Semi-supervised learning from side information

Paul Christiano Follow

Feb 4, 2016 · 5 min read

Consider a learning task where generating labels is prohibitively expensive, but where it is possible to gather a number of auxiliary signals that are informative about the true label. These auxiliary signals are not available at test time.

That is, there is a sequence of increasingly expensive types of labels, $z^1, z^2, \ldots, z^n = y$. Write $x = z^0$ for the input features.

Many or all data points are labelled with z^1, fewer are labelled with both z^1 and z^2, and so on, down to a very small number labelled with every type of label (including y). Our goal is to predict y given x.

I think that this setting is quite common, and that it should be of broad interest to AI researchers. But I'm not aware of much work in this setting, and in particular I've never seen the algorithm in this post despite its simplicity. (I welcome pointers to the existing literature that I've missed!)

This problem is especially relevant to AI control, since it enables use of expensive human oversight administered with low probability. For example, good solutions play an important role in my view that we can probably build zero-overhead aligned AI systems.

Examples

1. x is a voice search query. y is an accurate human transcription of the query. z^1 is the text query the user made immediately after the voice query if any, or "null" otherwise. z^2 is a transcription produced by a mechanical turker without strong accuracy checks.

2. x is a question. The z^i are a sequence of answers to that question generated after increasingly lengthy reflection.

3. x is the observation/state of a virtual agent, and a proposed action in that state. y is a human's evaluation of how good that action is. z^1 is the cumulative reward obtained after taking that

action, where the reward function is a proxy for human approval.

4. x is the state of a Go game. y is the outcome of playing out a full game from that state. z^1 is the result of rollouts with a very low cost policy. z^2 is the outcome of a game played by an intermediate-quality policy network starting from the given state. z^3 is a human judgment about the quality of a board position.

Assumptions/elaborations

We assume that the presence of a particular feature is unrelated to the label—features might have a "null" value, but a feature being present with a null value is different from being absent because we didn't bother to collect it.

I'm going to consider the case where the z^i are sparse, but we could also ask a more general question where we are simply interested in variance reduction. I think the more general question is extremely interesting, and that similar techniques will be applicable.

A simple algorithm

Consider the following algorithm:

- Let $f^n = z^n$.
- For each i, train a predictor f^i to predict f^{i+1} given z^0, z^1, ..., z^i. This only occurs on datapoints where all of the relevant data is available. The loss is $\Sigma f^{i+1}(y) \log(f^i(y))$.
- Use f^1 as the classifier.

This is essentially TD learning without the time, and with a logarithmic rather than quadratic loss.

If we are predicting a scalar, then we can go back to using a quadratic loss.

If the space of possible predictions is very large, such that we are essentially training a generative model, then we could instead use a generative adversarial net at each step (with f^i trying to sample from the distribution produced by f^{i+1}). I won't discuss this version more in the post, since it involves some extra complications.

In contrast with a natural generative method for leveraging the z^i, this approach does not bother trying to reproduce the labels themselves. We will see shortly that this lets us prove that having intermediate labels is strictly better than nothing.

An improvement

If each classifier was more powerful than the one before it, then using this procedure directly would seem sensible. But each network is actually trained on much less data than the one before it, and so must have lower capacity or be more aggressively regularized.

That means that we can't count on (e.g.) f^3 being strictly smarter than f^2. So f^2 might actually predict better by predicting the outcome directly rather than predicting f^3.

We can get the best of both worlds, by using f^3 as a variance reduction strategy for f^2.

That is, we train f^i on the sum of the following targets:

- f^{i+1} for every data point where z^{i+1} is available.
- $(f^{i+2} - f^{i+1})/\varepsilon$ for data points where z^{i+2} is available, where ε is the probability that z^{i+2} is available conditioned on z^{i+1} being available.
- $(f^{i+3} - f^{i+2})/\varepsilon'$ for data points where z^{i+3} is available, where ε' is the probability that z^{i+3} is available conditioned on z^{i+1} being available.
- And so on.

In the case where f^{i+1} is completely uninformative, this reduces to only training f^i on data points where z^{i+2} is available. In the case where $f^{i+1} = y$, this amounts to training f^i on all of the data. (If the predictor f^i is constant, this essentially reduces to output distribution matching.)

A claim

Including the intermediate signals z^i can *only improve* the quality of a naive prediction algorithm which only uses the labelled data.

To see this, we use the fact that the naive prediction algorithm is equivalent (modulo choice of learning rates) to our improved algorithm with $f^i = 0$ for all i.

But improving the quality of f^i, as a predictor, is guaranteed to reduce the variance of our training signal (while any change to f will leave our signal unbiased)—at least if we assume that f^i is unbiased and that it does not have higher variance on data points where z^{i+1} is unavailable. This is because we can write the variance of the target as the variance of an unbiased predictor plus that predictor's MSE, and shifting more variance to the predictor decreases the total variance of our signal.

So using any reasonable training procedure for the f^i will result in a lower variance signal than simply setting them to 0.

Similarly, adding additional intermediate training signals can only help.

It's not clear *how much* these additions help. Intuitively, there are many cases where they would significantly reduce the variance of the training signal. Reducing the variance of a training signal by a factor of k corresponds roughly to increasing the amount of data by a factor of k. But it's hard to know how it will actually play out without trying it in some context.

More complex structures

I've assumed that z^i is available whenever z^{i+1} is. It would be nice to apply similar methods in cases where we don't have such a simple linear order.

In this setting, for each set of indices $T \subseteq \{0, 1, 2, \ldots, n\}$, we train a predictor f^T to predict y given the values z^i for $i \in T$. As before, we can use the predictors corresponding to big sets to do variance reduction for the predictors corresponding to small sets. Our classifier is of course the predictor for the subset $T = \{0\}$.

However, there are now many possible variance reduction schemes, each of which is quite complicated. I don't have any particular insight for how to choose amongst them, or even whether they will work at all. It seems like an interesting question if applications with complex structure seem important at some point, but it's probably a fine issue to set aside for now.