25 A Discussion of Semi-Supervised Learning and Transduction

The following is a fictitious discussion inspired by real discussions between the editors of this book and a number of people, including Vladimir Vapnik. It involves three researchers; for simplicity, we will call them A, B, and C, without implying any one-to-one mapping to real persons. The topic of the discussion is: What is the Difference between Semi-Supervised and Transductive Learning?

A: Let me start by saying that to me, the topic of our discussion seems strange. Rather than asking for the difference, we should ask what SSL[1] and transduction have in common, if anything. SSL is about how to use information contained in unlabeled data which we have in addition to the labeled training set. Transduction, on the other hand, claims that it is powerful because it is solving a simpler task than inductive learning.

B: Exactly. In inductive learning, one learns a function that makes predictions on the whole space. Transduction asks for less — it only concerns itself with predicting the values of the function at the test points of interest. This is an easier problem, since an inductive solution implies a transductive one — by evaluating the function at the given test points — but not vice versa.

A: But couldn't you easily build an inductive algorithm from a transductive one by carrying out the following procedure? For all *possible* test inputs $\mathbf{x}$: add $\mathbf{x}$ as a single unlabeled point to the labeled training set, and use the transductive algorithm to predict the corresponding output. This gives you a mapping from $\mathbf{x}$ to y, in other words, a *function*, just like any inductive algorithm. So a transductive solution implies an inductive one, and thus transduction is no easier than induction.

[1] We use the shorthand SSL for semi-supervised learning.

B: As soon as we have more than one unlabeled point, this argument fails. Nevertheless, in order to retain the distinction between induction and transduction, we may want to exclude the situation. Whatever it is called, even the case of one unlabeled point is interesting: it could be viewed as induction with a function class which is not given explicitly.

Transduction works because the test set can give you a nontrivial factorization of the function class. Let us call two functions equivalent if they cannot be distinguished based on any of the given training or test examples. It is then sufficient to use only one representative of each equivalence class, and forget about all other functions. Our function class is effectively finite, and we can directly write down a generalization error bound.

By the way, the size of the equivalence classes is important for generalization: I believe that functions from large classes generalize better. Think of the notion of a margin: if you have a large margin of separation between two classes of data, then there are usually many different functions that fit into this margin, and correctly separate the data (and thus are equivalent on the data).

C: This seems an interesting point. You said that one point is not enough for transduction — how about for SSL? Would one unlabeled point be of any use?

A: Every unlabeled point gives me information on $P(\mathbf{x})$. Whether the point is useful or not will of course depend on whether my distribution is benign. For example, if the distribution satisfies the semi-supervised smoothness assumption,[2] then even a single point gives me information. For instance, it affects my estimate of the local density of the points, and thus it affects where I will try to enforce smoothness. As a consequence, it affects my prediction of how the class label should behave as a function of the input.

B: For transduction to work, it is not necessary to make smoothness assumptions.

C: But surely, the factorization of the function class which you talked about before will also depend on $P(\mathbf{x})$?

B: Yes.

C: I think it should be possible to construct cases where large equivalence classes generalize worse than small ones. So I would claim that transduction, the way you view it, will not always work, but only if the data are benign in some sense...

A: ... and I would argue that one notion that captures whether the data are benign *is* the semi-supervised smoothness assumption. This also makes the connection with the margin, since large-margin separation is low-density separation.

[2] see chapter 1.

B: Maybe yes, maybe no.

C: And what happens toward the other extreme of an infinite number of points?

A: Usually, learning becomes easier if we have more points. With transduction, the more test points we have, the closer we get to inductive learning, because we will have to predict outputs for a set of points that eventually covers the whole domain. According to B, transduction would then become harder, since induction is harder. But that's absurd — how can one make a problem harder by adding more information?

C: Interesting point... However, I am tempted to defend B, albeit with an argument he may not like: In the limit of infinitely many test points, transduction should converge toward something like "induction *plus knowledge of* $P(\mathbf{x})$." This could well be statistically *easier* than standard supervised inductive learning, provided $P(x)$ contains useful information for our task. Which brings us again to the role of the smoothness assumption.

A: This seems to show that transduction relies on the same kind of assumptions as SSL. And, for increasing amounts of unlabeled points, SSL also converges to induction plus knowledge of $P(\mathbf{x})$. So where is the difference? In the limit of infinitely many unlabeled points, transduction cannot be easier than inductive SSL.

B: In the real world, we do not have infinitely many data points. Anyway, my point of view is more fundamental. It is based on what is behind the VC bounds for induction. To prove these bounds, one uses the symmetrization lemma — we upper-bound the difference between the error on the training set and the expected error by the error on the training sample and the error on a second sample — the ghost sample. This is exact transduction; it is a statement about the error on a given set of points. But the VC bounds for induction then have to take an expectation with respect to the unknown points, or even a supremum over the choice of the points. This is much worse than what one can do *knowing* the points.[3]

A: But a better bound does not necessarily imply a better algorithm..

B: True, but bounds guide us to design new algorithms. Transduction is a step on the way, which lies at the heart of induction. It looks deeper than induction.

C: But doesn't this contradict the no-free-lunch theorem?

B: There might exist distributions for which transduction can give worse results than induction.

[3] See chapter 24 and [Vapnik, 1982].

A: If I try to sum up the arguments of B, there are two different reasons why transduction can be useful. The first one is that the bounds for transduction are tighter than the bounds for induction, and the second one is that measuring the size of the equivalence classes is an opportunity to change the ordering in the structure of our class of functions. This second reason seems closely related to the motivations in SSL.

C: Maybe we should look at a more concrete issue: Is the "transductive SVM" an example of a transductive algorithm?

A: No. It is semi-supervised and inductive. It uses unlabeled data, and it provides a function defined everywhere. Would you agree, B?

B: Maybe it is semi-supervised. My point is that transduction is orthogonal from SSL. SSL stresses new technical ideas while transduction stresses new philosophical ideas related to noninductive inference. I am convinced that in ten years the concept of noninductive inference will be much more popular than inductive inference.

A: I surely agree that the two notions are orthogonal, but for different reasons. To make my point clear let me consider two sets. One of them is a set of unlabeled data which we have for training. I don't care about the predictions on this set, I only care about how to use the information this set provides about $P(\mathbf{x})$. So I need to assume that this set actually comes from $P(\mathbf{x})$, or at least from a distribution that is related to $P(\mathbf{x})$ in some way. The other set is the actual test set. I do not care where it comes from; it could be anything. In my view, a transductive algorithm is one whose solution depends on the test points that I am given. The opposite of a transductive algorithm is an inductive one. A semi-supervised algorithm, on the other hand, is one that depends on the unlabeled set (as opposed to a supervised algorithm). It does not care which test points are used in the end to evaluate its performance.

B: This does not make sense to me. The test points need to be meaningful. Transduction is intrinsically simpler than induction: it does not make predictions for arbitrary test points.

A: Coming back to the idea of avoiding to solve a more complicated problem than necessary, what about *local learning*?[4] The idea behind it is that, given a test point, one should focus on the training points which are in a neighborhood of this test point, construct a local decision rule, and predict the label of the test point according to this ad hoc rule. Isn't it almost the same idea as in transduction?

B: Indeed, the philosophy is similar, since in both cases one solves a simpler problem. However, local learning is still inductive because there exists an implicit decision function,

[4] See [Bottou and Vapnik, 1992].

even though it is never explicitely constructed. The concept of local learning is actually almost the same as transduction with one test point which we were talking about earlier.

C: This local learning idea might also be present in TSVM. Indeed, I can see an advantage in using as unlabeled points the test points rather than an arbitrary set of unlabeled points: by doing so, the algorithm concentrates in the regions of the space where it is important to be accurate, as in local learning.[5]

A: The way I view them, transductive algorithms can also be designed for computational reasons. Take, for instance, the Bayesian committee machine.[6] The solution returned by this algorithm is an expansion on a set of basis functions. But for computational efficiency, only basis functions centered at the test points are considered. So the solution will depend on the test set and the algorithm is transductive according to my definition...

C: ... but not according to the definition of B, since for this algorithm the test points can be arbitrary.

A: If we cannot agree on a definition of transduction, maybe we can at least agree on some examples of transductive algorithms?

C: Graph-based algorithms[7] can be interpreted as both semi-supervised and transductive. They are transductive because there is no straightforward way of making a prediction on a test point which is not drawn from $P(\mathbf{x})$. Indeed, including that point in the graph could be harmful, since it may provide misleading information about $P(\mathbf{x})$. In transduction, the test points have to be from $P(\mathbf{x})$, or at least some distribution related to $P(\mathbf{x})$.

A: And this shows why transductive methods are always semi-supervised: they use information contained in the test points. Otherwise there would be no reason not to consider arbitrary test points.

B: I have a typical example of transductive learning. Consider zip code recognition: since all the digits have been written by the same person, one can gain by trying to recognize all the digits simultaneously instead of one by one.

C: This is an interesting example. But it seems different from the standard i.i.d. framework: in this case, if viewed as drawn from the distribution of all possible digits, the test points are dependent, because they have been written by the same person.

[5] Some experimental evidence for this claim is presented in [Collobert et al., 2005].
[6] See [Tresp, 2000].
[7] See part III of the book.

B: Indeed, and it is probably in this kind of situtation where transduction is most useful: when the test points have some special structure.

A: I do not think we have resolved the question we were asking. Read chapter 25 of Chapelle et al. [2006] and the references therein, and you will understand what I mean.