

Data Oriented Picture Parsing: A hybrid model

Dave Cochran

Linguistics and English Language

School of Philosophy, Psychology and Language Sciences

University of Edinburgh

The following paper is an outline proposal for an algorithm for visual parsing which hybridises Data Oriented Parsing (Bod 1998 for an overview) with Connectionist networks. Bod (2005) notes that trees are structures which connect higher level cognitive representations to lower level cognitive representations. It therefore necessary to propose other architectures to handle the lowest level of feature recognition, and it is here that I wish to integrate neural nets into my model of Data-Oriented Vision. This model gains in robustness by calculating probabilities by aggregating the output of the feature-recognition nets (as probabilities) with the probabilities of Stochastic Tree-Substitution derivations.

0. Introduction

The following paper is an outline proposal for an algorithm for visual parsing which hybridises Data Oriented Parsing (Bod 1998 for an overview) with Connectionist networks. Bod (2005) notes that trees are structures which connect higher level cognitive representations to lower level cognitive representations. It therefore necessary to propose other architectures to handle the lowest level of feature recognition, and it is here that I wish to integrate neural nets into my model of Data-Oriented Vision. This model gains in robustness by calculating probabilities by aggregating the output of the feature-recognition nets (as probabilities) with the probabilities of Stochastic Tree-Substitution derivations. My purpose in wishing to implement this algorithm is to provide a basis for grounded semantic representations for Data-Oriented Generation: More specifically, I wish to use paired corpora of images and their descriptions as the exemplar base for a Data-Oriented Generator which I hope will be able to produce grammatical, true descriptive sentences when presented with novel visual input.

Before proceeding, one caveat must be noted. This paper outlines an algorithm which has yet to be implemented. As such, some of the details of the implementation must be left unspecified, simply because I do not yet have a clear idea about how they should be specified, and probably will not until I have had the chance to try a few things out and see what works.

1. Training Data

For the first implementation of this model, I do not wish to overtax it by using complicated visual scenes for which human viewers' analysis would be modulated by imagining the static two-dimensional image as a representation of a dynamic, three dimensional event, or even imagining parts of the picture to represent intensional agents. However, if the images are to be paired with natural language descriptions, abstract images will probably not elicit sufficiently rich and precise descriptions. I therefore propose to use images comprising Roman letters and characters from other writing systems, in black print on a white background; for an example see figure 1. Using letters has a number of advantages:

- Letters and numerals are easily recognised and named, facilitating the provision of natural-language descriptions to train DOG (For an outline of two proposed DOG algorithms, see Cochran 2005).
- Roman letters and Arabic numerals use a small, easily defined set of basic features, making the lowest-level feature-recognition easier.
- Occasionally using Arabic, Greek, Chinese or Devanagari characters, as well as mixing different Roman-alphabet fonts will allow the paired descriptions in the training corpus for DOG to incorporate vague, intermediate, and detailed levels of description; “There’s an Arabic character above a row of three letter B’s, the right of which is in a Sans Serif font”.

The black parts of each image will be divided up into features, using a simple predefined feature set (see figure 2).

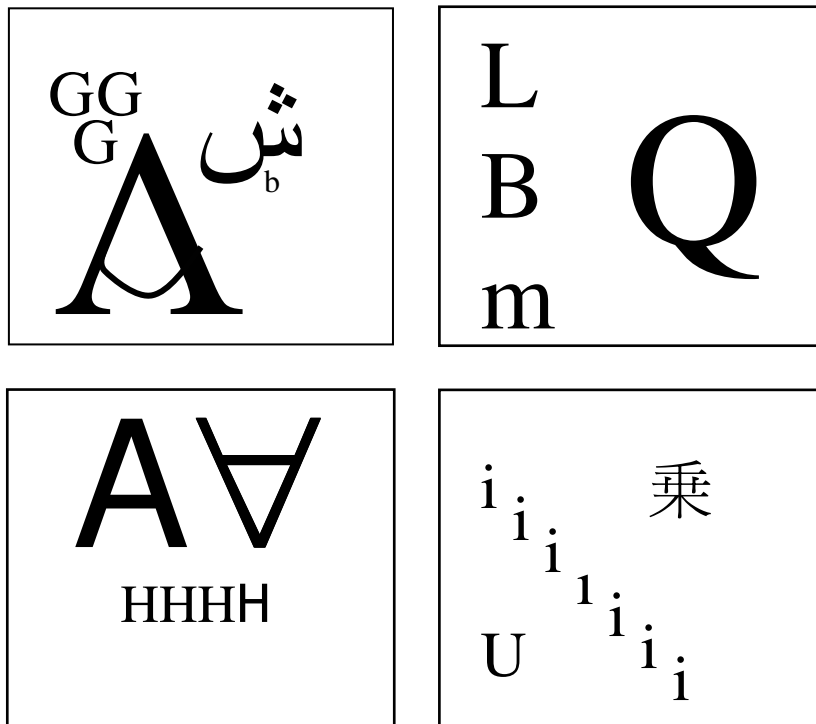


Figure 1a-d (clockwise from top left); four examples of possible input pictures

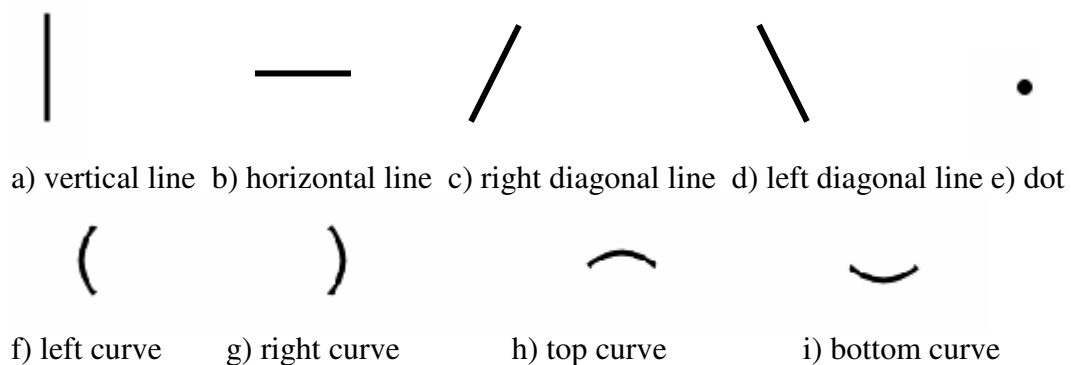


Figure 2; The basic features, $f_1 - f_9$.

The labelled tree-representations in the training corpus would be decomposed into their component subtrees in the normal DOP fashion. Features will then be grouped into pairs, threes, etc, of features or groups of features, and these nested groupings will provide the basis for generating tree-structures over the images, with nodes for all pairings/groupings, and for all individual features. The nodes on each tree will be labelled with vectors, expressed as an angle and a distance, describing the location of their central points (defined as the centre of the smallest circle to encompass all the terminal material under the node) in relation to the central point of the terminal material dominated by their immediate parent node. The root node of the whole tree will be annotated with a vector locating its central point in relation to the centre of the image field. Nodes immediately dominating terminal nodes will also be annotated with an angle, giving the angle of the terminal material in relation to the prototype for their feature-type (-90° - $+90^\circ$ in the case of lines, -180° - $+180^\circ$ for curves, omitted for dots), plus length from tip to tip (for lines and curves).

2. Feature Spotting

When the parser is presented with a novel stimulus, the first stage of processing is to segment the input into labelled features; this is to be done using neural networks. We will assume here that the input image field comprises 300×300 pixels. We want one input node per pixel, and one output node per feature, so even if a two-layer network is computationally sufficient, a feature-spotting network covering the whole field would require 90,000 input nodes and 810,000 connections. Moreover, it would be much harder to localise features to a particular part of the field, particularly given that the field will contain many individual features. However, the job could instead be done by a few smaller high-resolution network, covering (for instance) 10×10 and 20×20 pixels at one pixel per node, 40×40 pixels at 2×2 pixels per node, and 80×80 at 4×4 pixels per node, able to move about the field, generating a feature-map. The networks would move left to right along horizontally along the screen, moving at one-pixel increments, and move down at the end of each sweep by a number of pixels equal to half the height of the area covered by the network. The networks themselves are to be dual channel modular networks similar to those employed by Jacobs, Jordan and Barto (1991) for a similar visual what/where task. All networks would be pre-trained before being used in the DOPP system, and would have their connection weights fixed once they reached a stable state. As each network scans the rows, activations would be sent through the “what” channel of the network to the nine output nodes, representing $f_1 - f_9$. When the activation on any feature node f_{n_x} gets over the threshold value θ , the “where” channel is activated. The input layer to the “where” channel is the same pixels-to-nodes correspondence as the “what” channel, but with an additional input from one (and only one) of nine nodes, corresponding to the output nodes of the “what” channel. The output layer will comprise the same layout of pixel-correlate nodes as the input layer, and the network will be trained to identify all the black pixels in the pixel-input which are part of a feature of the type specified by the input from the “what” network. If more than one “what” network output node exceeds θ , then the “where” network will be activated once for each node. The “where” network output will then be fed to the “what” network, allowing it to see the feature it identified without interference from any nearby black pixels belonging to other features. The “where” network writes its input to a feature-map of

the image (with the same number of pixels as the image itself) as a label containing three parts:

1. a note of feature-input node on the “where” network was activated,
2. a note of *all* activations on the output nodes of the what network on it’s “second look”, expressed as a probability equal to the activation on the node divided by the summed activation on all output nodes.
3. A unique identification code.

A copy of the identification code will be written to every pixel in the feature on the feature map. A pixel of the feature map may contain multiple labels.

When all the networks have scanned the whole image, the feature map will be adjusted in the following ways, in the following order;

1. Any feature which is a proper subset of another feature will be deleted.
2. Any remaining two features which overlap and have the same value in the first parts of their labels will be merged. The values of the second parts of their labels will be averaged, weighted for the number of pixels not shared with the other feature, plus half the pixels shared with the other feature. The new merged feature will be assigned a new identifying code, which will be written to all of its pixels.
3. All labels will have their first part deleted.

All labelled areas with a shared ID code will be identified as single features ready to be parsed. Overlapping non-identical features are left unaltered because sometimes two features may intersect – for example, a letter X crosses a left diagonal with a right.

3. The Probability of Derivations

In principle, any parse tree could be placed over any feature-map, provided the number of features and the number of terminal nodes match. Any terminal node may be matched to any feature, but this is modulated by the probabilities in the second part of the feature label. Figure 3 below shows a feature and its label:


	Part 2	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
		0.052	.0302	.0201	0.886	.0001	.0006	0.001	.00007	.00003
	Part 3	hG5z								

Figure 3; a labelled feature; a diagonal from a letter “A” in 48-point.

In practice, the probability of this feature being interpreted by the parser as anything other than a left diagonal is small. In a derivation of a parse, the probability of the parse is the product of the probabilities of each tree-substitution operation, each tag giving spatial information (more on this below) and each terminal node. The probability of a terminal node is determined by which type of feature it labels the feature it dominates as; a terminal node labelling the feature in fig. 3 as a dot has a

probability of 0.00003, and a terminal node which interprets it as a left diagonal has a probability of 0.886. This confers two advantages; firstly, this gives the system a way of handling ambiguous features, and secondly, it gives the system a way of handling features such as the bottom curve where the horizontal line should be in the large letter “A” in figure 1a above. In such cases, the low probability of the terminal node (i.e. interpreting a bottom curve as a horizontal line) is balanced out by the fact that many more subtrees in the training corpus are available to be applied in that area if the feature is interpreted as a horizontal line.

The parser is programmed to automatically recognise equivalencies between rotated features, and the training data by which the feature-recogniser networks are trained will be skewed to train the nets to slightly inflate the probabilities of wrong outputs that are rotationally symmetrical with the correct output. So, if a feature whose highest f_x -value is f_2 , (horizontal line) but the parser interprets it as f_1 (vertical line) it will immediately be flagged as possibly being f_1 rotated through 90° . Rotation can be passed up the tree at no extra probability cost, and having passed up from daughter node y to mother node x , will automatically pass down to sister nodes $x_1 \dots x_n$ automatically, at no extra probability cost.

Similarly node-labels on subtrees recording the vectors giving the location of their centres in relation to the centres of their immediate parents, and the node-labels on nodes immediately dominating terminal nodes recording the rotation and length of the feature, can all be distorted at a probability cost, which will be low for small distortions, but rises exponentially as the distortion increases. Any distortion applied to a nonterminal node will also apply to all the nodes it dominates at no extra probability cost.

These provisions for distortion will allow inverted characters, such as the inverted “A” in fig 1c., or characters significantly larger or smaller than usual, such as the lower case “b”, or the distorted uppercase “A” in fig 1a., to be correctly interpreted. I have left the mathematics of how these probability costs are calculated unspecified for now, as this is simply something I am not yet decided on, and which will probably require some experimentation to get right.

4. Parsing

One disadvantage of this model, computationally speaking, is the massive proliferation of possibilities caused by the rules for distortion and the selection of low-probability interpretations of the identity of features. Exactly how this is to be dealt with will probably be a matter for experimentation, but my first thought is to:

1. Do several rounds of Monte Carlo sampling, at first with a strict limit set to how much probability cost the system is prepared to absorb from distortions, and a limit on the maximum number of features which may be interpreted as one of the lower-probability outputs, initially set to 0. These limits are the “Probability Cost Limits” (PCLs)
2. The output of this round is set as the “Best Guess So Far” (BGSF).
3. A further three rounds will follow, one with an incremental slackening of the distortion limit on one round, another with an increase of the lower-probability output limit by 1, and another with both.

4. If all three of these slackenings of the PCLs results in no significant increase in performance (measured by the total probabilities of all derivations yielding the majority parse in the round, minus the summed probabilities of all the other derivations), then the current BGSF wins overall and is taken to be the correct parse.
5. If not, the round with the highest performance “wins” and becomes the BGSF, and the PCLs are reset accordingly.
6. Steps 3-5 are repeated until a BGSF becomes overall winner.

As for the process of derivation itself, because the parse tree is derived over a two-dimensional input, and the branchings of the tree are not guaranteed to be pairwise, there is no guarantee that the tree can be mapped onto a two-dimensional tree over a one-dimensional input, similar to those developed for musical and linguistic structures. This, unfortunately, disallows the tidy top-down, leftmost-first progression of substitutions in normal DOP derivations. Instead, I make the following tentative suggestion, loosely based on Simplicity-Likelihood DOP (Bod 2002) and employing a combination of bottom-up and top-down parsing:

1. The derivation begins bottom-up. The database of training corpus subtrees is searched for the n most “explanatory” subtrees; that is to say, those which can map on to the largest number of features, without violating the PCLs for that round.¹ A subtree is selected from these at random, modulated by the probability of the subtree itself (its frequency divided by the total number of subtrees not ruled out by the feature map and PCLs) and any probability costs incurred by getting it to fit onto the image.
2. If this subtree has nonterminal leafnodes, top-down substitutions begin. Since a leftmost-first progression is not guaranteed, substitution sites will be picked randomly from the all the nonterminal leaf-nodes.
3. If at any point there are no remaining nonterminal leaf-nodes, but still features on the feature map unaccounted-for, an upwards substitution may be made, substituting a new subtree for the root-node of the tree derived so far. This operation would join a nonterminal leaf-node of the new tree to the root node of the existing subtree. The subtree would be selected in the same way as the subtree in stage 1.
4. Stages 2 and 3 would be repeated until there were no more unaccounted-for features on the feature map.

5. Conclusions and Further Directions

Everything about the model proposed in this paper is tentative and untested, and as such, the process of implementation is bound also to be one of experimentation and revision at all stages. However, I believe that the basic intuition behind the proposed algorithm is sound; that if sufficient ways can be found to keep the computational complexity of derivations under check, the aggregation of probabilities from the feature-spotting level to the topmost levels of analysis, modulated by probability costs for distortions, will result in a parser with a high level of robustness and accuracy, at least for the types of input that it is designed. However, there is much more that could

¹ Subtrees with a number of non-terminal leaf-nodes greater than the number of features in the image, minus the number of features accounted for by the subtree are excluded.

be done with it, and I wish to end with a few thoughts as to ways the algorithm could be extended:

1. Labels similar to the tags used in DOP for language could be added to the nodes in the trees, indicating O = object, P = part (of object), G = grouping (of objects).
2. The derivation system could be modified to also allow (at a probability cost) the inclusion of features in the parse absent from the map altogether, the intuition here being that in image 1d., the “i” with the missing dot will normally be perceived either as actually having the dot, or it will be perceived as an “i minus the dot” rather than just an “i”.
3. The range of features the networks could identify could be extended. In particular, I would like, in later versions, to go from using pre-trained networks with pre-defined feature-types to networks trained *in situ* using Hebbian learning, unsupervised (but receiving some feedback from the parser). This would, I hope, be a step towards training a parser capable of handling real scenes.
4. The feature mapping could be developed into something more sophisticated, by allowing to map features from paired 2-dimensional images into a 3-dimensional map, allowing stereoscopic vision.
5. The subsystem governing the movement of the hi-res feature spotting nets could be developed to incorporate a more realistic model of saccade planning, foveal and parafoveal vision. Combined with the development of the projection of unseen elements suggested in 2, this could be developed into a model of active vision, in which projection from the tiny, hi-res, moving foveal field, and the much larger lo-res surrounding parafoveal field, using the experiential knowledge contained in the training corpus combine to present an actively constructed visual field with a much larger central hi-res area, using visual stimuli that change over time.
6. Extending the tree model of visual scenes to incorporate the temporal structure of events, and incorporating the ability to project from a static image as a representation of a dynamic event.
7. Incorporating FINST (finger of instantiation, Pylyshyn 1989) nodes into the visual analysis, allowing the tracking and indexing of objects, even when periodically occluded
8. The re-adaptation of the basic architecture of the original model (connectionist feature spotters aggregating probabilities with DOP parsing for higher level analysis) into other modalities of perception; in particular, for developing a version of DOP that can work with corpora of recorded, rather than written, speech.
9. Integration of multi-sensory information into a holistic sensory environment. For instance, DOP using video recordings of speech, integrating audio-DOP with lip-reading.

References

Bod, R. (1998), *Beyond Grammar; An Experience-Based Theory of Language*, Stanford, California: Centre for the Study of Language and Information.

- Bod, R., (2002). "A Unified Model of Structural Organization in Language and Music". *Journal of Artificial Intelligence Research*, 17(2002): 289-308.
- Bod, R., (2005). "Towards Unifying Perception and Cognition". Prepublication.
- Cochran, D., (2005), *Using Stochastic Tree-Substitution Grammar in Iterative Learning Simulations as a way of approaching issues in Diachronic Syntax*, paper given at the College of Arts and Social Sciences Postgraduate Conference at the University of Aberdeen on 23rd June 2005.
- Jacobs, R.A., Jordan, M.I., and Barto, A.G. (1991) Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive Science*, 15, 219-250.
- Pylyshyn, Z.W. (1989). The role of location indexes in spatial perception: A sketch of the FINST spatial-index model. *Cognition*, 32, 65-97.