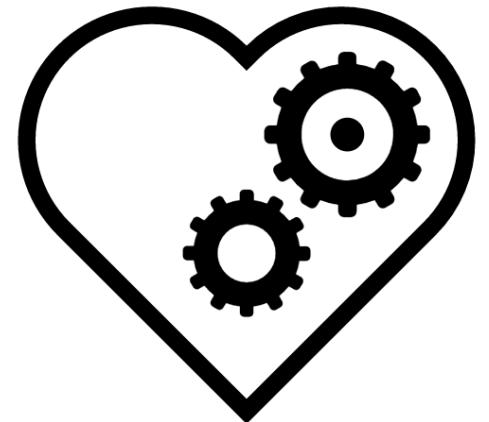


Tech Names as a Service and the Joy of Lambda



Vincent D. Warmerdam
koaning.io - fishnets88 - GoDataDriven

Today

In this talk I will discuss the maths behind a hobby project.

The first half of the talk will focus on the machine learning part of what I did and what I can do better.

The second part of the talk will focus on the architecture/cloud part of what I did and what I can do better.

The latter part of the talk will focus on improvements and general architecture for the future.

Who is this guy?

3 years @ GDD
koaning.io

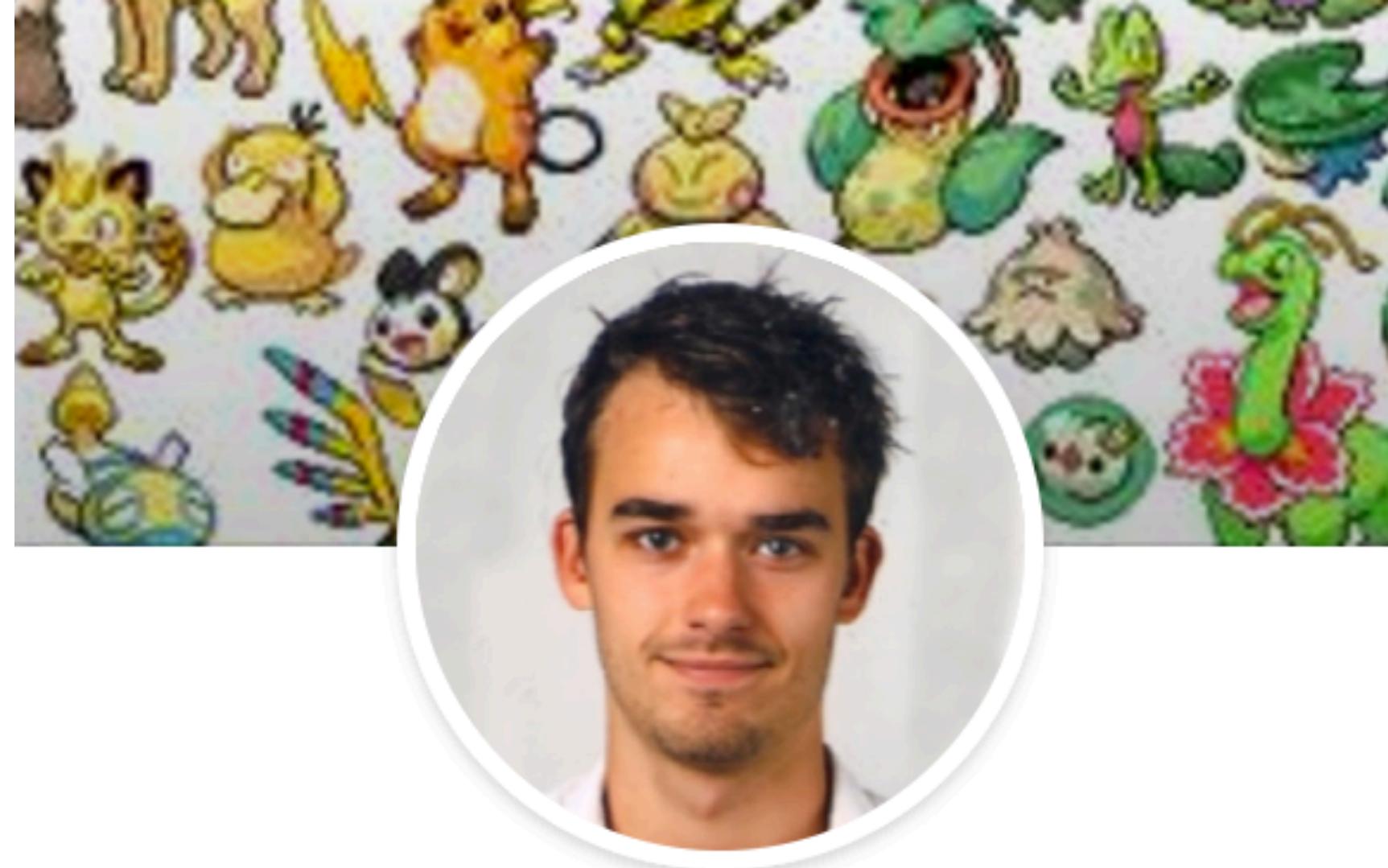
PyData Amsterdam

Rstudio partner

Machine Learning Meetup
free open sessions in data

11 endorsements for awesomeness

I write code,
I solve data problems,
ask me anything.



Vincent Warmerdam

Pokemon Master at GoDataDriven

GoDataDriven • Vrije Universiteit Amsterdam

Amsterdam Area, Netherlands • 500+ 

I'm also this guy

my linkedin profile

R, python, javascript, shiny, dplyr, purrr, ditto,
ggplot, d3, canvas, spark, sawk, pyspark, sparklyR,
lodash, lazy, bootstrap, jupyter, vulpix, git,
flask, numpy, pandas, feebas, scikit, pgm, bayes,
h2o.ai, sparkling-water, tensorflow, keras, onyx,
ekans, hadoop, scala, unity, metapod, gc, c#/c++,
krebase, neo4j, hadoop.

Outsourcing Creativity

This whole thing started when I realized that recruiters have a hard time distinguishing pokemon names against names from the coding ecosystem.

I figured making a small web-app that can generate tech names as a service may be a great adventure for fun and profit.

Outsourcing Creativity

It is a general problem if you think of it. Given a sequence of tokens, can we generate a sequence of tokens that are different but believably similar?

Outsourcing Creativity

Things like;

- pokemon

Outsourcing Creativity

Things like;

- pokemon
- red hot chili pepper lyrics

Outsourcing Creativity

Things like;

- pokemon
- red hot chili pepper lyrics
- ikea furniture names

Outsourcing Creativity

Things like;

- pokemon
- red hot chili pepper lyrics
- ikea furniture names
- notes on a piano

Outsourcing Creativity

Things like;

- pokemon
- red hot chili pepper lyrics
- ikea furniture names
- notes on a piano
- anything ipsum

Outsourcing Creativity

To keep things simple I ventured to make an app that just generates names for now.

You can find the webapp over at tnaas.com.

It is an acronym for Tech Names as a Service.

Outsourcing Creativity

To keep things simple I ventured to make an app that just generates names for now.

You can find the webapp over at tnaas.com.

It is an acronym for Tech Names as a Service.

The whole point of the app is to find a better name for it.

TNaaS Features

You can select a corpus to generate from.

You can select a model to apply.

You can click 'go'.

TNaaS Bonus Features

You can select a corpus to generate from.

You can select a model to apply.

You can click 'go'.

Bonus!

Complete a set of tokens; ***base, mo***, ***ly

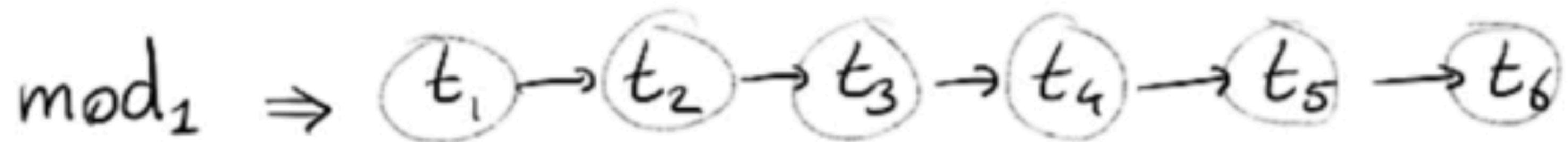
Part 1

Machine Learning

A Sequence of Tokens

Mathematically it all sounds very similar to a markov chain. The simplest markov chain is drawn below and is defined via a probability distribution:

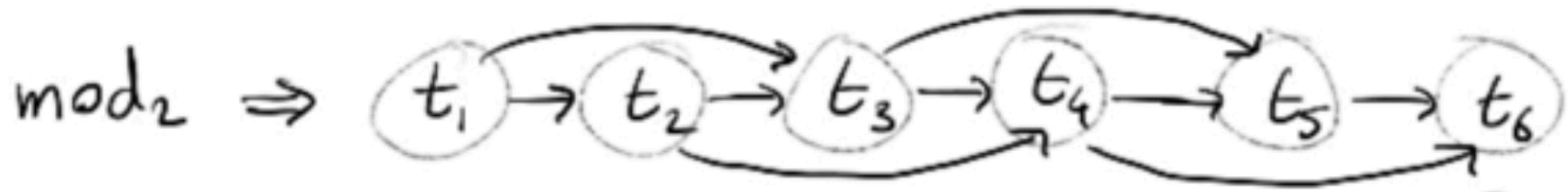
$$p(t_i | t_{i-1})$$



A Sequence of Tokens

A more expressive method might be to not just look at the previous state, but perhaps the state before as well.

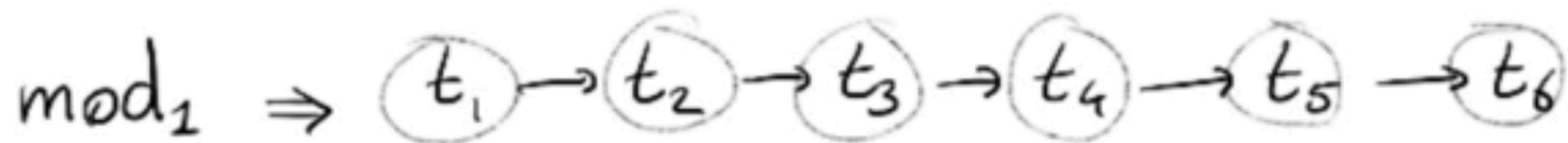
$$p(t_i | t_{i-1}, t_{i-2})$$



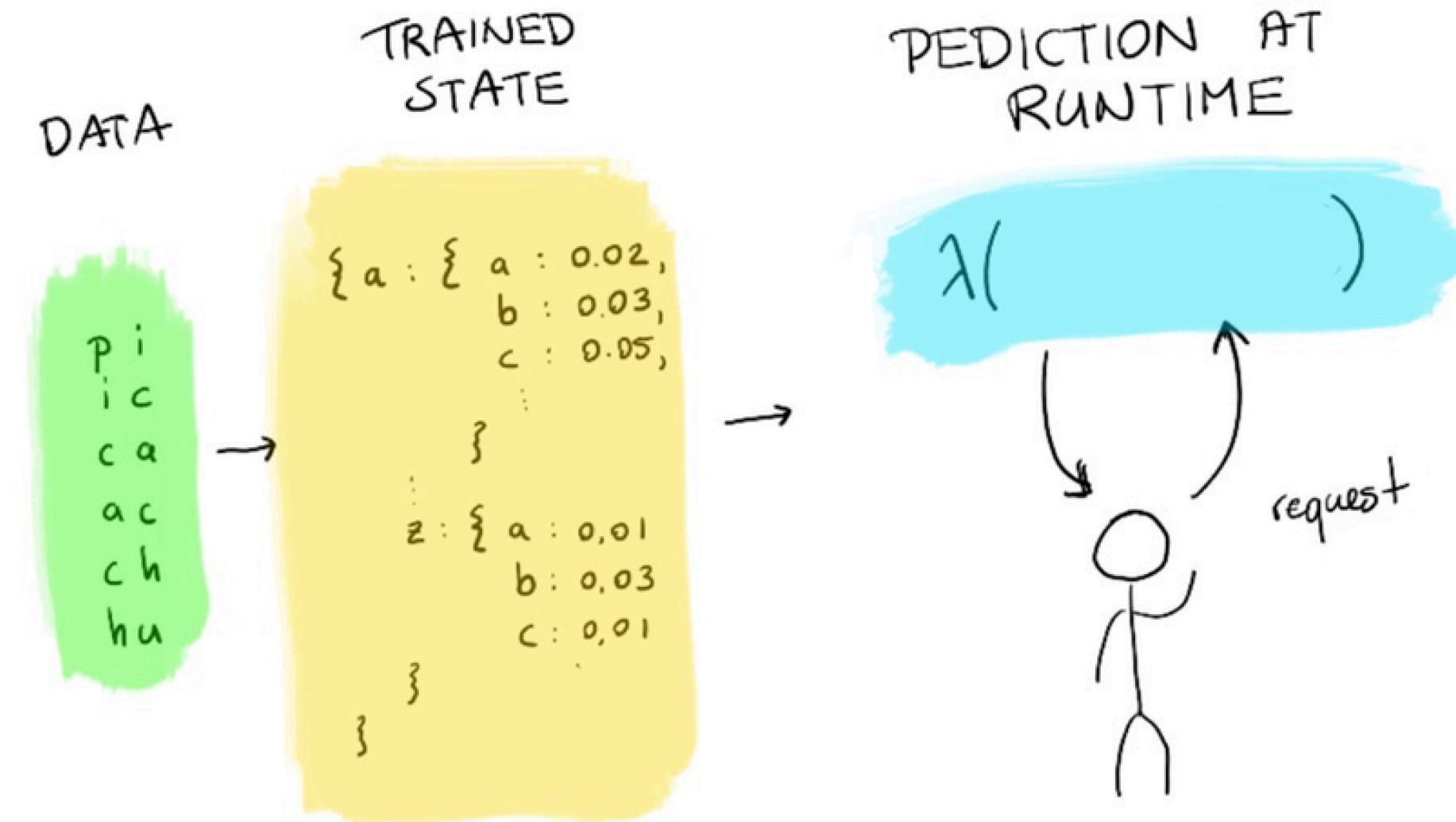
A Sequence of Tokens

Let's say now we want to generate a pokemon name. Let's say we go for the simple markov chain. We first determine how long the pokemon name needs to be, then we sample tokens.

$$p(T) = p(T|\lambda)p(\lambda)$$



Super Basic Flow



Intermediate Results

For Fun

We'll mainly discuss names from here on, but I also wanted to show that this works for RHCP lyrics as well.

Can you believe. Hold me please.
By the way I wonder what the wave meant.
White heat is screaming in the nearest bin.
When I was fortunate I know you must be fat this year.
And eat the sun and a Bottom Dollar.
Fox hole love Pie in your house
now let me spin Feather light but you cant move this

**Let's make things
harder**

Completing Tokens is Harder

Let's take as input '*u*a' and 'ou*a'.

What is the most likely 3rd character?

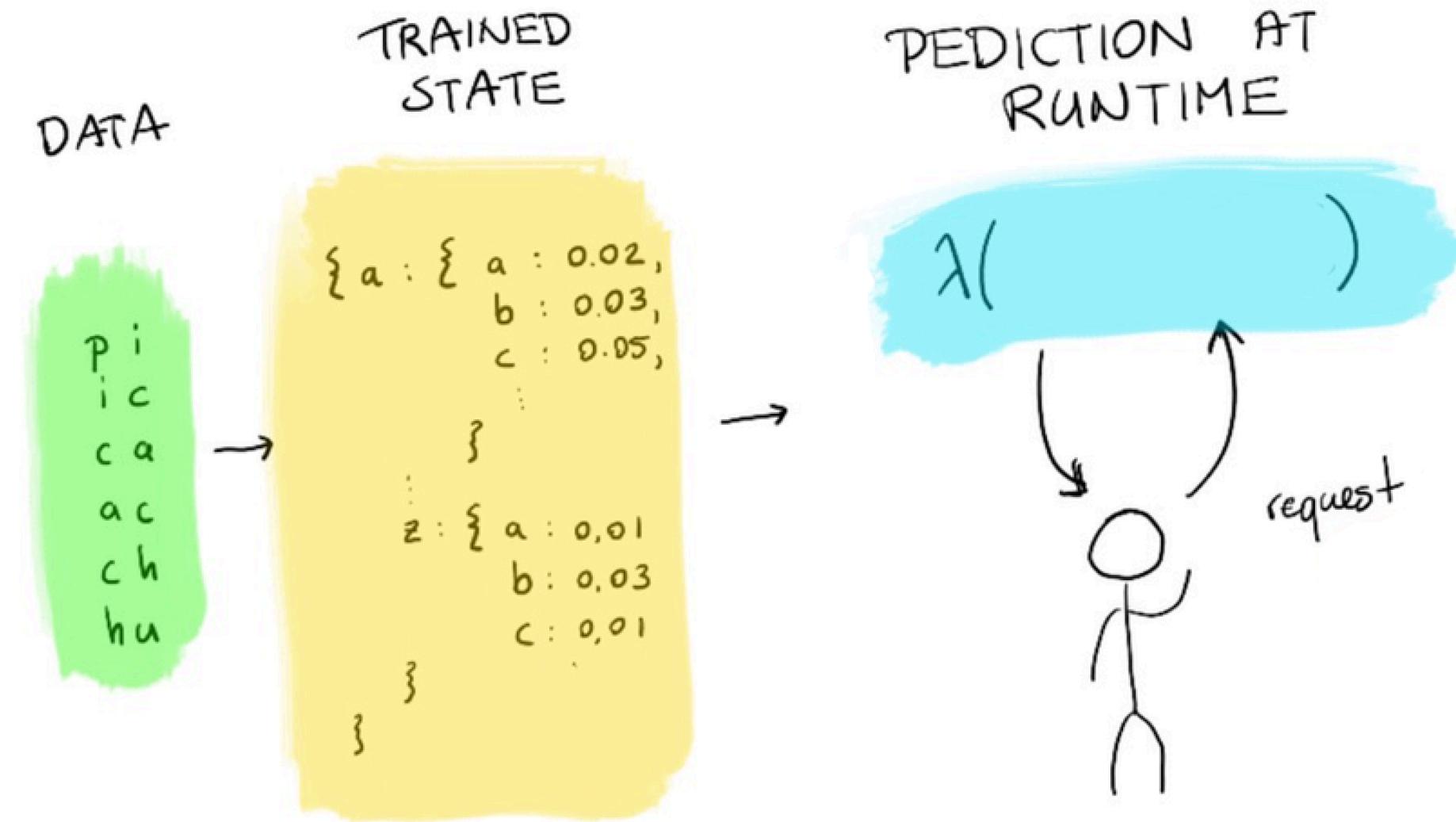
Completing Tokens is Harder

Let's take as input '*u*a' and 'ou*a'.

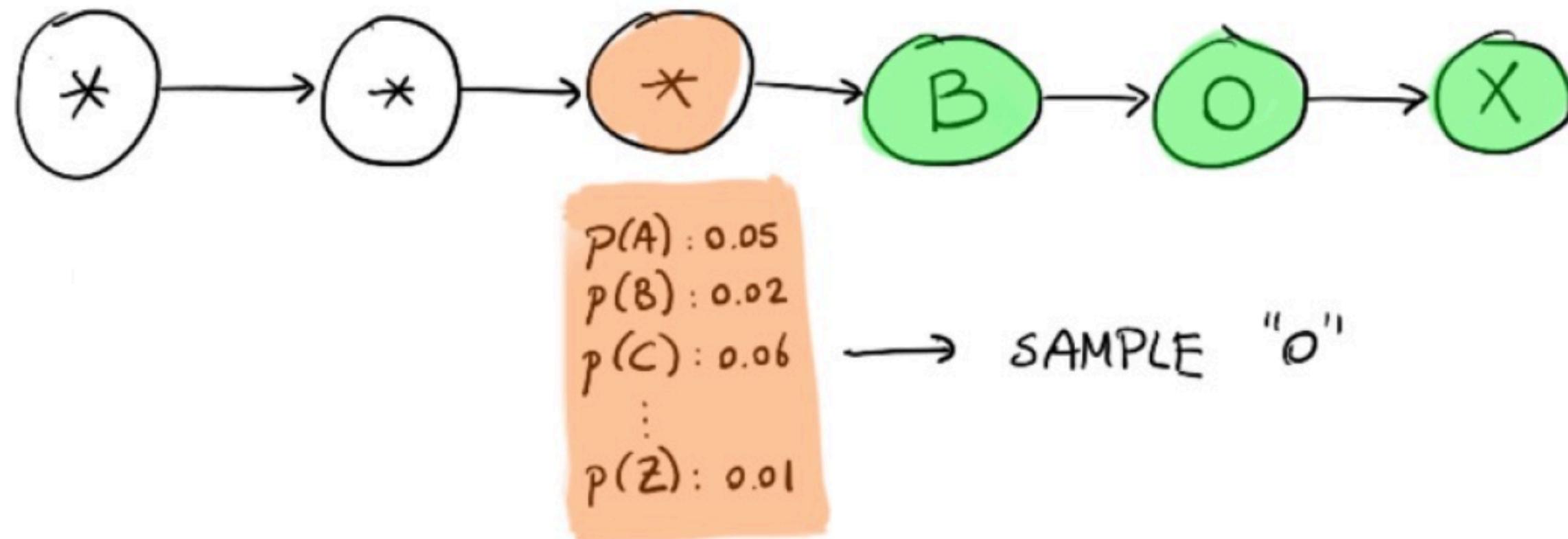
What is the most likely 3rd character?

We cannot just count and estimate probabilities and just sample from them. If we know some part of the sequence then this information needs to propagate forwards and backwards.

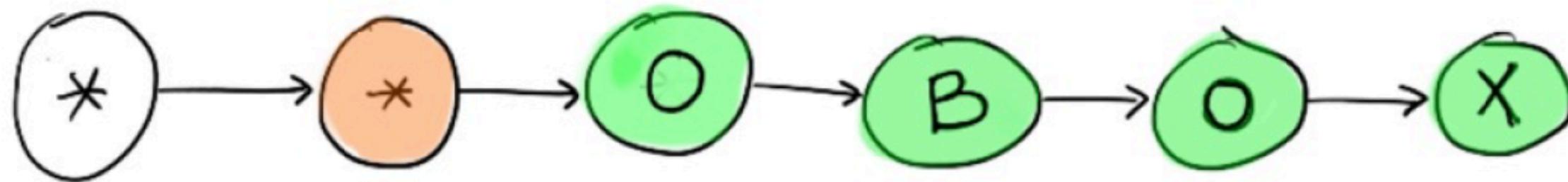
Super Basic Flow



Algorithm Intuition



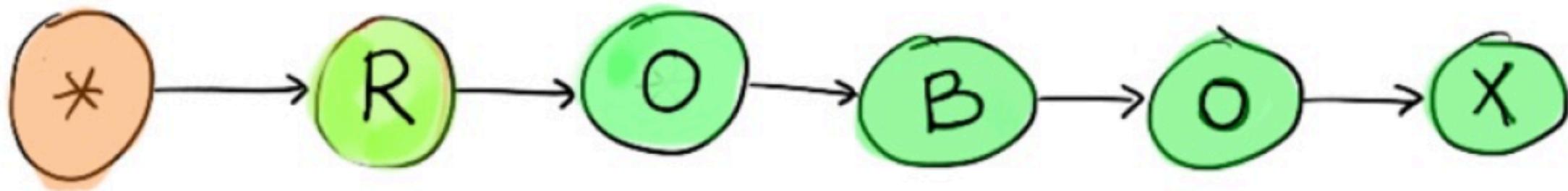
Algorithm Intuition



$p(A) : 0.02$
 $p(B) : 0.06$
 $p(C) : 0.01$
⋮
 $p(Z) : 0.02$

→ SAMPLE "R"

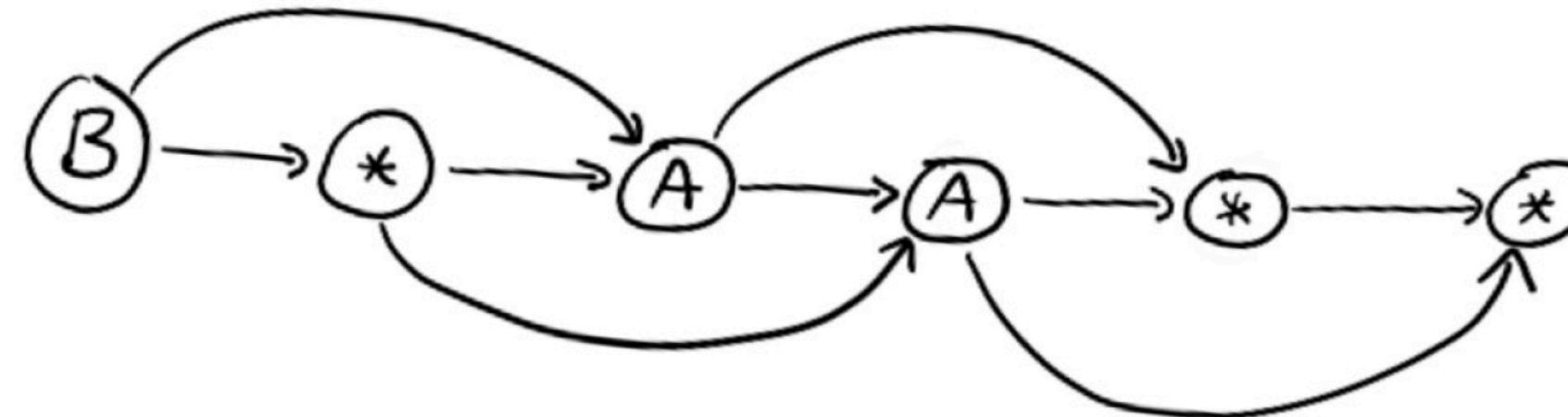
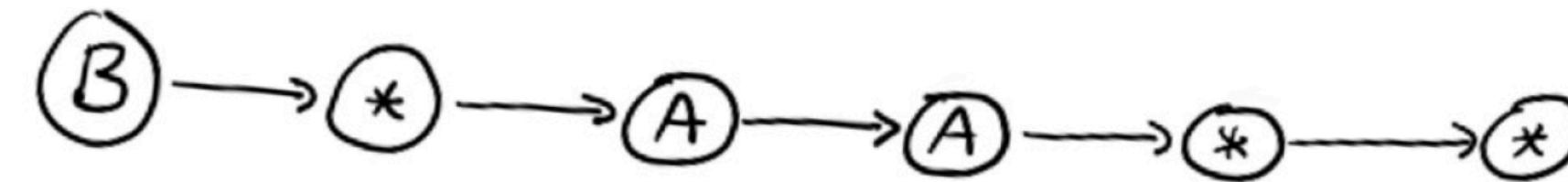
Algorithm Intuition



$p(A) : 0.01$
 $p(B) : 0.03$
 $p(C) : 0.07$
⋮
 $p(Z) : 0.00$

→ SAMPLE "B"

Same Intuition, Advanced Model



Part 2

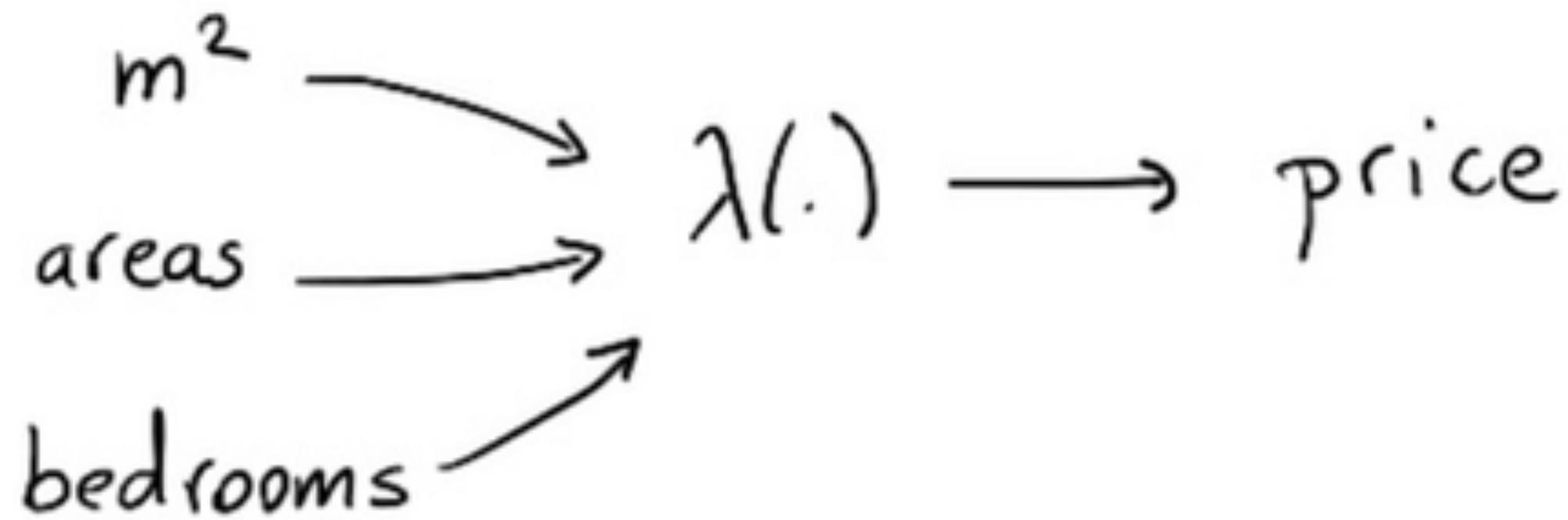
ML to AWS Lambda

Machine Learning Models

In general, machine learning models are lambda functions. They are mostly stateless and merely predict an output based on an input.

Since I am sampling my function won't be a pure function since I need some form of state to use a random number generator. If we omit this detail then we can regard my sampling function as pure enough.

General ML in Production



Machine Learning Models

Once you realize this, you can already recognize that some serverless setup with just functions may be good enough for what I want to do.

$$ML \propto \lambda$$

This applies to most trained machine learning models but not all. Some machine learning models learn via streaming data in, these would be an exception.

Super Naive Implementation

Python to the rescue.

```
import markovify

poke_names = [<all_pokemon_names>]

def lambda_handler(event, context):
    poke_names = [list(_) for _ in poke_names]
    text_model = markovify.Chain(poke_names, state_size=3)
    return ''.join(text_model.walk())
```

Super Naive Implementation

This is a **very** naive implementation.

In the function call I am both training and then taking a single sample. This is inefficient but it seemed fine to try with a corpus that is only 750 words long.

Super Naive Implementation

Implementing this in AWS lambda is super easy. You just need to make sure that all dependencies are installed and that you zip everything together.

I have only one dependency, so;

```
pip install markovify -t /some/path/markov
```

Then I place the aforementioned code into a file named `lambda_function.py`. I then zip up `/some/path/markov`

upon next

Button clicking!

First in AWS Lambda



Remove

Please go to the [IAM console](#) to configure the security for your API endpoint.



We'll set up an API Gateway endpoint with a [proxy integration type](#) (learn more about the [input](#) and [output](#) format for your function). Any method (GET, POST, etc.) will trigger your Lambda function. To set up more advanced method mappings or subpath routes, visit [Amazon API Gateway console](#).

API name

LambdaMicroservice



Deployment stage

prod



Security

AWS IAM



Lambda will add the necessary permissions for Amazon API Gateway to invoke your Lambda function. [Learn more](#) about the Lambda permissions model.

Configure function

A Lambda function consists of the custom code you want to execute. [Learn more about Lambda functions.](#)

Name* pokemon-python

Description super-useful-task-pokemon

Runtime* Python 2.7

Lambda function code

Provide the code for your function. Use the editor if your code does not require custom libraries (other than boto3). If you need custom libraries, you can upload your code and libraries as a .ZIP file.

Code entry type Upload a .ZIP file

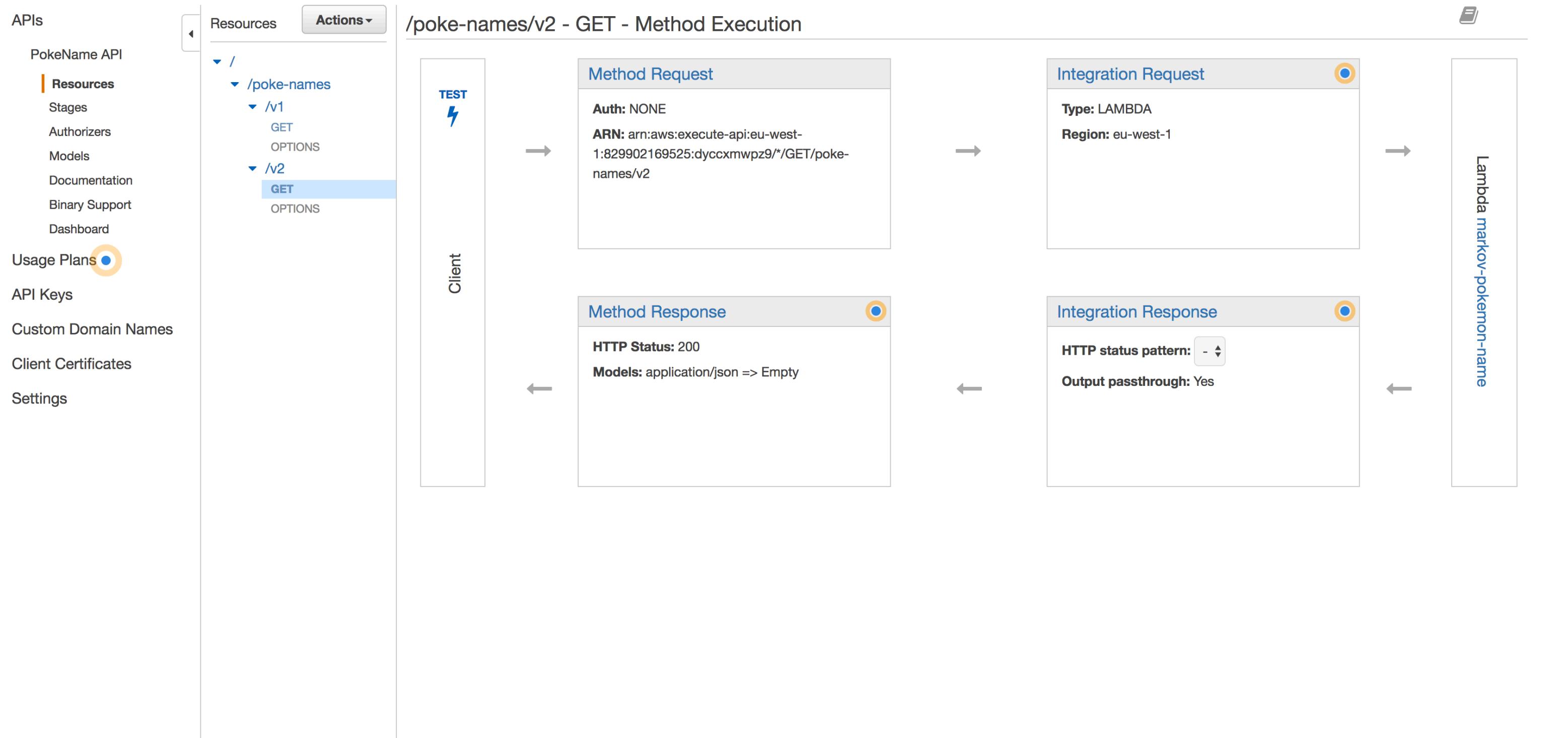
Function package* 

For files larger than 10 MB, consider uploading via S3.

Up Next

Button clicking!

Now in API Gateway





APIs

PokeName API

Resources

Stages

Authorizers

Models

Documentation

Binary Support

Dashboard

Usage Plans

API Keys

Custom Domain Names

Client Certificates

Settings

Resources

Actions ▾

/poke-names/v2 - GET - Method Execution

METHOD ACTIONS

Edit Method Documentation

Delete Method

RESOURCE ACTIONS

Create Method

Create Resource

Enable CORS

Edit Resource Documentation

Delete Resource

API ACTIONS

Deploy API

Import API

Edit API Documentation

Delete API

Method Request

Auth: NONE

ARN: arn:aws:execute-api:eu-west-1:829902169525:dyccxmwpz9/*/GET/poke-names/v2

Method Response

HTTP Status: 200

Models: application/json => Empty

What do I have?

- Supposedly 1 million requests for free.
- Latency of 200ms when hot. More like 1s when cold.
- Only 20 cents for every next million.
- No real scaling concerns
- Easy to update, but immature architecture.
- Modest cloud lockin.

Downsides

I'm not so sure how much I like the logging.

It is a cloud service it can go down.

S3 was briefly down yesterday. It pulled down my blog but also other websites like <http://isitdowntoday.com/>.

Downsides

I'm not so sure how much I like the logging.

It is a cloud service it can go down.

S3 was briefly down yesterday. It pulled down my blog but also other websites like <http://isitdowntoday.com/>.

Internet responded via <http://isisitdowntodaydowntoday.com>.

Downsides

I'm not so sure how much I like the logging.

It is a cloud service it can go down.

S3 was briefly down yesterday. It pulled down my blog but also other websites like <http://isitdowntoday.com/>.

Internet responded via <http://isisitdowntodaydowntoday.com>.

This website is now down.

Part 3

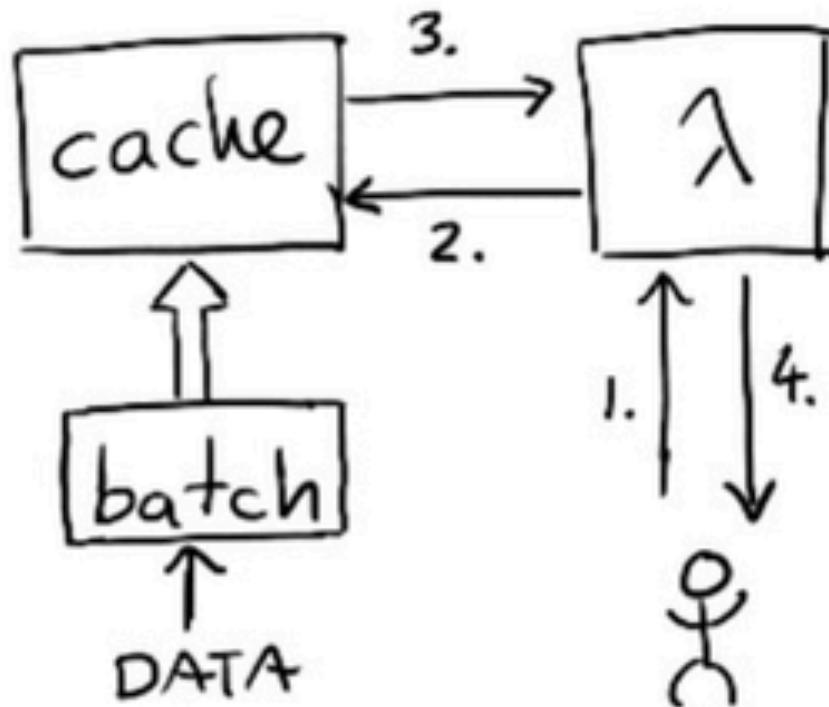
Mature ML/AWS Lambda Flow

Now to improve

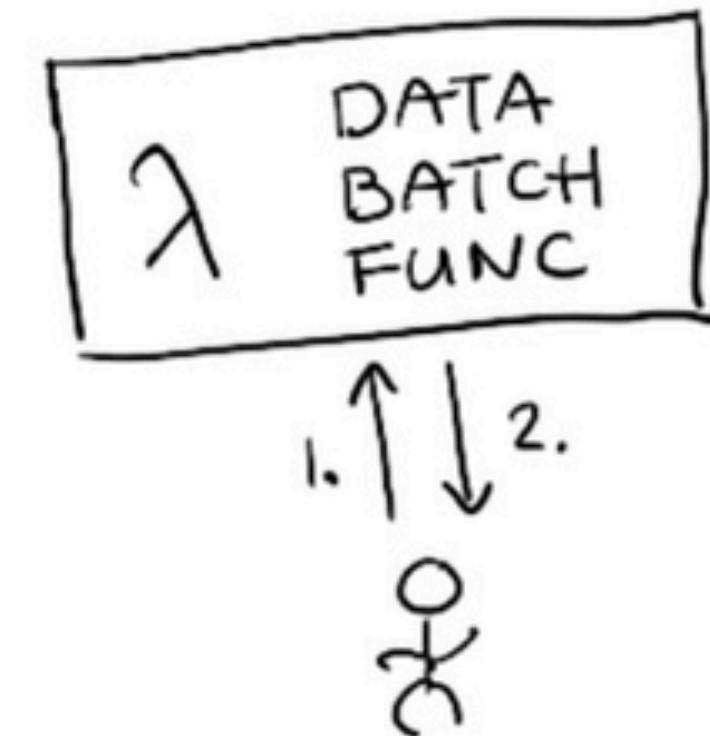
- Think about the future
- Allow for better python models
- Automate this workflow

Flow

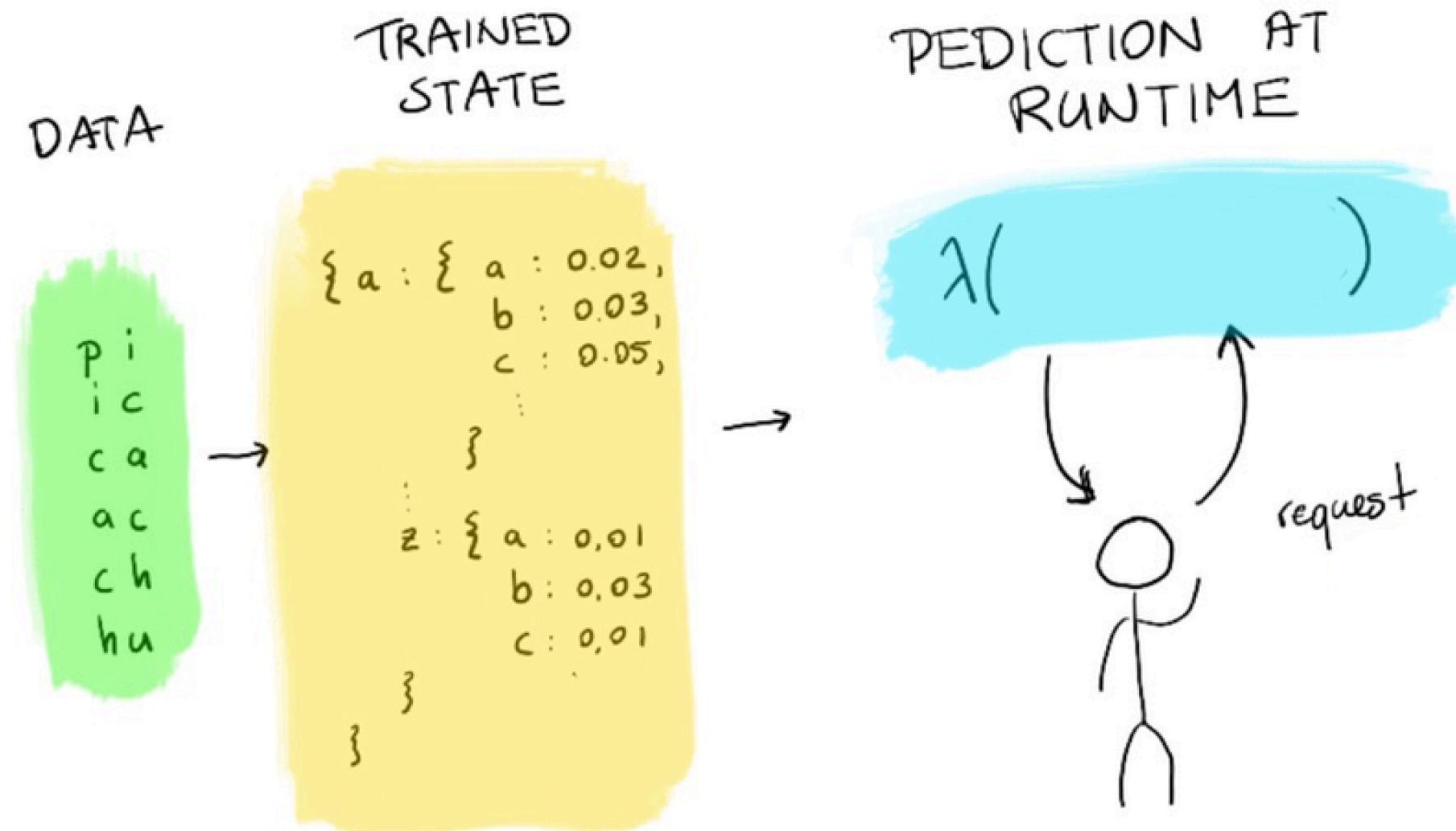
IDEAL



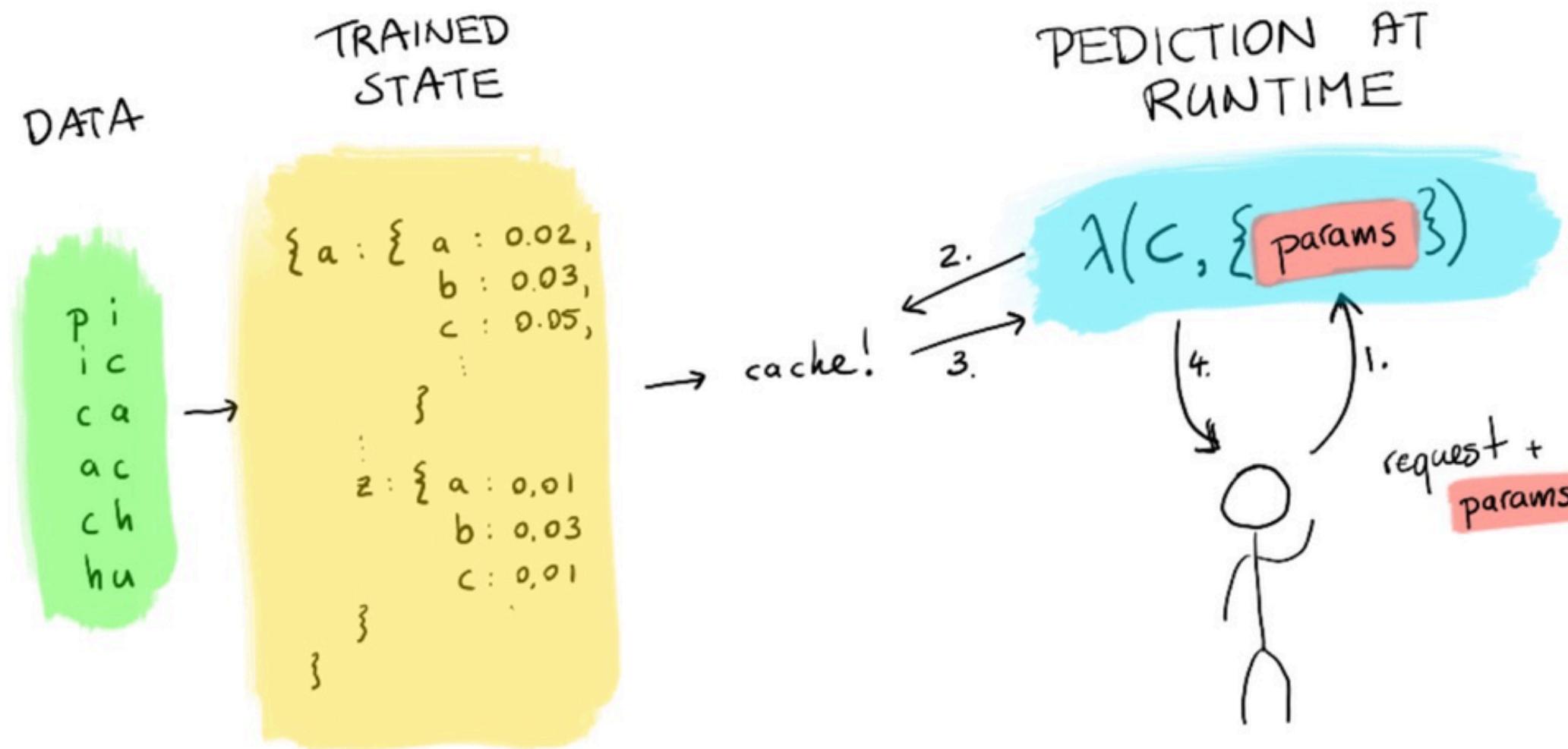
CURRENT



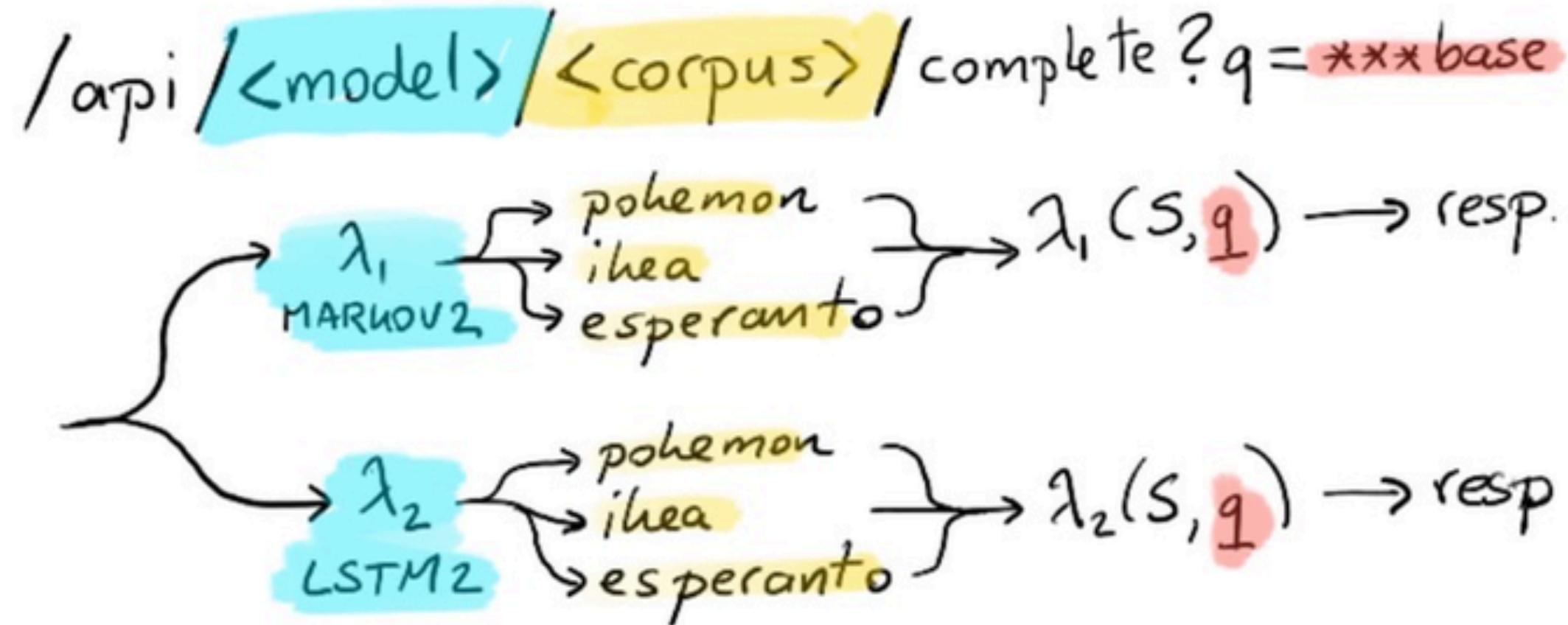
Flow Before



Flow After



Flow



Better Python Models

Pomegranate has a bunch of dependencies that are nasty, the main culprit being numpy. It has BLAS and all sorts of low level stuff that can be machine specific and AWS runs functions on their AMI.

Quick trick; prepare the zip file via docker!

See <https://github.com/ryansb/sklearn-build-lambda>

Better Python Models

Step one; edit the 'build.sh' file.

```
do_pip () {  
    pip install --upgrade pip wheel  
    pip install --use-wheel --no-binary numpy numpy  
    pip install --use-wheel --no-binary cython cython  
    pip install --use-wheel --no-binary scipy scipy  
    pip install --use-wheel networkx  
    pip install --use-wheel joblib  
    pip install --use-wheel pomegranate  
}
```

Better Python Models

Step two; generate zip file.

```
$ docker pull amazonlinux:2016.09
$ docker run -v $(pwd):/outputs -it amazonlinux:2016.09 \
/bin/bash /outputs/build.sh
```

Automate!

.yml all the things!

```
aws cloudformation package  
  --template-file poke-mod-adv.yml  
  --s3-bucket lambda-intermediate  
  --output-template-file poke-template.yml
```

```
aws cloudformation deploy  
  --template-file /somepath/poke-template.yml  
  --stack-name pokemon-tut
```

Conclusion

These lambda cloud services seem like a thing to invest in.
Even when you're not a devops/cloud kind of person.

There is a bit of lock in but I imagine every cloud vendor to have an offering that does something similar. My gut tells me to go to either AWS (more mature lambda at the moment) or wait for google cloud (better tensorflow support but functions is limited at the moment).

Pitch

CFP is still open. Some tickets too.



<http://pydata.org/amsterdam2017/cfp/>