, and

TABLE II
DATASETS AND THEIR STATISTICS. LABEL CARDINALITY (I.E., THE AVERAGE NUMBER OF CONCEPTS PRESENTED PER IMAGE/VIDEO SHOT), CONCEPT CARDINALITY (I.E., THE AVERAGE NUMBER OF POSITIVE IMAGES/VIDEO SHOTS PER CONCEPT), AND MISSING LABELS (I.E., THE AVERAGE NUMBER OF NON-ANNOTATED LABELS PER IMAGE/VIDEO SHOT) HAVE BEEN CALCULATED ON THE TRAINING SET FOR EACH DATASET.

Dataset	Training Instances	Testing Instances	Training set Concepts	Test set Concepts	Concept Cardinality	Label Cardinality	Missing Labels
TRECVID-SIN	239495	112677	346	38	3206.3	2.2	294.6
PASCAL-VOC2012	5717	5823	20	20	416.6	1.5	0.0
PASCAL-VOC2007	5011	4952	20	20	379.2	1.4	0.0
NUS-WIDE	161789	107859	81	81	3066.1	1.9	0.0

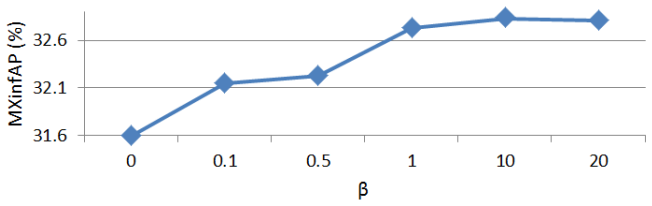

Fig. 2. MXinfAP (%) for different values of β (Eq 5) for the proposed FV-MTL with CCE-LC cost.

the values calculated in the last layer of the network were used as feature vectors to subsequently train one Support Vector Machine (SVM) per concept. Then, each test image was firstly forward-propagated by the DCNN to extract the features and subsequently was served as input to the trained SVM classifiers.

C. Preliminary Experiments - Design Choices

In Table III we examine the best way of using the proposed FV-MTL with CCE-LC cost by comparing different parameters and intermediate versions of them. We performed this set of experiments on the TRECVID-SIN dataset using as a starting point the ResNet-50 network.

- As a baseline we used the extension strategy proposed in [2], i.e., the DCNN architecture illustrated in Fig. 1 (ii). The results are presented in Table III: (d)). The dimensionality of the extension layer (Fig. 1: step (c)) is indicated in Table III: (a). Sigmoid cross-entropy was used as the network's cost function.
- We compared the baseline approach with: i) The proposed CCE-LC cost when used on the top of the baseline DCNN architecture, replacing the sigmoid cross-entropy cost (Table III: (e)), i.e., the FV-MTL method was ignored. ii) The proposed FV-MTL with CCE-LC, where for the latter parameter β was set to 0, i.e., the concept correlation term λ_2 in Eq. 5 was ignored (Table III: (f)). iii) The complete proposed FV-MTL with CCE-LC cost for $\beta = 10$, i.e., both cost terms, λ_1 and λ_2, were considered (Table III: (g)). Each row of Table III corresponds to a different dimension of our FV-MTL first FC layer (shown in Fig. 1: step (f)).

Each of the above DCNN architectures was fine-tuned on the 346 TRECVID-SIN concepts using the TRECVID development dataset [53]. Using these results, we assess i) how the number of the latent tasks k and feature dimensionality d affect FV-MTL (Table III: (a)-(c)), ii) the usefulness of exploiting semantic-level (explicit) concept relations using the CCE-LC cost instead of the typical sigmoid cross-entropy cost, iii) the usefulness of exploiting visual-level (implicit) concept relations using the proposed FV-MTL with CCE-LC when ignoring the concept correlation term λ_2 in Eq. 5 (Table III: (f)), and iv) the usefulness of jointly exploiting visual-level and semantic-level concept relations by adopting MTL and structured output prediction using the proposed FV-MTL with CCE-LC cost when both cost terms (λ_1, λ_2 in Eq. 5) are considered (Table III: (g)). It should be noted that our proposed FV-MTL with CCE-LC cost is most beneficial when used on datasets with non-exclusive labels (e.g., TRECVID SIN, PASCAL-VOC, NUS-WIDE) where CCE-LC can exploit and capture concept correlations across the labels. Such concept correlations are missing in single-label classification datasets such as ImageNet.

The choice of parameter k, which determines the number of latent tasks, is important because it determines the amount of sharing between the tasks. If k is very high, the tasks are not forced to share information with each other. On the other hand, if k is very low, the latent space may shrink too much. In Table III we compare different values for this parameter in order to see how it affects the proposed FV-MTL method (Table III: (f),(g)). We observe that the larger the value of k the better the accuracy of the FV-MTL approach. According to the rest of the results, we observe that structured output prediction using the proposed CCE-LC cost (Table III:(e)), and MTL using the proposed FV-MTL approach (Table III: (f)) are two different ways to improve concept annotation accuracy, as according to Table III the two methods always outperform the baseline (Table III:(d)). Jointly using MTL and structured output prediction, in a DCNN architecture (Table III: (g)) almost always outperforms all the other methods, reaching the best result of 32.83% when parameter k equals to 32 and parameter d equals to 64, i.e., the columns of $\boldsymbol{L_x}$ equal to 2048. One exception is seen in the first row of Table III, where we observe a small decrease in performance of $\beta = 10$ compared to $\beta = 0$. This is due to the low number of feature dimensions and latent tasks, which are not sufficient for the CCE-LC term to capture well the correlation information.

D. Main Findings - Comparisons With Related Methods

Table IV compares the proposed complete FV-MTL with CCE-LC (for $\beta = 10$) with other related methods on the three datasets. In addition, we evaluate the two intermediate versions of our complete DCNN architecture that were also evaluated in Table III. I.e., a) Extension strategy [2] for DCNNs with

TABLE III

PERFORMANCE (MXINFAP, %) FOR DIFFERENT DIMENSIONS OF THE COLUMNS OF THE $\boldsymbol{L_x}$ MATRIX (FIG. 1 STEP (E)) THAT WE USED IN THE EXPERIMENTS. FOR THE METHODS (D), (E) THAT DO NOT USE MTL, I.E., SIMPLY USE ONE EXTENSION LAYER AND ONE CLASSIFICATION LAYER ON THE TOP OF IT, COL. C INDICATES THE DIMENSIONALITY OF THIS EXTENSION LAYER. EXTENSION STRATEGY [2] WITH SIGMOID CROSS-ENTROPY COST SERVES AS OUR BASELINE.

$\boldsymbol{L_x}$ **#columns** $d \times k$	**Latent tasks** k	**Feature dimension** d	**DCNN (extension strategy [2]) with STL sigmoid cross-entropy**	**Proposed CCE-LC cost**	**Proposed FV-MTL with CCE-LC** ($\beta = 0$)	**Proposed FV-MTL with CCE-LC** ($\beta = 10$)
(a)	(b)	(c)	(d)	(e)	(f)	(g)
128	4	32	23.18	28.76	30.01	29.43
256	4	64	26.91	30.84	30.50	31.38
512	8	64	28.44	30.95	30.37	31.92
1024	16	64	29.76	31.21	30.25	32.1
2048	32	64	30.95	32.44	31.60	**32.83**
4096	32	128	31.06	31.94	31.65	32.02
4096	64	64	31.06	31.94	31.71	32.07

TABLE IV

MXINFAP (%) FOR 38 TRECVID-SIN AND MAP (%) FOR 20 PASCAL-VOC2007, 20 PASCAL-VOC2012 AND 81 NUS-WIDE CONCEPTS, RESPECTIVELY, FOR DIFFERENT STL, MTL, STRUCTURED OUTPUT AND JOINT MTL AND STRUCTURED PREDICTION METHODS USING THE IMAGENET RESNET-50 AS THE BASE NETWORK.

Category	**Method**	**TRECVID-SIN**		**PASCAL-VOC2007**		**PASCAL-VOC2012**		**NUS-WIDE**	
		(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)
		direct	last layer	direct	last layer	direct	last layer	direct	last layer
i) Baseline (without fine-tuning)	ResNet-50 [10] as feature generator	29.21	29.78	83.90	83.76	82.98	83.04	51.30	56.20
ii) Typical DCNN fine-tuning	ResNet-50 [10]	27.35	28.66	76.38	83.06	81.20	82.15	51.17	56.32
iii) DCNNs (extension strategy [2]) with STL cost functions	Hinge-loss	29.08	30.06	78.32	79.23	86.6	87.23	52.80	57.49
	Sigmoid cross-entropy	31.06	32.2	80.74	84.97	86.94	86.80	53.94	57.20
	CCE [13]	31.93	32.52	84.07	84.92	85.52	85.39	54.58	55.0
	DWE [12]	28.03	29.17	77.25	78.12	85.14	86.00	51.10	56.08
iv) MTL for DCNNs or shallow linear models	AMTL [21]	29.36	30.15	83.15	84.37	83.17	84.05	53.40	54.22
	CMTL [20]	29.89	30.45	83.44	84.42	83.55	84.60	51.80	52.40
	2-sidedNN [24]	29.91	30.01	83.50	84.53	83.70	84.45	51.97	52.67
v) Structured outputs	Stacking-LP [29]	30.01	31.05	84.68	85.12	84.25	85.30	51.96	52.98
	LMGE [31]	30.17	31.24	84.32	85.02	84.52	85.64	53.07	54.62
vi) Joint MTL + Structured outputs	ELLA_LC [22]	28.15	29.09	81.98	82.84	82.15	83.17	52.40	54.68
	DMTL_LC [8]	28.23	31.71	82.01	84.07	82.23	84.30	52.35	54.70
vii) Proposed	CCE-LC cost	32.44	33.55	85.40	86.73	86.32	86.39	**56.40**	**60.73**
	FV-MTL with CCE-LC ($\beta = 0$)	31.60	32.15	82.21	86.96	87.10	88.51	55.45	54.69
	FV-MTL with CCE-LC ($\beta = 10$)	**32.83**	**33.77**	**85.70**	**87.00**	**87.54**	**88.69**	55.54	60.22

the proposed CCE-LC cost, i.e., the typical complete DCNN architecture illustrated in Fig. 1 replacing the sigmoid cross-entropy cost with the proposed CCE-LC cost, and b) FV-MTL with CCE-LC for $\beta = 0$. We set k equal to 32 and d equal to 64, which was the pair that reached the best overall MXinfAP according to Table III; similarly, in the case that CCE-LC is used alone the dimension of the extension layer was set to 2048. We performed comparisons with the following methods:

- i) A baseline where we use the ResNet-50 pre-trained network as feature generator; one SVM classifier per concept was trained using as features either the ResNet's output or its last FC layer.
- ii) The typical DCNN architecture with sigmoid cross-entropy cost, i.e., the ResNet-50 pre-trained network fine-tuned on each of the four datasets by simply replacing the classification layer with a new layer with dimension that equals to the number of concepts in the target domain as illustrated in Fig. 1 (i).
- iii) Extension strategy [2] for DCNNs, i.e., the DCNN architecture illustrated in Fig. 1 (ii), and four different STL cost functions: a) hinge-loss, b) sigmoid cross-entropy, c) cost sigmoid cross-entropy (CCE) [13], an extended version of (b) that also addresses the class-imbalance problem, and d) dynamic weighted euclidean loss (DEW) [12], an extension of the euclidean loss suitable for multi-label classification giving a greater penalty to concept prediction scores that have been ranked higher than the negative ground-truth annotated concepts. The size of the extension layer was set to 4096, according to the findings of Table III. This category of methods uses exactly the same architecture with the first intermediate version of our complete architecture (denoted as a) above), with the difference that each of the above three cost functions is used instead of the CCE-LC cost.
- iv) MTL, either as an integral part of DCNNs or for shallow linear models: a) AMTL [21], b) CMTL [20] and c) the 2-sided NN that was proposed in [24] for solving the GO-MTL method objective function [7].
- v) Structured output prediction: a) Stacking-LP [29], a two-layer stacking architecture combined with the label

power-set algorithm [29]. b) LMGE [31], an inner learning approach that uses the extracted features and exploits concept correlations in a single step.

- vi) Methods jointly using MTL and structured output prediction: a) DMTL_LC [8], and b) ELLA_LC [22].

We selected all the parameter values for these methods based on the training data, and in accordance with the recommendations provided in the corresponding papers.

We apply and evaluate all the above methods in two different ways (in a direct analogy to what is discussed in the last paragraph of Section IV-B); the specifics of these depended on whether they are complete DCNN architectures or shallow models that use pre-computed DCNN features. To the first category belong the following methods: Typical DCNN fine-tuning (group (ii)), all methods of group (iii) above, the 2-sided NN of [24], DMTL_LC [8], the proposed FV-MTL with CCE-LC and the latter's two intermediate versions. These methods are used a) as standalone classifiers, where the direct output of the complete network is evaluated (denoted as "direct" in Table IV), b) as feature generators, where SVM classifiers are trained on DCNN-based features. In the latter case, the output of the last layer of the complete trained network for each method was used as a feature vector to train one SVM per concept (denoted as "last layer" in Table IV). The remaining methods (that belong to the second category), i.e., the baseline of group (i) above, AMTL [21], CMTL [20], ELLA_LC [22], Stacking-LP [29] and LMGE [31], use the pre-trained ResNet-50 network as feature generator and the extracted features were used to train each of these methods. The methods specifically used in our experiments a) the ResNet-50 output layer (denoted as "direct" in Table IV), b) the ResNet-50 last FC layer (denoted as "last layer" in Table IV).

Table IV presents the results in terms of MXinfAP for the TRECVID-SIN dataset and in terms of MAP for the PASCAL-VOC and NUS-WIDE datasets. With respect to the direct output (Table IV: (a),(c),(e),(g)) we observe that the two intermediate versions of our proposed method perform quite well, outperforming the compared methods in the majority of cases. One exception is observed between the compared extension strategy [2] with sigmoid cross-entropy cost and the proposed FV-MTL with CCE-LC for $\beta = 0$, where their difference is that the latter also incorporates MTL. The results present fluctuations concerning which of the two methods performs better, depending on the dataset. However, jointly combining MTL and structured output prediction, using the proposed FV-MTL with CCE-LC for $\beta = 10$, further improves the concept annotation accuracy and outperforms all the other previously-published methods across all of the evaluated datasets, reaching the best overall concept annotation accuracy of 32.83%, 85.70%, 87.54% and 55.54% for TRECVID-SIN, PASCAL-VOC2007, PASCAL-VOC2012 and NUS-WIDE, respectively. The only exception is the NUS-WIDE dataset, where our intermediate version of the typical extension strategy with CCE-LC cost presents the best accuracy, and our complete architecture reaches the second-best performance. It should be noted that we compare our method with very recent methods; even our baseline is the ResNet-50 network that was ranked first in the ImageNet 2016 competition and our method outperforms it by approximately 3 to 4 percentage points. Similarly clear differences can be observed with respect to all the other compared methods. Even compared to the most recent DCNN with CCE cost [13], although the differences are smaller, we consistently outperform it by approximately 1 to 1.5 percentage points in all three datasets. Similar conclusions can be reached regarding the results presented in columns (b), (d), (f) and (h) of Table IV that refers to the second way of applying the compared methods, as described in the beginning of this section. We also evaluated the XinfAP per task regarding the proposed FV-MTL with CCE-LC and the other two best performing methods (i.e., DCNN with sigmoid cross-entropy cost and DCNN with CCE cost [13]) in the TRECVID-SIN dataset. Besides our overall best result (33.77% - Table IV), our method performs better than these other two well-performing methods for 25 out of the 38 evaluated concepts.

To investigate the statistical significance of the difference of the results of each method from the best performing method, i.e., the proposed FV-MTL, we used a paired t-test as suggested by [61]. We found that differences between the proposed FV-MTL with CCE-LC ($\beta = 10$) and all other previously-published mehtods that we compare with, per column of Table IV, are significant at 5% significance level.

Finally, we assess the robustness of the the proposed and the other two best performing methods (i.e., Sigmoid cross-entropy, and CCE costs [2], [13]) with respect to the TRECVID SIN dataset according to Table IV , when they are trained on smaller datasets for the same number of concepts. Specifically, Fig. 3 presents the reduction of MXinfAP when each of the compared methods is trained a) on only half of the keyframes of TRECVID SIN training set and b) on only a quarter of the keyframes for the same dataset, compared to the complete training set. We observe that the DCNN with sigmoid cross-entropy cost is affected by the smaller training datasets, as according to Fig. 3 its concept annotation accuracy is reduced by approximately 6 and 3 percentage points when the half and quarter training sets are used instead of the complete training set, respectively. In contrast, the proposed FV-MTL with CCE-LC for $\beta = 10$ and its intermediate versions, i.e., CCE-LC cost and FV-MTL with CCE-LC for $\beta = 0$, are robust to smaller training sets, exhibiting only a small reduction of MXinfAP compared to the case of using the complete training set.

E. Execution Times

We continue the analysis of our results by assessing the execution times during the training and classification phase of the different methods compared in this study. Table V summarizes the required execution time in hours for the proposed FV-MTL with CCE-LC for $\beta = 10$ and its two intermediate versions, defined in earlier sections, and also compares it with the rest of the methods. We observe that the proposed method is not considerably more computationally expensive than DCNN methods that use STL cost functions. Training of the baseline, AMTL and CMTL methods that use pre-computed features is a bit faster than the proposed method and

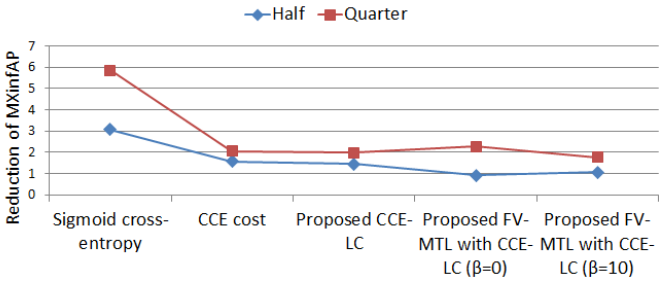

Fig. 3. Reduction of MXinfAP when only a half and a quarter of the training samples respectively are used instead of the complete training set, for the top-5 best performing methods with respect to the TRECVID SIN dataset according to Table IV. Lower values are better.

TABLE V
MEAN EXECUTION TRAINING/TESTING TIMES IN HOURS.

Category	Method	TRECVID-SIN	
		training	testing
i) Baseline (without fine-tuning)	ResNet-50 [10] as feature generator	14.25	2.45
ii) Typical DCNN fine-tuning	ResNet-50 [10]	17.33	2.47
iii) DCNNs (extension strategy [2]) with STL cost functions	Hinge-loss	17.35	2.48
	Sigmoid cross-entropy	17.45	2.47
	CCE [13]	17.75	2.50
	DWE [12]	17.85	2.50
iv) MTL for DCNNs or shallow linear models	AMTL [21]	14.75	2.50
	CMTL [20]	14.85	2.58
	2-sidedNN [24]	48.12	6.80
v) Structured outputs	Stacking-LP [29]	23.15	4.51
	LMGE [31]	15.17	2.68
vi) Joint MTL + Structured outputs	ELLA_LC [22]	20.97	2.53
	DMTL_LC [8]	49.27	6.84
vii) Proposed	CCE-LC cost	17.75	2.67
	FV-MTL with CCE-LC ($\beta = 0$)	17.53	3.17
	FV-MTL with CCE-LC ($\beta = 10$)	18.15	3.10

its intermediate versions; however, all these previous methods achieved lower accuracy than the proposed one, according to Table IV. During classification all the compared methods are executed on very similar time, except for the 2-sidedNN, Stacking-LP and DMTL_LC that are significantly slower. We conclude that our proposed FV-MTL with CCE-LC is faster than other MTL methods for DCNNs (2-sidedNN, DMTL_LC) both during training and classification, and also comparable in execution time with the second-best performing method of Table IV, i.e., DCNN with CCE cost [13].

F. Data Augmentation and Comparisons

Recently, improved accuracy has been achieved by image augmentations, i.e., feeding the DCNN with more than one image crops of the same image. For example, in the PASCAL-VOC2007 dataset this was shown to improve the MAP by 6 percentage points [11]. In Table VI we compare our proposed method with these approaches, however, due to the fact that this is a very computational intensive and time consuming process we present results only on the PASCAL-VOC2007

TABLE VI
MAP (%) FOR 20 PASCAL-VOC2007 CONCEPTS FOR METHODS THAT USE IMAGE AUGMENTATIONS.

Method	PASCAL-VOC2007
Simonyan et al. [9]	89.3
Wei et al. [11]	90.9
Wang et al. [12]	92.5
FV-MTL with CCE-LC ($\beta = 10$) + augmentations (VGG16)	**93.3**
FV-MTL with CCE-LC ($\beta = 10$) + augmentations (ResNet-50)	**94.6**

dataset. The following three SoA PASCAL-VOC2007 methods were selected: (i) Simonyan et al. [9]: A pre-trained ImageNet DCNN is applied on multiple image representations that are extracted and aggregated across multiple locations and scales. The resulting aggregated image descriptor (using the second-last layer as image feature representation) is used to train a linear SVM per concept. (ii) Wei et al. [11]: Many object segment hypotheses are given as input to a shared DCNN that has been pre-trained in the ImageNet dataset. The shared network's output is aggregated with max pooling in order to return a single multi-label prediction. The shared network is fine-tuned on the PASCAL-VOC dataset. (iii) Wang et al. [12]: Similar to Wei et al. [11], a pre-trained ImageNet DCNN is fine-tuned using many random crop hypotheses. Stochastic scaling and cropping over images is performed in this case in order to choose the most useful image crops by proposing the random crop pooling approach (RCP), instead of object proposal methods such as [62], [11]. Furthermore, the DWE loss function, also presented in Table IV, is used on the top of the network. One linear classifier is trained finally for each object class. Concerning the proposed architecture (FV-MTL with CCE-LC ($\beta = 10$)), this is fine-tuned on 20 random image object segment proposals per image extracted using the RCP method [12]. Similarly to [11] and [12] a shared DCNN is used to aggregate the probability scores w.r.t. each proposal using max-pooling and one SVM is trained for each object class. In this set of experiments, we firstly used the VGG16 [9] ImageNet pre-trained network as the base network of the proposed architecture, in order to have a fair comparison with methods [9], [11] and [12] that also use VGG16. Then, similarly to all of our previous experiments, we repeated this experiment using ResNet-50 [10] as our architecture's base network. We observe that the proposed method once again outperforms all the other compared methods and also that image augmentation is a robust way of increasing the accuracy of our architecture by approximately 7 percentage points.

V. CONCLUSION AND FUTURE WORK

In this study we proposed a DCNN architecture that jointly exploits implicit visual-level and explicit semantic-level concept relations. We built on ideas from MTL and structured output prediction in order to develop the FV-MTL approach for learning shared latent representations across the different tasks, and the new CCE-LC cost function that exploits the correlations between the concepts, respectively. The integrated DCNN architecture that emerges from combining these ap-

proaches was shown to improve concept annotation accuracy and outperformed the related state-of-the-art methods, without introducing any significant computational overhead. Specifically, it outperforms methods that do not impose MTL and structure outputs (e.g. [10], [2], [13], [12]), by approximately 4 percentage points, it also outperforms methods that solely consider either MTL or structured outputs (e.g., [21], [20], [24], [29], [31]) by approximately 2 percentage points, and finally it outperforms methods that jointly consider MTL and structured outputs (e.g., [22], [8]) by approximately 4 percentage points, in all three evaluated datasets. Finally, introducing image augmentations during the network's training was successfully applied to our method, further increasing its accuracy by approximately 7 percentage points.

Our plans for future research focus on the extension of the FV-MTL method in order to improve the way that implicit concept relations are captured, for example, by introducing a non-linear function. Furthermore, we will focus on different constraints in order to capture different kinds of semantic relations (i.e., not only correlations) between concepts. Another possible direction of future work is to extend our proposed architecture with an incremental learning methodology that will allow a trained network to be extended with new classes, without losing the ability of recognising the previously learned classes. This could speed-up the network's training, which, similarly to all other methods considered in this study, currently requires that the network is fine-tuned again when a new class needs to be learned.

REFERENCES

[1] M. Everingham, L. Van Gool, and et al., "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results."

[2] N. Pittaras, F. Markatopoulou, V. Mezaris, and I. Patras, "Comparison of Fine-Tuning and Extension Strategies for Deep Convolutional Neural Networks," in *Proc. of the 23rd Int. Conf. on MultiMedia Modeling (MMM 2017)*. Reykjavik, Iceland: Springer, 2017, pp. 102–114.

[3] X. Liang, L. Lee, and E. P. Xing, "Deep variation-structured reinforcement learning for visual relationship and attribute detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 4408–4417.

[4] L. Zhang, L. Lin, X. Liang, and K. He, "Is faster r-cnn doing well for pedestrian detection?" in *ECCV*, 2016.

[5] X. Liang, S. Liu, Y. Wei, L. Liu, L. Lin, and S. Yan, "Towards computational baby learning: A weakly-supervised approach for object detection," in *IEEE International Conference on Computer Vision (ICCV)*, Dec 2015, pp. 999–1007.

[6] A. Jalali, S. Sanghavi, C. Ruan, and P. K. Ravikumar, "A dirty model for multi-task learning," in *Advances in Neural Information Processing Systems*. Curran Associates, 2010, pp. 964–972.

[7] A. Kumar and H. Daume, "Learning task grouping and overlap in multi-task learning," in *Proc. of the 29th ACM Int. Conf. on Machine Learning (ICML 2012)*, Edinburgh, Scotland, 2012, pp. 1383–1390.

[8] F. Markatopoulou, V. Mezaris, and I. Patras, "Deep multi-task learning with label correlation constraint for video concept detection," in *Proc. of the Int. Conf. ACM Multimedia (ACMMM 2016)*. Amsterdam, The Netherlands: ACM, 2016, pp. 501–505.

[9] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *CoRR*, vol. abs/1409.1556, 2014. [Online]. Available: http://arxiv.org/abs/1409.1556

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

[11] Y. Wei, W. Xia, M. Lin, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan, "Hcp: A flexible cnn framework for multi-label image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 9, pp. 1901–1907, Sept 2016.

[12] M. Wang, C. Luo, R. Hong, J. Tang, and J. Feng, "Beyond Object Proposals: Random Crop Pooling for Multi-Label Image Recognition," *IEEE Transactions on Image Processing*, vol. 25, no. 12, pp. 5678–5688, Dec 2016.

[13] M. Bishay and I. Patras, "Fusing multilabel deep networks for facial action unit detection," in *Proc. of the 12th IEEE Int. Conf. on Automatic Face and Gesture Recognition (FG)*, 2017.

[14] A. Argyriou, T. Evgeniou, and M. Pontil, "Multi-task feature learning," *Advances in Neural Information Processing Systems (NIPS 2007)*, 2007.

[15] G. Obozinski and B. Taskar, "Multi-task feature selection," in *Proc. of the 23rd Int. Conf. on Machine Learning (ICML 2006). Workshop of Structural Knowledge Transfer for Machine Learning*, Pittsburgh, Pennsylvania, 2006.

[16] H. Mousavi, U. Srinivas, V. Monga, Y. Suo, M. Dao, and T. Tran, "Multi-task image classification via collaborative, hierarchical spike-and-slab priors," in *the IEEE Int. Conf. on Image Processing (ICIP 2014)*, Paris, France, 2014, pp. 4236–4240.

[17] T. Evgeniou and M. Pontil, "Regularized multi–task learning," in *Proc. of the 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2004)*, Seattle, WA, 2004, pp. 109–117.

[18] H. Daumé, III, "Bayesian multitask learning with latent hierarchies," in *the 25th Conf. on Uncertainty in Artificial Intelligence (UAI 2009)*. Quebec, Canada: AUAI Press, 2009, pp. 135–142.

[19] A. Argyriou, T. Evgeniou, and M. Pontil, "Convex multi-task feature learning," *Machine Learning*, vol. 73, no. 3, pp. 243–272, 2008.

[20] J. Zhou, J. Chen, and J. Ye, "Clustered multi-task learning via alternating structure optimization," *Advances in Neural Information Processing Systems (NIPS 2011)*, 2011.

[21] G. Sun, Y. Chen, X. Liu, and E. Wu, "Adaptive multi-task learning for fine-grained categorization," in *Proc. of the IEEE Int. Conf. on Image Processing (ICIP 2015)*, Sept 2015, pp. 996–1000.

[22] F. Markatopoulou, V. Mezaris, and I. Patras, "Online multi-task learning for semantic concept detection in video," in *Proc. of the IEEE Int. Conf. on Image Processing (ICIP 2016)*, Sept 2016, pp. 186–190.

[23] Z. Zhang, P. Luo, C. C. Loy, and X. Tang, "Facial landmark detection by deep multi-task learning," in *Proc. of the 13th Europ. Conf. on Computer Vision (ECCV 2014)*. Zurich, Switzerland: Springer, 2014, pp. 94–108.

[24] Y. Yang and T. Hospedales, "A unified perspective on multi-domain and multi-task learning," in *Proc. of the Int. Conf. on Learning Representations (ICLR 2015)*, San Diego, California, 2015.

[25] J. Smith, M. Naphade, and A. Natsev, "Multimedia semantic indexing using model vectors," in *Proc. of the Int. Conf. on Multimedia and Expo (ICME 2003)*. NY: IEEE, 2003, pp. 445–448.

[26] M.-F. Weng and Y.-Y. Chuang, "Cross-Domain Multicue Fusion for Concept-Based Video Indexing," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 10, pp. 1927–1941, 2012.

[27] J. Deng, N. Ding, Y. Jia, A. Frome, K. Murphy, S. Bengio, Y. Li, H. Neven, and H. Adam, *Large-Scale Object Classification Using Label Relation Graphs*. Zrich, Switzerland: Springer, 2014, pp. 48–64.

[28] N. Ding, J. Deng, K. P. Murphy, and H. Neven, "Probabilistic label relation graphs with ising models," in *Proc. of the 2015 IEEE Int. Conf. on Computer Vision (ICCV 2015)*. Washington, DC, USA: IEEE, 2015, pp. 1161–1169.

[29] F. Markatopoulou, V. Mezaris, N. Pittaras, and I. Patras, "Local features and a two-layer stacking architecture for semantic concept detection in video," *IEEE Transactions on Emerging Topics for Computing*, vol. 3, pp. 193–204, 2015.

[30] G.-J. Qi et al., "Correlative multi-label video annotation," in *Proc. of the 15th Int. Conf. on Multimedia*. NY: ACM, 2007, pp. 17–26.

[31] Y. Yang, F. Wu, F. Nie, H. T. Shen, Y. Zhuang, and A. G. Hauptmann, "Web and personal image annotation by mining label correlation with relaxed visual graph embedding," *IEEE Transactions on Image Processing*, vol. 21, no. 3, pp. 1339–1351, March 2012.

[32] H. Wang, H. Huang, and C. Ding, "Image annotation using bi-relational graph of images and semantic labels," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2011)*, June 2011, pp. 793–800.

[33] ——, "Image annotation using multi-label correlated green's function," in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2009)*, Sept 2009, pp. 2029–2034.

[34] M.-L. Zhang and K. Zhang, "Multi-label learning by exploiting label dependency," in *Proc. of the 16th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD 2010)*. NY, USA: ACM, 2010, pp. 999–1008.

[35] Y. Lu, W. Zhang, K. Zhang, and X. Xue, "Semantic context learning with large-scale weakly-labeled image set," in *Proc. of the 21st ACM*

Int. Conf. on Information and Knowledge Management. NY, USA: ACM, 2012, pp. 1859–1863.
[36] M. Baumgartner, "Uncovering deterministic causal structures: a boolean approach," *Synthese*, vol. 170, no. 1, pp. 71–96, 2009.
[37] Q. Luo, S. Zhang, T. Huang, W. Gao, and Q. Tian, "Superimage: Packing semantic-relevant images for indexing and retrieval," in *Proc. of the Int. Conf. on Multimedia Retrieval (ICMR 2014)*. NY, USA: ACM, 2014, pp. 41–48.
[38] X. Cai, F. Nie, W. Cai, and H. Huang, "New graph structured sparsity model for multi-label image annotations," in *Proc. of the IEEE International Conference on Computer Vision (ICCV 2013)*, 2013, pp. 801–808.
[39] B. Taskar, C. Guestrin, and D. Koller, "Max-margin markov networks," in *Proc. of the 16th Int. Conf. on Neural Information Processing Systems (NIPS 2003)*. MIT Press, 2003.
[40] J. Deng, S. Satheesh, A. C. Berg, and F. Li, "Fast and balanced: Efficient label tree learning for large scale object recognition," in *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2011, pp. 567–575.
[41] L. E. Sucar, C. Bielza, E. F. Morales, P. Hernandez-Leal, J. H. Zaragoza, and P. Larraaga, "Multi-label classification with bayesiannetwork-based chain classifiers," *Pattern Recognition Letters*, vol. 41, pp. 14 – 22, 2014.
[42] A. G. Schwing and R. Urtasun, "Fully connected deep structured networks," *CoRR*, vol. abs/1503.02351, 2015.
[43] Z. Deng, A. Vahdat, H. Hu, and G. Mori, "Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition," *CoRR*, vol. abs/1511.04196, 2015.
[44] S. Zheng, S. Jayasumana, and et al., "Conditional random fields as recurrent neural networks," in *Proc. of the Int. Conf. on Computer Vision (ICCV 2015)*, 2015.
[45] X. Zhao, X. Li, and Z. Zhang, "Joint structural learning to rank with deep linear feature learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 10, pp. 2756–2769, Oct 2015.
[46] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" *Advances in Neural Information Processing Systems (NIPS 2014)*, pp. 3320–3328, 2014.
[47] Y. Yang and T. M. Hospedales, "Deep multi-task representation learning: A tensor factorisation approach," in *Proc. of the International Conference on Learning Representations (ICLR)*, 2017.
[48] W. Ouyang, X. Chu, and X. Wang, "Multi-source deep learning for human pose estimation," in *Proc. of the the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR 2014)*. Columbus, Ohio: IEEE, 2014, pp. 2337–2344.
[49] X. Zhai, Y. Peng, and J. Xiao, "Learning cross-media joint representation with sparse and semisupervised regularization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 24, no. 6, pp. 965–978, June 2014.
[50] X. Xu, T. M. Hospedales, and S. Gong, "Discovery of shared semantic spaces for multiscene video query and summarization," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 6, pp. 1353–1367, June 2017.
[51] J. Li, Y. Wu, and K. Lu, "Structured domain adaptation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 8, pp. 1700–1713, Aug 2017.
[52] G. S. Xie, X. Y. Zhang, S. Yan, and C. L. Liu, "Hybrid cnn and dictionary-based models for scene recognition and domain adaptation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 6, pp. 1263–1274, June 2017.
[53] P. Over and et al., "TRECVID 2013 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics," in *TRECVID 2013*. NIST, USA, 2013.
[54] M. Wang, X. S. Hua, R. Hong, J. Tang, G. J. Qi, and Y. Song, "Unified video annotation via multigraph learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 5, pp. 733–746, May 2009.
[55] R. Hong, M. Wang, Y. Gao, D. Tao, X. Li, and X. Wu, "Image annotation by multiple-instance learning with discriminative feature mapping and selection," *IEEE Transactions on Cybernetics*, vol. 44, no. 5, pp. 669–680, May 2014.
[56] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.
[57] T.-S. Chua, J. Tang, R. Hong, H. Li, Z. Luo, and Y.-T. Zheng, "Nus-wide: A real-world web image database from national university of singapore," in *Proc. of ACM Conf. on Image and Video Retrieval (CIVR 2009)*, Santorini, Greece, July 8-10, 2009.
[58] O. Russakovsky, J. Deng, and H. S. et al., "ImageNet Large Scale Visual Recognition Challenge," *Int. Journal of Computer Vision (IJCV 2015)*, vol. 115, no. 3, pp. 211–252, 2015.
[59] E. Yilmaz, E. Kanoulas, and J. A. Aslam, "A simple and efficient sampling method for estimating ap and ndcg," in *Proc. of the 31st ACM Int. Conf. on Research and Development in Information Retrieval (SIGIR 2008)*, Singapore, 2008, pp. 603–610.
[60] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.
[61] H. M. Blanken, A. P. de Vries, H. E. Blok, and L. Feng, *Multimedia Retrieval*. NY: Springer, 2005.
[62] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, and A. W. M. Smeulders, "Segmentation as selective search for object recognition," in *Int. Conf. on Computer Vision*, Nov 2011, pp. 1879–1886.

Foteini Markatopoulou received her BSc in Informatics from Aristotle University of Thessaloniki in 2011 and her MSc in Advanced Computing: Machine Learning and Data Mining from University of Bristol. Her research is in the area of Machine Learning with application in Multimedia Analysis. She is a PhD student in Queen Mary University of London and works as a research assistant at the Information Technologies Institute (ITI) of the Centre of Research & Technology Hellas (CERTH).

Vasileios Mezaris received the BSc and PhD in Electrical and Computer Engineering from the Aristotle University of Thessaloniki, Greece, in 2001 and 2005 respectively. He is a Research Director at the Information Technologies Institute (ITI) of the Centre for Research of Technology Hellas (CERTH). His research interests include image and video analysis, machine learning for multimedia analysis and big data analytics, event detection in multimedia, content-based and semantic image and video retrieval, and applications of image/video analysis and machine learning. He has coauthored more than 40 journal papers, 15 book chapters, 150 conference papers and 3 patents. He serves as Associate Editor for the IEEE Signal Processing Letters and IEEE Transactions on Multimedia, and is a Senior Member of the IEEE.

Ioannis Patras received the B.Sc. and M.Sc. degrees in computer science from the Computer Science Department, University of Crete, Heraklion, Greece, in 1994 and 1997, respectively, and the Ph.D. degree from the Department of Electrical Engineering, Delft University of Technology, Delft (TU Delft), The Netherlands, in 2001. He is a Reader (Associate Professor) in the School of Electronic Engineering and Computer Science Queen Mary University of London, London, U.K. His current research interests are in the area of Computer Vision, Machine Learning and Affective Computing, with emphasis on the analysis of visual data depicting humans and their activities. He is an Associate Editor of the Image and Vision Computing Journal, Pattern Recognition, and Computer Vision and Image Understanding.