GeoSocial-Aurora

Applying ML techniques for automatic tweet events classification

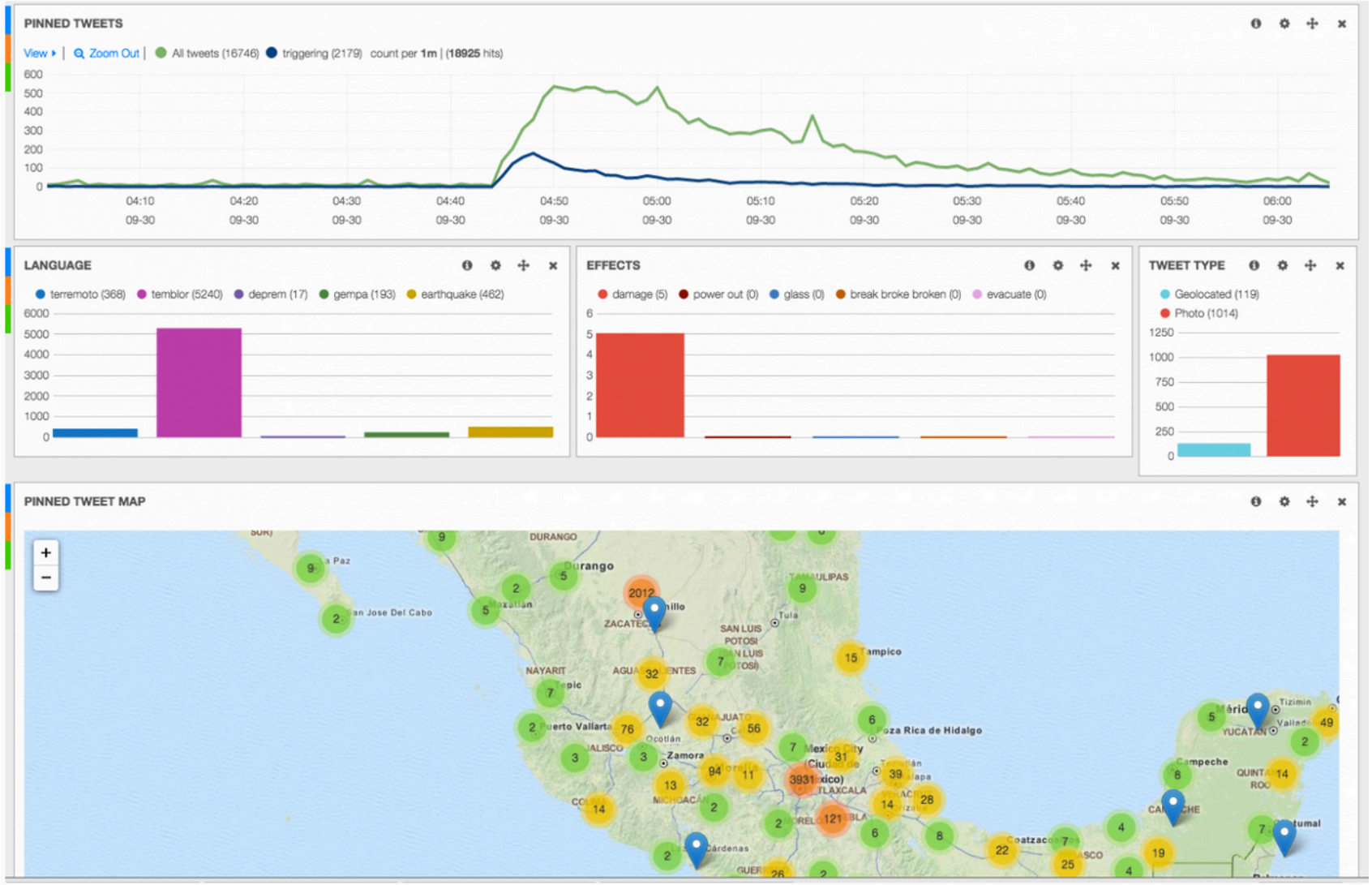
Rosa Filgueira

Introduction

- Constant flow of information in the form of **short texts** makes (it in theory possible) to
  - collect **real time information** about all sorts of events that are being written about

- In GeoSocial-Aurora:
  - **catching the events related with** "Auroras Borealis"

How the USGS uses Twitter data to track earthquakes



Re

Tak
The Ur
Yayoi 2-
To
sakaki@
to

ABSTRACT

Twitter, a popular mi
attention recently. A
is its real-time natur
occurs, people make
to the earthquake, v
occurrence promptly,
described in this pap
action of events such
pose an algorithm to
event. To detect a
tweets based on feat
the number of words
produce a probabilis

2.1 Semantic Analysis on Tweet

To detect a target event from Twitter, we search from Twitter and find useful tweets. Tweets might include mentions of the target event. For example, users might make tweets such as "Earthquake!" or "Now it is shaking". Consequently, *earthquake* or *shaking* might be keywords (which we call *query words*). But users might also make tweets such as "I am attending an Earthquake Conference", or "Someone is shaking hands with my boss". Moreover, even if a tweet is referring to the target event, it might not be appropriate as an event report. For instance, a user makes tweets such as "The earthquake yesterday was scaring", or "Three earthquakes in four days. Japan scares me." These tweets are truly descriptions of the target event, but they are not real-time reports of the events. Therefore, it is necessary to clarify that a tweet is truly referring to an actual earthquake occurrence, which is denoted as a positive class.

To classify a tweet into a positive class or a negative class, we use a support vector machine (SVM) [14], which is a widely used machine-learning algorithm. By preparing positive and negative examples as a training set, we can produce a model to classify tweets automatically into positive and negative categories.

We prepare three groups of features for each tweet as follows:

Features A (statistical features) the number of words in a tweet message, and the position of the query word within a tweet.

yo
-ku

.t.u-

e, has received
al network used
remain socially
and co-workers
s [18]. Twitter
nswers must be
nessage, called a
nd colleagues. A
read her tweets.
r need not nec-
, which renders
r its launch on
oidly. They are

Problem description

- Not all the tweets with the term “Auroras” are related with “Auroras borealis”.

Problem description

Non-Aurora-B.' related events

Can we just take a moment to admire Aurora's face in this pic

I'm at Aurora Boulevard in Quezon City

Hello Aurora, This is customized Tees and Hoodies with your name!

NIKE DUNK HIGH PREMIUM SB NORTHERN LIGHTS

Auroras' B. related events

#northernlights visible over @kawarthalakes right now @AuroraMAX

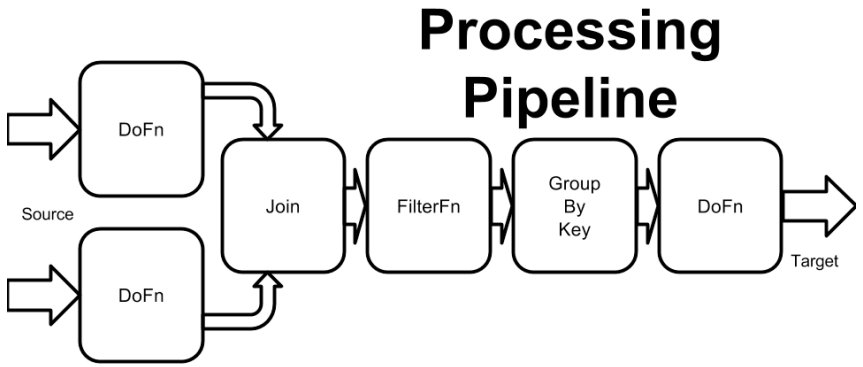
Aurora Prediction: Kp: 4.67(20:15) 5.33(23:15) MDT (earth 5.33) Status: Moderate Storm

theyre was northern lights last night they were beautiful:)

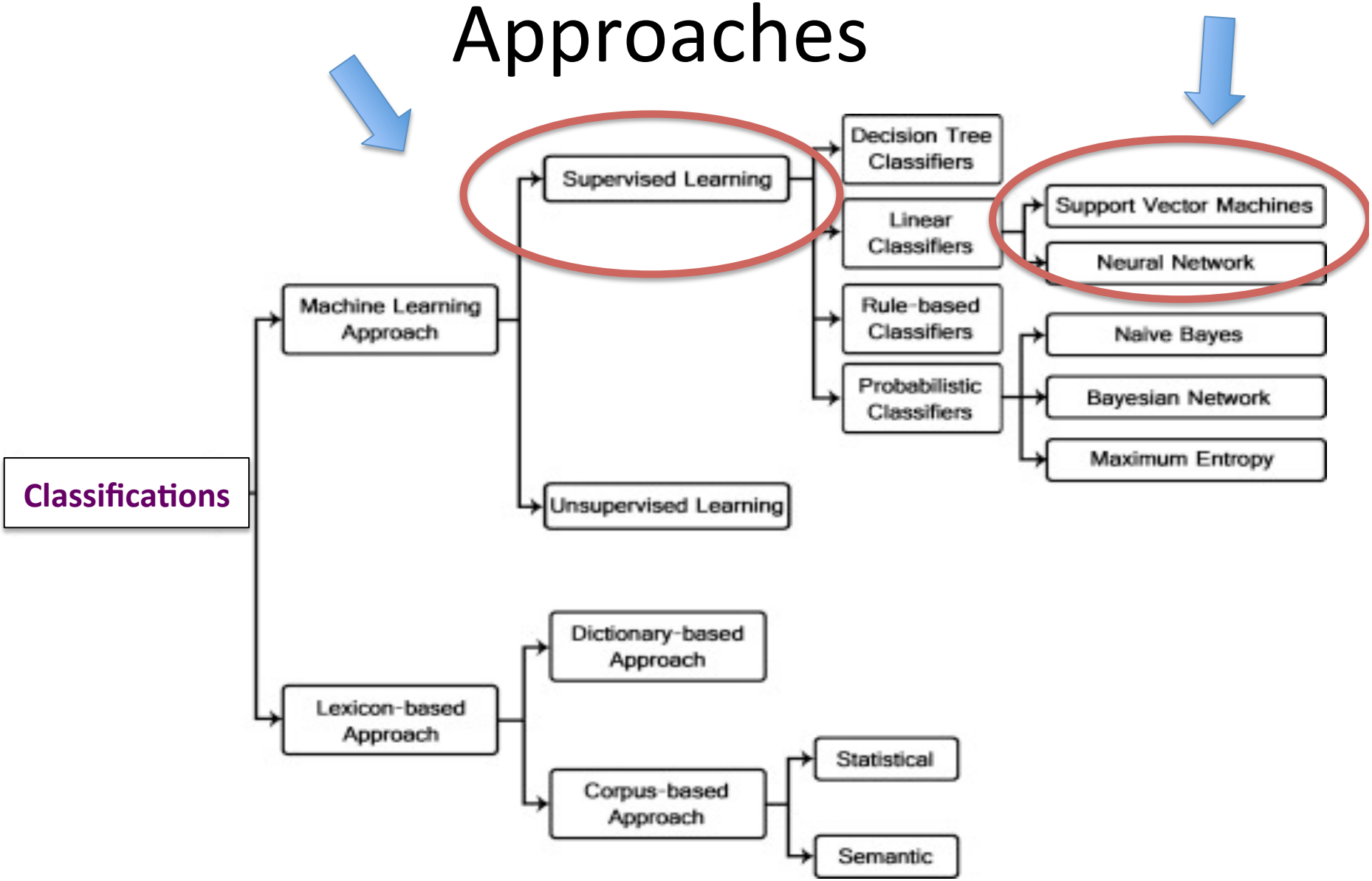
We currently have a G3 solar storm in progress brilliant chance of seen the Aurora tonight

“Simple” objective(s)

- From the tweets that are being collected:
  1. Classify "automatically” the tweets that are related with **Auroras borealis** before plotting
     - Artificial Intelligence/Machine Learning/NLP
       - “Text” / “Sentence” / ”Topic” classification
  2. Run some statistics in “parallel”

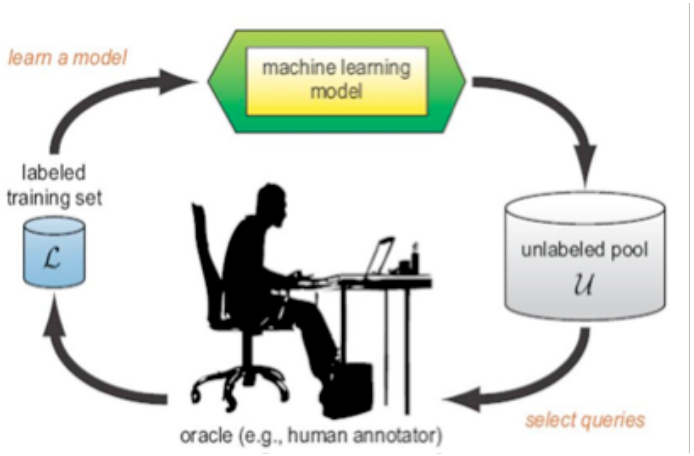


Approaches


Classifications
Machine Learning Approach
Supervised Learning
Decision Tree Classifiers
Linear Classifiers
Support Vector Machines
Neural Network
Rule-based Classifiers
Probabilistic Classifiers
Naive Bayes
Bayesian Network
Maximum Entropy
Unsupervised Learning
Lexicon-based Approach
Dictionary-based Approach
Corpus-based Approach
Statistical
Semantic

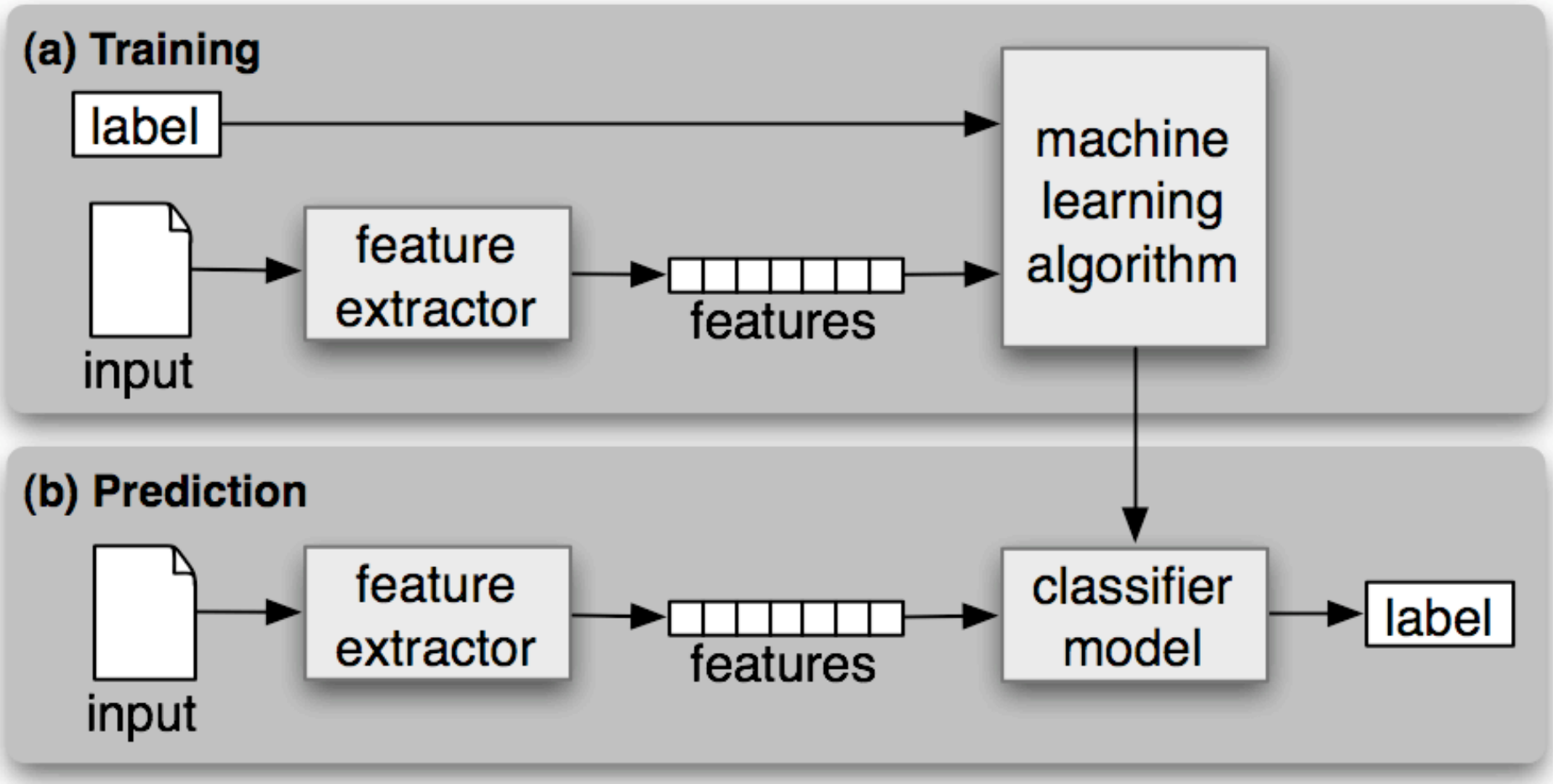
Machine Learning - Supervised Methods

- Create training dataset :
  - **Labels: "Others"** and **"Aurora"**



- Create a model/classifier :
  - Training process in which it is required:
    - to make predictions
    - Correct the model when those predictions are wrong.
  - The training process continues until
    - the **model** achieves a desired level of accuracy

Machine Learning - Supervised Methods

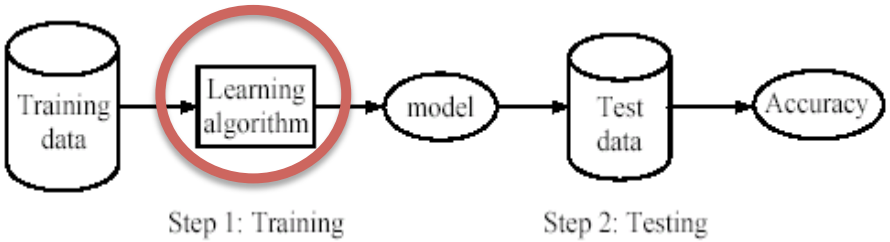


Some ML algorithm types

- Regression:
  - is concerned with modeling the relationship between variables that is iteratively refined using a measure of error in the predictions made by the model.
    - Logistic Regression, Linear Regression
- Neural Networks:
  - are models that are inspired by the structure and/or function of biological neural networks.
  - Pattern matching that are commonly used for regression and classification problems.
- Bayesian:
  - are those that explicitly apply Bayes' Theorem for problems such as classification and regression.
    - Naïve Byes, Gaussian Naïve Bayes, Bayesian Network, ..
- Deep Learning:
  - a modern update to Artificial Neural Networks that exploit abundant cheap computation.
  - They are concerned with building much larger and more complex neural
    - Convolutional Neural Networks (CNN), Deep Belief Networks (DBN), ….

First Approach- Support Vector Machine Classifiers

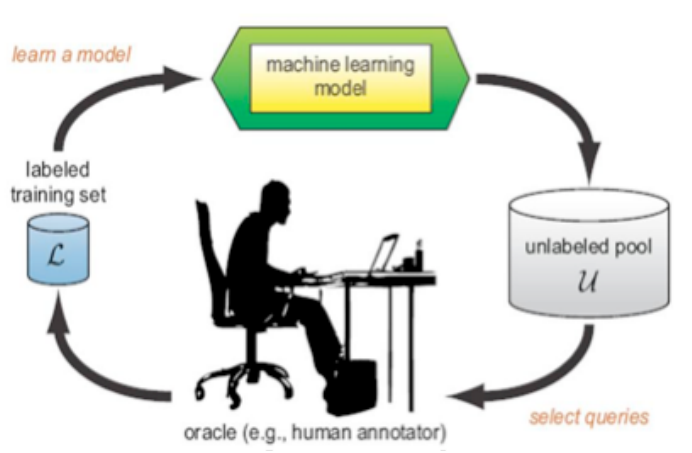
- Support Vector Machines (SVM) classify data into two classes (which is why it is called non-probabilistic binary classifier).
- Learning algorithms (**ScikitLearn** Library)
  - Linear Regression
  - Logistic Regression
  - Naïve Bayesian
  - Stochastic Gradient Descent



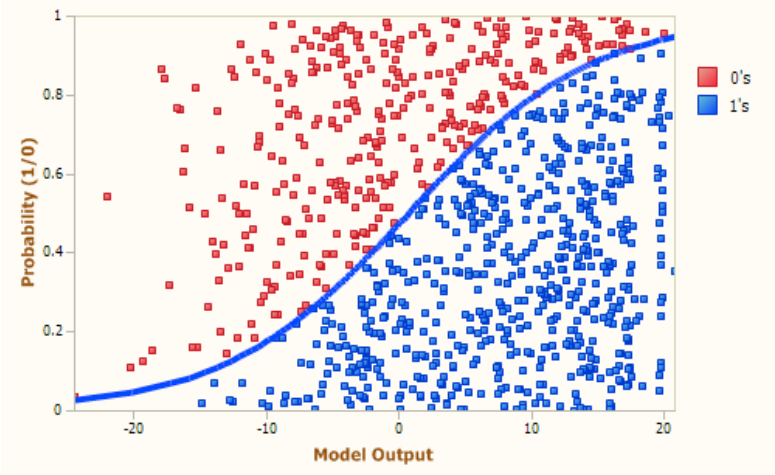
(given an input,

the classifier returns its probabilities to belong to each class)

Methodology taken



- Label manually 1300 tweets
  - "Auroras" or "Others"
- Program a SVM model
  - Tested several algorithms and selected the one with the best accuracy → **Logistic Regression**



Example Logistic Regression model

Label tweets

- Create a pipeline program (**dispel4py**) that automatically extract in parallel the text field from an collection of tweets and save them in a json file
- Download data --- Diego gave them to me ---
- Label manually the tweets
- Code: https://github.com/rosafilgueira/Data_Wrangling/blob/master/SocialComputing_Example/labeling.py
- dispel4py multi labeling.py –n 12 -d '{"read" : [{"input" : "tweets_5000_page1.json"}]}'

Program SVM model

- Create a python program using several SVM algorithms from **ScikitLearn** Library (a Machine learning library)
- Algorithms:
  - Multinomial Naïve Bayes
  - Linear SVC
  - **Logistic Regression**
  - Stochastic Gradient Descent
- 921 labeled tweets: 736 tweets for training each classifier, and 185 tweets for testing them
- Code: https://github.com/rosafilgueira/Data_Wrangling/blob/master/SocialComputing_Example/svm_model/train_classifier_svm.py

SVM- Logistic Regression - Accuracy

```
Building for evaluation
Train size 736
Y train size 736
I have chosen LogisticRegression
Evaluation model fit in 4.168 seconds
Classification Report:

Number of test 185, numer of errors 20
             precision    recall  f1-score   support

     Aurora       0.93      0.77      0.84        69
      Other       0.88      0.97      0.92       116

avg / total       0.90      0.89      0.89       185
```

Apply the SVM classifier

- Download data (**5000 tweets**) from a period of time with high activity of Auroras Borealis (**8-9 May 2016**)
- Build a data-pipeline workflow (**dispel4py**):
  - Preprocess (using NLTK library)
  - Classification (**Logistic Regression Model)**
    - Store the classification results into a file
  - Run statistics with the 'auroras' tweets based in their locations:
    - Which US states registered more activity
    - Which UK countries registered more activity
    - Which UK regions registered more activity

Some “statistics” based on Country/ State location – MapReduce --

- Input data: 3419 tweets
- Apply the classification model to the input data:
  - 2024 tweets classified as “Others”
  - 990 tweets classified as “Auroras” but their location is not from US neither UK
  - 405 tweets classified as “Auroras” from US or UK
    - 369 tweets classified as “Auroras”- mapped by “US” - reduced by “state”
      - Washington, 60
      - Alaska, 50
      - California, 26
      - Montana ,24
      - Indiana, 22
    - 36 tweets classified as “Auroras” - mapped by “UK” - reduced by “country”
      - Scotland, 18
      - England, 14
      - Wales, 4

Some “statistics” based on Country/ State location -- MapReduce

- 69 tweets classified as “Auroras” - mapped by “UK”- reduced by “region”
  - Northern Ireland, 12
  - Yorkshire, 9
  - North West, 9
  - South East, 7
  - Great London, 7

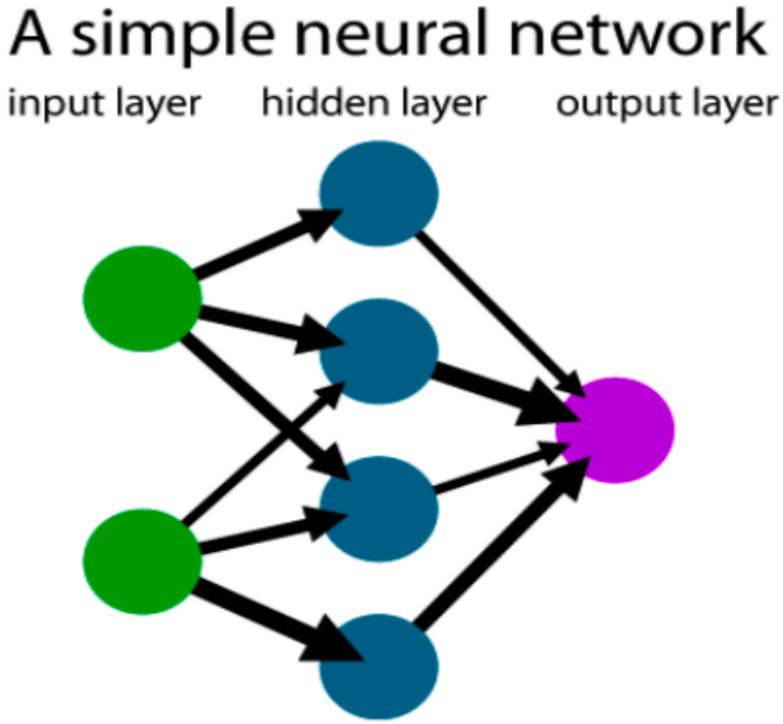
Second approach – Deep Learning

- Convolutional Neural Network (CNN) – “deep learning”
- Build the CNN classifier/model
- TensorFlow as a data-pipeline as an engine
- Compare both approaches

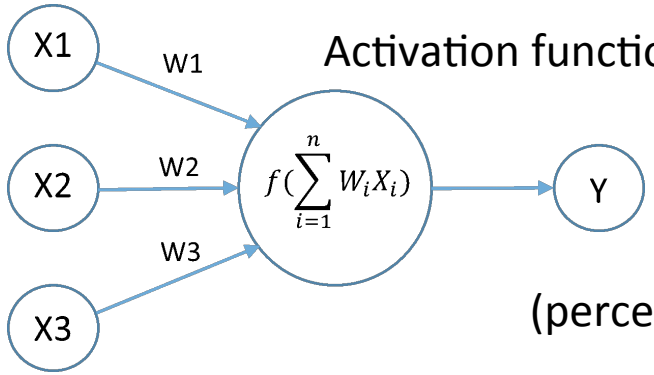
Artificial Neural Network

An Artificial Neural Network is an information processing paradigm that is inspired by the way biological nervous systems process information:

"a large number of highly interconnected simple processing elements (neurons) working together to solve specific problems"



"Learn the weights"

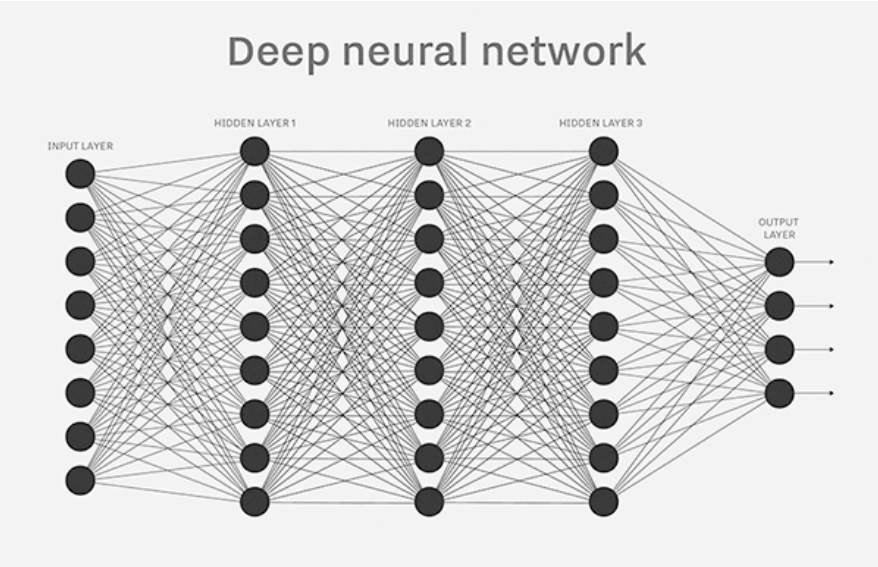


Activation function

(perceptron neural networks)

Deep Learning

- Many layered neural networks
- The more layers in a neural network, the more abstract features can be represented



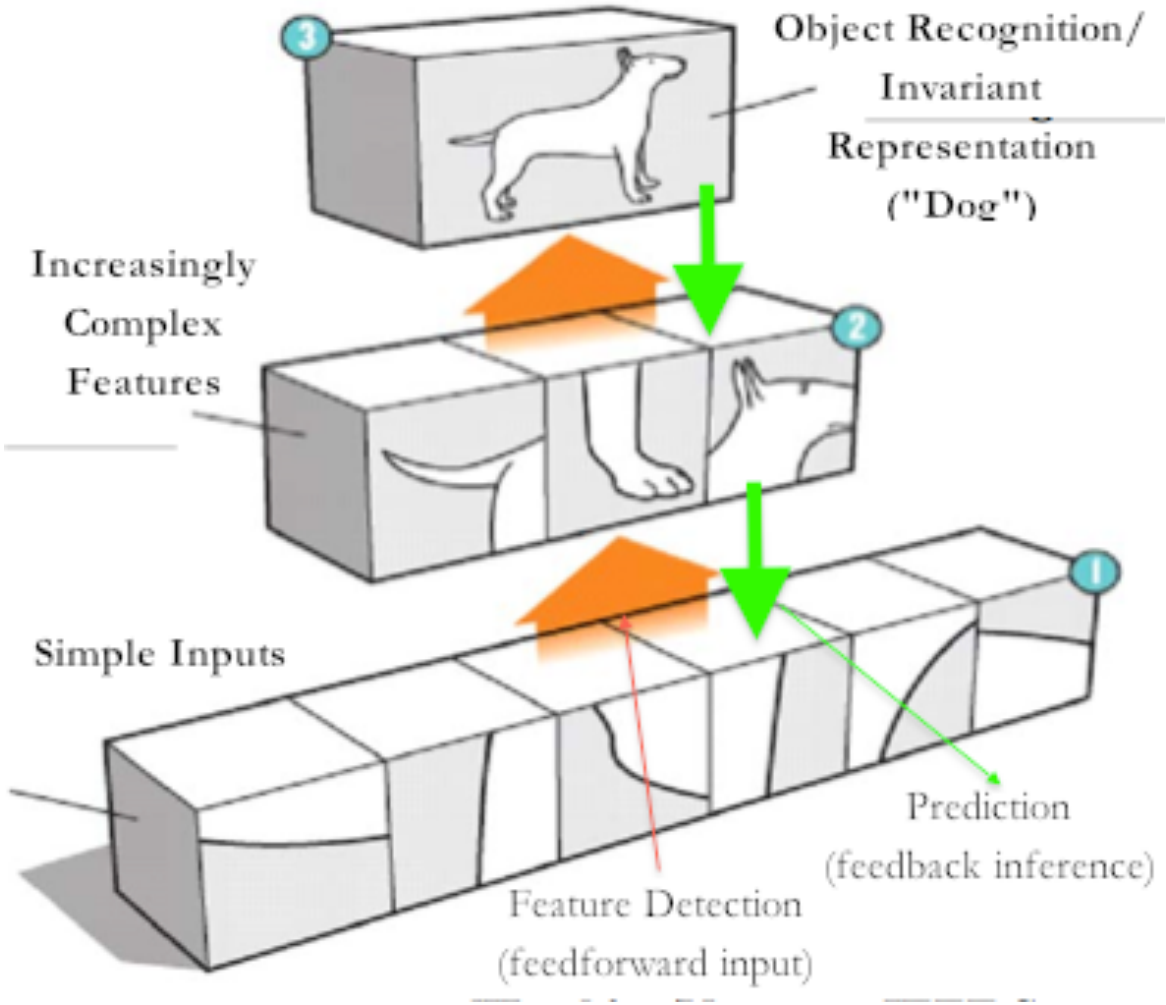
Classification between cat vs dog:

Top Layer: Cat or Dog

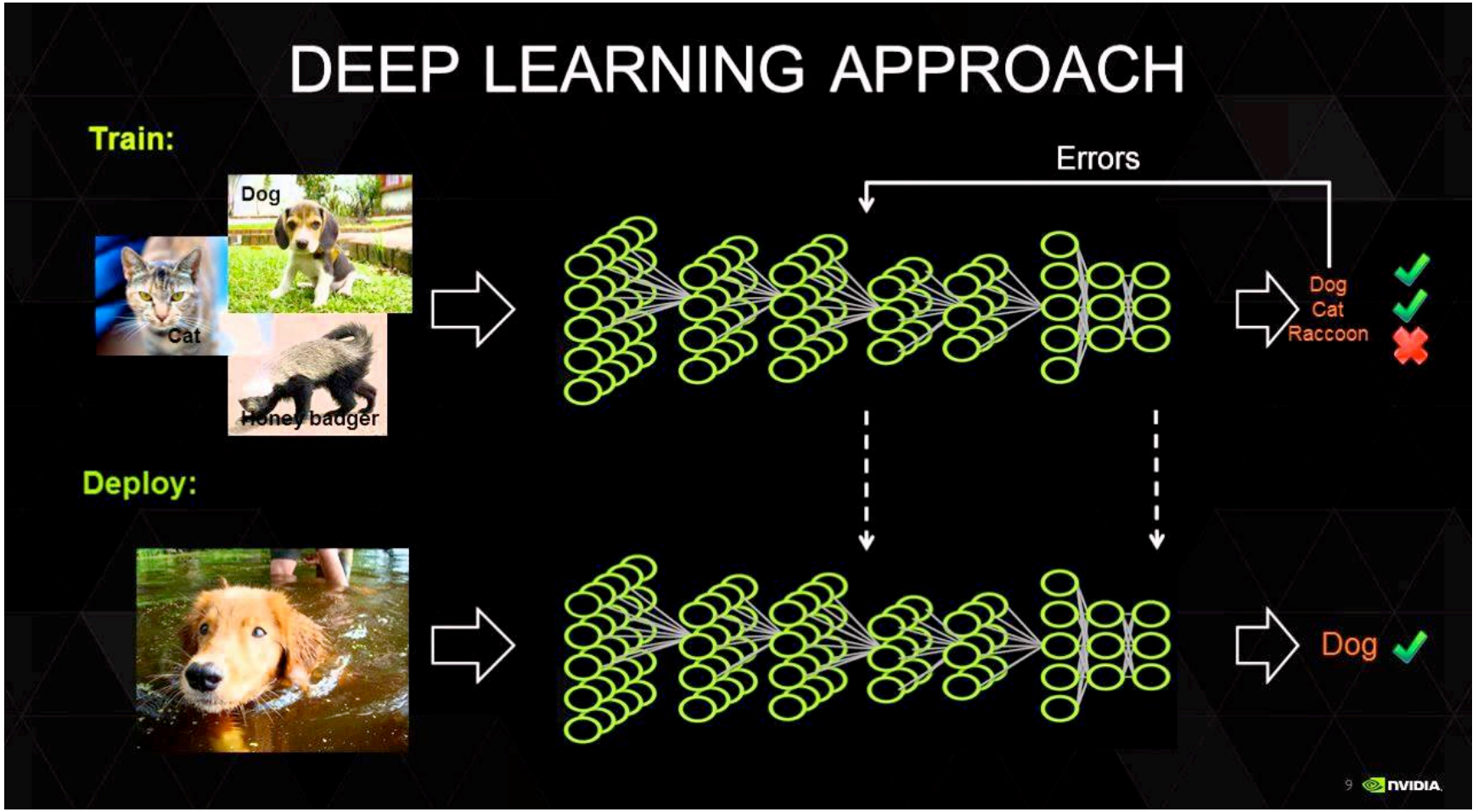
Higher Layers: Body, head, legs

Middle Layers: Fur patterns, eyes, ears

Bottom Layers: Edge detectors, curves, corners straight lines



Deep Learning



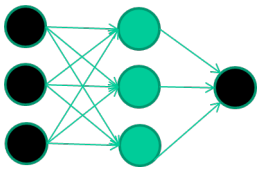
hmmm… OK, but:

Multilayer neural networks have been around for 25 years. What's actually new?

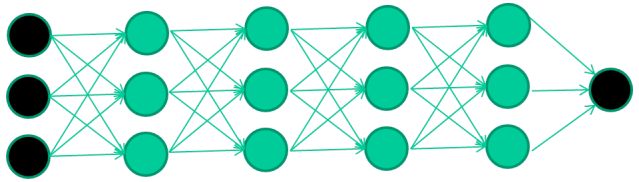
hmmm… OK, but:

Multilayer neural networks have been around for 25 years. What's actually new?

we have always had good algorithms for learning the weights in networks with 1 hidden layer



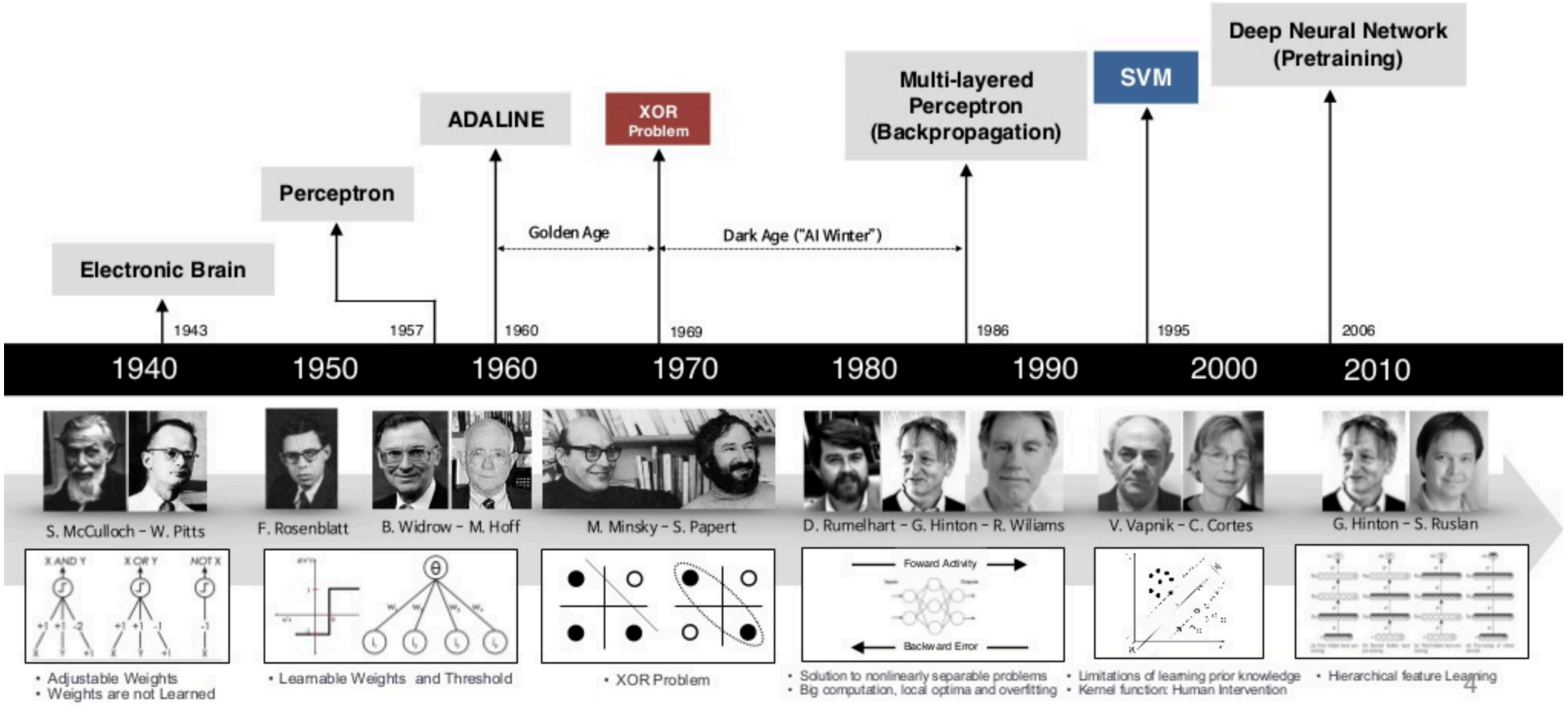
but these algorithms are not good at learning the weights for networks with more hidden layers



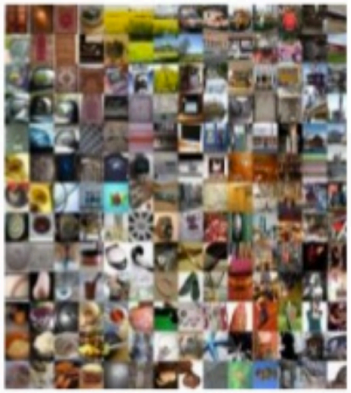
what's new is: **algorithms for training many-later networks**

hmmm… OK, but:

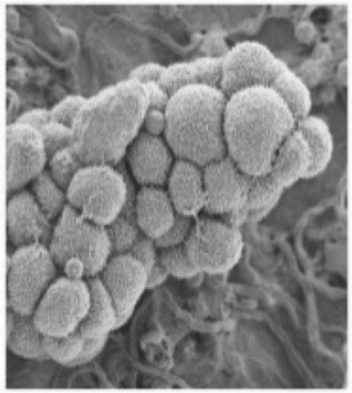
Multilayer neural networks have been around for 25 years. What's actually new?



Deep Learning – Many applications



INTERNET & CLOUD



MEDICINE & BIOLOGY

MEDIA & ENTERTAINMENT

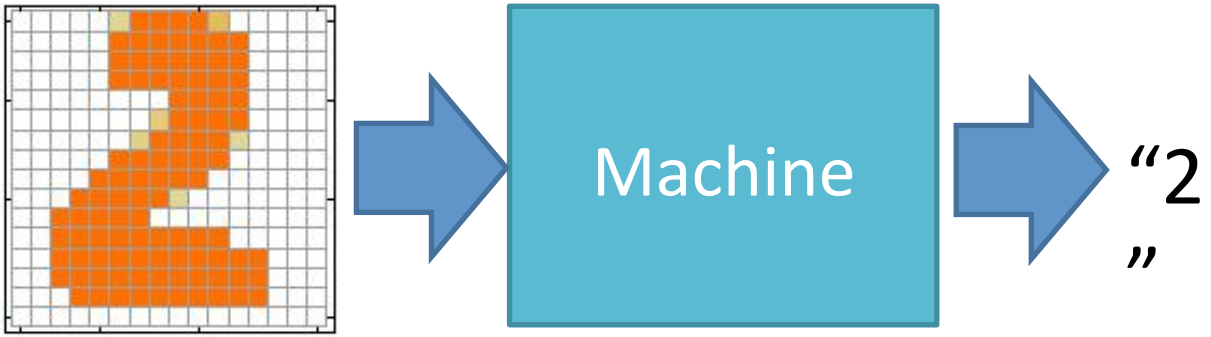
SECURITY & DEFENSE

AUTONOMOUS MACHINES

- Many fields

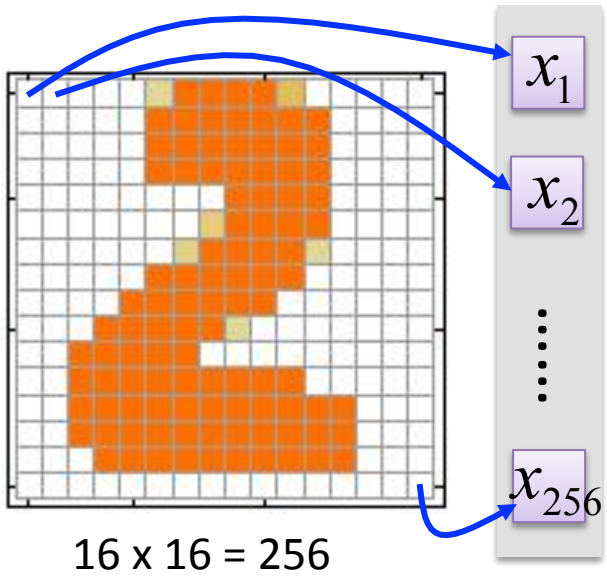
Example Application

- Handwriting Digit Recognition



Handwriting Digit Recognition

Input

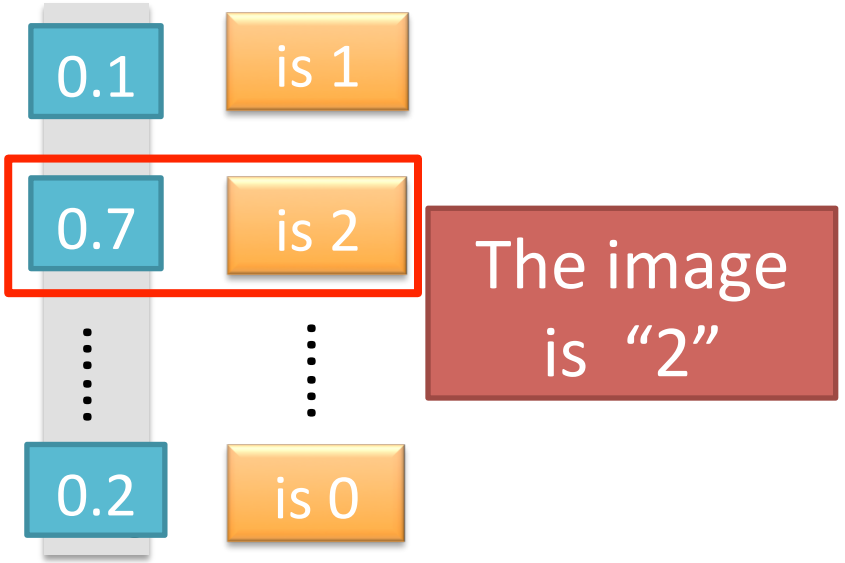


16 x 16 = 256

Ink → 1

No ink → 0

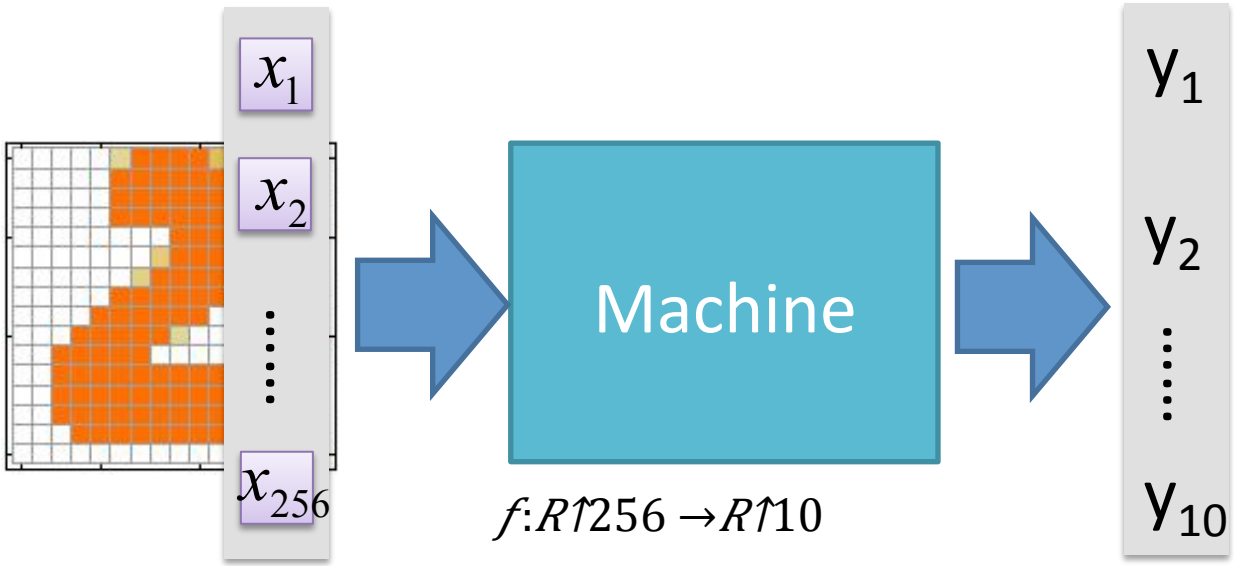
Output



Each dimension represents the confidence of a digit.

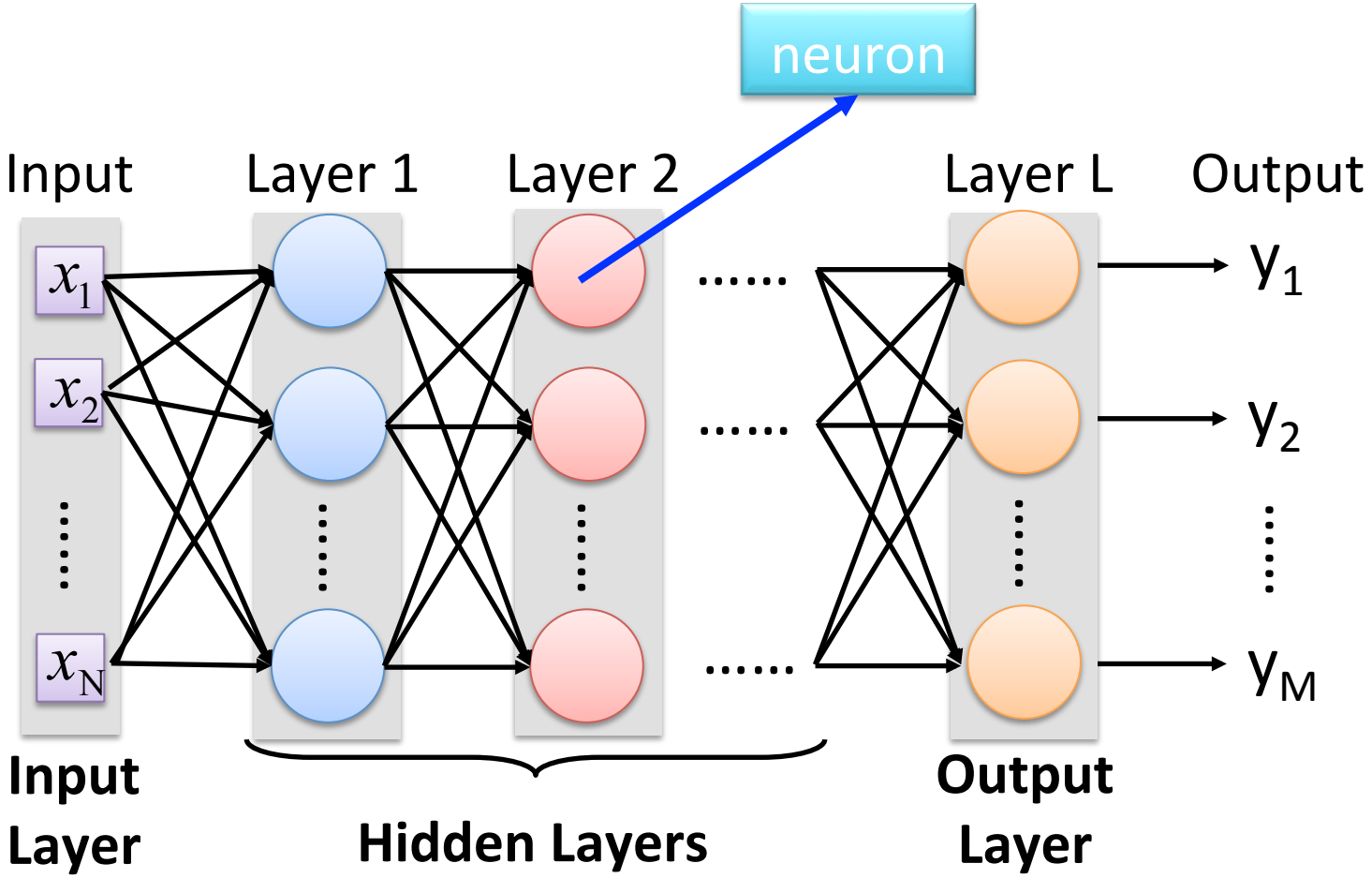
Example Application

- Handwriting Digit Recognition



In deep learning, the function f is represented by neural network

Neural Network



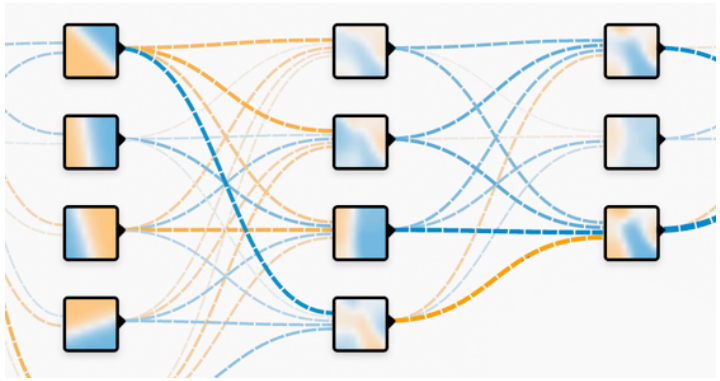
Deep means many hidden layers

What is Tensorflow?

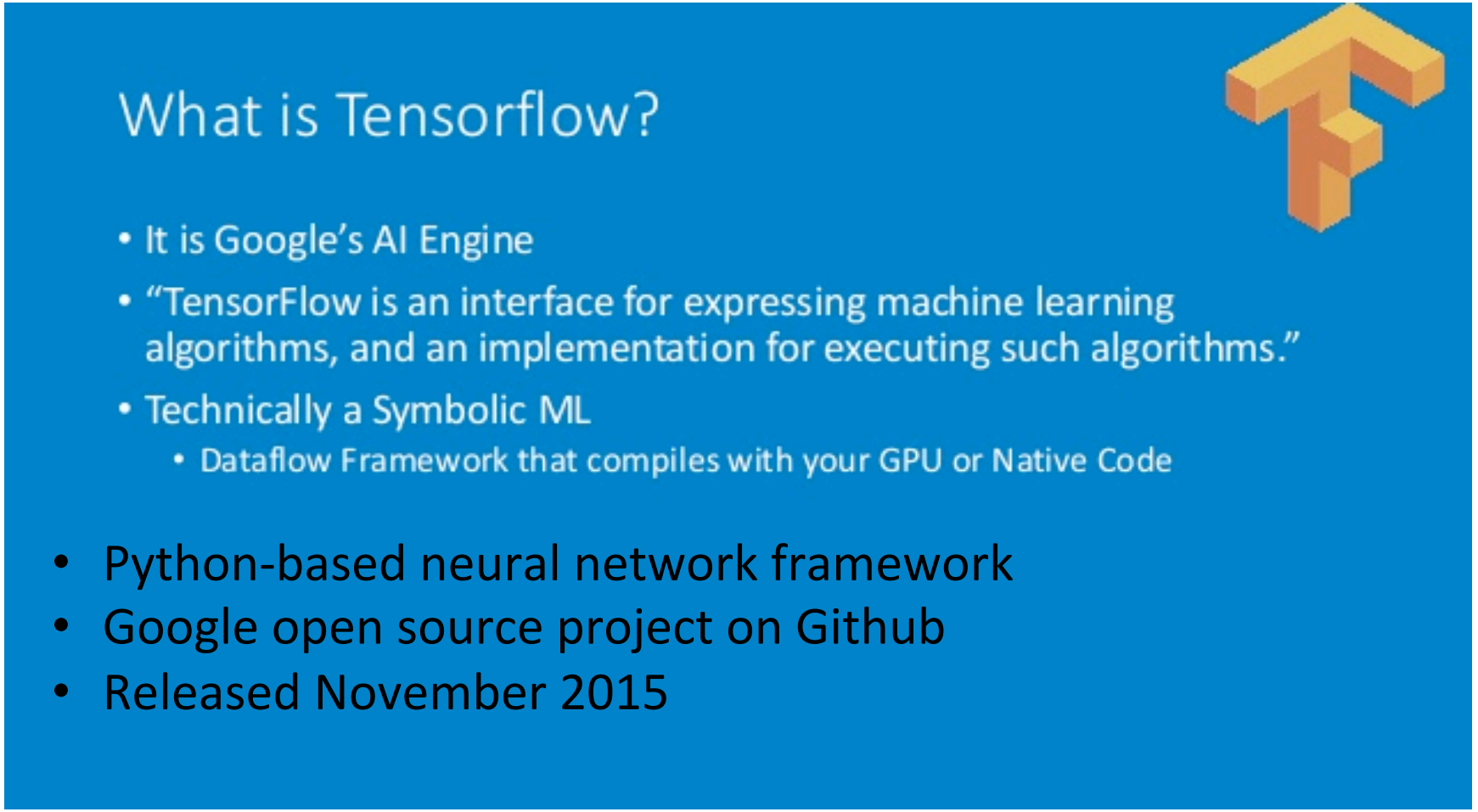
- Python-based neural network framework
- Google open source project on Github
- Released November 2015

TensorFlow: open source software library for numerical computation using data flow graphs

- Nodes in the graph represent mathematical operations
- Edges represent the multidimensional data arrays/tensors that flow between



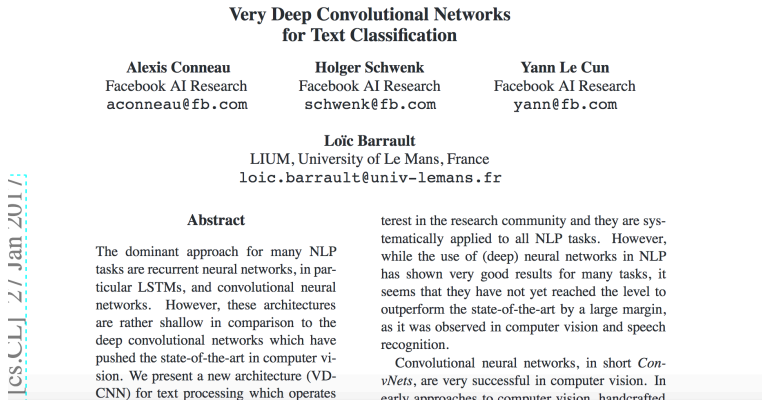
Tensorflow - Google



- Python-based neural network framework
- Google open source project on Github
- Released November 2015

Convolutional Neural Network

- Convolutional Neural Network (CNNs)
  - Traditional applied to computer vision.
- More recently CNNs to problems in Natural Language Processing (NLP)



Very Deep Convolutional Networks for Text Classification

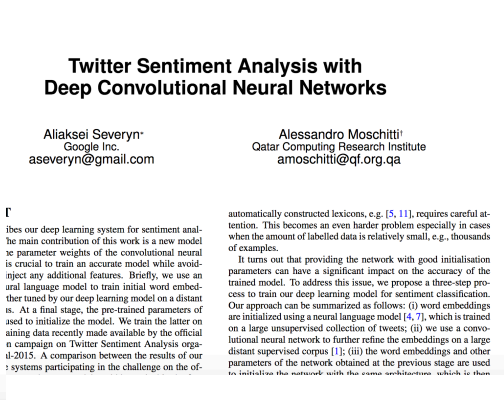
Alexis Conneau, Facebook AI Research, aconneau@fb.com
Holger Schwenk, Facebook AI Research, schwenk@fb.com
Yann Le Cun, Facebook AI Research, yann@fb.com
Loïc Barrault, LIUM, University of Le Mans, France, loic.barrault@univ-lemans.fr

Abstract

The dominant approach for many NLP tasks are recurrent neural networks, in particular LSTMs, and convolutional neural networks. However, these architectures are rather shallow in comparison to the deep convolutional networks which have pushed the state-of-the-art in computer vision. We present a new architecture (VD-CNN) for text processing which operates

terest in the research community and they are systematically applied to all NLP tasks. However, while the use of (deep) neural networks in NLP has shown very good results for many tasks, it seems that they have not yet reached the level to outperform the state-of-the-art by a large margin, as it was observed in computer vision and speech recognition.

Convolutional neural networks, in short *ConvNets*, are very successful in computer vision. In

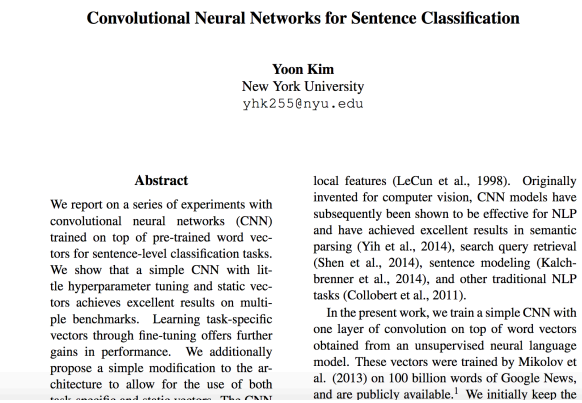


Twitter Sentiment Analysis with Deep Convolutional Neural Networks

Aliaksei Severyn, Google Inc., aseveryn@gmail.com
Alessandro Moschitti, Qatar Computing Research Institute, amoschitti@qf.org.qa

automatically constructed lexicons, e.g. [5, 11], requires careful attention. This becomes an even harder problem especially in cases when the amount of labelled data is relatively small, e.g., thousands of examples.

It turns out that providing the network with good initialisation parameters can have a significant impact on the accuracy of the trained model. To address this issue, we propose a three-step process to train our deep learning model for sentiment classification. Our approach can be summarized as follows: (i) word embeddings are initialized using a neural language model [4, 7], which is trained on a large unsupervised collection of tweets; (ii) we use a convolutional neural network to further refine the embeddings on a large distant supervised corpus [1]; (iii) the word embeddings and other parameters of the network obtained at the previous stage are used



Convolutional Neural Networks for Sentence Classification

Yoon Kim, New York University, yhk255@nyu.edu

Abstract

We report on a series of experiments with convolutional neural networks (CNN) trained on top of pre-trained word vectors for sentence-level classification tasks. We show that a simple CNN with little hyperparameter tuning and static vectors achieves excellent results on multiple benchmarks. Learning task-specific vectors through fine-tuning offers further gains in performance. We additionally propose a simple modification to the architecture to allow for the use of both

local features (LeCun et al., 1998). Originally invented for computer vision, CNN models have subsequently been shown to be effective for NLP and have achieved excellent results in semantic parsing (Yih et al., 2014), search query retrieval (Shen et al., 2014), sentence modeling (Kalchbrenner et al., 2014), and other traditional NLP tasks (Collobert et al., 2011).

In the present work, we train a simple CNN with one layer of convolution on top of word vectors obtained from an unsupervised neural language model. These vectors were trained by Mikolov et al. (2013) on 100 billion words of Google News, and are publicly available.[1] We initially keep the

Convolutional Neural Network

Kim (2014): Sentence classification

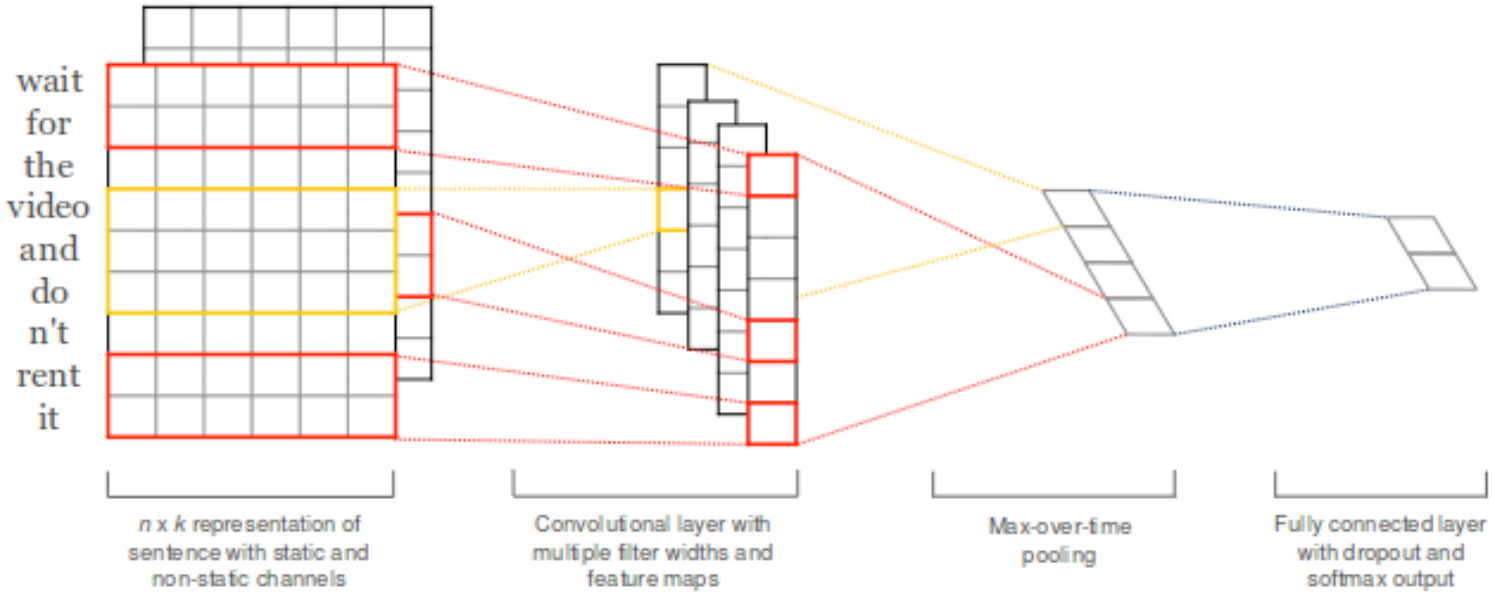


Figure 1: Model architecture with two channels for an example sentence.

Convolutional Neural Networks

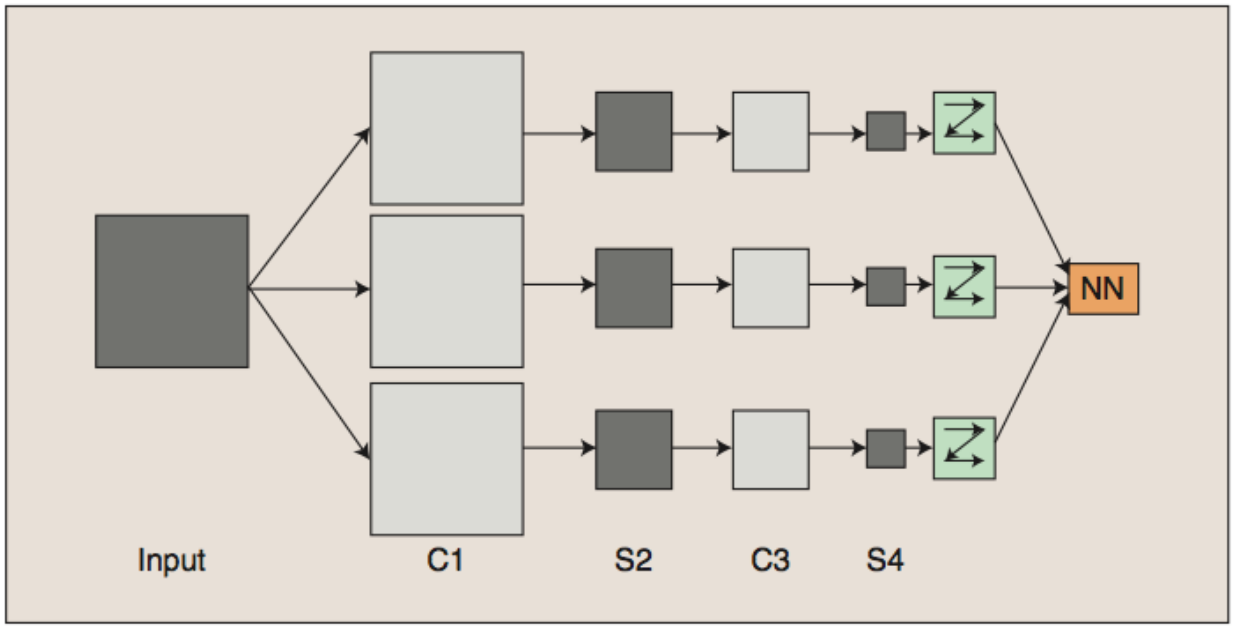
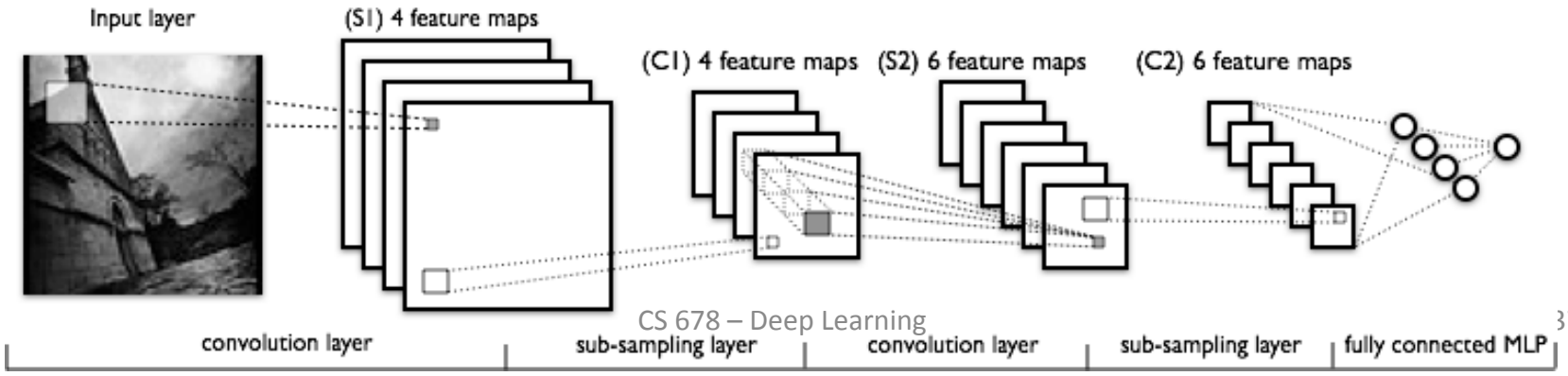


FIGURE 2 Conceptual example of convolutional neural network. The input image is convolved with three trainable filters and biases as in Figure 1 to produce three feature maps at the C1 level. Each group of four pixels in the feature maps are added, weighted, combined with a bias, and passed through a sigmoid function to produce the three feature maps at S2. These are again filtered to produce the C3 level. The hierarchy then produces S4 in a manner analogous to S2. Finally these pixel values are rasterized and presented as a single vector input to the "conventional" neural network at the output.

C layers are convolutions, S layers pool/sample

Often starts with fairly raw features at initial input and lets CNN discover improved feature layer for final supervised learner – eg. MLP/BP



CNN + TensorFlow

Convolutional Neural Networks for Sentence Classification

Yoon Kim
New York University
yhk255@nyu.edu

Abstract

We report on a series of experiments with convolutional neural networks (CNN) trained on top of pre-trained word vectors for sentence-level classification tasks. We show that a simple CNN with little hyperparameter tuning and static vectors achieves excellent results on multiple benchmarks. Learning task-specific vectors through fine-tuning offers further gains in performance. We additionally propose a simple modification to the architecture to allow for the use of both

local features (LeCun et al., 1998). Originally invented for computer vision, CNN models have subsequently been shown to be effective for NLP and have achieved excellent results in semantic parsing (Yih et al., 2014), search query retrieval (Shen et al., 2014), sentence modeling (Kalchbrenner et al., 2014), and other traditional NLP tasks (Collobert et al., 2011).

In the present work, we train a simple CNN with one layer of convolution on top of word vectors obtained from an unsupervised neural language model. These vectors were trained by Mikolov et al. (2013) on 100 billion words of Google News, and are publicly available.[1] We initially keep the

HOME • DEEP LEARNING NEWSLETT

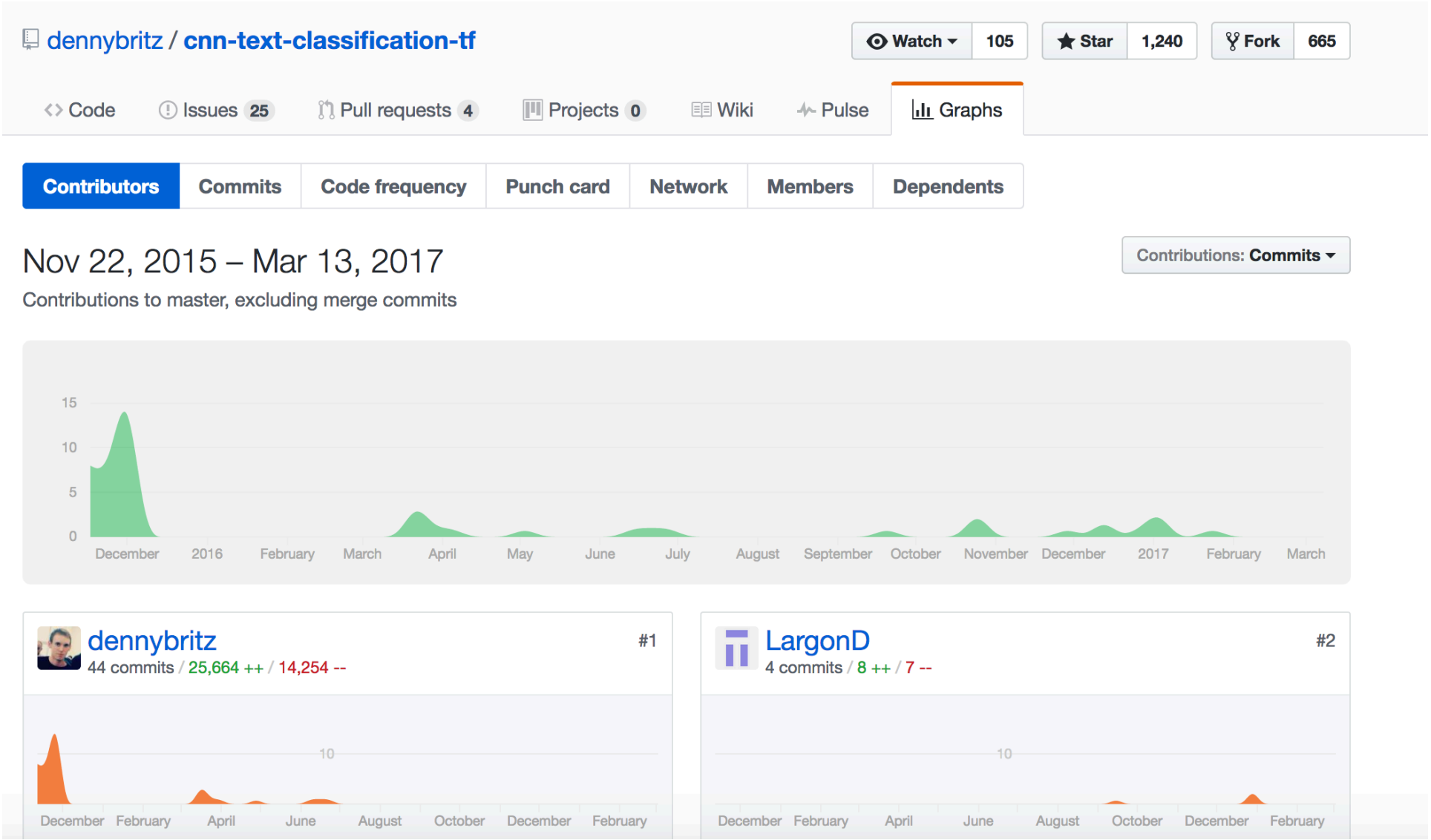
IMPLEMENTING A CNN FOR TE TENSORFLOW

nber 11, 2015

The full code is available on Github.

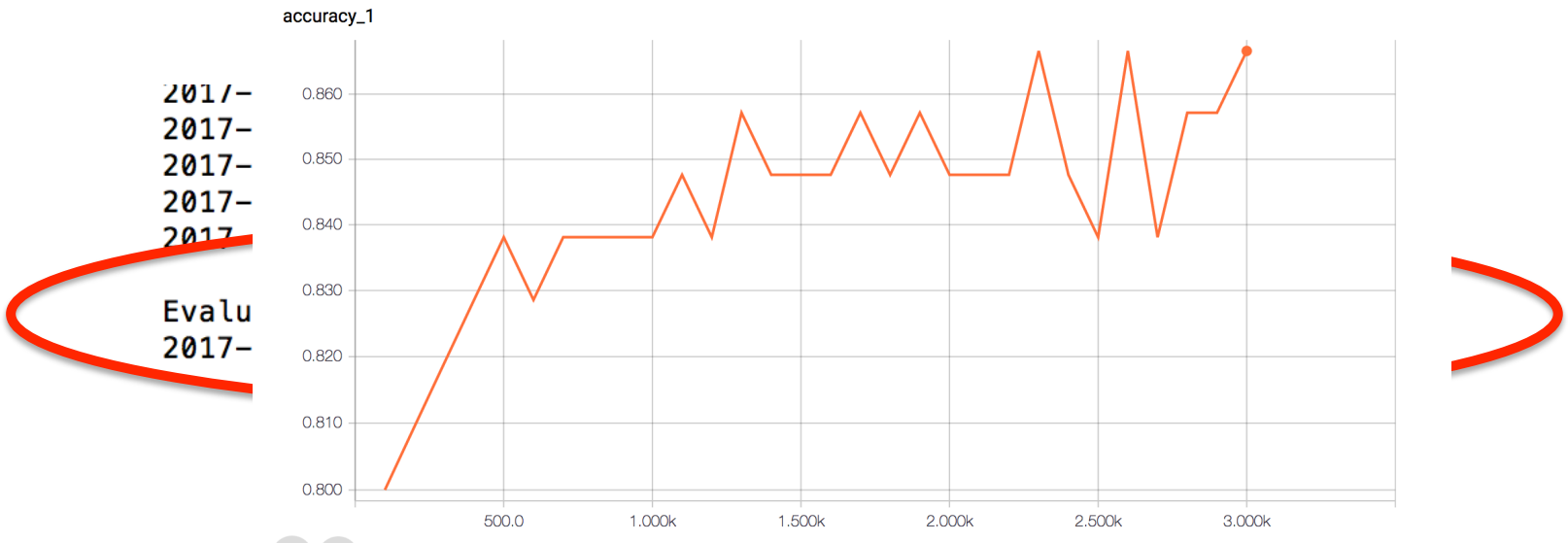
In this post we will implement a model similar to Kim Yoon's Convolutional Neural Networks for Sentence Classification. The model presented in the paper achieves good classification performance across a range of text

cnn-text-classification-tf



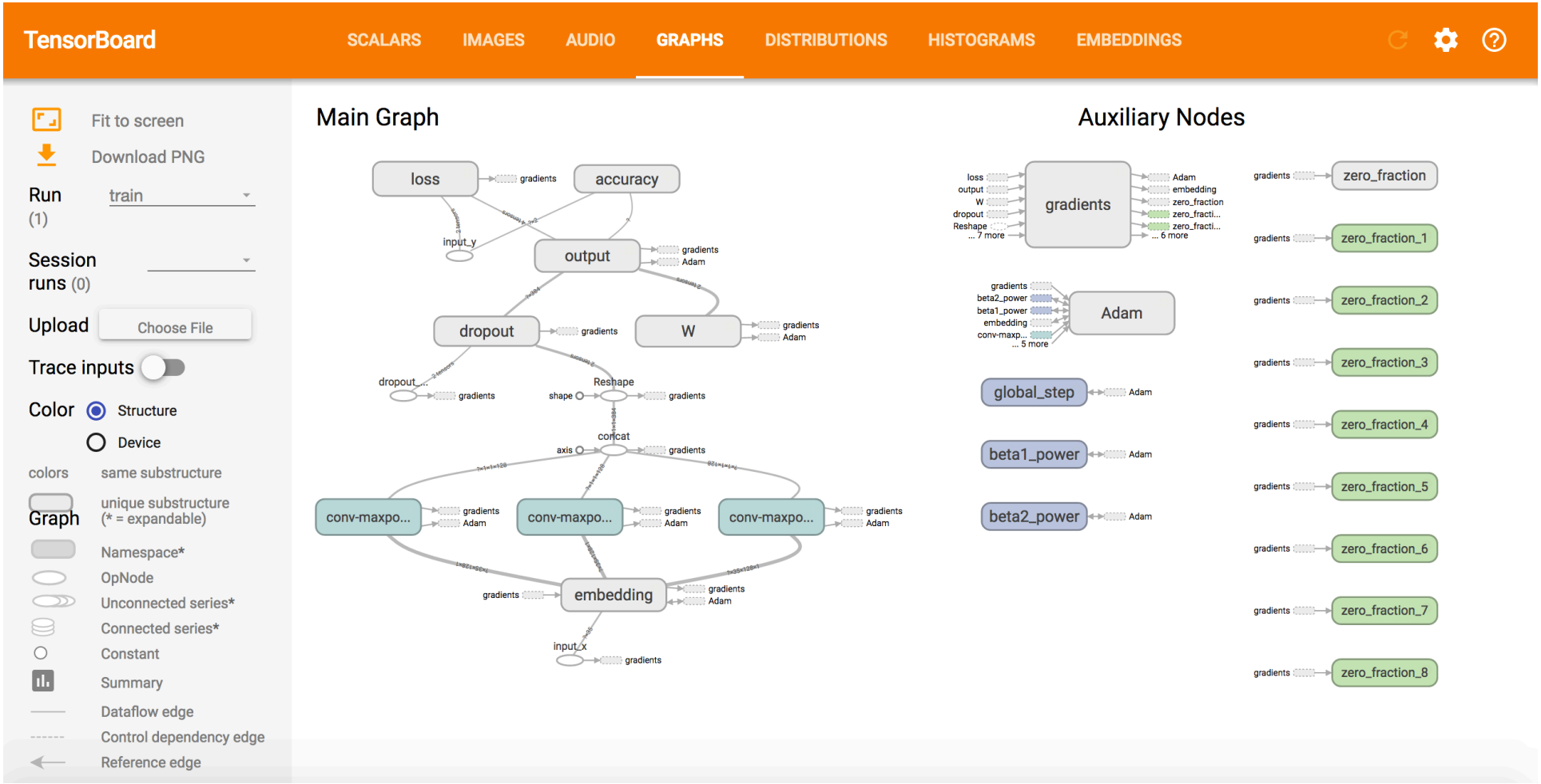
Methodology taken

- Modification of the original CNN code for working with our data
- Train data and build CNN classifier



- Run the CNN with non-labeled data

TensorBoard



CNN vs LR

	A	B	C	D	E	F	G
1	**Tweet**	**Logistic_Regressio**	**Convolutional_Neural_Network**	**Difference**	**Total_Eq**	**Perecentage**	
2	@AngelicaAnna40 nope): we're going to Aurora !!	0	1	-1	2626	65.2995277	
3	@AngelicaAnna40 nope): we're going to Aurora !!	0	1	-1	**Only_LR_Positive**		
4	Give Mom the Aurora Tonight / Mercury Transit Update: Si	1	1	0	342		
5	wag kang bastos				**Only_CNN_Positive**		
6	bibigyan kita ng dos				417		
7	mamayang alas dos						
8	gaya gaya						
9	putumaya				**Total_Aurora_LR**	1382	
10	sapang palay				**Total_Other_LR**	2002	
11	aurora						
12	cubao"	0	0	0			
13	Seen the northern lights last night, sooooo beautiful ðŸ˜³ð	1	1	0	**Total_Aurora_CNN**	1457	
14	Camping under the Northern Lights in Norway https://t.co/	1	1	0	**Total_Other_CNN**	1927	
15	quero ter uma filha pra chamÃ¡-la de Aurora	0	0	0			
16	Aurora Made An Appearance Last Night https://t.co/xDQV	1	1	0			
17	Something good amidst the politics. Benham Rise: PH's nev	0	0	0			
18	PAG NAMAN KINUHA PA 'TO NG CHINA EWAN KO NA						
19	Benham Rise: PH's new territory off Aurora https://t.co/ufl	0	0	0			
20	Oh what a beauty! https://t.co/911tI7xT4N	0	0	0			
21	Aurora Afx Model Motoring Pit Row Special-ii Electric Ho R	0	0	0			
22	AURORA PRIZE CEREMONY 2016 STREAM https://t.co/E7G	0	0	0			
23	@ChrisConwayBC camera shake notwithstanding, a good s	0	0	0			
24	@chris_greenf @EmmsStarGaze @TheAstroCamp Suppose	1	1	0			
25	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
26	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
27	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
28	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
29	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
30	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
31	Northern lights from last night just south of Southey. #stor	1	1	0			
32	Aurora's house https://t.co/Ic5EaUKl7J	0	0	0			
33	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
34	The Aurora Australis on the South Island of New Zealand. T	1	1	0			
35	The Aurora Australis on the South Island of New Zealand. T	1	1	0			

Deep learning vs SVM

- Deep learning is a way of having **several transformations in a row**, to combine layers and layers of features.
  - SVMs typically only allow a **single transformation**.
- Neural networks allows dozens (to hundreds in the latest papers) of layers.
  - Kernel SVMs are like a **2 layer neural network**.