

Workflow

O PyTorch

What we're going to cover A PyTorch workflow









- 5. Improve through experimentation
- 6. Save and reload your trained model



Where can you get help?

- Follow along with the code
- Try it for yourself
- Press SHIFT + CMD + SPACE to read the docstring
- Search for it
- Try again
- Ask



https://www.github.com/mrdbourke/pytorch-deep-learning/discussions

Zero to Mastery Learn PyTorch for Deep Lea

🖞 🖈 🛈 🔝 M 🗯 🚮 🚯 🚆

Table of content

2. Build model

What we're going to cove Where can can you get help

1. Data (preparing and loading

Split data into training an

PvTorch model buildin

CO Open in Col

01. PvTorch Workflow Fundamentals

arning and deep learning is to take some data from the past, build an algorithm (like

But let's start smal w about we start with a straight line?

What we're going to cove

In this module we're going to cover a standard PyTorch workflow (it can be chopped and changed as necessary bu t covers the main outline of steps)

necking the contents of a PvTorch model Making predictions us 3. Train model Creating a loss function an ptimizer in PyTorch

Motto #1: "If in doubt, run the code"

size: _size, *, generator: Generator | No

oat32 by defaul



					Mana — adva
Q	We'l	write t	he code first and then explain everything		None, pin_ _bool = Fa
<>{x}□	Let's	1 # 2 cl 3 4 5 6 7 8	<pre>te a standard linear regression model u Create a Linear Regression model (ass LinearRegressionModel(nn.Modul definit(self): super()init() self.weights = nn.Parameter(</pre>	1/8 v torc car	_bool = Fa A kind of Ten Parameters a very special p assigned as M its parameter h.randn(1, h we update cch loves f:
		9 10 11 12 13 14 15	<pre>self.bias = nn.Parameter(tor requires_grad=True, # <- dtype=torch.float # <- F))</pre>	ch.r can yTor	andn(1, # - we update ch loves f:
		16 17	<pre>def forward(self, x: torch.Tenso return self.weights * x + se</pre>	r) -	<pre>-> torch.Ter >ias # <- th</pre>
=	Alrig	ht ther	e's a fair bit going on above but let's brea	ak it c	lown bit by b
		Resou	rce: We'll be using Python classes to cre	ate b	its and piece

s for building neural networks. If you're unfamiliar with Pytho

Os completed at 08:3

O PyTorch documentation – PyTo X +	F								~
\leftarrow \rightarrow C \triangle $$ pytorch.org/docs/stable/inde	ex.html						⊕_ტ	☆ 🛈 🔽 M	🗯 🚳 E
() PyTorch	Get								
	Started	Ecosystem	Mobile	Blog	Tutorials	Docs ∨ ●	Reso	ources 🗸	GitHul
1.10 🔻	Docs > PyTorcl	n documentation				Edit o	n GitHub	Shortcuts	
Q Search Docs	ργτο	RCH DOC	CUMEN	ΤΑΤΙΟ	NC			PyTorch docun Indices and tab	nentation les
Notes [+]	PyTorch is ar	yTorch is an optimized tensor library for deep learning using GPUs and CPUs.							
Language Bindings [+]	Features des	Features described in this documentation are classified by release status:							
Python API [-]	Sta	Stable: These features will be maintained long-term and there							
torch	doc	umentation. We als	o expect to ma	intain backv	wards				
torch.nn	con	npatibility (although	n breaking chan	ges can hap	pen and notice				
torch.nn.functional	will	be given one releas	e ahead of time	e).					
torch.Tensor	Bet	a: These features ar	e tagged as Be	a because t	he API may				
Tensor Attributes	cha	nge based on user f	eedback, becau	use the perfo	ormance needs				
Tensor Views	to i	mprove, or because	e coverage acro	ss operators	s is not yet				
torch.autograd	con	nplete. For Beta fea	tures, we are co	ommitting to	o seeing the				
torch.cuda	feat	ture through to the	Stable classific	ation. We ar	e not, however,				
torch.cuda.amp	con	nmitting to backwar	ros compatibilit	y.					
torch.backends	Pro	totype: These featu	res are typicall	y not availab	ole as part of				
torch.distributed	bina	ary distributions like	e PyPI or Conda	, except sor	metimes behind				
torch.distributed.algorithms.join	run	-time flags, and are	at an early stag	e for feedba	ack and testing.				





Let's code.



Daniel Bourke @mrdbourke · Nov 1 "How do I learn #machinelearning?"

What you want to hear: 1. Learn Python Learn Math/Stats/Probability Learn software engineering 4. Build

What you need to do:

- 1. Google it
- 2. Go down the rabbit hole
- Resurface in 6-9 months and reassess

See you on the other side.

Inputs

Numerical encoding

[[116, 78, 15],

[117, 43, 96],

[125, 87, 23],

••• 9



Part 1: Turn data into numbers

Part 2: Build model to learn patterns in numbers



Inputs

Numerical encoding



Model learns patterns from here



Course materials (training set)

Generalization

The ability for a machine learning model to perform well on data it hasn't seen before.

Three datasets

(possibly the most important concept in machine learning...)



```
•••
 1 # Create a linear regression model in PyTorch
   class LinearRegressionModel(nn.Module): 
      def ___init__(self):
          super().__init__()
          # Initialize model parameters
          self.weights = nn.Parameter(torch.randn(1,
              requires_grad=True,
 8
              dtype=torch.float
 9
          ))
10
11
          self.bias = nn.Parameter(torch.randn(1,
12
              13
14
              dtype=torch.float
15
16
        forward() defines the computation in the model
     def forward(self, x: torch.Tensor) -> torch.Tensor:
18
19
           return self.weights * x + self.bias
20
```

Subclass nn. Module (this contains all the building blocks for neural networks)

Initialise model parameters to be used in various computations (these could be different layers from torch.nn, single parameters, hard-coded values or functions)

requires_grad=True means PyTorch will track the gradients of this specific parameter for use with torch.autograd and gradient descent (for many torch.nn modules, requires_grad=True is set by default)

Any subclass of nn. Module needs to override forward() (this defines the forward computation of the model)







PyTorch essential neural network building modules

PyTorch module

Contains all of the building blocks for computational graphs (essentially a series of computations executed in a particular way).

<u>torch.nn.Module</u>

The base class for all neural network modules, all the building blocks for neural networks are subclasses. If you're building a neural network in PyTorch, your models should subclass nn.Module.Requires a forward() method be implemented.

<u>torch.optim</u>

Contains various optimization algorithms (these tell the model parameters stored in <u>nn.Parameter</u> how to best change to improve gradient descent and in turn reduce the loss).

<u>torch.utils.data.Dataset</u>

Represents a map between key (label) and sample (features) pairs of your data. Such as images and their associated labels.

Creates a Python iterable over a torch Dataset (allows you to iterate over your data).

<u>torch.utils.data.DataLoader</u>

See more: <u>https://pytorch.org/tutorials/beginner/ptcheat.html</u>

What does it do?





See more: <u>https://pytorch.org/tutorials/beginner/ptcheat.html</u>



Mean absolute error (MAE)



MAE_loss = torch.mean(torch.abs(y_pred-y_test)) or $MAE_loss = torch.nn.L1Loss$

See more: <u>https://pytorch.org/tutorials/beginner/ptcheat.html#loss-functions</u>



Daniel Bourke @mrdbourke

Let's sing the @PyTorch optimization loop song

It's train time! do the forward pass, calculate the loss, optimizer zero grad, lossss backwards!

Optimizer step step step

Let's test now! with torch no grad: do the forward pass, calculate the loss, watch it go down down down!

Source: @mrdbourke Twitter & see the video version on YouTube.

PyTorch training loop

.......

•	<pre>1 # Pass the data through the model for a number of epochs (e.g. 100) 2 for epoch in range(epochs):</pre>
	<pre>3 4 # Put model in training mode (this is the default state of a model) 5 model.train() 6</pre>
	<pre>7 # 1. Forward pass on train data using the forward() method inside 8 y_pred = model(X_train) 9</pre>
1	<pre># 2. Calculate the loss (how different are the model's predictions to the true loss = loss_fn(y_pred, y_true)</pre>
1 1 1	<pre># 3. Zero the gradients of the optimizer (they accumulate by default) 4 optimizer.zero_grad()</pre>
1 1 1	.5 .6 # 4. Perform backpropagation on the loss .7 loss.backward()
1 1 2	<pre>18 19 # 5. Progress/step the optimizer (gradient descent) 20 optimizer.step()</pre>

Note: all of this can be turned into a function

Pass the data through the model for a number of epochs (e.g. 100 for 100 passes of the data)

Pass the data through the model, this will perform the forward() method located within the model object

Calculate the loss value (how wrong the model's predictions are)

Zero the optimizer gradients (they accumulate every epoch, zero them to start fresh each forward pass)

Perform backpropagation on the loss function (compute the gradient of every parameter with requires grad=True)

Step the optimizer to update the model's parameters with **respect to the gradients calculated by** loss.backward()





PyTorch testing loop

```
# Setup empty lists to keep track of model progress
  epoch_count = []
  train_loss_values = []
  test loss values = []
6 # Pass the data through the model for a number of epochs (e.g. 100) pochs):
 7 for epoch in range(epochs):
      ### Training loop code here ###
10
      ### Testing starts ###
        Put the model in evaluation mode
      ⊓model.eval()
     # Turn on inference mode context manager
     with torch.inference_mode():
    # 1. Forward pass on test data
18
       test_pred = model(X_test)
10
        # 2. Caculate loss on test data
        test_loss = loss_fn(test_pred, y_test)
           Print out what's happening every 10 epochs
        epoch % 10 == 0:
25
          epoch_count.append(epoch)
26
          train_loss_values.append(loss)
          test_loss_values.append(test_loss)
28
                   poch: {epoch} | MAE Train Loss: {loss} | MAE
```

Note: all of this can be turned into a function

See more: <u>https://discuss.pytorch.org/t/model-eval-vs-with-torch-no-grad/19615</u> & <u>PyTorch Twitter announcement of torch.inference_mode()</u>

Create empty lists for storing useful values (helpful for tracking model progress)

Tell the model we want to evaluate rather than train (this turns off functionality used for training but not evaluation)

(faster performance!)

Turn on torch. inference mode() context manager to disable functionality such as gradient tracking for inference (gradient tracking not needed for inference)

Pass the test data through the model (this will call the model's implemented forward() method)

Calculate the test loss value (how wrong the model's predictions are on the test dataset, lower is better)

Display information outputs for how the model is doing during training/testing every ~10 epochs (note: what gets) printed out here can be adjusted for specific problems)



```
1 # Setup optimization loop(s)
 2 \text{ epochs} = 10000
                     4 ### Train time!
 5 # Loop through the epochs
 6 for epoch in range(epochs):
    # Set the model to train mode (this is the default)
     model.train()
 8
9
     # 1. Do the forward pass
10
    y_pred = model(X_train)
11
12
     # 2. Calculate the loss (how wrong the model is)
13
    loss = loss fn(y pred, y train)
14
15
     # 3. Zero the optimizer gradients (they accumulate by default)
16
     optimizer.zero_grad()
17
18
     # 4. Perform backpropagation (with respect to the model's parameters)
19
     loss.backward()
20
21
    # 5. Step the optimizer (gradient descent)
22
23
     optimizer.step()
24
         Test time!
25
    # Set the model to eval mode (this turns off settings not needed for testing)
26
     model.eval()
27
28
     # Turn on inference mode context manager (removes even more things not needed for inference)
     with torch.inference_mode():
29
      # 1. Do the forward pass
30
31
       test pred = model(X test)
32
       # 2. Calculate the loss
33
       test loss = loss fn(test pred, y test)
34
35
36 # Print out what's happenin'!
    print(f"Epoch: {epoch} | Train loss: {loss:.4f} | Test loss: {test_loss:.4f}")
37
```



```
1 # Train function
 2 def train_step(model, loss_fn, optimizer, data, labels):
    # Turn on train mode (this is default but we turn it on anyway)
    model.train()
    # 1. Forward pass
    y_pred = model(data)
    # 2. Calculate the loss
    loss = loss_fn(y_pred, labels)
    # 3. Zero optimizer gradients
    optimizer.zero_grad()
10
    # 4. Perform backpropagation
11
    loss.backward()
12
    # 5. Perform gradient descent
13
    optimizer.step()
14
    return loss
```

```
1 # Test function
  def test_step(model, loss_fn, data, labels):
     # Turn on evaluation mode
    model.eval()
    # Setup inference mode context manager
    with torch.inference_mode():
 6
      # 1. Forward pass
      test_pred = model(data)
      # 2. Calculate the loss
 0
       test_loss = loss_fn(test_pred, labels)
10
     return test_loss
```



```
1 # Create a linear regression model in PyTorch
   class LinearRegressionModel(nn.Module):
       def __init__(self):
            super().__init__()
           # Initialize model parameters
            self.weights = nn.Parameter(torch.randn(1,
                requires_grad=True,
                dtype=torch.float
 9
10
            ))
            self.bias = nn.Parameter(torch.randn(1,
12
                requires_grad=True,
13
                dtype=torch.float
14
15
            ))
16
       # forward() defines the computation in the model
       def forward(self, x: torch.Tensor) -> torch.Tensor:
    return self.weights * x + self.bias
18
19
20
```

Linear regression model with nn. Parameter



Linear regression model with nn.Linear

Supervised learning

(overview)

1. Initialise with random weights (only at beginning)



[[116, 78, 15], ____ [117, 43, 96], ____ [125, 87, 23],

4. Repeat with more examples

••• •

Inputs

Numerical encoding

Learns representation (patterns/features/weights)

[[0.092, 0.210, 0.415],[0.778, 0.929, 0.030],[0.019, 0.182, 0.555],



[[0.983, 0.004, 0.013],Ramen, ▶[0.110, 0.889, 0.001], **__** Spaghetti [0.023, 0.027, 0.985],

3. Update representation outputs

> Representation outputs



Outputs



Daniel Bourke @mrdbourke · Nov 1 "How do I learn #machinelearning?"

What you want to hear: 1. Learn Python 2. Learn Math/Stats/Probability 3. Learn software engineering Build

What you need to do:

1. Google it

- 2. Go down the rabbit hole
- 3. Resurface in 6-9 months and reassess

See you on the other side.

(before data gets used with an algorithm, it needs to be turned into numbers)

[[116.] 15], 78 [117, 43, 96] [125, 87, 23],

••• 9

Inputs

Numerical encoding

(patterns/features/weights)





Inputs

Numerical encoding

Learns representation (patterns/features/weights)



Representation outputs

Outputs





How to approach this course

1	# 1. Construct a model class that subclasses nn.Module
2	<pre>class CircleModelV0(nn.Module):</pre>
3	<pre>definit(self):</pre>
4	<pre>super()init()</pre>
5	# 2. Create 2 nn.Linear layers
6	self.layer_1 = nn.Linear(in_features=2, out_features=5)
7	self.layer_2 = nn.Linear(in_features=5, out_features=1)
8	
9	# 3. Define a forward method containing the forward pass computation
10	<pre>def forward(self, x):</pre>
11	# Pass the data through both layers
12	<pre>return self.layer_2(self.layer_1(x))</pre>
13	
14	# 4. Create an instance of the model and send it to target device
15	<pre>model_0 = CircleModelV0().to(device)</pre>
16	model_0





1. Code along

Motto #1: if in doubt, run the code!



(including the "dumb" ones)

4. Ask questions

2. Explore and experiment



5. Do the exercises

Motto #3: visualize, visualize, visualize!



3. Visualize what you don't understand



6. Share your work

How not to approach this course



Avoid:

"I can't learn "

This course

				Ś	
Search or jump to	Pull requests Issu	es Marketplace Exploi	re		↓ +• � • •
💂 mrdbourke / pytorch-dee	ep-learning Public		⊙ Unwatch 4 →	양 Fork 11	Starred 76 -
<> Code · Issues 28	1) Pull requests 🛛 🖓 Discussion	ns 🕑 Actions 🖽	Projects 1	Wiki 🕛 Security	🗠 Insights 🛛 …
<mark>ਇ main →</mark> ਇ 3 branches 🤇	>0 tags	Go to file Add fi	le • Code •	About	ŝ
mrdbourke update exercises		✓ 4a92ab9 2 days ago	140 commits	Materials for upcomi PyTorch course (wor	ing beginner-friendly k in progress).
.github/workflows	Update make_docs.yml		19 days ago	♂ learnpytorch.io	
docs	update readme and docs homep	age	9 days ago	machine-learning d	eep-learning
extras	add exercises and solutions for (01	12 days ago	pytorch	
going_modular	update datasetup		2 months ago	Readme	
images	add ztm and pytorch logo		14 days ago	▲ MIT License	
in models	add folder for trained models		4 months ago	 4 watching 	
🗅 .gitignore	update gitignore		23 days ago	양 11 forks	
00_pytorch_fundamentals.i	cleanup annotations, add exercis	ses and solutions links	20 days ago		
01_pytorch_workflow.ipynb	fix typo		2 days ago	Releases	
02_pytorch_classification.ip	update exercises		2 days ago	No releases published Create a new release	
03_pytorch_computer_visio	update 03		3 months ago		

https://www.github.com/mrdbourke/pytorch-deep-learning

	O PyTorch
<u>PyTorch website</u> & <u>forums</u>	FROM RESE DRODUC
All things PvTorch	< PyTorch

https://www.github.com/mrdbourke/pytorch-deep-learning/ discussions



$\bullet \bullet \bullet \quad \Box \lor \prec \quad \lor \quad \bigtriangledown \quad \boxdot \quad \textcircled{0} \quad \textcircled{0} \quad \textcircled{0}$

🛱 mrdbou	rke / py	torch	n-de	ep-	lear	ning	Pu	blic
<> Code	⊙ Issue	es 28		îΊ Ρ	ull red	quest	S	Q) D
+								
+ +								
Announce	ements +							
Welco	me to p	ytoro	ch-c	leep	-lea	rnin	ng D	iscu
😽 mrdl	bourke +							
Q Search a	l discussio	ons						
Categories				6	, Di	scus	sion	s
∞ View all						(† 2		6
≽ Annour	ncements							
💬 Genera	I							
💡 Ideas								
🙏 Q&A								

Search or jump to.

Resources

Course Q&A



Course online book

••• [] · < > 🛛 🕜 🔘 🗉	earnpytorch.io	5	④ ① +
	Q Searc	sh	Mrdbourke/pytorch_dee ☆ 76 ぷ 11
Welcome to the Learn PyTorch for De progress)	ep Learning boc	ok (work in	
This will be the homepage for the online book version of the Zero to Mas	stery Learn PyTorch for Deep Le	earning course.	
This course will teach you foundations of deep learning and PyTorch (a	deep learning framework writte	n in Python).	
The course is video based. However, the videos are based on the conten	ts of this online book.		
For full code and resources see the course GitHub.			
Expected release date: Early 2022.			
Get updates: Follow the pytorch-deep-learning repolog or sign up for	or emails.		
		00. F	_{Next} PyTorch Fundamentals
Made with Material for MkDocs			y q

https://learnpytorch.io

	●●●	■ pytorch.org	· 1 + :::		. disc	cuss.pytorch.org	⊕ ¹ + :::	•••
	O PyTorch			O PyTorch		Sign Up		
			Alex -	all categories Categories Latest Top				
	FROM			Category	Topics	Latest		
Forch website &	RESEARCH	TO		Vision Topics related to either pytorch/vision or vision research related topics kornia	141 / month	Model train, val, test workflow verification in PyTorch vision	1 24m	
<u>forums</u>	PRODUCTION			nlp Topics related to Natural Language Processing	38 / month	Input, output must be on the current device	0 34m	
	An open source machine learning framework that accelerate the path from research prototyping to production deployme	es lent.		Uncategorized Topics that don't need a category, or don't fit into any other existing category.	361 / month	Top 2 accuracy ValueError: Number of classes in 'y_true' (10) not equal to the number of classes in 'y_score' (11)	1 1h	
	Install >			autograd A category of posts relating to the autograd engine itself.	70 / month	Calculate differential of a tensor before los	s 1 1h	
				mixed-precision	7 / month	The loss is not decreasing	0 1h	
	PyTorch 1.10 Release, including CUE	DA Graphs APIs, TorchScript improvements	>	C++ Topics related to the C++ Frontend, C++ API or C++ Extensions	37 / month	How to only sample data from a dataloade with target == class	- 1 1h	
things PvTorch				distributed	35 / month			





Daniel Bourke @mrdbourke

It's train time! do the forward pass, calculate the loss, optimizer zero grad, losssss backwards!

Optimizer step step step

Let's test now! with torch no grad: do the forward pass, calculate the loss, watch it go down down down!

Let's sing the @PyTorch optimization loop song