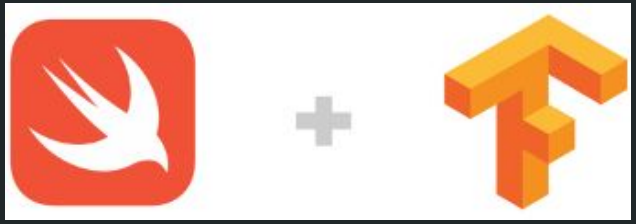


Swift for Tensorflow

Initial observations

What's the future of fastai?

- *Write custom GPU kernels in Swift*
- *Differentiate any arbitrary code*

Pace of progress:

- pytorch was created because of a gap in tensorflow
- fastai was created because of a gap in tooling for pytorch
- ...but now we're hitting the limits of python
- so we need to jump this gap too - so we're working towards Swift 4 TF

🐍 Python is nice... but:

- **Slow:** forces things into external C libraries - see lesson 8!
- **Concurrency:** GIL forces more into external C libraries
- **Accelerators:** forces more into CUDA, etc.

Tooling (for Linux)

Swift setup on Linux - scripts on fastai - may take some (or a lot) of debugging

Swift-jupyter

$ conda activate swiftai
$ jupyter notebook

```
In [1]: %install-location $cwd/swift-install
        %install '.package(path: "$cwd/FastaiNotebook_00_load_data")' FastaiNotebook_00_load_data

Installing packages:
        .package(path: "/mnt/963GB/Data/Python/Courses/fastai/course-v3/nbs/swift/FastaiNotebook_00_l
                FastaiNotebook_00_load_data
With SwiftPM flags: []
Working in: /tmp/tmpfxs15dgt/swift-install
[1/2] Compiling jupyterInstalledPackages jupyterInstalledPackages.swift
[2/3] Merging module jupyterInstalledPackages
Initializing Swift...
Installation complete!
```

VScode + Swift Language Server Protocol (sourcekit-lsp)-> jump to, autocomplete

CLion: direct swift package manager support -> autocomplete, debug, test

Proprietary Apple framework

Access all functions using unqualified names

Adds properties to existing types.

Struct vs Class

Core functionality including data storage and persistence, text processing, date and time calculations, sorting and filtering, and networking.

Sequence protocol
Collection protocol

Sourcekit-lsp gives some useful information

Classic Computer Science Problems in Swift.playground > Pages > Chapter 7.xcplaygroundpage > Contents.swift > ...

```swift
// limitations under the License.

/// Fairly Simple Neural Networks

// mac only
// import Accelerate
import Foundation

// MARK: Randomization & Statistical Helpers

// A derivative of the Fisher-Yates algorithm to shuffle an array
extension Array {
    public func shuffled() -> Array<Element> {
        var shuffledArray = self // value semantics (Array is Struct) makes this a copy
        if count < 2 { return shuffledArray } // already shuffled
        for i in (1..<count).reversed() { // count backwards
            let position = Int(arc4random_uniform(UInt32(i + 1))) // random to swap
            if i != position { // swap with the end, don't bother with self swaps
                shuffledArray.swapAt(i, position)
            }
        }
        return shuffledArray
    }
}

struct Random {
    private static var seeded = false

    // a random Double between *from* and *to*, assumes *from* < *to*
    static func double(from: Double, to: Double) -> Double {
        if !Random.seeded {
            srand48(time(nil))
```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL

Contents.swift Classic Computer Science Problems in Swift.playground/Pages/Chapter 7.xcplaygroundpage 8

use of unresolved identifier 'arc4random_uniform' sourcekitd [33, 32]

https://github.com/tensorflow/swift-apis

In [1]:
```
import TensorFlow
```

In [2]:
```
let hiddenSize: Int = 10
```

In [3]:
```
struct Model: Layer {
    var layer1 = Dense<Float>(inputSize: 4, outputSize: hiddenSize, activation: relu)
    var layer2 = Dense<Float>(inputSize: hiddenSize, outputSize: hiddenSize, activation: relu)
    var layer3 = Dense<Float>(inputSize: hiddenSize, outputSize: 3, activation: identity)

    @differentiable
    func callAsFunction(_ input: Tensor<Float>) -> Tensor<Float> {
        return input.sequenced(through: layer1, layer2, layer3)
    }
}
```

Attribute that tells compiler to differentiate the fn: guarantees it's differentiable

In [10]:
```
var classifier = Model()
let optimizer = SGD(for: classifier, learningRate: 0.02)
Context.local.learningPhase = .training
let x: Tensor<Float> = [[3,2,1, 4], [5,2,7, 2]]
let y: Tensor<Int32> = [1,2]
```

One way to define a training epoch is to use the `Differentiable.gradient(in:)` method.

In [11]:
```
for _ in 0..<1000 {
    let ∇model = classifier.gradient { classifier -> Tensor<Float> in
        let ŷ = classifier(x)
        let loss = softmaxCrossEntropy(logits: ŷ, labels: y)
        print("Loss: \(loss)")
        return loss
    }
    optimizer.update(&classifier, along: ∇model)
}
```

```
Loss: 0.8170202
```

struct vs class

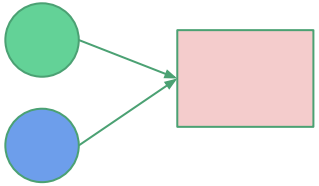
Used more often

Can’t subclass

Value semantics

Class

Reference semantics



```
struct SomeStruct {
    var name: String
    init(name: String) {
        self.name = name
    }
}

var aStruct = SomeStruct(name: "Bob")
var bStruct = aStruct // aStruct and bStruct are two
bStruct.name = "Sue"

println(aStruct.name) // "Bob"
println(bStruct.name) // "Sue"
```

```
class SomeClass {
    var name: String
    init(name: String) {
        self.name = name
    }
}

var aClass = SomeClass(name: "Bob")
var bClass = aClass // aClass and bClass now reference the same
bClass.name = "Sue"

println(aClass.name) // "Sue"
println(bClass.name) // "Sue"
```

closures

```
numbers.map({ (number: Int) -> Int in
    let result = 3 * number
    return result
})
```

```
{ ( parameters ) -> return type in
    statements
}
```

Surround code with {}

Separate args and return type with ‘in’

```
var someClosure = {(days: Int, name: String) -> String in
    return "\(name), closures are coming for you in \(days) days" }
```

```
someClosure(2,"Five")
```

```
"Five, closures are coming for you in 2 days"
```

Similar to python lambdas

```
x = lambda a, b : a * b
print(x(5, 6))
```

protocol and extension

Protocol: similar to a Java interface/abstract methods and classes - a contract that implementation must adhere to

```
protocol Person{
  func getWage()-> Int
}
```

The contract (no implementation)

Worker adheres to person protocol ('contract')

```
class Worker:Person {
  var name: String = ""
  var age : Int = 0

  init(name: String, age: Int){
    self.name = name
    self.age = age
  }

  func getWage() -> Int{
    return self.age * 10
  }
}
```

```
let p = Worker(name:"Zaid", age: 3)
p.getWage()
```

```
30
```

```
struct Complex{
  var real:Float
  var imag:Float
}

Complex(real:3 , imag: 2)
```

Can extend the struct in a principled way -adds functionality to existing type

```
extension Complex{
  func getReal() -> Float{
    return real
  }
}

var a = Complex(real:3 , imag: 2)
print(a.getReal())
```

```
3.0
```

generics

T is a placeholder for any type

But is must implement the SignedNumeric protocol

```
struct Complex<T : SignedNumeric> {
    var real, imag : T

    // This is a read only computed property.
    var conj : Complex { return Complex(real: real, imag: -imag) }

    // Here's a computed property with a setter, that returns the imaginary
    // component negated, just to show how to do this.  A more realistic
    // use case would be to provide a polar coordinate projection.
    var imagNegated : T {
        get { return -imag }
        set { imag = -newValue }
    }
}
```

PythonObject

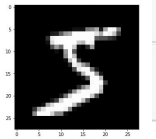
```
import Python
```

```
public let np = Python.import("numpy")
public let plt = Python.import("matplotlib.pyplot")
```

```
import TensorFlow
let (xTrain, yTrain, xValid, yValid) = loadMNIST(path: mnistPath, flat: true)
```

```
let img = xTrain[0].makeNumpyArray().reshape(28, 28)
```

```
plt.figure(figsize: [5,5])
plt.show(plt.imshow(X: img, cmap: "gray"))
```



Intercept all calls to a python object (eg x())

Handle member lookups (eg x.y)

```
@dynamicCallable
@dynamicMemberLookup
@frozen
public struct PythonObject {
```

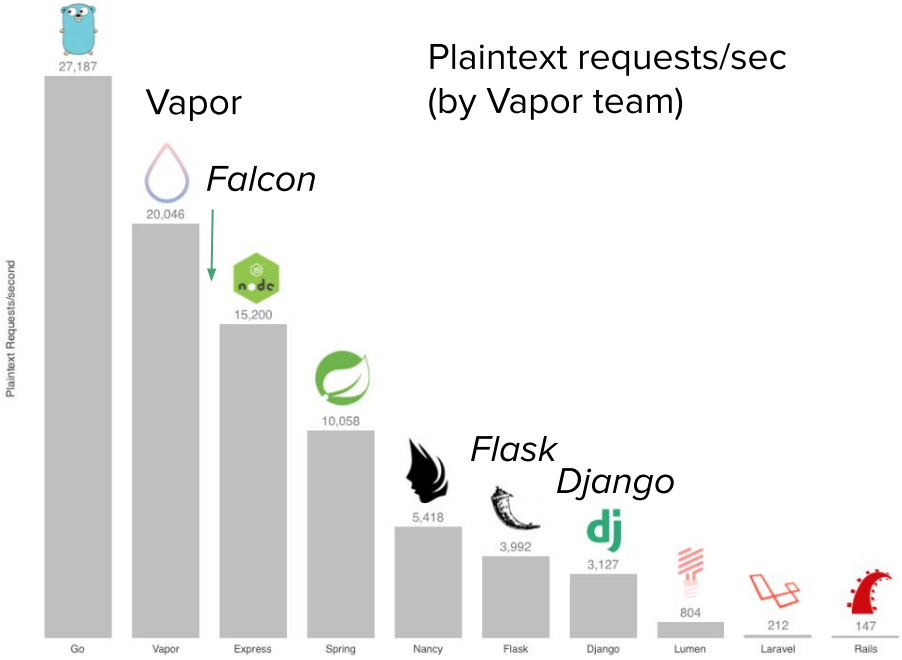
`PythonObject` represents an object in Python and supports dynamic member lookup. Any member access like `object.foo` will dynamically request the Python runtime for a member with the specified name in this object.

`PythonObject` is passed to and returned from all Python function calls and member references. It supports standard Python arithmetic and comparison operators.

Internally, `PythonObject` is implemented as a reference-counted pointer to a Python C API `PyObject`.

https://github.com/apple/swift/blob/tensorflow/stdlib/public/Python/Python.swift

Server-side swift: Webapps



Perfect webapp

EXPLORER

OPEN EDITORS 2 UNSAVED
- Package.swift
- main.swift Sources/PerfectTempl...

PERFECTTEMPLATE
- .build
- Sources
  - PerfectTemplate
    - main.swift
- .gitignore
- LICENSE
- LICENSE.zh_CN
- Package.resolved
- Package.swift
- README.md
- README.zh_CN.md

Package.swift ● main.swift ●

Sources > PerfectTemplate > main.swift

```swift
import PerfectHTTP
import PerfectHTTPServer
import MongoSwift

let client = try MongoClient("mongodb://localhost:27017")
let db = client.db("myDB")
let collection = try db.createCollection("myCollection")

// An example request handler.
func handler(request: HTTPRequest, response: HTTPResponse) {
    let query: Document = ["a": 1]
    let documents = try collection.find(query)
    response.setHeader(.contentType, value: "text/html")
    var text = "<html><title>Hello, world!</title><body>"
    for doc in documents {
        text = text + doc
    }
    text = text + "</body></html>")
    response.appendBody(string: text)
    response.completed()
}


var routes = Routes()
routes.add(method: .get, uri: "/", handler: handler)
routes.add(method: .get, uri: "/**",
           handler: StaticFileHandler(documentRoot: "./webroot", allowResponseFilters: true).handleRequest)
try HTTPServer.launch(name: "localhost",
                      port: 8181,
                      routes: routes,
                      responseFilters: [
                        (PerfectHTTPServer.HTTPFilter.contentCompression(data: [:]), HTTPFilterPriority.high
```

Whats difficult

Dead kernels

Inscrutable errors

```
error: Couldn't lookup symbols:
  AD__$s14__lldb_expr_537FALayerP02__a1_B3_59E14callAsFunctiony6OutputQz5InputQzF__jvp_s
```

Swift package manager - how to version dependencies?

Code installs

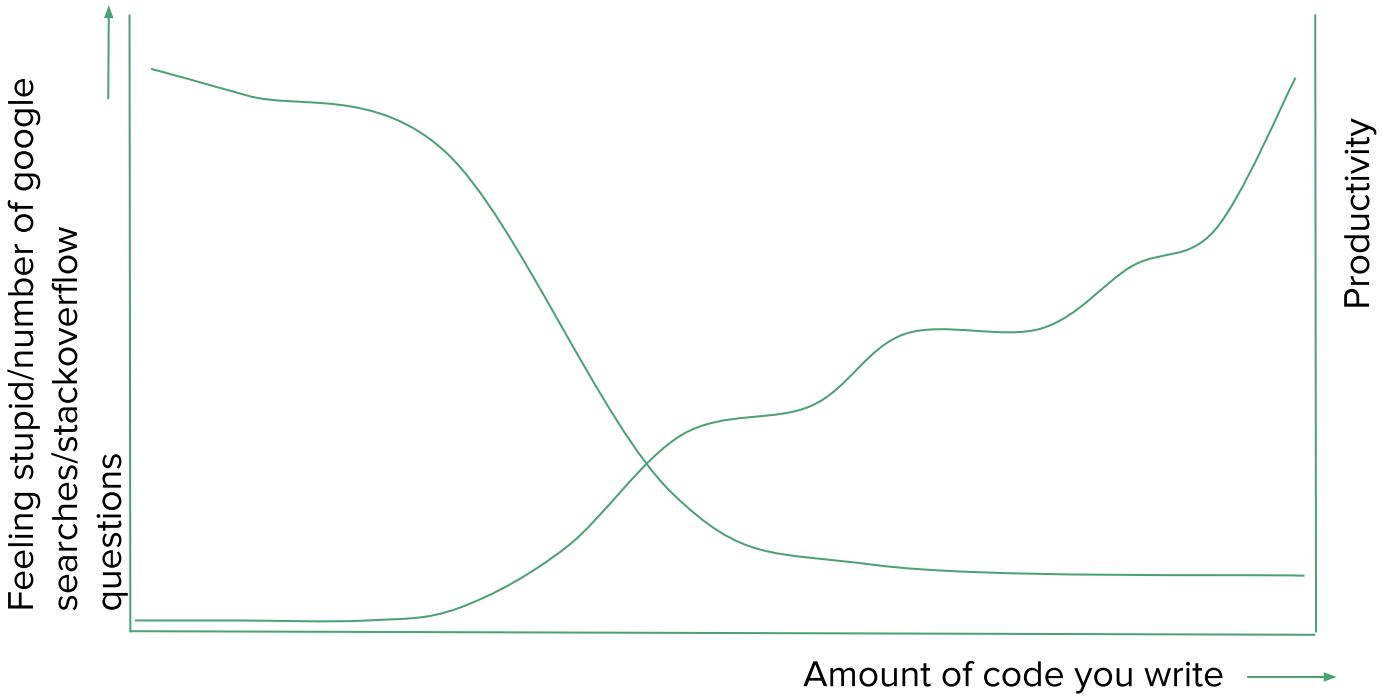
install.sh

```
OSABR=$(echo $(uname)|tr '[:upper:]' '[:lower:]')
VERSION=1.4.0
DWN=/tmp/libtensorflow.tgz
URL=https://storage.googleapis.com/tensorflow/libtensorflow/libtensorflow-cpu-$OSABR-x86_64-$VERSION.tar.gz
echo $URL
wget $URL -O $DWN
tar xvf $DWN -C /usr/local ./lib/libtensorflow.so ./lib/libtensorflow_framework.so
rm -f $DWN
echo 'download AI model'
wget https://storage.googleapis.com/download.tensorflow.org/models/inception5h.zip -O /tmp/in.zip
echo 'unzip model files'
unzip /tmp/in.zip -d /tmp
swift build
```

Developer

Xcode

Learning a new language

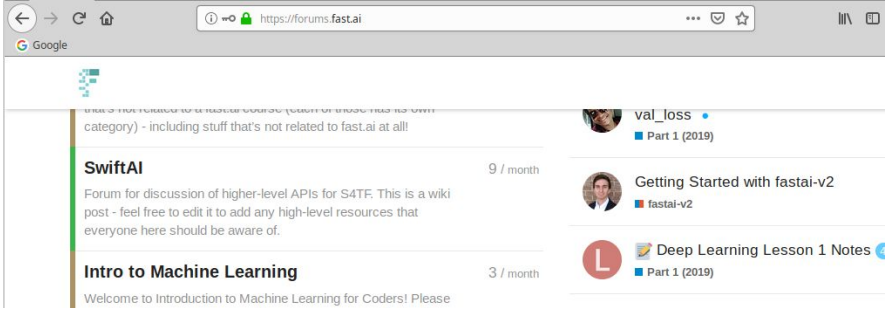


Final comments

“In the data science world, we’re mainly stuck using either R (which is the least pleasant language I’ve ever used, but with the most beautifully designed data munging and plotting libraries anywhere) or Python (which is painfully slow, very hard to parallelize, but is extremely expressive and has the best deep learning libraries available). We really need another option. Something that is fast, flexible, and provides good interop with existing libraries.” Jeremy Howard (2019)

“Overall, the Swift language itself looks to be exactly what we need…This is a really good place to be spending time if you’re interested in being part of something that has a huge amount of potential” Jeremy Howard (2019)

Links



Swift install scripts:

https://gist.github.com/adriangrepo/28df6977e30062104717b11542939838

https://gist.github.com/eliask/2d674aae83ca5f36c921097a5efd85ae

Sourcekit-lsp installation for VScode on Linux:

https://medium.com/@pvzig/swift-development-with-visual-studio-code-on-linux-99cac3918582

Swift intro on colab:

https://colab.research.google.com/github/zaidalyafeai/Notebooks/blob/master/TF_Swift.ipynb