Improving Image Understanding with Concept Graph

Libo Yin
The Australian National University

October 9, 2015

1 The Original HEX Model

1.1 Structure, or what the original paper said

The original HEX model [1] 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 assigned 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 it is not possible 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.

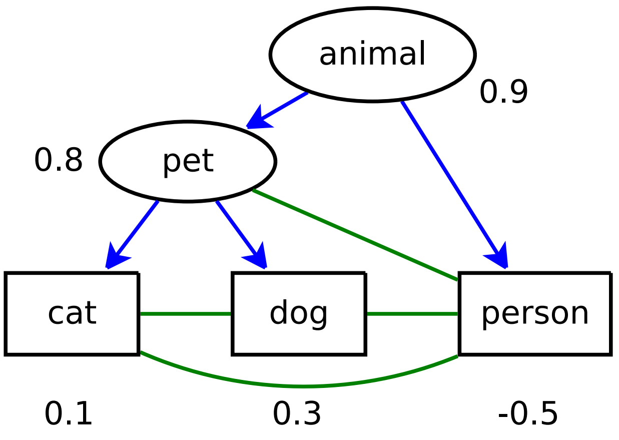

Figure 1: A simple HEX graph with three nodes in the original concept space (denoted by rectangles) and their hypernyms (ellipsoids). Directed edges denote semantical hierarchy: $(a \to b)$ if a is a hypernym of b according to WordNet; and undirected edges denote exclusion. Since exclusive relationship 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_e, and the set of exclusive edges by E_h, 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 constraint 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 much smaller than independent multiclass classification on the same concept space. For example, with local confidence given beside nodes in figure 1, the valid states of the HEX graph and their respective potentials are listed in table 1:

state	potential
$\varnothing$	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 classifier predicts {animal, pet, dog}. Note that $\varnothing$ is a valid state according to (2).

1.2 Observations, or what the original paper did but didn't say

Note that a valid state does not have to have an active node in the original concept space. This allows an image to be classified to abstract concepts when the classifier is not confident enough to classify to a more concrete concept. This is by no doubt a desirable feature for deployment. However, since all testing images are labelled in the original concept space, classifying to the extended concept space makes performance evaluation troublesome. 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 concept 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 between {animal, pet, dog} and {animal, person} requires examining the confidence along different paths. From this process 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 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.

Listed below are two less important observations:

[1]According to [1]. It shall be explained in section 1.3 that the choice of underlying classifier is actually not arbitrary.

1. There are no learnable variables in this CRF. In other words, all learning is performed in the underlying 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, or what the original paper didn't do

Consider the situation in figure 2:

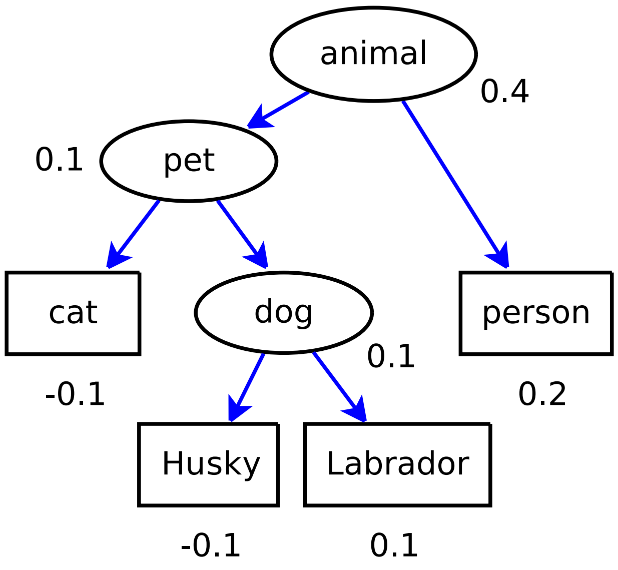

Figure 2: Illustration of the hierarchy depth problem. Exclusive edges are omitted, as they do not carry further information after the state space has been calculated.

Following the same logic as in table 1, the classifier predicts {animal, pet, dog, Labrador}. However, this is a result of a larger number of active nodes rather than active nodes of higher confidence. In this case, it is more desirable if the classifier can predict {animal, person}. This example reveals the first problem of the original HEX model: the prediction is not normalised by the number of active nodes in the state; and the potential function only considers active nodes, while ignoring inactive ones.

The second problem is a combined consequence of the greedy exclusion setup, as discussed in table 1, and the greedy nature of the potential function (1). The prerequisite for the original HEX model to work is that the range of the unary prediction from the underlying classifier is be both positive and negative. This is not a problem in [1], where a convolutional neural network [4] is used as the underlying classifier. However, if a probabilistic classifier is used, in which case the range of the unary prediction from the underlying classifier is in $[0, 1]$, then it is guaranteed that a bottom-level node is activated (see appendix ? for proof). This means the effective state space size is reduced to the that of the original concept space. In such case, the classifier cannot predict to an abstract concept any more. This defeats the purpose of HEX, although inference can be performed more easily with brute force calculation of (1).

2 Reimplementation on PASCAL

3 Improvements

To solve the hierarchy depth problem, we define:

$$\tilde{p}(y|x) = [y \text{ legal}] \prod_i \exp\{f_i(x; w)[y_i = 1] + (1 - f_i(x; w))[y_i = 0]\}$$

There are two issues in Deng's paper. First, the potential function does not handle different depth problem, and is greedy in labelling to bottom-layer nodes. As shall be discussed later, this has similar effects to reducing the state space, and is not a desired thing. Second, the inference system has no learnable part.

To fix the first problem, we redefine the potential function to consider not only the active nodes, but also the inactive ones. However, this fix contradicts with the realistic labelling assumption. Deng's paper assumes that a high proportion of images are actually labelled to their immediate parents, e.g. Husky are labelled as dog. As a result, the bottom layer classifiers (I did not use word "leaf layer" because the hierarchy graph is a DAG in general, and HEX is a loopy CRF. In the case of PASCAL, it happens to be a forest.) have very low confidence, due to the lack of training data. As a result, stopping at the next-to-bottom layer is almost always more preferable.

Of course, this problem can be fixed by limiting the state space to those with one active bottom-layer node. With this fix, the advantage of revised potential function is clear. However, this fix removed one of the core advantages of the HEX model: the possibility to label to an intermediate node, when the classifier is not confident enough to classify to a bottom-layer node.

On this issue, there is one more point to make. All of the ImageNet or PASCAL labels are on the bottom-layer. If the classifier is allowed to label an image to an intermediate layer, then the label space is enlarged. While this can be problematic during the validation and testing stage, it is without doubt a desirable feature during the deploy stage.

To sum up here, 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 addition, the classifier should consider both active and inactive nodes.

Use all images to train the CNN, and images labelled to leaf nodes to train the CRF. Computation of partition function is by brute force, thanks to the tiny state space of PASCAL. Binary weights in the learned model should not suffer from the depth problem, as weights are able to adjust themselves.

4 Dataset

Not augmented by rotation or flip since the original paper didn't

5 HEX with learning

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

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

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

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

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

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

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

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

A 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 [3]. [TODO: discuss dataset properties] The possibility of using CNN as feature extractor has been confirmed in [5].

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.

B 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 [2]. 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.

References

[1] 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.

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

[3] 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.

[4] 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.

[5] 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.