Inductive Bias: 
How to generalize on novel data
Overfitting

Noise vs. Exceptions
Non-Linear Tasks

- Linear Regression will not generalize well to the task below
- Needs a non-linear surface
- Could do a feature pre-process as with the quadric machine
  - For example, we could use an arbitrary polynomial in $x$
  - Thus it is still linear in the coefficients, and can be solved with delta rule, etc.

$$Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_n X^n$$

- What order polynomial should we use? – Overfit issues can occur
Regression Regularization

- How to avoid overfit – Keep the model simple
  - For regression, keep the function smooth
    - Inductive bias is that \( f(x) \approx f(x \pm \varepsilon) \)

- Regularization approach: \( F(h) = \text{Error}(h) + \lambda \cdot \text{Complexity}(h) \)
  - Tradeoff accuracy vs complexity

- Ridge Regression – Minimize:
  - \( F(w) = TSS(w) + \lambda \|w\|^2 = \sum (\text{predicted}_i - \text{actual}_i)^2 + \lambda \sum w_i^2 \)
  - Gradient of \( F(w) \): \( \Delta w_i = c(t - \text{net})x_i - \lambda w_i \) (Weight decay)
  - Especially useful when the features are a non-linear transform from the initial features (e.g. polynomials in \( x \))
  - Also when the number of initial features is greater than the number of examples
  - Lasso regression uses an L1 vs an L2 weight penalty: \( TSS(w) + \lambda \sum |w_i| \)
Hypothesis Space

The Hypothesis space $H$ is the set all the possible models $h$ which can be learned by the current learning algorithm

- e.g. Set of possible weight settings for a perceptron

Restricted hypothesis space

- Can be easier to search
- May avoid overfit since they are usually simpler (e.g. linear or low order decision surface)
- Often will underfit

Unrestricted Hypothesis Space

- Can represent any possible function and thus can fit the training set well
- Mechanisms must be used to avoid overfit
Avoiding Overfit - Regularization

- Regularization: *any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error*
- Occam’s Razor – William of Ockham (c. 1287-1347)
- Simplest accurate model: accuracy vs. complexity trade-off. Find $h \in H$ which minimizes an objective function of the form:
  $$F(h) = Error(h) + \lambda \cdot Complexity(h)$$
  - Complexity could be number of nodes, size of tree, magnitude of weights, order of decision surface, etc. L2 and L1 common.
- More Training Data (vs. overtraining on same data)
  - Also Data set augmentation – Fake data, Can be very effective, Jitter, but take care…
  - Denoising – add random noise to inputs during training – can act as a regularizer
  - Adding noise to nodes, weights, outputs, etc. E.g. Dropout (discuss with ensembles)
- Most common regularization approach: *Early Stopping* – Start with simple model (small parameters/weights) and stop training as soon as we attain good generalization accuracy (before parameters get large)
  - Use a validation Set (next slide: requires separate test set)
- Will discuss other approaches with specific models
Stopping/Model Selection with Validation Set

There is a different model $h$ after each epoch

Select a model in the area where the validation set accuracy flattens
  − When no improvement occurs over $m$ epochs

The validation set comes out of training set data

Still need a separate test set to use after selecting model $h$ to predict future accuracy

Simple and unobtrusive, does not change objective function, etc
  − Can be done in parallel on a separate processor
  − Can be used alone or in conjunction with other regularizers
Inductive Bias

- The approach used to decide how to generalize novel cases
- One common approach is Occam’s Razor – The *simplest* hypothesis which *explains/fits* the data is usually the best
- Many other rationale biases and variations

\[
\begin{align*}
ABC &\Rightarrow Z \\
\overline{A}BC &\Rightarrow Z \\
AB\overline{C} &\Rightarrow Z \\
\overline{A}\overline{B}\overline{C} &\Rightarrow Z \\
\overline{A}BC &\Rightarrow \overline{Z} \\
\overline{A}BC &\Rightarrow ?
\end{align*}
\]

- When you get the new input $\overline{A}B\overline{C}$. What is your output?
One Definition for Inductive Bias

Inductive Bias: Any basis for choosing one generalization over another, other than strict consistency with the observed training instances.

Sometimes just called the *Bias* of the algorithm (don't confuse with the bias weight in a neural network).

Bias-Variance Trade-off – Will discuss in more detail when we discuss ensembles.
Some Inductive Bias Approaches

- **Restricted Hypothesis Space** - Can just try to minimize error since hypotheses are already simple
  - Linear or low order threshold function
  - $k$-DNF, $k$-CNF, etc.
  - Low order polynomial

- **Preference Bias** – Prefer one hypothesis over another even though they have similar training accuracy
  - Occam’s Razor
  - “Smallest” DNF representation which matches well
  - Shallow decision tree with high information gain
  - Neural Network with low validation error and small magnitude weights
# Need for Bias

$2^n$ Boolean functions of $n$ inputs

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$x_3$</th>
<th>Class</th>
<th>Possible Consistent Function Hypotheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>?</td>
</tr>
</tbody>
</table>
Need for Bias

$2^n$ Boolean functions of $n$ inputs

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>Class</th>
<th>Possible Consistent Function Hypotheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>? 0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>
# Need for Bias

$2^n$ Boolean functions of $n$ inputs

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>Class</th>
<th>Possible Consistent Function Hypotheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1 1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0 0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0 0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0 0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>?</td>
<td>0 1</td>
</tr>
</tbody>
</table>
Need for Bias

$2^n$ Boolean functions of $n$ inputs

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>Class</th>
<th>Possible Consistent Function Hypotheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td>0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td>0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
<td>0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td>0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 1</td>
</tr>
</tbody>
</table>

Without an Inductive Bias we have no rationale to choose one hypothesis over another and thus a random guess would be as good as any other option.
Need for Bias

$2^2$ Boolean functions of $n$ inputs

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>Class</th>
<th>Possible Consistent Function Hypotheses</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 1 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 1 1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>?</td>
<td>0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1</td>
</tr>
</tbody>
</table>

Inductive Bias guides which hypothesis we should prefer? What happens in this case if we use simplicity (Occam’s Razor) as our inductive Bias?
Learnable Problems

- “Raster Screen” Problem
- Pattern Theory
  - Regularity in a task
  - Compressibility
- Don’t care features and Impossible states
- Interesting/Learnable Problems
  - What we actually deal with
  - Can we formally characterize them?
- Learning a training set vs. generalizing
  - A function where each output is set randomly (coin-flip)
  - Output class is independent of all other instances in the data set
- Computability vs. Learnability (Optional)
Computability and Learnability – Finite Problems

- Finite problems assume finite number of mappings (Finite Table)
  - Fixed input size arithmetic
  - Random memory in a RAM
- Learnable: Can do better than random on novel examples
Finite problems assume finite number of mappings (Finite Table)
- Fixed input size arithmetic
- Random memory in a RAM

Learnable: Can do better than random on novel examples
Computability and Learnability – Infinite Problems

- Infinite number of mappings (Infinite Table)
  - Arbitrary input size arithmetic
  - Halting Problem (no limit on input size)
  - Do two arbitrary strings match
Computability and Learnability – Infinite Problems

- Infinite number of mappings (Infinite Table)
  - Arbitrary input size arithmetic
  - Halting Problem (no limit on input size)
  - Do two arbitrary strings match

Infinite Problems

Learnable Problems:
A reasonably queried infinite subset has regularity

Computable Problems:
Only those where all but a finite set of mappings have regularity
No Free Lunch

- Any inductive bias chosen will have equal accuracy compared to any other bias over all possible functions/tasks, assuming all functions are equally likely. If a bias is correct on some cases, it must be incorrect on equally many cases.

- Is this a problem?
  - Random vs. Regular
  - Anti-Bias? (even though regular)
  - The “Interesting” Problems – subset of learnable?

- Are all functions equally likely in the real world?
Interesting Problems and Biases

All Problems
Structured Problems
Interesting Problems

Inductive Bias

P_I

Inductive Bias

Inductive Bias

Inductive Bias

Inductive Bias

CS 478 - Inductive Bias
More on Inductive Bias

- Inductive Bias requires some set of prior assumptions about the tasks being considered and the learning approaches available.
- Tom Mitchell’s definition: Inductive Bias of a learner is the set of additional assumptions sufficient to justify its inductive inferences as deductive inferences.
- We consider standard ML algorithms/hypothesis spaces to be different inductive biases: C4.5 (Greedy best attributes), Backpropagation (simple to complex), etc.
Which Bias is Best?

- Not one Bias that is best on all problems
- Our experiments
  - Over 50 real world problems
  - Over 400 inductive biases – mostly variations on critical variable biases vs. similarity biases
- Different biases were a better fit for different problems
- Given a data set, which Learning model (Inductive Bias) should be chosen?
Automatic Discovery of Inductive Bias

- Defining and characterizing the set of Interesting/Learnable problems
- To what extent do current biases cover the set of interesting problems
- Automatic feature selection
- Automatic selection of Bias (before and/or during learning), including all learning parameters
- Dynamic Inductive Biases (in time and space)
- Combinations of Biases – Ensembles, Oracle Learning
Dynamic Inductive Bias in Time

- Can be discovered as you learn
- May want to learn general rules first followed by true exceptions
- Can be based on ease of learning the problem
- Example: SoftProp – From Lazy Learning to Backprop
Dynamic Inductive Bias in Space
ML Holy Grail: We want all aspects of the learning mechanism automated, including the Inductive Bias

Just a Data Set or just an explanation of the problem
BYU Neural Network and Machine Learning Laboratory
Work on Automatic Discover of Inductive Bias

- Proposing New Learning Algorithms (Inductive Biases)
- Theoretical issues
  - Defining the set of Interesting/Learnable problems
  - Analytical/empirical studies of differences between biases
- Ensembles – Wagging, Mimicking, Oracle Learning, etc.
- Meta-Learning – A priori decision regarding which learning model to use
  - Features of the data set/application
  - Learning from model experience
- Automatic selection of Parameters
  - Constructive Algorithms – ASOCS, DMPx, etc.
  - Learning Parameters – Windowed momentum, Automatic improved distance functions (IVDM)
- Automatic Bias in time – SoftProp
- Automatic Bias in space – Overfitting, sensitivity to complex portions of the space: DMP, higher order features