SemanticSky

Suppose we have a set of text-based documents and we want to rank evaluate all 2-place similarity relations occurring between them. Following a typical machine learning approach, we would then first extract a list of features from each document. Secondly, we would write a few algorithms that compare such features pairwise to induce equivalence classes of documents (for similarity is an equivalence relation). At that point we are likely to discover that different algorithms have different strengths and weaknesses, which we might think of as expertises. Suppose for example that (either because the data is partially annotated, or because we do some smart preprocessing beforehand) we know, for each document, the author, title, the content and a list of tags. Suppose also that there are big differences between the quantity and quality of the content of each document: some documents have many tags, some have none, some documents lack an 'author field', some have four authors. Then, the algorithms that base their evaluations on these features will clearly give results that are influenced by the quality of the data. But, taking this step further, we can also imagine that our documents belong to different classes, or groups, such that each class has some typical features that other classes totally miss. We can see this as identifying areas of expertise for the algorithms.

Once we have noticed that each algorithm is better than others in a subset of the input spaces, we will naturally ask ourselves how we can combine their inputs in a smart way, such that each area of the input space is covered and that we exploit the areas of expertise of each algorithm in the best way.

This is the situation that SemanticSky tries to achieve, by employing a supervisor algorithm that gates and collects the various algorithms' opinions and merges them into a single one, trying to make best use of their individual capacities.

Ideally, we would like having some sort of round table where the experts contribute each according to its strengths, and stay silent where they have nothing to say or where they know they are not keen. Making consistently good decisions on pairs of documents of a particular content-type would then result in a high contextual self-confidence for that category of documents, and vice versa. Thus areas of expertise arise for each algorithm through a feedback system that affects the algorithms' self-confidence relative to the content-type of what they are evaluating upon. Such self-confidence is then used to weight the raw algorithmic output, and the suggestion thus obtained is forwarded to the supervisor, that updates its belief state therewith.