TWITTER SENTIMENT ANALYSIS

CLOUD & MACHINE LEARNING - CSCI-GA 3030-025

Srishti Bhargava, Daniel Sabba
Courant Institute of Mathematical Sciences
New York University

PROJECT DESCRIPTION

MOTIVATION: With billions of people in lockdown, their **perception of reality** is exclusively defined by information received through **digital media**.

LANDSCAPE: In this environment, a tool that uses machine learning algorithms to gather sentiment from this digital information would allow for **uniquely precise representation of overall sentiment**.

GOAL: Leverage the Kubernetes infrastructure to enable **scalable** and **tailored-by-user** sentiment analysis of tweets.

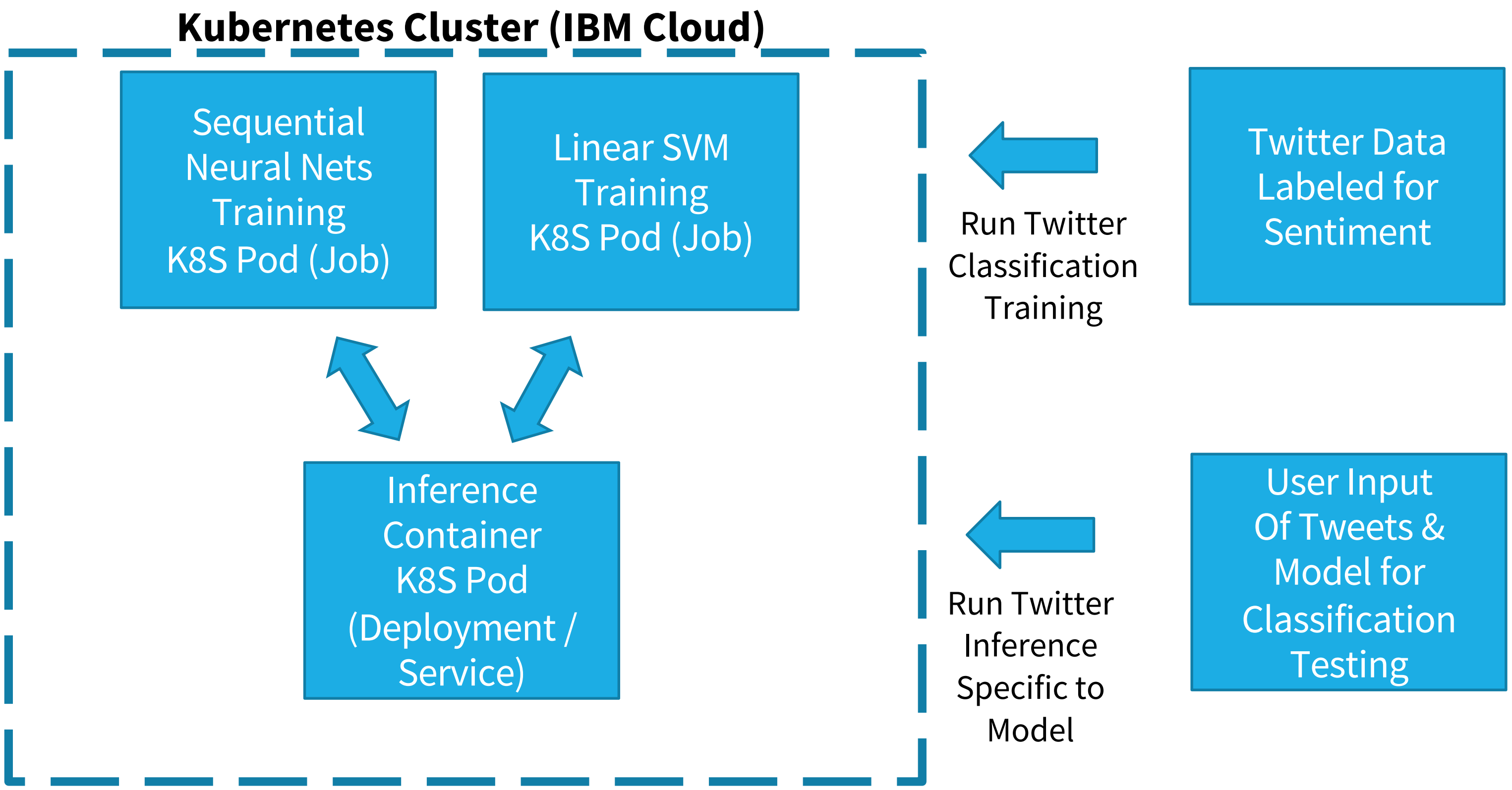
HOW DOES IT WORK?

Two training containers (jobs) run text classification training into positive, negative, neutral tweets using **Deep Sequential Neural Networks** and **Linear Support Vector Machines.**

An inference container (deployment) performs inference in the **pre-trained models** saved in **volumes mounted** in each training container.

In a web-based User Interface, a selection can be made between machine learning models, and a new tweet can be provided for inference.

BACKEND OF CLOUD/ML SENTIMENT ANALYSIS


Kubernetes Cluster (IBM Cloud)
Sequential Neural Nets Training K8S Pod (Job)
Linear SVM Training K8S Pod (Job)
Inference Container K8S Pod (Deployment / Service)
Run Twitter Classification Training
Twitter Data Labeled for Sentiment
Run Twitter Inference Specific to Model
User Input Of Tweets & Model for Classification Testing

DEEP SEQUENTIAL NEURAL NETWORKS

Converted tweet text into one-hot matrices and made all tweets of same length

Used a 3 layer dense neural network with one layer each for input and output, along with a hidden layer.

Layer 1: 512 layer dense network which accepts one-hot matrix of specified length

Layer 2: 256 layer dense network (hidden layer)

Layer 3: 3 layer dense network, output layer with 3 possible outputs

DEEP SEQUENTIAL NEURAL NETWORKS

Model uses 2 dropout layers which are used to randomly remove data which helps to avoid overfitting.

Activation Function : ReLU for layer 1 and 2, softmax for output layer

Optimizer: Adam, to update parameters and minimize the cost of the neural network

Loss function: Categorical crossentropy, calculates a score that summarizes the average difference between actual and predicted probability distributions for all classes

LINEAR SUPPORT VECTOR MACHINES

Popular machine learning algorithm to perform text classification.

Performed tf-idf transformation on the tweets before feeding them to the model.

Linear SVM performs classification by finding the best hyperplane to differentiate n classes in an n-dimensional space.

The hyperplane acts as a decision boundary for any new data point

DATASET & MODEL OBSERVATIONS

From the 15,000 labeled tweets, over 9,000 were labeled negative, leading Linear SVM to predict some tweets (including neutral) as negative.

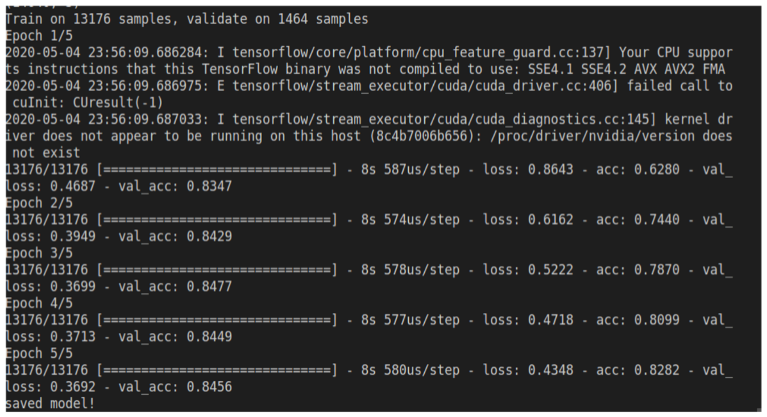
Deep sequential neural networks exhibit better performance and superior ability to handle outliers.

The categorical crossentropy loss used in the Neural Network tends to perform well when aiming for highest percentage accuracy even with imbalances in the dataset.

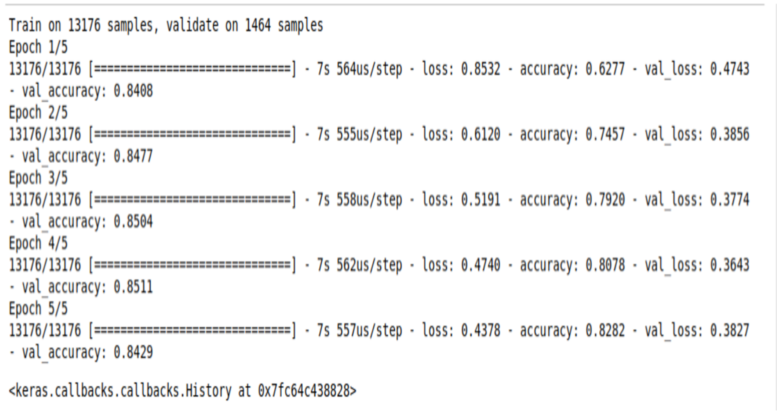
K(C)AAS: KUBERNETES (CONTAINER) AS A SERVICE

We used IBM Cloud to host our Kubernetes cluster – this service is a hybrid of KaaS and CaaS.

We compared training performance with AWS containers, leveraging their CaaS capabilities. Performance between AWS and IBM Cloud was very similar.



```
Train on 13176 samples, validate on 1464 samples
Epoch 1/5
2020-05-04 23:56:09.686284: I tensorflow/core/platform/cpu_feature_guard.cc:137] Your CPU suppor
ts instructions that this TensorFlow binary was not compiled to use: SSE4.1 SSE4.2 AVX AVX2 FMA
2020-05-04 23:56:09.686975: E tensorflow/stream_executor/cuda/cuda_driver.cc:406] failed call to
 cuInit: CUresult(-1)
2020-05-04 23:56:09.687033: I tensorflow/stream_executor/cuda/cuda_diagnostics.cc:145] kernel dr
iver does not appear to be running on this host (8c4b7006b656): /proc/driver/nvidia/version does
 not exist
13176/13176 [==============================] - 8s 587us/step - loss: 0.8643 - acc: 0.6280 - val_
loss: 0.4687 - val_acc: 0.8347
Epoch 2/5
13176/13176 [==============================] - 8s 574us/step - loss: 0.6162 - acc: 0.7440 - val_
loss: 0.3949 - val_acc: 0.8429
Epoch 3/5
13176/13176 [==============================] - 8s 578us/step - loss: 0.5222 - acc: 0.7870 - val_
loss: 0.3699 - val_acc: 0.8477
Epoch 4/5
13176/13176 [==============================] - 8s 577us/step - loss: 0.4718 - acc: 0.8099 - val_
loss: 0.3713 - val_acc: 0.8449
Epoch 5/5
13176/13176 [==============================] - 8s 580us/step - loss: 0.4348 - acc: 0.8282 - val_
loss: 0.3692 - val_acc: 0.8456
saved model!
```



```
Train on 13176 samples, validate on 1464 samples
Epoch 1/5
13176/13176 [==============================] - 7s 564us/step - loss: 0.8532 - accuracy: 0.6277 - val_loss: 0.4743
- val_accuracy: 0.8408
Epoch 2/5
13176/13176 [==============================] - 7s 555us/step - loss: 0.6120 - accuracy: 0.7457 - val_loss: 0.3856
- val_accuracy: 0.8477
Epoch 3/5
13176/13176 [==============================] - 7s 558us/step - loss: 0.5191 - accuracy: 0.7920 - val_loss: 0.3774
- val_accuracy: 0.8504
Epoch 4/5
13176/13176 [==============================] - 7s 562us/step - loss: 0.4740 - accuracy: 0.8078 - val_loss: 0.3643
- val_accuracy: 0.8511
Epoch 5/5
13176/13176 [==============================] - 7s 557us/step - loss: 0.4378 - accuracy: 0.8282 - val_loss: 0.3827
- val_accuracy: 0.8429

<keras.callbacks.callbacks.History at 0x7fc64c438828>
```

INFERENCE CONTAINER & USER INTERFACE

We created an inference container as a kubectl deployment as well as a service to allow that container to be accessed externally. This inference container accesses different trained models saved in volumes mounted in training containers to perform inference.

The user interface allows for a new tweet to be evaluated as well as for a training model to be selected.

This user interface will be hosted in a webserver and is accessing the inference container in Kubernetes external IP address and a NodePort.

CHALLENGES & LESSONS LEARNED

DATA: Over the course of project we realized how important it is to have “ready-to-use data” to be able to focus on substantive ML/Cloud aspects of the project. We realized this as we were dedicating a lot of time preparing raw data from the API.

CLUSTERING: We initially expected to be able to obtain meaningful results through clustering, and once this did not materialize, we decided looked for labeled datasets.

DATA CLASSIFICATION: We had intended to perform text classification (for example identifying topic) instead of sentiment identification. We were thwarted in that effort by the very limited amount of publicly available twitter datasets with topic labels.

MODEL SELECTION: We evaluated 4-5 training models, settling with the final two (SVM & SNN) for accuracy considerations. **In hindsight, we could have not made an ex-ante selection of these two models, but instead kept all of them as independent containers the user could select based on whatever data is to be evaluated.**

CONCLUSION & FUTURE WORK

Throughout this project, we were able to experience the benefits of scalability of the Kubernetes infrastructure. It allowed us to **make a job performed by source code into a dynamic living environment that is scalable, elastic and can be accessed everywhere.**

These features are particularly relevant as, while the skeleton of the model is completed, we need to continue to improve the data environment and fine tune the models appropriately – this can be uniquely done in the Kubernetes cluster framework.

CONCLUSION & FUTURE WORK

LIVE DEMO: We intend to use our Google Cloud credits to maintain the UI & cluster running as we continue to refine the tool and collaborate on this idea. This will allow us to have a continued demo as well as a repository page for source code.

DYNAMIC: The Kubernetes framework allows us to create new clusters as we experiment with new models, while old modes can continue to run for comparison.

DATA: The precision of this model can be enhanced with more labeled twitter data. Now this project is done, we are exploring alternatives to scale this data up so the tool can be more effective and useful.

APPLICATION: One immediate commercial application would be on the design of systematic trading strategies to capitalize on heterogeneous sentiment clustering across different markets. Immediate applications would be global long short equity and fixed income relative value.