Comprehensible Convolutional Neural Networks via Guided Concept Learning - Supplementary

Sandareka Wickramanayake
School of Computing
National University of Singapore
sandaw@comp.nus.edu.sg

Wynne Hsu
School of Computing
National University of Singapore
whsu@comp.nus.edu.sg

Mong Li Lee
School of Computing
National University of Singapore
leeml@comp.nus.edu.sg

I. Implementation details

As described in the main text, we select top-r word phrases for each class based on tf-idf scores, which correspond to the discriminative concepts of the class. In our experiments, we set $r = 20$ and obtain 398, 344, 296, and 52 word phrases for CUB, CUB-Fam, Flowers and VOC-Part datasets respectively. Algorithm 1 describes how we extract the set of discriminative word phrases. In the last section of this Appendix, we provide the discriminative word phrases we extracted for each dataset.

We set parameters as $\lambda = 0.4, \alpha = 1$ and $\beta = 0.5$. The common space vector size is set to 24 for CUB and CUB-Fam while for Flowers and VOC-Part it is set to 16 and 12 respectively.

CCNN can be implemented using the convolutional layers of different standard CNN architectures, e.g. VGG16 [1] , ResNet101 [2] or DenseNet161 [3]. Concept layer which consists of 1×1 convolutional filters is added on top of the base architecture. For example, when VGG16 is used, after `conv5_3` layer, we add the Concept layer. The Concept layer is followed by the GAP layer and a fully-connected layer. Parameters of the newly added layers are randomly initialized while other layers are initialized from weights of VGG-16 pre-trained on ImageNet [1]. We then fine-tune the CCNN using training images in the dataset. For a fair comparison, all other models are fine-tuned in the same manner. The input image size of our model is 448×448. In the training phrase we do not use part or bounding box annotations. CCNN is trained with online data augmentation. We use Tensorflow [4] for CCNN implementations and train on a NVIDIA Titan Xp GPU. The experiment results reported for CCNN are the averaged values over five trails with different random seeds.

A. Training procedure

A training instance of CCNN consists of an image, its ground truth label and indicator vector. Recall that training of CCNN also requires a counter-image for each image. To obtain a counter image, in each iteration we create a copy of the training batch. We shuffle the images in the copy before pairing the i^{th} image in the original training batch with the i^{th} image in the shuffled batch. For each concept k that is in image i but not in image i' (i.e. $z_k^i - z_k^{i'} > 0$), we calculate the counter loss.

We train CCNN in two stages. In the first stage, we fix the weights of the fully connected (FC) layer and train rest of the weights minimizing $\boldsymbol{L_A}$, $\boldsymbol{L_m}$ and $\boldsymbol{L_s}$. In the second stage, FC layer is fine-tuned by using $\boldsymbol{L_A}$ while keeping other weights fixed. Our training algorithm is given in Algorithm2. Further, following [5], we initialize weights of the FC layer such that CCNN can better learn discriminative concepts of each class. Let $W \in \mathcal{R}^{n \times C}$ be the weight matrix of the FC layer, where n is the number of concepts in the concept layer and C is the number of classes. We initialize $w_{k,c} = 0.1$ if the k^{th} concept is a discriminative concept to class c, in other words $p_k \in P_c^*$. $w_{k,c} = 0$ otherwise. Initializing weight matrix of FC layer as above forces CCNN to extract discriminative concepts related to class c from image i of class c in order to minimize classification loss. In our implementation, we use the union of the indicator vectors in class c, which is called the *class indicator vector* for class c, to initialize the FC layer as described above. The stage two training helps to further improve the classification accuracy.

II. Additional experiments

A. Ablation study

In our ablation study, we observe that introducing uniqueness loss, $\boldsymbol{L_u}$ and mapping consistency loss $\boldsymbol{L_m}$ into the training objective increases classification accuracy, see Table I. To understand why adding $\boldsymbol{L_u}$ and $\boldsymbol{L_m}$ increases discriminative power of the model, we can analyze the features of the final convolutional layer of CCNN and the model trained with only classification loss $\boldsymbol{L_A}$. Let's call the model trained with only $\boldsymbol{L_A}$ as $\boldsymbol{L_A}$-MODEL. Recall that $\boldsymbol{L_A}$-MODEL has the same architecture as CCNN. In both CCNN and $\boldsymbol{L_A}$-MODEL, the last convolutional layer features directly contribute to the model decision via GAP layer. Hence, we analyze the output of GAP layer to understand the behavior of two types of models.

We use CUB dataset in this analysis. We conduct a class wise comparison between two models. For each class we calculate the GAP layer features averaged over all the test images in that class. Figure 1 shows the averaged GAP layer features of two models for four classes, along with the classification accuracy for each class. Consider the classes $\boldsymbol{L_A}$-MODEL has achieved lower accuracy, Blue Grosbeak (36.7) and Bank Swallow (53.3). For those classes not only many features are activated but also their values are lower. This indicates that

Algorithm 1: Discriminative word phrases extraction algorithm

input : Training image descriptions - S_{D_T}, # of discriminative phrases per class - r
output: Set of discriminative word phrases - P^*

for $c \in [1, 2, ..., C]$ **do**
  $S_{D_{T,c}} \leftarrow get\_image\_descriptions\_for\_class(S_{D_T}, c)$
  for $S \in S_{D_{T,c}}$ **do**
    $W \leftarrow tokenize(S)$
    $tokens, pos\_tags \leftarrow get\_pos\_tags(W)$
    $phrases \leftarrow extract\_noun\_adjective\_pairs(tokens, pos\_tags)$
    $P_c \leftarrow P_c + phrases$
  end
  $P \leftarrow P + P_c$
end
$scores \leftarrow calculate\_tfidf(P)$
$P^* \leftarrow select\_top\_r(P, scores, r)$

Algorithm 2: Training Algorithm

input : Training dataset - D_T, Indicator vectors for D_T - Z_{D_T}, word phrase embeddings - W, λ, α, β
output: Trained weights θ

$\theta \leftarrow$ weight initialization;
▷ Initialize weights of conv layers and embedding layers
$\theta_{fc} \leftarrow$ weight initialization;
▷ Initialize weights of the fully connected (fc) layer

for $epoch \in [0, max\_epochs]$ **do**
  $D_T \leftarrow$shuffle(D_T)
  for $B \in [batch(D_T)]$ **do**
    $V, \hat{V}, logits \leftarrow$ CCNN(B)
    ▷ V -output of concept layer, $\hat{V}$- output of GAP
    $\boldsymbol{L_u} \leftarrow uniqueness\_loss(\hat{V}, Z_B)$
    $\boldsymbol{L_s} \leftarrow semantic\_loss(V, W, \alpha, Z_B)$
    $V', Z'_B \leftarrow random\_shuffle(V, Z_B)$
    $\boldsymbol{L_c} \leftarrow counter\_loss(V, V', W, \beta, Z_B, Z'_B)$
    ▷ $V'_i \in V'$ is the visual features of the counter image of $V_i \in V$
    $\boldsymbol{L_m} \leftarrow (\boldsymbol{L_s} + \boldsymbol{L_c})$
    $\boldsymbol{L_A} \leftarrow cross\_entropy\_loss(logits)$
    $\boldsymbol{L} \leftarrow \boldsymbol{L_A} + \lambda(\boldsymbol{L_u} + \boldsymbol{L_m})$
    $\theta \leftarrow Backprop(\boldsymbol{L}, \theta)$
  end
end
for $epoch \in [0, max\_epochs\_fc]$ **do**
  $D_T \leftarrow$shuffle(D_T)
  for $B \in [batch(D_T)]$ **do**
    $logits \leftarrow$ CCNN(B)
    $\boldsymbol{L_A} \leftarrow cross\_entropy\_loss(logits)$
    $\boldsymbol{L} \leftarrow \boldsymbol{L_A}$
    $\theta \leftarrow Backprop(\boldsymbol{L}, \theta_{fc})$
  end
end

TABLE I: Ablation study on CUB and CUB-Fam. Mean and standard deviation are reported over five trials with different random seeds.

Model	CUB	CUB-Fam
$\boldsymbol{L_A}$	73.3±0.2	88.2±0.1
$\boldsymbol{L_A + L_u + L_m}$	**80.1±0.5**	**90.6±0.3**

TABLE II: Classification Acc. and Association Acc. as λ varies for CUB dataset.

λ	0.1	0.2	0.3	0.4	0.5	0.6	0.8	1.0	2.0	3.0
Class. Acc.	79.9	80.0	79.8	80.1	79.7	79.4	78.9	78.7	77.9	78.1
Assoc. Acc.	82.1	83.2	83.2	84.1	83.1	82.9	82.2	82.4	81.5	75.5

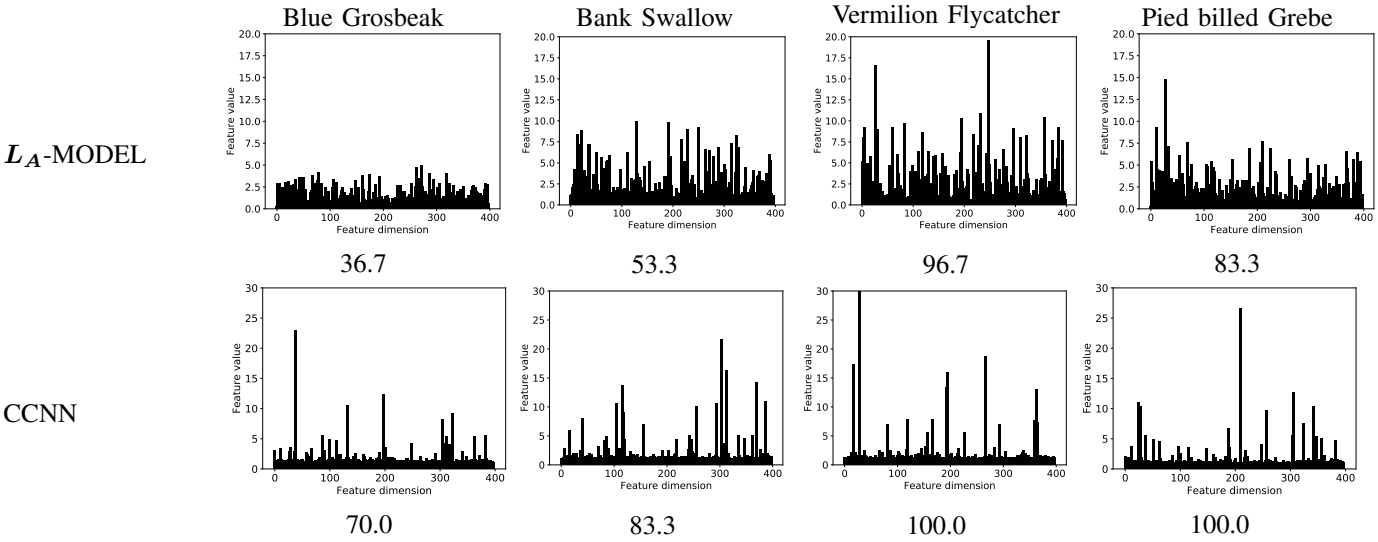

Fig. 1: The features of GAP layer averaged over all test images of the class, shown for four different classes. Classification accuracy of the model for each class is indicated under the respective graph.

TABLE III: Classification accuracy on the test sets of original, *ROAR_CCNN* and *RANDOM* versions of CUB, CUB-Fam, Flowers and VOC-Part datasets.

Dataset	CUB	CUB-Fam	Flowers	VOC-Part
Original	80.1	90.6	93.5	90.9
RANDOM	62.0	80.4	84.2	76.1
ROAR_CCNN	20.4	44.4	70.0	71.3

$\boldsymbol{L_A}$-MODEL has not learned discriminative features for these classes. In contrast, for the classes $\boldsymbol{L_A}$-MODEL has achieved higher accuracy, e.g. Vermilion Flycatcher (96.7), few features are highly activated, indicating $\boldsymbol{L_A}$-MODEL has learned to extract discriminative features for these classes.

In CCNN we observe that for every class in Figure 1 only few features have very high values compared to the rest. Further, for different classes the set of features with higher values are different. $\boldsymbol{L_u}$ and $\boldsymbol{L_m}$ cause this behavior as they encourage CCNN to activate only a subset of features for each class. We believe this characteristic of CCNN makes it easier to discriminate among classes and thereby leads to a higher overall accuracy.

B. λ hyper-parameter selection

We carry out experiments to set the value of λ in our training objective. Table II shows the classification accuracy and association accuracy as λ varies for CUB dataset. We choose $\lambda = 0.4$ as that value gives the best classification accuracy as well as best association accuracy.

C. Visual explanations for CCNN decisions

In CCNN the concepts learned in concept layer directly influence the model decision. Hence, we can combine visualizations of most contributed filters in the concept layer to derive visual explanations for the decisions made by CCNN. Given an image and its class, we rank the concepts learned in the concept layer according to their contributions to the class score and select the top most concepts until their accumulated contribution is at least 90% of the score. As described in our main paper, we create receptive fields of the feature maps corresponding to the selected concepts and take the union of receptive fields to generate the visualization indicating which region of the image influenced the model decision. We use ROAR [6] to evaluate faithfulness those visualizations. In ROAR, a fraction of image pixels estimated to be most important are replaced with the per channel mean. We then retrain CCNN on the modified dataset and measure the change in test accuracy. If visualizations highlight the most important pixels, ROAR should cause a higher accuracy degradation. In our experiment, we replace 25% of the most influential pixels and generate a new dataset, *ROAR_CCNN*. As a control study, we randomly replace 25% of the pixels and generate another new dataset, *RANDOM*. New classifiers are trained and tested using these datasets. Results given in Table III show that ROAR_CCNN have the largest drop in accuracy across multiple datasets, demonstrating that the visual explanations by CCNN are faithful to the model decision.

III. Extracted discriminative concepts

This section provides the discriminative concepts we extracted for CUB dataset.

TABLE IV: Concepts extracted for CUB dataset.

brown eyering	black neck	striped primaries	orange feet
yellow primaries	striped cheek	long tarsus	small head
red and white wingbar	black cheek	short tarsus	grey crown
orange cheek-patch	curved beak	straight beak	brown wingbar
yellow and brown belly	purple eye	long feet	webbed feet
grey and white feathers	blue side	white line	green tail
black and white back	blue wing	brown nape	brown neck
yellow and black crown	black feathers	long beak	yellow tarsus
green and yellow wing	orange leg	black and white wing	brown and black wing
white and black breast	white belly	black crown	white feet
brown and yellow breast	blue throat	black eyering	red feet
yellow secondaries	black leg	red leg	grey side
brown cheek-patch	yellow throat	green beak	brown breast
grey and yellow wing	blue face	yellow wing	grey tail
spotted head	brown belly	speckled head	orange neck
white and black feathers	red eyering	brown feathers	pink wing
red and white breast	white tip	speckled breast	blue cheek
spotted feathers	pink feet	striped breast	blue beak
grey and white wing	curved tip	red coverts	grey tarsus
speckled tail	small eyering	orange tarsus	grey face
white and black beak	blue feathers	black coverts	white cheek
brown and white belly	orange eyering	small beak	cream throat
striped eye	spotted beak	grey belly	black face
orange crown	green throat	white coverts	black tarsus
brown rectrices	white eyering	hooked beak	black tail
black rump	thin neck	yellow neck	grey primaries
white cheek-patch	yellow nape	blue crown	skinny beak
brown eye	brown pattern	red wing	green nape
grey wing	white eyebrow	large wingspan	brown head
blue belly	black primaries	blue head	yellow belly
black secondaries	curved neck	black lines	spiky crown
orange and black beak	small breast	tan breast	blue nape
white and red head	spotted crown	speckled nape	white wingbar
round head	yellow breast	black nape	blue primaries
white nape	brown side	spotted breast	pink neck
blue secondaries	yellow tip	yellow crown	green neck
orange patches	red side	speckled belly	black wingbar
brown and red wing	red throat	striped neck	large beak
white and orange beak	green belly	spotted belly	orange wing
black eyebrow	green feathers	pink beak	orange back
blue and black head	black back	small eye	pink belly
blue cheek-patch	small throat	striped belly	orange beak
red hair	red patches	thin tarsus	pink head
white and gray breast	pink leg	grey nape	black rectrices
green secondaries	cream breast	red beak	green head
green primaries	brown tail	white side	grey eyering
orange face	grey wingbar	striped nape	white tail
pink crown	brown throat	grey feathers	black throat
needlelike beak	long head	striped tail	thin beak
brown and white breast	slim beak	white breast	grey back
black and yellow wing	black head	long tail	speckled throat
blue and white beak	red eyebrow	red neck	yellow side
red and white wing	long neck	large head	red feathers
brown and white head	brown face	blue coverts	black side
yellow and black belly	brown tarsus	grey beak	yellow eye
white secondaries	large wing	orange eye	red tip
black and white primaries	pink tarsus	green breast	brown coverts
brown and white feathers	speckled back	pointed head	grey secondaries
striped eyebrow	pink breast	spotted throat	orange head
red back	orange eyebrow	striped feathers	brown rump
speckled rectrices	long wing	blue tail	thick beak
long leg	yellow rump	orange coverts	orange breast
brown crown	yellow head	brown patches	cream belly
red and yellow coverts	orange belly	red nape	brown eyebrow
white beak	white feathers	black patches	spotted wing
white patches	grey coverts	white face	yellow feathers
green and blue breast	large eye	grey rectrices	red and black beak
grey cheek-patch	red cheek-patch	long wingspan	red face
speckled feathers	long throat	pointed crown	white head
brown and black feathers	striped back	blunt beak	speckled crown
orange tip	red breast	grey feet	green wing
yellow patches	black tip	orange nape	yellow eyering
red tail	yellow and orange beak	yellow cheek-patch	black and white crown
grey breast	blue breast	red belly	spotted tail
grey throat	small wing	blue feet	red head
white rectrices	spotted side	large throat	black eye
black and yellow beak	striped beak	yellow beak	short beak
black and white tail	black feet	yellow wingbar	long feathers
black wing	conical beak	large feet	yellow cheek
brown secondaries	red eye	grey head	orange tail
green and brown wing	brown feet	white back	white throat
cream nape	red crown	small feet	long rectrices
yellow and black breast	yellow tail	grey neck	green rectrices
red and white beak	grey eye	rounded belly	purple head
triangular beak	green side	orange side	large neck
black and white head	flat beak	striped crown	striped throat
brown primaries	yellow feet	striped side	stubby beak
grey and white breast	spotted back	brown beak	pink nape
striped head	curved throat	large wingbar	red tarsus
yellow eyebrow	yellow face	speckled pattern	blue eye
speckled wing	large crown	black beak	white rump
yellow and black head	green crown	striped secondaries	brown and white wing
orange and black wing	blue back	blue rectrices	orange throat
black cheek-patch	short breast	striped face	rounded beak
black and white belly	black breast	white crown	green back
red and yellow wing	white wing	brown wing	yellow coverts
orange wingbar	blue wingbar	white eye	short tail
red and yellow wingbar	yellow back	black belly	green coverts
large breast	brown back	blue neck	white neck
black and white wingbar	striped wing		

REFERENCES

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv*, 2014.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[3] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *CVPR*, 2017.

[4] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[5] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, "This looks like that: deep learning for interpretable image recognition," in *NeurIPS*, 2019.

[6] S. Hooker, D. Erhan, P.-J. Kindermans, and B. Kim, "A benchmark for interpretability methods in deep neural networks," in *NeurIPS*, 2019.