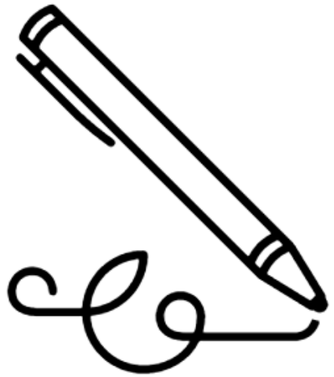
# Tech Names as a Service



Vincent D. Warmerdam
koaning.io - fishnets88 - GoDataDriven - imgs @ nounproject

# Today

In this talk I will discuss 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.

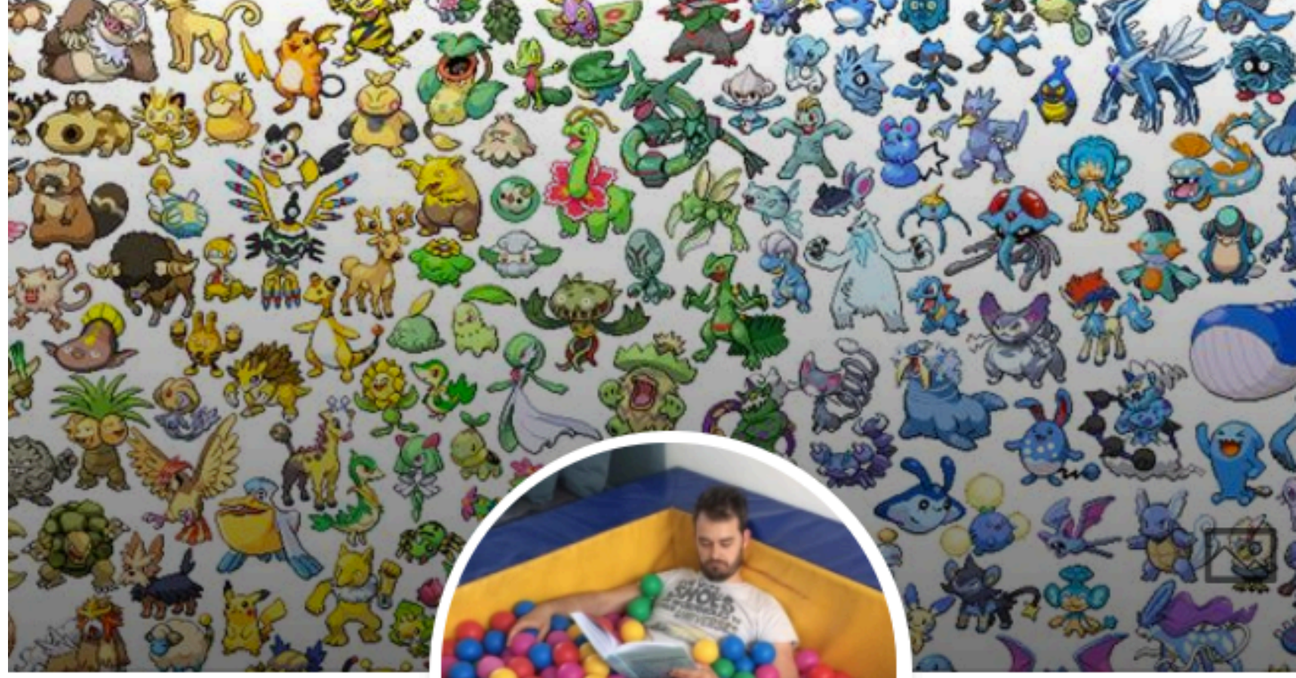
The second part of the talk will focus on the architecture/cloud part of what I did and what I can do better.

I will make sure to mention lessons learned along the way.

# Who is this guy?

3 years @ GDD
koaning.io
PyData Amsterdam
Rstudio training partner
Machine Learning Meetup
free open sessions in data
39 endorsements for awesomeness

I write code,
I solve data problems,
ASK ME ANYTHING.



Vincent Warmerdam

Pokemon Master at GoDataDriven

GoDataDriven • Vrije Universiteit Amsterdam

# 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.

# 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.

I typically ask recruiters to point out which of these are pokemon.

# 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 can't blame 'em too much ...

# Repokémon

Showcase of GitHub repos with Pokémon names

Star 238 Fork 23 Tweet Like 237 Share

**Jolteon**
Babel + Electron + React + Browserify + SASS application stack. El...
★ 999 ⑂ 38

**Golem**
Golem project is to create a global prosumer market for computing pow...
★ 922 ⑂ 71

**Electrode**
Electrode Platform for Universal React + Node application
★ 801 ⑂ 86

**Ditto**
Lightweight Markdown Documentation System
★ 366 ⑂ 43

**Lucario**
The best flat theme for Vim, Atom, Sublime Text, Jetbrains Editors, T...
★ 334 ⑂ 57

**Snorlax**
The ultimate lazy library. An example for being a good library citize...
★ 181 ⑂ 10

**Rapidash**
Rapidly develop your API client
★ 151 ⑂ 17

**Mew**
Messaging in the Emacs World
★ 134 ⑂ 36

**Delibird**
Shipment tracking library for Golang.
★ 131 ⑂ 28

**Tangrowth**
Python3.5 async crawler example with aiohttp and asyncio
★ 130 ⑂ 13

**Shellder**
Featured zsh/fish shell theme
★ 129 ⑂ 22

**Slowpoke**
Rack::Timeout is great. Slowpoke makes it better.
★ 123 ⑂ 5

**Eevee**
基于 github page 的在线编辑 blog 平台
★ 101 ⑂ 24

**Porygon**
the unofficial Pokémon Go Plus SDK
★ 83 ⑂ 5

**Klang**
Clojurescript logging library
★ 83 ⑂ 1

**Gloom**
A dark and gloomy pastel color syntax theme for Atom.
★ 83 ⑂ 10

**Geodude**
A tiny command-line utility for geocoding addresses.
★ 65 ⑂ 3

**Charmander**
Charmander Scheduler Lab - Mesos, Docker, InfluxDB, Spark
★ 57 ⑂ 12

**Binacle**
Full-bin indexation of binary files
★ 56 ⑂ 5

**Smeargle**
Vim plugin for colouring the text background based on information min...
★ 52 ⑂ 1

**Abra**
Assembly Based ReAligner
★ 50 ⑂ 10

**Vulpix**
Automated code grading system
★ 46 ⑂ 9

**Gastly**
Create screenshots or previews of web pages using Gastly.
★ 45 ⑂ 8

**Tentacool**
REST API to manage Linux networking via netlink
★ 41 ⑂ 12

**Onix**
A convenient mapping between ruby objects and the ONIX XML specificat...
★ 38 ⑂ 34

**Treecko**
A collection of functional and immutable helpers for working with tre...
★ 29 ⑂ 1

**Dratini**
Dratini is a neat network abstraction layer.
★ 29 ⑂ 4

**Scyther**
The Scyther Tool for the symbolic analysis of security protocols
★ 27 ⑂ 12

**Rhydon**
Save file editor for Pokemon Red/Blue/Yellow.
★ 24 ⑂ 3

**Shelmet**
A JVM Heap dump viewer - a souped-up jhat in scala
★ 22 ⑂ 1

**Natu**
Natural units in Python
★ 21 ⑂ 4

**Elekid**
Resolver for React's Server Side Render on Module, ReactElement or El...
★ 18 ⑂ 0

**Munna**
Simple wrapper cache for ActiveRecord, Ruby Object
★ 18 ⑂ 4

**Castform**
Async form validation on the client and server.
★ 16 ⑂ 1

**Golbat**
Golbat is an Android library that helps you on working with camera, ...
★ 16 ⑂ 2

**Blastoise**
tiny relational database
★ 14 ⑂ 1

**Kadabra**
[DEPRECATED] Kadabra is my automatic LFI Exploiter and Scanner, writt...
★ 13 ⑂ 6

**Klink**
A Simple and Clean Sphinx Docs Theme
★ 13 ⑂ 5

**Klefki**
Simple substitution cipher module.
★ 12 ⑂ 2

**Staryu**
Open-source 5-key Keypad
★ 12 ⑂ 3



**Hoothoot**
hoothoot hoothoot hoothoot. Find your path in the forest of nested da...
★ 12 ⑂ 1



**Diglett**
Diglett is a cron management system that manage all your cron jobs wi...
★ 11 ⑂ 1

# 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: Enterprice 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`

# DEMO TIME

## Crowd Participation!

# Part 1

# Machine Learning

# A Sequence of Tokens

The simplest model you think of;

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

For every pair of tokens, keep track of how often they occur. You can learn these probabilities given a corpus and then sample;

$$p(t_1) \rightarrow p(t_2 | t_1) \rightarrow p(t_3 | t_2) \rightarrow \ldots$$

# 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.

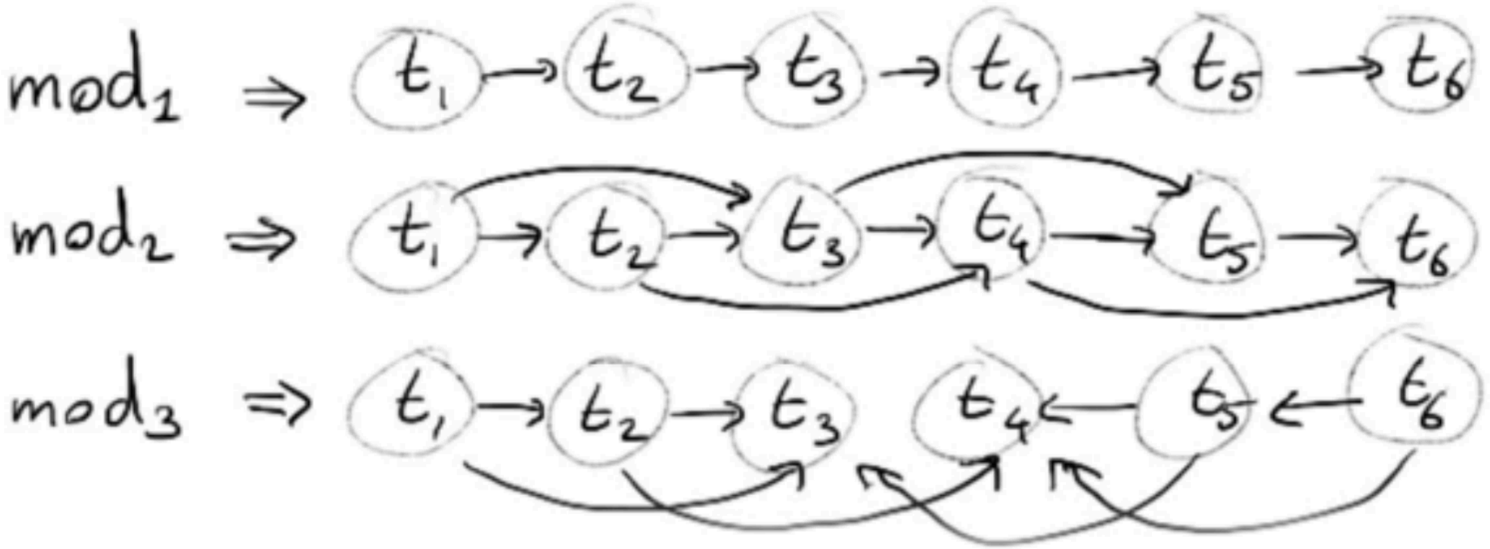
$$p(t_i | t_{i-1}, t_{i-2})$$

Similar method of sampling.

$$p(t_1) \rightarrow p(t_2|t_1) \rightarrow p(t_3|t_2, t_1) \rightarrow p(t_4|t_3, t_2) \rightarrow \dots$$

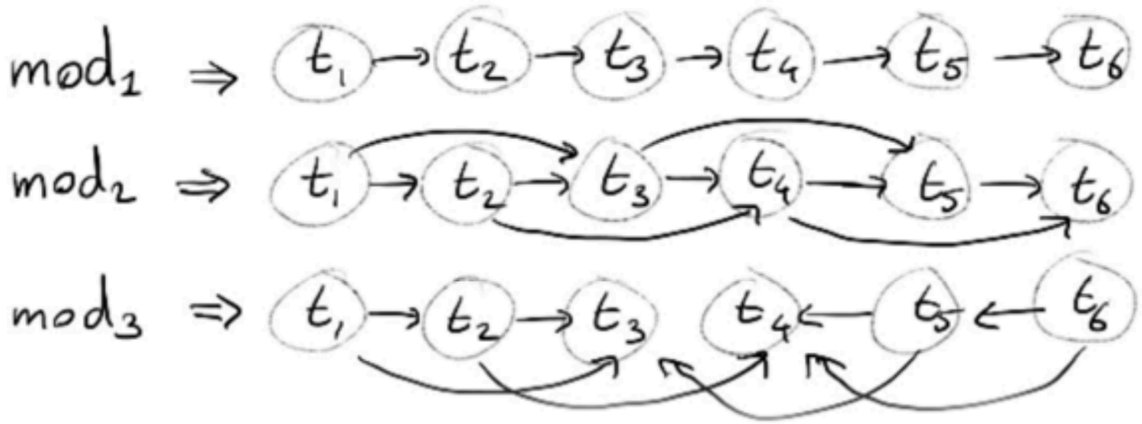
# A sequence of Tokens

But generally, these chains can have any shape;



# A sequence of Tokens

But generally, these chains can have any shape;

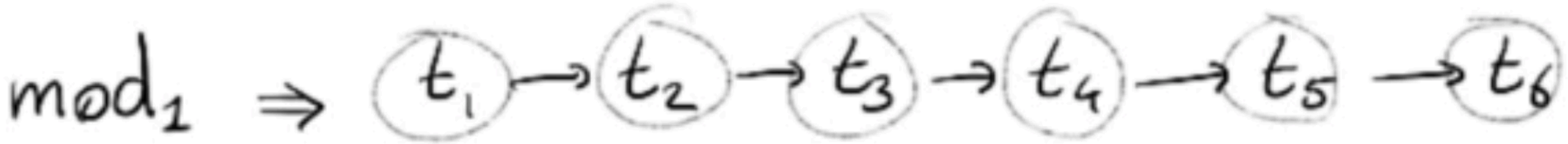



Who can spot the issue I need to deal with?

# A Sequence of Tokens

Markov chains tend to keep on going. They need to stop at some point. Thankfully, bayesians have a solution. We first determine how long the pokemon name needs to be, then we sample tokens.

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



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

POKEMON results

```
lydo
keen
wqool
ryrys
poole
utcala
youtail
olma
elttyp
```

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

IKEA results

```
anapa
frodok
pasig
ripe
latrank
vis
gsoo
yirbs
ilosseln
```

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

## RHCP Lyrics

```
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
```

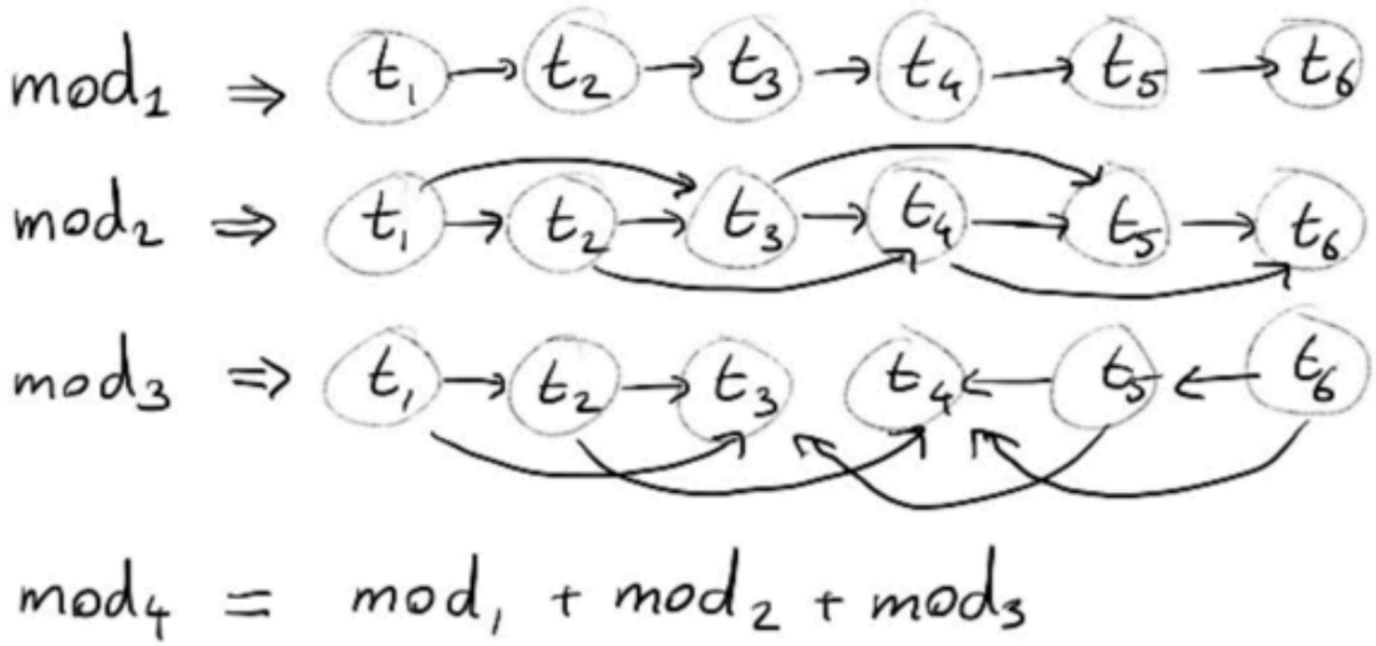
# Ok ... but.

The results are fun and all but not good enough.

1. The model is a bit naive and the results could be better.
2. Perhaps the end user does not want a name from scratch. Perhaps the user can supply me with some given tokens that I need to finish. Whatever method I come up with should be able to do this too.

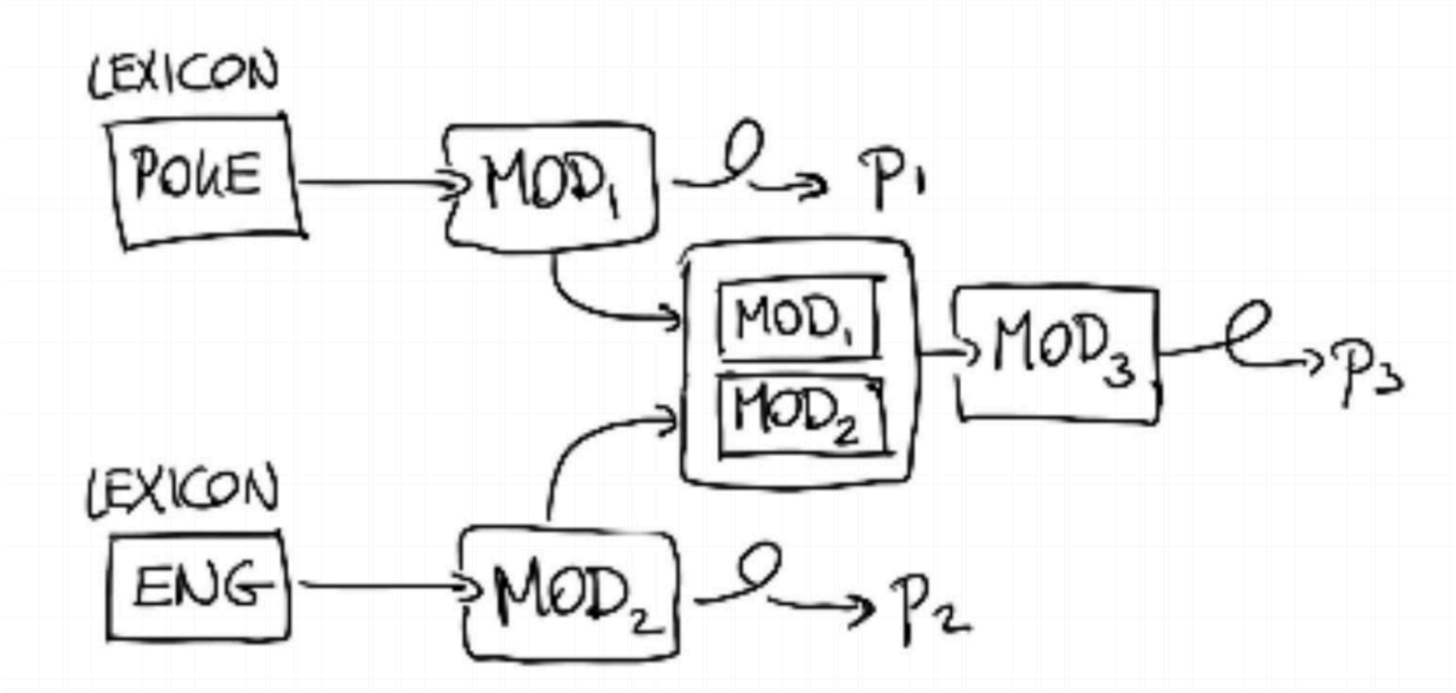
# Improving the model: idea 1

Ensembles, anyone?



# Improving the model: idea 2

Add lexicons manually?

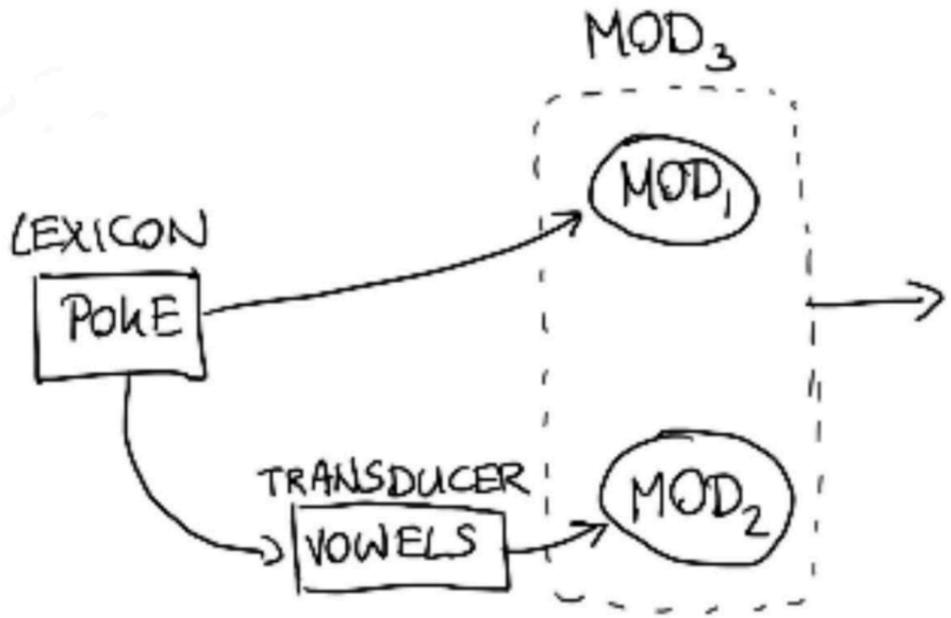


# Improving the model: idea 3

Or take different datasets

# Improving the model: idea 4

Map the tokens to another space? Phonetics?



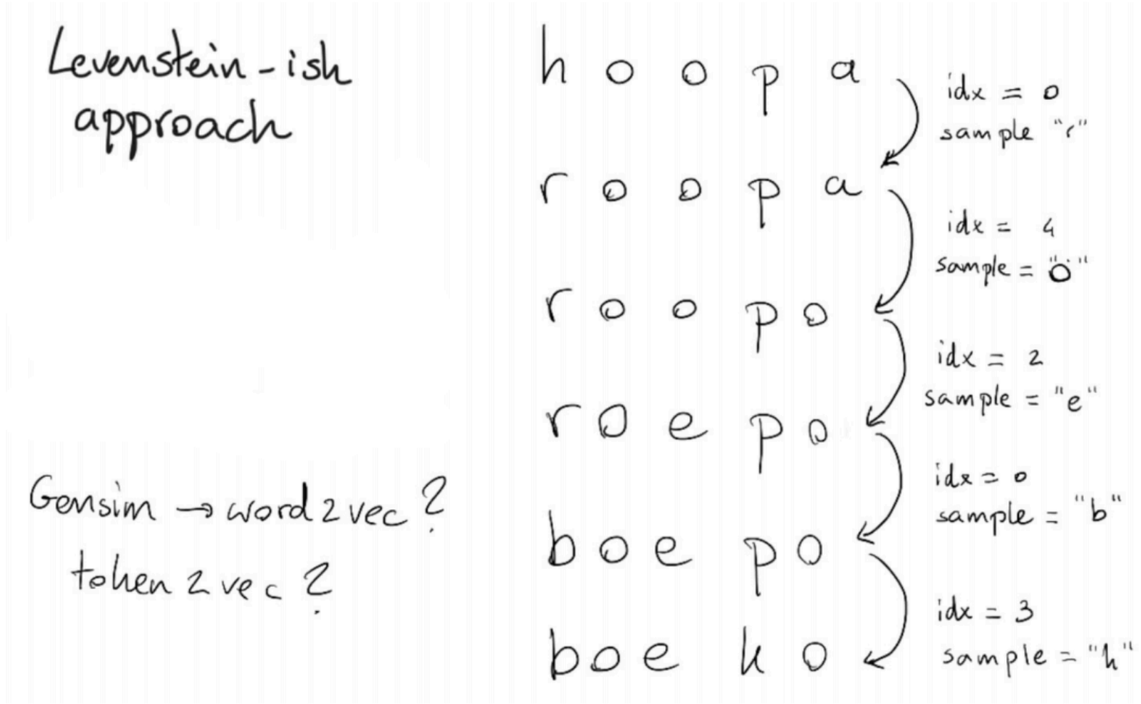
# Improving the model: idea 5

Adverserial judges?

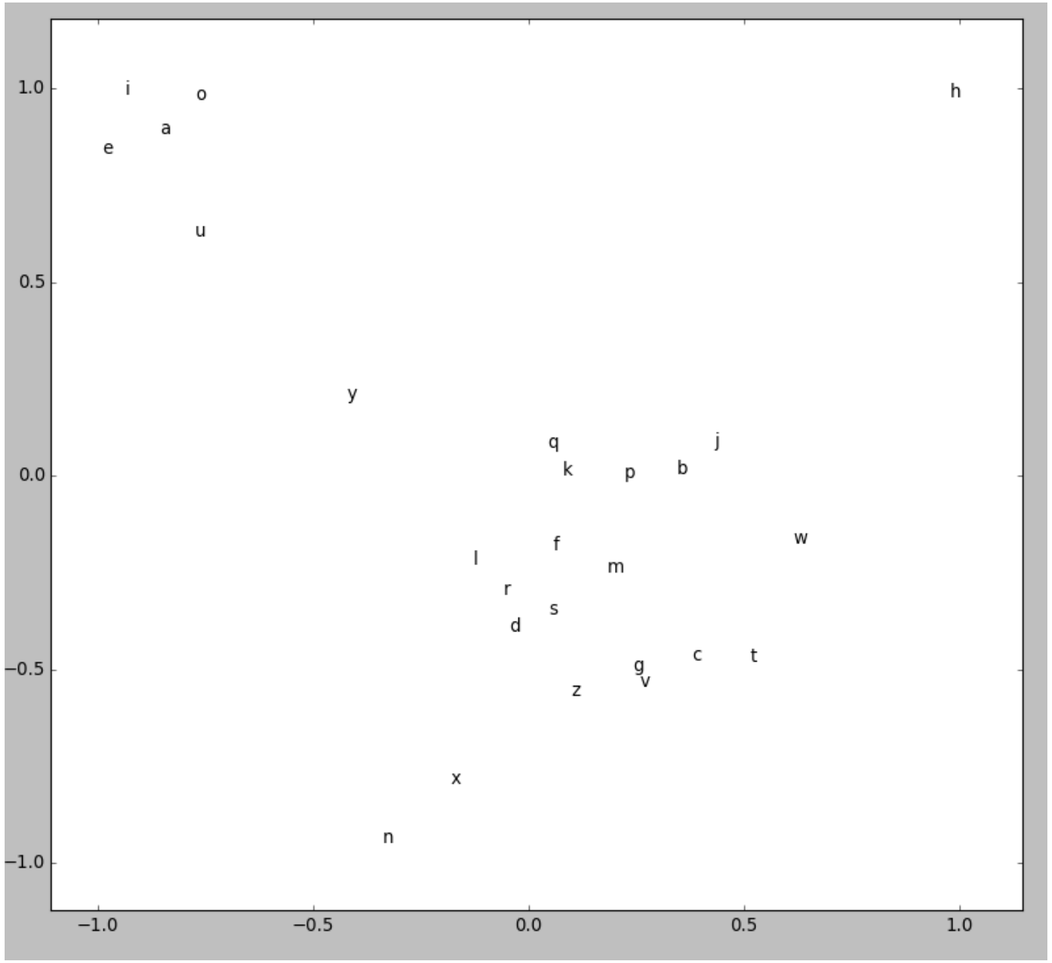
Sample 100 samples, have another algorithm sort them on believability and pick the best performing one.

# Improving the model: idea 6

Heuristics?

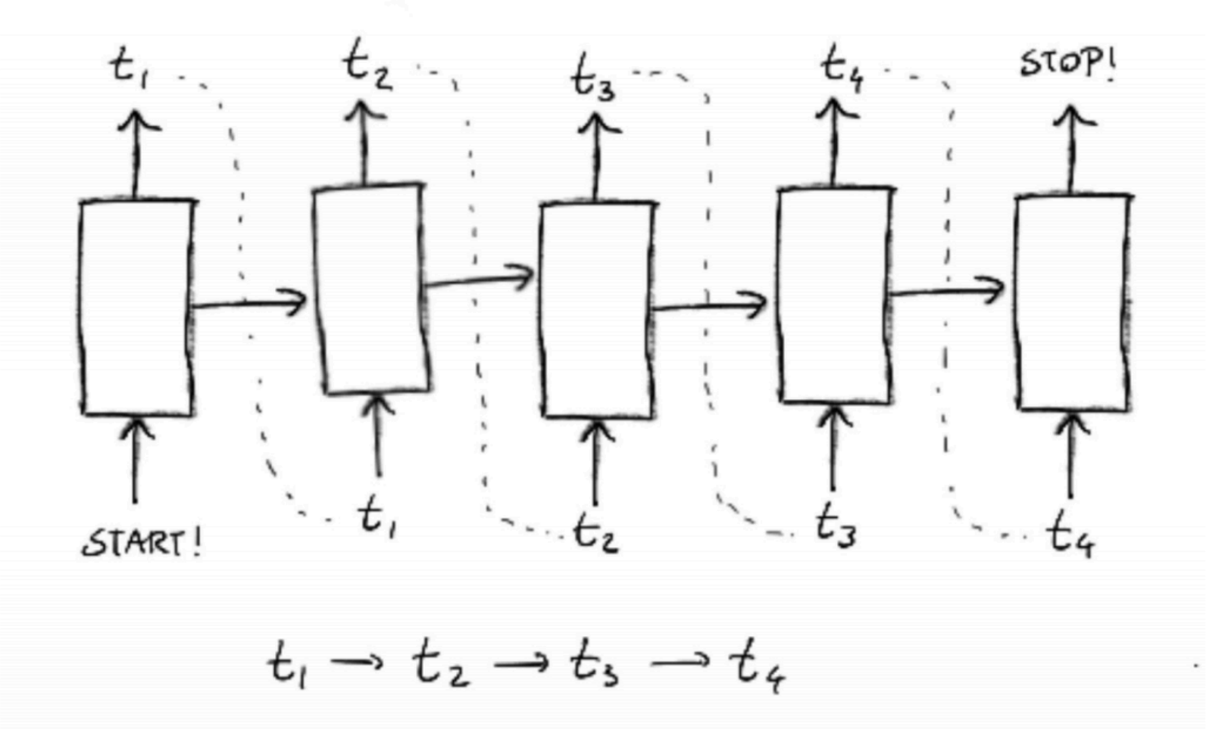


# Distance measure suggestion. Char2Vec.



# Improving the model: idea 7

Why not deep models?



# Deep Model Failure

The standard LSTM trick will not work for my **ultimate** usecase.

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

What is the most likely 3rd character? I hope you agree that we cannot 'just sample forward'. It part algorithm problem but part data structure problem too.

# Completing Tokens is Harder

Completing tokens may be a better features for end users and it entropy wise it should reduce the search space considerably.

**This feature changes the ML task, but may be a better feature.**

# Completing Tokens is Harder

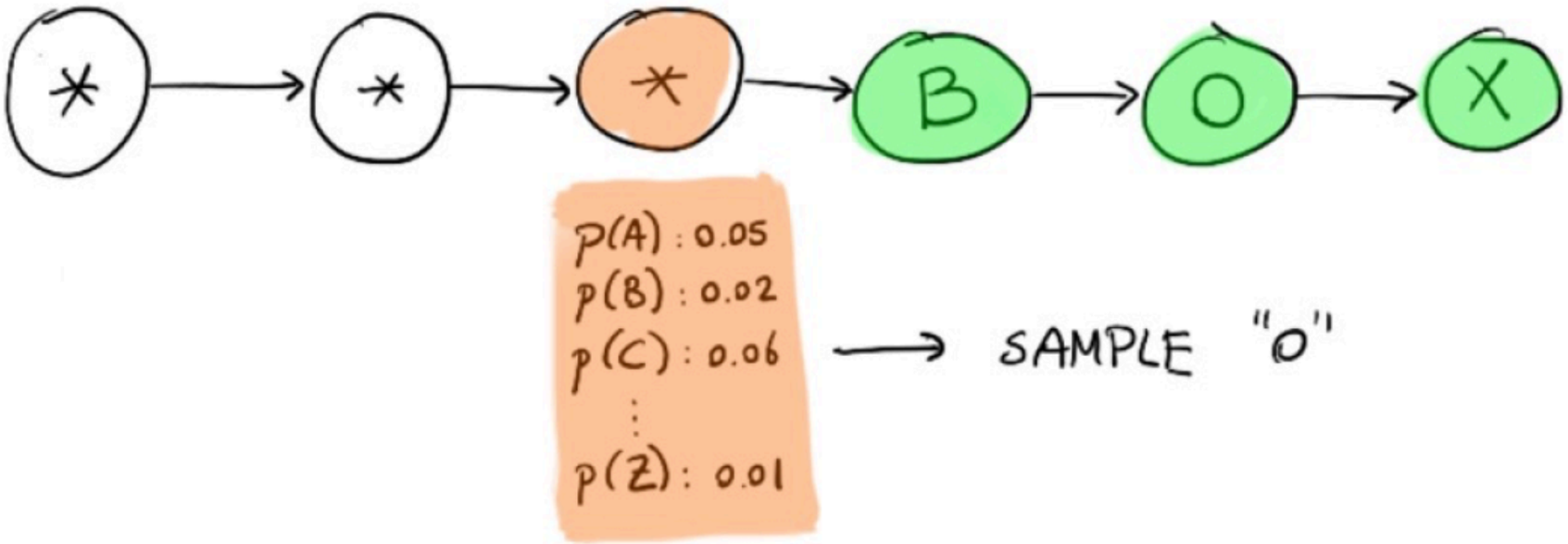
Completing tokens may be a better features for end users and it entropy wise it should reduce the search space considerably.

**This feature changes the ML task, but may be a better feature.**

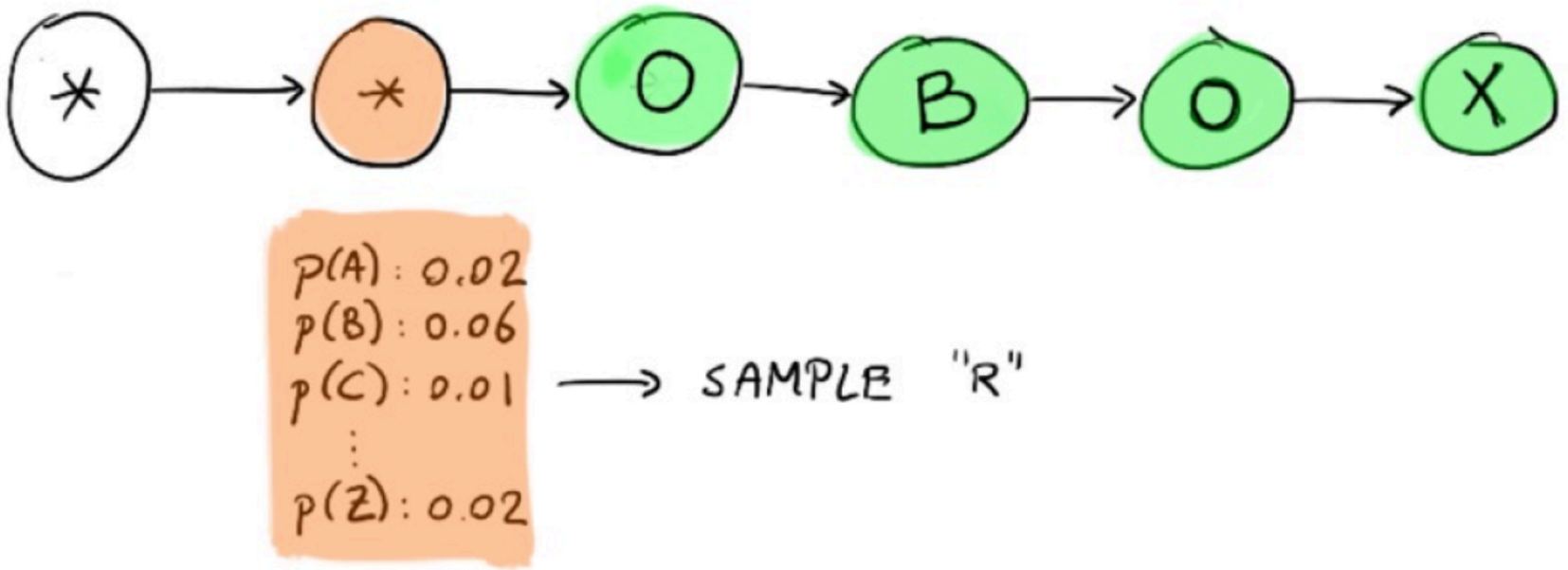
We cannot just count and estimate probabilities and just sample from them. If we know some part of the sequence then this information needs to propogate forwards and backwards.

How to deal with this?

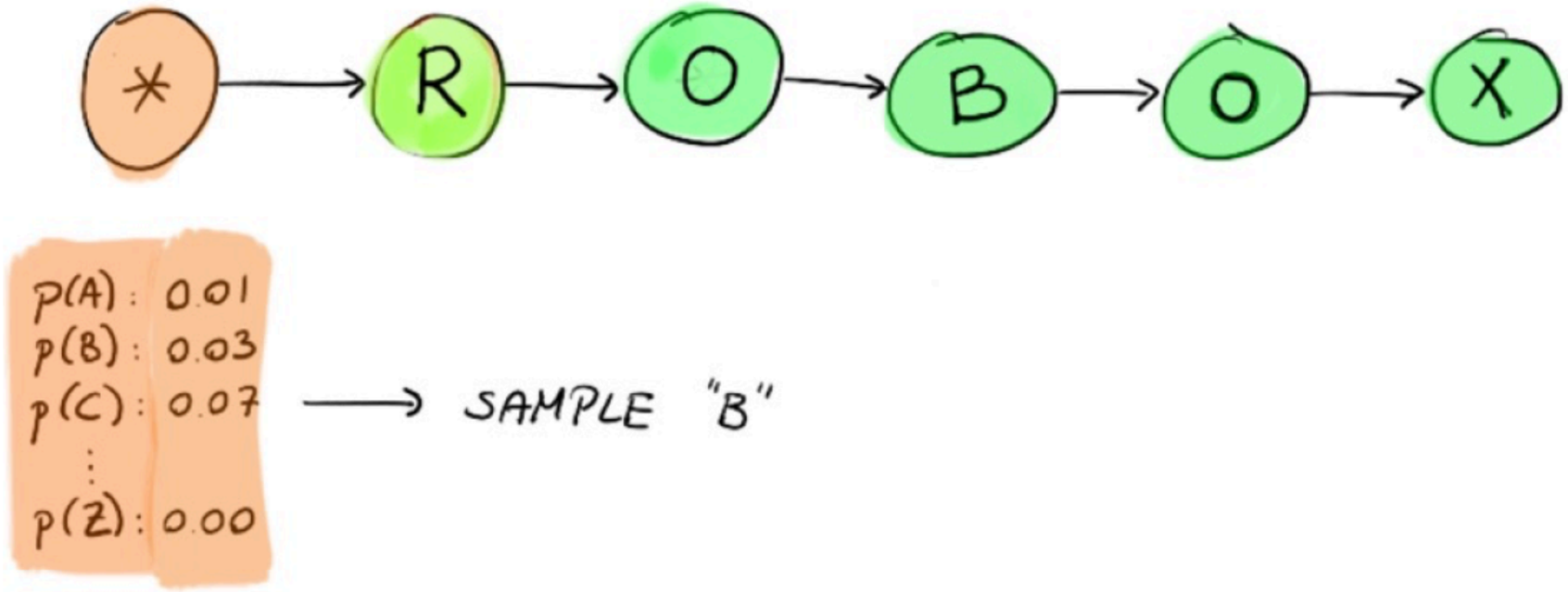
# Algorithm Intuition



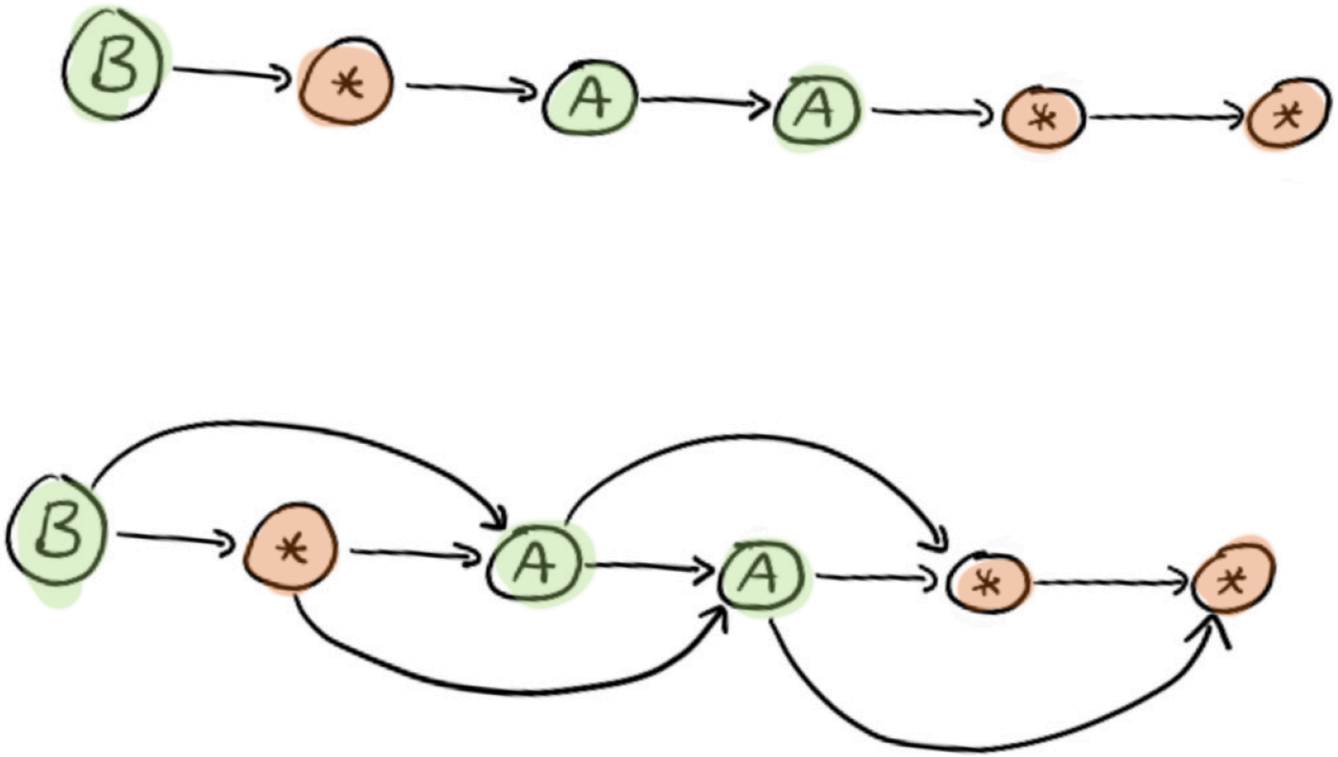
# Algorithm Intuition



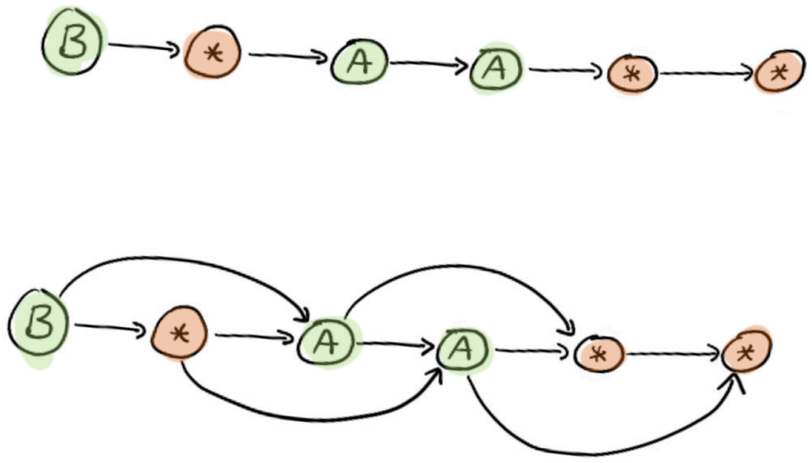
# Algorithm Intuition



# Same Intuition, Advanced Mode



# Implementation




Every arc is a discrete probability distribution. If you have those you have a defined system, you merely need a tool to sample from it.

# Towards Implementation

For anything simple I've used `markovify` it is a small python library that does nothing more but sampling forward markov chains.

A more proper inference tool for the missing characters situation is `pomegrenate`. It is a very likeable library for more bayesian-type of algorithms. It is written in cython by a cool guy named Jacob who uses it in the field of genome science.

Might make a cool NumFocus project someday. Keep an eye!

# Implementation

Bulk of the training is done via;

```
import pomegranate as pg
chain = pg.MarkovChain.from_samples(names, k=lookback)
dists = [chain.distributions[i] for i in range(lookback)]
```

These distributions are basically nodes in the probibalistic network. This bit will train the transition probabilities.

# Implementatio

Bulk of the inference;

```
network = pg.BayesianNetwork("name-problem")
network.add_states(*states)
network.bake()
network.predict_proba(givens)
```

tokens you could sample

We need to load the trained probabilities, create a new network that is as long as it needs to be and then give it the `givens` to get a probability to sample from.

# Part 3

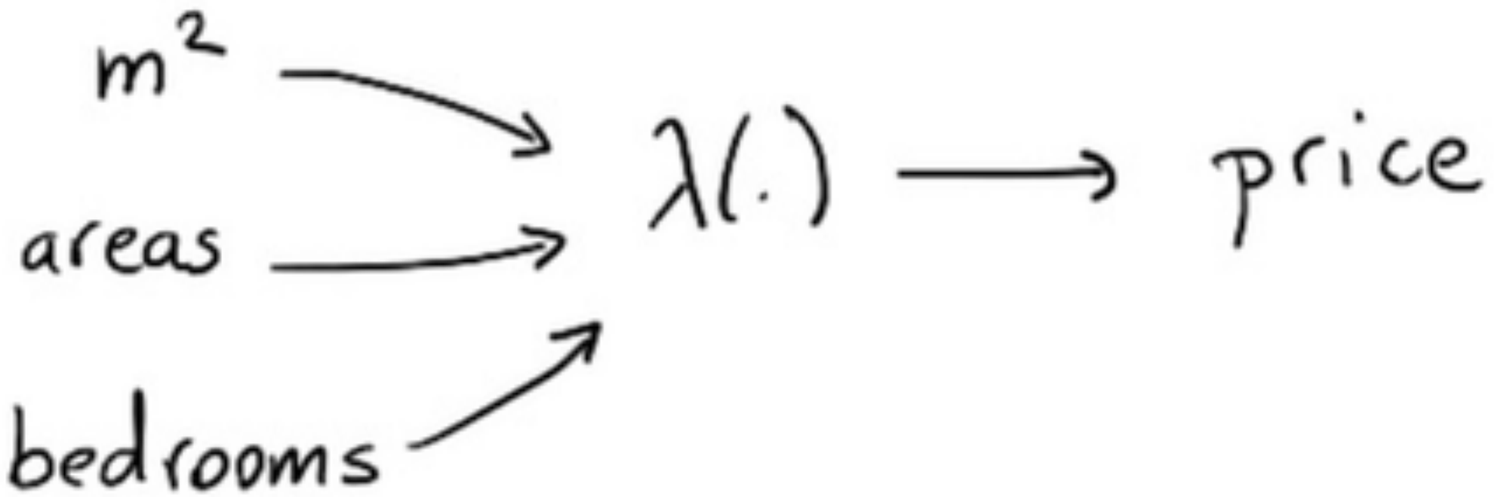
## Getting this live

# 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 Productio



# Machine Learning Model

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.

$$\mathrm{ML} \propto \lambda$$

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

# Super Naive Implementatio

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 Implementatio

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 Implementatio

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.

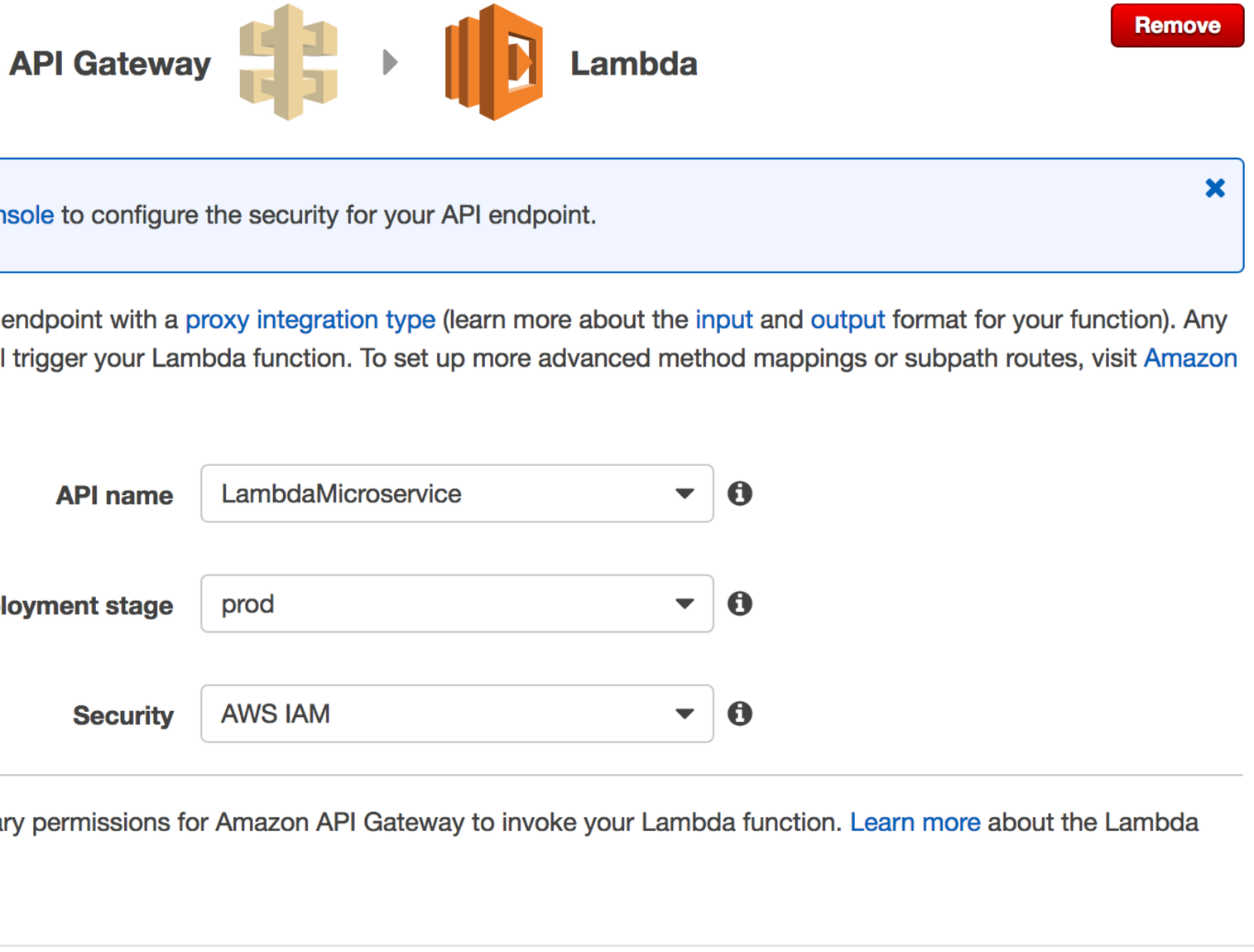
I have only one dependency, so;

```
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`

# Up Next

# Button Clicking First in AWS Lambd


API Gateway
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.

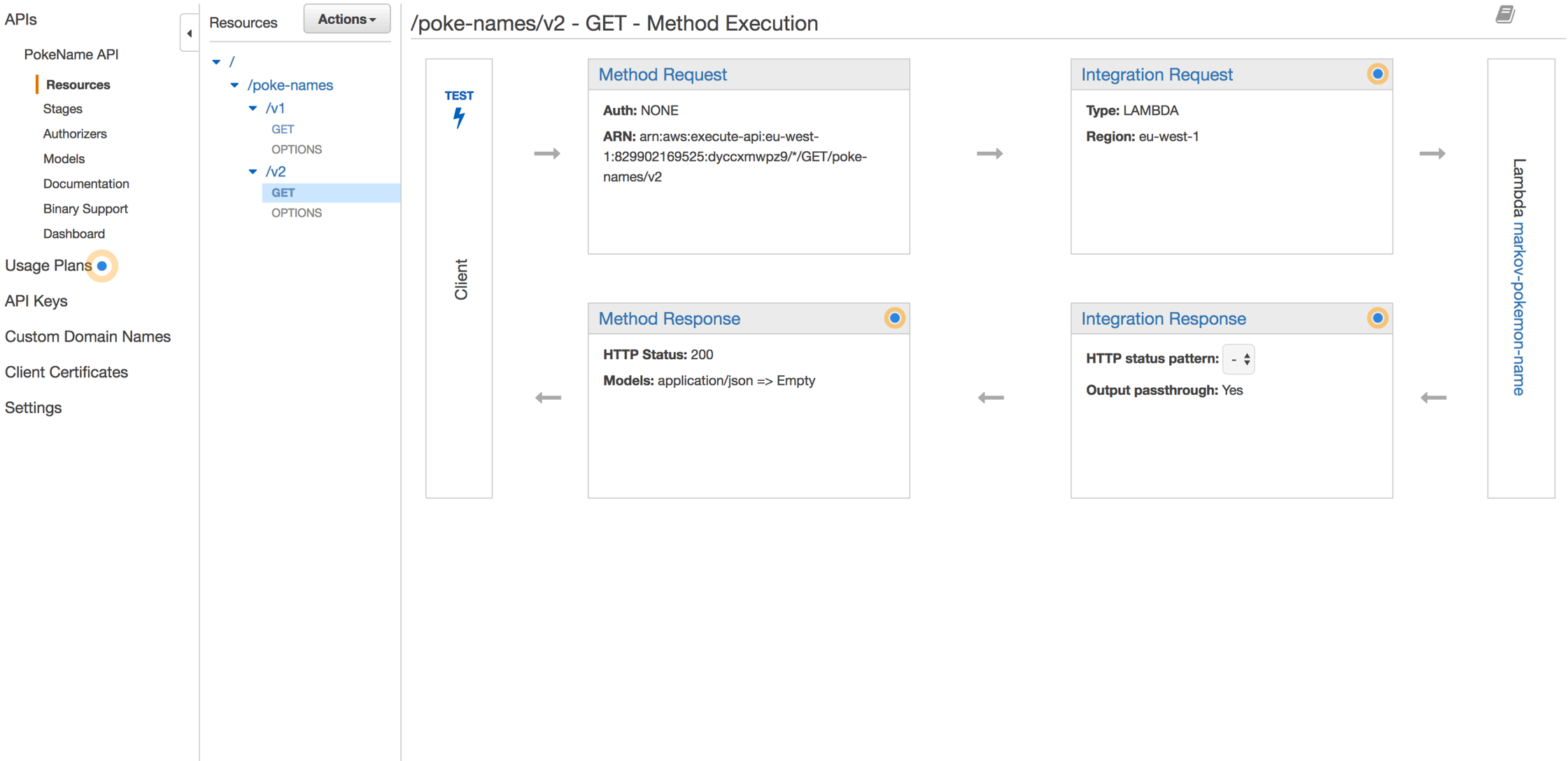
**Code entry type** Upload a .ZIP file

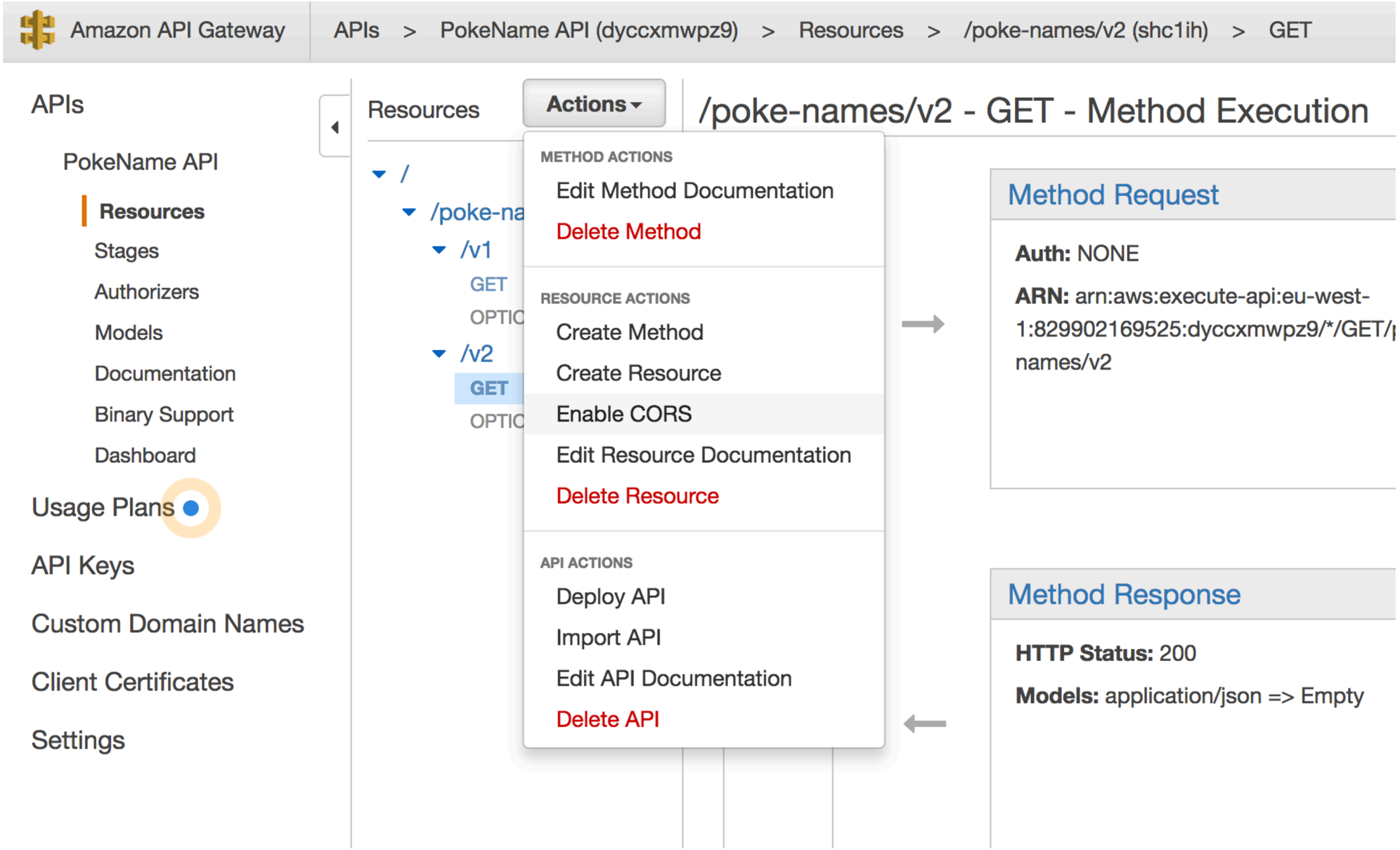
**Function package*** Upload

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
/v1
GET
OPTIONS
/v2
GET
OPTIONS
/poke-names/v2 - GET - Method Execution
TEST
Client
Method Request
Auth: NONE
ARN: arn:aws:execute-api:eu-west-1:829902169525:dyccxmwpz9/*/GET/poke-names/v2
Integration Request
Type: LAMBDA
Region: eu-west-1
Lambda markov-pokemon-name
Method Response
HTTP Status: 200
Models: application/json => Empty
Integration Response
HTTP status pattern: -
Output passthrough: Yes


Amazon API Gateway
APIs > PokeName API (dyccxmwpz9) > Resources > /poke-names/v2 (shc1ih) > GET
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
/
/poke-na
/v1
GET
OPTIC
/v2
GET
OPTIC
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/
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.

# Quick cost analysis

$10 flask server $\rightarrow$ 100 $\mathrm{rq/s}$

Let's compare this to AWS lambda.

$$\underbrace{\underbrace{\underbrace{100 \times 60}_{\text{per minute}} \times 60}_{\text{per hour}} \times 8 \times 30}_{\text{per month}} \times \frac{\$0.1}{10^6} = \$8.64$$

# Downsides

I'm not so sure how much I like the logging/live debugging. This is still an issue that requires some engineering. Probably you want to push to elasticsearch/kibana and probably you'll still want something like sentry to keep an eye on the front end.

# Downside

It is a cloud service it can go down.

S3 was briefly down a few months ago. It pulled down my blog but also other websites like http://isitdowntoday.com/.

# Downsides

S3 was briefly down a few months ago. It pulled down my blog but also other websites like http://isitdowntoday.com/.

Internet responded via http://isisitdowntodaydowntoday.com.

# Downsides

S3 was briefly down a few months ago. It pulled down my blog but also other websites like http://isitdowntoday.com/.

Internet responded via http://isisitdowntodaydowntoday.com.

This website is now down.

# Downsides

So you've put your function on the internet? **THINK SECURITY!**

You still need to worry.

You need to *remember* to set CORS headers. Otherwise you might be able to reach the endpoint from outside of your website.

You need to still think about *auth*.

It may be a good idea to add cloudformation.

# Downsides

Suppose you get a DDOS attack.

In a server situation your server would go down.

In a lambda situation amazon will horizontally scale on your behalf.

# Downsides

Suppose you get a DDOS attack.

In a server situation your server would go down.

In a lambda situation amazon will horizontally scale on your behalf.

Monotoring makes sense but you may also want to look at throttling some traffic. There are features for this on AWS but you need to do read up on this.

# Quick DDOS cost analysis

When am I p0wned for $1000?

$$\underbrace{\underbrace{\underbrace{\underbrace{n}_{\text{per minute}} \times 60}_{\text{per hour}} \times 60 \times 24}_{\text{per month}} \times 30 \times \frac{\$0.1}{10^6}}_{} = \$1000$$

When $n = 3800$ you'll get near. If you have an enemy out there, this is double with a few servers.

# Part 3

# Mature ML/AWS Lambda Flow

# Now to improve

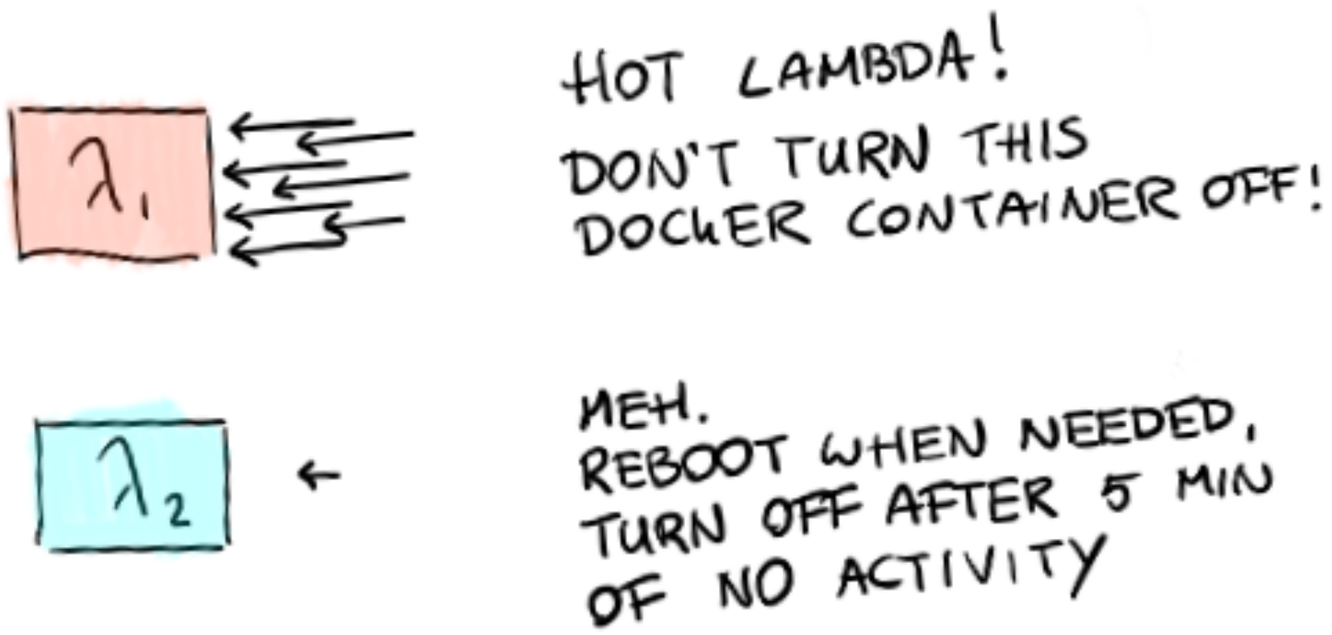
- Think about the future
- Allow for better python models
- Automate this workflow

The simple function that is live now is doing training per request! You as a user do not notice but it is suboptimal.

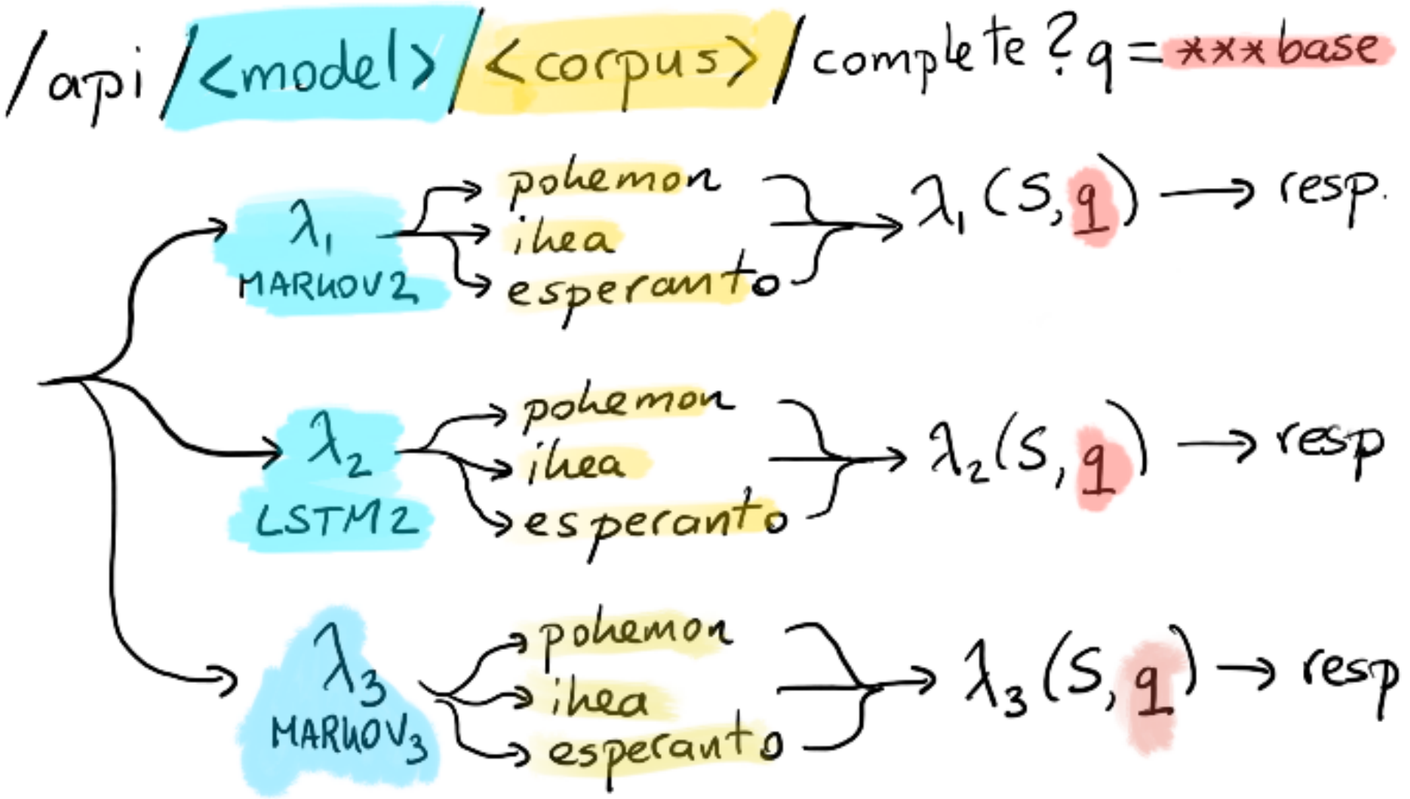
Also, current this is a request that has no input.

# Hot Cold

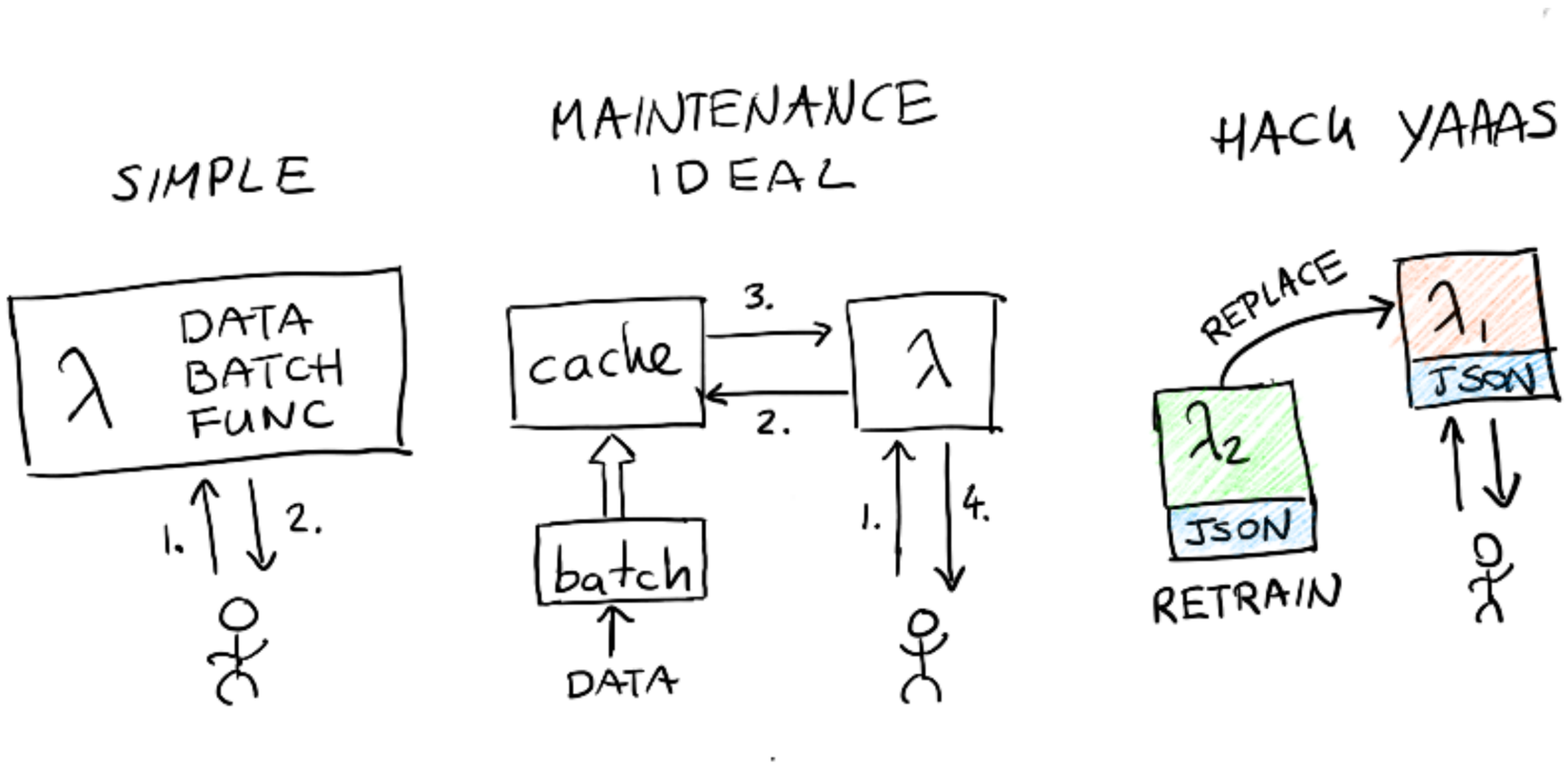
It may be smart to add more resources to the function.



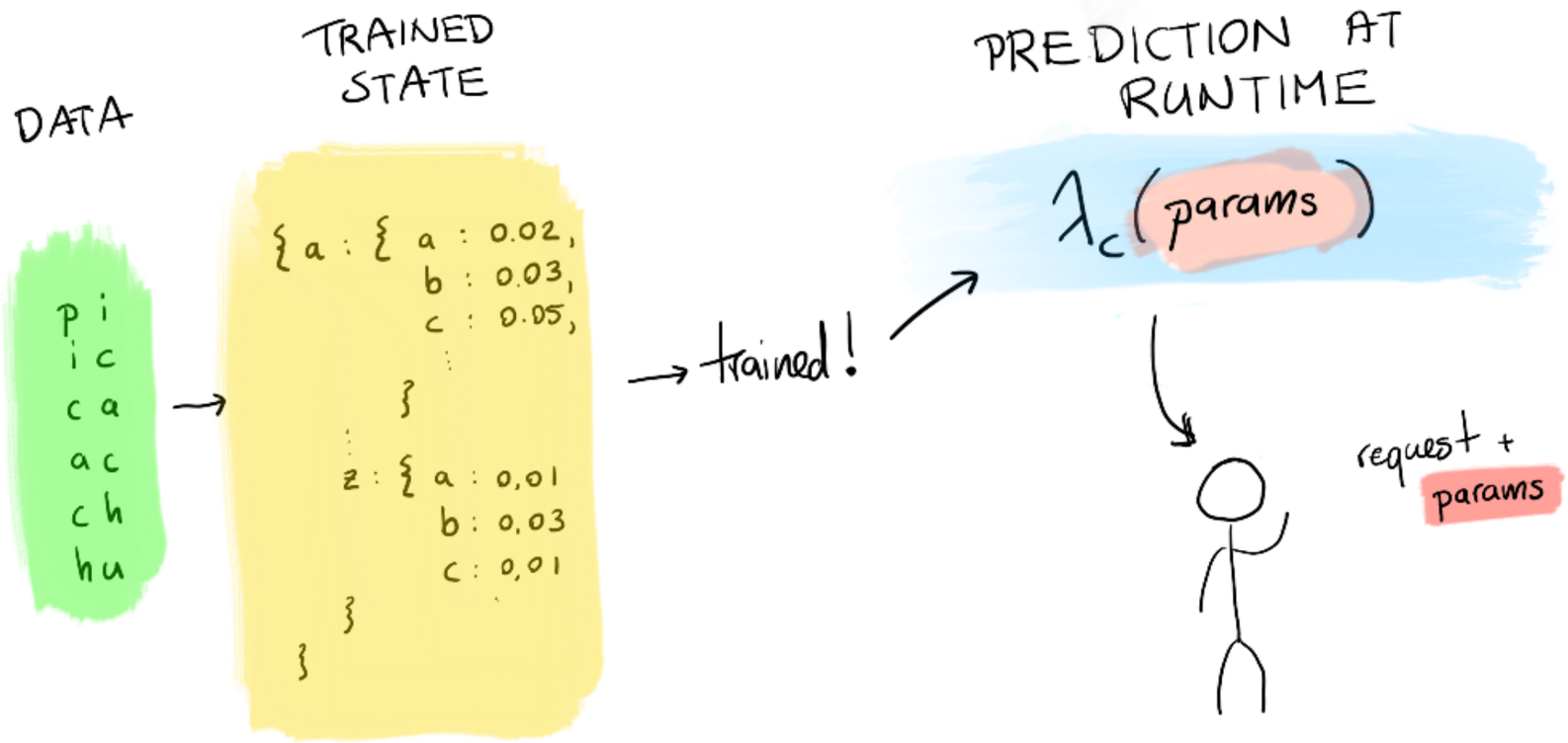
# Api Design


/api/<model>/<corpus>/complete?q=***base
λ1
MARKOV2
pokemon
ihea
esperanto
λ1(S, q) → resp.
λ2
LSTM2
pokemon
ihea
esperanto
λ2(S, q) → resp
λ3
MARKOV3
pokemon
ihea
esperanto
λ3(S, q) → resp

# Flow


SIMPLE
λ
DATA
BATCH
FUNC
1.
2.
MAINTENANCE
IDEAL
cache
3.
2.
λ
batch
DATA
1.
4.
HACK YAAAS
REPLACE
λ1
JSON
λ2
JSON
RETRAIN

# Steps


DATA
TRAINED STATE
PREDICTION AT RUNTIME
pi
ic
ca
ac
ch
hu
{a : { a : 0.02,
b : 0.03,
c : 0.05,
}
z : { a : 0,01
b : 0,03
c : 0,01
}
}
trained!
λc ( params )
request + params

# Steps

You'll also need to integrate this with a machine learning pipeline. Thusfar I've found that automating this with a custom command line app works the best for me.

fire? command line python app

```
> tnaas-learn config
project root path: /Users/code/Development/tnaas-learn
datasets folder: /Users/code/Development/tnaas-learn/datasets
model output folder: /Users/code/Development/tnaas-learn/model-json
```

# Other Options

I like testable automation.

```
> tnaas-learn generate-all-models
GENERATE ALL overwrite=True
will remove old files in /Users/code/Development/tnaas-learn/model-json
MARKOV-2 pokemon : started!
MARKOV-2 pokemon : model file will be /path/model-json/markov2/pokemon.json
MARKOV-2 pokemon : learning from 798 names
MARKOV-2 pokemon : succes new file has st_size 62.024K
MARKOV-3 pokemon : started!
MARKOV-3 pokemon : model file will be /path/model-json/markov3/pokemon.json
MARKOV-3 pokemon : learning from 798 names
MARKOV-3 pokemon : succes new file has st_size 2195.031K
```

# Other Options

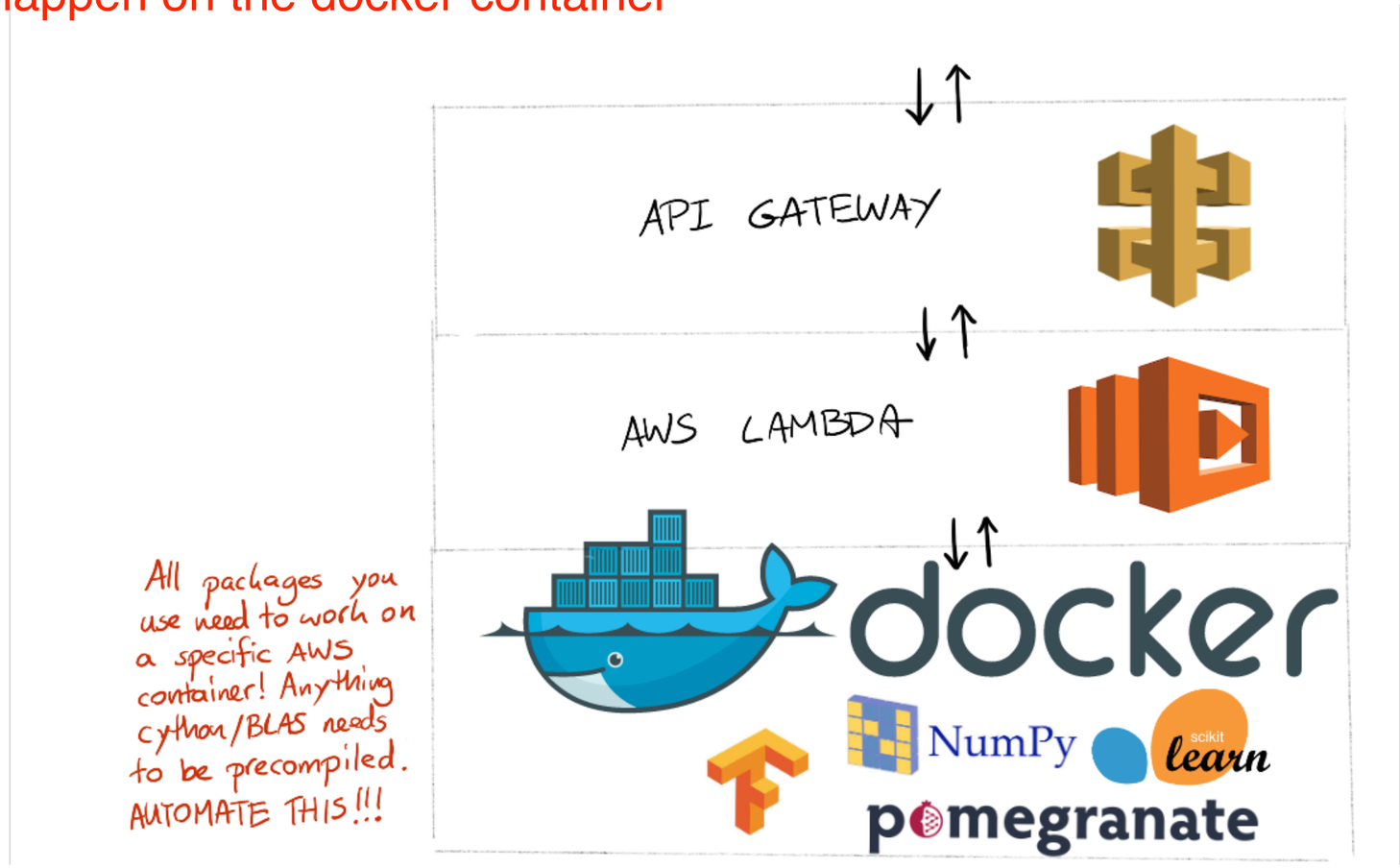
I like testable automation.

```
> tnaas-learn deploy-all-models
MARKOV-2 pokemon deployed!
MARKOV-3 pokemon deployed!
you have deployed!
```

This can all go to **git** and is as vendor independant as I can get it.

# Some internal details

this shit needs to happen on the docker container



# Some internal details

Pomegrenate 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

# Some internal details

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
}
```

# Some internal details

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 with cloudformation!

```
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
```

# Other Options

All this clicking stinks and cloudformation can be tricky. There are easier methods.

There are other methods of automating.

- codestar
- chalice
- zappa less risk of lockin

# Other Options

There are other cloud services out there.

- google. later!
- azure. meh.

I have a personal history of **megahard** issues with **microsoft**. I applaud the more open source direction they are taking now but I still remember their attitude in the 90s. It will take a lot of time and effort to convince me.

# Conclusion

These lambda cloud services seem cool. Even when you're not a devops/cloud kind of person. I may also start using AWS lambda as an endpoint for user tracking instead of sending it to google analytics. Requires thought.

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 or wait for google cloud.

# Conclusion

You could also just provide a server. Just sayin'.

Probibalistic graphical models rock though! Please try tangible models and heuristics before you do deep learning.

They are easier to design, debug and deploy.

These are things that I care for.