

DATA BOT BOX

bitfusion

'Build your Own AI-Driven Bot' Workshop

A Brief History

- Chatbots were developed from the artificial intelligence sub-discipline of Conversational Informatics first initiated in 1960s
 - Conversational informatics is the study of communication between:
 - Humans and other humans
 - Humans and intelligent agents (artificial intelligence-driven agents)
 - Intelligent agents (artificial intelligence-driven agents)
 - Key ‘forefathers’ of Conversational Informatics include:
 - Toyoaki Nishida, Atushi Nakazawa, Yoshimasa Ohmoto, Yasser Mohammad, many others
 - Modern conversational agents or Chatbots sometimes talk to humans or even to each other!

Conversational Informatics as a Discipline

- Conversational informatics combines the study of:
 - Linguistics
 - Natural Language Processing
 - Machine Learning, Neural Networks
 - Cognitive Processes such as Intent
 - Speech Act Theory
 - Non-verbal communication
 - Implicit and explicit communication



bitfusion

Goals of Conversation



- Conversation is a tool both humans and artificial agents use to achieve specific goals including:
 - *Entertainment*: conversations and language can be considered like playing a ‘game’ with another person or group of people
 - *Social*: Building a joint story to share in a community
 - *Understanding*: Articulating tacit thoughts and examining other’s tacit thoughts
 - *Sales and Marketing*: Facilitation of transactional relationships (buying, selling goods for example)
 - *Information gathering*: Survey, testing
- **The Turing Test:** Developed by Alan Turing in 1950, is a test of a machine’s ability to exhibit intelligent behavior equivalent to, or indistinguishable from that of a human

Common Elements of Conversation

- Asking
- Informing
- Asserting
- Proposing
- Summarizing
- Checking
- Building
- Including
- Excluding
- Self-promotion
- Supporting
- Disagreeing
- Avoiding
- Challenging
- Attacking
- Defending
- Blocking
- Building rapport



DATA BOT BOX

Data Bot Box, LLC Copyright 2017

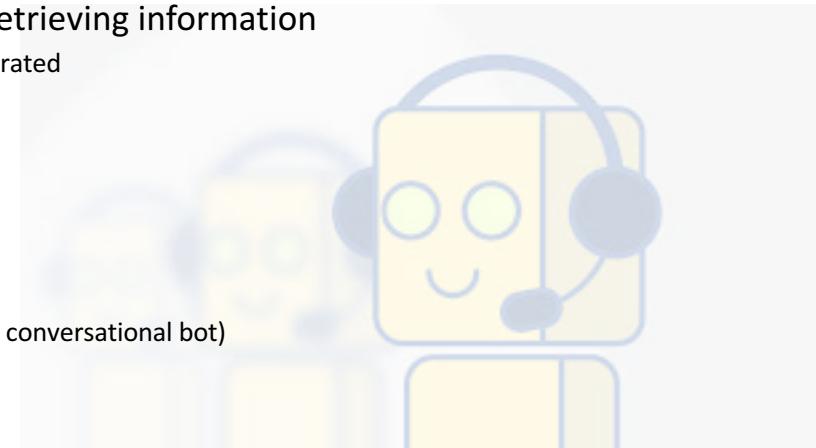
bitfusion

An Overview of Conversational Agents

- Common Chatbot/Conversational Agents:
 - Intent-driven
 - Information Retrieval
- Types of Chatbot Dialog Generation –
 - Generative (the most challenging)
 - Conversation is generated de novo from single words or groups of words called ‘utterances’
 - Narrative (the most common)
 - A conversation tree is generated and a story board or ‘narrative’ is created
 - Design thinking is a helpful technique in creating this type of chatbot

Chatbot Conversation Tactics

- **Intent-Driven** – guiding a decision through understanding of customer's intent
 - Facebook messenger chatbots in which businesses can send targeted consumers promotional messages
- **Exploratory/Information Retrieval** – exploring a topic, retrieving information
 - Whole Foods Market chatbot for finding recipes with emoji incorporated
- **Debate** – arguing with zero sum
 - Heyday's chatbot using Donald Trump tweets
- **Discussion** – open conversation
 - Amazon Lex conversational interfaces (make your own open-ended conversational bot)
- **Deliberation** – joint decision making
 - X2AI AI chatbots for mental-health services
 - Narrative driven chatbots (more on this later)



Intent-Driven Conversational Agents

- Companies can interact with shoppers in real-time
 - Answer questions shoppers have about a product
 - Steer shoppers to a particular product or suite of products
 - Suggest ‘add-on’ products
 - No attempt to pass the Turing Test as these chatbots are clearly utilitarian



Information Retrieval Agents

- Examples:
 - Alexa, Siri, Google Home, Sirius (open source intelligent personal assistant)
 - Typically based upon knowledge graphs (more later)
 - Can easily retrieve information and answer complex questions
 - Do not display empathy well typically
 - Do not understand human emotion
 - Do not understand social cues
 - Does not pass the Turing Test



Types of Dialog for Conversational Agents

- Generative Dialog
 - Common use case Seq-to-seq on Tensorflow uses a recurrent neural net consisting of stacks of ‘Long Short Term Memory’ architecture (LSTM cells), and an attention mechanism
 - Requires large quantities of ‘utterances’
 - Typical dataset would be Ubuntu Dialog Corpus ~ 1 million multi-turn dialogs
- Narrative dialog
 - Dialog narrative is generated by writers based upon conversation trees
 - Software to assist with this process is available, but generally anyone who is familiar with design thinking or movie script writing techniques can write a good narrative
 - Popular software to assist with narrative dialog generation includes:
 - Pullstring
 - BotSociety.io

Generative Dialog

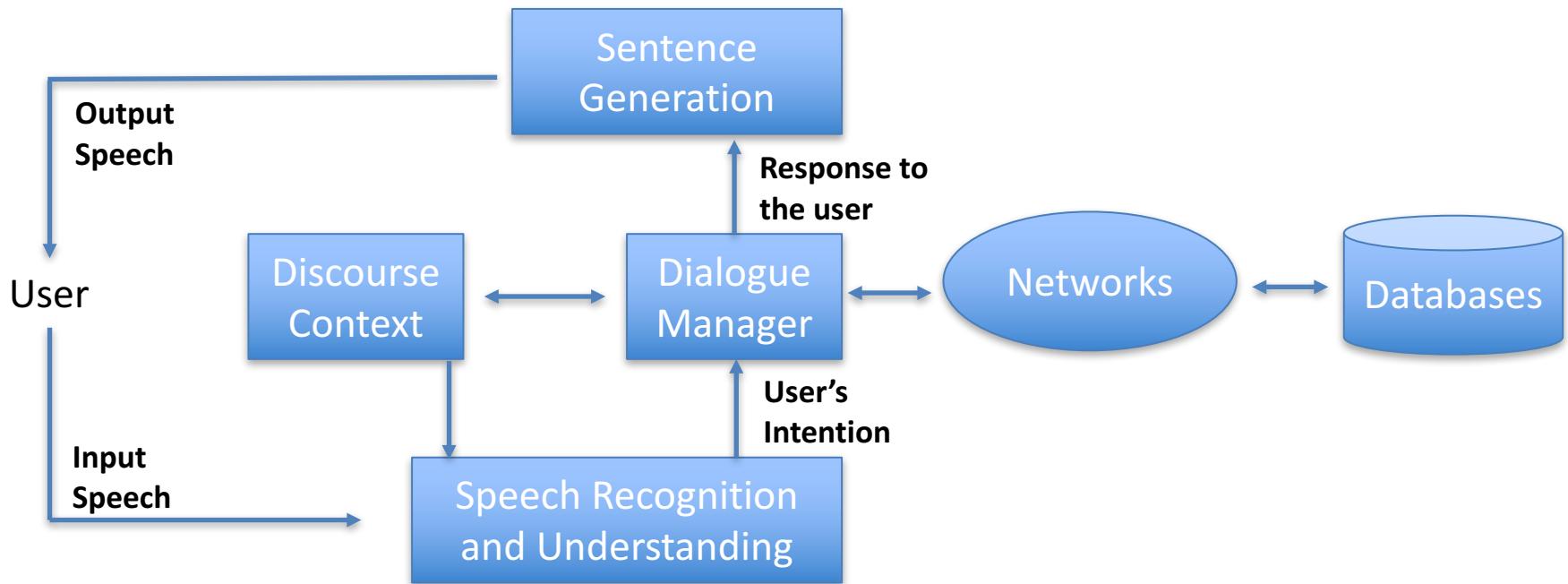
- Generative Dialog
 - Common use case Seq-to-seq on Tensorflow uses a recurrent neural net consisting of stacks of ‘Long Short Term Memory’ architecture (LSTM cells), and an attention mechanism
 - Requires large quantities of ‘utterances’
 - Typical dataset would be Ubuntu Dialog Corpus ~ 1 million multi-turn dialogs

Challenges with Generative Dialog

- Large quantities of data are required to obtain a relatively good chatbot
- Chatbots will often exhibit poor grammar
- Even a well-trained chat bot will occasionally misinterpret the intent of the human in the conversation

This intelligent agent does come closer to passing the Turing test as it can interpret on its own, the human's intent

Architecture of a Generative Dialog Conversational Agent



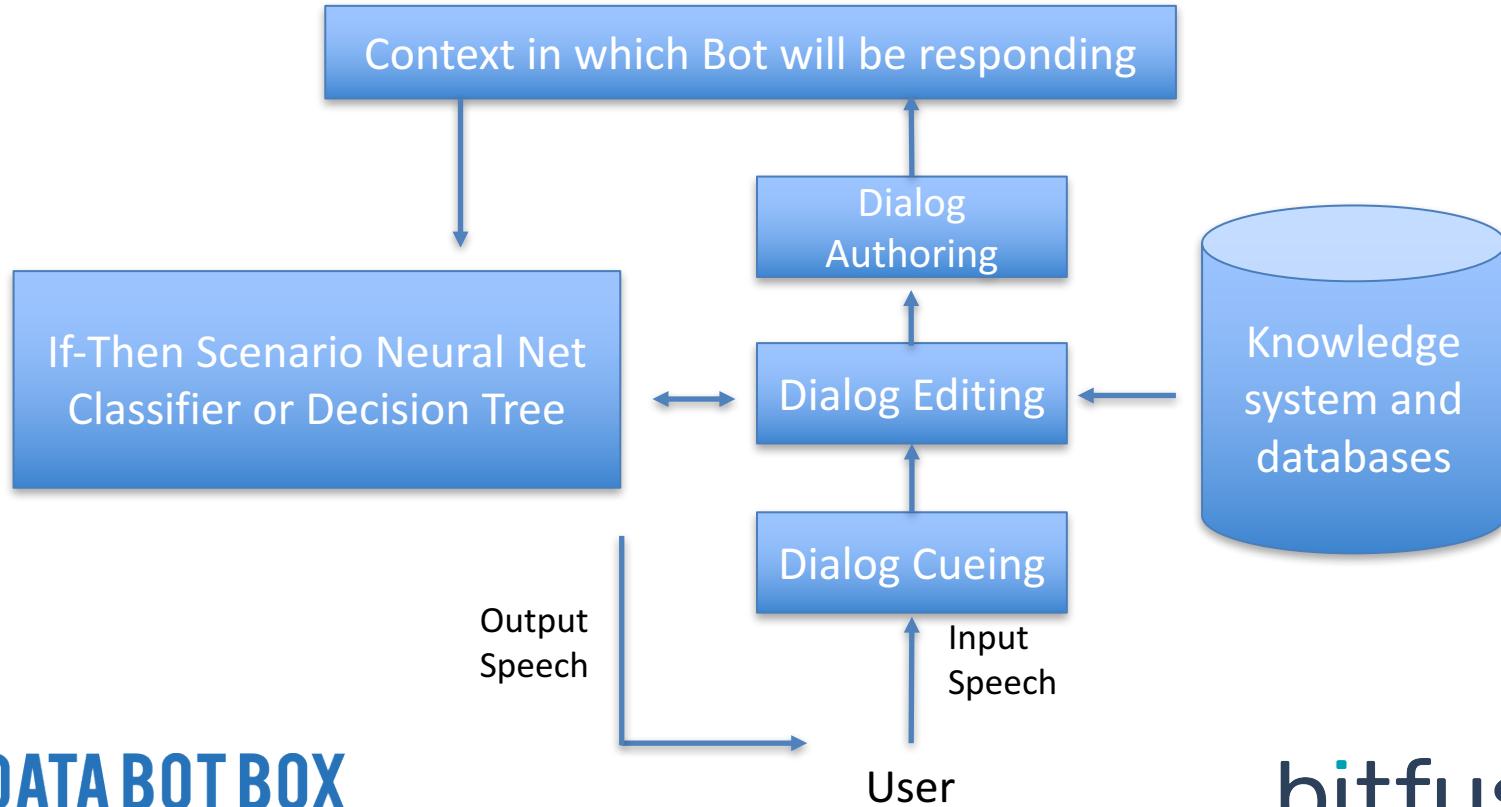
Narrative Dialog

- Narrative dialog
 - Dialog narrative is generated by writers based upon conversation trees
 - Software to assist with this process is available, but generally anyone who is familiar with design thinking or movie script writing techniques can write a good narrative
 - Popular software to assist with narrative dialog generation includes:
 - Pullstring
 - BotSociety.io

Challenges with a Narrative Dialog

- Dialog can seem contrived
- Narrative is always semi-situated
- Not all situations will be covered by the narrative so many times the chatbot will not know how to respond to items that are ‘off script’

Architecture of a Narrative Dialog Conversational Agent



Some Artificial Intelligence Advancements

ADVANCED CONVERSATIONAL AGENTS

Conversational Agents with Computer Vision

- Computer vision allows for facial recognition
- Computer vision allows for recognition of emotion
- Cameras are common on mobile phones, tablets and computers
- Most chatbots in the future will incorporate computer vision



Conversational Agents Leveraging Knowledge Graphs

- Siri, Alexa, Google Home all leverage Knowledge Graph
- A Knowledge Graph is a graphical representation of connections among thousands of bits of data
- Google databases are an example of one of the world's largest knowledge graphs



DATA BOT BOX

Data Bot Box, LLC Copyright 2017

bitfusion

Conversational Agents Embedded within Robotics

- Most robots have Conversational Agents Embedded
- Some famous ones include Pepper but more advanced one such as Q.bo use complex AI for movement, speech and emotive qualities



Artificial Intelligence at home
Open Source Code Robot & Linux OS

Some of Qbo's skills:
Stereoscopic vision, Speech Recognition System, Speech Synthesis System, Qbo's API & WEB control panel, WiFi & BLUETOOTH connections, obstacles, the robot avoids crashes and falls thanks to ultrasound sensors, AND MUCH MORE...

Q.bo®

Qbo Specifications

Size and Weight

- Height 456 mm
- Width 314 mm
- Depth 292.5 mm
- Weight 8.11 kg Aprox.
- Charge Autorecharging Dock Station

Head

- Ears 2 Omnidirectional microphones
1 Uni-directional microphone (noise reduction)
- Eyes 2 cameras HD@1
- 2 Eyes

Movements 4 Servos (Up-Down, Left-Right)

Motors 20 Leds

Wheels 1.5 cm diameter

Connectivity WiFi Pcb 802.11n & antenna

Controller 1 GHz pcb controller

Body

- Sensors 4 Ultrasonic sensors
1 Sharp Infrared sensor (detect a hole)
3 Generic infrared sensor (autocharge)
- Motors 2 DC Motors with Magnetic encoder (170 RPM)
- Wheels 1 Free Wheel (front)

Sound 2 High Quality Speakers

Controllers 1 Mini main board powered by ATOM x Nvda iON Graphics

Battery 1 Sealed battery (7.5Ah)

Robot Status 1 LCD Display 20x4

by thecorpora®



DATA BOT BOX

Data Bot Box, LLC Copyright 2017

bitfusion

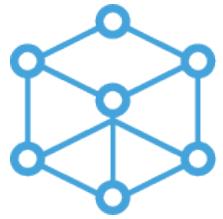
Emotionally Intelligent Conversational Agents

- Emotive Agents were first proposed by Justine Cassell
- The SARA project at Carnegie Mellon University Aims to integrate social intelligence into a Conversational Agent
- At Fjord, we built a multi-modal, emotionally intelligent bot for this year's SXSW.
 - <https://www.engadget.com/2017/03/16/party-bot-decides-whos-on-the-guest-list/>
 - <https://digtalrends.com/cool-tech/party-bot-ai-sxsw/#/2>



Steps To Making Your Own Conversational Agent

- Determine overall goal of the conversational agent
- Determine conversational tactics to be employed
- Determine style of dialog
- Determine expected outcomes
- Determine deep learning and other appropriate algorithms
- Determine platform and DevOps requirements
- Determine data requirements



DATA BOT BOX

bitfusion

WORKSHOP AND CODE

Jennifer Dawn Davis and Tim Gasper

Styles of Seq-to-Seq Chatbots

Google's Seq-to-seq Style

- Consists of series of Long term-Short-term Memory functions
- Does not need rules-based logic
- Uses Tensorflow library with Keras on the backend
- Remember Tensorflow does not place constraints on data input pattern
- Let's watch:
<https://youtu.be/SJDEOWLHYVo>

Facebook Innovation

- Dynamic memory network
- Creates a question-and-answer question for large volumes of data for chatbots
- Does not need rules-based logic
- Consists of series of stacked LSTM with an activity function linking them
- Remember neither Keras or Tensorflow place constraints on data input patterns
- Uses Keras
- Let's watch: <https://youtu.be/t5qgjJIBy9g>

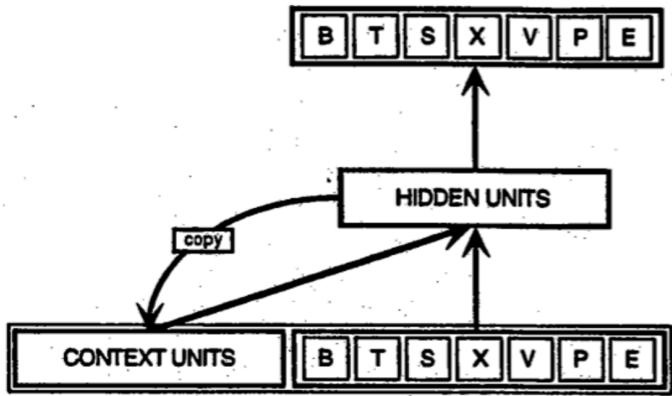


DATA BOT BOX

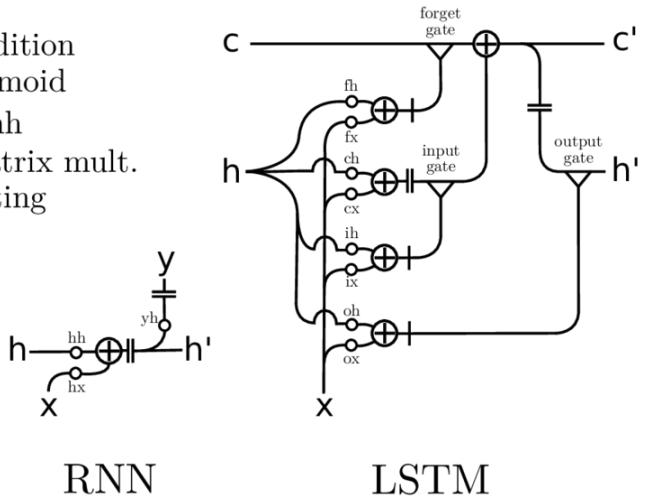
Data Bot Box, LLC Copyright 2017

bitfusion

Neural Nets & LSTM



⊕ addition
+ sigmoid
† tanh
○ matrix mult.
▽ gating



<https://deeplearning4j.org/lstm>

Advances in Humanoid Control & Perception, Stollenga, May 2016

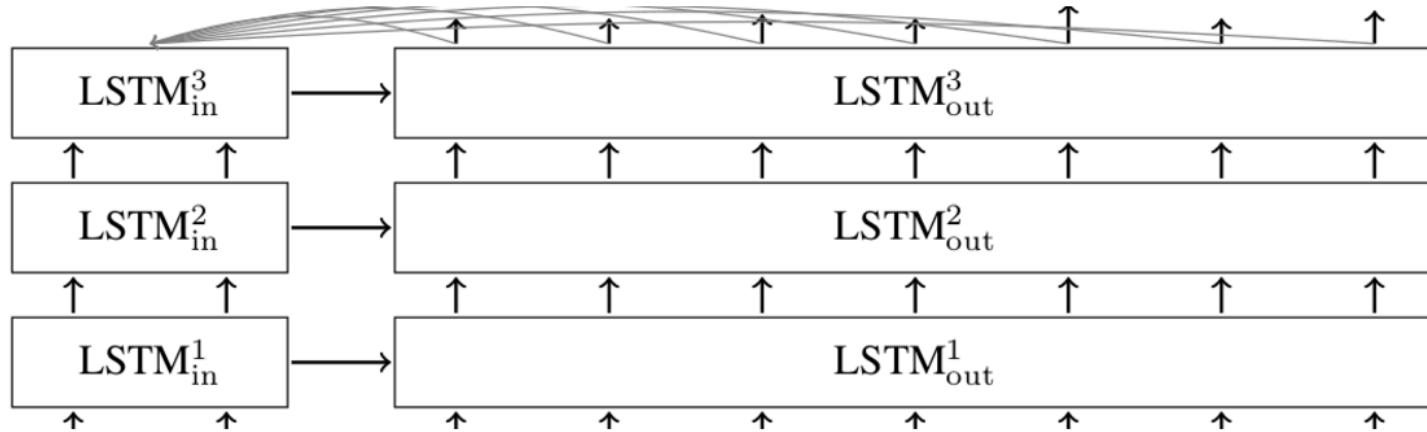


DATA BOT BOX

Data Bot Box, LLC Copyright 2017

bitfusion

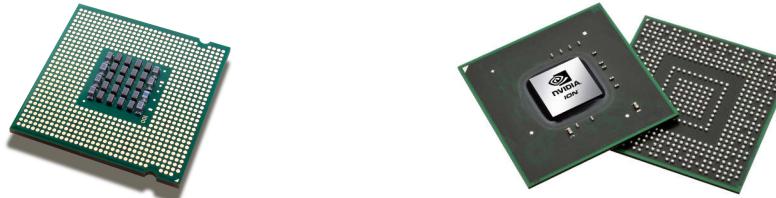
Seq-2-Seq LSTM Model from Tensorflow



<https://www.tensorflow.org/tutorials/seq2seq>

CPU vs. GPU and Why it Matters

- Chatbots that run on artificial intelligence require machine learning, deep learning or neural nets that run on LOTS of complex data (TBs at minimum)
- Training can take years if not done in a concurrent fashion
- GPUs will allow for a high level of concurrency
 - Highly parallel structure allows processing of large blocks of data concurrently (1000s of cores)
 - Graphical Processing Unit (GPU) Chip was first popularized by Nvidia in 1999 for Gamers
- CPU (central processing unit) efficiently interprets and executes most of the commands from the computer's other hardware and software (best is 8-16 cores, 16-32 threads)



DATA BOT BOX

Data Bot Box, LLC Copyright 2017

bitfusion

Services to Run Training for AI-Driven Chatbots

- Bitfusion: works out of the box, most AI software programs are pre-loaded
- Pipeline.io: needs to be set-up in a hadoop-like fashion, but will contain most AI software programs
- Google Cloud GPU: Very expensive and you are charged by the minute. Does not have many of the AI software uploaded
- IBM GPU Cloud: Charged by API Call
- Custom Centers (Typically have NVIDIA Tesla GPU) in CUDA environment

Examples: A practical understanding of Seq2Seq

- ***Setting up your virtual environment and Tensorflow***
 - In this class our bots will actually talk to themselves in future classes we can create bots that talk to humans
- ***Example I:*** This example is taken from TensorFlow website
 - We run the code step-by-step although on a regular laptop it will take several hours to run
 - At the end we show an idea of a fun way you can change this code to make a party game where you feed the bot a movie line and responds with another movie line!
 - Possibilities are endless, just use your imagination
- ***Example II*** we will demonstrate a chatbot that is made using twitter data and Seq2Seq, created by Suryadeepan Ram
 - Try changing the data source to Reddit's famous 'Shower Thoughts' for a fun and quirky bot. Stay tuned for our Github repository for an example of Ram's modified code using Reddit comments!
- ***Example III*** using the Bitfusion platform to train your Reddit bot!

Getting Started

**SETTING UP TENSORFLOW ON YOUR LAPTOP (NOT IDEAL, AND
DEMONSTRATES WHY WE NEED GPU ACCELERATED PLATFORMS LIKE
BITFUSION)**

The screenshot shows a web browser window with the URL jupyter.org/install.html in the address bar. The page title is "Installing Jupyter". The content includes sections on prerequisites, installing using Anaconda, and an alternative for experienced Python users using pip.

Prerequisite: Python

While Jupyter runs code in many programming languages, **Python** is a requirement (Python 3.3 or greater, or Python 2.7) for installing the Jupyter Notebook.

We recommend using the [Anaconda](#) distribution to install Python and Jupyter. We'll go through its installation in the next section.

Installing Jupyter using Anaconda and conda

For new users, we **highly recommend** installing [Anaconda](#). Anaconda conveniently installs Python, the Jupyter Notebook, and other commonly used packages for scientific computing and data science.

We recommend using the Anaconda distribution to install Python and Jupyter. We'll go through its installation in the next section.

- Download [Anaconda](#). We recommend downloading Anaconda's latest Python 3 version (currently Python 3.5).
- Install the version of Anaconda which you downloaded, following the instructions on the download page.
- Congratulations, you have installed Jupyter Notebook. To run the notebook:

```
jupyter notebook
```

See [Running the Notebook](#) for more details.

Alternative for experienced Python users: Installing Jupyter with pip

As an existing Python user, you may wish to install Jupyter using Python's package manager, [pip](#), instead of Anaconda.

First, ensure that you have the latest pip; older versions may have trouble with some dependencies:



Work — python2.7 — 80x24

```
Jennifers-MacBook-Pro:~ Work$ jupyter notebook
/Users/Work/anaconda/lib/python2.7/site-packages/cffi/model.py:526: UserWarning:
  'point_conversion_form_t' has no values explicitly defined; next version will r
efuse to guess which integer type it is meant to be (unsigned/signed, int/long)
    % self._get_c_name())
[W 12:18:29.401 NotebookApp] Widgets are unavailable. Please install widgetsnbx
tension or ipywidgets 4.0
[I 12:18:29.420 NotebookApp] Serving notebooks from local directory: /Users/Work
[I 12:18:29.421 NotebookApp] 0 active kernels
[I 12:18:29.421 NotebookApp] The Jupyter Notebook is running at: http://localhost:8888/?token=0d3d949646bf1fb3d52bcb8d13f9a228d0b38816cc9ed23a
[I 12:18:29.421 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[C 12:18:29.423 NotebookApp]
```

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:

```
http://localhost:8888/?token=0d3d949646bf1fb3d52bcb8d13f9a228d0b38816cc9
ed23a
```

```
[I 12:18:29.836 NotebookApp] Accepting one-time-token-authenticated connection f
rom ::1
```

```
□
```

[Logout](#)[Files](#)[Running](#)[Clusters](#)[Duplicate](#)[Shutdown](#)[Upload](#)[New ▾](#)

1



/ generative_chatbot



..



__MACOSX



cornell movie-dialogs corpus



Galvanize Workshop on Generative Dialog Chatbots 2017.ipynb



bad_words.txt

Text File

Folder

Terminal

Notebooks

Python 2

R

```
bash-3.2$ virtualenv --system-site-packages ~/tensorflow
New python executable in /Users/Work/tensorflow/bin/python
Installing setuptools, pip, wheel...done.
bash-3.2$ source ~/tensorflow/bin/activate
(tensorflow) bash-3.2$ █
```

```
(tensorflow) bash-3.2$ cd ~tensorflow/
(tensorflow) bash-3.2$ ls
bin          lib
include      pip-selfcheck.json
(tensorflow) bash-3.2$ git clone https://github.com/tensorflow/tensorflow.git
Cloning into 'tensorflow'...
remote: Counting objects: 191010, done.
remote: Total 191010 (delta 0), reused 1 (delta 0), pack-reused 191009
Receiving objects: 100% (191010/191010), 104.60 MiB | 13.38 MiB/s, done.
Resolving deltas: 100% (146287/146287), done.
Checking connectivity... done.
(tensorflow) bash-3.2$ ls
bin          lib          tensorflow
include      pip-selfcheck.json
(tensorflow) bash-3.2$ git clone https://github.com/tensorflow/models.git
Cloning into 'models'...
remote: Counting objects: 4817, done.
remote: Total 4817 (delta 0), reused 0 (delta 0), pack-reused 4817
Receiving objects: 100% (4817/4817), 153.35 MiB | 13.09 MiB/s, done.
Resolving deltas: 100% (2438/2438), done.
Checking connectivity... done.
(tensorflow) bash-3.2$
```

```
bash-3.2$ virtualenv --system-site-practices
Usage: virtualenv [OPTIONS] DEST_DIR

virtualenv: error: no such option: --system-site-practices
bash-3.2$ virtualenv --system-site-packages ~/tensorflow
New python executable in /Users/Work/tensorflow/bin/python
Installing setuptools, pip, wheel...done
bash-3.2$ source ~/tensorflow/bin/activate
(tensorflow) bash-3.2$ cd models/tutorials/rnn/translate
bash: cd: models/tutorials/rnn/translate: No such file or directory
(tensorflow) bash-3.2$ ls
Applications           Public          nltk_data
Desktop                PycharmProjects  setuptools
Documents               Users           software
Downloads              VirtualBox_VMs   tensorflow
Library                anaconda        tensorflow_chatbot
Movies                 data.csv       ~tensorflow
Music                  generative_chatbot
Pictures               keras
(tensorflow) bash-3.2$ cd ~tensorflow
(tensorflow) bash-3.2$ ls
bin                   lib             pip-selfcheck.json
include               models          tensorflow
(tensorflow) bash-3.2$ mkdir data
(tensorflow) bash-3.2$ ls
bin                   lib             tensorflow
data                  models          tensorflow
include               pip-selfcheck.json
(tensorflow) bash-3.2$ pwd
/Users/Work/~/tensorflow
(tensorflow) bash-3.2$ cd models/tutorials/rnn/translate
(tensorflow) bash-3.2$ python translate.py --data_dir data
Preparing WMT data in data
Creating directory data
Downloading http://www.statmt.org/wmt10/training-giga-fren.tar to data/training-giga-fren.tar
```

A first example – translation and idea to change to a ‘game’ of movie line quizzes!

EXAMPLE I: BECOMING FAMILIAR WITH TENSORFLOW SEQ-2-SEQ MODEL WITH AN ENGLISH-FRENCH TRANSLATION



Jennifer Davis, Data Bot Box

This notebook will explain how to create a conversational agent with the architecture of a generative dialog. Our database is the Cornell Movie Dialog Database, and our neural network will be a recursive neural net, using Tensorflow's Seq-to-Seq, a Long-term-short-term Memory Stack (LSTM).

```
In [*]: import tensorflow as tf
import keras
import numpy as np
import six
import scipy as sc
import sklearn as sk
import pandas as pd
import flask
import kivy
```

To first become familiar with TensorFlow and Seq-to-Seq, we will do a walk-through of TensorFlow's tutorial on translation from English to French. This methodology will be similar to that used in our conversational agent. We will start to see some of the quirks with conversational agents and think of ways to address them.

In the shell script, using bash, we downloaded and ran a script which prepared the translation data for us. It takes about 20 minutes on a good speed internet connection to download the data. For the overall information on this TensorFlow tutorial see:

https://www.tensorflow.org/tutorials/seq2seq#lets_run_it

It can take up to an hour to run this script, download and process teh data. So meanwhile we will start to work on the Kivy front end of our application. Kivy is a very nice multi-modal python framework for making applications on mobile, tablet, 'air' screen, and touch-screen. It is open-source and relatively easy to use!

This takes >3 hours
on a laptop using
CPUs

jupyter

Logout

```
tokenizing line 4400000
tokenizing line 4500000
tokenizing line 4600000
tokenizing line 4700000
tokenizing line 4800000
tokenizing line 4900000
tokenizing line 5000000
tokenizing line 5100000
tokenizing line 5200000
tokenizing line 5300000
tokenizing line 5400000
tokenizing line 5500000
tokenizing line 5600000
tokenizing line 5700000
tokenizing line 5800000
tokenizing line 5900000
tokenizing line 6000000
tokenizing line 6100000
tokenizing line 6200000
tokenizing line 6300000
tokenizing line 6400000
tokenizing line 6500000
tokenizing line 6600000
tokenizing line 6700000
tokenizing line 6800000
tokenizing line 6900000
tokenizing line 7000000
tokenizing line 7100000
tokenizing line 7200000
tokenizing line 7300000
tokenizing line 7400000
tokenizing line 7500000
tokenizing line 7600000
tokenizing line 7700000
tokenizing line 7800000
tokenizing line 7900000
```

Running and Bucketing the translation

```
global step 200 learning rate 0.5000 step-time 1.39 perplexity 1720.62
eval: bucket 0 perplexity 184.97
eval: bucket 1 perplexity 248.81
eval: bucket 2 perplexity 341.64
eval: bucket 3 perplexity 469.04
global step 400 learning rate 0.5000 step-time 1.38 perplexity 379.89
eval: bucket 0 perplexity 151.32
eval: bucket 1 perplexity 190.36
eval: bucket 2 perplexity 227.46
eval: bucket 3 perplexity 238.66
```

Command to begin translation

```
python translate.py
--data_dir [your_data_directory] --train_dir [checkpoints_directory]
--size=256 --num_layers=2 --steps_per_checkpoint=50
```

Output of the translation

```
python translate.py --decode
--data_dir [your_data_directory] --train_dir [checkpoints_directory]

Reading model parameters from /tmp/translate.ckpt-340000
> Who is the president of the United States?
Qui est le président des États-Unis ?
```



Galvanize Workshop on Artificially Intelligent Driven Chatbots 2017 By Jennifer Davis, Data Bot Box

This notebook will explain how to create a conversational agent with the architecture of a generative dialog. Our database is the Cornell Movie Dialog Database, and our neural network will be a recursive neural net, using Tensorflow's Seq-to-Seq, a Long-term-short-term Memory Stack (LSTM).

```
In [7]: import tensorflow as tf
import keras
import numpy as np
import six
import scipy as sc
import sklearn as sk
import pandas as pd
```

```
In [16]: conversations_data = pd.read_csv('/Users/Work/generative_chatbot/cornell_movie-dialogs_corpus/mc
```

To make a chatbot based on movie dialog one would download the Cornell Movie Dialog Corpus (as demonstrated here) and follow a similar technique.

Instead of question and answer translation, ask the chat a line from a movie and get a response! It's a fun party-game with the correct user interface! Stay tuned to our repository for examples and updates.

Jupyter

Logout

```
bash-3.2$ ls
bad_words.txt                               cornell_movie_dialogs_corpus.zip
clean.py                                     profanity_filter.py
bash-3.2$ unzip cornell_movie_dialogs_corpus.zip
Archive: cornell_movie_dialogs_corpus.zip
      creating: cornell_movie-dialogs_corpus/
      inflating: cornell_movie-dialogs_corpus/.DS_Store
      creating: __MACOSX/
      creating: __MACOSX/cornell_movie-dialogs_corpus/
      inflating: __MACOSX/cornell_movie-dialogs_corpus/.._.DS_Store
      inflating: cornell_movie-dialogs_corpus/chameleons.pdf
      inflating: __MACOSX/cornell_movie-dialogs_corpus/..chameleons.pdf
      inflating: cornell_movie-dialogs_corpus/movie_characters_metadata.txt
      inflating: cornell_movie-dialogs_corpus/movie_conversations.txt
      inflating: cornell_movie-dialogs_corpus/movie_lines.txt
      inflating: cornell_movie-dialogs_corpus/movie_titles_metadata.txt
      inflating: cornell_movie-dialogs_corpus/raw_script_urls.txt
      inflating: cornell_movie-dialogs_corpus/README.txt
      inflating: __MACOSX/cornell_movie-dialogs_corpus/.._README.txt
bash-3.2$ ls
__MACOSX                                cornell_movie-dialogs_corpus
bad_words.txt                           cornell_movie_dialogs_corpus.zip
clean.py                                 profanity_filter.py
bash-3.2$ █
```

cristian Danescu-Niculescu-Mizil and Lillian Lee
Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011.

(this paper is included in this zip file)

NOTE: If you have results to report on these corpora, please send email to cristian@cs.cornell.edu or llee@cs.cornell.edu so we can add you to our list of people using this data. Thanks!

Contents of this README:

- A) Brief description
- B) Files description
- C) Details on the collection procedure
- D) Contact

A) Brief description:

This corpus contains a metadata-rich collection of fictional conversations extracted from raw movie scripts:

- 220,579 conversational exchanges between 10,292 pairs of movie characters
- involves 9,035 characters from 617 movies
- in total 304,713 utterances
- movie metadata included:
 - genres
 - ie with the same title was found in IMBD, the match was made with the most popular title (the one that received most IMDB votes)

After discarding all movies that could not be matched or that had less than 5 IMDB votes, we were left with 617 unique titles with metadata including genre, release year, IMDB rating and no. of IMDB votes and cast distribution. We then identified the pairs of characters that interact and separated their conversations automatically

bash-3.2\$

Seq2Seq Chatbots Based on Google's Neural Conversational Model

.CL] 22 Jul 2015

A Neural Conversational Model

Oriol Vinyals
Google

Quoc V. Le
Google

VINYALS@GOOGLE.COM

QVL@GOOGLE.COM

Abstract

Conversational modeling is an important task in natural language understanding and machine intelligence. Although previous approaches exist, they are often restricted to specific domains (e.g., booking an airline ticket) and require hand-crafted rules. In this paper, we present a simple approach for this task which uses the recently proposed sequence to sequence framework. Our

than just mere classification, they can be used to map complicated structures to other complicated structures. An example of this is the task of mapping a sequence to another sequence which has direct applications in natural language understanding (Sutskever et al., 2014). The main advantage of this framework is that it requires little feature engineering and domain specificity whilst matching or surpassing state-of-the-art results. This advance, in our opinion, allows researchers to work on tasks for which domain knowledge may not be readily available or for tasks which

Example of Seq2Seq Chatbot on the Web



DATA BOT BOX

Data Bot Box, LLC Copyright 2017

bitfusion

What would twitter say if it could talk to itself?

EXAMPLE II: A CHATBOT MADE FROM TWITTER DATA

The screenshot shows a web browser window with the URL <https://suriyadeepan.github.io/2016-12-31-practical-seq2seq/>. The page title is "Suriyadeepan Ram". The main content area contains a block of terminal commands:

```
# clone the repository
git clone https://github.com/suriyadeepan/practical_seq2seq
# pull the pretrained model

cd practical_seq2seq/ckpt/twitter/
./pull # extract the archive

# this will take some time ~830 MB

cd ../../
# then you need the datasets to test the model

cd datasets/twitter/
./pull # this wont take long

# extract the archive

cd ../../
# instead you could just test with your own queries

# in that case you could just skip the step above

# now.. open up jupyter notebook and run "03-Twitter-chatbot.ipynb"

# the (pieces of) code is fairly intuitive

# figure out how to use it
```

First we will clone Suriyadeepan's Practical Seq2Seq repository. He has very clear instructions on how to run his code.

For fun, try running his twitter bot on Reddit data. We'll post that after today's workshop so stay tuned to the Github repo!!

```
bash-3.2$ git clone https://github.com/suriyadeepan/practical_seq2seq
Cloning into 'practical_seq2seq'...
remote: Counting objects: 167, done.
remote: Total 167 (delta 0), reused 0 (delta 0), pack-reused 167
Receiving objects: 100% (167/167), 7.13 MiB | 3.91 MiB/s, done.
Resolving deltas: 100% (73/73), done.
Checking connectivity... done.
bash-3.2$ cd practical_seq2seq/ckpt/twitter/
bash-3.2$ ./pull
--2017-05-26 16:41:06-- https://www.dropbox.com/s/vwu9if99rvv4dab/seq2seq_twitter_1024x3h_i43000.tar.gz?dl=0
Resolving www.dropbox.com... 162.125.3.1
Connecting to www.dropbox.com|162.125.3.1|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.dropboxusercontent.com/content_link/AvIGvJJQQcMHigClzkoU53Fmrt0J572Ts2ZmWDwC8OajR6okwsdWxuwyXr97FDB/file [following]
--2017-05-26 16:41:07-- https://dl.dropboxusercontent.com/content_link/AvIGvJJQQcMHigClzkoU53Fmrt0J572Ts2ZmWDwC8OajR6okwsdWxuwyXr97FDB/file
Resolving dl.dropboxusercontent.com... 162.125.49.131
Connecting to dl.dropboxusercontent.com|162.125.49.131|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 830428500 (792M) [application/octet-stream]
Saving to: 'seq2seq_twitter_1024x3h_i43000.tar.gz'

seq2seq_twitter_1024x 100%[=====] 791.96M 12.0MB/s in 1m 46s
2017-05-26 16:42:54 (7.50 MB/s) - 'seq2seq_twitter_1024x3h_i43000.tar.gz' saved [830428500/830428500]
bash-3.2$ █
```

Okay, now you have downloaded all the data, processed it, and run your twitter bot! Let's go to the graphical user interface in our Jupyter notebook to see what kinds of responses we achieved!

For fun, try running his twitter bot on Reddit data. We'll post that after today's workshop so stay tuned to the Github repo!!

```
bash-3.2$ git clone https://github.com/suriyadeepan/practical_seq2seq
Cloning into 'practical_seq2seq'...
remote: Counting objects: 167, done.
remote: Total 167 (delta 0), reused 0 (delta 0), pack-reused 167
Receiving objects: 100% (167/167), 7.13 MiB | 3.91 MiB/s, done.
Resolving deltas: 100% (73/73), done.
Checking connectivity... done.
bash-3.2$ cd practical_seq2seq/ckpt/twitter/
bash-3.2$ ./pull
--2017-05-26 16:41:06-- https://www.dropbox.com/s/vwu9if99rvv4dab/seq2seq_twitter_1024x3h_i43000.tar.gz?dl=0
Resolving www.dropbox.com... 162.125.3.1
Connecting to www.dropbox.com|162.125.3.1|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.dropboxusercontent.com/content_link/AvIGvJJQQcMHyigClzkoU53Fmrt0J572Ts2ZmWDwC8OajR6okwsdWxuwyXr97FDB/file [following]
--2017-05-26 16:41:07-- https://dl.dropboxusercontent.com/content_link/AvIGvJJQQcMHyigClzkoU53Fmrt0J572Ts2ZmWDwC8OajR6okwsdWxuwyXr97FDB/file
Resolving dl.dropboxusercontent.com... 162.125.49.131
Connecting to dl.dropboxusercontent.com|162.125.49.131|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 830428500 (792M) [application/octet-stream]
Saving to: 'seq2seq_twitter_1024x3h_i43000.tar.gz'

seq2seq_twitter_1024x 100%[=====] 791.96M 12.0MB/s in 1m 46s
2017-05-26 16:42:54 (7.50 MB/s) - 'seq2seq_twitter_1024x3h_i43000.tar.gz' saved [830428500/830428500]

bash-3.2$ █
```

jupyter 03-Twitter-chatbot (autosaved) Logout

File Edit View Insert Cell Kernel Help Python 2 O CellToolbar

Demonstrate Seq2Seq Wrapper with twitter chat log

```
In [12]: import tensorflow as tf
import numpy as np
import tensorflow as tf
import keras
import numpy as np
import six
import scipy as sc
import sklearn as sk
import pandas as pd

# preprocessed data
#import datasets
#from keras.datasets.twitter import data
import data_utils
```

```
In [15]: import tensorflow as tf
import numpy as np

# preprocessed data
#import datasets
#from datasets.twitter import data
import data_utils
```

```
In [16]: # load data from pickle and npy files
metadata, idx_q, idx_a = data.load_data(PATH='datasets/twitter/')
(trainX, trainY), (testX, testY), (validX, validY) = data_utils.split_data
```

```
In [4]: import seq2seq_wrapper
```

```
In [6]: import importlib  
importlib.reload(seq2seq_wrapper)
```

```
Out[6]: <module 'seq2seq_wrapper' from '/home/suriya/_/tf/tf-seq2seq-wrapper/seq2  
seq_wrapper.py'>
```

```
In [5]: model = seq2seq_wrapper.Seq2Seq(xseq_len=xseq_len,  
                                         yseq_len=yseq_len,  
                                         xvocab_size=xvocab_size,  
                                         yvocab_size=yvocab_size,  
                                         ckpt_path='ckpt/twitter/',  
                                         emb_dim=emb_dim,  
                                         num_layers=3  
                                         )
```

```
<log> Building Graph </log>
```

jupyter 03-Twitter-chatbot (autosaved) Logout

File Edit View Insert Cell Kernel Help Python 2

<log> Building Graph </log>

```
In [6]: val_batch_gen = data_utils.rand_batch_gen(validX, validY, 256)
test_batch_gen = data_utils.rand_batch_gen(testX, testY, 256)
train_batch_gen = data_utils.rand_batch_gen(trainX, trainY, batch_size)
```

```
In [9]: sess = model.train(train_batch_gen, val_batch_gen)

<log> Training started </log>

Model saved to disk at iteration #500
val loss : 3.370399
Interrupted by user at iteration 565
```

```
In [7]: sess = model.restore_last_session()
```

```
In [10]: input_ = test_batch_gen.__next__()[0]
output = model.predict(sess, input_)
print(output.shape)

(256, 20)
```

```
In [11]: replies = []
for ii, oi in zip(input_.T, output):
    q = data_utils.decode(sequence=ii, lookup=metadata['idx2w'], separator=' ')
    decoded = data_utils.decode(sequence=oi, lookup=metadata['idx2w'], separator=' ')
    if decoded.count('unk') == 0:
        if decoded not in replies:
            print('q : [{0}]; a : [{1}]'.format(q, ' '.join(decoded)))
            replies.append(decoded)
```

The screenshot shows a Jupyter Notebook interface running on a local host. The title bar indicates the notebook is titled "03-Twitter-chatbot.ipynb". The menu bar includes File, Edit, View, Insert, Cell, Kernel, and Help. The toolbar includes icons for file operations like Open, Save, and Run, along with Code and CellToolbar buttons. The main area contains a code cell with Python code and its output.

```
File Edit View Insert Cell Kernel Help
Code CellToolbar
In [ ]:
```

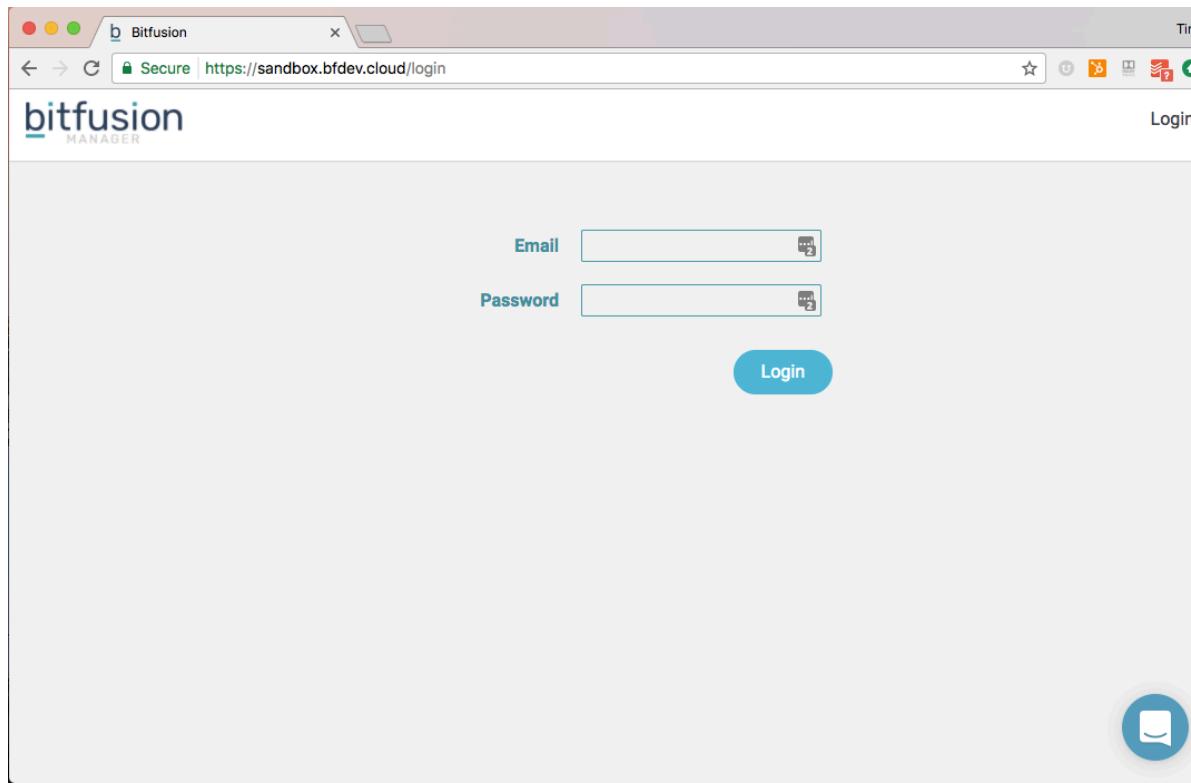
```
q = data_utils.decode(sequence=ii, lookup=metadata['idx2w'], separator=' ')
decoded = data_utils.decode(sequence=oi, lookup=metadata['idx2w'], separator=' ')
if decoded.count('unk') == 0:
    if decoded not in replies:
        print('q : [{0}]; a : [{1}]'.format(q, ' '.join(decoded)))
        replies.append(decoded)

q : [go see them]; a : [i was thinking of that]
q : [a unk of the before we say goodnight from last at unk state park i
n maine]; a : [this is a great idea to be a great day for a while]
q : [unless you want this ]; a : [what is he]
q : [my fav moment from the debate last night]; a : [wait for the first
time]
q : [ok i dont see any catholic terrorist]; a : [not even to be a threa
t to the point]
q : [thats your sister]; a : [i dont know what i was talking about]
q : [radical islam unk a threat to the united states and our leaders ne
ed to realize this before its too late]; a : [is that a joke]
q : [literally the same thing happened to me ]; a : [thats what i said]
q : [last chance to win vip passes to new york comic con amp meet the w
alking dead cast go]; a : [if you were in the same place i dont have to
see it]
q : [4 correction that video is from a few years ago]; a : [its been a
fan of my life but i dont know what to do]
q : [we are doing band there are many unk in the band]; a : [i have to
be a good job]
```

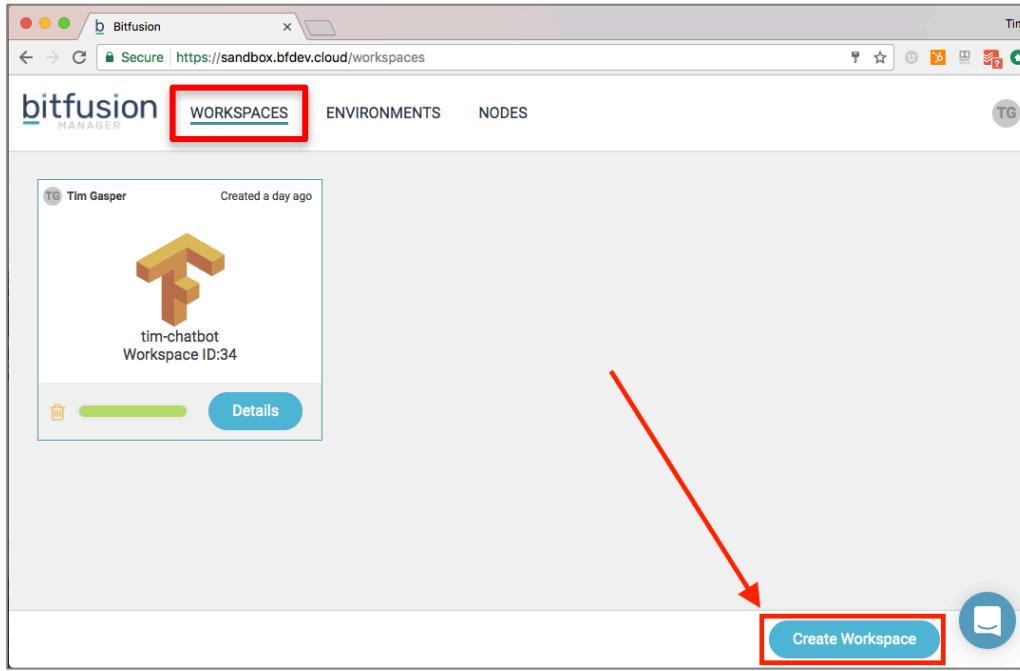
Why use GPU/CPU Platforms?

EXAMPLE III: THE BENEFITS OF USING A GPU PLATFORM LIKE BITFUSION TO TRAIN AN AI-DRIVEN CHATBOT

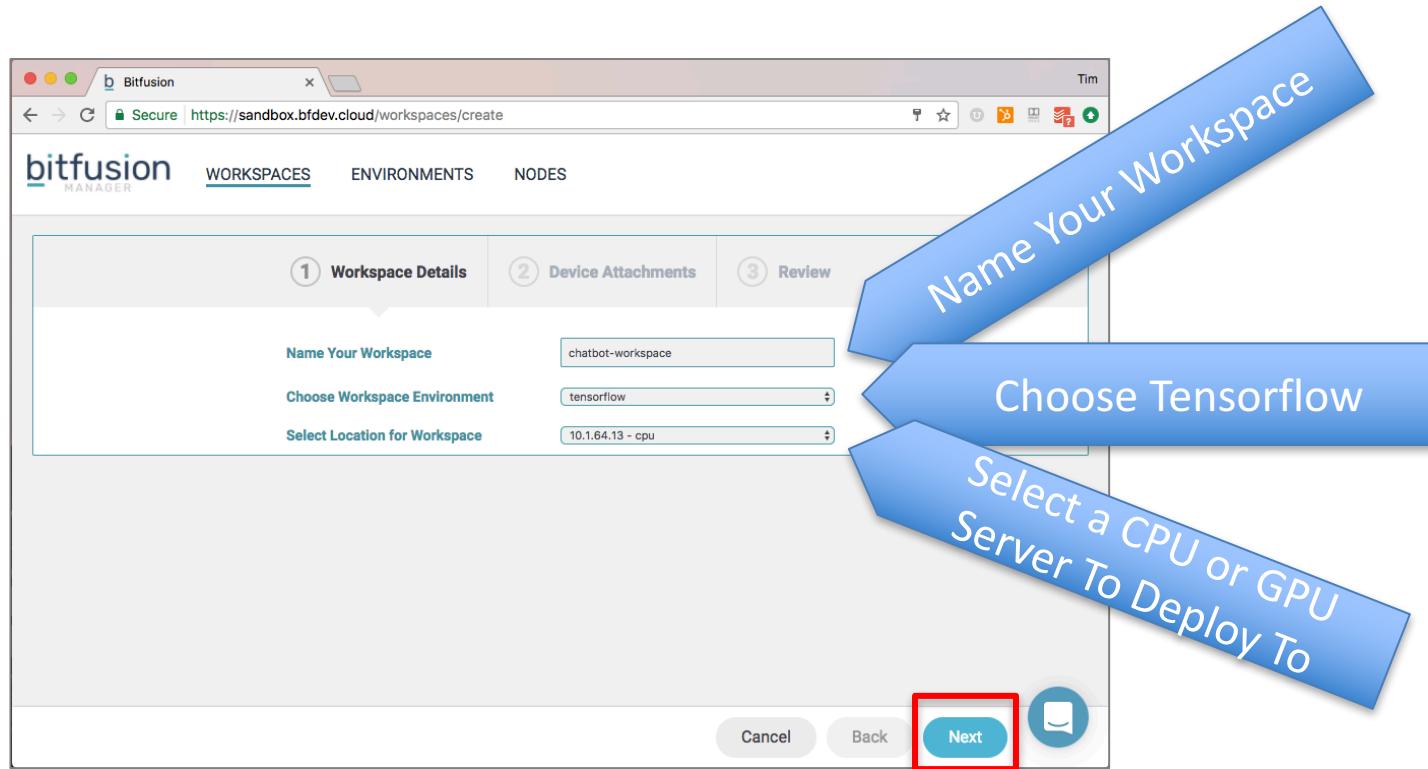
Step I: Log-in to the platform



Step 2A: Create a Workspace



Step 2B: Create Workspace Details



Step 2C: Finalize Workspace Options

The screenshot shows the Bitfusion Manager interface for creating a new workspace. The top navigation bar includes 'bitfusion' and tabs for 'WORKSPACES', 'ENVIRONMENTS', and 'NODES'. The 'WORKSPACES' tab is selected, showing a progress bar at step 2.

The main area displays 'Workspace Details' and 'Device Attachments' sections. A red box highlights the 'Select Local Devices' and 'Select Remote Devices (optional)' fields, both currently showing '(none)'. Below these fields is a large blue callout box with white text:

If GPUs are available, you can attach them here.
Please only use one at a time, and only when actively using the GPU to keep GPUs in the available pool!

The 'Device Attachments' section shows a table with one row:

Name	chatbot-workspace
Environment	tensorflow
Location	10.1.64.13 CPU
Device Attachments	(none)

At the bottom of the workspace creation window, there are 'Cancel', 'Back', and 'Next' buttons. The 'Next' button is highlighted with a red box.

On the right side of the interface, there is a separate 'Review' window showing a summary of the workspace configuration. It includes 'Cancel', 'Back', and 'Create' buttons. The 'Create' button is highlighted with a red box.

Step 3: Access Jupyter

The screenshot shows the Bitfusion Manager interface with two workspaces listed:

- chatbot-workspace** (Workspace ID:36) - Created a few seconds ago. The "Details" button is highlighted with a red box.
- tim-chatbot** (Workspace ID:34) - Created a day ago. The "Details" button is also present.

A large blue arrow points from the bottom left towards the "Details" button of the first workspace, with the text "When Green, Hit 'Details'" overlaid.

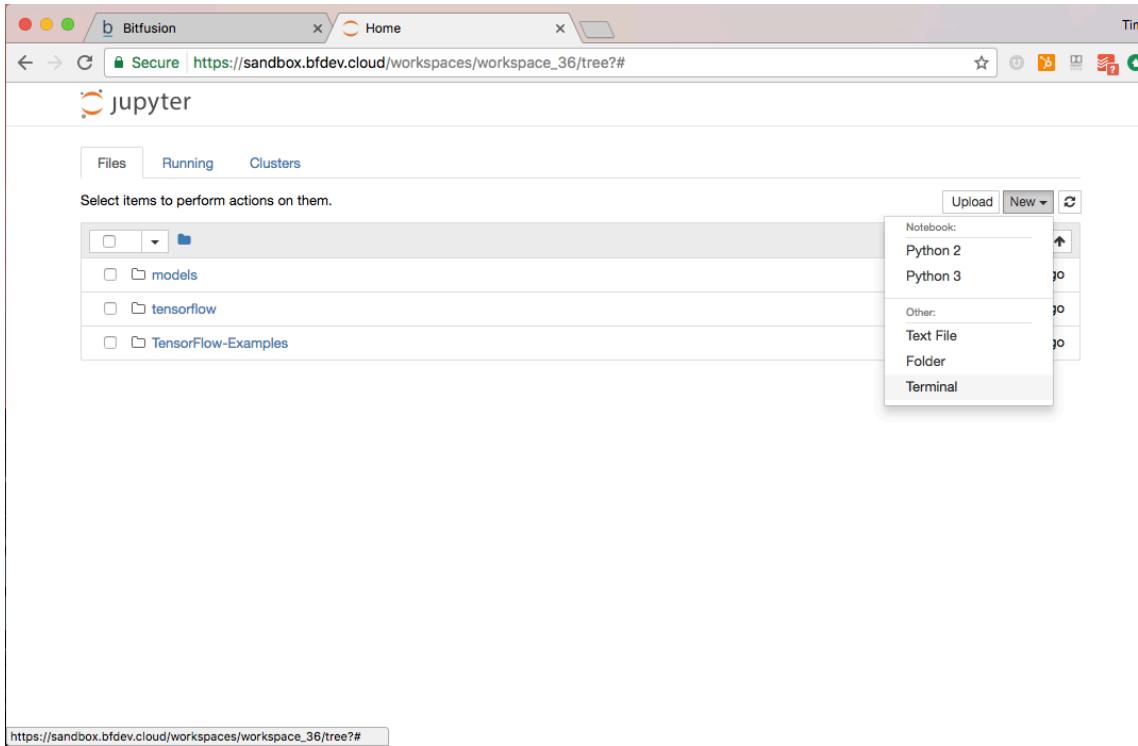
On the right side, a detailed view of the "chatbot-workspace" is shown:

- Volumes:**

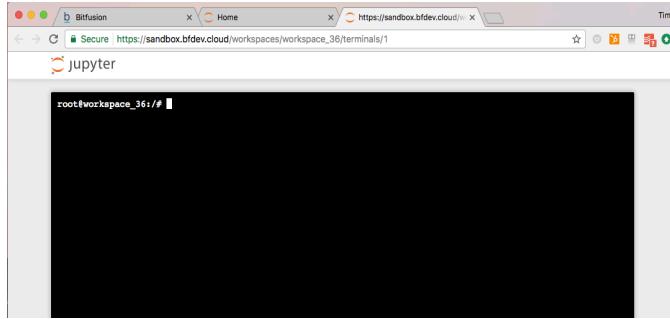
Name	Host Path	Container Path
share	/bitfusion/share	/var/bitfusion/share
- Applications:**
 - jupyter (with a "Go" button highlighted with a red box)

A second blue arrow points from the bottom right towards the "Go" button of the jupyter application, with the text "Go!" overlaid.

Step 4: Open Terminal



Step 5: Run the Following Commands

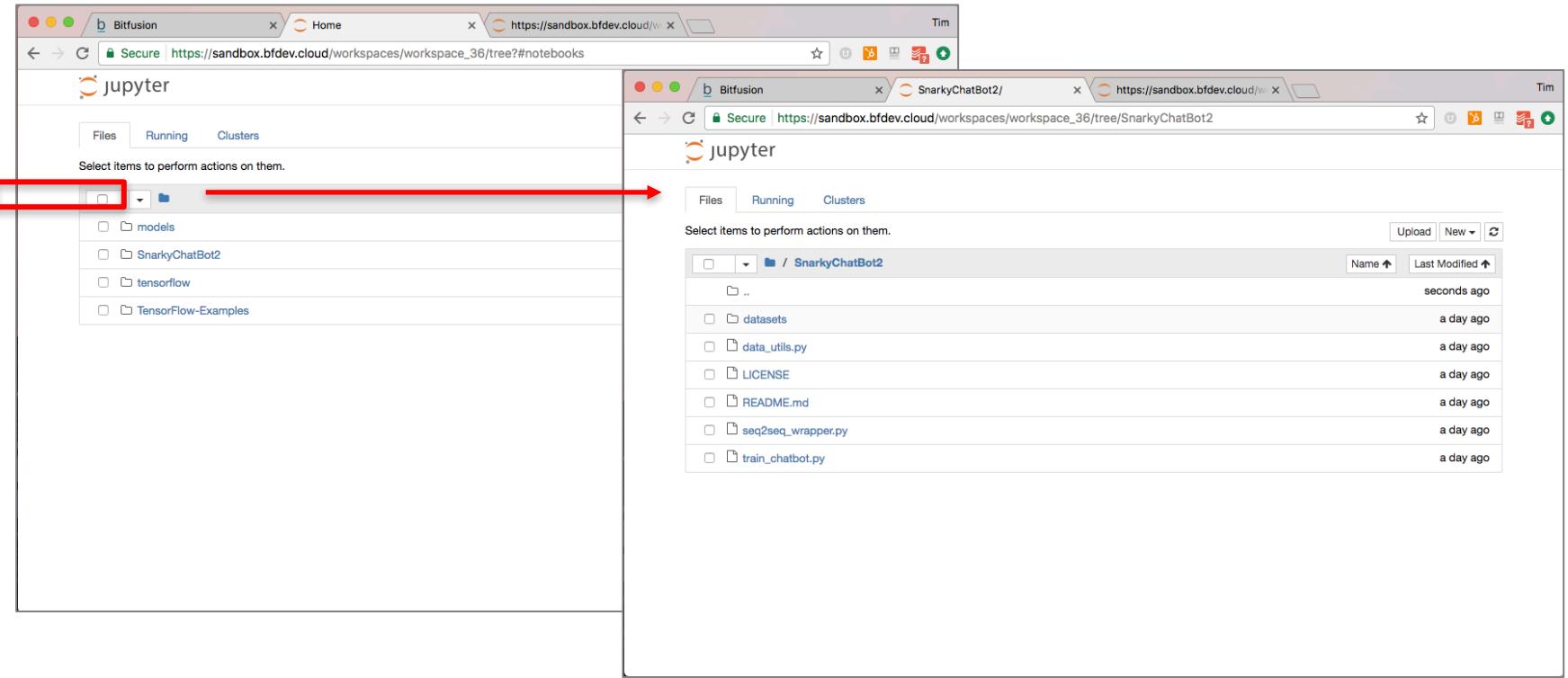


We have the data and code you need in the shared filesystem! Link it into Jupyter by running:

- `ln -s /var/bitfusion/share/SnarkyChatBot2/ /opt/pynb/SnarkyChatBot2`

(if you want to assume there is no shared filesystem and set up from scratch ping me tim@bitfusion.io and I'll send you an additional set of instructions)

Step I: Log-in to the platform





jupyter train_chatbot.py ✓ Yesterday at 10:00 AM

```
File Edit View Language Python
1 # preprocessed data
2 from datasets.may_reddit import data
3 from data_utils import split_dataset, rand_batch_gen
4 import seq2seq_wrapper
5
6 # load data from pickle and npy files
7 metadata, idx_q, idx_a = data.load_data(PATH='datasets/may_reddit/')
8 (trainX, trainY), (testX, testY), (validX, validY) = split_dataset(idx_q,
9                                     idx_a)
10
11 # parameters
12 xseq_len = trainX.shape[-1]
13 yseq_len = trainY.shape[-1]
14 batch_size = 32
15 xvocab_size = len(metadata['idx2w'])
16 yvocab_size = xvocab_size
17 emb_dim = 1024
18
19
20 # In[7]:
21
22 model = seq2seq_wrapper.Seq2Seq(xseq_len=xseq_len,
23                                 yseq_len=yseq_len,
24                                 xvocab_size=xvocab_size,
25                                 yvocab_size=yvocab_size,
26                                 ckpt_path='ckpt/reddit/',
27                                 emb_dim=emb_dim,
28                                 num_layers=3)
```

Step 6A: Chatbot Model Training via CLI

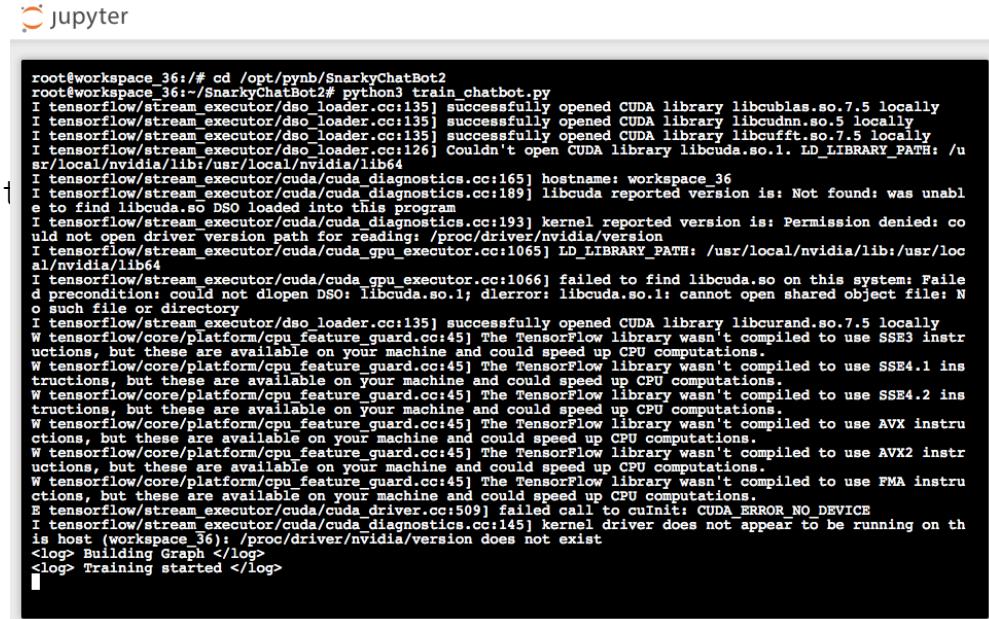
In the terminal window...

Navigate to the SnarkyChatBot2 folder

- `cd /opt/pynb/SnarkyChatBot2`

Run `train_chatbot.py`

- `python3 train_chatbot.py`

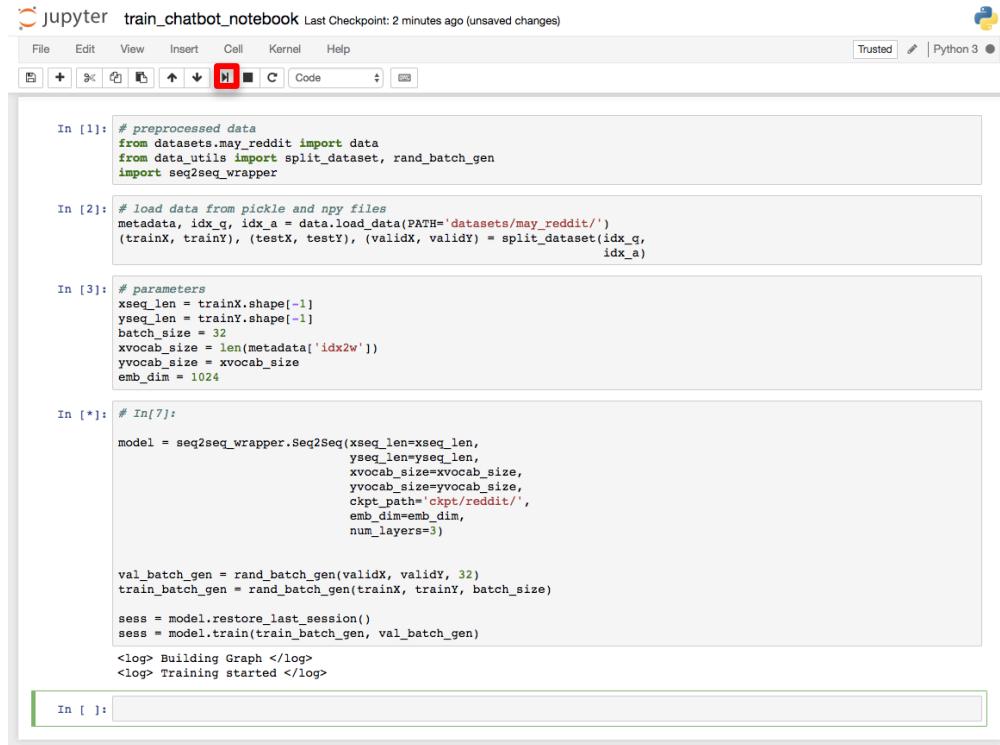


```
root@workspace_36:/# cd /opt/pynb/SnarkyChatBot2
root@workspace_36:/SnarkyChatBot2# python3 train_chatbot.py
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcublas.so.7.5 locally
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcudnn.so.5 locally
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcufft.so.7.5 locally
I tensorflow/stream_executor/dso_loader.cc:126] Couldn't open CUDA library libcuda.so.1. LD_LIBRARY_PATH: /usr/local/nvidia/lib:/usr/local/nvidia/lib64
I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:165] hostname: workspace_36
I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:189] libcuda reported version is: Not found: was unable to find libcuda.so DSO loaded into this program
I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:193] kernel reported version is: Permission denied: could not open driver version path for reading: /proc/driver/nvidia/version
I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1065] LD_LIBRARY_PATH: /usr/local/nvidia/lib:/usr/local/nvidia/lib64
I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:1066] failed to find libcuda.so on this system: Failed precondition: could not dlopen DSO: libcuda.so.1; dlerror: libcuda.so.1: cannot open shared object file: No such file or directory
I tensorflow/stream_executor/dso_loader.cc:135] successfully opened CUDA library libcurand.so.7.5 locally
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE3 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use AVX2 instructions, but these are available on your machine and could speed up CPU computations.
W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow library wasn't compiled to use FMA instructions, but these are available on your machine and could speed up CPU computations.
E tensorflow/stream_executor/cuda/cuda_driver.cc:509] failed call to cuInit: CUDA ERROR NO DEVICE
I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:145] kernel driver does not appear to be running on this host (workspace_36): /proc/driver/nVidia/version does not exist
<log> Building Graph </log>
<log> Training started </log>
```

Step 6B: Chatbot Model Training via Notebook

In Jupyter...

Click on:
`train_chatbot_notebook.ipynb`
and hit the play/pause button for
each of the code cells!



```
In [1]: # preprocessed data
from datasets.may_reddit import data
from data_utils import split_dataset, rand_batch_gen
import seq2seq_wrapper

In [2]: # load data from pickle and npy files
metadata, idx_g, idx_a = data.load_data(PATH='datasets/may_reddit/')
(trainX, trainY), (testX, testY), (validX, validY) = split_dataset(idx_g,
idx_a)

In [3]: # parameters
xseq_len = trainX.shape[-1]
yseq_len = trainY.shape[-1]
batch_size = 32
xvocab_size = len(metadata['idx2w'])
yvocab_size = xvocab_size
emb_dim = 1024

In [*]: # In[7]:
model = seq2seq_wrapper.Seq2Seq(xseq_len=xseq_len,
yseq_len=yseq_len,
xvocab_size=xvocab_size,
yvocab_size=yvocab_size,
ckpt_path='ckpt/reddit/',
emb_dim=emb_dim,
num_layers=3)

val_batch_gen = rand_batch_gen(validX, validY, 32)
train_batch_gen = rand_batch_gen(trainX, trainY, batch_size)

sess = model.restore_last_session()
sess = model.train(train_batch_gen, val_batch_gen)
<log> Building Graph </log>
<log> Training started </log>

In [ ]:
```

Bitfusion Platform

- Makes development of AI-driven software simple
- Train, tune and optimize models with low visibility and minimal optimization
- Save several months in development and years in production for artificial intelligence products

Benefits of Bitfusion Platform

- Abstracts the need to create a separate training environment (GPU) from the running environment (CPU) for the chatbot
- Removes the need to create a complex Hadoop-like software integration platform de novo



DEEP LEARNING AND AI ARE RADICALLY CHANGING OUR LIVES AND THE WAY WE DO BUSINESS

AI Revolution

ROBOTICS	Computer Vision & Speech, Drones, Droids
ENTERTAINMENT	Interactive Virtual & Mixed Reality
AUTOMOTIVE	Self-Driving Cars, Co-Pilot Advisor
FINANCE	Predictive Price Analysis, Dynamic Decision Support
PHARMA	Drug Discovery, Protein Simulation
HEALTHCARE	Predictive Diagnosis, Wearable Intelligence
ENERGY	Geo-Seismic Resource Discovery
EDUCATION	Adaptive Learning Courses
SALES	Adaptive Product Recommendations
SUPPLY CHAIN	Dynamic Routing Optimization
CUSTOMER SERVICE	Bots And Fully-Automated Service
MAINTENANCE	Dynamic Risk Mitigation And Yield Optimization

Deep Learning with the Do-It-Yourself Approach



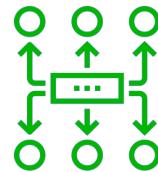
“The time scale and process for building AI
is more like developing hardware rather than agile software.”

INDUSTRY PROBLEM



Development Challenges

- Installing and Constantly Updating Software and Drivers
- Transitioning from Development to Training to Deployment
- Single-Threaded Model Training With Massive Wait Times



Management Challenges

- Expensive, Specialized Infrastructure Sitting Idle
- No Centralized Policy Management
- Poor Tools for Managing and Monitoring GPUs and other Co-Processors



Scale Challenges

- Massive Increases in Infrastructure Spending to Get Speed Ups
- Managing Shared Resources Across Multiple Workloads, Users, and Data Sets

bitfusion

Bitfusion Flex

End-to-end, elastic infrastructure for building deep learning and AI applications

- Can utilize RDMA and Infiniband when remote GPUs are attached
- Never have to transition beyond multi-GPU code environment (remote servers look like local GPUs and can be utilized without code changes)
- Rich CLI & GUI, interactive Jupyter workspaces, batch scheduling, smart shared resourcing for max efficiency, and much more

The Bitfusion Flex interface is a web-based management console. It features a top navigation bar with links for Projects, Workspaces, Runs, Data, Images, Nodes, Volumes, and Users. Below this, there are several sections:

- Nodes:** Shows a list of nodes with columns for Status (Running, Stopped, Failed, Success), User, Project Name, Version, Run ID, Resources (GPU count), Type, Started (time), and Duration.
- Workspaces:** Shows a list of workspaces with columns for Title, Workspace ID, and Details.
- Data:** Shows a list of data items with columns for Name, Type, and Details.
- Images:** Shows a list of images with columns for Name, Type, and Details.
- Users:** Shows a list of users with columns for Name, Email, and Details.

On the left side, there are two main sections:

- GPUs:** Displays a table of GPU resources with columns for IP Address, Memory, and Allocation. It shows four GPUs with 7.8 GB of memory each.
- CPUs:** Displays a table of CPU resources with columns for IP Address, Memory, Attached Workspaces, and Health. It shows four CPUs with 7.8 GB of memory each.

In the center, there is a large workspace area with tabs for Details and Logs. The Logs tab displays a terminal session with the following output:

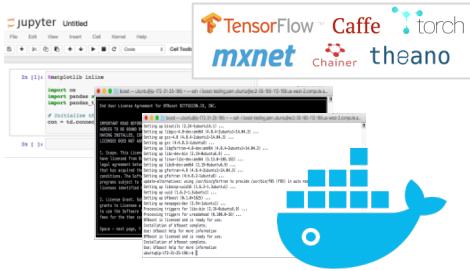
```
2017-04-05 08:36:58.096 INFO - Iter 998400, Minibatch Loss= 0.034193, Training Accuracy= 1.000000
2017-04-05 08:36:58.297 INFO - Iter 999600, Minibatch Loss= 0.074418, Training Accuracy= 0.97696
2017-04-05 08:36:58.353 INFO - Optimization Finished!
2017-04-05 08:36:58.354 INFO - Testing Accuracy= 0.996
2017-04-05 08:36:58.607 INFO -
2017-04-05 08:36:58.668 INFO - Waiting for container to complete...
2017-04-05 08:36:59.103 INFO - [Success] for Container finishing execution in 179 seconds for TaskInstance<TaskId:64b2e03a57072526a28b0db732ed9d3> (last update: 2017-04-05 08:36:59.09701) [success]
```

At the bottom right, there is a Jupyter notebook interface titled "convolutional_network.ipynb". The notebook contains Python code for training a neural network, including imports for numpy, tensorflow, and other libraries, and a loop for training and testing the model.

bitfusion

The Deep Learning and AI Lifecycle with Bitfusion Flex

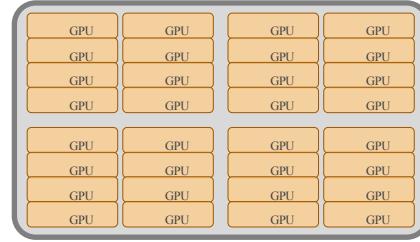
1 DEVELOP



Develop on pre-installed, quick start deep learning containers.

- Get to work quickly with workspaces with optimized pre-configured drivers, frameworks, libraries, and notebooks.
- Start with CPUs, and attach Elastic GPUs on-demand.
- All your code and data is saved automatically and sharable with others.

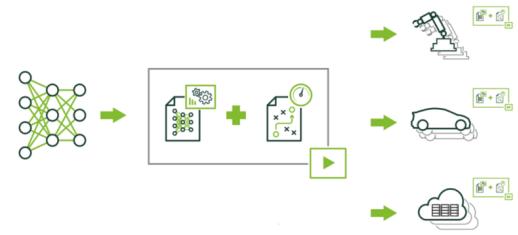
2 TRAIN



Transition from development to training with multiple GPUs.

- Seamlessly scale out to more GPUs on a shared training cluster to train larger models quickly and cost-effectively.
- Support and manage multiple users, teams, and projects.
- Train multiple models in parallel for massive productivity improvements.

3 DEPLOY



Push trained, finalized models into production.

- Deploy a trained neural network into production and perform real-time inference across different hardware.
- Manage multiple AI applications and inference endpoints corresponding to different trained models.

bitfusion

Value Proposition

Deep Learning with the Do-It-Yourself Approach



Deep Learning with Bitfusion



bitfusion

Getting Started

Enterprise Software Platform

- Deploy in your own cloud or data center
- Per node pricing
- Manager GUI and command line interface
- No-setup software, interactive workspaces, job runner w/ parallel training, and smart resourcing
- Advanced user, container, and resource management features
- Enterprise support & SLAs available

Sandbox Hosted

- Hosted and managed by Bitfusion
- Free for evaluation purposes
- Manager GUI and command line interface
- No-setup software, interactive workspaces, job runner w/ parallel training, and smart resourcing
- Basic support

bitfusion

How to Reach Bitfusion (for more of our Awesome Platform)

- How to reach Bitfusion:
 - Business: <https://bitfusion.io/>
tim@bitfusion.io
1-866-750-9656
@bitfusionio
 - Individual: @timgasper

How to Reach Data Bot Box, LLC

- Web: <https://www.databotbox.com>
 - Business: info@databotbox.com
 - Workshops: jennifer.d.davis@databotbox.com
 - Twitter: @databotbox

Thanks for your attention!

THE END

References

- Conversational Informatics: A Data Intensive Approach with Empahsis on Nonverbal Communication, Nishida et. al., Springer 2014
- Facebook messenger chatbot: Adweek, October 14, 2016: <http://www.adweek.com/digital/scott-horn-247-guest-post-chat-bot-technology-consumer-intent/>
- WFM chatbot: Venture Beat, July 12, 2016 : <https://venturebeat.com/2016/07/12/whole-foods-just-launched-a-messenger-chatbot-for-finding-recipes-with-emojis/>
- Heyday chatbot: Business Insider: October 19, 2016: <http://www.businessinsider.com/debate-trump-clinton-chat-bots-2016-10>
- Amazon Lex: <https://aws.amazon.com/lex/>
- The Chatbot Will See you Now: The New Yorker, December 25, 2016:
<http://www.newyorker.com/tech/elements/the-chatbot-will-see-you-now>
- Google's Deep Mind Expert, Blog WildMind: <http://www.wildml.com/2016/04/deep-learning-for-chatbots-part-1-introduction/>

Where to find the Code and Datasets

- <https://github.com/Data-Bot-Box>
- <https://github.com/niderhoff/nlp-datasets>
- <https://www.kaggle.com/reddit/reddit-comments-may-2015>
- Suriyadeepan.github.io/2016-06-28-easy-seq2seq/
- <https://www.tensorflow.org/tutorials/seq2seq>
- <https://deeplearning4j.org/lstm>
- <https://github.com/inikdom/neural-chatbot>

More on Deep Learning

- <https://deeplearning4j.org/lstm.html>
- Lowe et. Al. 'The Ubuntu Dialogue Corpus: A Large Dataset for Research in Unstructured Multi-Turn Dialogue Systems'
- Uthus, DC and Aha, DW (2013) The Ubuntu Chat Corpus for Multiparticipant Chat Analysis
- Download the Corpus: <http://daviduthus.org/UCC/>
- Other tensorflow chatbots:
- https://github.com/llSourcell/tensorflow_chatbot
- <https://github.com/Conchylicultor/DeepQA#web-interface>
- Facebook chatbot: <https://youtu.be/t5qgjJIBy9g>
- Google chatbot: <https://youtu.be/SJDEOWLHYVo>
- Cognition and memory models: <http://byuipt.net/564/2013/08/23/cognition-sensory-memory/>
- <http://neuralconvo.huggingface.co/>