Improving Image Understanding with Concept Graph

Libo Yin
The Australian National University

October 19, 2015

1 The Original HEX Model

1.1 Structure

The original HEX model [2] is an extension of the baseline flat multiclass classification model. There are two types of multiclass classification: exclusive, where the classifier predicts exactly one out of all possible states to be true; and independent, where each state is predicted true or false independently. HEX finds a balance between these two ends of the spectrum. It models the Hierarchical and EXclusive relationship within the concept space of dataset, extended by their hypernyms, with a semantic graphical model. Each node in the HEX graph corresponds to a concept in the extended concept space being true or false. States of neighbouring nodes are constrained in that if a is a hypernym of b, then is not allowed that $a = 0,\ b = 1$; and if a and b are exclusive, then they cannot both be true. HEX model classifies an image into a joint assignment of the states of nodes, corresponding to a hierarchy in the graphical model, that satisfies the above semantic consistency. A simple HEX graph is shown in figure 1.

Denoting the set of vertices by V, the set of hierarchical edges by E_h, and the set of exclusive edges by E_e, the joint assignment $y \in \{0,1\}^V$ can be defined as a conditional random field:

$$\tilde{p}(y|x) = \prod_{i \in V} \exp\{f_i(x;w) I[y_i = 1]\} I[y \text{ legal}] \tag{1}$$

where unary input $f_i(x;w)$ is the confidence on concept i, predicted by an arbitrary underlying classifier[1]. Local semantic constraints on hierarchical and exclusive edges is formalised by:

$$I[y \text{ legal}] = \prod_{(v_i,v_j) \in E_h} I[(y_i, y_j) \neq (0,1)] \prod_{(v_i,v_j) \in E_e} I[(y_i, y_j) \neq (1,1)] \tag{2}$$

Thanks to these semantic constraints, the state space of a HEX graph, i.e. the set of all legal joint assignments, is significantly smaller than the state space of independent multiclass classification on the same concept space. For example, with local confidence given beside each node in figure 1, the valid assignments of the HEX graph and their respective potentials are listed as follows:

Finally, an important assumption of [2] is that mechanical Turks tend to label an image to more abstract concepts. For example, an image of a yellow Labrador is more likely to be labelled to "dog" than "Labrador". Such behaviour is modelled by randomly relabelling images to their immediate parents. As will be shown in section 2.3, the higher the relabelling rate is, the larger is the advantage of the HEX model compared to the softmax baseline.

[1] According to [2]. It shall be explained in section 1.2 that the choice of underlying unary classifier is actually not arbitrary.

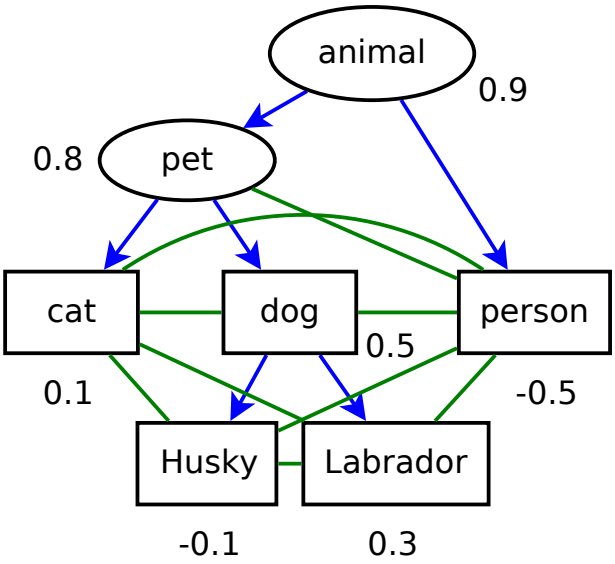

Figure 1: A simple HEX graph with three nodes in the original concept space (in rectangles) and their hypernyms (in ellipsoids). Directed edges denote semantical subsumption: $(a \to b)$ if a is a hypernym of b according to WordNet; and undirected edges denote exclusion. Since exclusive relationship is not covered by WordNet, the exclusive subgraph is initialized greedily: two concepts are exclusive unless they share a common descendant in the hierarchical subgraph. Note that the hierarchical subgraph is in general a DAG rather than a tree; and it is not necessarily the case that all concepts in the original concept space are bottom-level nodes in the hierarchical subgraph. Values beside each nodes are to be used in table 1.

assignment	$\tilde{p}(y\|x)$
$\varnothing$	$\exp(0)$
{animal}	$\exp(0.9)$
{animal, pet}	$\exp(0.9 + 0.8)$
{animal, person}	$\exp(0.9 - 0.5)$
{animal, pet, cat}	$\exp(0.9 + 0.8 + 0.1)$
{animal, pet, dog}	$\exp(0.9 + 0.8 + 0.5)$
{animal, pet, dog, Husky}	$\exp(0.9 + 0.8 + 0.5 - 0.1)$
{animal, pet, dog, Labrador}	$\exp(0.9 + 0.8 + 0.5 + 0.3)$

Table 1: The extended state space, i.e. all valid assignments of the HEX graph in figure 1, and their respective potentials. In this example, the most likely joint assignment is {animal, pet, dog, Labrador}. The softmax baseline classifier, in the language of HEX, classifies to the original state space: {animal, pet, cat}, {animal, pet, dog}, {animal, pet, dog, Husky}, {animal, pet, dog, Labrador}, and {animal, person}.

1.2 Observations

Note that a valid assignment does not have to have an active node in the original concept space. (Actually, it does not have to have an active node at all, as $\varnothing$ is a valid joint assignment according to (2).) This allows an image to be classified to a joint assignment in which all active concepts are abstract. For example, in figure 1, {$\varnothing$}, {animal}, or {animal, pet}. This is by no doubt a desirable feature for deployment. However, since all images are labelled in the original concept space, classifying to the extended concept space makes performance evaluation troublesome. While this issue is not addressed in [2], the evaluation strategy used for this work will be discussed in section 2.3.

Also note that in table 1, the competition between assignment {animal, pet, dog, Husky} and {animal, pet, dog, Labrador} depends entirely on the confidence of node "Husky" and "Labrador". However, to discriminate between {animal, pet, dog, Husky} and {animal, person} requires examining the confidence along different paths. From this example it is clear that, in the testing stage, confidence is passed down the hierarchy from more abstract concepts to more concrete ones. The other side of the same coin is that, during the training stage of the underlying unary classifier, a node corresponding to abstract concepts receive all training data of its children. This can be seen as confidence being passed up in the hierarchy, from more concrete concepts to more abstract ones. Such bidirectional propagation of confidence explains the advantage of the HEX model under the realistic labelling assumption. Note that the exclusive subgraph does not take part in the propagation of confidence.

The third observation is a combined consequence of the greedy exclusion setup, as discussed under table 1, and the greedy nature of the potential function (1). The prerequisite for the original HEX model is that the decision boundary of $f_i(x; w)$ is zero for all i: A unary prediction above zero gives support to a node being active (true), whereas a unary prediction below zero gives support to that node being inactive (false). In [2] this is not a problem, as a convolutional neural network [7] is used as the underlying unary classifier; and that the a neural network can learn its decision boundary form training data. However, if a probabilistic classifier is used, in which case $f_i(x; w) \in [0, 1]$ and the decision boudary is 0.5, then it is guaranteed that a bottom-level node will be activated (see appendix A for proof). This means that the effective state space size of the HEX graph is reduced to the number of bottom-level nodes, which is no larger than the original state space size. This defeats the purpose of HEX, although a smaller state space means faster inference and higher accuracy.

Finally, listed below are two less important observations:

1. There are no learnable variables in this CRF. In other words, all learning is performed in the underlying unary classifier.

2. Mathematically, CRF requires $\forall y : \tilde{p}(y|x) > 0$. However, computationally, assigning zero to $\tilde{p}(y|x)$ can be interpreted as assigning an infinitesimal value. Therefore, the above definition is computationally a legitimate CRF.

1.3 Problems

Consider the situation illustrated in figure 2. Following the same logic as table 1, the original HEX model predicts {animal, pet, dog, Labrador}. However, this is a result of more active nodes rather than active nodes of higher confidence. Intuitively, {animal, person} seems a more reasonable prediction. In addition, the two aforementioned assignments are only separated by 0.1 in log unnormalised potential. It will be desirable if the model can make a prediction that is further from the decision boundary by taking into account the confidence of other nodes, especially those with unary predictions far from the decision boundary.

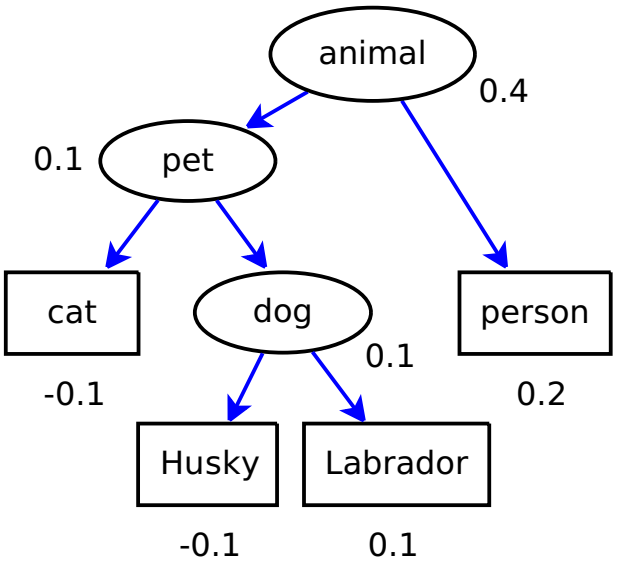

Figure 2: Where an assignment with more active nodes of lower confidence wins over an assignment with fewer active nodes of higher confidence. Exclusive edges are not drawn, as they do not carry further information once the state space has been calculated.

The second problem is a combined consequence of the realistic labelling assumption and using CNN as the underlying unary classifier. When the relabelling rate is high, most images are relabelled to their immediate parents. This means that for each concept X, most instances of X are correctly labelled to all its ancestors, but incorrectly labelled "NOT X". This create a highly unbalanced and internally inconsistent dataset. In such case, because CNN is trained as an independent multiclass classifier, the unary predictions will highly likely be "NOT X" for images of X. (This hypothesis will be confirmed in section 2.3.) The original HEX model does not cope with such situation, as the potential function (1) deactivates a bottom-level node as long as its unary confidence is below the decision boundary.

2 Reimplementation

2.1 Dataset

The original HEX model is reimplemented as the baseline of this work. While [2] used ILSVRC 2012 dataset for its rich structure in the label space, this work employs PASCAL VOC 2012 [4] for its simplicity. The complete hierarchical subgraph of HEX is shown in figure 4. Note that the two datasets are designed for different tasks: ILSVRC is an exclusive multiclass classification problem, whereas PASCAL is an independent multiclass classification problem. To adapt to ILSVRC task 1, the train+val dataset of PASCAL is filtered through the following criteria:

1. If there is only one annotated object in the image, the image is labelled to that object.
2. If the largest annotated object in the image is more than twice as large as the second largest one, the image is labelled to its dominating object.

Another major difference between ImageNet and PASCAL is that the former one is a balanced dataset, while the latter is highly unbalanced. For example, after filtering, there are 5,324 images labelled as "person", whereas the second most frequent label "dog" has only 817 instances. To rebalance the dataset, 950 images of "person" are subsampled from the filtered dataset. After preprocessing, the dataset contains 8,473 images in total. These images are then split 3:1:1 into train/validation/test set. The distribution of images across labels is illustrated in figure 3.

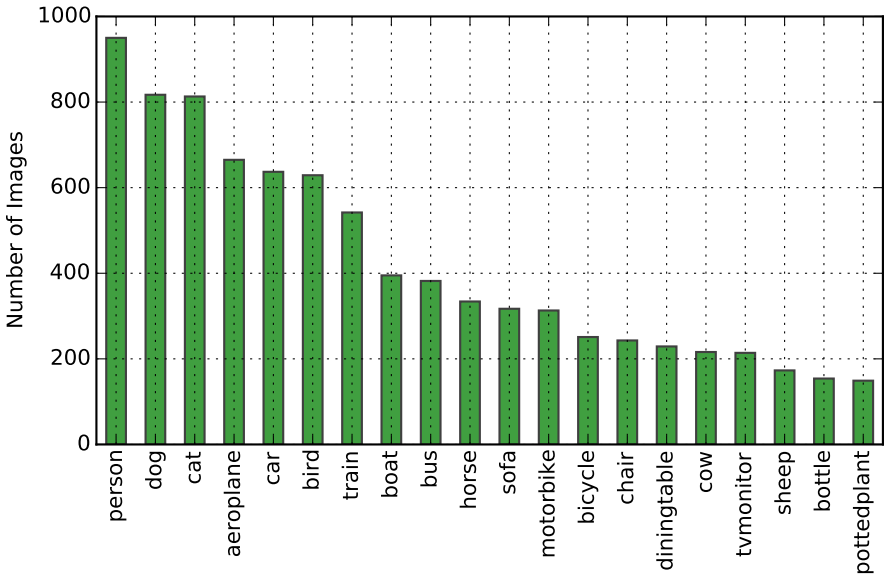

Figure 3: Distribution of images according to labels, after preprocessing.

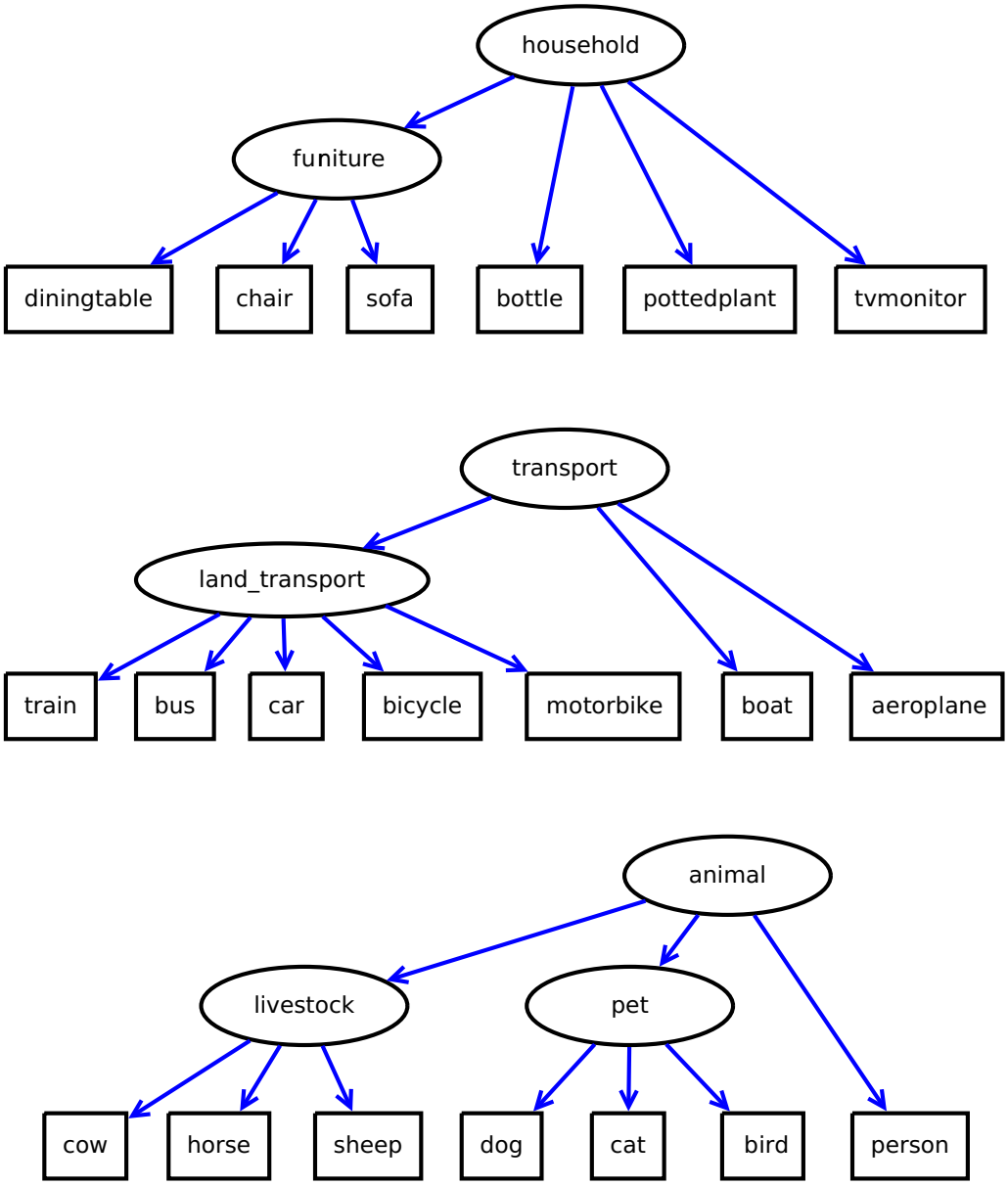

Figure 4: The PASCAL dataset has 20 concepts, as shown in rectangles. In this work, these concepts are extended by 7 of their hypernyms, creating a forest of three trees with 27 nodes and 24 hierarchical edges. Exclusive edges are not drawn since they can be implied by the hierarchical subgraph. The state space of this HEX graph has size 28. In comparison, the HEX graph in [2] contains 1000 nodes corresponding to the original concept space, and 820 nodes corresponding to their hypernyms. Note that there is no hierarchical relationship within the original concept space.

2.2 Algorithms

For consistency with [2], the underlying unary classifier is based on [7] instead of the current state-of-the-art [9]. The CNN, originally trained on ImageNet, is fine-tuned as an independent multiclass classifier on PASCAL, where each image is projected to a 27-dimensional vector. Note that since the original HEX model has no learning part, it was built as a layer into the CNN, achieving end-to-end learning. In this work, CNN and CRF are implemented separately, such that multiple variants of the HEX model can be tested with the same CNN as underlying unary classifier. For details of CNN setup, refer to appendix B.

A simpler concept space also lead to the change of inference algorithm. In [2], inference is performed by MAP loopy belief propagation on the HEX graph, where the local state space of each clique is constrained by semantic restrictions. In this work, due to the small concept space of dataset, the inference is performed by calculating the potential function directly for each state in the global state space.

Finally, to guarantee that the benefit of the HEX model is consistent regardless of the underlying unary classifier, different variants of the HEX model are also paired with SVM as the underlying unary classifier. An array of 27 SVMs is trained, each corresponding to a node in the HEX graph. The SVM array accepts the output of the last-but-one layer of [7] as input, assuming that 4,096 neurons provide sufficiently diverse and accurate feature responses *without* fine-tuning on the PASCAL dataset. The SVM array predicts either distance to decision boundaries (corresponding to raw CNN output), or the probability of each concept being activated (corresponding to CNN output with sigmoid transformation). For the detail of SVM setup, refer to appendix B.

2.3 Experiments

In this work, accuracy is tested both in the extended state space and in the original one, by limiting the legal state space to assignments with an active bottom-level node in the hierarchical subgraph; or in case of flat classification baseline, to the original concept space. Note that this is only valid for datasets where all concepts in the original concept space are bottom-level nodes in the hierarchical subgraph.

The empirical test result of [2] and reimplementation are compared in table 2. Due to the size of PASCAL dataset, relabelling rate of 95% is not attempted in this work. Observations on the result are listed as follows:

1. Regardless of implementation and inference algorithm, the accuracy drops as the relabelling rate grows. This is within expectation.

2. With the extended state space, independent multiclass classification by itself tend to have the lowest accuracy. However, with the original state space, the accuracy of independent multiclass classification can sometimes beat the softmax baseline. This is a clear sign that nodes corresponding to abstract concepts in the HEX graph receive higher confidence than bottom-level nodes. With the propagation of confidence discussed in section 1.2, the HEX model is able to recover classification information on bottom-level nodes to a similar level as the softmax baseline.

3. In [2], the original HEX model managed to beat the softmax baseline under all relabelling rates (except 0%, for which the accuracy is not provided). However, with the reimplemented system in the extended state space, the accuracy of HEX falls below the softmax baseline. In addition, as the relabelling rate grows, the accuracy of the reimplemented system drops faster than in [2]. This is a clear sign of insufficient confidence on bottom-level concepts, as discussed in section 1.3. This problem is also inspected in table 3.

	0%	50%	90%	95%
Softmax	0.626(0.843)	0.564(0.796)	0.529(0.772)	0.508(0.760)
Independent	N/A	0.210(0.452)	0.093(0.272)	0.056(0.172)
HEX	N/A	0.582(0.808)	0.553(0.794)	0.524(0.772)
Softmax	0.669(0.906)	0.332(0.816)	0.007(0.738)	N/A
Independent	0.171(0.863)	0.004(0.811)	0.000(0.407)	N/A
HEX	0.657(0.893)	0.318(0.869)	0.000(0.838)	N/A
Softmax	0.673(0.906)	0.622(0.896)	0.561(0.871)	N/A
Independent	0.726(0.937)	0.673(0.930)	0.323(0.640)	N/A
HEX	0.720(0.894)	0.685(0.888)	0.502(0.872)	N/A

Table 2: Comparison of empirical test result in [2] (top) with reimplementation (middle: extended concept space; bottom: original concept space). Performance is reported in accuracy, with top-5 accuracy in the bracket. The softmax baseline is obtained by training a CNN that maps an image to exactly one node in the HEX graph, discarding all hierarchical information.

	0%	50%	90%
diningtable	0.4285 ± 0.3011	0.2490 ± 0.2288	0.0108 ± 0.0116
furniture	0.6976 ± 0.3298	0.7524 ± 0.2784	0.7330 ± 0.3065
household	0.8431 ± 0.2729	0.8598 ± 0.2490	0.8590 ± 0.2494
dog	0.7517 ± 0.3424	0.4533 ± 0.2970	0.0222 ± 0.0209
pet	0.7971 ± 0.3271	0.7828 ± 0.3327	0.7832 ± 0.3287
animal	0.9004 ± 0.2301	0.9005 ± 0.2290	0.8999 ± 0.2322

Table 3: Under different relabelling rates, with fully trained CNN, the mean and standard deviation of sigmoid-transformed response on validation images labelled "diningtable" (upper) and "dog" (lower). Note that "household" is a hypernym of "furniture", which is further a hypernym of "diningtable"; and "animal" is a hypernym of "pet", which is further a hypernym of "dog". With 490 instances in the training set, label "dog" is a frequent concept, whereas rare concept "diningtable" has 137 instances. Clearly, the unary response on non-bottom-level concepts are not affected by the relabelling rate, yet on bottom-level concepts, the confidence drops to below the decision boundary. This experiment also shows that CNN learns highly abstract concepts (such as "household") accurately, even for concepts without visual clues.

3 Improvements

3.1 Algorithms

Based on the original HEX model, this work delivers improvements in three stages, where each stage is built on top of the previous one. The high-level goal of improvements is stated as follows: With little confidence on bottom-level nodes, the classifier should attempt to classify to an assignment with an active bottom-level concept. However, in case the classifier really cannot make a decision, it should be allowed to predict to a joint assignment in the extended concept space.

In the first stage, the confidence on inactive nodes are taken into consideration, in addition to active ones. The resulting potential function is shown as follows:

$$\tilde{p}(y|x) = \prod_{i \in V} \exp\{f_i(x;w)I[y_i = 1] + (1 - f_i(x;w))I[y_i = 0]\}I[y \text{ legal}] \tag{3}$$

where $f_i(x;w)$ denotes the confidence of node i being true, and $1 - f_i(x;w)$ for being false. This requires the range of unary prediction to be within $[0, 1]$, and the decision boundary to be 0.5. To achieve this, unary predictions from the underlying CNN is normalised with the sigmoid function.

Theoretically, stage 1 fix should benefit in two aspects: First, the potential function considers the same number of nodes for every legal assignment, therefore fixing the unnormalised depth problem discussed in section 1.3. Second, the potential function is able to refer to the confidence of inactive nodes, hence benefiting from those that are far from the decision boundary. However, a massage on (3) shows that it is not far from the original potential function (1):

$$\begin{aligned}
\tilde{p}(y|x) &= \prod_{i \in V} \exp\{f_i(x;w)I[y_i = 1] + (1 - f_i(x;w))(1 - I[y_i = 1])\}I[y \text{ legal}] \\
&= \prod_{i \in V} \exp\{(2f_i(x;w) - 1)I[y_i = 1] + (1 - f_i(x;w))\}I[y \text{ legal}] \\
&= \tilde{p}(\varnothing|x) \prod_{i \in V} \exp\{(2f_i(x;w) - 1)I[y_i = 1]\}I[y \text{ legal}]
\end{aligned} \tag{4}$$

where $\tilde{p}(\varnothing|x)$ denotes the unnormalised potential of assignment $\varnothing$. Since $\tilde{p}(\varnothing|x)$ only depends on x, it can be absorbed into $\frac{1}{Z(x)}$. Also, the transformation on $\exp\{(2f_i(x;w) - 1)I[y_i = 1]\}$ does not affect the rank of $\tilde{p}(y|x)$ for all y. Therefore, the difference between (1) and (3) is no more than the sigmoid transformation.

In the second stage, pairwise term is added to the potential function. This can be seen as a remedy to the low confidence on bottom-level nodes when the relabelling rate is high, as the confidence on the immediate parents of bottom-level nodes are not affected. In addition, two regularisation terms are added to the potential function so that the contributions from unary terms and pairwise terms are balanced. The resulting potential function is shown as follows:

$$\begin{aligned}
p(y|x) = \frac{1}{Z(x)} \exp &\left\{ \frac{1}{|V|} \sum_{i \in V} x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0] \right\} \\
&\cdot \exp \left\{ \frac{1}{|E_h|} \sum_{(i,j) \in E_h} x_i x_j \cdot I[y_i = y_j = 1] \right\} \cdot I[y \text{ legal}]
\end{aligned} \tag{5}$$

In the third stage, a weighting factor is multiplied to each unary and pairwise term. This weighting factor is learned form all images labelled to bottom-level nodes in the dataset. Note

that such training scheme cannot cover concepts from the original label space that are not bottom-level nodes in the HEX graph, as it is not possible to distinguish between images that are correctly labelled to that concept, or images that are relabelled to its immediate parents. The resulting potential function is shown as follows:

$$p_\theta(y|x) = \frac{1}{Z(x)} \exp\left\{\frac{1}{|V|}\sum_{i\in V} w_i\big(x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0]\big)\right\} \cdot \exp\left\{\frac{1}{|E_h|}\sum_{(i,j)\in E_h} t_{ij} \cdot x_i x_j \cdot I[y_i = y_j = 1]\right\} \cdot I[y \text{ legal}] \tag{6}$$

where θ is the concatenation of $\{\forall i \in V : w_i\}$ and $\forall (i, j) \in E_h : t_{ij}$. θ is selected by optimising for log-likelihood:

$$\theta = \arg\min_\theta \left\{-\frac{C}{N}\log\prod_{(x,y)\in D} p_\theta(y|x) + \frac{1}{2}\|\theta\|^2\right\} \tag{7}$$

where the weighting of regularisation term $\frac{1}{2}\|\theta\|^2$ is controlled by C, a constant chosen by cross-validation. In this work, $C = 1000$. The gradient is calculated piecewise:

$$\begin{aligned}\nabla_\theta \log\prod_{(x,y)\in D} p_\theta(y|x) &= \begin{bmatrix}\nabla_w \log\prod_{(x,y)\in D} p_\theta(y|x)\\ \nabla_t \log\prod_{(x,y)\in D} p_\theta(y|x)\end{bmatrix}\\ &= \begin{bmatrix}\sum_{(x,y)\in D} \nabla_w \log p_\theta(y|x)\\ \sum_{(x,y)\in D} \nabla_t \log p_\theta(y|x)\end{bmatrix}\end{aligned} \tag{8}$$

The log-likelihood in (8) is expanded as follows:

$$\begin{aligned}\log p_\theta(y|x) = &\frac{1}{|V|}\sum_{i\in V} w_i(x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0])\\ &+ \frac{1}{|E_h|}\sum_{(i,j)\in E_h} t_{ij} \cdot x_i x_j \cdot I[y_i = y_j = 1] - \log\sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x)\end{aligned} \tag{9}$$

Therefore, the first line of (8) is calculated as:

$$\begin{aligned}\nabla_w \log p_\theta(y|x) &= \frac{1}{|V|}\Big[x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0]\Big]_{i\in V} - \nabla_w \log\sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x)\\ &\quad \text{define } \phi_u(x, y) = \frac{1}{|V|}\Big[x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0]\Big]_{i\in V}\\ &= \phi_u(x, y) - \sum_{\hat{y}} p_\theta(\hat{y}|x) \cdot \phi_u(x, \hat{y})\end{aligned} \tag{10}$$

where the gradient of the partition function in (10) is calculated as:

$$\begin{aligned}
\nabla_w \log \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) &= \frac{1}{\sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x)} \nabla_w \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) \\
&= \frac{1}{Z(x)} \sum_{\hat{y}} \nabla_w \exp\left\{ \frac{1}{|V|} \sum_{i \in V} w_i \big(x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0] \big) \right\} \\
&\quad \cdot \exp\left\{ \frac{1}{|E_h|} \sum_{(i,j) \in E_h} t_{ij} \cdot x_i x_j \cdot I[y_i = y_j = 1] \right\} \\
&= \frac{1}{Z(x)} \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) \cdot \frac{1}{|V|} \Big[x_i \cdot I[\hat{y}_i = 1] + (1 - x_i) \cdot I[\hat{y}_i = 0] \Big]_{i \in V} \\
&= \sum_{\hat{y}} p_\theta(\hat{y}|x) \cdot \phi_u(x, \hat{y}) \qquad (11)
\end{aligned}$$

Similarly, the second line of (8) is calculated as:

$$\begin{aligned}
\nabla_t \log p_\theta(y|x) &= \underbrace{\frac{1}{|E_h|} \Big[t_{ij} \cdot x_i x_j \cdot I[y_i = y_j = 1] \Big]_{(i,j) \in E_h}}_{\phi_t(x,y)} - \nabla_t \log \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) \\
&= \phi_t(x, y) - \sum_{\hat{y}} p_\theta(\hat{y}|x) \cdot \phi_t(x, \hat{y}) \qquad (12)
\end{aligned}$$

where the gradient of the partition function in (12) is calculated as:

$$\begin{aligned}
\nabla_t \log \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) &= \frac{1}{\sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x)} \nabla_t \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) \\
&= \frac{1}{Z(x)} \sum_{\hat{y}} \nabla_t \exp\left\{ \frac{1}{|V|} \sum_{i \in V} w_i \big(x_i \cdot I[y_i = 1] + (1 - x_i) \cdot I[y_i = 0] \big) \right\} \\
&\quad \cdot \exp\left\{ \frac{1}{|E_h|} \sum_{(i,j) \in E_h} t_{ij} \cdot x_i x_j \cdot I[y_i = y_j = 1] \right\} \\
&= \frac{1}{Z(x)} \sum_{\hat{y}} \tilde{p}_\theta(\hat{y}|x) \cdot \frac{1}{|E_h|} \Big[t_{ij} \cdot x_i x_j \cdot I[\hat{y}_i = \hat{y}_j = 1] \Big]_{(i,j) \in E_h} \\
&= \sum_{\hat{y}} p_\theta(\hat{y}|x) \cdot \phi_t(x, \hat{y}) \qquad (13)
\end{aligned}$$

The resulting derivation agrees with the general property that the gradient of the log potential of CRF is the expectation of a feature function. Note that since weights represent the credibility of unary confidence, all entries of θ are non-negative.

3.2 Experiments

The summary of accuracy is shown in table 4.

Stage 1: enhanced numerical stability with low confidence???

State 2: Original relationships are weak. A promising direction is to emphasize on more relationships

Stage 3: It has been noticed that some weights are set to zero by optimisation. CRF is particularly useful for small training dataset, where confidence on bottom-level nodes are low.

	0%	50%	90%
Softmax	0.6698(0.8580)	0.3325(0.7161)	0.0071(0.5635)
Independent	0.1716(0.7339)	0.0047(0.6258)	0.0000(0.1728)
Original HEX	0.6573(0.7779)	0.3182(0.7193)	0.0000(0.4774)
Stage 1	0.6579(0.7790)	0.3182(0.7197)	0.0000(0.5154)
Stage 2	0.6923(0.7969)	0.4631(0.7470)	0.0005(0.5908)
Stage 3	0.6882(0.7802)	0.4643(0.6496)	0.1454(0.3646)
Softmax	0.6739(0.8580)	0.6223(0.8301)	0.5617(0.7909)
Independent	0.7268(0.8889)	0.6739(0.8705)	0.3230(0.5385)
Original HEX	0.7209(0.8788)	0.6852(0.8574)	0.5029(0.7838)
Stage 1	0.7214(0.8622)	0.6799(0.8337)	0.5000(0.7850)
Stage 2	0.7238(0.7957)	0.6252(0.7470)	0.3129(0.5908)

Table 4: extended (upper), original (lower). Performance reported in top1 accuracy (top 3 accuracy, due to smaller state space). Signs of overtrainning of CRF on the training set, but the benefit soon takes over as relabelling rate grows. Stage 3 only apply to the extended concept space.

Concept	Softmax	Original	Stage 2	Stage 3	2-Original	3-2
diningtable	0.2321	0.3571	0.4821	0.4642	0.1250	-0.0179
chair	0.0888	0.0666	0.1333	0.1111	0.0667	-0.0222
sofa	0.1206	0.0689	0.2068	0.1379	0.1379	-0.0689
bottle	0.3000	0.1666	0.2666	0.3000	0.1000	0.0334
pottedplant	0.1071	0.0357	0.0357	0.1071	0.0000	0.0714
tvmonitor	0.5250	0.4500	0.4750	0.4750	0.0250	0.0000
train	0.1583	0.5000	0.6583	0.5666	0.1583	-0.0917
bus	0.0294	0.5000	0.6911	0.8235	0.1911	0.1324
car	0.0703	0.3125	0.4921	0.4531	0.1796	-0.039
bicycle	0.1481	0.3333	0.4629	0.8518	0.1296	0.3889
motorbike	0.3214	0.3750	0.5000	0.4821	0.1250	-0.0179
aeroplane	0.5034	0.3586	0.6000	0.4551	0.2414	-0.1449
boat	0.2602	0.1780	0.3150	0.2739	0.1370	-0.0411
cow	0.1153	0.0384	0.1153	0.0576	0.0769	-0.0577
horse	0.1304	0.1304	0.2028	0.6956	0.0724	0.4928
sheep	0.2352	0.1470	0.2352	0.5000	0.0882	0.2648
dog	0.6363	0.3863	0.5738	0.5056	0.1875	-0.0682
cat	0.4137	0.2275	0.3724	0.3241	0.1449	-0.0483
bird	0.5000	0.4056	0.5660	0.5566	0.1604	-0.0094
person	0.5373	0.4328	0.5572	0.5373	0.1244	-0.0199
household	N/A	0.7859	0.7743	0.7626	-0.0116	-0.0117
furniture	N/A	0.6729	0.6981	0.6729	0.0252	-0.0252
transport	N/A	0.9534	0.9565	0.9549	0.0031	-0.0016
landtransport	N/A	0.9201	0.9413	0.9389	0.0212	-0.0024
animal	N/A	0.8825	0.8786	0.8735	-0.0039	-0.0051
livestock	N/A	0.5419	0.6129	0.6967	0.071	0.0838
pet	N/A	0.8337	0.8548	0.8548	0.0211	0.0000

Table 5: Extended state space, 50% relabelling rate

Node	w_i	Edge	t_{ij}
diningtable	2.1692	(household, bottle)	0.9686
chair	1.4113	(household, pottedplant)	1.0023
sofa	2.1080	(household, tvmonitor)	0.9444
bottle	0.6178	(household, furniture)	0.4723
pottedplant	0.8744	(furniture, diningtable)	0.9642
tvmonitor	0.4812	(furniture, chair)	0.9579
train	2.0026	(furniture, sofa)	0.9367
bus	0.0000	(transport, aeroplane)	0.8654
car	1.0358	(transport, boat)	0.9185
bicycle	0.0000	(transport, landtransport)	2.2961
motorbike	2.1447	(landtransport, train)	0.8297
aeroplane	2.0975	(landtransport, bus)	1.0487
boat	1.5493	(landtransport, car)	0.8752
cow	1.5618	(landtransport, bicycle)	0.9760
horse	0.0000	(landtransport, motorbike)	0.9074
sheep	0.0000	(animal, person)	0.8808
dog	1.8704	(animal, livestock)	0.5204
cat	1.9724	(animal, pet)	1.8062
bird	3.7825	(livestock, cow)	0.9623
person	1.3138	(livestock, horse)	0.9450
household	0.6735	(livestock, sheep)	0.9741
furniture	1.7698	(pet, dog)	0.7774
transport	0.1756	(pet, cat)	0.8273
landtransport	2.9717	(pet, bird)	3.2935
animal	2.6498		
livestock	0.0000		
pet	4.0644		

Table 6: 0% relabelling rate, learned weights

Node	w_i	Edge	t_{ij}
diningtable	2.1832	(household, bottle)	0.9846
chair	1.3210	(household, pottedplant)	1.0215
sofa	2.1596	(household, tvmonitor)	0.9575
bottle	0.6195	(household, furniture)	0.5110
pottedplant	0.9273	(furniture, diningtable)	0.9699
tvmonitor	0.7008	(furniture, chair)	0.9891
train	2.1006	(furniture, sofa)	0.9622
bus	0.0000	(transport, aeroplane)	0.8935
car	1.1823	(transport, boat)	0.9540
bicycle	0.0000	(transport, landtransport)	2.2119
motorbike	2.1935	(landtransport, train)	0.8858
aeroplane	2.1355	(landtransport, bus)	1.1213
boat	1.7257	(landtransport, car)	0.9193
cow	1.5908	(landtransport, bicycle)	0.9812
horse	0.0000	(landtransport, motorbike)	0.9335
sheep	0.0000	(animal, person)	0.9133
dog	2.0003	(animal, livestock)	0.5164
cat	2.0752	(animal, pet)	1.7930
bird	2.6603	(livestock, cow)	0.9867
person	1.3928	(livestock, horse)	0.9645
household	0.7853	(livestock, sheep)	0.9828
furniture	1.6832	(pet, dog)	0.8477
transport	0.3454	(pet, cat)	0.8884
landtransport	2.9475	(pet, bird)	2.7109
animal	2.4981		
livestock	0.0000		
pet	3.9751		

Table 7: 50% relabelling rate, learned weights

Node	w_i	Edge	t_{ij}
diningtable	2.2378	(household, bottle)	0.9974
chair	1.2302	(household, pottedplant)	1.0097
sofa	2.2351	(household, tvmonitor)	0.9947
bottle	0.9062	(household, furniture)	0.5250
pottedplant	0.8470	(furniture, diningtable)	0.9981
tvmonitor	0.5482	(furniture, chair)	1.0003
train	2.2922	(furniture, sofa)	0.9981
bus	0.0000	(transport, aeroplane)	0.9963
car	1.2010	(transport, boat)	0.9972
bicycle	0.0000	(transport, landtransport)	2.4417
motorbike	2.2931	(landtransport, train)	0.9977
aeroplane	2.3083	(landtransport, bus)	1.0210
boat	1.7288	(landtransport, car)	1.0055
cow	1.5191	(landtransport, bicycle)	1.0009
horse	0.0000	(landtransport, motorbike)	0.9972
sheep	0.0000	(animal, person)	0.9949
dog	2.2625	(animal, livestock)	0.5153
cat	2.2553	(animal, pet)	1.9839
bird	0.0000	(livestock, cow)	0.9993
person	1.6450	(livestock, horse)	0.9993
household	0.8827	(livestock, sheep)	0.9997
furniture	1.8698	(pet, dog)	0.9897
transport	0.9849	(pet, cat)	0.9816
landtransport	3.3086	(pet, bird)	1.3539
animal	3.0661		
livestock	0.0000		
pet	4.0960		

Table 8: 90% relabelling rate, learned weights

Node	0%	50%	90%	90%-0%
diningtable	2.1692	2.1832	2.2378	0.0686
chair	1.4113	1.3210	1.2302	-0.1811
sofa	2.1080	2.1596	2.2351	0.1271
bottle	0.6178	0.6195	0.9062	0.2884
pottedplant	0.8744	0.9273	0.8470	-0.0274
tvmonitor	0.4812	0.7008	0.5482	0.0670
train	2.0026	2.1006	2.2922	0.2896
bus	0.0000	0.0000	0.0000	0.0000
car	1.0358	1.1823	1.2010	0.1652
bicycle	0.0000	0.0000	0.0000	0.0000
motorbike	2.1447	2.1935	2.2931	0.1484
aeroplane	2.0975	2.1355	2.3083	0.2108
boat	1.5493	1.7257	1.7288	0.1795
cow	1.5618	1.5908	1.5191	-0.0427
horse	0.0000	0.0000	0.0000	0.0000
sheep	0.0000	0.0000	0.0000	0.0000
dog	1.8704	2.0003	2.2625	0.3921
cat	1.9724	2.0752	2.2553	0.2829
bird	3.7825	2.6603	0.0000	-3.7825
person	1.3138	1.3928	1.6450	0.3312
household	0.6735	0.7853	0.8827	0.2092
furniture	1.7698	1.6832	1.8698	0.1000
transport	0.1756	0.3454	0.9849	0.8093
landtransport	2.9717	2.9475	3.3086	0.3369
animal	2.6498	2.4981	3.0661	0.4163
livestock	0.0000	0.0000	0.0000	0.0000
pet	4.0644	3.9751	4.0960	0.0316

Table 9: Learned unary weights

Edge	0%	50%	90%	90%-0%
(household, bottle)	0.9686	0.9846	0.9974	0.0288
(household, pottedplant)	1.0023	1.0215	1.0097	0.0074
(household, tvmonitor)	0.9444	0.9575	0.9947	0.0503
(household, furniture)	0.4723	0.5110	0.5250	0.0527
(furniture, diningtable)	0.9642	0.9699	0.9981	0.0339
(furniture, chair)	0.9579	0.9891	1.0003	0.0424
(furniture, sofa)	0.9367	0.9622	0.9981	0.0614
(transport, aeroplane)	0.8654	0.8935	0.9963	0.1309
(transport, boat)	0.9185	0.9540	0.9972	0.0787
(transport, landtransport)	2.2961	2.2119	2.4417	0.1456
(landtransport, train)	0.8297	0.8858	0.9977	0.1680
(landtransport, bus)	1.0487	1.1213	1.0210	-0.0277
(landtransport, car)	0.8752	0.9193	1.0055	0.1303
(landtransport, bicycle)	0.9760	0.9812	1.0009	0.0249
(landtransport, motorbike)	0.9074	0.9335	0.9972	0.0898
(animal, person)	0.8808	0.9133	0.9949	0.1141
(animal, livestock)	0.5204	0.5164	0.5153	-0.0051
(animal, pet)	1.8062	1.7930	1.9839	0.1777
(livestock, cow)	0.9623	0.9867	0.9993	0.0370
(livestock, horse)	0.9450	0.9645	0.9993	0.0543
(livestock, sheep)	0.9741	0.9828	0.9997	0.0256
(pet, dog)	0.7774	0.8477	0.9897	0.2123
(pet, cat)	0.8273	0.8884	0.9816	0.1543
(pet, bird)	3.2935	2.7109	1.3539	-1.9396

Table 10: Learned pairwise weights

4 Discussions

4.1 Scalability

Scalable until stage 2 with MAP LBP. Stage 3 is not scalable due to the brute force calculation of partition function. [download large hex graph from pHEX paper, see state space size]

4.2 Relationship to Probabilistic HEX

Afte the ECCV 14 paper, the original authors extended HEX to pHEX by relaxing 0/1 hard constraints to $u \in (0, 1)$, and turning the HEX graph into an Ising model [3]. pHEX managed to beat HEX by [TODO]. However, the crucial u is chosen by cross-validation, and it still applies to all constraints in the graph. In addition, pHEX require a long time to train, although still much shorter than training the underlying CNN. Also, it uses the same potential function as original HEX. Compared to pHEX, this work focuses on using a more flexible potential function. In term of running time, these two works are not directly comparable since this work is not scalable.

A Original HEX Fails with Positive Unary Inputs

Denote the set of active nodes in a state by $\mathcal{S}$. Then there are three cases:

1. Assume there exists exactly one node t, such that $\forall s \in \mathcal{S} : s \to t$. In other words, t is the only node among all active ones with no out-edges. If t has children, then labelling any of them as true improves (1).

2. Assume there exists two active nodes with no out-edges. Denote them by t_1 and t_2. Then they must share a common descendant, as otherwise they would be connected by an exclusive edge.

   (a) If they share a common immediate child c, then labelling c as true improves (1).

   (b) If they only share a distant descendant, then the state of t_1 and t_2 are independent. The problem is thus equivalent to two case 1 in parallel.

3. There exists more than two active nodes with no out-edges. then each pair of them must have a common descendant, and the case is equivalent to pairwise case 2.

Hence, it is guaranteed that for positive unary inputs, the original HEX labels to a node in the original concept space.

B CNN and SVM Setup

The fine-tuning of the CNN from [7] is implemented using Caffe [6]. The final layer of the network is shrunk from 1000 neurons to 27, each corresponding to a node in figure 4. Since the CNN is fine-tuned as an independent multiclass classifier, the output from the last layer is transformed with the sigmoid function; and the loss is calculated as the Euclidean distance between prediction and ground truth.

The global learning rate is set to 5×10^{-5}. This is significantly lower than in [7], as all layers except the last are assumed well-trained. The global learning rate decreases linearly by 10^{-5} every 5,000 iterations; and the training process terminates after 50,000 iterations. The learning rate of the last layer is set to 5 times the global learning rate. A snapshot is taken every 5,000 iterations, allowing 10 models for selection. With a GeForce GTX 470 graphical card, the fine-tuning process finished within five hours.

The SVM unary classifier is implemented using `LibSVM` [1]. Attempted kernels are linear, polynomial (degree=3), and RBF. The training process for the SVM array finished within 8 hours without parallel training.

C Attributed HEX

In earlier stages of this project, one of the proposed directions was the joint analysis of concept and attributes. That is, for example, to classify an image of a yellow Labrador into "animal, pet, dog, yellow, furry", etc. An example HEX graph is shown as follows:

While the training data is provided by aPascal & aYahoo dataset [5]. [TODO: discuss dataset properties] The possibility of using CNN as feature extractor has been confirmed in [8].

We only connected attributes to bottom-level concepts. In other words, the aHEX graph can be seen as a semantic part and an attribute part. Such design is for the sake of reducing loops in the aHEX graph. While junction-tree algorithm could potentially handle such a loopy graph during inference stage, the loops makes the graph very tricky to learn as a CRF. Similar design has been applied to pixel labelling problem such as [TODO: find references].

However, it was not the loops that caused such idea to be dropped. Under the same non-learning model, potential function is submodular, therefore such extension s trivial. Under the learnable CRF model, the aHEX graph can be learned part-wise: the semantic subgraph can be learned in the same way as discussed in [TODO: cross-ref learnable CRF], whereas for the attributes part, all bottom-level concepts can be grouped into a super-concept with 20 states (corresponding to 20 PASCAL labels), then the attribute graph becomes a tree, which is learnable[TODO: find references]. With such model, the inference is the same as an non-learning system, as the potential function for the attributed part still has a submodular structure.

References

[1] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

[2] Jia Deng, Nan Ding, Yangqing Jia, Andrea Frome, Kevin Murphy, Samy Bengio, Yuan Li, Hartmut Neven, and Hartwig Adam. Large-scale object classification using label relation graphs. In *Computer Vision–ECCV 2014*, pages 48–64. Springer, 2014.

[3] Nan Ding, Jia Deng, Kevin Murphy, and Hartmut Neven. Probabilistic label relation graphs with ising models. *arXiv preprint arXiv:1503.01428*, 2015.

[4] 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.

[5] Alireza Farhadi, Ian Endres, Derek Hoiem, and David Forsyth. Describing objects by their attributes. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1778–1785. IEEE, 2009.

[6] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

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

[8] Ali S Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.

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