Github Issue Title Generation

Text Summarization using GRU & LSTM

Ruan Dezheng, Tan Yong Jia Glenn, Han Simeng, Xu Yanbin

CE9010
Prof. Xavier Bresson
April 17th, 2019

GitHub

Github Collaboration Repository
https://github.com/ShirleyHan6/CE9010-Group-Project

CONTENT

01 DEZHENG

Data Problem
Data Acquisition

02 GLENN

Data Exploration
Pre-Processing

03 SIMENG

Data Analysis:
- Model Architecture

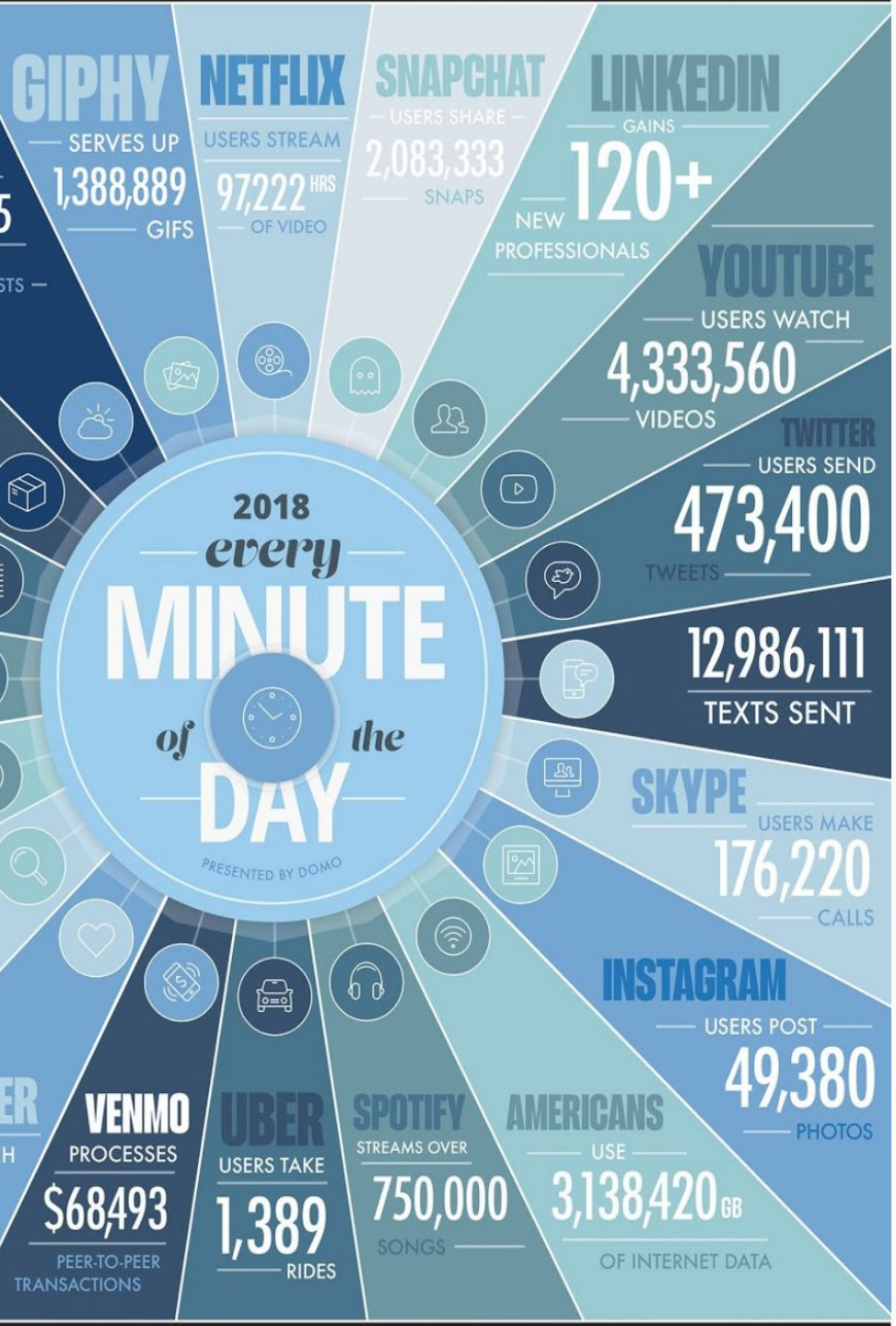
04 YANBIN

Data Analysis:
- Parameter tuning
- Cross validation

Analysis of Results

01

Introduction
Data Problem
Data Acquisition



Disturbing Problem

Information Boom

How to capture the main idea accurately and fast?

Efficient Communication?

Natural Language Processing

A Machine Learning Solution

Source: DOMO

Long Short Term Memory (LSTM)

An Natural Language Processing Algorithm

An improved type of recurrent neural networks

Mimic Human Recognition Process

General Architecture

a cell, an input gate, an output gate and a forget gate.

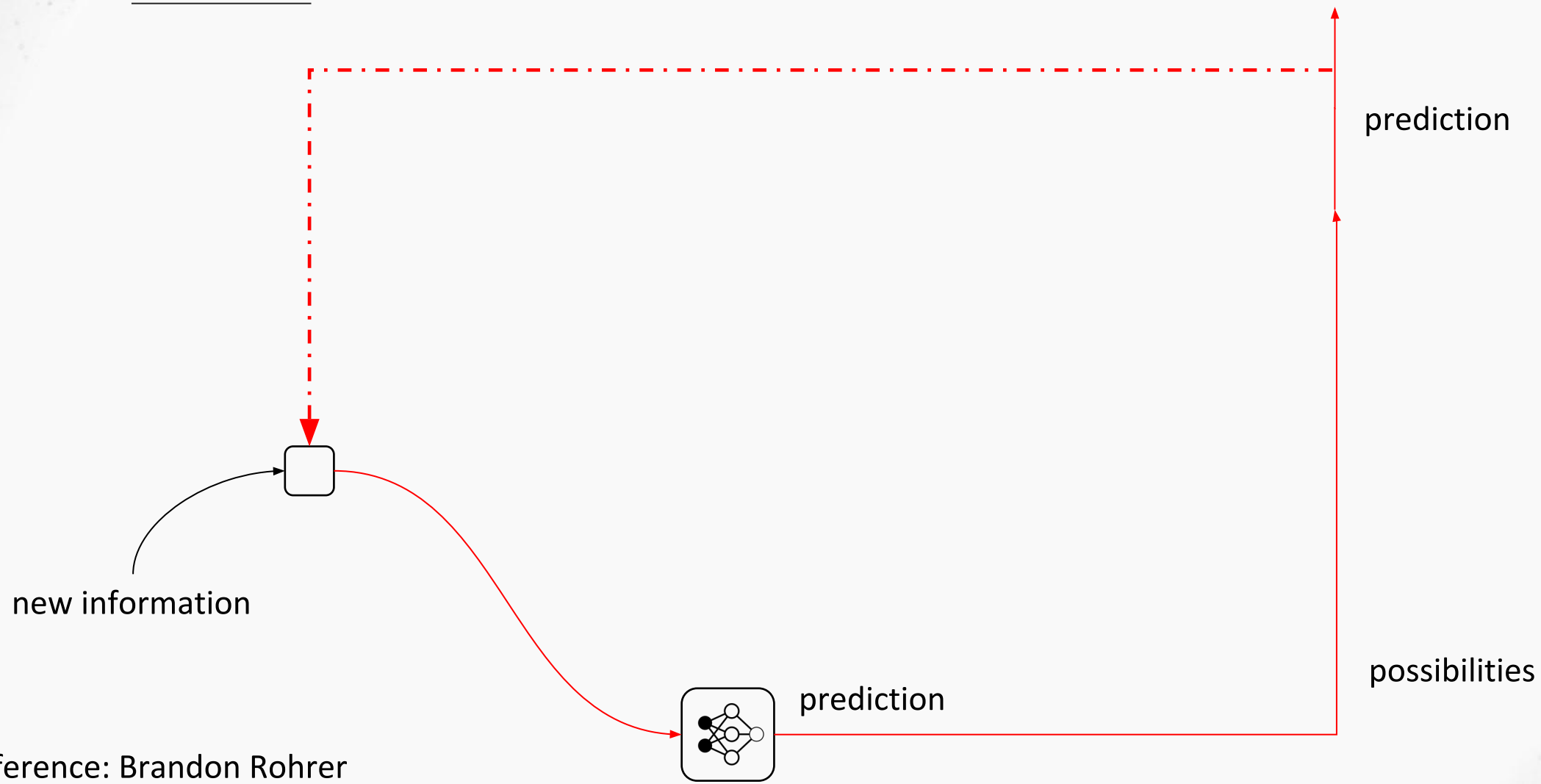
Long Short Term Memory (LSTM)

An Natural Language Processing Algorithm

Amazing! This box of cereals gave me a **perfectly balanced** breakfast, as all things should be. I only ate half of it yesterday but completely fall **in love** with it. Will definitely **buy again!**

Long Short Term Memory (LSTM)

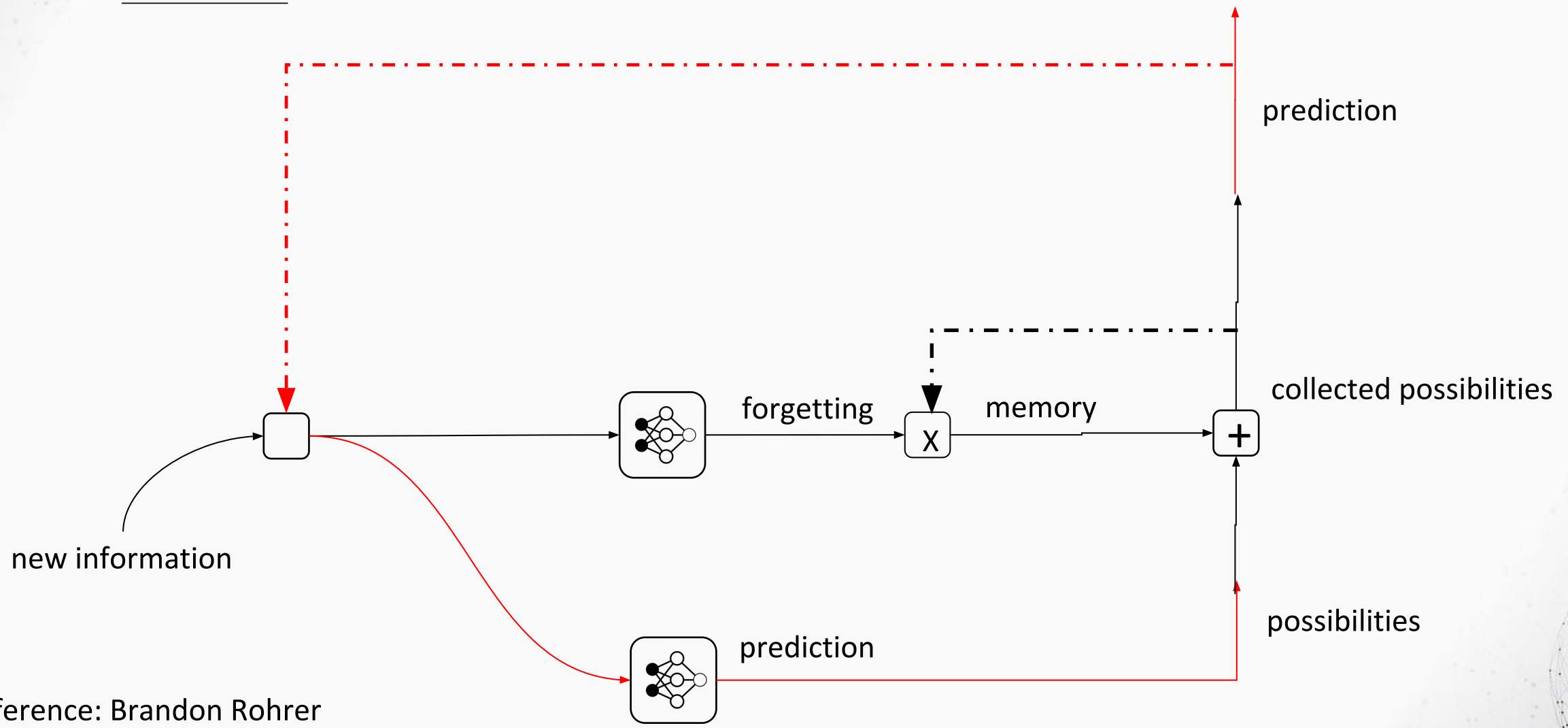
An Natural Language Processing Algorithm



Reference: Brandon Rohrer

Long Short Term Memory (LSTM)

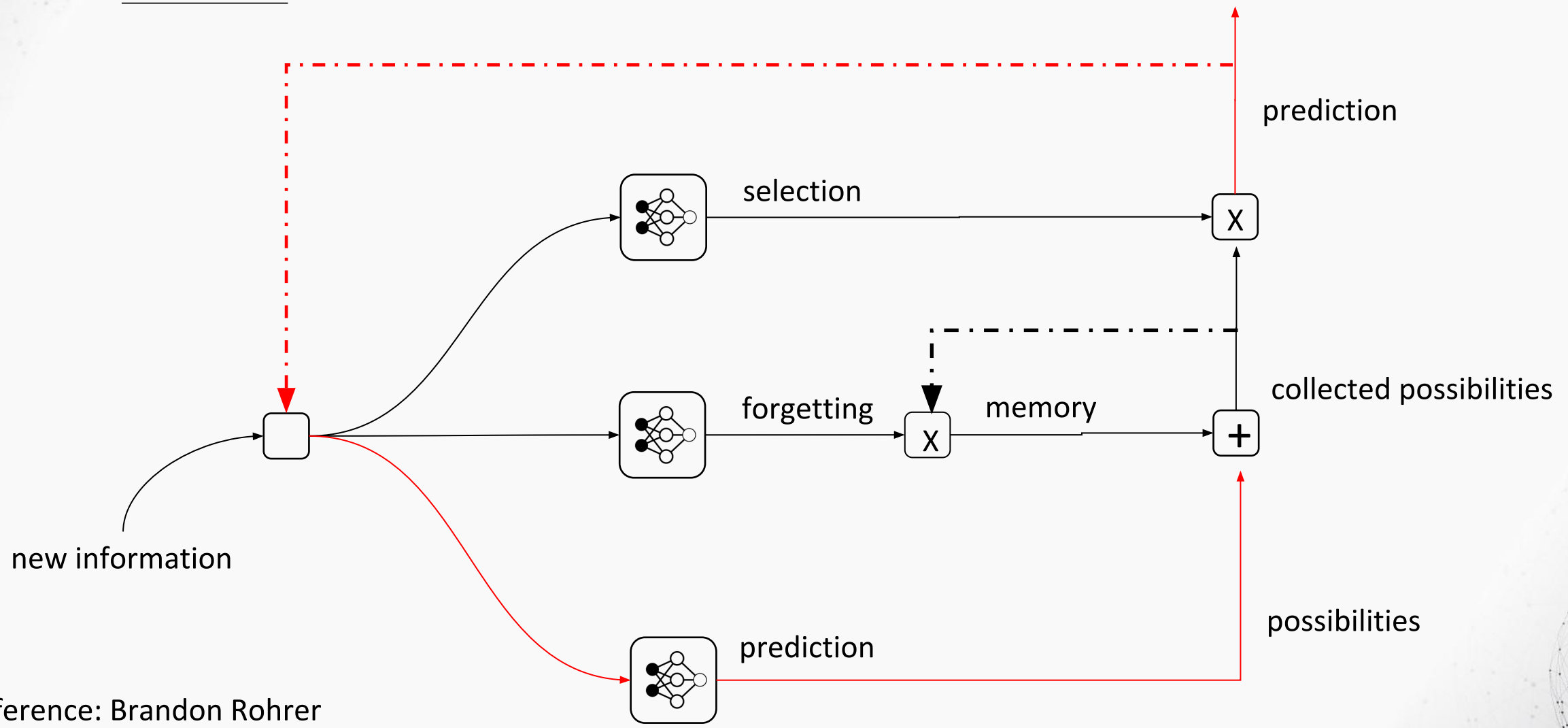
An Natural Language Processing Algorithm



Reference: Brandon Rohrer

Long Short Term Memory (LSTM)

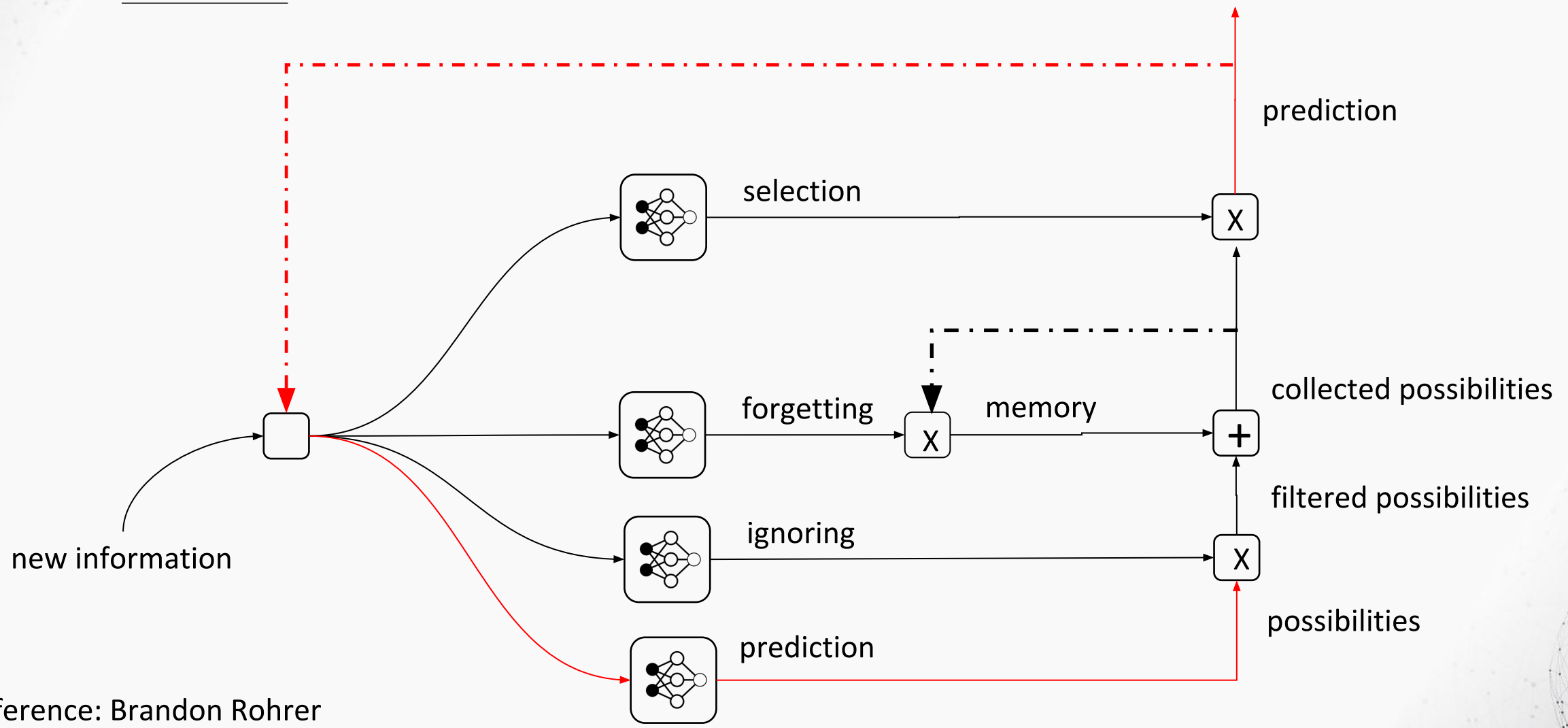
An Natural Language Processing Algorithm



Reference: Brandon Rohrer

Long Short Term Memory (LSTM)

An Natural Language Processing Algorithm



Reference: Brandon Rohrer

Data Acquisition

Dataset generated by ou

- Github Issue API
- Obtain data easily and efficie
- `requests`

GitHub Issues from kag

- 5 million GitHub issue titles an
- Reasonably cleaned and forma

```
def write_issues(response, csvout):
    "output a list of issues to csv"
    print ("  : Writing %s issues" % len(response.json()))
    for issue in response.json():
        labels = issue['labels']
        label_string = ''
        for label in labels:
            label_string = "%s, %s" % (label_string, label['name'])
        label_string = label_string[2:]

        csvout.writerow([issue['number'], issue['title'].encode('utf-8'), issue['body'].encode('ut
f-8'), label_string.encode('utf-8'), issue['created_at'], issue['updated_at']])


def get_issues(url):
    kwargs = {
        'headers': {
            'Content-Type': 'application/vnd.github.v3.raw+json',
            'User-Agent': 'GitHub issue exporter'
        },
        'params': params_payload
    }
    if GITHUB_TOKEN != '':
        kwargs['headers']['Authorization'] = 'token %s' % GITHUB_TOKEN
    else:
        kwargs['auth'] = (GITHUB_USER, GITHUB_PASSWORD)

    print ("GET %s" % url)
    resp = requests.get(url, **kwargs)
    print ("  : => %s" % resp.status_code)

    # import ipdb; ipdb.set_trace()
    if resp.status_code != 200:
        raise Exception(resp.status_code)

    return resp


def next_page(response):
    #more pages? examine the 'link' header returned
    if 'link' in response.headers:
        pages = dict(
            [(rel[6:-1], url[url.index('<')+1:-1]) for url, rel in
                [link.split(';') for link in
                    response.headers['link'].split(',')]])
        # import ipdb; ipdb.set_trace()
        if 'last' in pages and 'next' in pages:
```

02

Data Exploration
Data Pre-Processing

Data Set

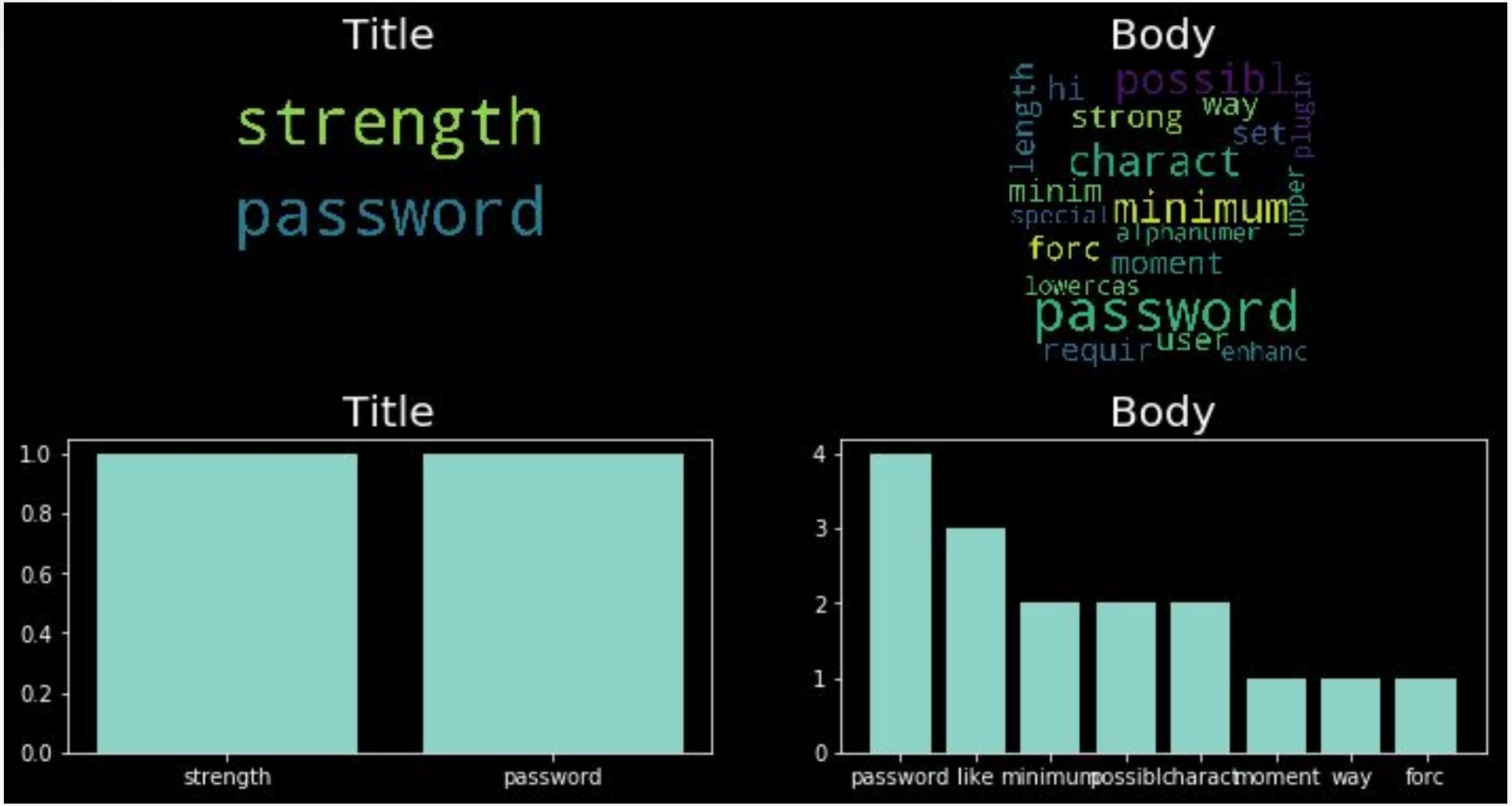
GitHub Issues

Data Set

	issue_url	issue_title	body
4967185	"https://github.com/kennethreitz/pipenv/issues/790"	install is failing in travis ci	i'm getting this error when running pipenv install in my travisci build step. $ pipenv install installing dependencies from pipfile.lock 4d9d13 … an error occurred while installing numpy==1.13.3! will try again. an error occurred while installing pandas==0.20.3! will try again. 🐍 ▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉▉ 10/10 — 00:00:05 installing initially–failed dependencies… collecting numpy==1.13.3 using cached numpy-1.13.3-cp36-cp36m-manylinux1_x86_64.whl these packages do not match the hashes ...
176760	"https://github.com/mher/flower/issues/667"	error when trying to view worker	i got this page when attempting to view a worker at this url: http://example.com:5555/worker/celery@raynor ! image https://cloud.githubusercontent.com/assets/2185159/22704708/7cd2e622-ed1d-11e6-9a65-626a0e4df3c8.png traceback: unknown celery version traceback most recent call last : file /home/azureuser/codalab-competitions/venv/local/lib/python2.7/site-packages/tornado/web.py , line 1415, in _execute result = yield result file /home/azureuser/codalab-competitions/venv/local/lib/python2.7/si...
4382977	"https://github.com/koorellasuresh/UKRegionTest/issues/58514"	first from flow in uk south	first from flow in uk south

Number of Data Rows: 5,332,153
Data Headers: issue_url ; issue_title ; body

Data Exploration : Word Cloud and Bar Chart

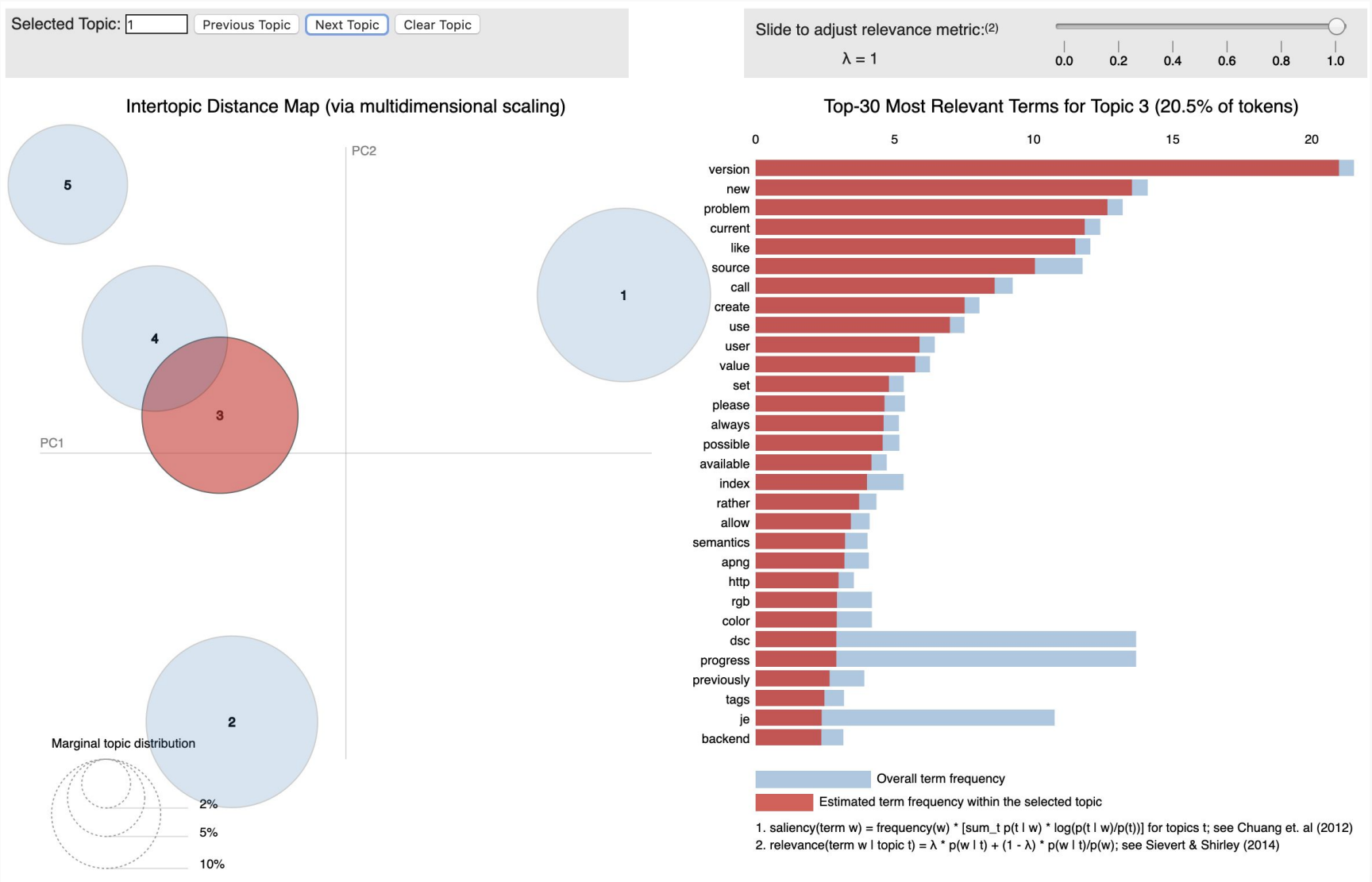


Title: a password strength

Body: hi. at this moment there's isn't way to force users to set strong passwords with minimal requirements like a minimum password length, minimum of alphanumeric characters upper and lowercase and special characters. is it possible for enhancing the password plugin like so? and with the possiblity to store old passwords like so you can create a

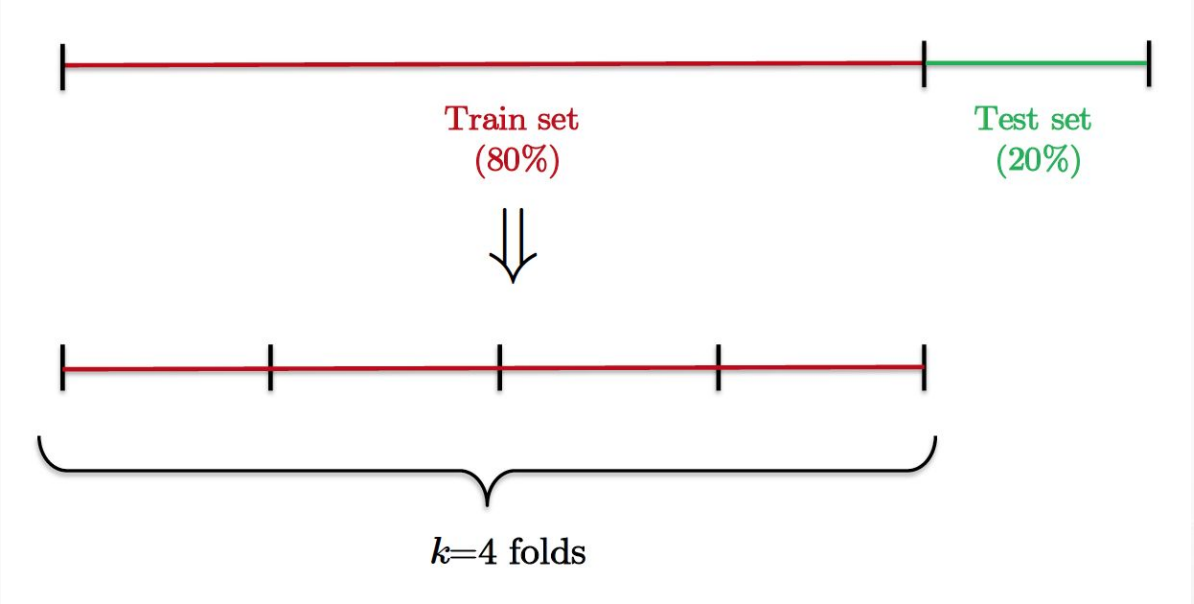
Data Exploration :

Dynamic Visualization using pyLDAvis



Data Pre-Processing

1. Train Test Split
2. Validation Set Split
3. Removing on Non-English Titles
4. Keras Text Pre-Processing



Data Pre-Processing : Keras Text Pre-Process

01 **Cleaning of text**

```
["The quick brown fox jumped over the lazy dog 42 times.",
"The dog is lazy"]
```

02 **Tokenization**

```
[['the', 'quick', 'brown', 'fox', 'jumped', 'over', 'the',
'lazy', 'dog', '*number*', 'times'], ['the', 'dog', 'is',
'lazy']]
```

03 **Build Vocabulary**

```
[[2, 3, 4, 5, 6, 7, 2, 8, 9, 10, 11],
 [2, 9, 12, 8]]
```

04 **Padding**

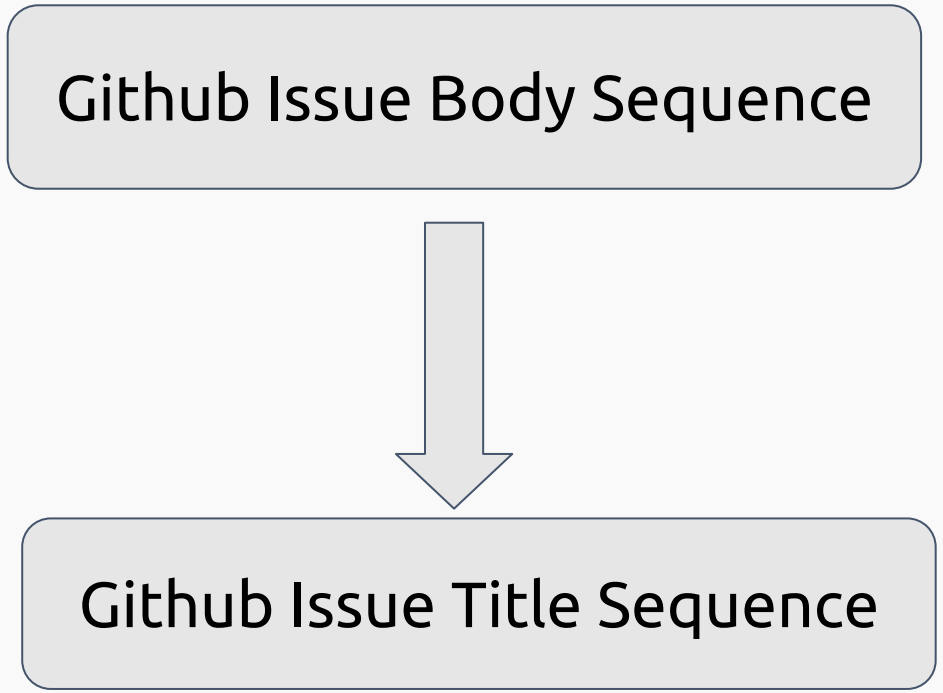
```
[[2, 3, 4, 5, 6, 7, 2, 8, 9, 10, 11],
 [0, 0, 0, 0, 0, 0, 0, 2, 9, 12, 8]]
```

03

Model Architecture

Sequence to Sequence Model

Sequence to Sequence Model with an Encoder-Decoder Architecture



- **Encoder**

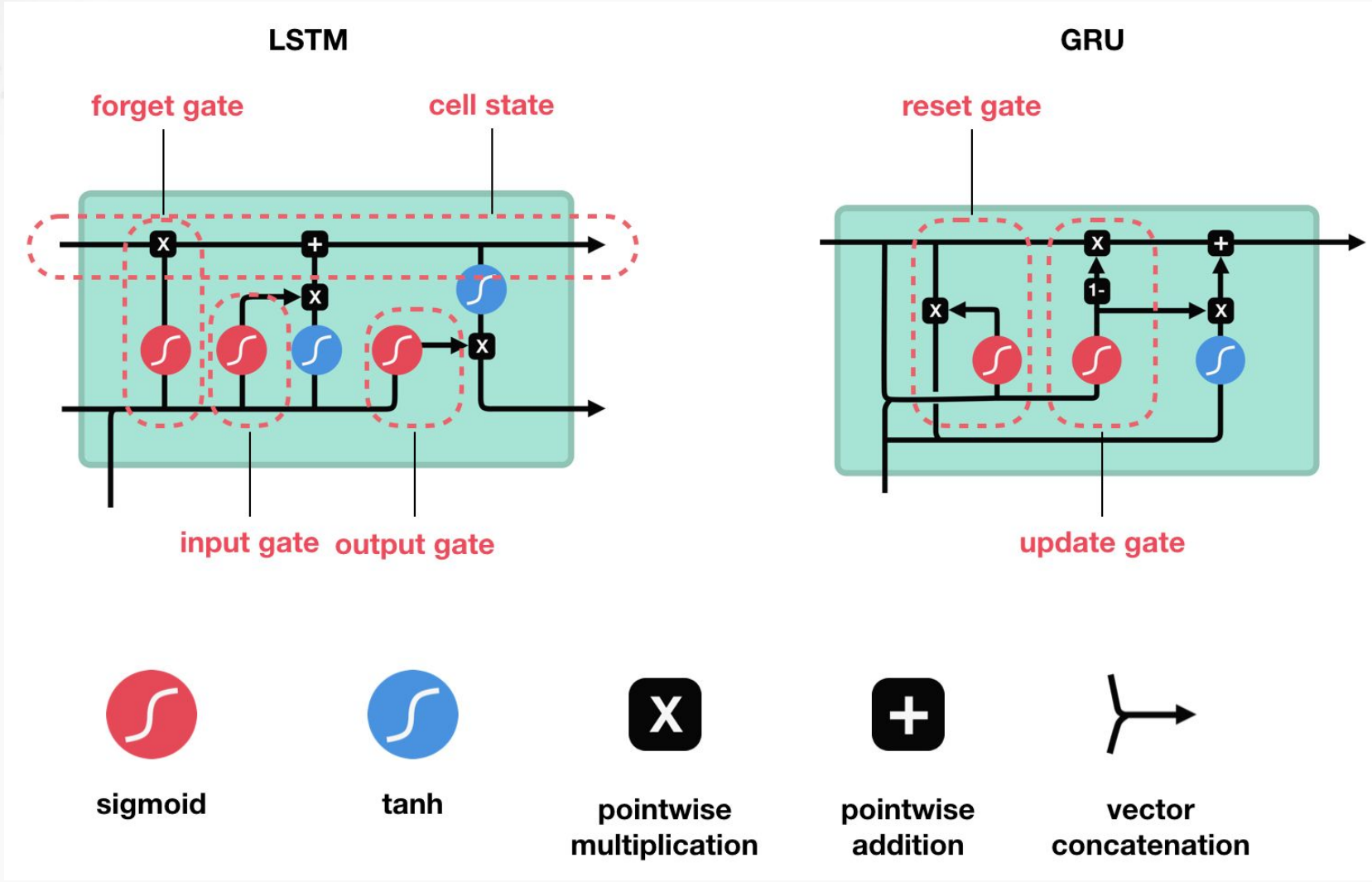
  entire input sequence - > an internal representation

- **Decoder**

  encoded input sequence + word already generated> output word

Use LSTM for the Encoder & Decoder?
Good at processing sequential data

LSTM vs Gated Recurrent Unit(GRU)



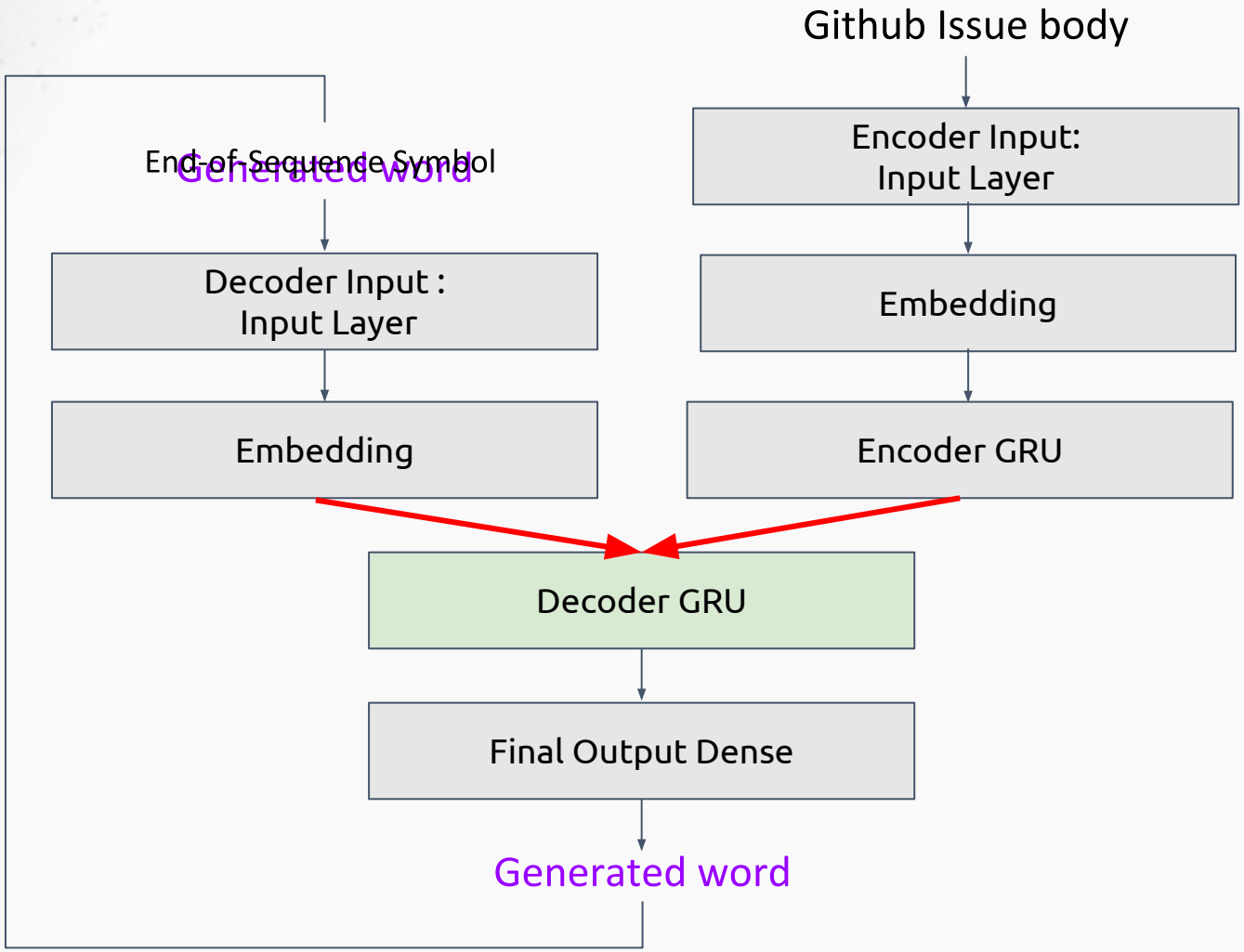
Memory-Long Sequence data

GRU Advantages

- Less complex,
- Takes less parameters to train
- Faster training speed.

Keras

Model Architecture



Encoder - Decoder Architecture

- Input layer: Instantiate data into a Keras tensor.
- Embedding layer - turns the word into a distributed representation
- GRU layer: gated recurrent unit for processing sequential data.
- Final Output Dense: densely connected layer.

Train

Hyper - Parameters

- Learning rate: 0.001
- Optimizer: Nadam, enables adaptive learning rates for each parameter
- Loss function: categorical cross entropy
- Batch size: 1200
- Epochs: 5

Data Size: 5 million

Nvidia Tesla V100

Around 500 seconds per epoch

Result on Test Set

ROUGE Score
Standard metric for automatic title generation evaluation

Metric	Score
ROUGE-1	0.2357
ROUGE-2	0.0859
ROUGE-l	0.2065

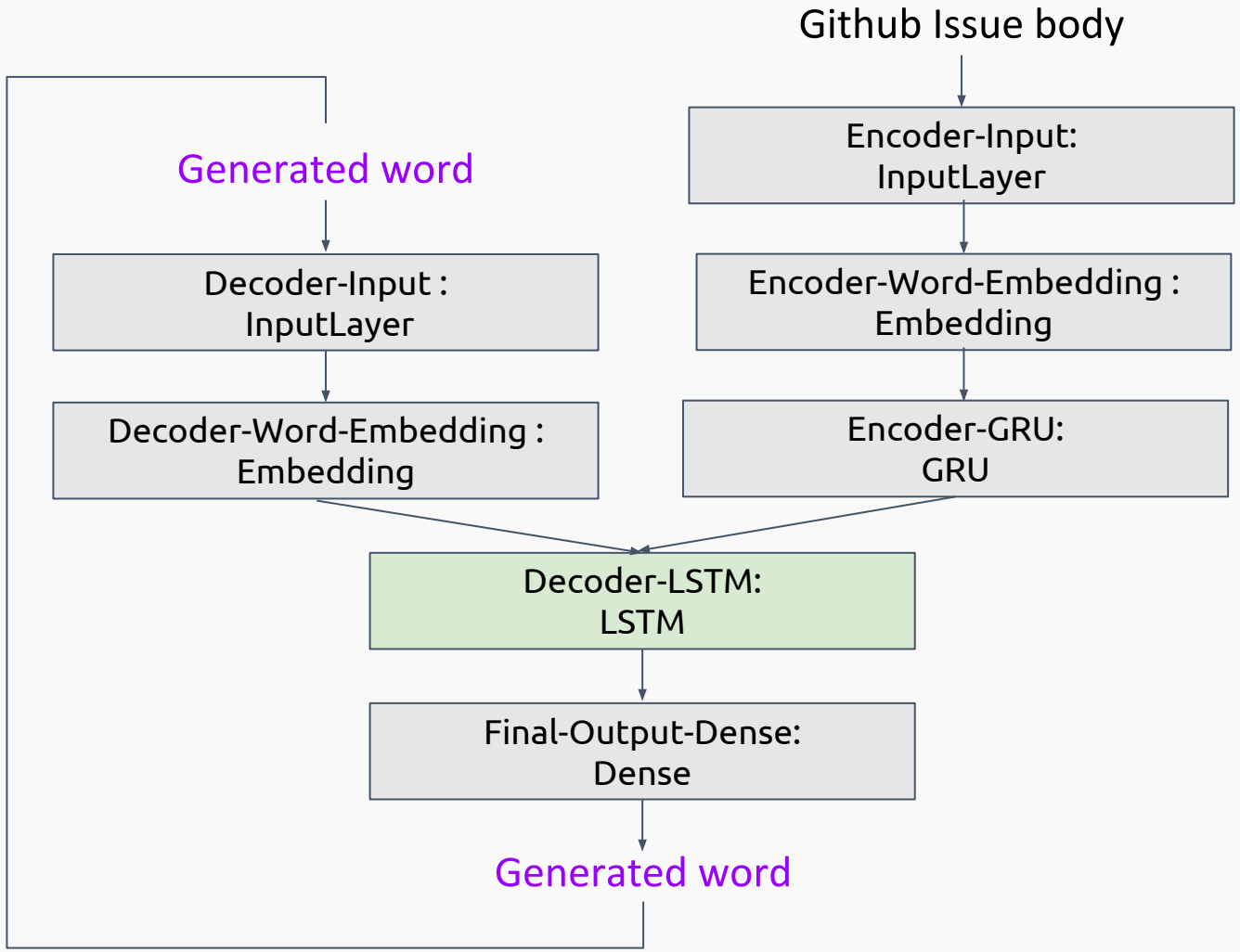
Our Model

Metric	Score
ROUGE-1	0.35
ROUGE-2	0.18
ROUGE-l	0.34

State-of-the-art Short Text Summary Generation Result*

*For Comparison: https://aclweb.org/anthology/P18-2027

Alternative Algorithm for Comparison: Replace GRU with LSTM



04

Analysis of results
Conclusion

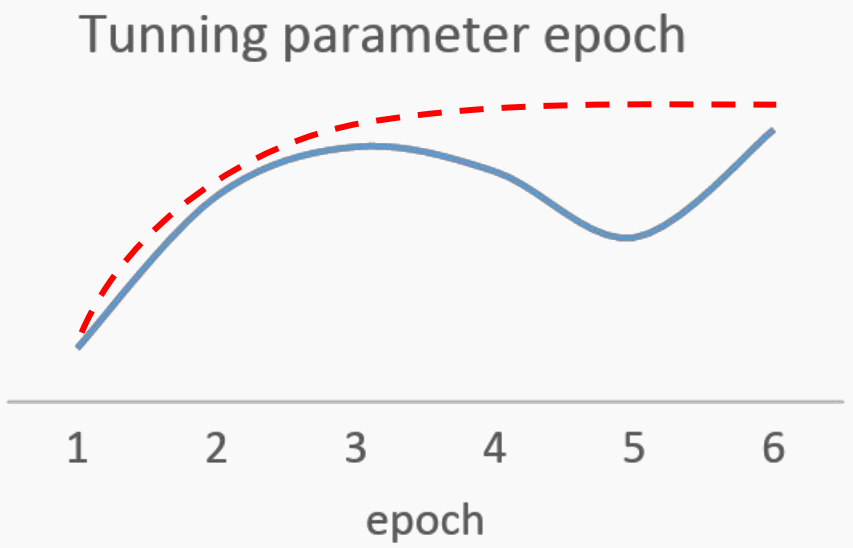
Tuning of 3 Key Hyperparameters

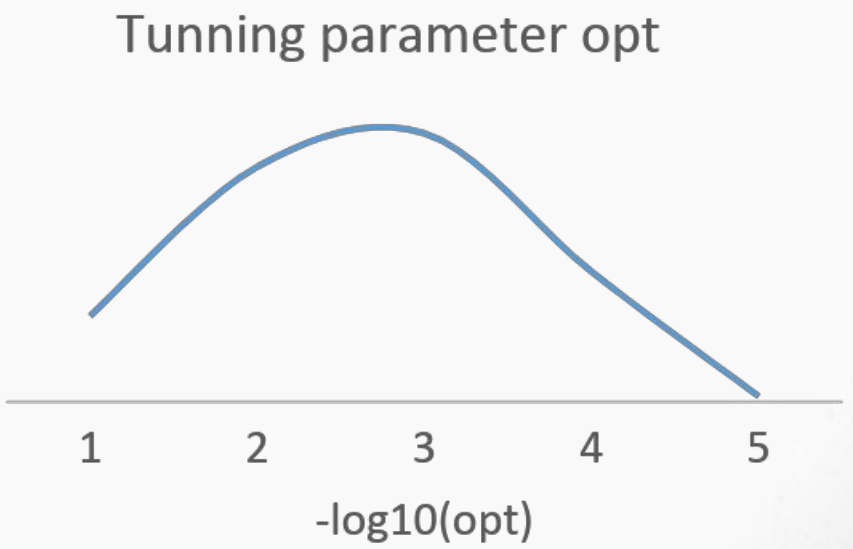
Learning rate: correlated with learning trate
Batch Size: the number of samples processed before model is updated
Epoch: the number of complete passes through the training dataset

K-Fold Cross validation with K = 5 is introduced to limit the over-fitting problem

Final settings:

Learning rate = 0.001
Batch Size = 1200
Epoch = 5





Cross comparison between models

Observations:

1. **Data imbalance** (ROUGE vs Number of Epochs reduces training loss. This has an increasing trends, but varies in different runs)
2. **Cross Validation increases ROUGE**
3. **Training data size positively correlated with ROUGE**

Model 1: GRU

Metric	Score
ROUGE-1	0.2165
ROUGE-2	0.0745
ROUGE-l	0.1896

Average: 0.1602

Model 2: LSTM

Metric	Score
ROUGE-1	0.1435
ROUGE-2	0.0474
ROUGE-l	0.1286

Average: 0.1065

Model 1 outperforms Model 2 in this project.

1. This can be attributed to that the number of training data we have is not large enough
2. Casual and informal nature of GitHub issues

Example of Results

Body:
issue overview add a new property to disable detection of image stream files those ended with -is.yml from target directory. expected behaviour by default cube should not process image stream files if user does not set it. current behaviour cube always try to execute -is.yml files which can cause some problems in most of cases, for example if you are using kuberentes instead of openshift or if you use together fabric8 maven plugin with cube.

Original Title:
add a new property to disable detection of image stream files

GRU Title:
add a way to disable image detection

LSTM Title:
add image detection * number *

Future Work

1. More extensive hyper parameter training to obtain a hyper parameters that optimize the performance of Neural Network
2. Train the model by feeding in a larger and more comprehensive dataset
3. Extend our project to other datasets, for example:
    a. News articles
    b. Research papers
    c. Forum posts
4. Add more layers to make the output more grammarly reasonable

Appendix

Reference:

Lin, C.-Y. (n.d.). ROUGE: A Package for Automatic Evaluation of Summaries. *Information Sciences Institute University of Southern California*.

Brownlee, J. (2017, July 3). *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Brownlee, J. (2017, December 8). *Encoder-Decoder Models for Text Summarization in Keras*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/encoder-decoder-models-text-summarization-keras/

GitHub Collaboration Repository:
https://github.com/ShirleyHan6/CE9010-Group-Project

Contributions

01 DEZHENG

Presentation Slides
Data Problem
Data Acquisition
LSTM & GRU
Comments on notebook

02 GLENN

Data Scraper
Data Exploration
- Word Cloud
- Bar Chart
- Dynamic Visualization

Pre-Processing

03 SIMENG

Data Analysis:
- Model Architecture
- ROUGE Script
- Server Setup

04 YANBIN

Data Analysis:
- Parameter tuning
- Cross validation

Analysis of Results