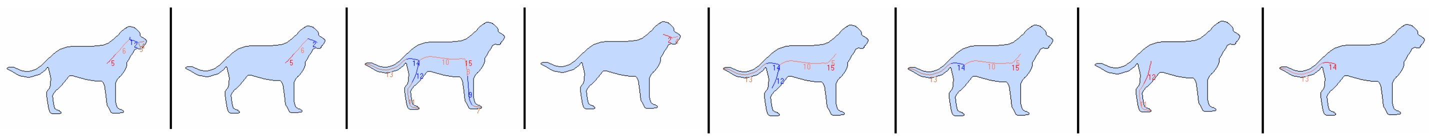
Part Learning to Support Graph-Based Object Recognition

Zoya Gavrilov, Sven Dickinson, University of Toronto and Diego Macrini, Queen's University



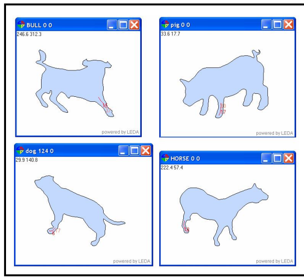
With no prior assumptions, our framework is capable of decomposing an object into semantically-meaningful parts. As a result, we can hope to be on the right track to capturing the efficiency and accuracy of human perceptual organization.

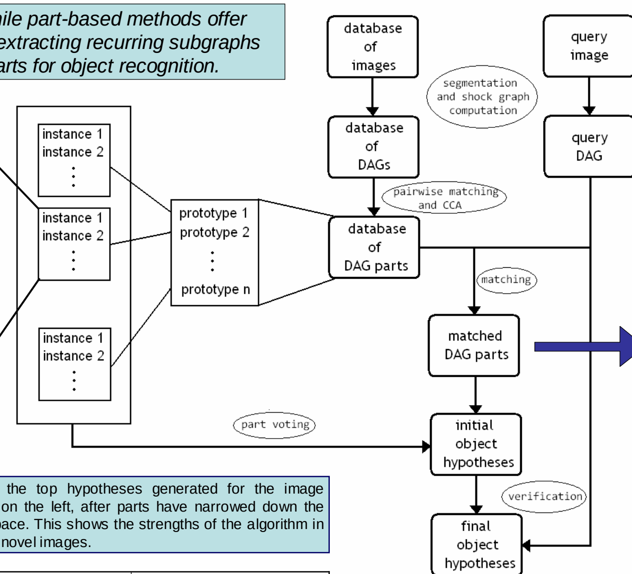
ABSTRACT

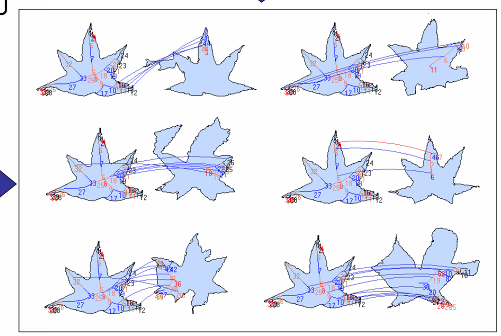
Storing and matching full-blown object models is an intractable task for object recognition, while part-based methods offer greater hope for approaching human-like perceptual abilities. We present a methodology for extracting recurring subgraphs from graph-based representations of objects and using them in building up a vocabulary of parts for object recognition.

FRAMEWORK

- Given a database of object silhouettes (segmented images [1]), we build directed acyclic graph (DAG) representations of the object shapes.
- We pairwise match all the DAGs and extract recurrent subDAGs to be our parts.
- Similar parts are treated as instances of the same part, and are represented by the part prototype which is stored along with other part prototypes in the part database. Instances corresponding to each part live separately in instance databases.
- Query DAGs are matched against the part database and part instances corresponding to each matched prototype vote for object hypotheses.

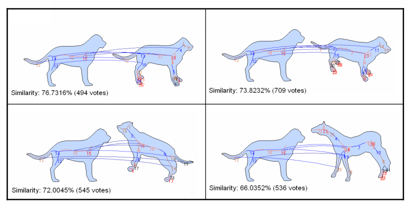






Although any graph representation may be used with the same framework, we utilize Shock Graphs [2] for their ability to capture the shape of an object, while maintaining invariance to position, scale, and rotation. These qualities help to make our parts even more generalizable.

Some of the top hypotheses generated for the image fragment on the left, after parts have narrowed down the search space. This shows the strengths of the algorithm in matching novel images.

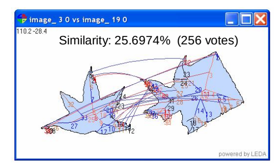


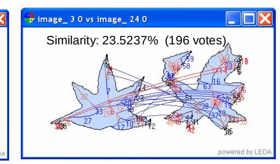
Allowing parts to vote for object hypotheses avoids a linear search and full-graph match against all stored database objects.

A few parts may be enough to recognize a whole object, both reducing the complexity of the match, and allowing the recognition of degraded and occluded objects.

Above: some of the matches generated when a query DAG was matched against the part database. This shows the strengths of the algorithm in matching to noisy and occluded objects.

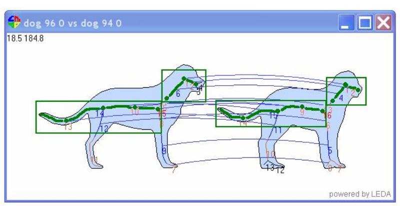
Below: some of the top hypotheses generated for the query object after the parts above have voted.





Low similarity values of these matches means these hypotheses are not likely to have been discovered unless the matching was done by parts.

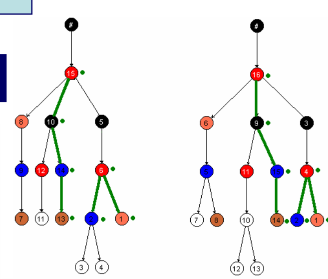
Connected Component Analysis



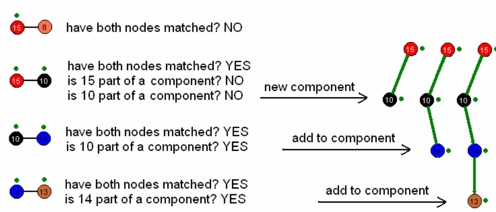
After a match is forced between two DAGs, the graph matching algorithm [2] assigns similarity values to nodes in both graphs.

We say that "nodes have matched" (right) if their similarity values are above a set threshold.

1. Levinshtein, A., Sminchisescu, C., Dickinson, S.: Optimal contour closure by superpixel grouping. In: European Conference on Computer Vision. (2010)
2. Shokoufandeh, A., Macrini, D., Dickinson, S., Siddiqi, K., Zucker, S.: Learning to detect objects in images via a sparse, part-based representation. IEEE Transactions on Pattern Analysis and Machine Intelligence. Volume 27. (2005) 1125-1140.



Looping through all the edges of the first image DAG:



Maximal component for the first DAG has been built.
This process is repeated for the second DAG, and then gain for the other connected component common to both DAGs.

Recomputing Prototypes

A newly-formed connected component is compared to all the part prototypes in the database. If it closely matches a part prototype, then it becomes an instance of that part. Otherwise, it becomes a new part prototype in the database.

Each time a part is added as an instance, the part prototype (P) is recomputed. If the new part is a better representative (according to the formula below) of all the part instances (p_i), then it becomes the new part prototype. A part prototype meets the following property:

$$\sum_{i=1}^{n} SIM(P,p_i) > \sum_{i=1}^{n} SIM(P_j,p_i), \forall j \in [i..n]$$

(where n is the size of P's instance database)

EXTENSIONS

- utilizing multiple segmentations
- running the framework on other graph-based representations

FUTURE WORK

- encoding part relations in the form of a part hierarchy
- offering resegmentations based on part hypotheses