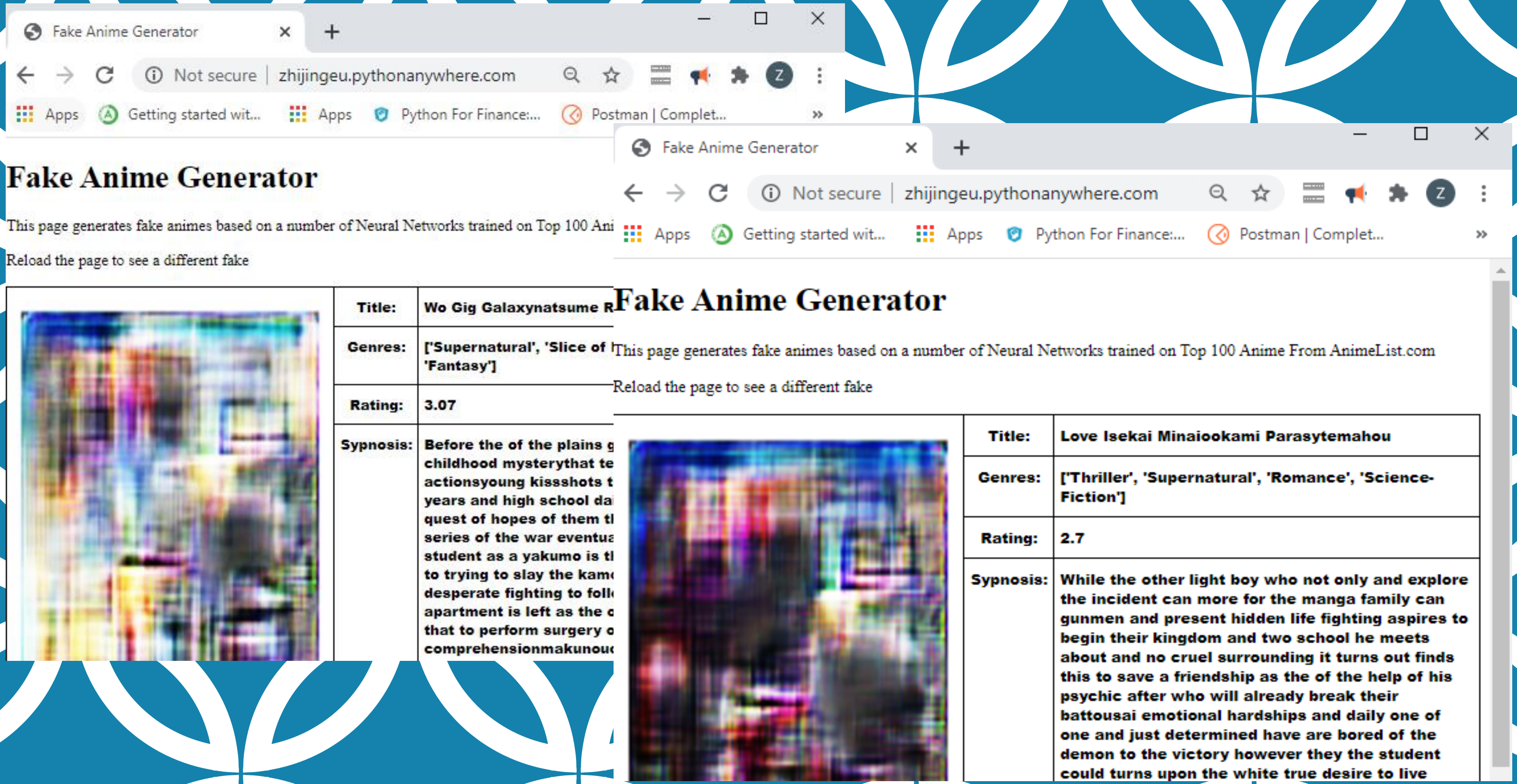


Fake Anime Generator

This page generates fake animes based on a number of Neural Networks trained on Top 100 Ani

Reload the page to see a different fake

Title:	Wo Gig Galaxynatsume R
Genres:	['Supernatural', 'Slice of 'Fantasy']
Rating:	3.07
Sypnosis:	Before the of the plains g childhood mysterythat te actionsyoung kissshots t years and high school da quest of hopes of them tl series of the war eventua student as a yakumo is tl to trying to slay the kam desperate fighting to foll apartment is left as the c that to perform surgery o comprehensionmakunou

Fake Anime Generator

This page generates fake animes based on a number of Neural Networks trained on Top 100 Anime From AnimeList.com

Reload the page to see a different fake

Title:	Love Isekai Minaiookami Parasytemahou
Genres:	['Thriller', 'Supernatural', 'Romance', 'Science-Fiction']
Rating:	2.7
Sypnosis:	While the other light boy who not only and explore the incident can more for the manga family can gunmen and present hidden life fighting aspires to begin their kingdom and two school he meets about and no cruel surrounding it turns out finds this to save a friendship as the of the help of his psychic after who will already break their battousai emotional hardships and daily one of one and just determined have are bored of the demon to the victory however they the student could turns upon the white true desire to live

http://zhijingeu.pythonanywhere.com/

My Key Lessons From (ATTEMPTING) To Build & Deploy A Simple ML App As Experienced By A N00b Programmer

Aug 2020

zhijingeu@yahoo.com

AGENDA

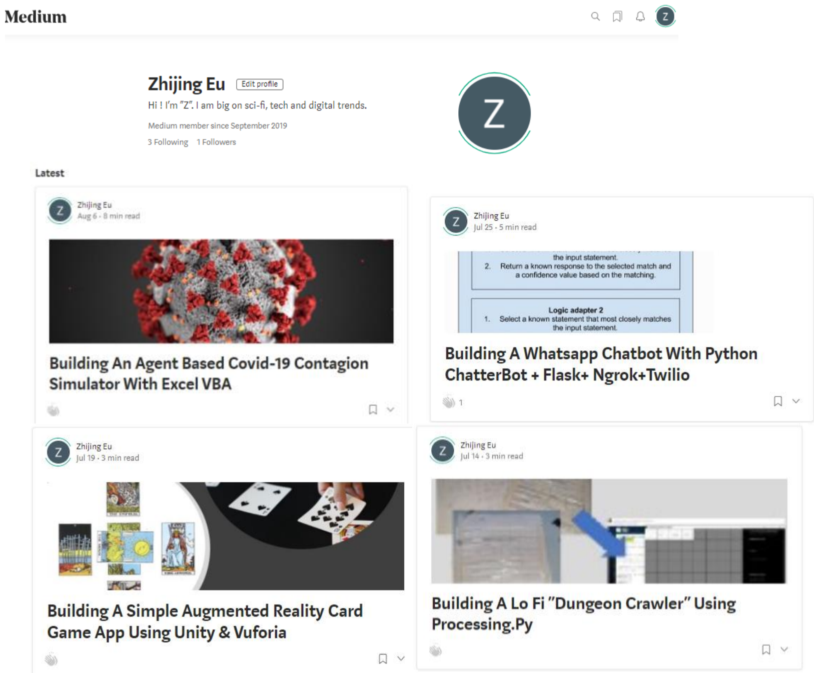
Part 1: Background – Inspiration & Motivation
https://youtu.be/4B8sGQgTUmg

Part 2: App Development & Deployment Workflow & Tools Used
https://youtu.be/R_lVjGwUhIA

ATTEMPTING
Part 3: My Key Lessons From ^ To Build & Deploy A Simple ML App
https://youtu.be/_VNsYQfGz_l

1.BACKGROUND: MOTIVATION

- As a noob (non CS background) programmer / tech enthusiast , I have been shotgun style learning different toolsets (e.g. Python, Unity (C++), Android Studio (Java) , etc) with general interest in Machine Learning
- Have been building mini-apps for my own entertainment but these are usually ‘stand-alone’ and local (on my own PC/phone only)
- This time , I wanted to build & deploy a rough demo of Machine Learning application to the web with intention to:-
  - Learn by doing across all the different steps in the journey
  - Try to focus on using a Neural Network solution
  - Have some fun and (hopefully) provide a few seconds of entertainment to others



Zhijing Eu

Hi ! I’m ”Z”. I am big on sci-fi, tech and digital trends.

Medium member since September 2019

3 Following 1 Followers

Zhijing Eu
Aug 6 · 8 min read

Building An Agent Based Covid-19 Contagion Simulator With Excel VBA

Zhijing Eu
Jul 25 · 5 min read

the input statement.
2. Return a known response to the selected match and a confidence value based on the matching.

Logic adapter 2
1. Select a known statement that most closely matches the input statement.

Building A Whatsapp Chatbot With Python ChatterBot + Flask+ Ngrok+Twilio

1

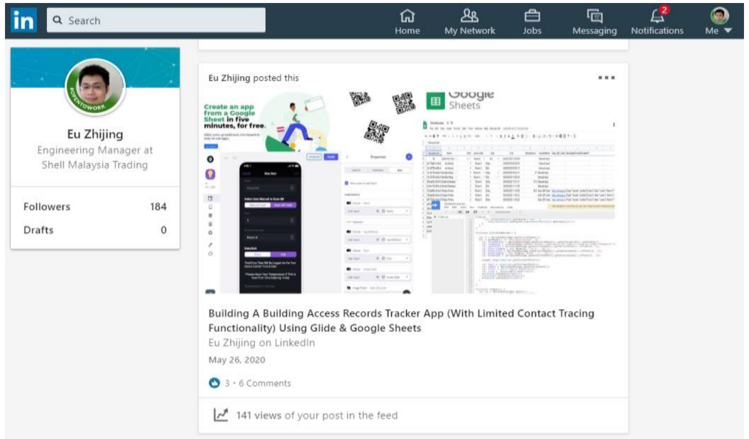
Zhijing Eu
Jul 19 · 3 min read

Building A Simple Augmented Reality Card Game App Using Unity & Vuforia

Zhijing Eu
Jul 14 · 3 min read

Building A Lo Fi ”Dungeon Crawler” Using Processing.Py

https://medium.com/@zhijingeu



Eu Zhijing
Engineering Manager at Shell Malaysia Trading

Followers	184
Drafts	0

Eu Zhijing posted this

Building A Building Access Records Tracker App (With Limited Contact Tracing Functionality) Using Glide & Google Sheets
Eu Zhijing on LinkedIn
May 26, 2020

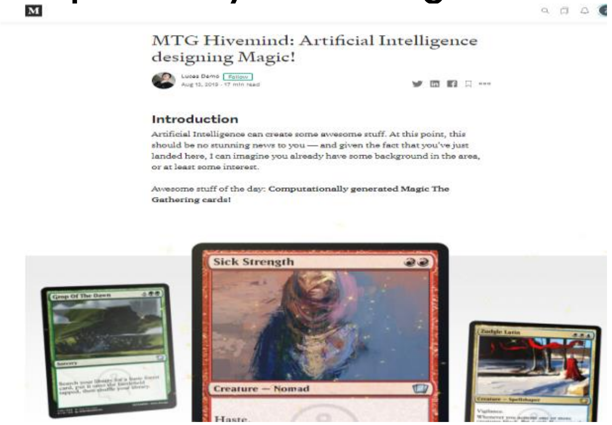
3 · 6 Comments

141 views of your post in the feed

https://www.linkedin.com/in/eu-zhijing-25a4362/

1.BACKGROUND: INSPIRATION

- Inspired by amazing work by others in Magic The Gathering to auto-generate 'artificial' cards



MTG Hivemind: Artificial Intelligence designing Magic!

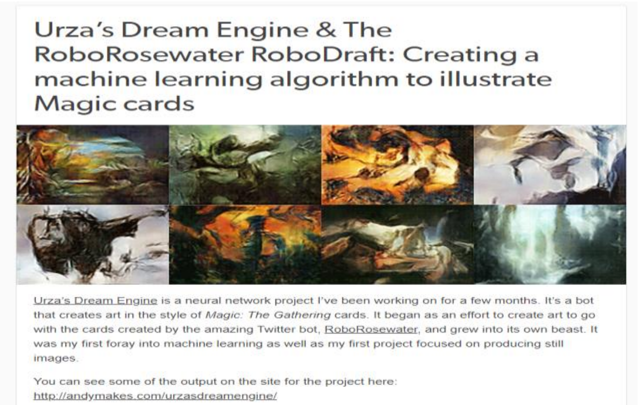
Lucas Demo Follow
Aug 13, 2019 · 17 min read

Introduction

Artificial Intelligence can create some awesome stuff. At this point, this should be no stunning news to you — and given the fact that you've just landed here, I can imagine you already have some background in the area, or at least some interest.

Awesome stuff of the day: **Computationally generated Magic The Gathering cards!**

https://medium.com/@lukbebalduke/mtg-hivemind-artificial-intelligence-designing-magic-372530640cc1



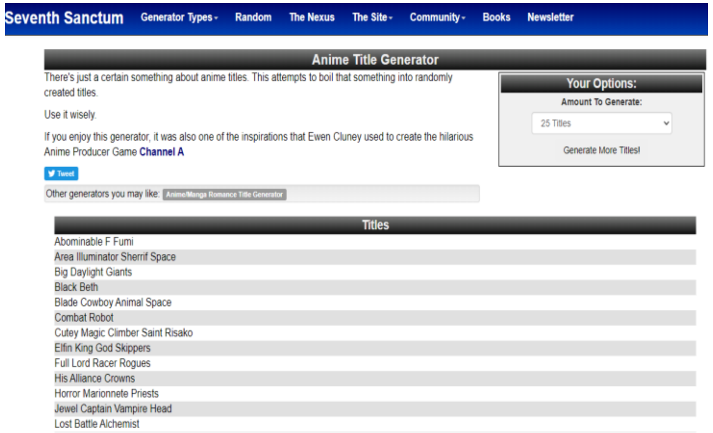
Urza's Dream Engine & The RoboRosewater RoboDraft: Creating a machine learning algorithm to illustrate Magic cards

Urza's Dream Engine is a neural network project I've been working on for a few months. It's a bot that creates art in the style of *Magic: The Gathering* cards. It began as an effort to create art to go with the cards created by the amazing Twitter bot, RoboRosewater, and grew into its own beast. It was my first foray into machine learning as well as my first project focused on producing still images.

You can see some of the output on the site for the project here:
http://andymakes.com/urzasdreamengine/

https://andymakesgames.tumblr.com/post/167733819029/urzas-dream-engine-the-roborosewater-robodraft

- However, I realised I could not find any sites that did the same for Anime.



Anime Title Generator

There's just a certain something about anime titles. This attempts to boil that something into randomly created titles.

Use it wisely.

If you enjoy this generator, it was also one of the inspirations that Ewen Cluney used to create the hilarious Anime Producer Game **Channel A**

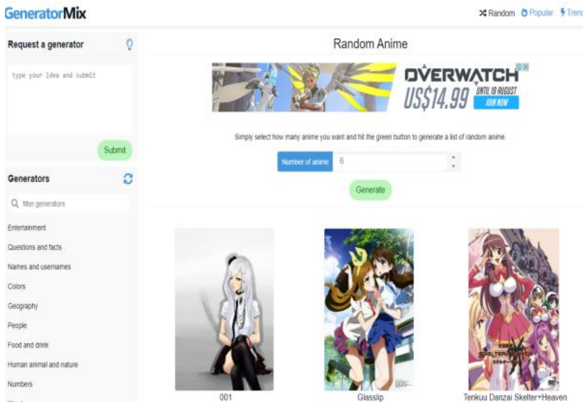
Other generators you may like: Anime/Manga Romance Title Generator

Your Options:
Amount To Generate: 25 Titles
Generate More Titles!

Titles
Abominable F Fumi
Area Illuminator Sherrif Space
Big Daylight Giants
Black Beth
Blade Cowboy Animal Space
Combat Robot
Cutey Magic Climber Saint Risako
Elfin King God Skippers
Full Lord Racer Rogues
His Alliance Crowns
Horror Marionnete Priests
Jewel Captain Vampire Head
Lost Battle Alchemist

Some sites existed but they generated only Titles and no descriptions or images.

https://www.seventhsanctum.com/generate.php?Genname=animetitle



Random Anime

Simply select how many anime you want and hit the green button to generate a list of random anime.

Number of anime 6

001
Glasslip
Tenkuu Danzai Skelter+Heaven

Others had just used existing anime images paired with randomly generated Titles

https://www.generatormix.com/random-anime

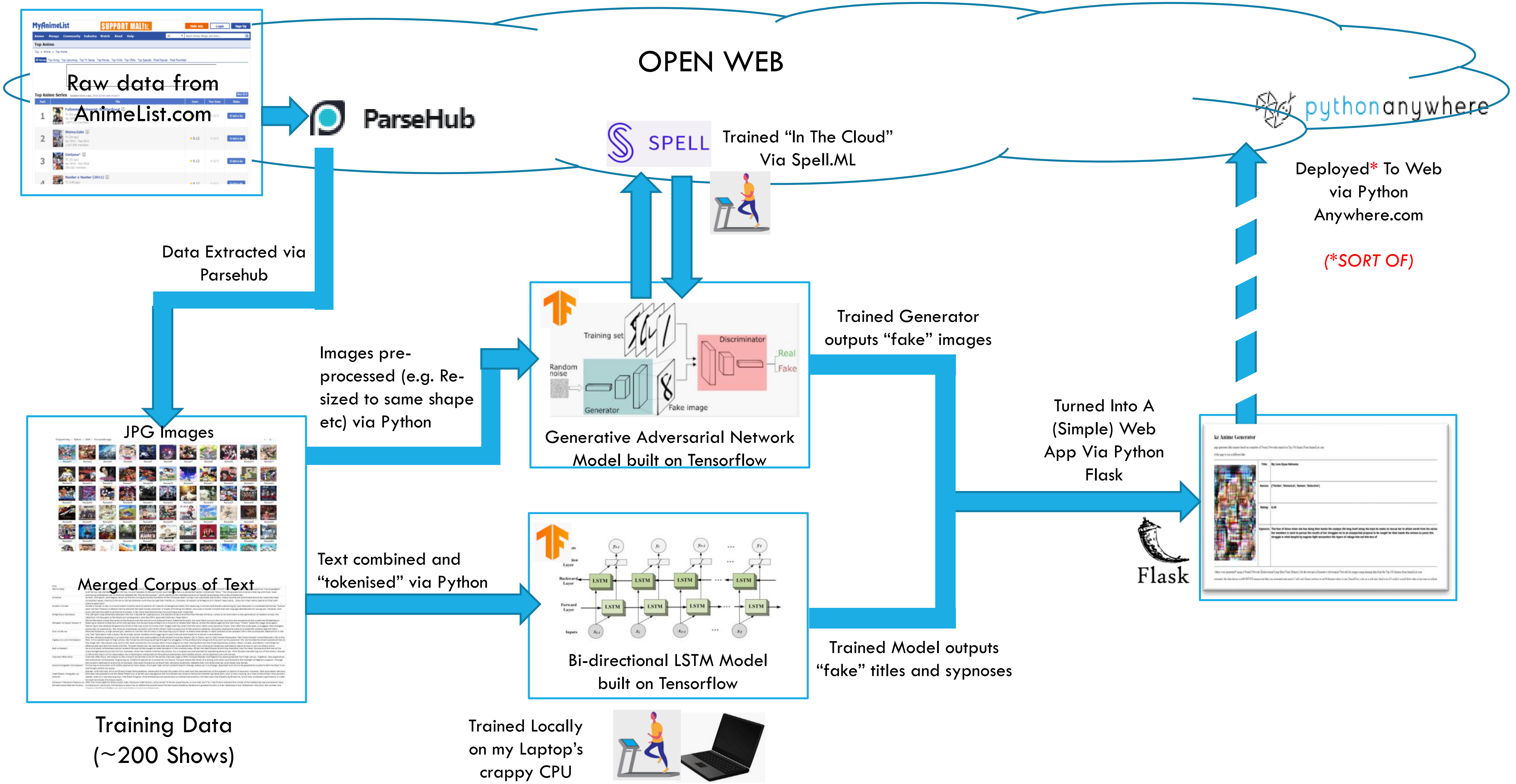
AGENDA

Part 1: Background – Inspiration & Motivation

Part 2: App Development & Deployment Workflow & Tools Used

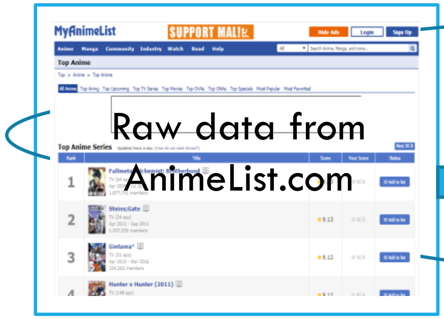
ATTEMPTING
Part 3: My Key Lessons From ^ To Build & Deploy A Simple ML App

ML Application Model Build + Deployment Flow



OPEN WEB
Raw data from AnimeList.com
MyAnimeList
ParseHub
SPELL
Trained "In The Cloud" Via Spell.ML
pythonanywhere
Deployed* To Web via Python Anywhere.com
(*SORT OF)
Data Extracted via Parsehub
JPG Images
Merged Corpus of Text
Training Data (~200 Shows)
Images pre-processed (e.g. Re-sized to same shape etc) via Python
Training set
Random noise
Generator
Discriminator
Real
Fake
Fake image
Generative Adversarial Network Model built on Tensorflow
Trained Generator outputs "fake" images
Text combined and "tokenised" via Python
Backward Layer
Forward Layer
Inputs
LSTM
Bi-directional LSTM Model built on Tensorflow
Trained Model outputs "fake" titles and sypnoses
Trained Locally on my Laptop's crappy CPU
Turned Into A (Simple) Web App Via Python Flask
Flask

2. ML Application Model Build + Deployment Flow



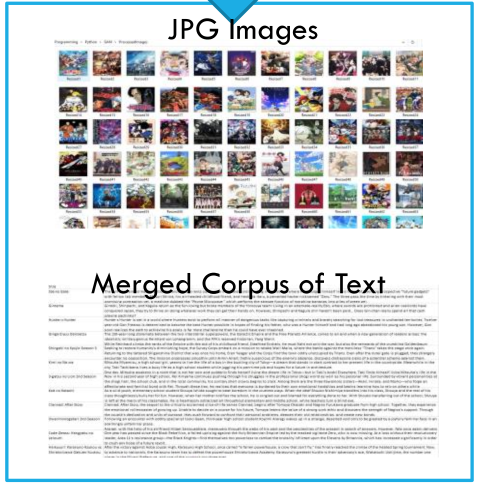
Rank	Title	Score	Your Score	Status
1	Fullmetal Alchemist: Brotherhood TV (64 eps) Apr 2009 - Jul 2010 1,877,750 members	★9.1	N/A	Add to list
2	Steins;Gate TV (24 eps) Apr 2011 - Sep 2011 1,327,555 members	★9.13	N/A	Add to list
3	Gintama° TV (51 eps) Apr 2015 - Mar 2016 324,263 members	★9.12	N/A	Add to list
4	Hunter x Hunter (2011) TV (148 eps)	★9.12	N/A	Add to list

Raw data from AnimeList.com

OPEN WEB

ParseHub

Data Extracted via Parsehub

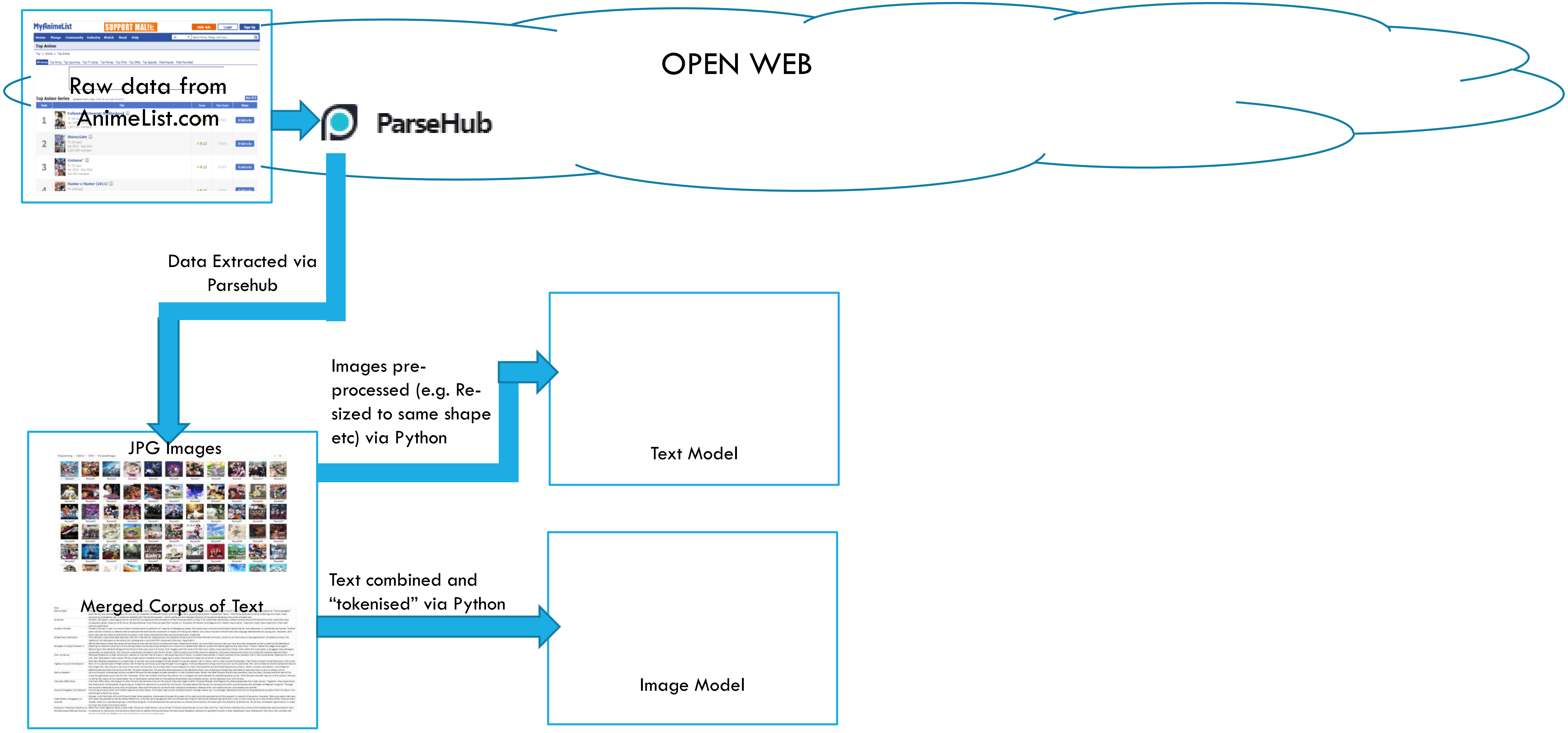


JPG Images

Merged Corpus of Text

Training Data (~100 Shows)

2. ML Application Model Build + Deployment Flow



OPEN WEB

Raw data from AnimeList.com

ParseHub

Data Extracted via Parsehub

JPG Images

Merged Corpus of Text

Images pre-processed (e.g. Re-sized to same shape etc) via Python

Text combined and “tokenised” via Python

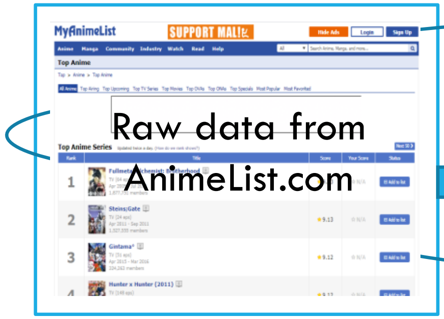
Text Model

Image Model

Training Data
(~100 Shows)

2. ML Application Model Build + Deployment Flow

OPEN WEB



Top Anime

Top > Anime > Top Anime

Top Anime Series Updated twice a day. (How do we rank shows?)

Rank	Title	Score	Your Score	Status
1	Fullmetal Alchemist: Brotherhood TV (64 eps) Apr 2009 - Jul 2010 1,877,750 members	[illegible]	N/A	Add to list
2	Steins;Gate TV (24 eps) Apr 2011 - Sep 2011 1,327,355 members	9.13	N/A	Add to list
3	Gintama° TV (51 eps) Apr 2015 - Mar 2016 324,263 members	9.12	N/A	Add to list
4	Hunter x Hunter (2011) TV (148 eps)	9.12	N/A	Add to list

Raw data from AnimeList.com

ParseHub

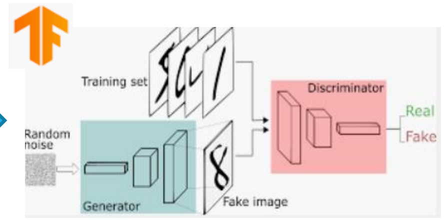
Data Extracted via Parsehub

JPG Images

Merged Corpus of Text

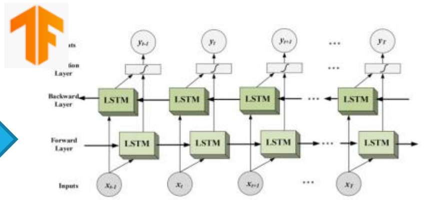
Training Data (~100 Shows)

Images pre-processed (e.g. Re-sized to same shape etc) via Python



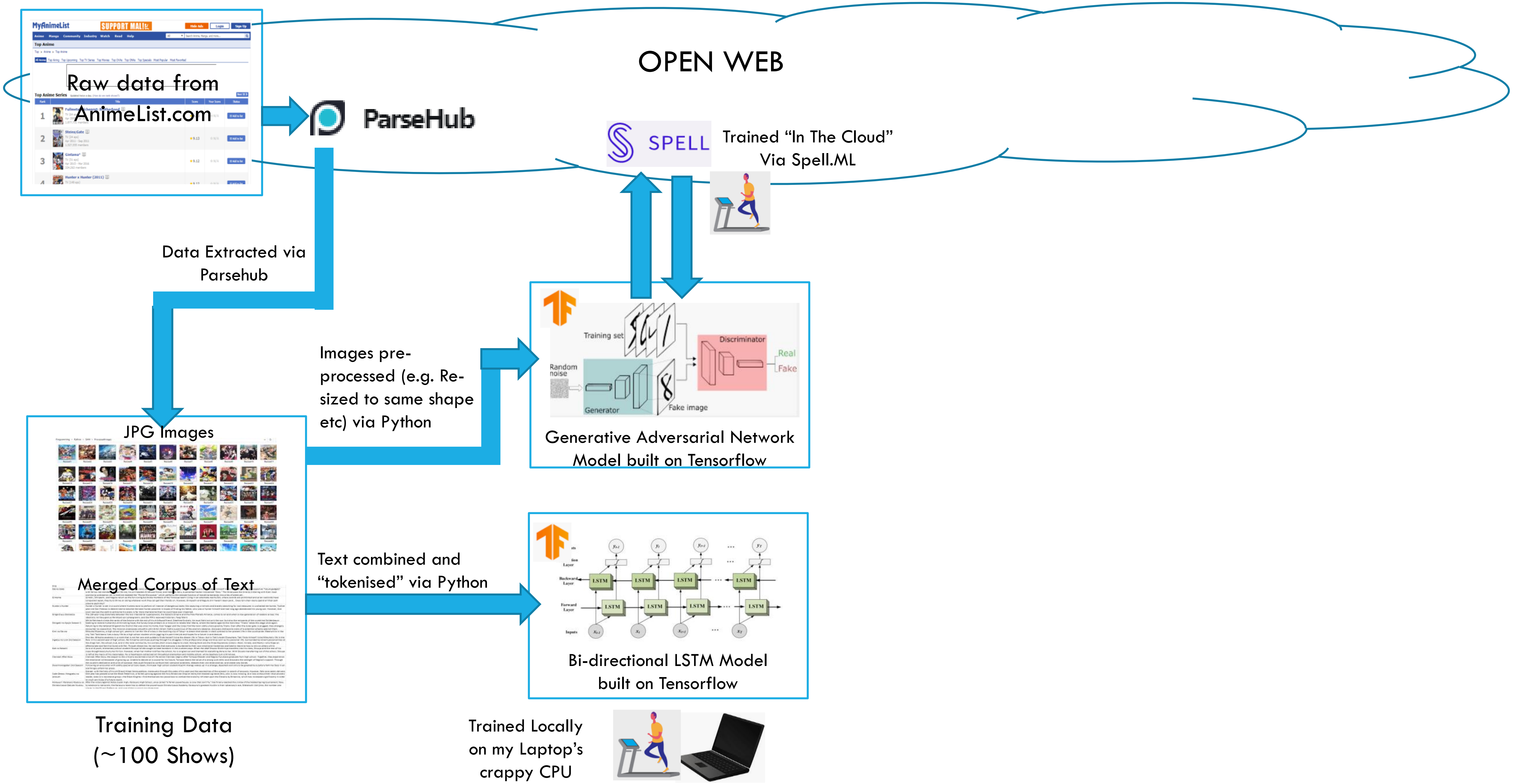
Generative Adversarial Network Model built on Tensorflow

Text combined and "tokenised" via Python



Bi-directional LSTM Model built on Tensorflow

2. ML Application Model Build + Deployment Flow



OPEN WEB

Raw data from AnimeList.com

ParseHub

Data Extracted via Parsehub

SPELL

Trained "In The Cloud" Via Spell.ML

JPG Images

Merged Corpus of Text

Training Data (~100 Shows)

Images pre-processed (e.g. Re-sized to same shape etc) via Python

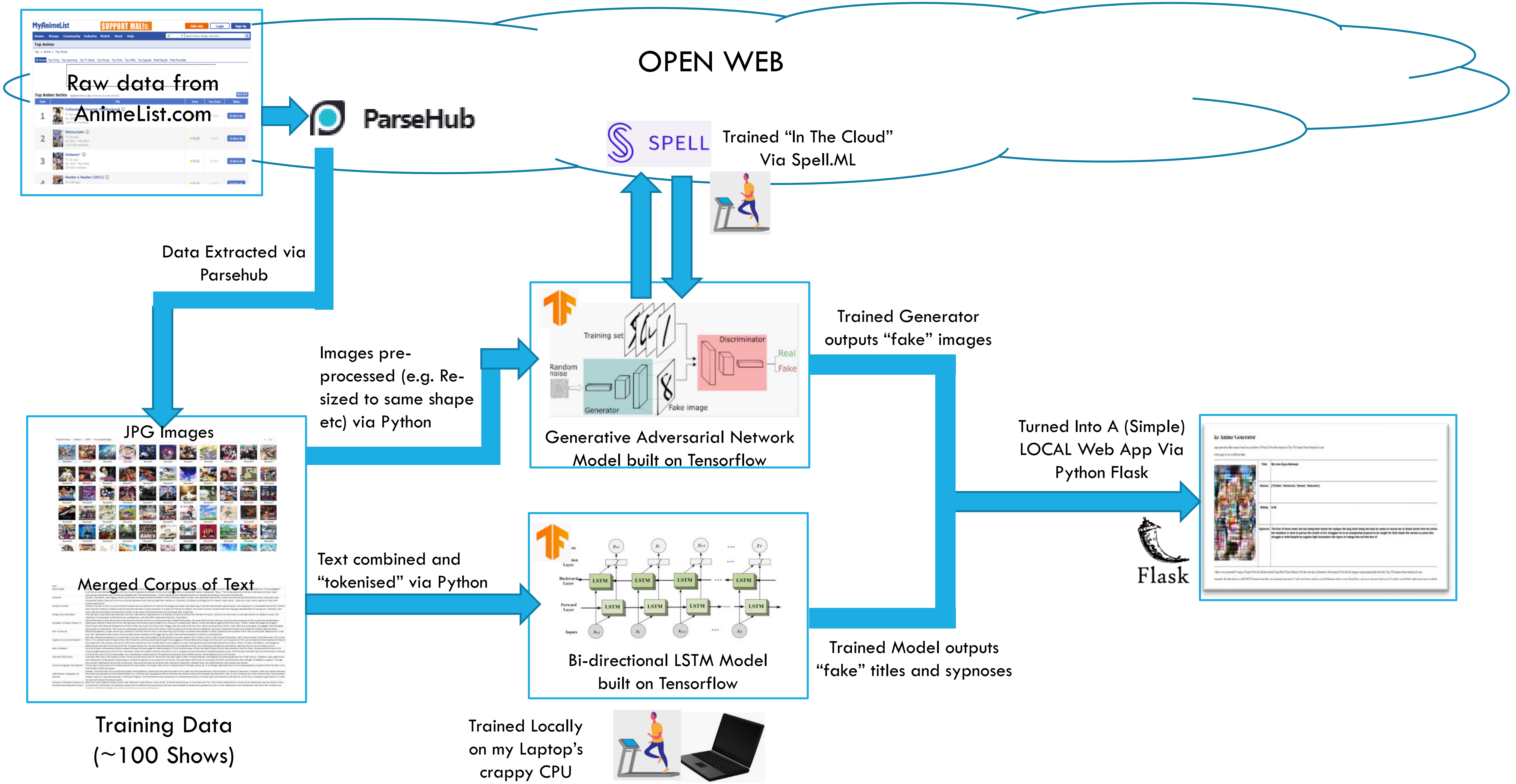
Generative Adversarial Network Model built on Tensorflow

Text combined and "tokenised" via Python

Bi-directional LSTM Model built on Tensorflow

Trained Locally on my Laptop's crappy CPU

2. ML Application Model Build + Deployment Flow



OPEN WEB

Raw data from AnimeList.com

Data Extracted via Parsehub

Trained "In The Cloud" Via Spell.ML

JPG Images

Merged Corpus of Text

Training Data (~100 Shows)

Images pre-processed (e.g. Re-sized to same shape etc) via Python

Text combined and "tokenised" via Python

Generative Adversarial Network Model built on Tensorflow

Bi-directional LSTM Model built on Tensorflow

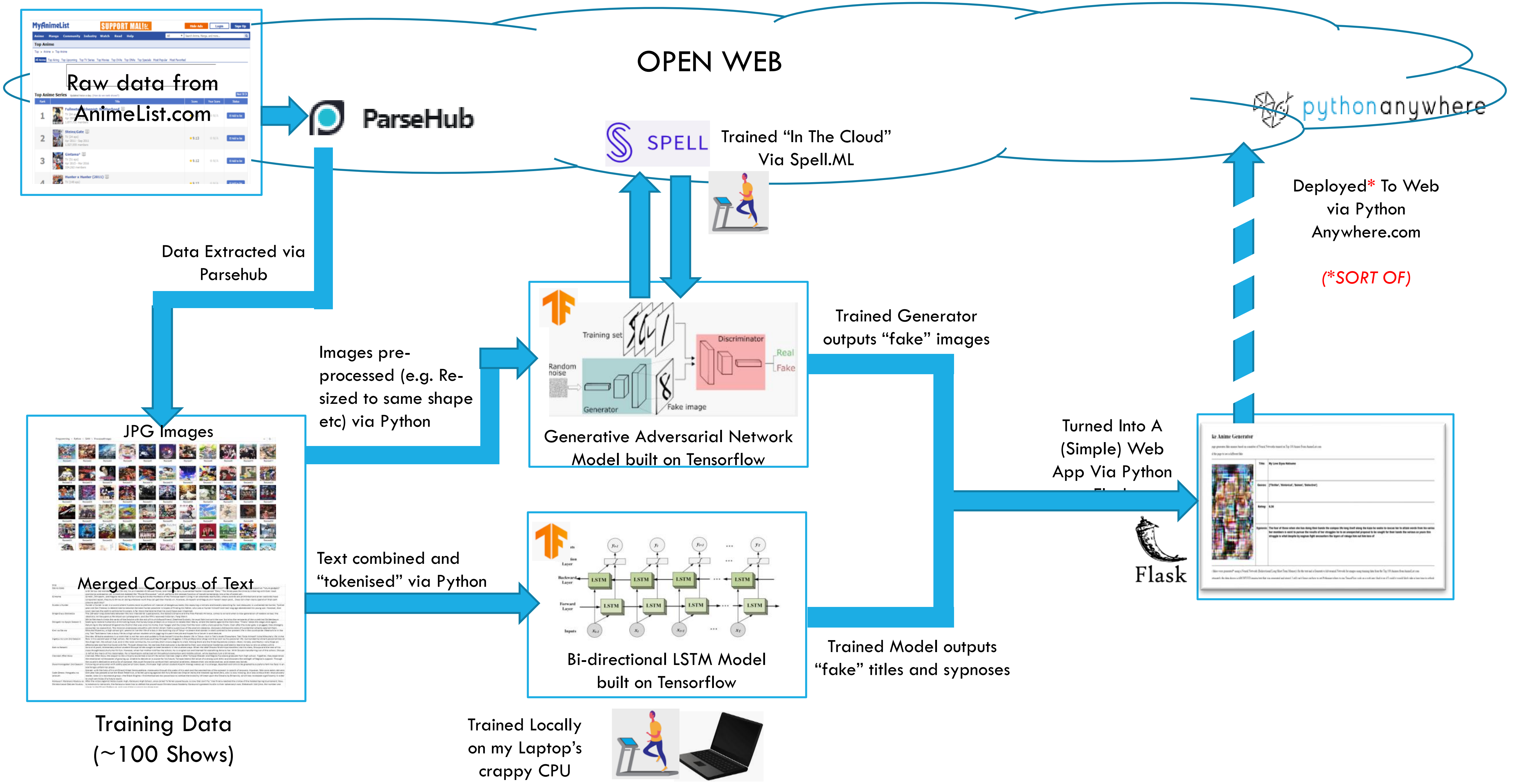
Trained Locally on my Laptop's crappy CPU

Trained Generator outputs "fake" images

Trained Model outputs "fake" titles and sypnoses

Turned Into A (Simple) LOCAL Web App Via Python Flask

2. ML Application Model Build + Deployment Flow



OPEN WEB

Raw data from AnimeList.com

Data Extracted via Parsehub

Trained "In The Cloud" Via Spell.ML

Deployed* To Web via Python Anywhere.com

(*SORT OF)

JPG Images

Merged Corpus of Text

Training Data (~100 Shows)

Images pre-processed (e.g. Re-sized to same shape etc) via Python

Generative Adversarial Network Model built on Tensorflow

Text combined and "tokenised" via Python

Bi-directional LSTM Model built on Tensorflow

Trained Locally on my Laptop's crappy CPU

Trained Generator outputs "fake" images

Trained Model outputs "fake" titles and sypnoses

Turned Into A (Simple) Web App Via Python Fl

AGENDA

Part 1: Background – Inspiration & Motivation

Part 2: App Development & Deployment Workflow & Tools Used

ATTEMPTING
Part 3: My Key Lessons From ^ To Build & Deploy A Simple ML App

3.MY KEY LESSONS LEARNT FROM *Attempting* TO BUILD & DEPLOY A SIMPLE ML APP

1) Characteristics of Training Data is key
2) Deep Learning can be computationally/hardware intensive
3) Deep Learning is easy to run but requires lot of math/theory knowledge to fine-tune and optimize
4) Trade-off of "build your own" vs 'buying' packaged solutions
5) End To End Development & Deployment Requires Wide Skillsets

PART 3.LESSONS LEARNT
1. CHARACTERISTICS OF TRAINING DATA IS KEY

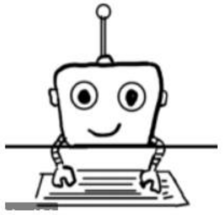
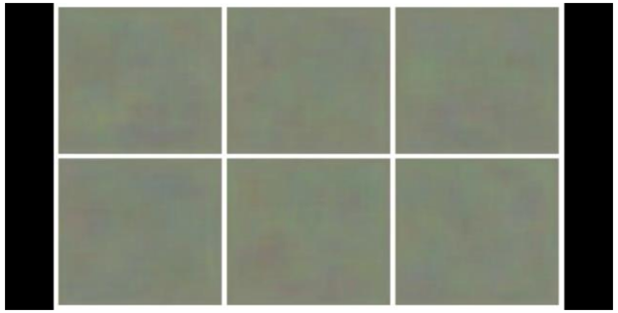
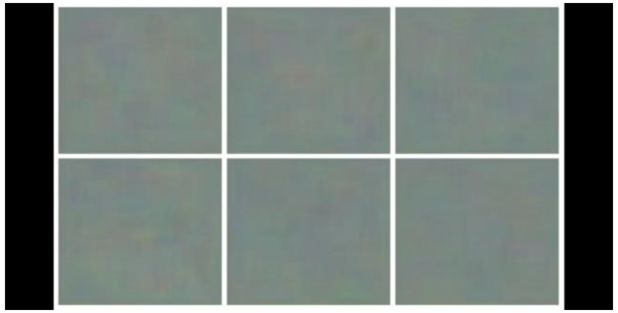


Image Source :https://machinelearnings.co

- Increasing no of iterations even from 1000 to 10,000 Epochs doesn't (seem to) help improve the image quality





- Maybe dataset too small ~200 images or should I have applied more pre-processed (e.g. cropping areas without features ? , color adjustment ?)

- More fundamentally – perhaps there are no defining traits across the dataset for the network to learn from ?

- Potentially could have grouped into categories eg Based on size of faces against foreground/background etc and train indiv models by category ?

- For text generator dataset pre-processing – maybe could have optimised the corpus generation - results were ok for title but not so much for sypnosis

PART 3.LESSONS LEARNT
2.DEEP LEARNING COMPUTATIONALLY/HARDWARE INTENSIVE

- For the GAN model – training on CPU on local laptop for just 1000 Epochs took almost 8-12 hours
- Eventually gave up and went looking for a Cloud service to do this. Lots of options on big Cloud players (Google Cloud, Amazon Web Service, etc) but decided on Spell ML for simplicity – took 2-3 hours for 10,000 Epochs on a GPU
- GANs seem inherently tricky to train – as 2 different networks being trained in tandem and many hyperparameters to optimise – am certain it can be optimised to be less computationally intensive but did not pursue.

https://www.istockphoto.com

PART 3.LESSONS LEARNT
3. DL REQUIRES A LOT OF KNOWLEDGE TO OPTIMIZE

- Implemented by copy-pasta-ing example code but did not optimise the hyper-parameters
- E.g.
  - How many nodes for the LSTM and no of dimensions for the Word Embedding Layer
  - Selection of Learning Rates and Choice of Optimiser other than ADAM
  - Choice of Loss Function e.g. For text generation – used Cat Cross Entropy as loss function but…

"The hardships and sense particular towards entire free adventures aiming the destined hotaru awaits limit basketball till she met cat beings"

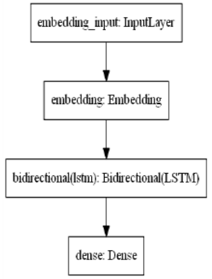
Sample Text After **1** Epoch Of Training
Loss: 6.4118 & Accuracy: 0.0712

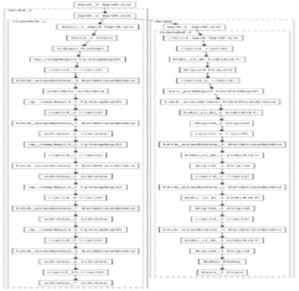
vs

"The owner of a fearsome attempt to their mission to impress haruko that not only reflect on a missing girl president to find the story of france"

Sample Text After **100** Epochs Of Training
Loss: 3.0129—accuracy: 0.4081

  - Use of early stopping or drop outs during training to avoid overfitting
  - Understanding of model architecture – text generation bi-directional LSTM vs image generation GAN


embedding_input: InputLayer
embedding: Embedding
bidirectional(lstm): Bidirectional(LSTM)
dense: Dense



  - GAN training troubles – I used Mini Batch Stochastic Gradient Descent– not sure if it was converging- Generator accuracy always low compared to Discriminator

PART 3.LESSONS LEARNT
4.TRADEOFF OF “BUILD YOUR OWN” VS PACKAGED SERVICES

Source : 123rf.com

- Build your own is FREE but still costs YOUR OWN TIME

- Packaged services good but normally the useful functionality is behind a pay-wall

- In retrospect, maybe could have been better off, using an end-to-end solutions from one of the major cloud players as integration may be easier

- However mid-way solutions (using mix-match’ed packaged services and build own ‘clean-up’ to post process) can still work as long as aware of what the trade-offs are

PART 3.LESSONS LEARNT
5.E2E DEPLOYMENT NEEDS AN DIFFERENT SKILL SETS

Source : PublicDomainPictures

- Even at start for the dataset building – EFFICIENT web-scraping requires knowledge on how to set Request Headers and use Proxy IP-s to avoid being 'booted' out as a robot by website

- With Flask – ran into some issues with caching as initially wanted the images to be generated "on the fly" and stored in memory/deleted upon reloading but realised it was harder than just serving new static files each time

- Professionally more complex applications are deployed in containerized strategy but even for a simple case
  - PythonAnywhere does not allow integration of Tensorflow in web app <LINK>
  - Advice by admin is to "modularize" the prediction portion into a separate 'always on task and deploy it separately or (or call an API'ed version of the TF model) so the flask app is front end only
  - Additional learning curve barriers in order to use Task Queue managers like Celery with Python Anywhere or mix-match solutions (e.g. Heroku , Google App Engine , etc)

- Ended up with a "fake" solution that cycles through archived data generated from the models

CONCLUSION – SO WAS IT WORTH IT ?

- Short answer - Yes

Learn by doing across all the different steps in the journey	✓
Get familiar with Neural Network solutions	✓
Have some fun	✓
Provide a few seconds of entertainment to others	?

- My key insight was that there are MANY different ways to solve a problem and it really depends on understanding the user behaviour e.g. anticipating the level of traffic , how people will use the results , etc and deciding what you can live with (reliability , latency , security/privacy concerns , etc)