Comparison of Fine-tuning and Extension Strategies for Deep Convolutional Neural Networks

Nikiforos Pittaras[1], Foteini Markatopoulou[1,2], Vasileios Mezaris[1], and Ioannis Patras[2]

[1] Information Technologies Institute (ITI), CERTH, Thermi 57001, Greece
{npittaras,markatopoulou,bmezaris}@iti.gr
[2] Queen Mary University of London, Mile end Campus, UK, E14NS
i.patras@qmul.ac.uk

Abstract. In this study we compare three different fine-tuning strategies in order to investigate the best way to transfer the parameters of popular deep convolutional neural networks that were trained for a visual annotation task on one dataset, to a new, considerably different dataset. We focus on the concept-based image/video annotation problem and use ImageNet as the source dataset, while the TRECVID SIN 2013 and PASCAL VOC-2012 classification datasets are used as the target datasets. A large set of experiments examines the effectiveness of three fine-tuning strategies on each of three different pre-trained DCNNs and each target dataset. The reported results give rise to guidelines for effectively fine-tuning a DCNN for concept-based visual annotation.

Keywords: Concept detection; deep learning; visual analysis

1 Introduction

Concept-based video annotation, also known as video concept detection, refers to the problem of annotating a video fragment (e.g., a keyframe) with one or more semantic concepts (e.g., "table", "dog") [14]. The state of the art approach to doing this is to train a deep convolutional neural network (DCNN) on a set of concepts [3], [5], [12]. Then, a test keyframe can be forward propagated by the DCNN, to be annotated with a set of concept labels. DCNN training requires the learning of millions of parameters, which means that a small-sized training set could easily over-fit the DCNN on the training data. It has been proven that the bottom layers of a DCNN learn rather generic features, useful for different domains, while the top layers are task-specific [18]. Transferring a pre-trained network in a new dataset by fine-tuning its parameters is a common strategy that can take advantage of the bottom generic layers and adjust the top layers to the target dataset and the new target concepts [18], [5], [2].

In this study we compare three fine-tuning methods in order to investigate the best way to transfer the parameters of a DCNN trained on a source dataset

Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavík, Iceland, Springer LNCS vol. 10132, pp. 102-114, Jan. 2017.

to a different target dataset that requires a different set of concepts. Although DCNN fine-tuning has been presented in previous studies [18], [5], [2], this is the first work, to our knowledge, that performs a large number of experimental comparisons considering three different pre-trained DCNNs, two different subsets of concepts for which the pre-trained networks will be fine-tuned, two different target datasets, and three fine-tuning strategies with many parameter evaluations for each of them, with the purpose of comparing these strategies. Experiments performed on the TRECVID 2013 SIN [10] and the PASCAL VOC-2012 classification [4] datasets show that increasing the depth of a pre-trained network with one more fully-connected layer and fine-tuning the rest of the layers on the target dataset can improve the network's concept detection accuracy compared to other fine-tuning approaches.

2 Related Work

Fine-tuning is a process where the weights of a pre-trained DCNN are used as the starting point for a new target training set and they are modified in order to adapt the pre-trained DCNN to the new target dataset. Then, the fine-tuned DCNN can be used either as feature generator, i.e., the output of one or more hidden layers is typically used as a global keyframe/image representation [13], or as standalone classifier that performs the final class label prediction directly.

Different DCNN-based transfer learning approaches have been successfully applied in many datasets. The most straight-forward approach replaces the classification layer of a pre-trained DCNN with a new output layer that corresponds to the categories that should be learned with respect to the target dataset, [2], [18], [5]. Generalizing this approach, the weights of the first K network layers can remain frozen, i.e., they are copied from the pre-trained DCNN and kept unchanged, and the rest of the layers (be it just the last one or more than one) are learned from scratch [9], [1]. Alternatively, the copies of the first K layers could be allowed to adapt to the target dataset with a low learning rate. For example, [18] investigates which layers of Alexnet [7] are generic, i.e., can be directly transferred to a target domain, and which layers are dataset-specific. Furthermore, experiments in [18] show that fine-tuning the transferred layers of a network works better than freezing them. However, neither of these studies investigates how low the learning rate for the aforementioned layers should be, relative to the new layers, during fine-tuning. Other studies extend the pre-trained network by one or more fully connected layers, which seems to improve the above transfer learning strategies [15], [8], [9], [1]. However, the optimal number of extension layers and the size of them has not been investigated before. Although fine-tuning has been applied in many studies, a complete understanding of what fine-tuning parameters (e.g., number/size of extension layers, learning rate) work better has not been extensively examined. Furthermore, a thorough comparison of all the available fine-tuning alternatives is yet to appear in the literature.

Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavík, Iceland, Springer LNCS vol. 10132, pp. 102-114, Jan. 2017.

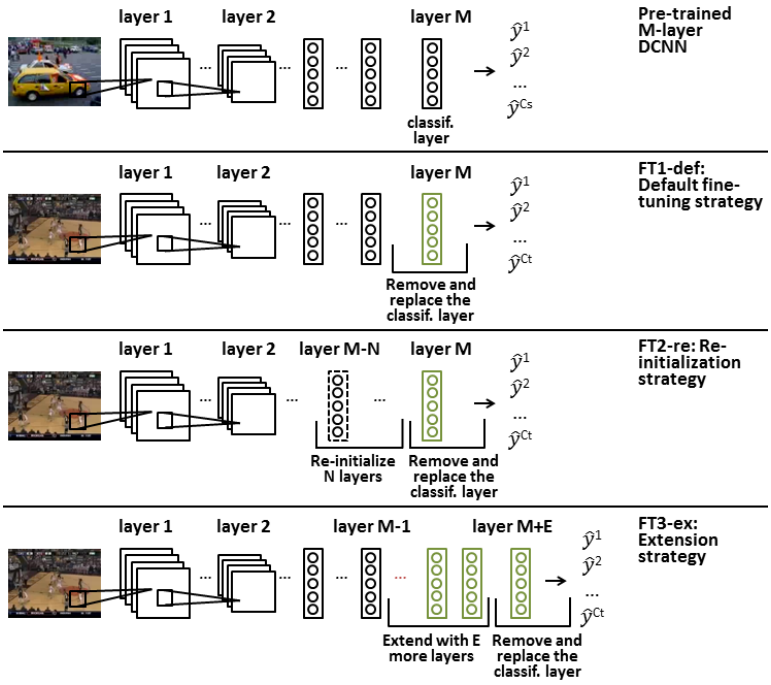

Fig. 1. Fine-tuning strategies outline.

3 Comparison of Fine-tuning and Extension Strategies for DCNNs

Let D_s denote a pre-trained DCNN, trained on C_s categories using a source dataset, and D_t denote the target DCNN, fine-tuned on C_t categories of a different target dataset. In this section we present the three fine-tuning strategies (Fig. 1) that we compare for the problem of visual annotation, in order to effectively fine-tune DCNNs D_s that were trained on a large visual dataset for a new target video/image dataset. These three fine-tuning strategies are as follows:

- FT1-def: Default fine-tuning strategy: This is the typical strategy that modifies the last fully-connected layer of D_s to produce the desired number of outputs C_t, by replacing the last fully-connected layer with a new C_t-dimensional classification fully-connected layer.
- FT2-re: Re-initialization strategy: In this scenario, similar to FT1-def, the last fully-connected layer is replaced by a new C_t-dimensional classification layer. The weights of the last N layers, preceding the classification layer, are also re-initialized (i.e., reset and learned from scratch).
- FT3-ex: Extension strategy: Similar to the previous two strategies, the last fully-connected layer is replaced by a new C_t-dimensional classification fully-connected layer. Subsequently, the network is extended with E fully-connected

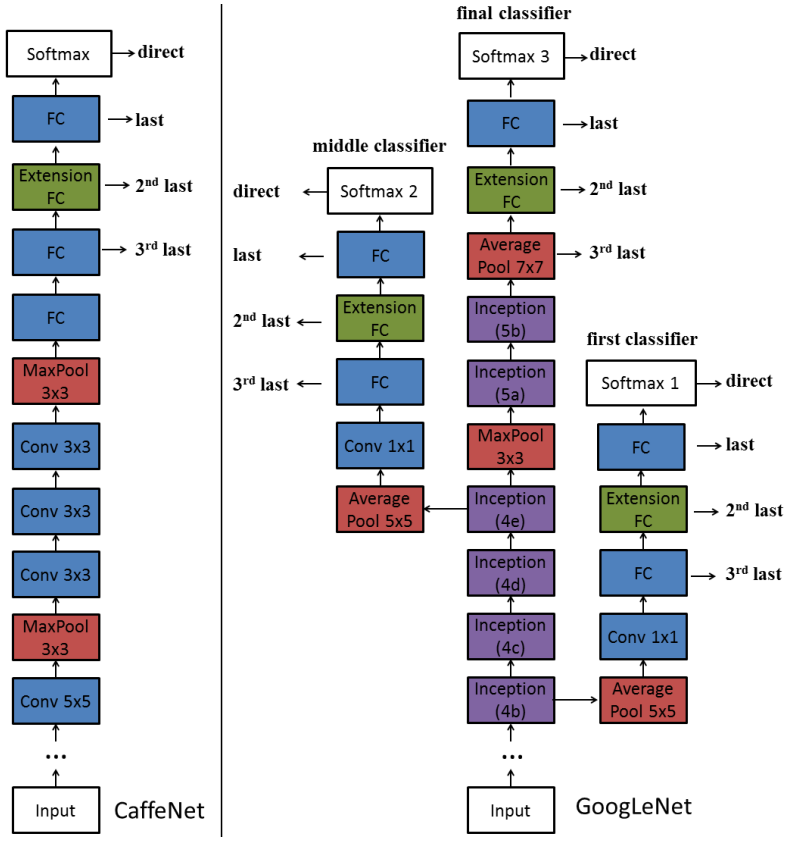

Fig. 2. A simplified illustration of the CaffeNet [7] (left) and GoogLeNet [16] (right) architectures used after insertion of one extension layer. Each of the inception layers of GoogLeNet consists of six convolution layers and one pooling layer. The figure also presents the direct output of each network and the output of the last three layers that were used as features w.r.t. FT3-ex strategy. Similarly, the corresponding layers were used for the FT1-def and FT2-re strategies.

layers of size L that are placed on the bottom of the modified classification layer. These additional layers are initialized and trained from scratch during fine-tuning, at the same rate as the modified classification layer. One example of a modified network after the insertion of one extension layer for two popular DCNN architectures that were also used in our experimental study in Section 4, is presented in Fig. 2. Regarding the GoogLeNet architecture, which has two additional auxiliary classifiers, an extension layer was also inserted in each of them.

Each fine-tuned network D_t is used in two different ways to annotate new test keyframes/images with semantic concepts. a) Direct classification: Each test keyframe/image is forward propagated by D_t and the network's output is used

as the final class distribution assigned to the keyframe/image. b) D_t is used as feature generator: The training set is forward propagated by the network and the features extracted from one or more layers of D_t are used as feature vectors to subsequently train one supervised classifier (e.g., Logistic Regression) per concept. Then, each test keyframe/image is firstly described by the DCNN-based features and subsequently these features serve as input to the trained Logistic Regression classifiers.

4 Experimental Study

4.1 Datasets and Experimental Setup

The TRECVID SIN task 2013 [10] dataset and the PASCAL VOC-2012 [4] dataset were utilized to train and evaluate the compared fine-tuned networks. The TRECVID SIN dataset consists of low-resolution videos, segmented into video shots; each shot is represented by one keyframe. The dataset is divided into a training and a test set (approx. 600 and 200 hours, respectively). The training set is partially annotated with 346 semantic concepts. The test set is evaluated on 38 concepts, i.e., a subset of the 346 concepts. The PASCAL VOC-2012 [4] dataset consists of images annotated with one object class label of the 20 available object classes. PASCAL VOC-2012 is divided into training, validation and test sets (consisting of 5717, 5823 and 10991 images, respectively). We used the training set to train the compared methods, 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).

For each dataset we fine-tuned the following three pre-trained DCNNs: i) CaffeNet-1k, the reference implementation of Alexnet [7] by Caffe [6], trained on 1000 ImageNet categories, ii) GoogLeNet-1k [16], trained on the same 1000 ImageNet [11] categories and iii) GoogLeNet-5k, trained using 5055 ImageNet [11]

k/e	0.25	0.5	1	2	4	8	16	32
0.050	0.348	0.362	0.402	0.417	0.437	0.434	0.451	0.462
0.075	0.341	0.349	0.388	0.412	0.438	0.453	0.462	0.462
0.100	0.346	0.354	0.388	0.420	0.434	0.455	0.463	**0.470**
0.250	0.328	0.361	0.397	0.421	0.430	0.450	0.455	0.468
0.500	0.306	0.354	0.388	0.415	0.439	0.447	0.451	0.444
0.750	0.284	0.349	0.381	0.410	0.431	0.443	0.448	0.448
1.000	0.257	0.321	0.367	0.390	0.430	0.442	0.450	0.436

Table 1. Classification accuracy for the CaffeNet-1k-60-SIN and the FT1-def strategy for different values of the learning rate multiplier of the pre-trained layers (k), in the vertical axis, and different number of training epochs (e), in the horizontal axis. For each value of parameter e the best accuracy reached is underlined. The globally best accuracy is bold and underlined.

Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavík, Iceland, Springer LNCS vol. 10132, pp. 102-114, Jan. 2017.

	(A) CaffeNet-1k-60-SIN					(B) CaffeNet-1k-345-SIN				
conf. / layer	direct	last	2nd last	3rd last	fused	direct	last	2nd last	3rd last	fused
	(a)	(b)	(c)	(d)	(e)	(a)	(b)	(c)	(d)	(e)
FT1-def	16.86	23.60	22.93	20.86	24.74	18.26	24.10	23.51	20.84	25.21
FT2-re1	15.35	22.09	21.52	20.79	23.93	17.43	23.36	22.88	20.71	24.48
FT2-re2	13.97	19.76	17.49	18.46	20.16	15.74	22.52	19.50	20.81	23.12
FT3-ex1-64	18.74	23.22	22.80	23.46	24.69	20.36	22.89	21.44	22.92	24.79
FT3-ex1-128	18.49	24.45	23.74	23.09	**25.35**	20.30	24.21	22.96	22.92	25.26
FT3-ex1-256	17.25	24.02	23.23	23.29	25.04	19.79	24.72	24.01	23.00	25.63
FT3-ex1-512	17.01	24.49	23.10	23.13	25.27	19.64	25.23	23.99	23.46	**25.94**
FT3-ex1-1024	16.53	24.19	21.37	22.16	24.78	18.59	25.29	22.39	23.65	25.64
FT3-ex1-2048	15.81	24.40	20.97	22.75	24.77	18.22	25.25	22.03	22.54	25.45
FT3-ex1-4096	14.94	24.11	19.88	21.42	24.02	17.67	24.79	21.50	23.18	25.05
FT3-ex2-64	17.26	20.53	21.03	21.48	21.61	17.02	19.53	20.23	19.62	20.67
FT3-ex2-128	17.72	22.17	22.27	22.36	22.88	19.42	22.93	22.24	21.79	23.13
FT3-ex2-256	17.17	23.46	23.12	23.20	24.19	19.35	24.09	23.98	23.60	24.80
FT3-ex2-512	16.77	23.71	23.65	23.72	24.96	19.00	24.89	24.01	23.90	25.52
FT3-ex2-1024	16.18	24.14	22.88	22.78	24.82	18.48	25.08	23.66	23.07	25.37
FT3-ex2-2048	15.67	24.17	22.36	21.42	24.59	18.20	25.55	22.13	22.50	25.08
FT3-ex2-4096	16.34	24.12	21.66	20.33	24.59	17.64	25.23	22.08	21.94	25.07

Table 2. MXinfAP (%) for CaffeNet-1k-60-SIN (sub-table (A), left) and CaffeNet-1k-345-SIN (sub-table (B), right). For each sub-table, the best result per column is underlined. The globally best result per sub-table is bold and underlined.

categories. Each of these networks was fine-tuned on the 345 TRECVID SIN concepts (i.e., all the available TRECVID SIN concepts, except for one which was discarded because only 5 positive samples are provided for it), which resulted to a training set of 244619 positive examples. CaffeNet-1k was also fine-tuned on a subset of 60 TRECVID SIN concepts. We refer to each of these fine-tuned networks as CaffeNet-1k-345-SIN, GoogLeNet-1k-345-SIN, GoogLeNet-5k-345-SIN and CaffeNet-1k-60-SIN, respectively. In addition, each of these original networks was fine-tuned on the positive examples of the PASCAL VOC-2012 training set. These networks are labeled as CaffeNet-1k-VOC, GoogLeNet-1k-VOC and GoogLeNet-5k-VOC, respectively.

In performing pre-trained DCNN fine-tuning, we compared the three fine-tuning strategies presented in Section 3. Specifically, in all cases we discarded and replaced the classification fully-connected (fc) layer of the utilized pre-trained network, with a 60-dimensional or a 345-dimensional fc classification layer for the 60 or 345 concepts of the TRECVID SIN dataset respectively, or with a 20-dimensional classification layer for the 20 object categories of the PASCAL VOC-2012 dataset. We examined two values for parameter N of the FT2-re strategy; we refer to each configuration as FT2-re1 (for $N = 1$) and FT2-re2 (for $N = 2$). The FT3-ex strategy was examined for two settings of network extensions $E \in \{1, 2\}$: i.e., extending the network by one or two fc layers, respectively, followed by ReLU (Rectified Linear Units) and Dropout layers. The size of each extension layer was examined for 7 different dimensions:

	final classifier					middle classifier		first classifier	
conf / layer	direct	last	2nd last	3rd last	fused	direct	fused	direct	fused
	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)
(A) GoogleNet-1k-345-SIN									
FT1-def	21.67	28.15	28.29	-	29.05	22.70	28.62	20.36	25.95
FT2-re1	19.80	26.85	26.97	-	27.97	22.17	28.13	20.08	25.29
FT2-re2	19.32	26.90	26.71	-	27.65	21.61	27.60	19.40	25.01
FT3-ex1-64	24.18	27.46	25.77	28.59	28.60	22.08	26.27	18.07	22.49
FT3-ex1-128	24.44	28.90	28.10	28.82	30.09	22.65	27.38	20.03	24.53
FT3-ex1-256	23.93	29.48	28.61	28.64	30.23	22.04	27.54	19.87	25.00
FT3-ex1-512	23.63	29.82	28.34	28.11	**30.43**	22.15	28.19	19.94	25.33
FT3-ex1-1024	23.01	29.95	27.37	28.28	30.05	21.87	28.41	19.90	26.03
FT3-ex1-2048	23.34	30.21	27.05	28.80	30.42	21.41	28.69	19.93	26.18
FT3-ex1-4096	22.47	30.43	26.90	28.44	30.27	21.60	28.33	19.48	26.59
FT3-ex2-64	15.91	16.72	18.55	17.83	18.49	10.65	13.58	9.50	12.03
FT3-ex2-128	22.09	24.38	24.86	24.74	25.59	17.58	21.65	14.10	18.05
FT3-ex2-256	23.91	27.80	27.82	27.95	28.75	20.77	24.46	17.56	21.51
FT3-ex2-512	23.06	28.31	28.02	28.45	29.04	21.16	26.19	18.95	23.18
FT3-ex2-1024	22.77	29.01	28.63	28.86	29.92	20.58	26.64	18.87	24.15
FT3-ex2-2048	22.84	29.53	28.53	27.99	29.83	20.44	27.07	18.86	24.87
FT3-ex2-4096	22.36	29.90	27.85	27.49	29.93	20.72	27.32	19.01	25.44
(B) GoogleNet-5k-345-SIN									
FT1-def	22.45	29.60	29.80	-	30.58	23.08	29.41	21.25	26.00
FT2-re1	20.88	28.44	28.43	-	29.58	22.51	28.55	20.37	25.16
FT2-re2	19.08	27.21	27.17	-	28.02	21.73	28.44	20.07	25.74
FT3-ex1-64	25.48	28.86	26.86	29.22	29.62	23.30	28.37	20.20	24.47
FT3-ex1-128	25.52	29.75	28.66	29.57	30.60	23.98	28.82	20.87	25.38
FT3-ex1-256	24.79	30.16	28.99	30.26	31.11	23.62	29.56	21.06	26.32
FT3-ex1-512	24.28	30.86	29.26	29.68	31.47	23.54	29.86	20.71	26.32
FT3-ex1-1024	24.03	31.02	28.78	29.35	31.55	23.43	29.90	20.53	26.57
FT3-ex1-2048	23.37	31.02	27.24	29.37	31.02	23.29	29.94	20.56	26.61
FT3-ex1-4096	23.07	30.91	28.98	29.61	**31.57**	22.85	29.64	20.82	26.26
FT3-ex2-64	16.44	17.51	19.62	19.95	20.09	11.43	15.12	10.65	13.33
FT3-ex2-128	23.87	26.19	26.73	26.05	27.02	18.70	23.64	14.87	19.95
FT3-ex2-256	24.46	28.94	28.69	28.68	29.57	22.68	26.98	18.75	23.10
FT3-ex2-512	23.95	29.44	29.07	28.94	30.14	22.72	28.22	20.20	24.79
FT3-ex2-1024	23.41	30.03	28.80	29.54	30.63	22.79	29.10	19.74	25.68
FT3-ex2-2048	23.38	30.74	28.98	28.21	30.61	22.29	29.34	19.57	26.23
FT3-ex2-4096	23.07	31.21	28.94	27.98	30.93	22.11	29.40	19.64	26.11

Table 3. MXinfAP (%) for GoogLeNet-1k-345-SIN (sub-table (A), top) and GoogLeNet-5k-345-SIN (sub-table (B), bottom). For each sub-table, the best result per column is underlined. The globally best result per sub-table is bold and underlined.

$L \in \{64, 128, 256, 512, 1024, 2048, 4096\}$. We refer to these configurations as FT3-exE-L. The new layers' learning rate and momentum was set to 0.01 and 5e−4, whereas the mini-batch size was restricted by our hardware resources and set to 256 and 128 for the CaffeNet and GoogLeNet configurations, respectively.

For the purpose of evaluation, we then tested each fine-tuned network on the TRECVID SIN 2013 test set that consists of 112677 representative keyframes and 38 semantic concepts on the indexing problem; that is, given a concept,

	CaffeNet-1k-VOC				
conf / layer	direct	last	2nd last	3rd last	fused
	(a)	(b)	(c)	(d)	(e)
FT1-def	72.77	69.80	72.59	69.95	73.29
FT2-re1	8.22	8.22	8.22	27.87	27.57
FT2-re2	23.53	25.05	26.33	25.74	29.27
FT3-ex1-64	72.61	70.27	72.75	69.70	73.86
FT3-ex1-128	73.53	71.59	74.20	69.96	74.52
FT3-ex1-256	73.63	71.90	74.59	69.78	74.81
FT3-ex1-512	<u>73.84</u>	72.34	74.18	69.64	74.85
FT3-ex1-1024	73.76	72.42	72.49	69.91	74.48
FT3-ex1-2048	73.41	<u>72.59</u>	72.46	69.39	74.30
FT3-ex1-4096	73.04	71.14	74.17	69.44	74.70
FT3-ex2-64	51.42	43.82	59.64	64.63	61.55
FT3-ex2-128	62.33	58.36	67.95	71.16	69.24
FT3-ex2-256	67.97	64.64	72.64	73.52	73.04
FT3-ex2-512	70.89	68.38	74.94	<u>73.80</u>	75.04
FT3-ex2-1024	72.55	71.22	<u>75.28</u>	73.26	**<u>75.65</u>**
FT3-ex2-2048	73.02	72.37	73.80	72.31	75.29
FT3-ex2-4096	66.43	60.83	69.10	71.69	72.67

Table 4. MAP % for CaffeNet-1k-VOC. For the FT-re strategy we trained the network with learning rate 10 times lower that of all the other cases. Otherwise, the network did not converge. The best result per column is underlined. The globally best result is bold and underlined.

return the 2000 test keyframes that are more likely to represent it. In addition, we examined classification performance on the PASCAL VOC-2012 validation set, consisting of 5823 images and 20 object categories. We fine-tuned the total of 17 configurations times 7 networks on a Tesla K40 GPU, over a period of 2 months. All networks were trained and implemented in Caffe [6].

4.2 Preliminary Experiments for Parameter Selection

A set of preliminary experiments on the CaffeNet-1k-60-SIN and the FT1-def strategy was performed, in order to investigate how the learning rate of the pre-trained layers and the number of training epochs affect the performance of a fine-tuned network. Specifically, we partitioned the training set of the TRECVID SIN dataset into training and validation sets, which resulted to 71457 and 3007 keyframes, respectively. Momentum and weight decay were set to 0.9 and 5e−4, respectively. We examined learning rate values for the pre-trained layers equal to $LR_{pre} = k \times LR_{new}$, where $k \in \{0, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1\}$ and $LR_{new} = 0.01$ is the learning rate of the new classification layer that will be trained from scratch. A value of $k = 0$ keeps the pre-trained layers' weights "frozen", while the value of 1 makes them learn as fast as the new layers. To investigate the effect of the training epochs, each fine-tuning run was examined for a range of maximum epochs equal to: $\{0.25, 0.5, 1, 2, 4, 8, 16, 32\}$.

	final classifier					middle classifier		first classifier	
conf / layer	direct	last	2nd last	3rd last	fused	direct	fused	direct	fused
	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)
(A) GoogleNet-1k-VOC									
FT1-def	82.00	85.83	86.31	-	87.69	80.71	84.01	78.93	79.01
FT2-re1	80.42	83.77	86.34	-	86.66	79.39	81.73	78.06	76.92
FT2-re2	77.34	77.43	82.51	-	82.31	75.94	77.46	74.28	73.01
FT3-ex1-64	78.98	80.29	85.19	85.76	86.11	76.07	81.93	71.46	76.58
FT3-ex1-128	80.14	83.86	87.74	86.22	87.64	77.89	83.52	73.64	78.00
FT3-ex1-256	81.11	84.97	88.04	86.46	87.95	78.95	83.96	75.00	79.01
FT3-ex1-512	81.42	85.41	87.70	86.57	**88.16**	79.54	84.35	76.03	79.39
FT3-ex1-1024	81.82	85.85	86.95	86.69	88.08	80.06	84.27	76.86	79.37
FT3-ex1-2048	82.18	86.36	86.31	86.81	87.99	80.70	84.38	77.33	79.08
FT3-ex1-4096	82.01	86.49	86.23	86.93	87.86	80.96	84.26	78.30	79.56
FT3-ex2-64	44.81	42.04	52.48	51.70	52.19	36.57	44.36	33.32	38.77
FT3-ex2-128	75.11	67.81	81.32	81.70	80.54	61.05	70.30	52.39	61.69
FT3-ex2-256	78.05	76.65	85.38	85.49	85.00	69.45	77.59	62.56	71.45
FT3-ex2-512	80.06	82.70	87.12	86.69	87.09	73.35	81.16	67.26	74.49
FT3-ex2-1024	81.26	84.32	86.75	86.39	87.39	76.46	82.53	70.19	76.38
FT3-ex2-2048	81.72	85.08	86.46	85.96	87.52	78.20	83.19	73.10	78.19
FT3-ex2-4096	81.71	84.82	86.14	85.05	87.00	78.93	83.14	74.08	77.80
(B) GoogleNet-5k-VOC									
FT1-def	82.39	86.75	86.74	-	88.01	81.10	84.25	78.96	79.06
FT2-re1	80.50	85.21	86.91	-	87.44	79.58	82.76	77.78	77.23
FT2-re2	77.73	78.81	83.13	-	83.11	75.28	77.34	71.99	69.65
FT3-ex1-64	79.74	82.86	86.41	86.26	86.92	76.36	82.72	72.32	77.51
FT3-ex1-128	80.47	85.50	88.26	86.56	88.12	78.57	84.12	74.01	78.76
FT3-ex1-256	81.43	85.81	88.33	86.73	**88.36**	79.31	84.48	75.29	79.12
FT3-ex1-512	81.65	85.91	87.84	86.90	88.33	79.99	84.76	76.25	79.69
FT3-ex1-1024	82.30	86.48	87.01	86.89	88.20	80.68	84.56	77.32	79.32
FT3-ex1-2048	82.51	86.93	86.80	86.96	88.23	81.15	84.51	77.97	79.62
FT3-ex1-4096	82.39	87.20	86.37	87.05	88.13	81.52	84.45	78.43	79.65
FT3-ex2-64	43.85	45.11	53.99	51.67	52.81	39.10	47.22	32.42	38.72
FT3-ex2-128	75.89	70.96	82.85	83.34	82.51	63.27	72.34	54.45	63.64
FT3-ex2-256	78.94	80.30	86.44	86.43	86.01	69.19	77.67	65.31	72.75
FT3-ex2-512	80.47	82.83	87.56	87.00	87.38	75.17	81.44	66.50	74.38
FT3-ex2-1024	81.47	84.54	86.81	86.53	87.58	76.99	82.85	71.09	76.74
FT3-ex2-2048	82.11	85.49	86.90	86.28	87.76	78.15	83.24	73.55	77.69
FT3-ex2-4096	80.50	83.83	85.82	84.71	86.64	77.49	81.79	74.66	78.21

Table 5. MAP % for GoogleNet-1k-VOC (sub-table (A), top) and GoogleNet-5k-VOC (sub-table (B), bottom). For each sub-table, the best result per column is underlined. The globally best result per sub-table is bold and underlined.

Table 1 presents the results w.r.t. the accuracy on the validation set, as this metric is implemented in the Caffe framework. We can observe that smaller learning rate values for the pre-trained layers and higher values for the training epochs improve accuracy. Consequently, we selected the best values of 0.1 and 32 for the learning rate multiplier and the maximum number of epochs, respectively, and kept them fixed for the rest of the experiments.

4.3 Main Findings of the Study

Table 2 presents the results, in terms of Mean Extended Inferred Average Precision (MXinfAP), of the CaffeNet-1k-60-SIN (left) and the CaffeNet-1k-345-SIN (right), for the three fine-tuning strategies of Section 3. In addition, Table 3 presents the MXinfAP of the GoogLeNet-1k-345-SIN (top) and GoogLeNet-5k-345-SIN (bottom). MXinfAP [17] is an approximation of the MAP, suitable for the TRECVID SIN partially annotated dataset. Similarly, Table 4 presents the results in terms of MAP of the CaffeNet-1k-VOC and Table 5 presents the MAP of the GoogLeNet-1k-VOC (top) and GoogLeNet-5k-VOC (bottom).

For each pair of utilized network and fine-tuning strategy we evaluate: i) The direct output of the network (Tables 2-5: col. (a)). ii) Logistic regression (LR) classifiers trained on DCNN-based features. Specifically, the output of each of the three last layers of each fine-tuned network was used as feature to train one LR model per concept (Tables 2-5: col. (b)-(d)). Furthermore, we present results for the late-fused output (arithmetic mean) of the LR classifiers built using the last three layers (Tables 2-5: col. (e)). For the GoogLeNet-based networks evaluations are also reported for the two auxiliary classifiers (Tables 3, 5: col. (f)-(i)). The details of the two DCNN architectures mentioned above (CaffeNet, GoogLeNet) and the extracted features are also illustrated in Fig. 2. Based on the results reported in the aforementioned tables, we reach the following conclusions:

a) According to Table 2, fine-tuning a pre-trained network on more concepts (going from 60 to 345) leads to better concept detection accuracy for all the fine-tuning strategies.

b) Across all the networks and for both datasets, the FT3-ex strategy almost always outperforms the other two fine-tuning strategies (FT1-def, FT2-re) for specific (L, E) values.

c) With respect to the direct output, FT3-ex1-64 and FT3-ex1-128 constitute the top-two methods for the TRECVID SIN dataset irrespective of the employed DCNN. On the other hand, FT3-ex1-2048 and FT3-ex1-4096 are the top-two methods for the PASCAL VOC-2012 dataset and the GoogLeNet-based networks, while FT3-ex1-512 and FT3-ex1-1024 are the best performing strategies for the CaffeNet network on the same dataset. That is, the FT3-ex strategy with one extension layer is always the best solution, but the optimal dimension of the extension layer varies, depending on the target domain dataset and the network architecture.

d) The highest concept detection accuracy for each network is always reached when LR classifiers are trained on features extracted from the last and the second last fully connected layer for TRECVID SIN and PASCAL VOC-2012 dataset,

Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavík, Iceland, Springer LNCS vol. 10132, pp. 102-114, Jan. 2017.

respectively, using the FT3-ex strategy. That is, features extracted from the top layers are more accurate than layers positioned lower in the network, but the optimal layer varies, depending on the target domain dataset.

e) DCNN-based features significantly outperform the direct output alternative in the vast majority of cases. However, in a few cases the direct network output works comparably well. The choice between the two approaches should be based on the application that the DCNN will be used. E.g., real time applications' time and memory limitations would most probably render using DCNNs as feature extractors in conjunction with additional learning (LR or SVMs) prohibitive. Furthermore, we observe that the features extracted from the final classifier of GoogLeNet-based networks outperform the other two auxiliary classifiers, in most cases.

f) Using DCNN layers' responses as feature vectors, on the one hand, FT3-ex1-512 is in the top-five methods irrespective of the employed DCNN, the extracted feature and the used dataset. Regarding the PASCAL VOC-2012 dataset this is always the case except for the features extracted from the third last layer of the CaffeNet network (Table 4: col. (d)). On the other hand, FT3-ex2-64 is always among the five worst fine-tuning methods. The rest of the FT3-ex configurations, present fluctuations of their performance across the different utilized DCNNs and DCNN-based features.

g) Finally, it is better to combine features extracted from many layers; specifically, performing late fusion on the output of the LR classifiers trained with each one of the last three fully connected layers almost always outperforms using a single such classifier irrespective of the employed network (Tables 2- 5: col. (e)). The above conclusion was also reached for the auxiliary classifiers of GoogLeNet-based networks but for space-limitations we only present the fused output for each of these auxiliary classifiers (Tables 2- 5: col. (g),(i)).

5 Conclusions

In this paper we presented a large comparative study of three fine-tuning strategies on three different pre-trained DCNNs and two different subsets of semantic concepts. Experiments performed on the TRECVID 2013 SIN dataset [10] and PASCAL VOC-2012 classification dataset [4] show that the method of increasing the depth of a pre-trained network with one fully-connected layer and fine-tuning the rest of the layers on the target dataset can improve the network's concept detection accuracy, compared to other fine-tuning approaches. Using layers' responses as feature vectors for a learning model such as logistic regression can lead to additional gains, compared to using the direct network's output, at an additional cost of computation time and memory.

Acknowledgements This work was supported by the European Commission under contract H2020-687786 InVID.

Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavík, Iceland, Springer LNCS vol. 10132, pp. 102-114, Jan. 2017.

References

1. Campos, V., Salvador, A., Giro-i Nieto, X., Jou, B.: Diving Deep into Sentiment: Understanding Fine-tuned CNNs for Visual Sentiment Prediction. In: 1st International Workshop on Affect and Sentiment in Multimedia (ASM 2015). pp. 57–62. ACM, Brisbane, Australia (2015)
2. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. In: British Machine Vision Conference (2014)
3. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: Decaf: A deep convolutional activation feature for generic visual recognition. CoRR abs/1310.1531 (2013)
4. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results
5. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Computer Vision and Pattern Recognition (CVPR 2014) (2014)
6. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T.: Caffe: Convolutional Architecture for Fast Feature Embedding. arXiv preprint arXiv:1408.5093 (2014)
7. Krizhevsky, A., Ilya, S., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems (NIPS 2012), pp. 1097–1105. Curran Associates, Inc. (2012)
8. Markatopoulou, F., et al.: ITI-CERTH participation in TRECVID 2015. In: TRECVID 2015 Workshop. NIST, USA, Gaithersburg, MD, USA (2015)
9. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: Computer Vision and Pattern Recognition (CVPR 2014) (2014)
10. Over, P., et al.: TRECVID 2013 – An Overview of the Goals, Tasks, Data, Evaluation Mechanisms and Metrics. In: TRECVID 2013. NIST, USA, Gaithersburg, MD, USA (2013)
11. Russakovsky, O., Deng, J., Su, H., et al.: ImageNet Large Scale Visual Recognition Challenge. Int. Journal of Computer Vision (IJCV) 115(3), 211–252 (2015)
12. Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., Lecun, Y.: Overfeat: Integrated recognition, localization and detection using convolutional networks (2014)
13. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv technical report (2014)
14. Snoek, C.G.M., Worring, M.: Concept-Based Video Retrieval. Foundations and Trends in Information Retrieval 2(4), 215–322 (2009)
15. Snoek, C., Fontijne, D., van de Sande, K.E., Stokman, H., et al.: Qualcomm Research and University of Amsterdam at TRECVID 2015: Recognizing Concepts, Objects, and Events in Video. In: TRECVID 2015 Workshop. NIST, USA, Gaithersburg, MD, USA (2015)
16. Szegedy, C., et al.: Going deeper with convolutions. In: Computer Vision and Pattern Recognition (CVPR 2015) (2015)
17. Yilmaz, E., Kanoulas, E., Aslam, J.A.: A simple and efficient sampling method for estimating AP and NDCG. In: 31st ACM SIGIR Int. Conf. on Research and Development in Information Retrieval. pp. 603–610. ACM, USA (2008)
18. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? CoRR abs/1411.1792 (2014)

Proc. 23rd Int. Conf. on MultiMedia Modeling (MMM'17), Reykjavík, Iceland, Springer LNCS vol. 10132, pp. 102-114, Jan. 2017.