III Inductive Bias.

1. Introduction
2. Generalisation and Inductive Bias
3. A Model of Learning Performance
4. Bias–free Learning
5. Restricting the Hypothesis Space Adds Bias
6. There's No Free Lunch
7. Summary of Lecture

1 Introduction

This lecture is not based on Anthony and Biggs. In it we look at inductive bias, an essential part of practical machine learning.

In lecture 2 we saw some function spaces: $B_n, M_n, M_{n,k}, D_{n,k}$.

In this lecture we'll see that the difficulty of learning depends on which concept space C and hypothesis space H are used, partly because of bias.

Outline of Lecture

Part 2

- We need to generalise from the training sample to unseen examples. That is, we need to generate inductive hypotheses.
- The training sample itself says nothing about unseen examples.
- Bias is any reason for classifying unseen examples a certain way.
  - So all inductive algorithms have a bias.

Part 3

- Induction is (often) more important than memorisation.

Part 4

- Trying to do induction without bias does not work - we can't decide.

Part 5

- One way to add bias is to use a restricted hypothesis space.

Part 6

When $C = B_n$:

- no bias is best.
- so no algorithm is best.
- learning when $C = B_n$ is hard!

Successful learning depends on having:

- an appropriate bias.
- a restricted concept space (since there is no appropriate bias when $C = B_n$).

2 Generalisation and Inductive Bias

Suppose the example space $X = \{A, B, C, D, E, F, G\}$ and the training sample is:

$$\big((A, 0), (B, 0), (C, 0), (D, 1), (E, 1), (F, 1)\big)$$

What would you guess the value of G is?

Another Example

Suppose the example space:

$$X = \{John, Martin, Stuart, Sarah, Lisa, Susan, Mary\}$$

and the training sample:

$$\big((John, 0), (Martin, 0), (Stuart, 0), (Sarah, 1), (Lisa, 1), (Susan, 1)\big)$$

What would you guess the value of $Mary$ is?

Generalisations

The problem in the previous two slides was to *generalise* from the training sample to unseen examples.

A generalisation is a hypothesis, i.e., a function.

So this is just the learning problem we've seen before, except now we consider how to classify unseen examples.

Classifying unseen examples is *inductive* (as opposed to *deductive*).

How to Guess the Correct Value?

Given the training sample $\mathbf{s} = \big((00,1), (01,1), (10,1)\big)$ there are 2 hypotheses in B_2 which are consistent with $\mathbf{s}$:

y	$h^{(1)}$	$h^{(2)}$
00	1	1
01	1	1
10	1	1
11	0	1

Clearly the value of 11 can be either 0 or 1.

The training sample *simply does not tell us* whether $h^{(1)}$ or $h^{(2)}$ is the target concept.

Bias

Normally the training sample does not tell us exactly what the target concept is.

In this case, we need to use something else to classify the unseen examples.

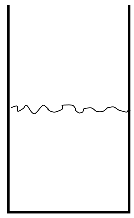
The "something else" is our *inductive bias*.

Bias is any factor which contributes to the classification of unseen examples.

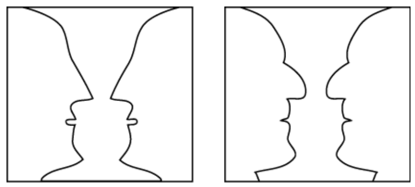
Some Sources of Bias

- Representation cannot express all possible generalisations.
- Generalisation procedure may have some form of bias.
- Intended use, e.g., cost of incorrect guesses (e.g., in detecting cancer).
- Domain knowledge (e.g., $John, Mary \ldots$ are human names, humans consider gender important).
- Bias towards simple hypotheses.
- Bias towards general hypotheses.
- Analogies (use a related concept as a guide).
- Common sense.

Is the Glass Half Full or Half Empty?



What do You See?



3 A Model of Learning Performance

In lecture 1 we modelled learning as the interaction between a teacher and a learner.

Let's measure how close the learner's hypothesis h is to the target concept t using *classification accuracy*.

Let the classification accuracy of h with respect to t be:

$$accuracy(h,t) = \frac{|consistent(h,t)|}{|inconsistent(h,t)| \ + \ |consistent(h,t)|}$$

where $consistent(h,t)$ is the subset of examples for which h is consistent with t.

Performance: Memory and Induction

The learner's performance depends on how well it classifies:

1. Examples it *has* seen before (memory).
2. Examples it *has not* seen before (induction).

The effect of memory depends on:

- How many labelled examples the learner has seen.
- How consistent the learner is.

The effect of induction depends on:

- How appropriate the inductive bias is.

Memorisation and Induction

The contribution of memory depends on how much of the example space X has been seen.

Unless X is very small, the proportion of examples we have seen is likely to be very small.

- In practice, the contribution of memory is often very small.
  - If X is infinite, the contribution of memory is infinitely small!
- So, in practice, performance often depends essentially on induction.
- Nonetheless, in CoLT we tend to ignore induction and focus on memorisation.

4 Bias–free Learning

Let's see what happens when we learn without bias. We need to find:

- A representation which does not introduce bias.
- A learning algorithm which does not introduce bias.

Unbiased Generalisation Language (UGL)

A representation may not be able to express all hypotheses, which introduces a bias.

To learn without bias we need a UGL: a language which can represent every possible subset of examples (i.e., every function).

That is, a UGL must be able to represent the power set (set of all subsets) of the example space X.

Truth tables are a UGL. Monomial formulae are not.

What is the power set of $\{A, B, C\}$*?*

Version Spaces

The *version space*† for a hypothesis space H and training sample s is the subset of H consistent with s.

What is the version space for the training sample $\big((0,0)\big)$*?*

y_1	$h^{(1)}$	$h^{(2)}$	$h^{(3)}$	$h^{(4)}$
0	0	0	1	1
1	0	1	0	1

† See Mitchell's book, chapter 2.

Unbiased Generalisation Procedure

1. For each example x:
2. Compute current version space VS using UGL.
3. If example was previously seen classify it:
4. Positive if it was positive.
5. Negative if it was negative.
6. Otherwise classify x:
7. Positive if most $h \in VS$ think it's positive.
8. Negative if most $h \in VS$ think it's negative.

Using the Unbiased Generalisation Procedure

Suppose $\Sigma = \{0, 1\}$, $H = B_2$, we are using truth tables to represent functions, and we have seen the training sample $\big((00, 0), (01, 0)\big)$.

Use the Unbiased Generalisation Procedure to classify the examples $\big((10), (11)\big)$.

y_1	y_2	$h^{(1)}$	$h^{(2)}$	$h^{(3)}$	$h^{(4)}$	$h^{(5)}$	$h^{(6)}$	$h^{(7)}$	$h^{(8)}$	$\cdots$	$h^{(16)}$
0	0	0	0	0	0	0	0	0	0	$\cdots$	1
0	1	0	0	0	0	1	1	1	1	$\cdots$	1
1	0	0	0	1	1	0	0	1	1	$\cdots$	1
1	1	0	1	0	1	0	1	0	1	$\cdots$	1

Bias–free Learning

Without bias, we have no basis for classifying unseen examples. The best you can do is:

- Memorise the training sample.
- Classify new examples at random.

Bias–free learning is induction-free learning.

5 Restricting H Adds Bias

Suppose we have the same task as before, but now $H = M_2$ (the set of 2-bit monomials).

How does the Unbiased Generalisation Procedure classify $\big((10), (11)\big)$?

y_1	y_2	$h^{(1)}$	$h^{(2)}$	$h^{(3)}$	$h^{(4)}$	$h^{(5)}$	$h^{(6)}$	$h^{(7)}$	$h^{(8)}$	$\cdots$	$h^{(16)}$
0	0	0	0	0	0	0	0	0	0	$\cdots$	1
0	1	0	0	0	0	1	1	1	1	$\cdots$	1
1	0	0	0	1	1	0	0	1	1	$\cdots$	1
1	1	0	1	0	1	0	1	0	1	$\cdots$	1

The Role of Bias

By decreasing the size of H (from B_2 to M_2), we focus induction onto a subset of hypotheses.

We also introduced a bias in our representation, which allowed the Unbiased Generalisation Procedure to make inductive hypotheses about the data.

6 No Free Lunch

Which bias is best? In general, none!

The No Free Lunch (NFL) theorem tells us no bias is better than any other, when we average over all possible target concepts.

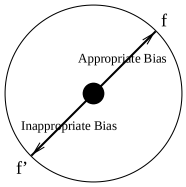
Consider the space B_n of n-bit Boolean functions. Each function f has an opposite f' which classifies each example $y \in X$ in the opposite way:

$$\forall y \in X \quad f(y) \neq f'(y)$$

If f is a good generalisation for y, then f' is a bad generalisation.

Functions and their Opposites

We can think of the space of all functions like this:



No Free Lunch

If a given bias is good for one target concept, it is equally bad for its opposite.

If it is good for n concepts, it is equally bad for n others.

So if we average performance over all concepts, it sums to zero.

Since this is true for all biases, all biases are equal, *when we average over all concepts*.

Learning Arbitrary Boolean Functions is Hard

We've just seen that all biases are equal when the concept space $C = B_n$.

Classifying examples at random is one form of bias.

Therefore, on average, no bias is better than random performance.

So, on average, learning arbitrary Boolean functions is hard.

This slide presents fundamental ideas from complexity theory informally. We will return to them in more detail in a later lecture.

Empirical Comparisons can be Misleading

Since each target function f has an opposite f', if we show experimentally that a learning algorithm performs well on f, we are implicitly showing that it must perform poorly on f'.

If we show algorithm A is *better than* algorithm B on f, we are implicitly showing that A is *worse than* B on f'.

Conclusions:

- Think carefully about what empirical results mean!
- Do not try to find an algorithm which is good for all target functions.

Is Induction Hopeless?

Does this mean we cannot learn inductively?

No. It means successful induction depends on choosing an appropriate bias for the current target function.

Normally, we try to use some *domain knowledge* to add an appropriate bias.

7 Lecture Summary

- Generalisation is a matter of proposing inductive hypotheses which classify unseen examples.
- The generalisations made by a learning system depend on its bias.
- Bias–free learning is just memorisation and guessing.
- Averaged over all target concepts, all biases are equally good.
  - So learning in this space is hard.
- Showing $A > B$ on f implicitly shows $A < B$ on f'.
  - So don't try to find one algorithm which is best for all problems.
- Successful learning depends on having an appropriate bias for the target concept.