Contents

1 Representation and unsupervised learning: Introduction (Sanjeev Arora)

I'll describe ways in which the current frameworks are not up to the task.

In statistical learning theory, we observe examples $x_1, \ldots, x_n$ with corresponding labels $y_1, \ldots, y_n$. ERM is finding

$$\text{argmin}_C \sum_{i=1}^{n} L_C(x_i, y_i) + R(C)$$

where L_C is the loss function. If $x_1, \ldots, x_n$ are iid distributed according to D and C comes from a class of "low complexity" (Rademacher, etc.) then the test loss is approximately the training loss.

This framework doesn't explain many things about deep nets. The following are mysteries.

1. Generalization in deep nets. See Moritz, Recht [Zha+16].[1] Take image data with *random* labels. Training a neural net with sufficient capacity (M nodes) achieves 0 training error. The neural net can fit even random labels.

   But neural nets trained with backprop still generalizes.

2. Unrelated classes seem to help. ImageNet has 1000 classes, each with 1000 examples. Knowing additional classes helps training for a specific class.

3. Transfer learning. Train a deep net on images. The top 2 layers of the deep net has great features for many other visual task.

4. Zero-shot learning: BM Lake, R Salakhutdinov, JB Tenenbaum [LST15].[2] Train on data set with letters from 50 languages. Learn a new character. The program can learn well with one or two examples.

   When you see a character, what does your mind do to remember it? Remember it as distinct strokes.

   In theory papers we ignore the distribution. The statistical learning theorem works for every distribution. But the classifier is very related to the distribution.

   Regularization can incorporate a prior: this is one way to relate with Bayesian approaches.

[1] `https://arxiv.org/abs/1611.03530`

[2] `https://staff.fnwi.uva.nl/t.e.j.mensink/zsl2016/zslpubs/lake15science.pdf`

5. Domain adaptation: If the distributions are very different there are no guarantees. But the distributions can be different: we can train on ImageNet and use for X-ray classification.

   The classifier is very related to the distribution. You can't think of choosing them independently. Clustering, classification with margin are examples. We want a mroe sophisticated language.

   In text, we work with bag-of-words models.

Representation learning is finding a map from a data space (raw pixels, etc.) to a representation space, $x \mapsto h$. This could be a many-to-one map.

We need a language to talk about such maps.

We can use simple models like cluster models, mixtures of Gaussians.

The Bayesian view has been to describe representation learning as distribution learning.The task of learning h is finding the MLE for what generated x. Their notion of representation learning is distribution learning: minimize KL divergence. Why should KL divergence lead to good learning?

We have a loglinear topic model with dynamics which comes up with a vector for each word. How to do this for fMRI data, or small corpora (100 documents, too small to do topic model)? Once you've understood how to understand meanings using Wikipedia, you can do a simple domain adaptation.

We had fMRI data (50k dimensional vector every time step) of people watching a movie (Sherlock). The semantic content of the movie was replaced by human annotation every 5 seconds, 2000 in total. Now we correlate.

Evaluation: Presented a 50k vector, given 5 annotations, choose the correct one.

If we want to classify according to any linear classifier; then we have to preserve all the bits. The human has a semantic model to generate the text; there is a mapping from the semantic representation to the text; we try to find the reverse map.

Some discussions:

- Example: Clustering should take into account the task: how they are going to be used. Representations can be profoundly affect by what you want to do. In clustering there is the concrete task of matching the clustering algorithm with the task. Coming up with tools about how to pick the clustering algorithm is an important problem.

- Early stopping in NN gives better generalization. This is more general than NN. Which algorithms have this property? You can also do early stopping with gradient descent with kernel methods. In principle you can overfit but you do not. (Note: The surrogate might never reach 0, but the classification error can be 0.)

- Can we phrase GANs as a problem rather than an algorithm?

- Neural networks act a little like our brain. This is fascinating. If you look at details in the brain, it's different. Are we functionally capturing what's happening? Once you have a good implemention, you can steal any brain. Ex. Train a neural net to repeat blood flow and use it as a feature.

- Human-in-the-loop: Practitioners have a problem where they want to achieve human accuracy. If they fail to get training error to 0, make network bigger. Look at the generalization error on the validation set. If that's too high, add regularization. Play with network structure, get more data, etc.

  What algorithms would work well with human-in-the-loop?

  For nonconvex problem, often human creativity is required to get good initialization. To prove anything nontrivial, initialization plays a role.

  See Curves dataset.

Some general topics:

- Nonconvex optimization
- Generalization vs. optimization for nonconvex models.
- Representation power of depth (measure other than number of nodes)
- Models for representations (GANs)
- Differentiable computing. (Real numbers, logic)

Layers are like function calls; recurrent nets are loops. There are primitive operations which you can compose: loop operator, multiply, divide. Kernel methods cannot represent these efficiently. You get power by reusing computation.

Many things you say about neural networks you can say about polynomials. Why don't people use polynomials in ML? There are certain polys you can write as a neural network.

How to interpret "Turing-complete" in the context of NN? Have a read-write memory. It can read things it wrote last time, write things for next time.

References

[LST15] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. "Human-level concept learning through probabilistic program induction". In: *Science* 350.6266 (2015), pp. 1332–1338.

[Zha+16] Chiyuan Zhang et al. "Understanding deep learning requires rethinking generalization". In: *arXiv preprint arXiv:1611.03530* (2016).