Improving Image Understanding with Concept Graph

Libo Yin
The Australian National University

October 12, 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 between concepts, extended by their hypernyms, with a semantic graphical model. Each node in this graphical model 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 hierarchy that satisfy the above semantic consistency. A simple HEX graph is shown in figure 1.

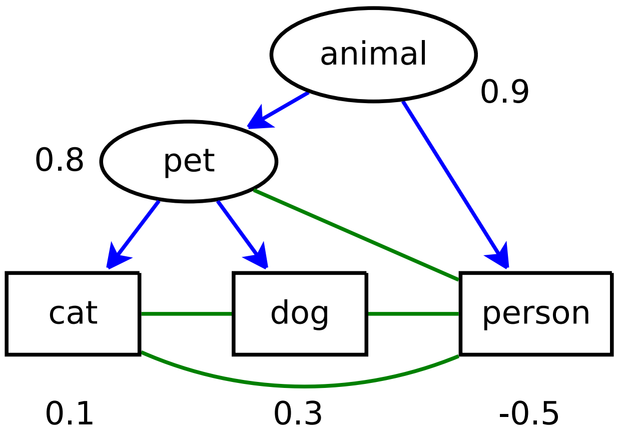

Figure 1: A simple HEX graph with three nodes in the original concept space (denoted by rectangles) and their hypernyms (in ellipsoids). Directed edges denote semantical hierarchy: $(a \to b)$ if a is a hypernym of b according to WordNet; and undirected edges denote exclusion. Since exclusive relationship 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.

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 CRF:

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

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 is significantly smaller than independent multiclass classification on the same concept space. For example, with local confidence given beside each node in figure 1, the valid states of the HEX graph and their respective potentials are listed in table 1:

state	$\tilde{p}(y\|x)$
$\varnothing$	$\exp(0)$
{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}	$\exp(0.9 + 0.8 + 0.3)$

Table 1: Valid states of the HEX graph in figure 1 and their respective potentials. In this example, the HEX model predicts {animal, pet, dog}.

Finally, an important assumption of [2] is that mechanical turks tend to label an image to more general concepts. For example, an image of a yellow Labrador may be labelled to "dog". Such behaviour is modelled by randomly relabelling an image to its immediate parents. With a high relabelling rate, the underlying unary classifiers for concepts in the original concept space are undertrained, whereas those for abstract concepts remain well-trained. As will be shown in section 4, the higher the relabelling rate is, the further is the advantage with the HEX model compared to the softmax baseline.

1.2 Observations

Note that a valid state 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 state according to (2).) This allows an image to be classified to abstract concepts when the classifier is not confident enough to classify to a more concrete one. This is by no doubt a desirable feature for deployment. However, since all images are labelled in the original concept space of the dataset, classifying to the extended concept space makes performance evaluation troublesome. This issue was not addressed in [2]. In this work, accuracy is tested both in the extended concept space and the original one, by limiting the state space to states with an active node in the original concept space.

Also note that in table 1, the competition between state {animal, pet, cat} and {animal, pet, dog} depends entirely on the confidence of node "dog" and "cat". However, to discriminate

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

between {animal, pet, dog} 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 abstract concepts to concrete ones. The other side of the same coin is that, during the training stage of the underlying unary classifier, nodes corresponding to abstract concepts receive more training data compared to nodes corresponding to concrete concepts. This can be seen as confidence being passed up in the hierarchy. This explains the advantage of the HEX model under the realistic labelling assumption.

The third observation is a combined consequence of the greedy exclusion setup, as discussed in the caption of table 1, and the greedy nature of the potential function (1). The prerequisite for the original HEX model to work is that the decision boundary of $f_i(x; w)$ is zero for all i. This is not a problem in [2], where a convolutional neural network [7] is used as the underlying unary classifier. 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 is reduced to the that of the original concept space. In other words, the HEX model will not be able to predict to an abstract concept. This defeats the purpose of HEX, although inference can be performed easily with brute force calculation of the potential function (1).

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 equivalent to a legitimate CRF.

1.3 Problems

Consider the situation illustrated in figure 2:

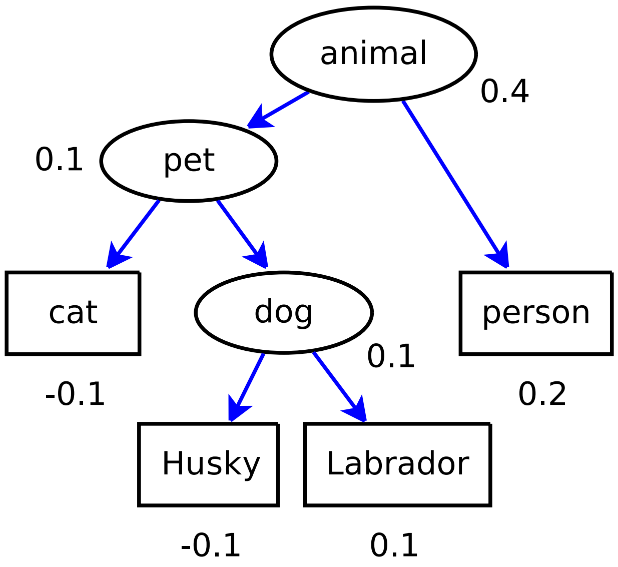

Figure 2: Where a state with more active nodes of lower confidence wins over a state with fewer active nodes of higher confidence. Exclusive edges are omitted, as they do not carry further information once the state space has been calculated.

Following the same logic as in 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 better prediction. In addition, the two aforementioned states are very close to the decision boundary between them. It will be desirable if the classifier can make a prediction further from the decision boundary by taking into account the confidence of other nodes, especially those with unary predictions far from the decision boundary.

The second problem is a direct consequence of the realistic labelling assumption. When the relabelling rate is high, with undertrained classifiers, the confidence for bottom-level nodes tend to be below the decision boundary. This is simply due to the highly unbalanced dataset. For example, if most Labrador images are labelled as {animal, pet, dog, NOT Labrador}. In such case, the underlying unary classifier predicts to below the decision boundary (confidently not), instead of around the decision boundary (unsure). The original HEX model cannot cope with such situation, and will deactivate the bottom-level concept.

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 label space, this work employed PASCAL VOC 2012 [4] for its simplicity (see figure 3 for the complete hierarchical subgraph). 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 the PASCAL dataset to ILSVRC task 1, the train + val dataset 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.

Figure 3: The PASCAL dataset has 20 concepts. These concepts are augmented by 7 of their hypernyms. The HEX graph therefore contains 27 nodes, with 24 hierarchical edges. Note that the hierarchical subgraph is a forest of three trees. In comparison, the HEX graph for ImageNet contains 1000 nodes corresponding to the original concept space, and 840 nodes corresponding to their hypernyms.

Another major difference between ImageNet and PASCAL is that the former one is a balanced dataset, while the latter is unbalanced. For example, after filtering, there are 5324 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 dataset. After preprocessing, the dataset contains 8473 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 4.

2.2 Algorithms

For consistency with [2], the underlying unary classifier is based on [7] rather than the current state-of-the-art [9]. The convolutional neural network is fine-tuned as an independent multiclass classifier with Caffe [6]. 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

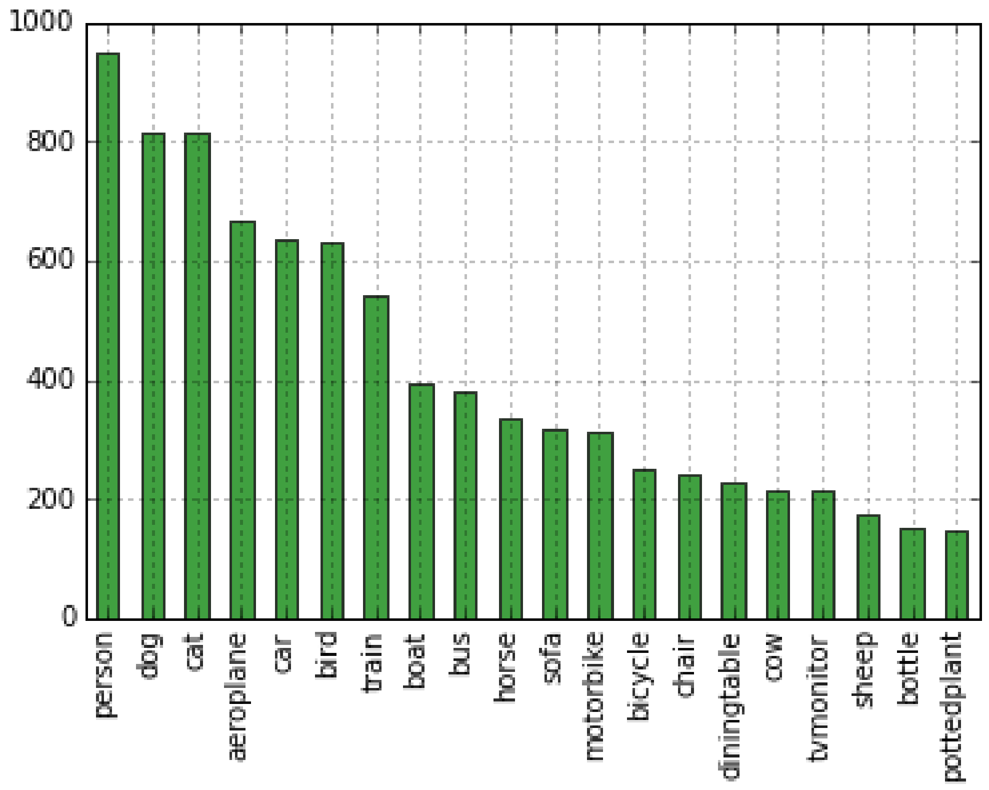

Figure 4: Distribution of images according to labels, after preprocessing

implemented separately in order to test multiple variations of the HEX model with the same underlying unary classifier. The fine-tuning process finished within five hours on a GeForce GTX 470 graphical card.

A consequence of the simpler concept space is the change of inference algorithm. In [2], inference is performed by MAP loopy belief propagation on the maximally sparsified HEX graph, where the local state space of each clique is limited by semantic restrictions. In this work, thanks to the tiny state space of PASCAL, the inference is performed by calculating the potential function directly for each possible states in the global state space.

Finally, to guarantee that the benefit of the HEX model is consistent regardless of the underlying unary classifier, different variations 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 extended concept space. The SVM array accepts the output of the last-but-one layer of [7] as input, assuming that 4096 neurons provide sufficiently diverse and accurate feature responses without fine-tuning on the PASCAL dataset. The SVM array predicts either distance to decision boundaries or the probability of each concept being activated. The training process for the SVM array finished within ten hours without parallel training.

3 Improvements

Based on the original HEX model, this work delivers improvements in three stages. In the first stage, the confidence on inactive nodes is taken into account, in addition to active ones. The resulting potential function is shown below:

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

This fix brings two benefits: First, the potential function considers the same number of nodes for different states, and therefore fixing the unnormalised depth problem mentioned in section 1.3. Second, the potential function is able to refer to other nodes, benefiting from those that are far from the decision boundary. [std dev on cnn output on bottom-level nodes vs internal ones]

In the second stage, the unsure bottom-level nodes problem is solve by taking into account the pairwise terms, especially the hierarchical edges that connect the bottom-level nodes and their immediate parents. Two regularisation terms are also added to the potential function so that the contributions from nodes and edges are balanced.

In the third stage, Enable learning on unary and pairwise term weighting. With little confidence on the bottom layer classifiers, the problem is to attempt to classify to the bottom layers. However, in case the classifier really cannot make a decision, it should be allowed to stop at an intermediate layer. In order to do so, the CRF is trained with all images that are labelled to bottom-level concepts.

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

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

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

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

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

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

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

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

4 Experiments

Stage 1: The problem is that, as discussed in section 2.1... so the pnHEX model holds back conservatively?

Stage 3: It has been noticed that some weights are set to zero by optimisation.

5 Discussions

5.1 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

Scheme	Setup
1	Softmax with subset of training data labeled to leaf nodes
2	Softmax with complete training data
3	Original CRF
4	Sigmoid + pn CRF
4	Sigmoid + pn CRF with pairwise term
6	Sigmoid + learnable CRF

Scheme	0%	50%	90%
1	n/a	0.TODO	0.TODO
2	0.1716/0.7268	0.0047/0.6739	0.0000/0.3230
3	0.6573/0.7209	0.3182/0.6852	0.0000/0.5029
4	0.6579/0.7214	0.3182/0.6799	0.0000/0.5000
5	0.6923/0.7238	0.4631/0.6252	0.0005/0.3129
6	0.6852	0.4643	0.1454

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.

5.2 Scalability

Scalable until stage 2. 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]

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 $\forall s \in \mathcal{S} : s \rightarrow t$. 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 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] Alfred V. Aho, John E. Hopcroft, and Jeffrey Ullman. *Data Structures and Algorithms*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1983.

[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.