Meta Learning

MIT

Iddo Drori, Fall 2020
Meta Learning

task = data splits, priors

task data → meta learning algorithm → learning algorithm

new task data

task data → test data → predictor → prediction
Meta Learning

\[ g' = h(S) \]
\[ f^\wedge = g'(D') \]
\[ y^\wedge = f^\wedge(X^\wedge) \]
Automated Machine Learning

tasks = \{\text{binary classification, multi-class classification, regression,}...\}
Automated Machine Learning (AutoML)
Machine Learning Systems
ML System

Figure source: Accurate protein structure prediction by embeddings and deep learning representations, Drori et al, 2019.
Transformers
Transformers

- Text
- Task agnostic architecture
- Task specific datasets of 1-100k samples for fine tuning or
- Few shot learning
Fine Tuning

- Examples for specific task
- Gradient updates
Meta Learning in Language Models

• Humans don’t require so many examples for new tasks instead use a handful
• Avoid collecting domain specific datasets and fine tuning for new tasks
• Broad learning during training
• Adapt to tasks at runtime
Few-Shot Learning

- Only examples at runtime: number of examples $k$, 1, 0
- $K$ examples of task
  - Translate English to Chinese
  - how are you -> ni hao ma
  - 1, 2, 3 -> yi, er, san
- 1 example of task
  - Translate English to Chinese
  - how are you -> ni hao ma
- No examples, language description
  - Translate English to Chinese
Math Word Problems and Question Answering
Math Word Problems and Question Answering

• Question
  “At the fair Adam bought 13 tickets. After riding the ferris wheel he had 4 tickets left. If each ticket cost 9 dollars, how much money did Adam spend riding the ferris wheel?”

• Answer
  81
Math Word Problems Approaches

- Template-based methods
- Prediction of operators and operands
- Search space of binary expression trees
- Deep neural networks
- Reinforcement learning, building expression trees
Expression Tree

• **Question**
  “At the fair Adam bought 13 tickets. After riding the ferris wheel he had 4 tickets left. If each ticket cost 9 dollars, how much money did Adam spend riding the ferris wheel?"

• **Answer**
  81

$$(13 - 4) \times 9 = 81$$

$x=13 \ y=4 \ z=9$

$(x \ op1 \ y) \ op2 \ z$

$x \ y \ op1 \ z \ op2$
Prediction

• Probability over operators and operands
• Probability over trees
20 Years of Superhuman Game Playing

- IBM Deep Blue vs. Kasparov 1997
- Google AlphaGo vs. Lee Sedol 2016
Deep Reinforcement Learning

- Deep neural network represents policy, value function, model
- Optimize loss function by stochastic gradient descent
Deep Reinforcement Learning Applications

- Video games
- Board games
- Rubik’s cube
- Protein folding
- Dialogue synthesis
- Automatic machine learning
- Robot control
- Self driving cars
Motivation: Dual Process Theory

Fast
Autonomous
May not require working memory

Slow
Involves mental simulation and decoupling
Requires working memory

Daniel Kahneman, Thinking Fast and Slow, 2011
Dual Process Theory: Simple Analogy

\[ 34^2 = ? \]
Dual Process Theory: Simple Analogy

Type 1

\[ 30 \times 30 = 900 \]
\[ 4 \times 30 = 120 \]
\[ 30 \times 4 = 120 \]
\[ 4 \times 4 = 16 \]

Type 2

\[ 34 \times 34 = 34 \times 30 + 34 \times 4 \]
\[ 4 \times 4 = 16 \]
\[ 34 \times 30 = 30 \times 30 + 4 \times 30 \]
\[ 34 \times 4 = 30 \times 4 + \]
Dual Process Theory: Simple Analogy
Dual Process Theory: Simple Analogy

Q: Second time, what is $34^2$?

A: $1156$ right away, since it's now type 1, so we'll keep the network which knows this rather than use previous network.

Q: Next, what is $34^4$, use $34^2$ etc.

Dual process iteration with self play.
Expert Iteration

\[ f_\theta(s) = (P(a|s), v(s)) \]

Anthony et al., Thinking fast and slow with deep learning and tree search, NeurIPS 2017.
Mastering chess and Shogi by self-play with a general reinforcement learning algorithm, Silver et al, 2018
Neural Network Loss Function

- Minimize loss function

\[ L(\theta) = -\pi \log p + (\nu - e)^2 + \alpha ||\theta||^2 \]

- \(\theta\): Neural network parameters
- \(\nu\): Neural network predicted value
- \(e\): Actual value
- \(p\): Neural network predicted probabilities
- \(\pi\): Actual search probabilities
Math Question Answering using a Transformer and Reinforcement Learning

- **Transformer**
- **$g'$** is RL

\[ f^\wedge = g'(D', T) \]
\[ y^\wedge = f^\wedge(X^\wedge) \]
Reinforcement Learning

- State: graph, tree, expression
- Actions: selected operator and operands
- Reward: correct action or expression evaluation

Tree

Expression

(13 - 4) * 9 = 81
Meta 6.036 Lab 1

- Machine Learning prerequisite
- Max cap of 50 students
Meta 6.036 Lab 1
1.1) Visual Intuition

We have some data that we know falls into two categories: positive (shown as a green plus sign) and negative (shown as a red minus sign). We want to predict whether new data points are positive or negative. The image below shows a linear classifier (the blue line) that our machine found, which perfectly classifies the existing data (though in practice, it's rare to perfectly classify real world data).

Note: Here we use $x_1$ and $x_2$ as axes instead of the typical $x$, $y$ axes.

A) A new data point is shown in the plot above as a question mark. What category will our linear classifier predict this new data point to be in? Positive or Negative?

The new point will be classified as:
- Positive
- Negative

Save  Submit  View Answer  Ask for Help  100.00%

As staff, you are always allowed to submit. If you were a student, you would see the following:
You have infinitely many submissions remaining.

B) Is it guaranteed that this prediction is correct?
1.1) Visual Intuition

We have some data that we know falls into two categories: positive (shown as a green plus sign) and negative (shown as a red minus sign). We want to predict whether new data points are positive or negative. The image below shows a linear classifier (the blue line) that our machine found, which perfectly classifies the existing data (though in practice, it's rare to perfectly classify real world data).

Note: Here we use $z_1$ and $z_2$ as axes instead of the typical $x$, $y$ axes

A) A new data point is shown in the plot above as a question mark. What category will our linear classifier predict this new data point to be in? Positive or Negative?

The new point will be classified as:
- Positive
- Negative

As staff, you are always allowed to submit. If you were a student, you would see the following:
You have infinitely many submissions remaining.

B) Is it guaranteed that this prediction is correct?
Meta 6.036 Lab 1

- Transformer
- CNN
1) Linear Classifiers

Machine Learning is about making decisions or predictions based on data. Often times, we will use models to help us make good predictions or decisions. Models are, essentially, fancy functions. Given new input data, models output something that helps us (or a computer) make predictions/make decisions.

The beauty of machine learning is that, rather than building a model ourselves that allows us to make predictions/decisions, we can make a machine train a model that makes good predictions and decisions for our use case!

Binary Linear Classifier

The first model we look at in this course is a binary linear classifier. As you can read about in the lecture notes, a binary linear classifier is a simple, yet powerful type of model (which is, again, a fancy function) that is linear and that classifies data into two (binary) categories. In other words, given some new data, it will predict which category that data falls into.

How does this work? Let's find out!
Meta 6.036 Lab 1

- Transformer
D) Graph the line $x_1 + 2 = 2x_2$ on a $x_1, x_2$ plane. Then, plot the vector $\theta = \begin{bmatrix} 1 \\ -2 \end{bmatrix}$. What do you notice about the vector in relation to the slope of the linear function it represents?
Meta 6.036 Lab 1

- Transformer
- Expression tree
- Meta learning
1.3) Predicting

So now we are familiar with how to represent a linear function as an array $\theta$ and an offset $\theta_0$. We also know from (A) and (B) that in theory, we can use the linear function to classify points as positive and negative. But how do we decide which side is positive? How do we classify a single point? Note: just a line is not a 2D classifier! We need to know which side of the line to designate as positive and which side is negative!

As you may have found in (D), the vector $\theta$ is perpendicular to the line it defines. This leads to a generalization which will help us classify points as positive or negative, given $\theta$ and $\theta_0$.

In a 2-dimensional space, our linear classifier takes the form of a 1-dimensional line that can be characterized by a scalar offset ($\theta_0$) and a $2 \times 1$-vector ($\theta$) that is normal or orthogonal to the line.

Turns out...In a 3-dimensional space, our linear classifier takes the form of a 2 dimensional plane that can be characterized by a scalar offset ($\theta_0$) and a $3 \times 1$-vector ($\theta$) that is orthogonal to the plane.

Notice the pattern?

In general, in a $d$-dimensional space, linear classifiers take the form of a $d - 1$ dimensional hyperplane which can be characterized by a scalar offset ($\theta_0$) and a $d \times 1$-vector ($\theta$) that is orthogonal to the hyperplane.

Given the offset and the $\theta$ vector, we can directly classify points as positive and negative (we don't need any more information and we don't even have to plot it as you'll see in a bit!). We will refer to classifiers and distinguish amongst classifiers using the appropriate $\theta$ vector and offset $\theta_0$.

So how do we use $\theta$ and $\theta_0$ to classify points?

Points that fall on the side of the binary linear classifier that the normal vector points in are classified as positive. So, let's see an example!
Meta 6.036 Lab 1

- Transformer
Suppose $\theta = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$ and $\theta_0 = 0$. Points below this particular classifier are classified as positive and points above this classifier are considered negative.
Meta 6.036 Lab 1

- Transformer, CNN
- Computational graph, expression tree
- Meta learning
Now suppose, we wanted to classify the point $(2, -2)$.

Looking at the graph we can see that $(2, -2)$ should be classified as positive.
Meta 6.036 Lab 1

- Transformer, CNN
- Computational graph, expression
- Meta learning
We can confirm this mathematically as well. We know that the equation for the line corresponding to \( \theta = \begin{bmatrix} 2 \\ -1 \end{bmatrix} \) and \( \theta_0 = 0 \) is \( 2x_1 = x_2 \) (If you don't see this, derive it as shown in 1.2). Any point \((d_1, d_2)\), for which \( d_1 = 2 \), will be classified as positive if \( d_2 < 2d_1 \). This can also be written as \( 0 < 2d_1 - d_2 \). Thus in our case...

\[
\begin{align*}
0 &< 2d_1 - d_2 \\
0 &< 2(2) - (-2) \\
0 &< 4 + 2 \\
0 &< 6
\end{align*}
\]

Since the above is true, we can classify the point as positive. For the point \((-1, 1)\), is \( 0 < 2d_1 - d_2 \)? Is \( 0 < 2 \times (-1) - 1 \)? It is not. Therefore, \((-1, 1)\) is classified as negative (and we can check this against the plot)! But notice how we do not require the plot to determine the classification.
Meta 6.036 Lab 1

- Transformer
E) Suppose we are given the classifier defined by \( \theta = \begin{bmatrix} \frac{-2}{5} \end{bmatrix} \) and \( \theta_0 = 0 \). How will our classifier classify the point \((1, 3)\)?

The classifier will classify the point \((1,3)\) as:
- Positive
- Negative

As staff, you are always allowed to submit. If you were a student, you would see the following:

*You have infinitely many submissions remaining.*
E) Suppose we are given the classifier defined by \( \theta = \begin{bmatrix} -2 \\ 5 \end{bmatrix} \) and \( \theta_0 = 0 \). How will our classifier classify the point \((1, 3)\)?

The classifier will classify the point \((1,3)\) as:

- Positive
- Negative

As staff, you are always allowed to submit. If you were a student, you would see the following:

*You have infinitely many submissions remaining.*
1.2) Representing as a matrix

We've established that one can use a linear binary classifier to predict a category, but we need a way to represent our classifier using an array. We need to know both how to represent the line itself, and also which side is designated as positive and which side is designated as negative.

So, let's start with the line...how do we represent a line as a matrix/array? Suppose we wish to represent the line $x_1 + 2 = 2x_2$ using matrices. We can represent this line using an array \( \theta \) and an offset \( \theta_0 \).

\[
\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}
\]

The equation will take the form

\[
\theta^T x + \theta_0 = 0
\]

\[
[\theta_1, \theta_2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \theta_0 = 0
\]

To derive \( \theta \) and \( \theta_0 \) from the equation...

\[
\begin{align*}
x_1 + 2 &= 2x_2 \\
x_1 - 2x_2 + 2 &= 0 \\
1x_1 + (-2)x_2 + 2 &= 0
\end{align*}
\]

\[
[1, -2] \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + 2 = 0
\]

Thus...

\[
\theta = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, \theta_0 = 2
\]
Meta 6.036 Lab 1

- Transformer
- Graph
Given a line in the form $\theta^T x + \theta_0 = 0$, where $\theta$ is a column vector $\theta = [\theta_1, \theta_2]$, convert the equation of that line into the more familiar, $mx + b = x_2$ slope-intercept form. What are $m$ and $b$ in terms of $\theta_1$, $\theta_2$, and $\theta_0$? Enter your answers as python expressions. Use `theta_1` for $\theta_1$, `theta_2` for $\theta_2$, and `theta_0` for $\theta_0$.

Formula for $m$:

As staff, you are always allowed to submit. If you were a student, you would see the following:

You have infinitely many submissions remaining.

Your entry was parsed as:

Formula for $b$:

As staff, you are always allowed to submit. If you were a student, you would see the following:

You have infinitely many submissions remaining.

Your entry was parsed as:
Given a line in the form $\theta^T x + \theta_0 = 0$, where $\theta$ is a column vector $\theta = [\theta_1, \theta_2]$, convert the equation of that line into the more familiar, $mx + b = x_2$ slope-intercept form. What are $m$ and $b$ in terms of $\theta_1$, $\theta_2$, and $\theta_0$? Enter your answers as python expressions. **Use theta_1 for $\theta_1$, theta_2 for $\theta_2$, and theta_0 for $\theta_0$**

**Formula for $m$:**
\[
\frac{\theta_1}{-\theta_2}
\]

**Formula for $b$:**
\[
\frac{\theta_0}{-\theta_2}
\]
Meta 6.036 Lab 1

- Transformer
- Graph
- Meta Learning
You are given a \( \theta, \theta_0 \), and data array \( z \) (where each column corresponds to one datapoint), and want to determine the classification of all points in \( z \).

For example, if:

\[
\begin{align*}
\theta &= \begin{bmatrix} 1 \\ -2 \end{bmatrix}, \theta_0 = 2 \\
\end{align*}
\]

\[
\begin{bmatrix} 2 & 4 & 6 & 8 \\ -2 & 4 & 1 & 1 \end{bmatrix}
\]

The correct output would be:

\[
\begin{bmatrix} -1 & -1 & 1 & 1 \end{bmatrix}
\]

Complete the following pseudocode for determining these classifications by filling in the blanks with the appropriate values.

Hint: You may use a function called \( \text{sign}(x) \), which returns \(-1\) for \( x < 0 \), \(0\) for \( x = 0 \), and \(1\) for \( x > 0 \).

As staff, you are always allowed to submit. If you were a student, you would see the following: You have infinitely many submissions remaining.
You are given a $\theta, \beta$, and data array $x$ (where each column corresponds to one datapoint), and want to determine the classification of all points in $x$.

For example, if:

$$\theta = \begin{bmatrix} 1 \\ -2 \end{bmatrix}, \beta = 2$$

$$x = \begin{bmatrix} 2 & 4 & 6 & 8 \\ -2 & 4 & 1 & 1 \end{bmatrix}$$

The correct output would be:

$$\begin{bmatrix} -1 & -1 & 1 & 1 \end{bmatrix}$$

Complete the following pseudocode for determining these classifications by filling the boxes with the appropriate values.

Hint: You may use a function called $\text{sign}(x)$, which returns $-1$ for $x < 0$, $0$ for $x = 0$, and $1$ for $x > 0$. 

As staff, you are always allowed to submit. If you were a student, you would see the following: 

You have infinitely many submissions remaining.
Meta 6.036 Lab 1

• Transformer
• Objectives
  – Masked prediction
  – Sequential
1.4) Visualizing and Building Intuition

For the questions below, please use the visualization tool below.

\[ \theta_1 | 2 \quad \theta_2 | -3 \quad \theta_0 | 0 \]

Line and Scatter Plot

- classifier
- negative
- positive
- normal vector
Probabilistic Programming Example

<table>
<thead>
<tr>
<th>Probabilistic program</th>
<th>Probabilistic outcomes</th>
</tr>
</thead>
<tbody>
<tr>
<td>var a = flip(0.3)</td>
<td>1 0 0 1 ...</td>
</tr>
<tr>
<td>var b = flip(0.3)</td>
<td>0 0 0 0 ...</td>
</tr>
<tr>
<td>var c = flip(0.3)</td>
<td>1 0 1 0 ...</td>
</tr>
<tr>
<td>return a+b+c</td>
<td>2 0 1 1 ...</td>
</tr>
</tbody>
</table>

Probability

0 1 2 3
Probabilistic Programming Example

```
probabilistic program  probabilistic outcomes

infer()
function() {
    var a = flip(0.3) 1 0 0 1 ···
    var b = flip(0.3) 0 0 0 0 ···
    var c = flip(0.3) 1 0 1 0 ···
    condition (a+b==1)  T  F  F  T
    return a+b+c 2 0 1 1 ···
}

infer a+b+c | a+b==1
rejection sampling: run program and reject return values that do not satisfy constraints (inefficient)
```

```
probability
```

```
0 1 2 3
```