2019 Hands-On Workshop

Put Deep Learning to Work:
A Practical Introduction on AWS

Wenming Ye,
Sr. Solution Architect

+

Miro Enev,
Sr. Solution Architect

AGENDA

KDD 2019

1. **Account Setup**
2. **Intro to AWS Services** [SageMaker, Rekognition, Comprehend]
3. **Intro to DL** [DL Architectures, Demo Links, Types of Learning]
4. **Intro to hands-on-labs**
    1. Object Detection, Single Shot Detector (SSD)
    2. GluonNLP: BERT fine-tuning for sentiment analysis
    3. Anomaly Detection in Time-Series Data

Logistics

- **Get** Lab Instructions: https://bit.ly/2YsxhuH

- **Create** an AWS Account
  - **Add** Credit Security Code to your Account

- **Join** Chime:
- **Paste** account number into Chime Window
  - **Wait** for confirmation for white listing

- **Ask** TA for help!
  - Leo, Haibing, Zha, Wen-ming, Ginni

AWS Services

All services

Compute
- EC2
- Lightsail
- ECR
- ECS
- EKS
- Lambda
- Batch
- Elastic Beanstalk
- Serverless Application Repository

Storage
- S3
- EFS
- FSx
- S3 Glacier
- Storage Gateway
- AWS Backup

Database
- RDS
- DynamoDB
- ElastiCache
- Neptune
- Amazon Redshift
- Amazon QLDB
- Amazon DocumentDB

Migration & Transfer
- AWS Migration Hub
- Application Discovery Service
- Database Migration Service
- Server Migration Service
- AWS Transfer for SFTP
- Snowball
- DataSync

Networking & Content Delivery
- VPC
- CloudFront
- Route 53
- API Gateway
- Direct Connect
- AWS App Mesh
- AWS Cloud Map
- Global Accelerator

Developer Tools
- CodeStar
- CodeCommit
- CodeBuild
- CodeDeploy
- CodePipeline
- Cloud9
- X-Ray

Robotics
- AWS RoboMaker

Blockchain
- Amazon Managed Blockchain

Satellite
- Ground Station

Management & Governance
- AWS Organizations
- CloudWatch
- AWS Auto Scaling
- CloudFormation
- CloudTrail
- Config
- OpsWorks
- Service Catalog
- Systems Manager
- Trusted Advisor
- Managed Services
- Control Tower
- AWS License Manager
- AWS Well-Architected Tool
- Personal Health Dashboard
- AWS Chatbot

Media Services
- Elastic Transcoder
- Kinesis Video Streams
- MediaConnect
- MediaConvert
- MediaLive
- MediaPackage
- MediaStore
- MediaTailor

Machine Learning
- Amazon SageMaker
- Amazon Comprehend
- AWS DeepLens
- Amazon Lex
- Machine Learning
- Amazon Polly
- Rekognition
- Amazon Transcribe
- Amazon Translate
- Amazon Personalize
- Amazon Forecast
- Amazon Textract
- AWS DeepRacer

Analytics
- Athena
- EMR
- CloudSearch
- Elasticsearch Service
- Kinesis
- QuickSight
- Data Pipeline
- AWS Glue
- MSK

Security, Identity, & Compliance
- IAM
- Resource Access Manager
- Cognito
- Secrets Manager
- GuardDuty
- Inspector
- Amazon Macie
- AWS Single Sign-On
- Certificate Manager
- Key Management Service
- CloudHSM
- Directory Service
- WAF & Shield
- Artifact
- Security Hub

AWS Cost Management
- AWS Cost Explorer
- AWS Budgets
- AWS Marketplace Subscriptions

Mobile
- AWS Amplify
- Mobile Hub
- AWS AppSync
- Device Farm

AR & VR
- Amazon Sumerian

Application Integration
- Step Functions
- Amazon EventBridge
- Amazon MQ
- Simple Notification Service
- Simple Queue Service
- SWF

Customer Engagement
- Amazon Connect
- Pinpoint
- Simple Email Service

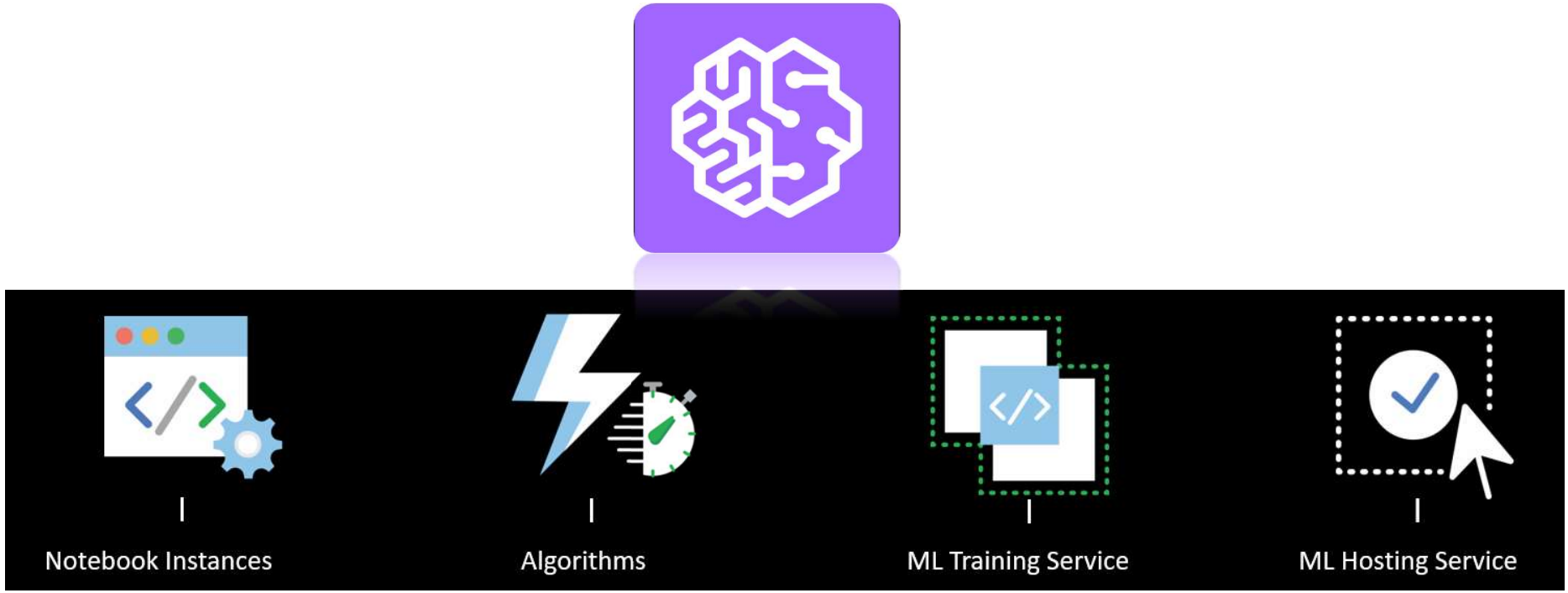
Business Applications
- Alexa for Business
- Amazon Chime
- WorkMail

End User Computing
- WorkSpaces
- AppStream 2.0
- WorkDocs
- WorkLink

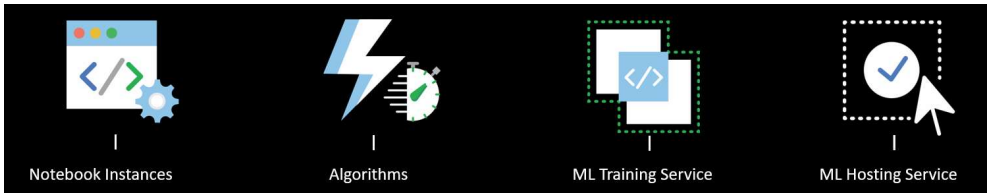
Internet of Things
- IoT Core
- Amazon FreeRTOS
- IoT 1-Click
- IoT Analytics
- IoT Device Defender
- IoT Device Management
- IoT Events
- IoT Greengrass
- IoT SiteWise
- IoT Things Graph

Game Development
- Amazon GameLift

AWS SageMaker



AWS SageMaker Algortithms



All Categories
Infrastructure Software (33)
DevOps (33)
Business Applications (2)
Machine Learning (243)
IoT (10)
Industries (45)
Filters Clear all filters
Vendors
Perception Health (31)
RocketML (23)
Sensifai (18)
Cloudwick (11)
Figure Eight (11)
Persistent Systems (10)
Twinword Inc. (8)
Modjout (8)
Intel® AI (8)
Amazon Web Services (8)
Show more
Software Pricing Plans
Free (75)
Hourly (168)
Software Free Trial
Free Trial (118)
Delivery Method
Clear
Amazon SageMaker (243)
Resource Type
Clear
Algorithm (62)
Model Package (181)
Region
US East (N. Virginia) (229)
US East (Ohio) (243)
US West (Oregon) (236)
US West (N. California) (229)
EU (Frankfurt) (211)
Show more

Text Similarity Analyzer
(0) | Version v1 | Sold by TIBCO Software Inc.
Engineers word/document features on a corpus with NLP methods, and uses these features to compare new text to the corpus.

Autoencoder for Anomaly Detection
(0) | Version v1 | Sold by TIBCO Software Inc.
Identifies potential anomalies from transaction and or sensor data with a deep learning autoencoder.

Hospital Readmission
(0) | Version v1 | Sold by TIBCO Software Inc.
Predicts hospital readmission rates from DRG codes, billing and EMR data.

GluonNLP Sentence Generator
(1) | Version 1.0 | Sold by Amazon Web Services
Pre-trained sequence sampler for sentence generation, powered by GluonNLP.

GluonNLP English to German Translation
(0) | Version 1.0 | Sold by Amazon Web Services
Model for english to german translation, powered by GluonNLP

GluonCV SSD Object Detector
(0) | Version 1.1 | Sold by Amazon Web Services
Single Shot MultiBox Detector (SSD) is a powerful network for fast and accurate object detection, powered by GluonCV.

GluonCV Faster-RCNN Object Detector
(0) | Version 1.1 | Sold by Amazon Web Services
Faster RCNN is a powerful network for accurate object detection, powered by GluonCV.

GluonCV ResNet50 Classifier
(0) | Version 1.0 | Sold by Amazon Web Services
Image feature extraction and ImageNet category prediction using ResNet50-v1d network, provided by GluonCV

GluonCV DeepLab Semantic Segmentation
(0) | Version 1.0 | Sold by Amazon Web Services
DeepLab is a powerful model for image semantic segmentation, powered by GluonCV.

GluonCV YOLOv3 Object Detector
(1) | Version 1.1 | Sold by Amazon Web Services
YOLOv3 is a powerful network for fast and accurate object detection, powered by GluonCV.

showing 1 - 10 1 2 3 4 5 ... 25

All Categories
Infrastructure Software (29)
DevOps (1)
Machine Learning (62)
Industries (2)
Filters Clear all filters
Vendors
RocketML (15)
Sensifai (11)
Intel® AI (7)
Outpace Systems (6)
Peak (5)
improve.ai (4)
H2O.ai (4)
TIBCO Software Inc. (3)
Dimensional Mechanics (2)
bigfinite (1)
Show more
Software Pricing Plans
Free (23)
Hourly (39)
Software Free Trial
Free Trial (32)
Delivery Method
Clear
Amazon SageMaker (62)
Resource Type
Clear
Algorithm (62)
Model Package (181)
Region
US East (N. Virginia) (61)
US East (Ohio) (62)
US West (Oregon) (61)
US West (N. California) (61)
EU (Frankfurt) (61)
Show more

Text Similarity Analyzer
(0) | Version v1 | Sold by TIBCO Software Inc.
Engineers word/document features on a corpus with NLP methods, and uses these features to compare new text to the corpus.

Autoencoder for Anomaly Detection
(0) | Version v1 | Sold by TIBCO Software Inc.
Identifies potential anomalies from transaction and or sensor data with a deep learning autoencoder.

Hospital Readmission
(0) | Version v1 | Sold by TIBCO Software Inc.
Predicts hospital readmission rates from DRG codes, billing and EMR data.

Image Recognition (Trainable Algorithm)
(0) | Version v1 | Sold by Sensifai
Automatic Image Tagging and Recognition (Trainable Algorithm)

Demand Forecasting for Intermittent Data
(0) | Version 0.1 | Sold by Peak
An ensemble demand forecasting model, for intermittent data

Cognitive QnA
(0) | Version v1.0.0beta | Sold by Active.ai
provides an FAQ data source that you can query from your bot or application

Implicit BPR
(0) | Version 0.9.36 | Sold by Outpace Systems
A recommender system for implicit feedback datasets using Bayesian Personalized Ranking.

RocketML Text Latent Semantic Analysis
(0) | Version 0.1 | Sold by RocketML
Semantic analysis and Indexing on a list of documents

Facial Recognition Algorithm for Images
(0) | Version v1 | Sold by Sensifai
Face Recognition/Identification in Images

RocketML Sparse RForest Classification
(0) | Version 0.1 | Sold by RocketML
Classification on LibSVM data type using Random Forests

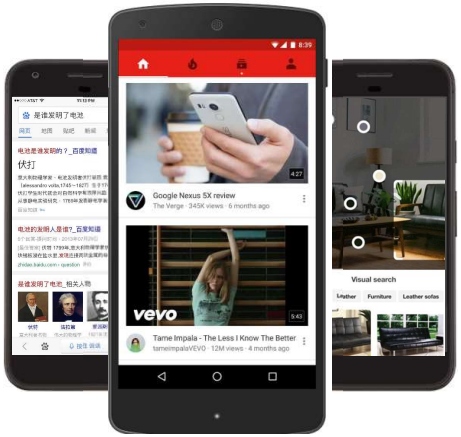
showing 1 - 10 1 2 3 4 5 6 7

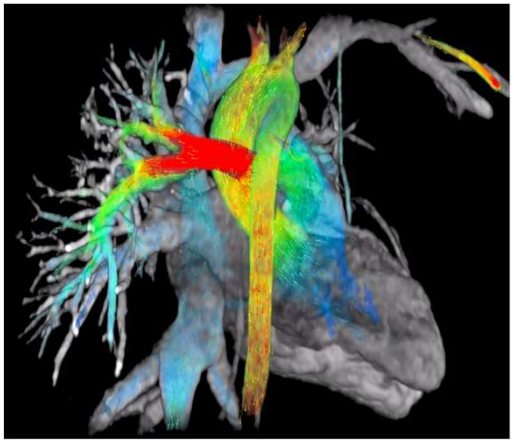
AWS ML/AI Services Console Walkthrough

Machine Learning

- Amazon SageMaker
- Amazon Comprehend
- AWS DeepLens
- Amazon Lex
- Machine Learning
- Amazon Polly
- Rekognition
- Amazon Transcribe
- Amazon Translate
- Amazon Personalize
- Amazon Forecast
- Amazon Textract
- AWS DeepRacer

Impact of ML/DL





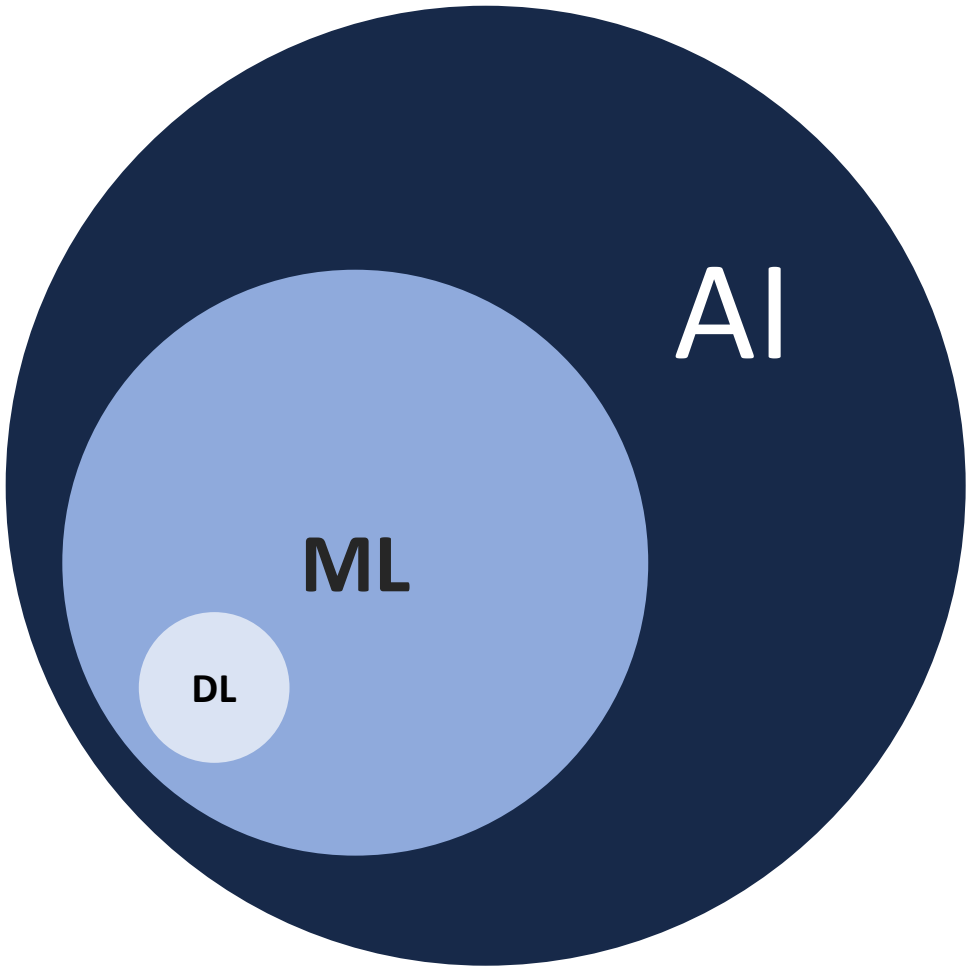
1 – Natural Language

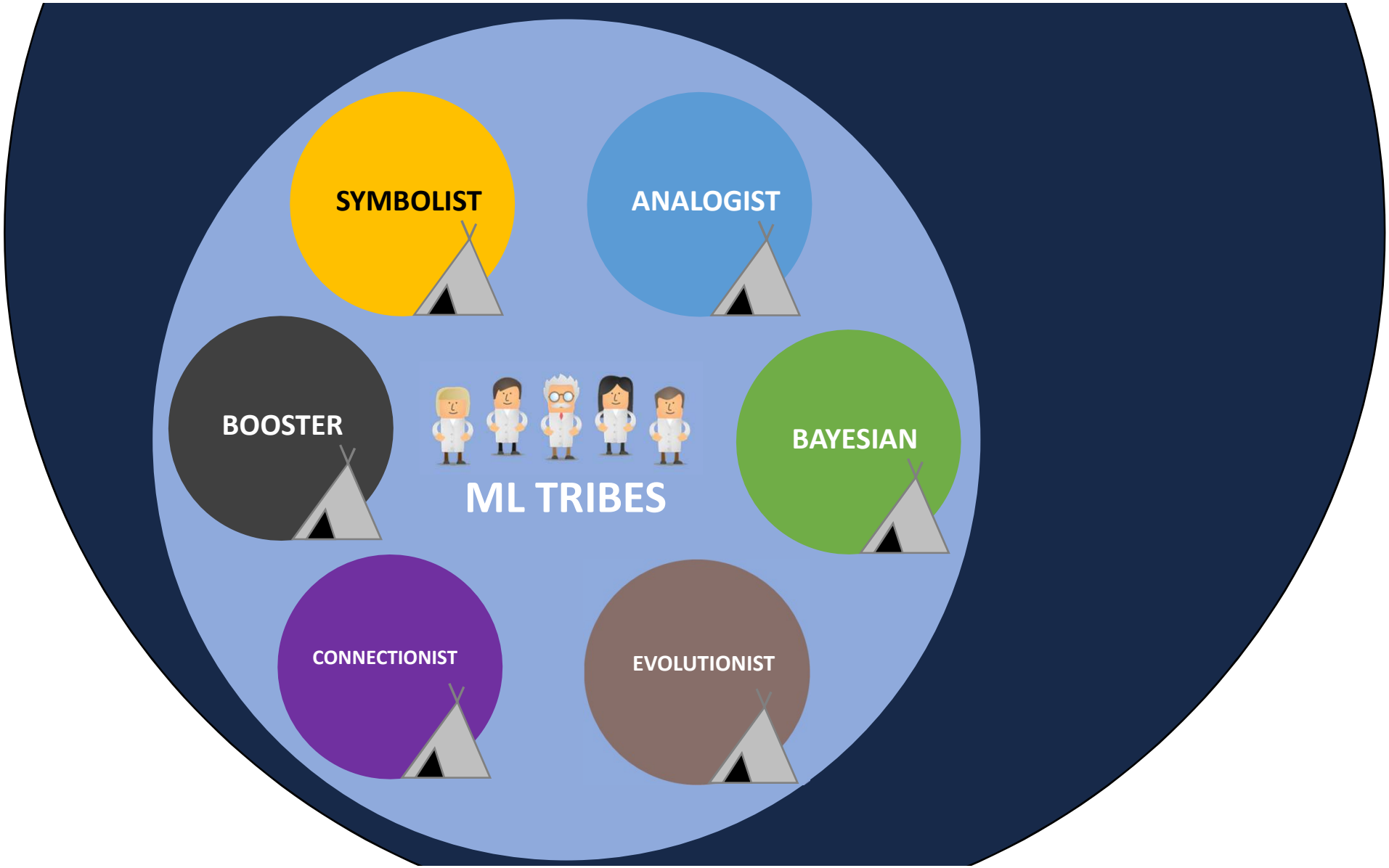
2 – Visual Perception / Robotics

3 – Anomaly Detection

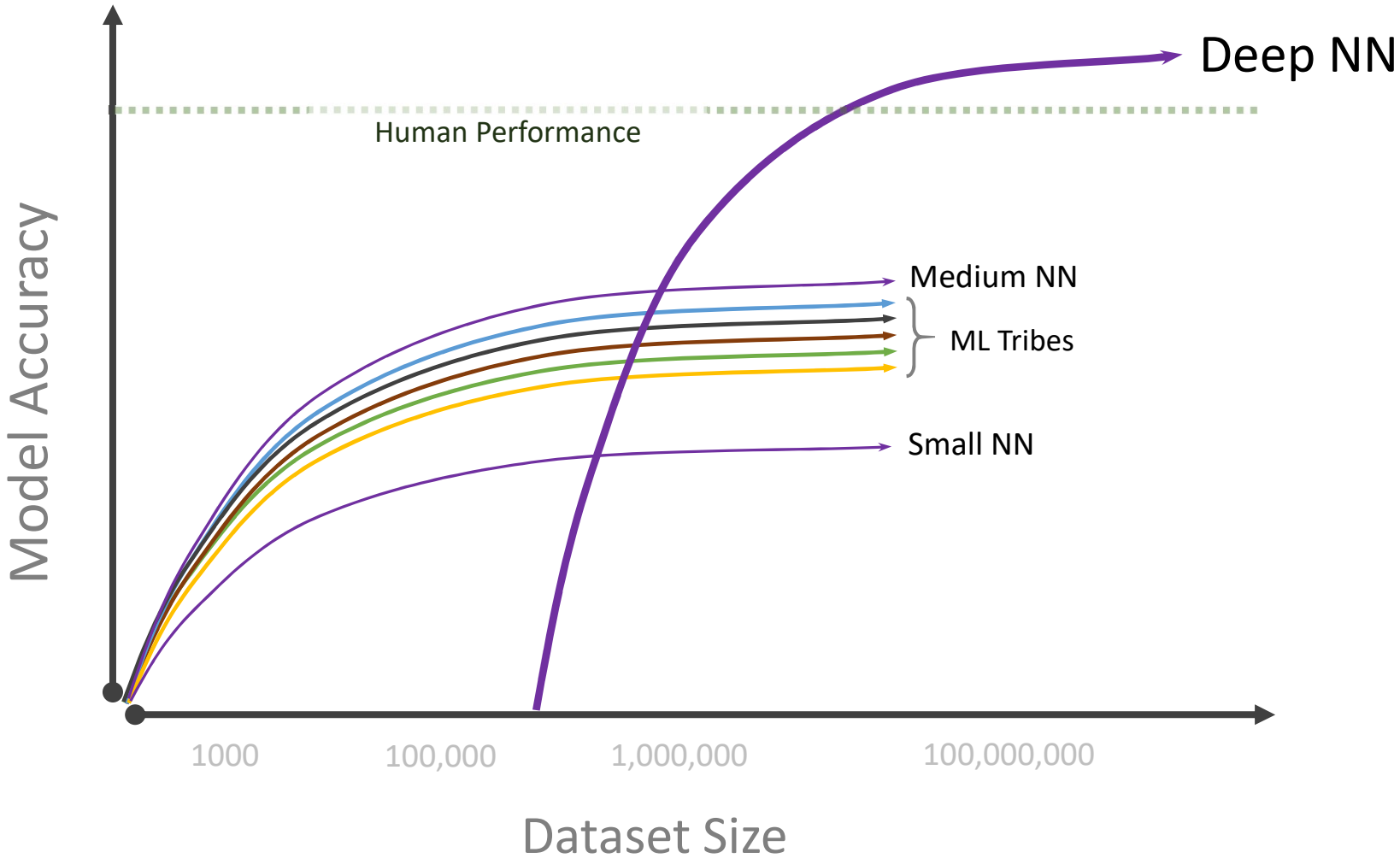
4 – Medicine [Radiology, Drug Discovery, Remote/Robo surgery]

DL in Context

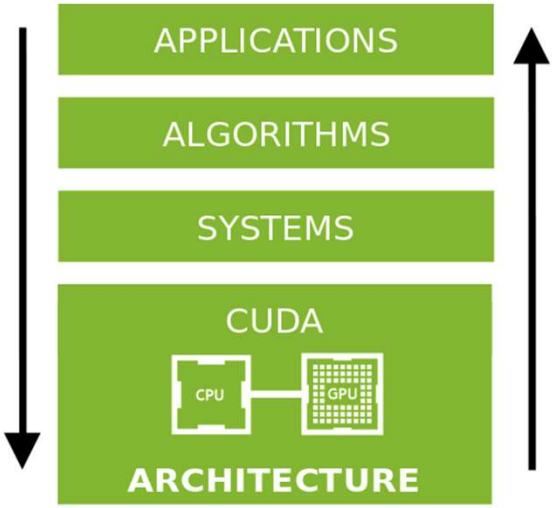


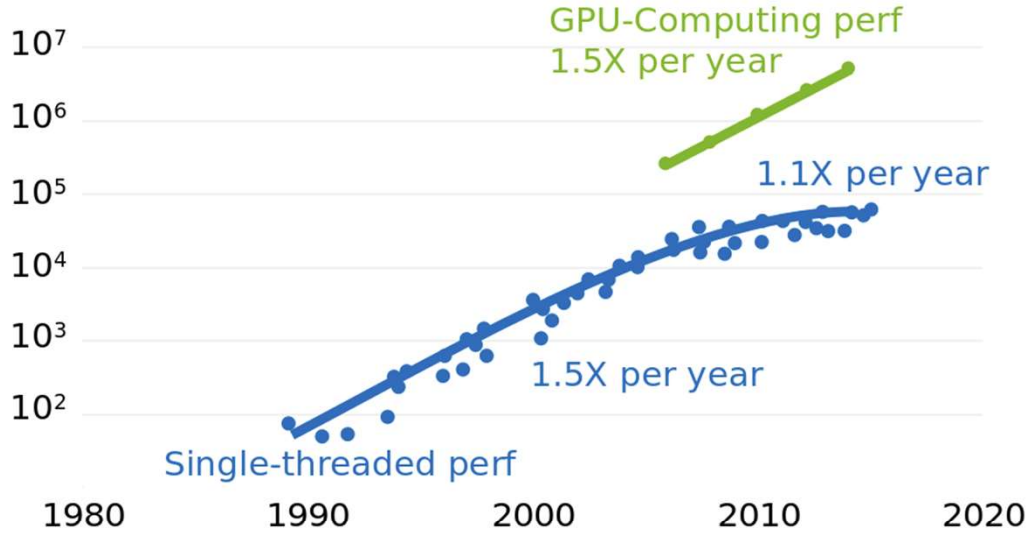

SYMBOLIST
ANALOGIST
BOOSTER
BAYESIAN
ML TRIBES
CONNECTIONIST
EVOLUTIONIST

Learning at Scale



GPU Computing Model

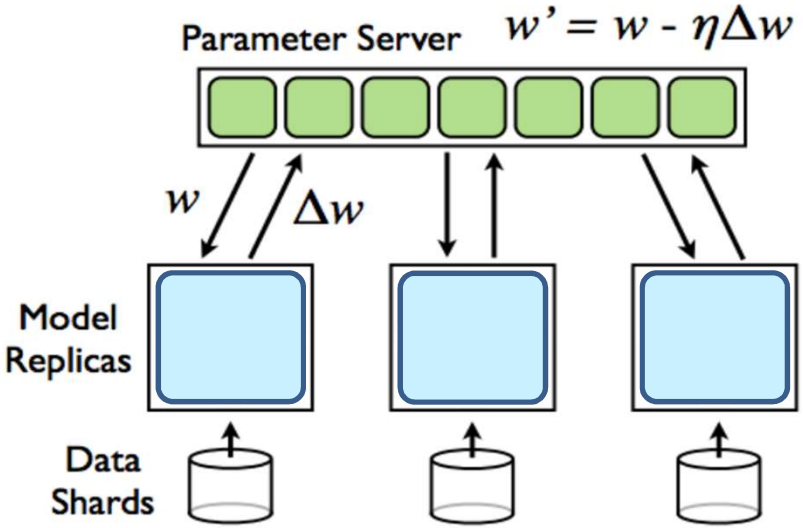




Original data up to the year 2010 collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond, and C. Batten New plot and data collected for 2010-2015 by K. Rupp

DL Compute = AWS + NVIDIA

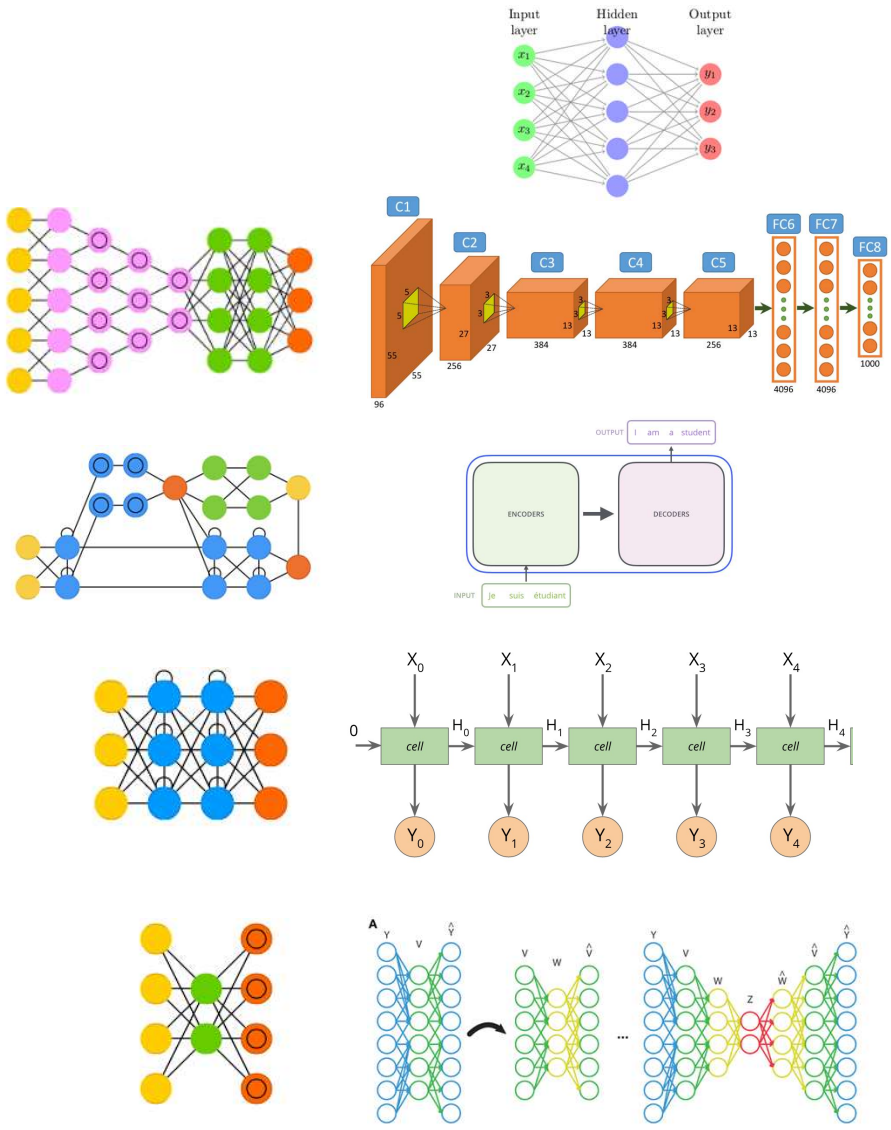
Data Parallel Training [SageMaker]



Volta100 [ec2.p3]

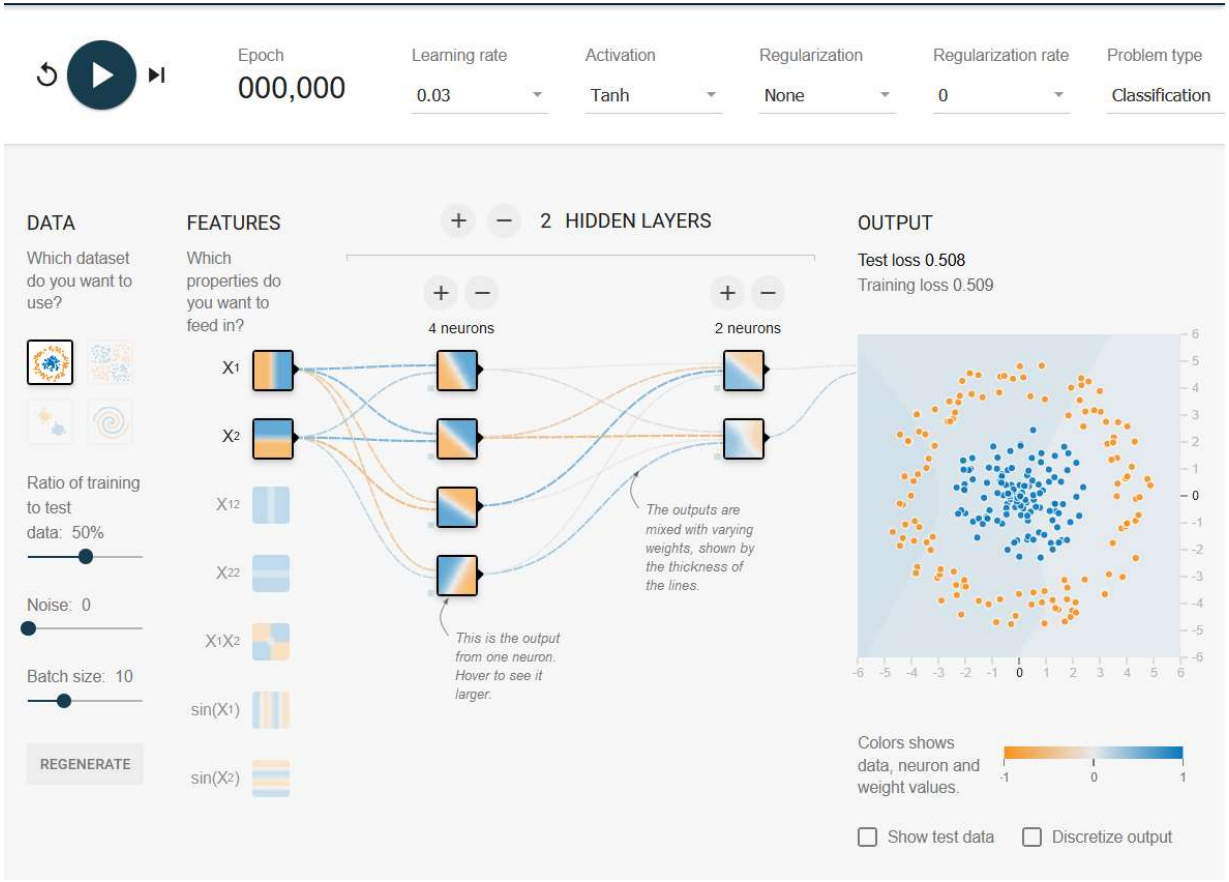
5,120 CUDA cores [640 Tensor cores]

Key DL Architectures



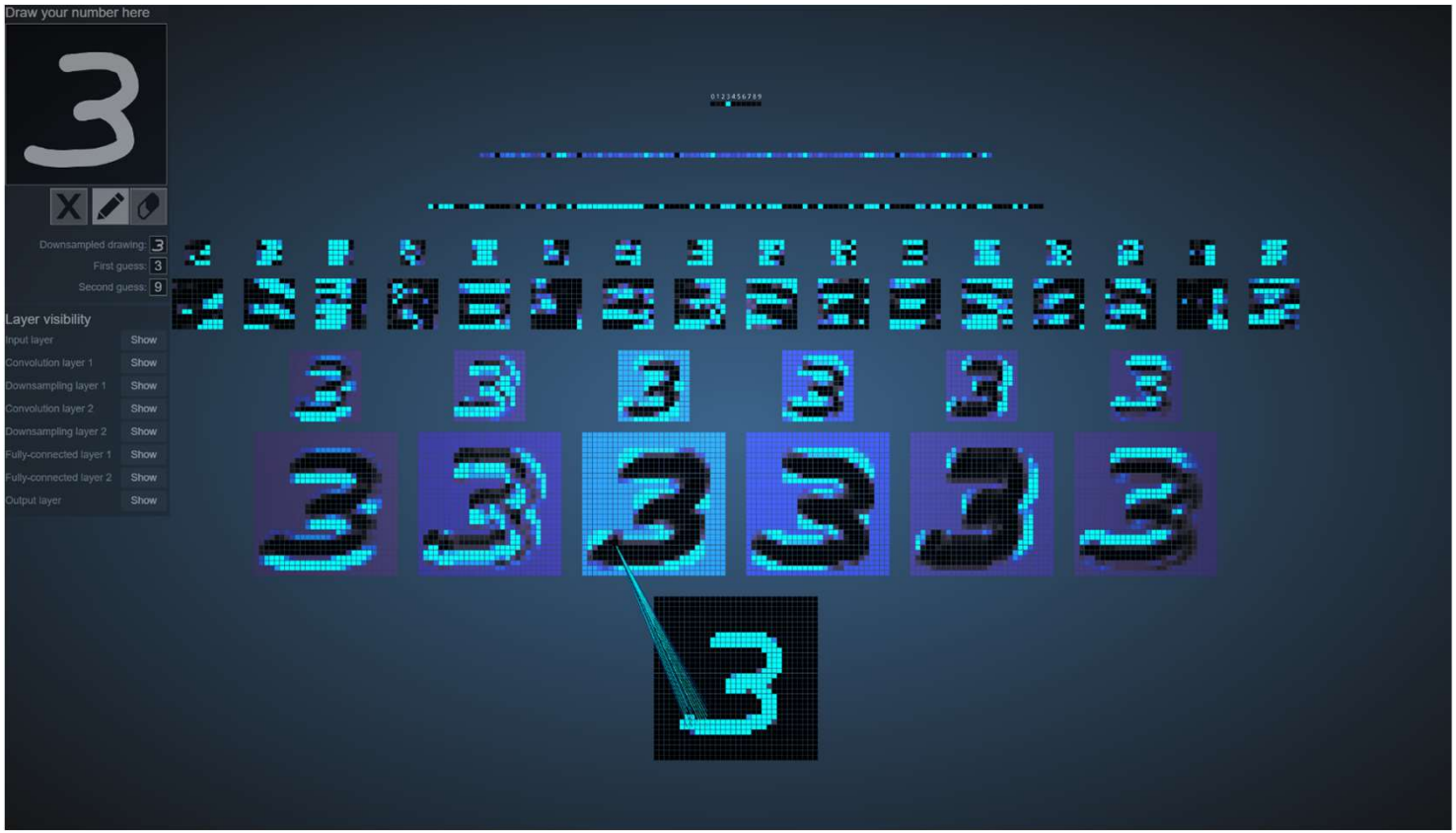
- MLP [demo]
- CNN **[Lab 1]** [demo]
  vision & speech perception
- Attention/Transformer **[Lab 2]**
- RNN
- Autoencoder **[Lab 3]**
  anomaly detection, model RL

MLP Demo



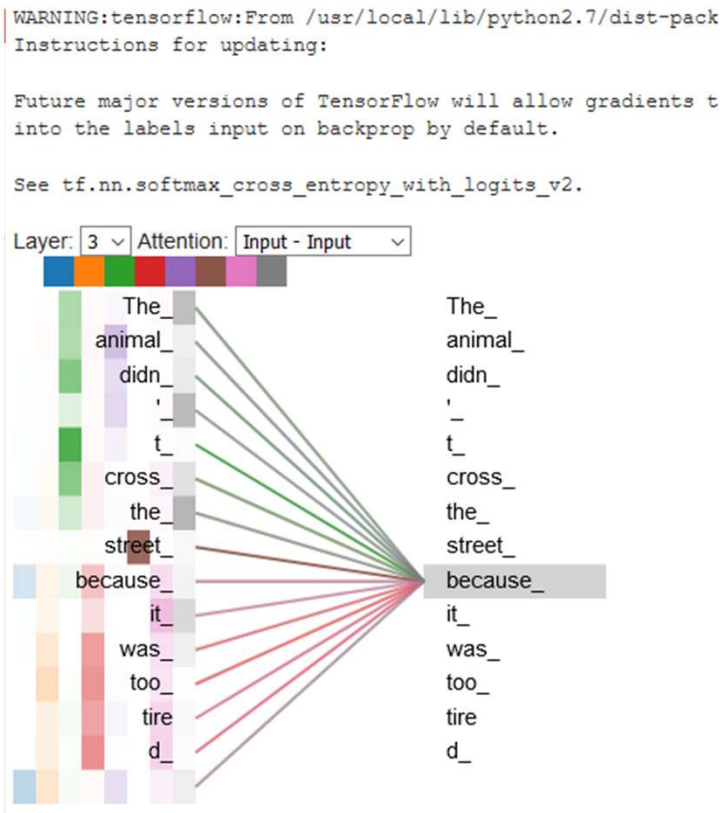
https://playground.tensorflow.org

CNN Demo



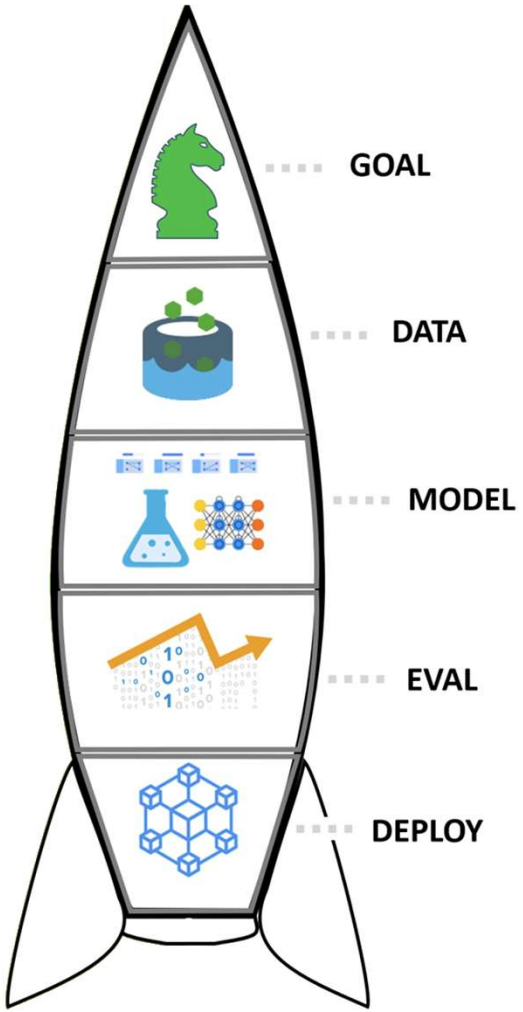
http://scs.ryerson.ca/~aharley/vis/

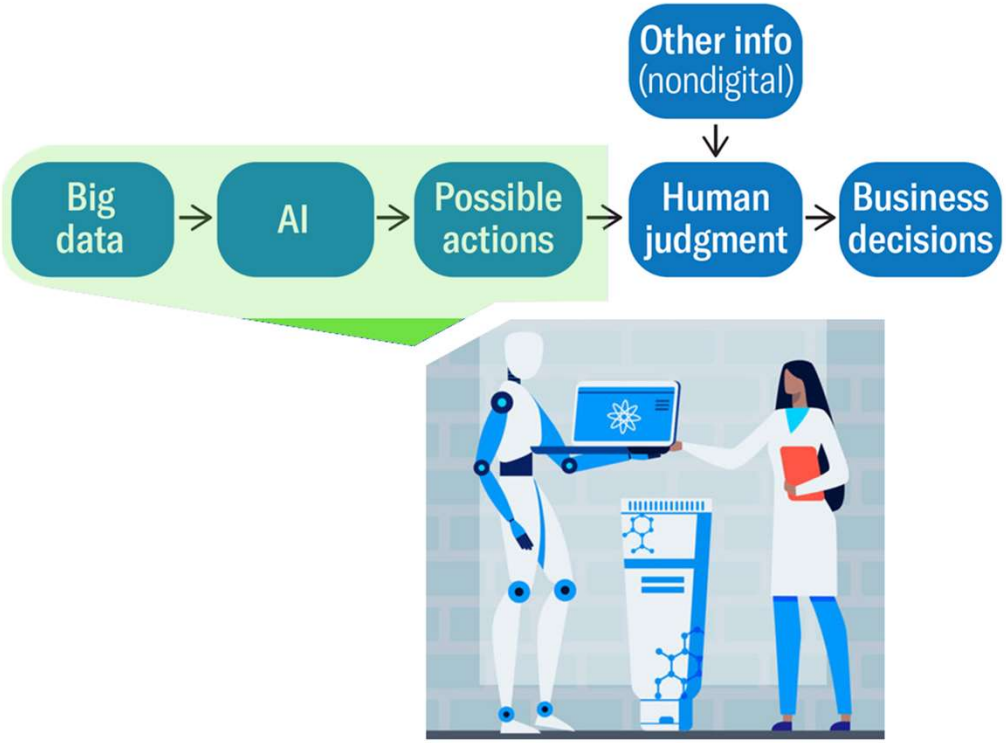
Transformer Demo



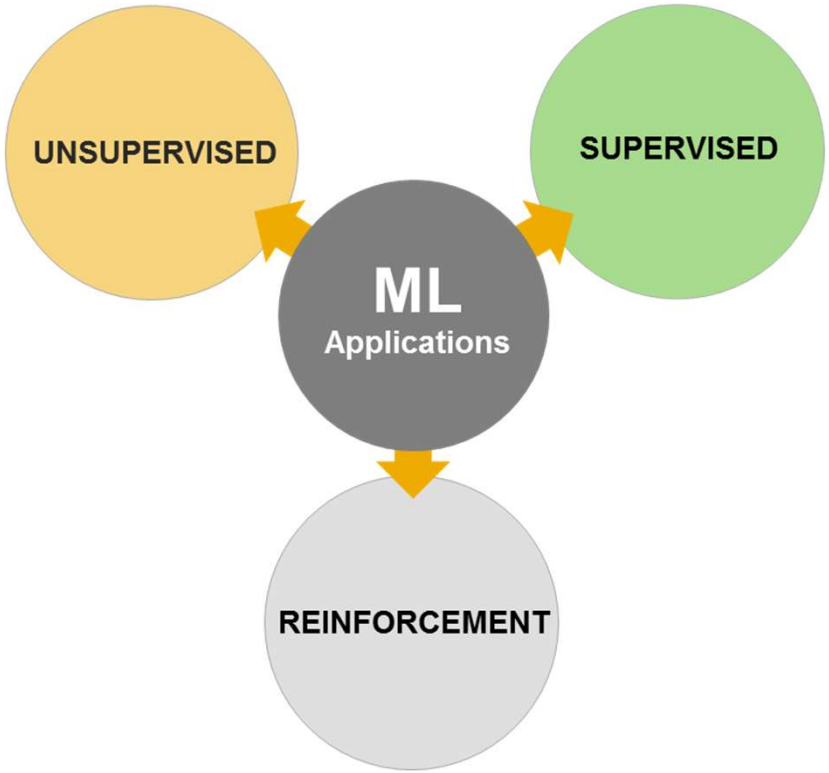
[English to German Transformer [Tensor2Tensor Demo]]

Applying ML


GOAL
DATA
MODEL
EVAL
DEPLOY

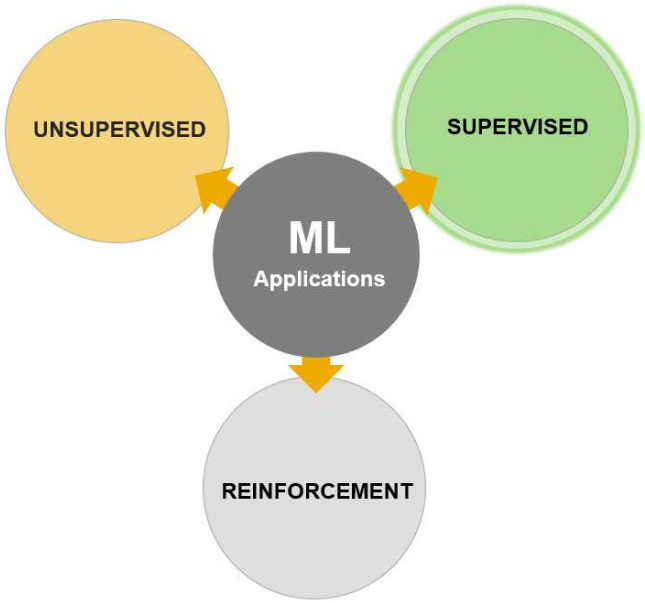

Other info
(nondigital)
Big data
AI
Possible actions
Human judgment
Business decisions

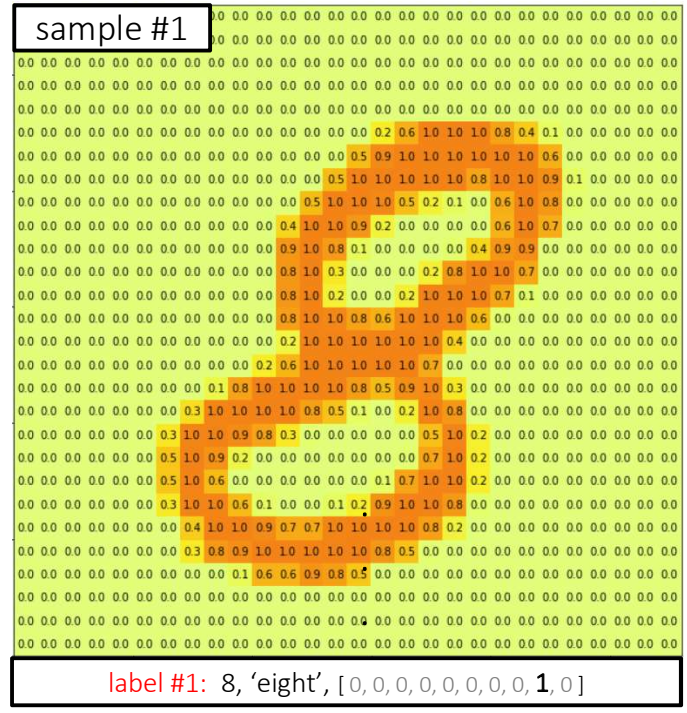
Types of Learning

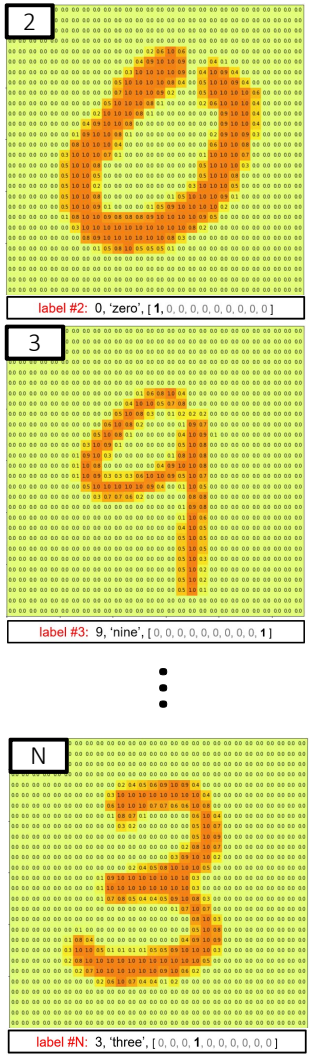


Supervised Learning

- ~100,000s of examples
  - Each example consists of
    - Data [Vector]
    - Label(s)

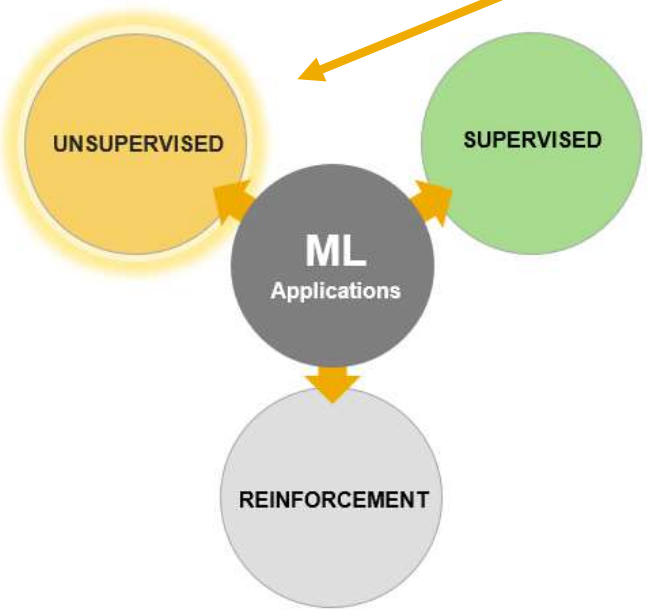




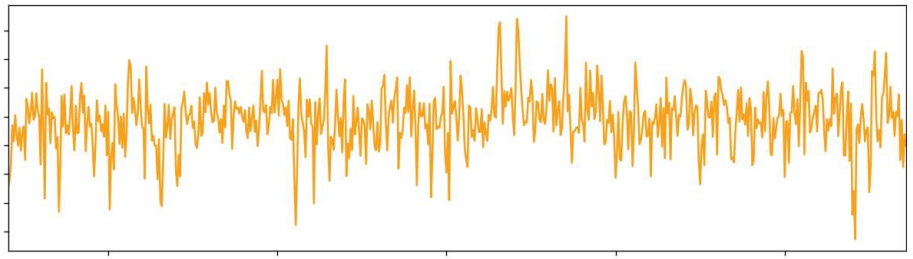


Unsupervised Learning

- In unsupervised learning, no labels are available
  - Typically larger datasets (millions)
  - Each example consists of
    - Data [Vector]
    - No Labels



Time Series Data



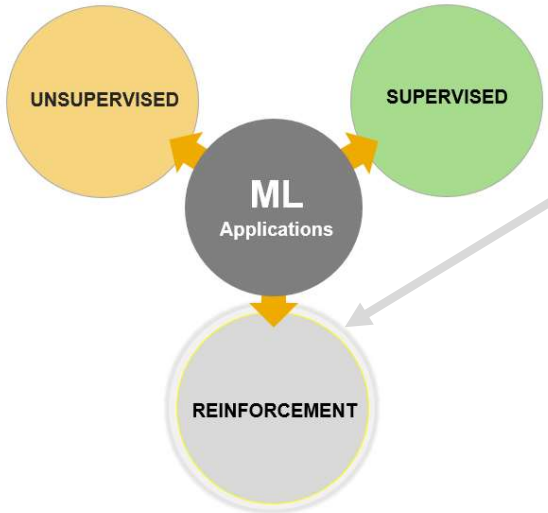
Gene Sequence Data

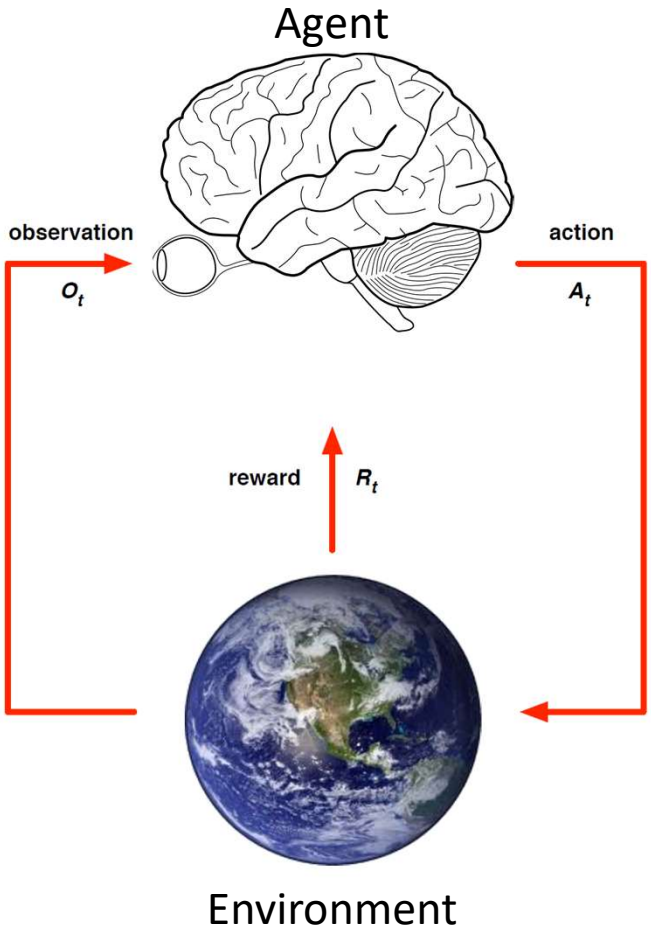
```
A G A T A A G A A G T G T T G T G G A A A A G T A T G T T G
T A T T T A G A A G T A A G T G A T G G T T A T T T G G A G
T A A G T A G G T A G A A T T A A G A G G G T T A T A G G G A
A A G G G A A T T G G G T G G A A G G T G A T T T A A T G A A
G A A T G T T T G T G A T T A G G A T A G T A T G T A A G
A T A G A G A T T G G A A T G G G G G G G A T G A T G A T T A
A T G A A G T T T A G G A G G T G T A T G A A T G A G A A A G
T T A G G G A A G G A T T A A T G T G T A A G G G T A T T A
G G A A A A A T G G A A A A A A T T A T G G G T T T T T G T G G
A G G G A G T G T A A A G T G G T G T G G A A A A A G A T T G
```

Text Data

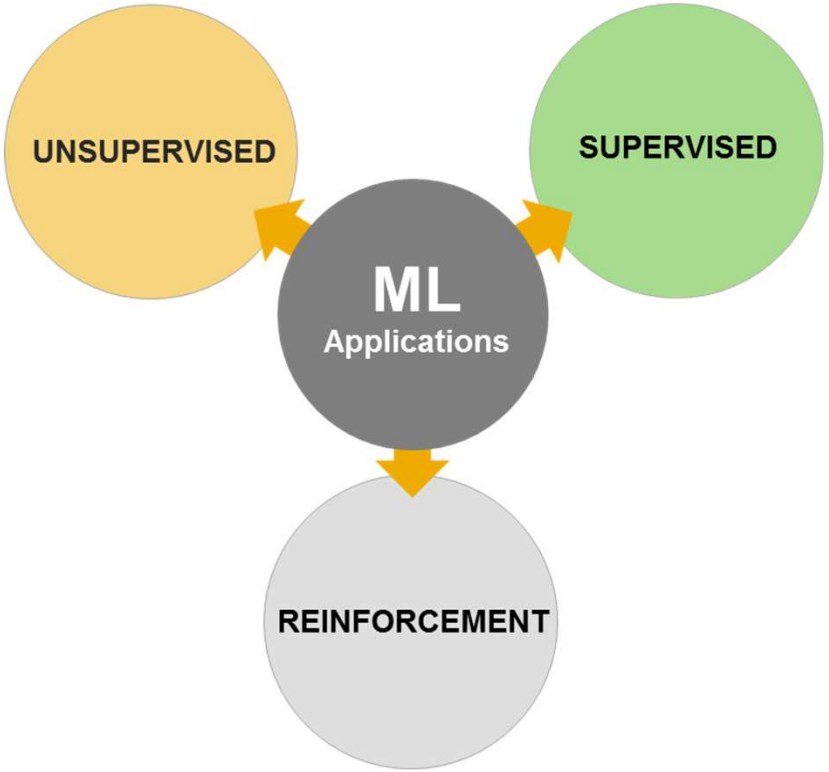
Reinforcement Learning

- **RL: Dataset built with experience**
  - Experience = Current Env. State + Action + Next Env. State
- **RLRL Feedback Loop**
  - At each step the **agent**
    - Executes action: A_t
    - Receives observation: O_t
    - Receives reward: R_t
  - The **environment**
    - Receives action: A_t
    - Emits observation: O_{t+1}
    - Emits reward: R_{t+1}





Three Types of Learning



Lab 1

Problem/Objective: Object Detection

AWS Service: Rekognition

DL Architecture Overview: CNN Demo [3D, flat]

Relevant Applications: Robotics, Retail, AV

SOTA : [paperswithcode.com/sota]

Lab 2

Problem/Objective: fine-tuning BERT for Question Answering

AWS Service: Comprehend

DL Architecture Overview: Illustrated BERT

Relevant Applications: Virtual Assistant/Chatbot

SOTA : [paperswithcode.com/sota]

Lab 3

Problem/Objective: Anomaly Detection

AWS Service: SageMaker, NEO

DL Architecture Overview: Autoencoder

Relevant Applications: Time-Series

Lab 1

Problem/Objective: Object Detection

AWS Service: Rekognition

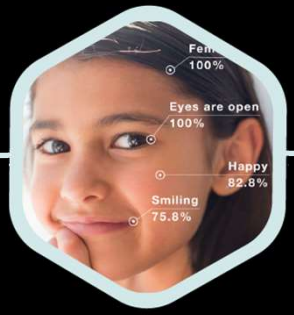
DL Architecture Overview: CNN Demo [3D, flat]

Relevant Applications: Robotics, Retail, AV

SOTA : [paperswithcode.com/sota]

Amazon Rekognition Image

Object and Scene Detection



Facial Analysis

Face Recognition

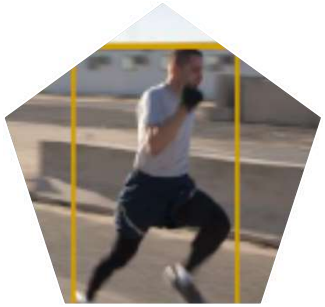
Unsafe Image Detection

Celebrity Recognition

Text in Image

Amazon Rekognition Video

Object and Activity Detection



Person Tracking

Face Recognition

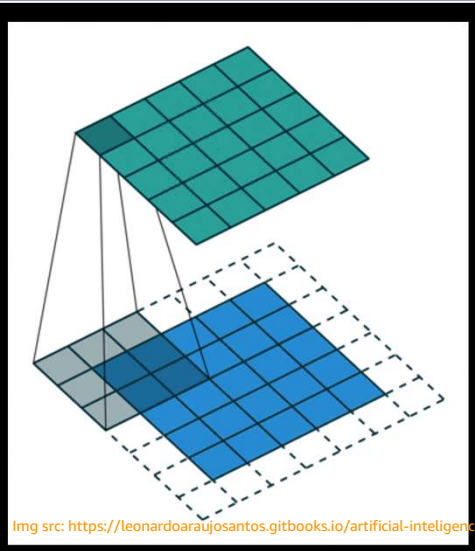
Real-time Live Stream

Content Moderation

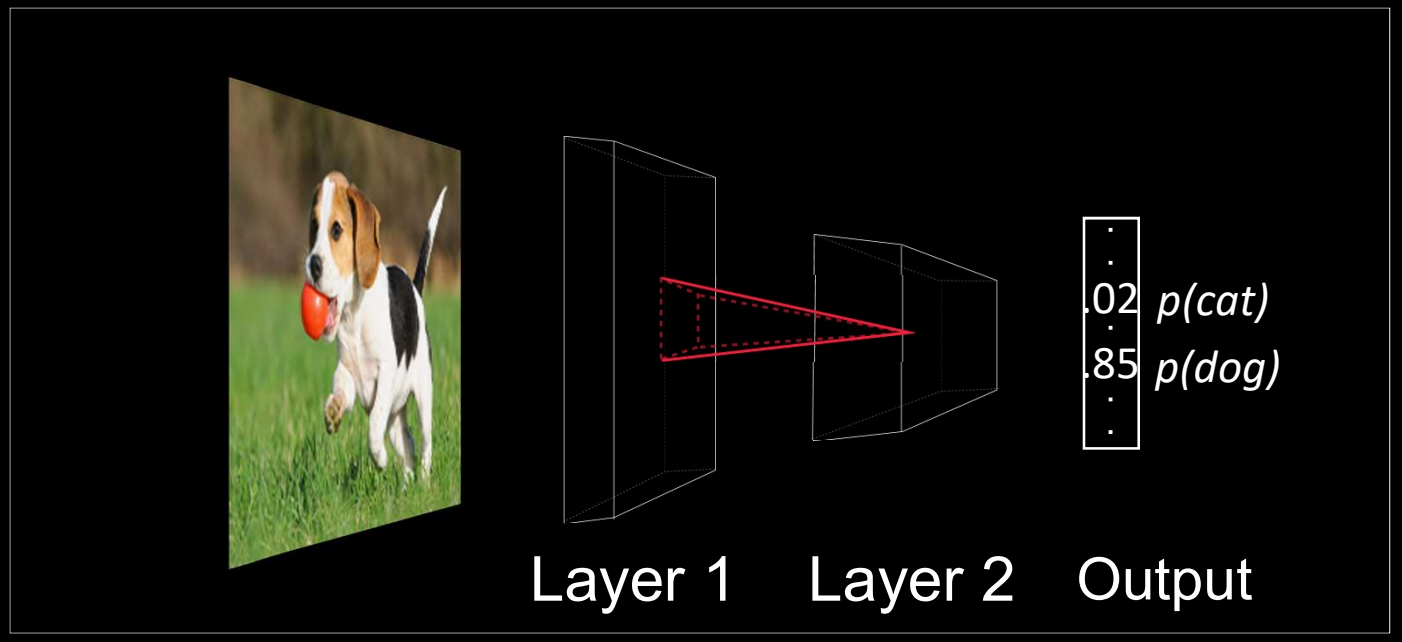
Celebrity Recognition

Deep Learning in Computer Vision

Convolutional neural network



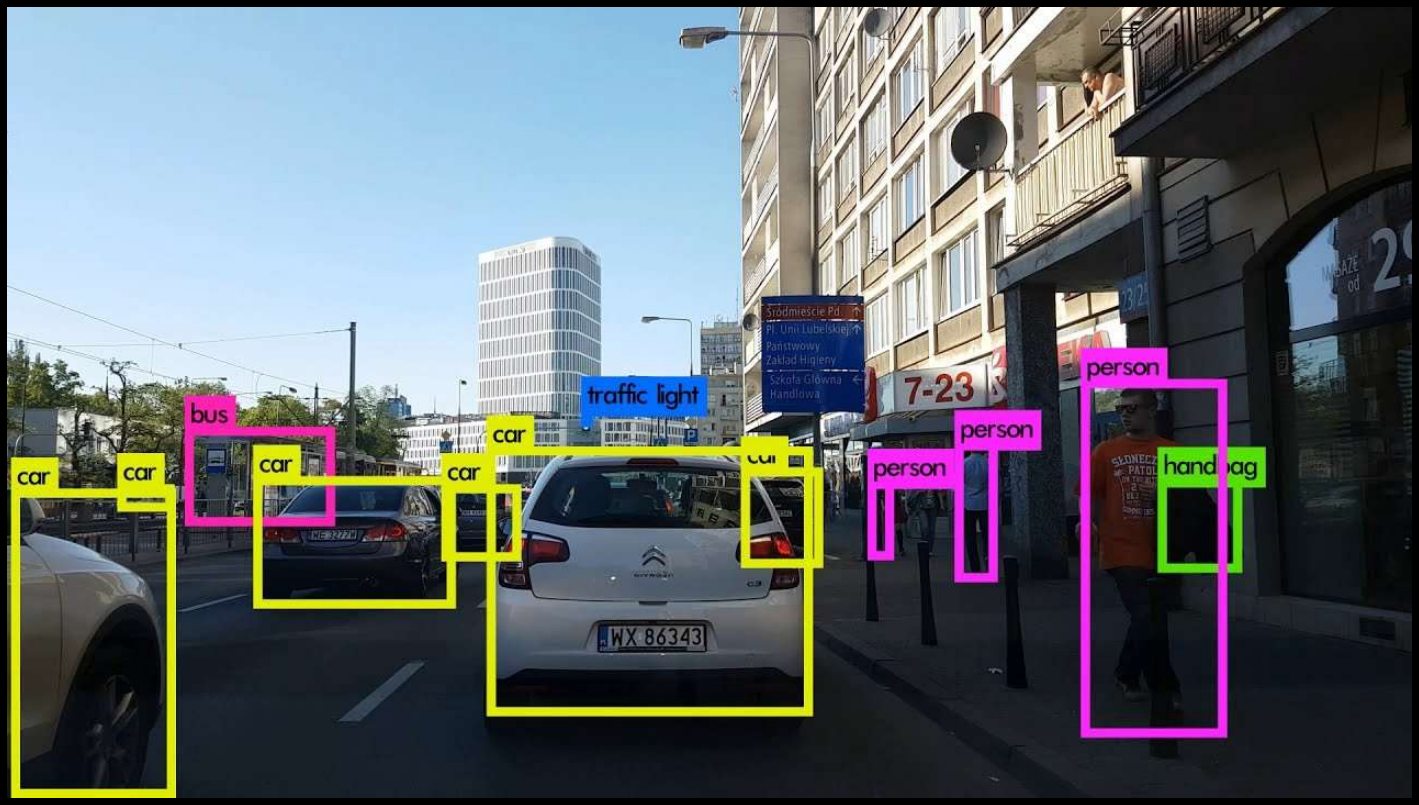
Img src: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence



Explore spatial information with convolution layers

aws SUMMIT

Object Detection



aws SUMMIT

Single Shot Detector

Lab 2

Problem/Objective: fine-tuning BERT for Question Answering

AWS Service: Comprehend

DL Architecture Overview: Illustrated BERT

Relevant Applications: Virtual Assistant/Chatbot

SOTA : [paperswithcode.com/sota]

https://github.com/astonzhang/KDD19-tutorial/blob/master/07_bert_app/bert.ipynb

2018 –NLP's Big Bang!

Leaderboard

SQuAD2.0 tests the ability of a system to not only answer reading comprehension questions, but also abstain when presented with a question that cannot be answered based on the provided paragraph. How will your system compare to humans on this task?

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Jul 22, 2019	XLNet + DAAF + Verifier (ensemble) *PINGAN Omni-Sinitic*	**88.592**	**90.859**
2 Jul 19, 2019	XLNet + SG-Net Verifier (ensemble) *Shanghai Jiao Tong University & CloudWalk*	88.050	90.645
3 Jul 23, 2019	XLNet + SG-Net Verifier (single model) *Shanghai Jiao Tong University & CloudWalk*	87.046	89.899
3 Mar 20, 2019	BERT + DAE + AoA (ensemble) *Joint Laboratory of HIT and iFLYTEK Research*	87.147	89.474
3 Jul 20, 2019	RoBERTa (single model) *Facebook AI*	86.820	89.795
4 Mar 15, 2019	BERT + ConvLSTM + MTL + Verifier (ensemble) *Layer 6 AI*	86.730	89.286

https://rajpurkar.github.io/SQuAD-explorer/

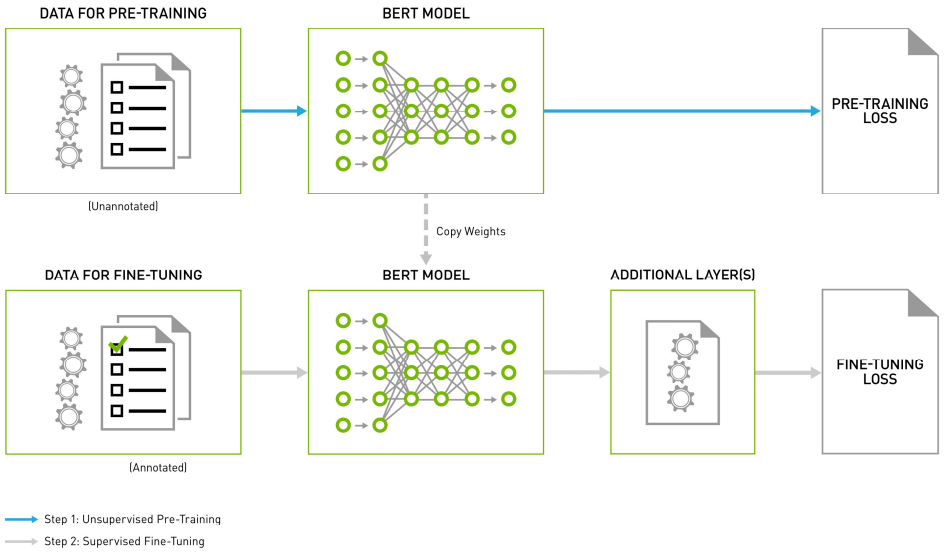
- pre-training representations on massive data
- fine tuning for specific tasks
- matching human performance [?]
- protocol for M2M, M2H

"overall, people are still much better than machines at comprehending the complexity and nuance of language." - J. Gao, MSR Asia

BERT Pipeline

[pretraining into Q&A]

https://github.com/NVIDIA/DeepLearningExamples



- BERT objective:
  - total_loss = masked_lm_loss + next_sentence_loss

Input: The man went to the $[MASK]_1$. He bought a $[MASK]_2$ of milk
Labels: $[MASK]_1$ = store; $[MASK]_2$ = gallon

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence

- Input Question:

  Where do water droplets collide with ice crystals to form precipitation?

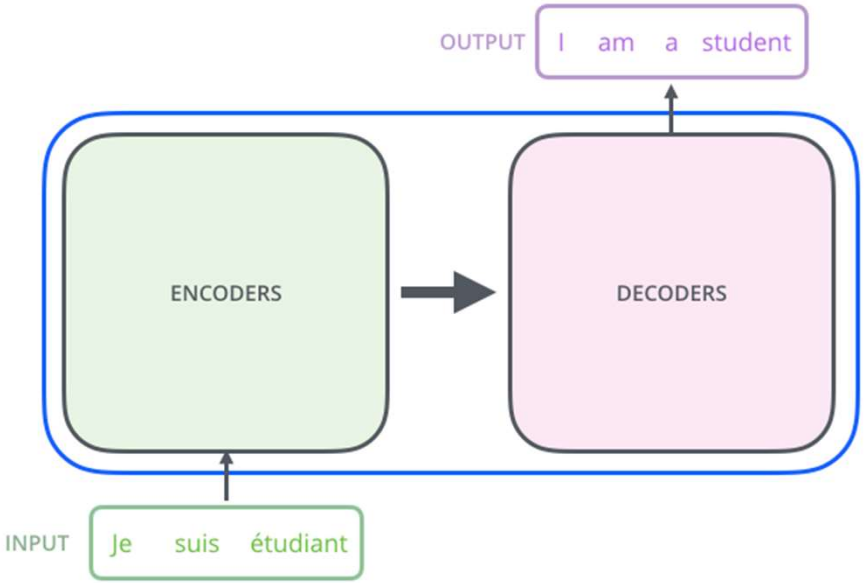
- Input Paragraph:

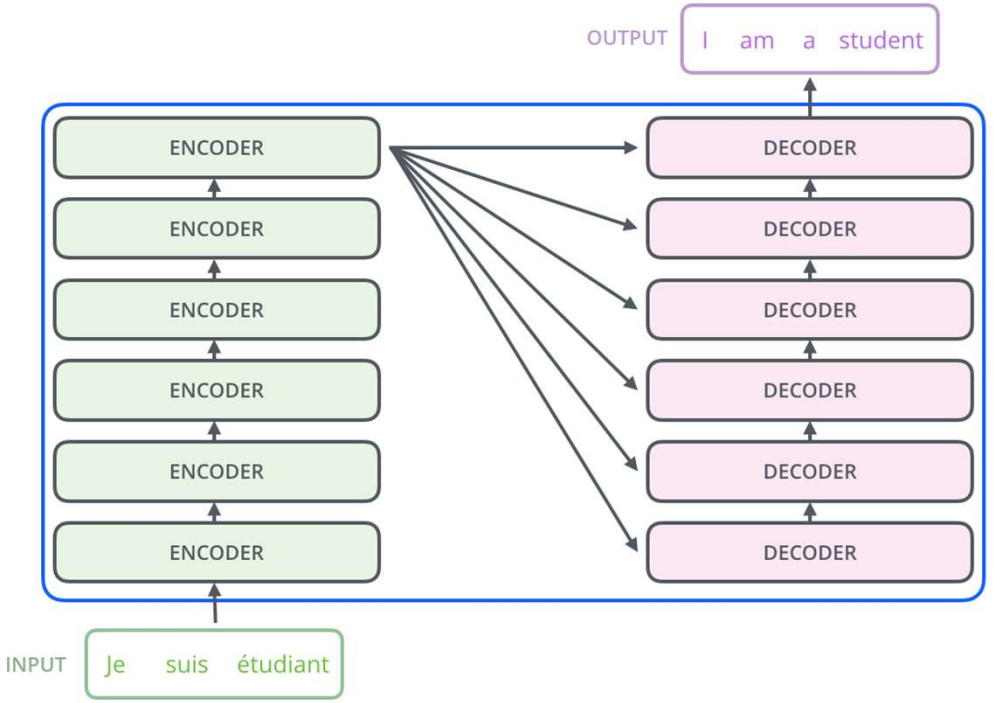
  ... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals within a cloud. ...

- Output Answer:

  within a cloud

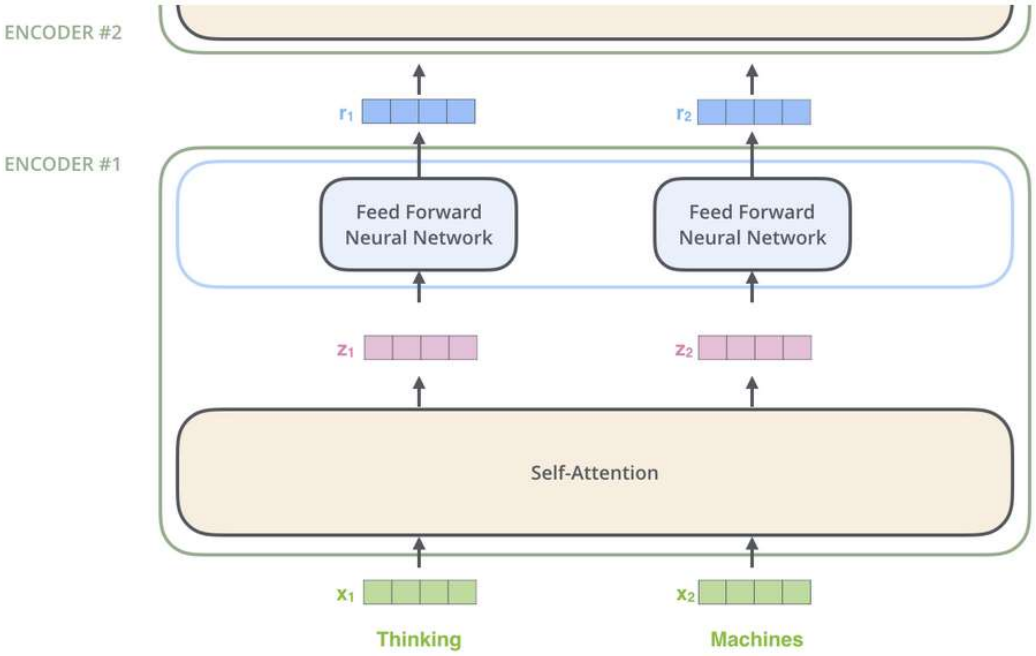
Transformer [Encoder – Decoder]



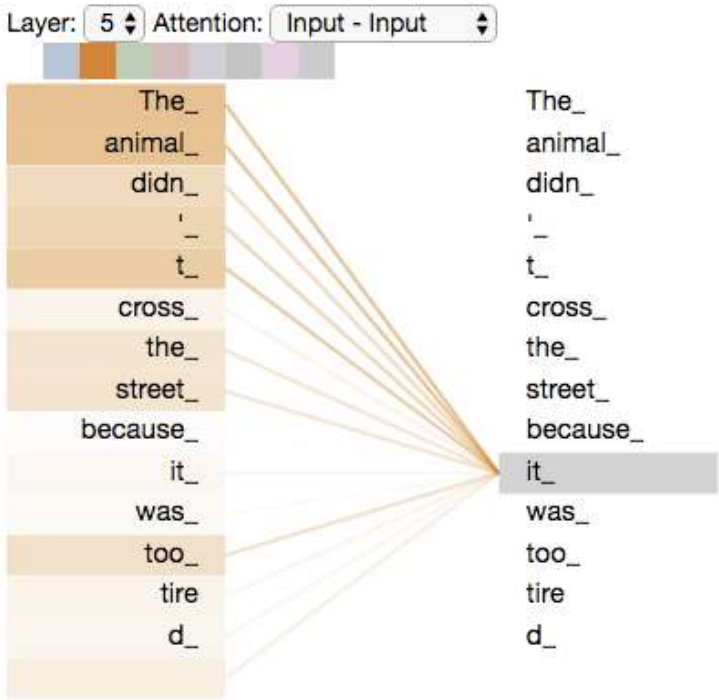


http://jalammar.github.io/illustrated-transformer/

Self-Attention



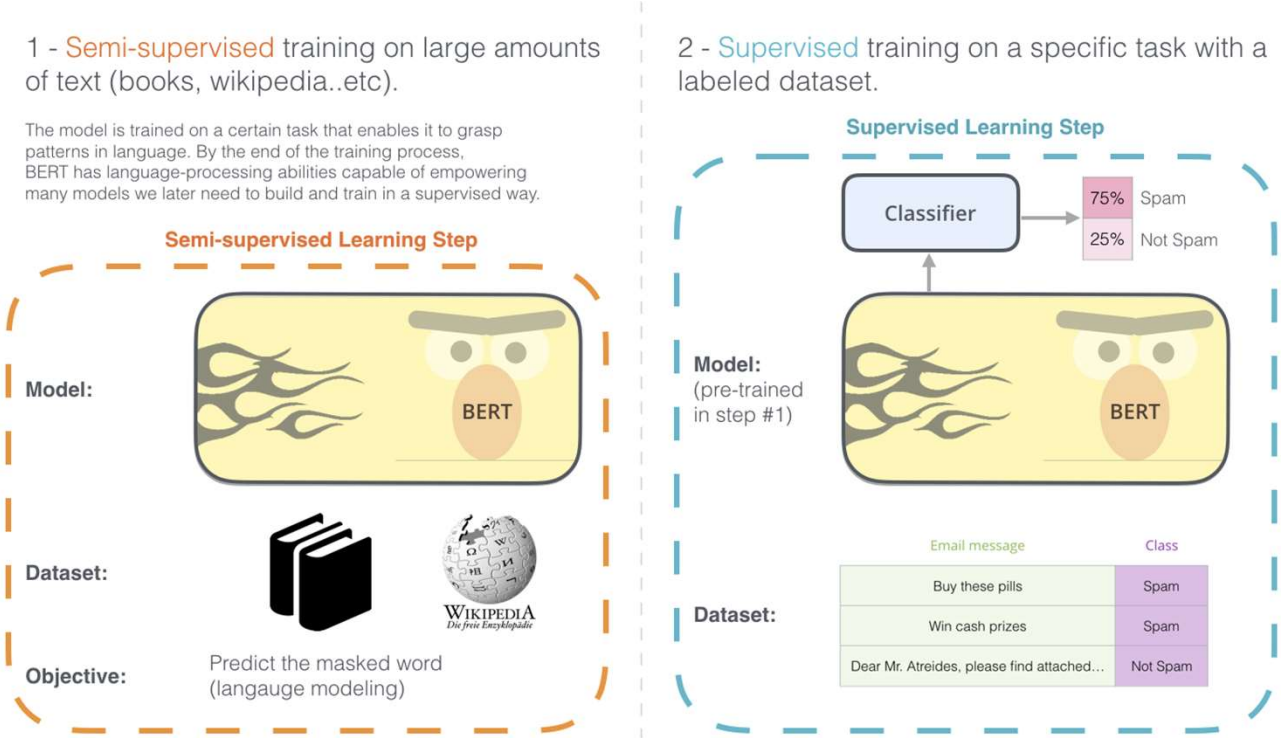
The word at each position passes through a self-attention process. Then, they each pass through a feed-forward neural network -- the exact same network with each vector flowing through it separately.



[Interactive Attention Viz]

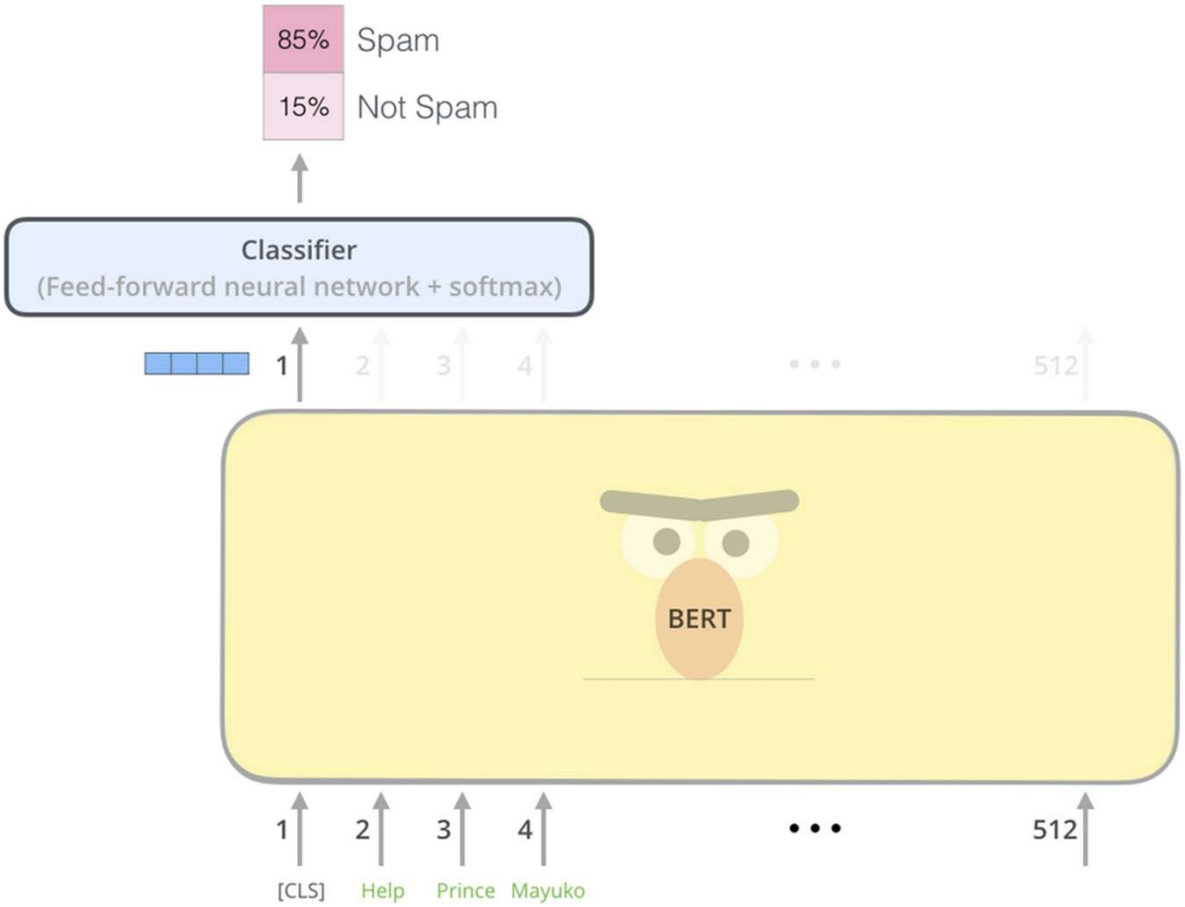
http://jalammar.github.io/illustrated-transformer/

References

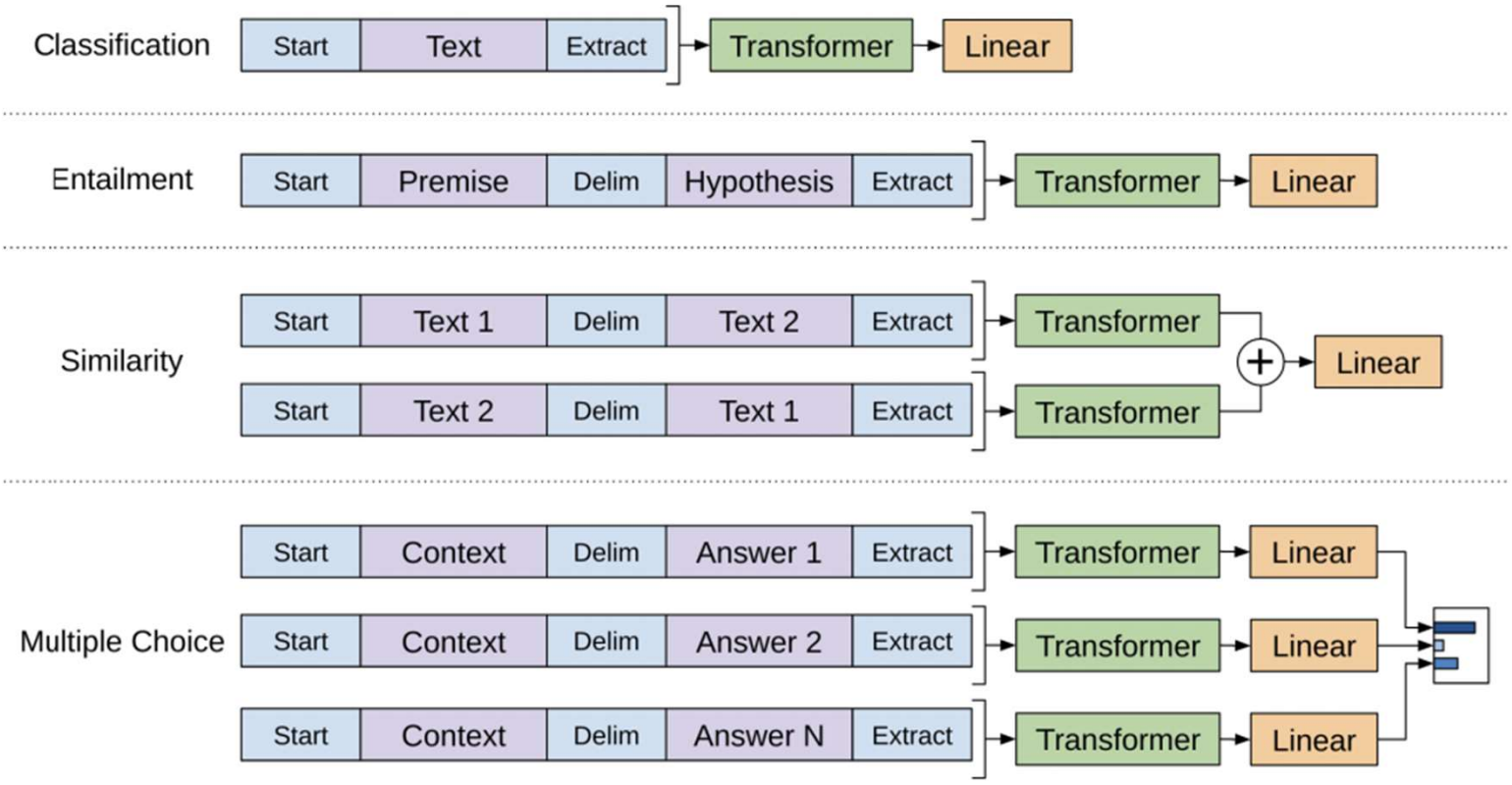


- https://yashuseth.blog/2019/06/12/bert-explained-faqs-understand-bert-working/
- https://arxiv.org/abs/1810.04805v2
- http://jalammar.github.io/illustrated-transformer/
- http://jalammar.github.io/illustrated-bert/

Transfomer + Upstream Classifier



Different Upstream Tasks



Gluon NLP Lab [Sentiment Analysis]

BERT Pre-training and Fine-tuning

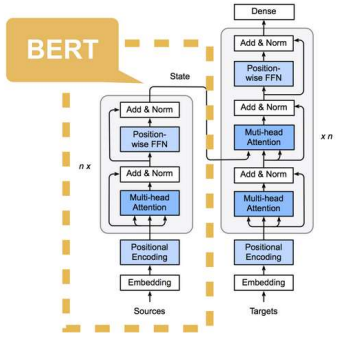
Preparation

First, let's import necessary modules.

Note that utils.py includes some Blocks defined in the previous transformer notebook

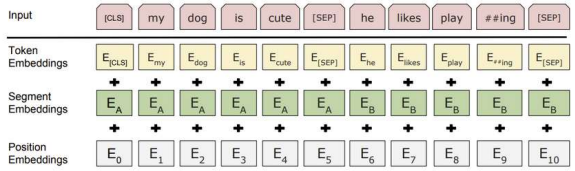
In [1]:
```
import random, math

import d2l
import numpy as np
import mxnet as mx
from mxnet import gluon, nd
import gluonnlp as nlp

from utils import PositionalEncoding, MultiHeadAttention
from utils import AddNorm, PositionWiseFFN, EncoderBlock
from utils import train_loop, predict_sentiment
```

Encoder



Segment Embedding

Different from the transformer encoder, the BERT encoder has an additional embedding for segment information.



Similar to the Transformer encoder defined in the previous section, the BERT encoder has embeddings for words and positions. The `EncoderBlock` contains position-wise feed-forward network and self-attention blocks to encode inputs. For BERT, the newly added segment embedding captures the segment information of the input sentence pairs, used for the next sentence prediction task.

BERT Encoder

In [2]:
```
class BERTEncoder(gluon.nn.Block):
    def __init__(self, vocab_size, units, hidden_size,
                 num_heads, num_layers, dropout, **kwargs):
        super(BERTEncoder, self).__init__(**kwargs)
        # segment_embed for segment information
        self.segment_embed = gluon.nn.Embedding(2, units)
        self.word_embed = gluon.nn.Embedding(vocab_size, units)
        self.pos_encoding = PositionalEncoding(units, dropout)
        self.blks = gluon.nn.Sequential()
        for i in range(num_layers):
            self.blks.add(EncoderBlock(units, hidden_size, num_heads, dropout))

    def forward(self, words, segments, mask, *args):
        X = self.word_embed(words) + self.segment_embed(segments)
        X = self.pos_encoding(X)
        for blk in self.blks:
            X = blk(X, mask)
        return X
```

Using the BERT Encoder

Now let's test the BERTEncoder with a data batch of 2 sentence pairs, each with 8 words. Random integers are used to represent words for demonstration purpose. For segment information, we use 0 to indicate the word comes from the first sentence, 1 to indicate the second setence.

In [3]:
```
encoder = BERTEncoder(vocab_size=30000, units=768, hidden_size=3072,
                      num_heads=12, num_layers=12, dropout=0.1)
encoder.initialize()

num_samples, num_words = 2, 8
# random words for testing
words = nd.random.randint(low=0, high=30000, shape=(num_samples, num_words))
# the corresponding segment information for each word
segments = nd.array([[0,0,0,0,1,1,1,1],[0,0,0,1,1,1,1,1]])

encodings = encoder(words, segments, None)
print(encodings.shape) # (batch_size, num_words, units)
```

```
(2, 8, 768)
```

Next Sentence Classifier

Let us take a look at the first pre-training task: next sentence prediction. For this task, the encoding of the first token (the "[CLS]" token) is passed to a feed-forward network to make prediction.

Since next sentence prediction is a binary classification problem, we can use `SigmoidBinaryCrossEntropyLoss` as the loss function. In the following code block, we pass the encoding results to the `NSClassifier` to get the next sentence prediction. We use 1 as the label for true next sentence, and 0 otherwise. The prediction result and the label are then passed to the loss function for loss evaluation.

In [4]:
```
class NSClassifier(gluon.nn.Block):
    def __init__(self, units=768, **kwargs):
```

Refs

PyTorch-Transformers

PyTorch-Transformers

PASSED

PyTorch-Transformers (formerly known as `pytorch-pretrained-bert`) is a library of state-of-the-art pre-trained models for Natural Language Processing (NLP).

The library currently contains PyTorch implementations, pre-trained model weights, usage scripts and conversion utilities for the following models:

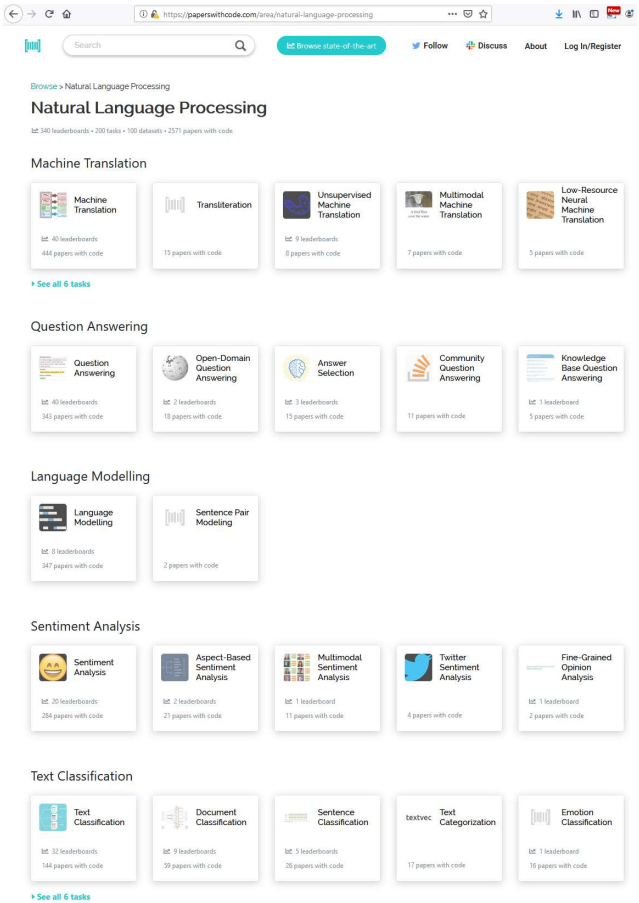
1. BERT (from Google) released with the paper BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding by Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova.
2. GPT (from OpenAI) released with the paper Improving Language Understanding by Generative Pre-Training by Alec Radford, Karthik Narasimhan, Tim Salimans and Ilya Sutskever.
3. GPT-2 (from OpenAI) released with the paper Language Models are Unsupervised Multitask Learners by Alec Radford*, Jeffrey Wu*, Rewon Child, David Luan, Dario Amodei** and Ilya Sutskever**.
4. Transformer-XL (from Google/CMU) released with the paper Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context by Zihang Dai*, Zhilin Yang*, Yiming Yang, Jaime Carbonell, Quoc V. Le, Ruslan Salakhutdinov.
5. XLNet (from Google/CMU) released with the paper XLNet: Generalized Autoregressive Pretraining for Language Understanding by Zhilin Yang*, Zihang Dai*, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, Quoc V. Le.
6. XLM (from Facebook) released together with the paper Cross-lingual Language Model Pretraining by Guillaume Lample and Alexis Conneau.

These implementations have been tested on several datasets (see the example scripts) and should match the performances of the original implementations (e.g. ~93 F1 on SQuAD for BERT Whole-Word-Masking, ~88 F1 on RocStories for OpenAI GPT, ~18.3 perplexity on WikiText 103 for Transformer-XL, ~0.916 Peason R coefficient on STS-B for XLNet). You can find more details on the performances in the Examples section of the documentation.

Section	Description
Installation	How to install the package
Quick tour: Usage	Tokenizers & models usage: Bert and GPT-2
Quick tour: Fine-tuning/usage scripts	Using provided scripts: GLUE, SQuAD and Text generation
Migrating from pytorch-pretrained-bert to pytorch-transformers	Migrating your code from pytorch-pretrained-bert to pytorch-transformers
Documentation	Full API documentation and more

https://github.com/huggingface/pytorch-transformers

Papers With Code



https://paperswithcode.com/

Lab 3

[optional]

Problem/Objective: Anomaly Detection

AWS Service: SageMaker, NEO

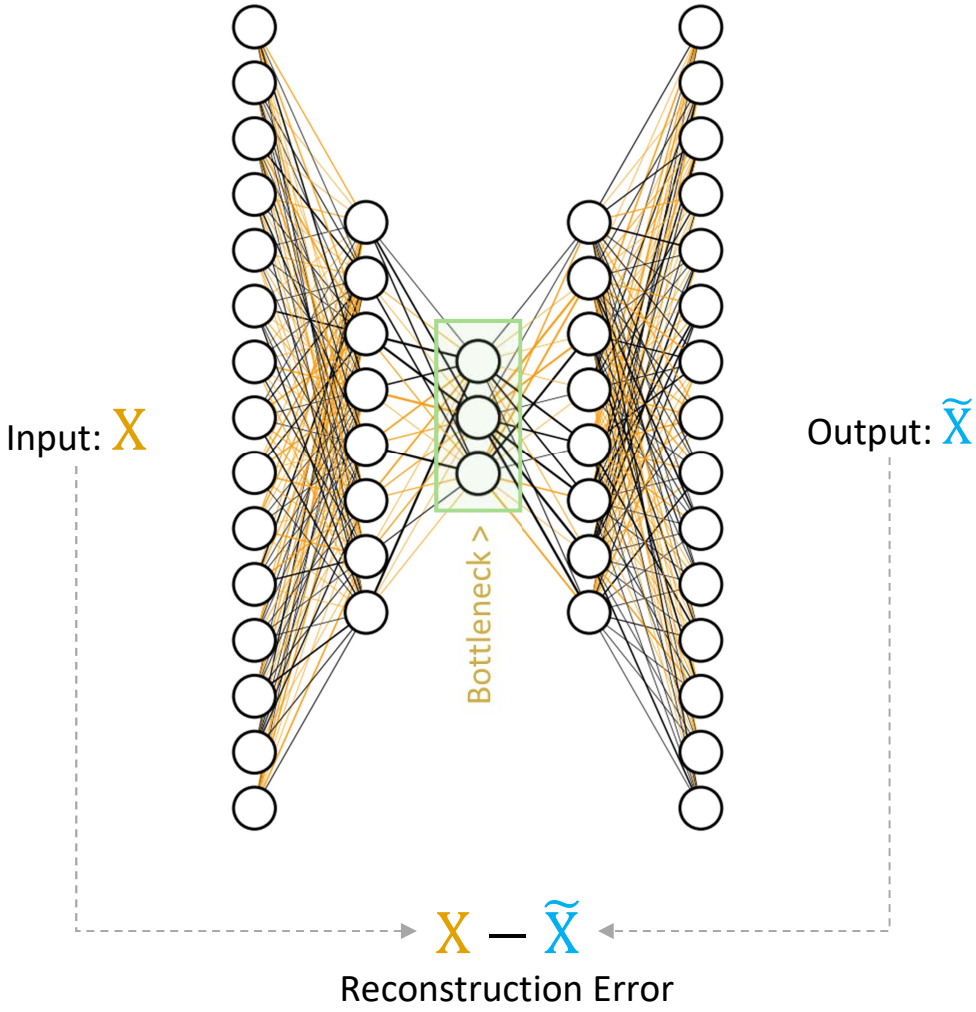
DL Architecture Overview: Autoencoder

Relevant Applications: Time-Series

https://github.com/miroenev/teach_DL/anomaly_detection

Deep Autoencoder

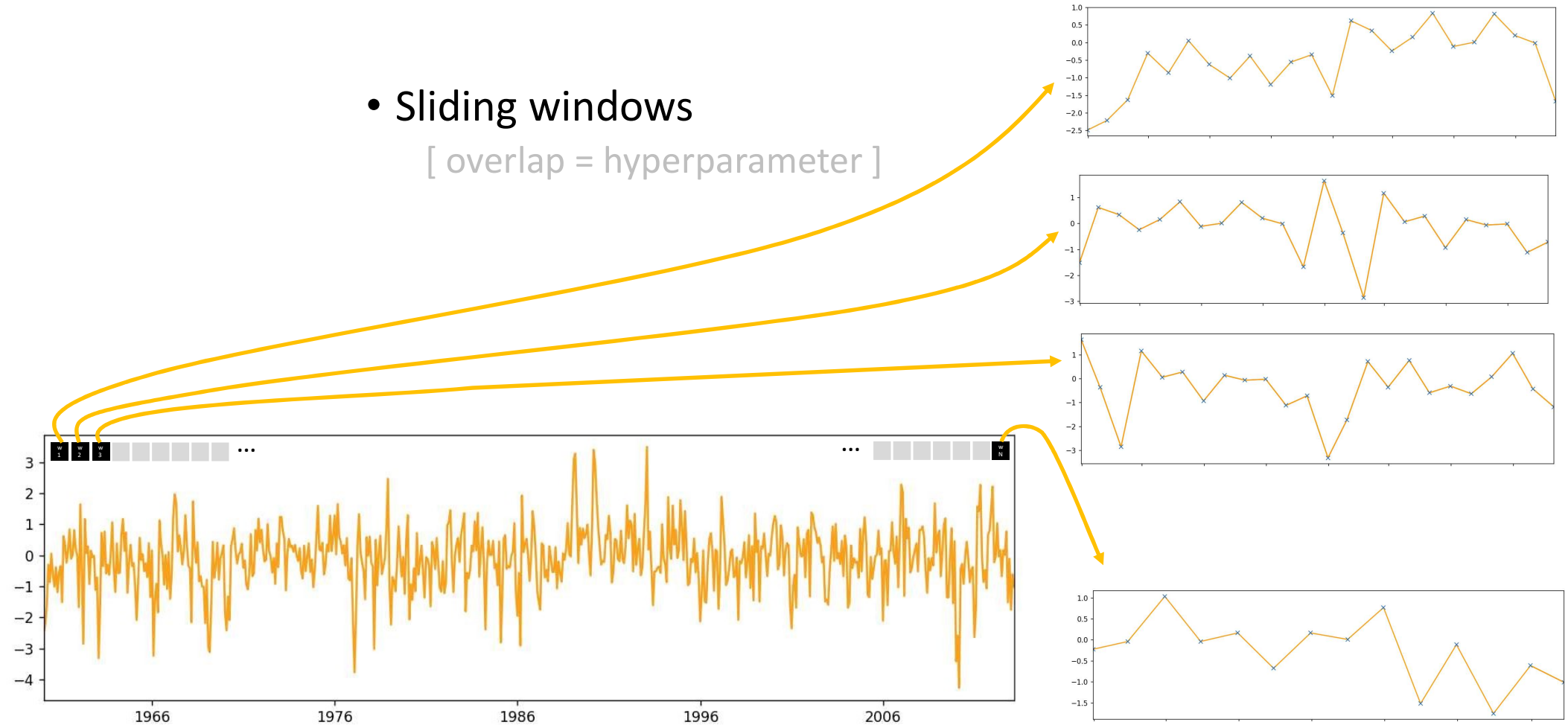
- Input layer
  - Time-Horizon
- Bottleneck layer
  - Summarized representation/‘embedding’
- Output layer
  - Same dimensionality as input
- Reconstruction error
  - High errors indicate potential anomaly



Data pre-processing

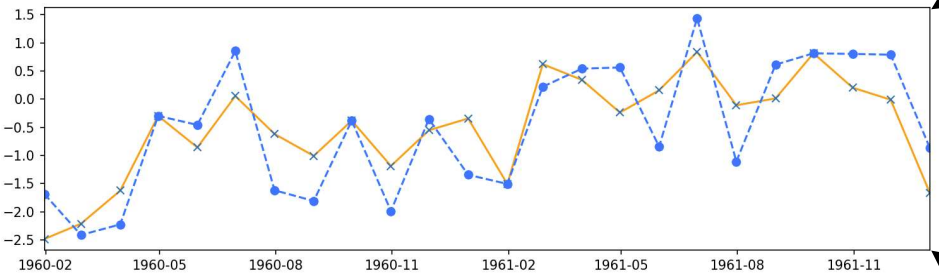
- Sliding windows

  [overlap = hyperparameter]



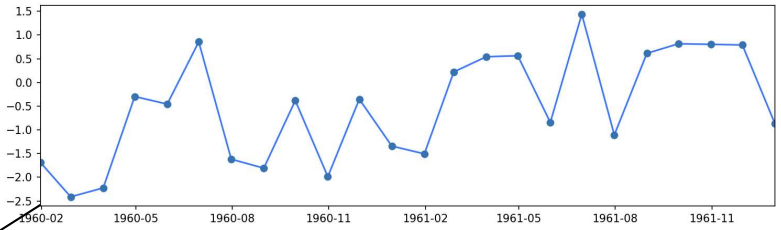
Reconstruction Error

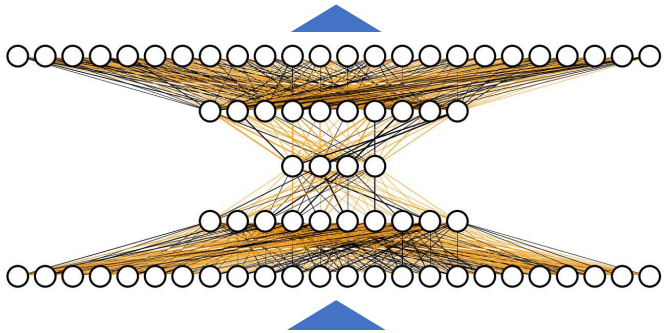
[Trigger for Anomaly]

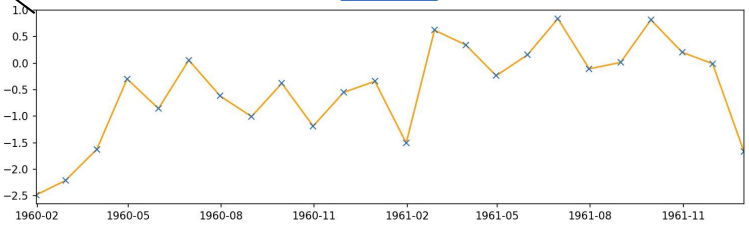


Reconstruction vs Input

Output (Reconstruction)



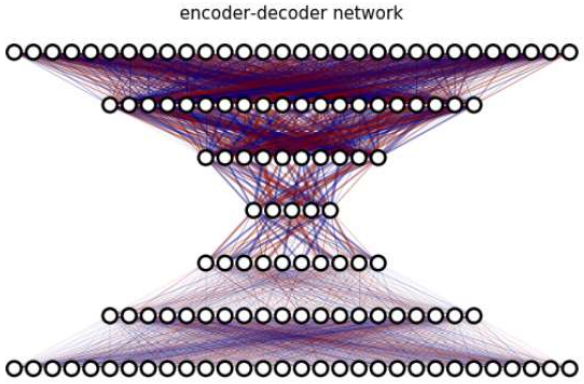


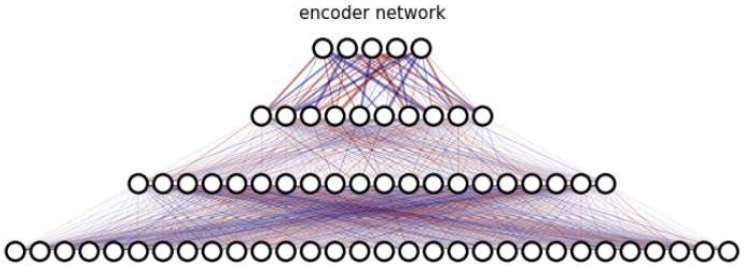


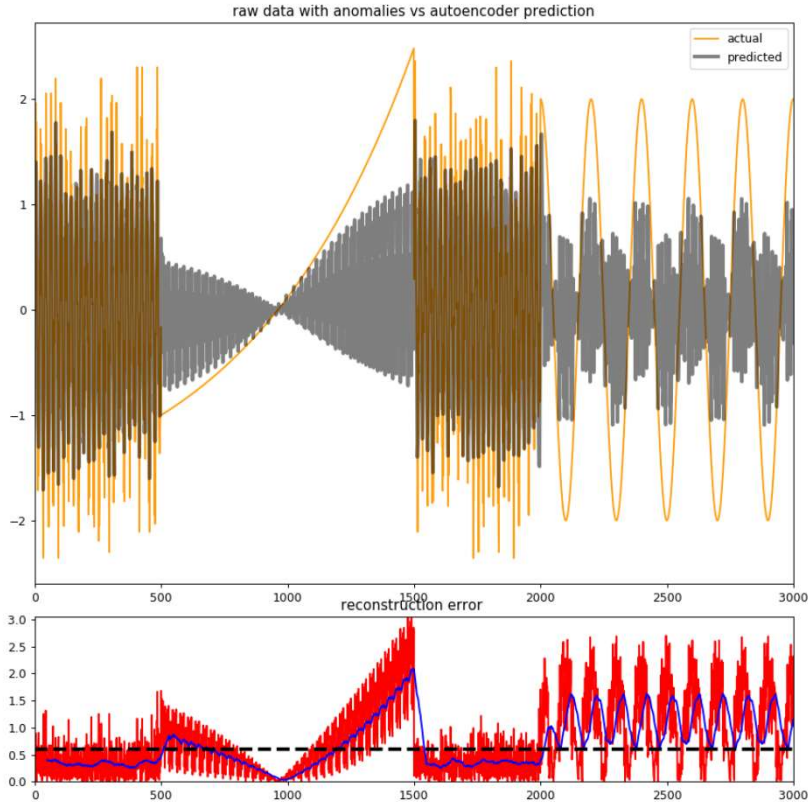
Input

Anomaly Detection on Time Series Data

In this lab we'll generate [synthetic] time-series data and train a DL model [autoencoder] to encode [summarize/embed] and reconstruct sliding windows from this dataset. In this context, the amount of reconstruction error (the difference between an input and the trained network's output) is used as an indicator of potential anomalies. To test this we'll mix in artificial anomalies into normal data and analyze the model's reconstruction error.



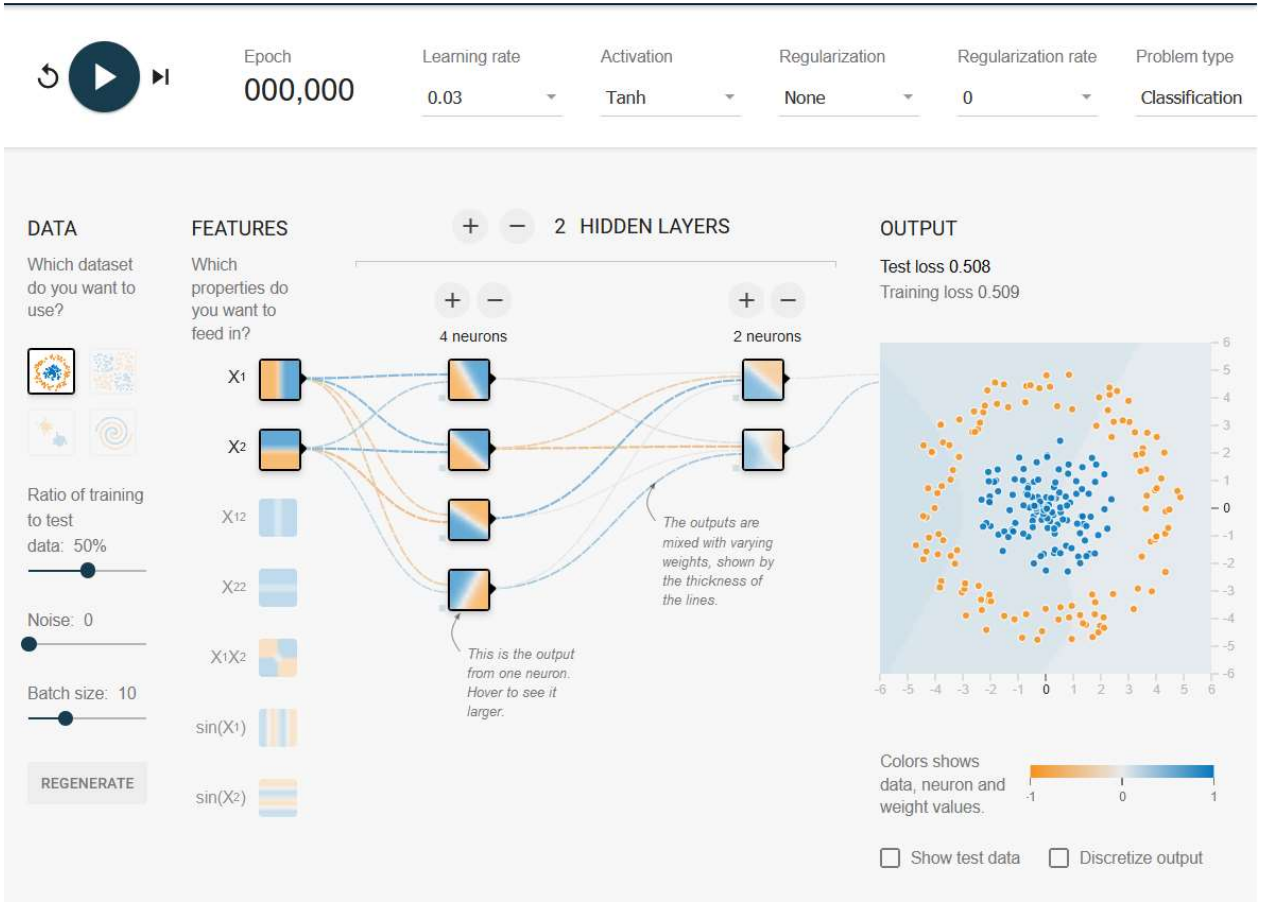




Thank you KDD!

Appendix

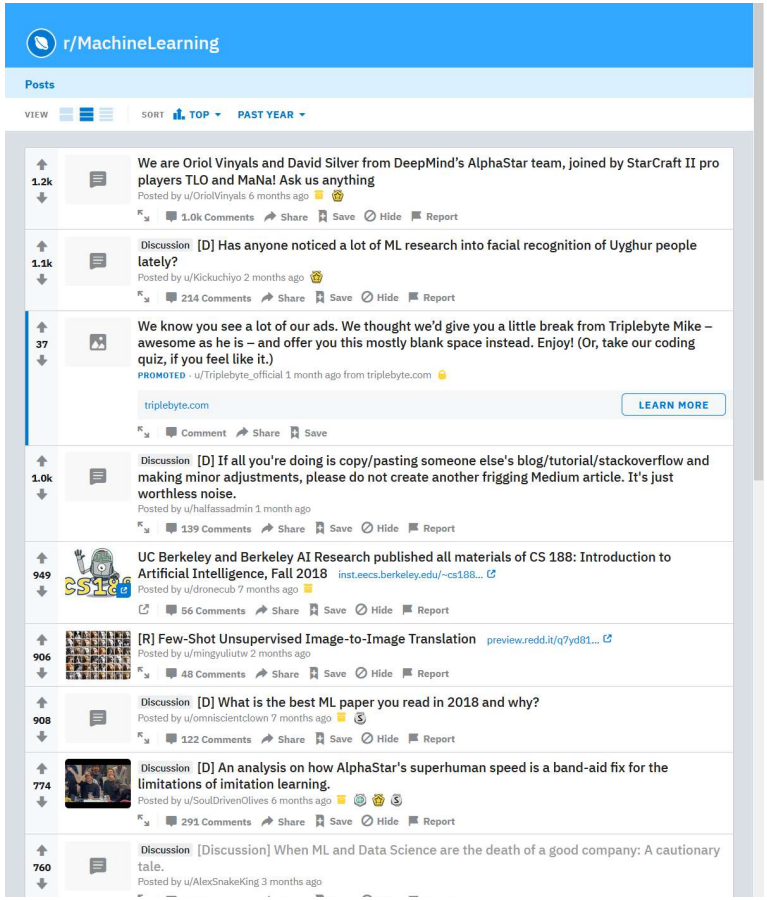
Demo


Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 0
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
2 HIDDEN LAYERS
4 neurons
2 neurons
The outputs are mixed with varying weights, shown by the thickness of the lines.
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.508
Training loss 0.509
Colors shows data, neuron and weight values.
Show test data
Discretize output

Resources

http://www.arxiv-sanity.com/top

https://www.reddit.com/r/MachineLearning/top/?t=year



DLI- Deep Learning Institute

https://www.nvidia.com/en-us/deep-learning-ai/education/

DEEP LEARNING FUNDAMENTALS

- Fundamentals of Deep Learning for Computer Vision
- Getting Started with AI on Jetson Nano
- Image Classification with DIGITS
- Object Detection with DIGITS
- Optimization and Deployment of TensorFlow Models with TensorRT
- Accelerating Data Science Workflows with RAPIDS
- Image Segmentation with TensorFlow
- Signal Processing with DIGITS

DEEP LEARNING FOR DIGITAL CONTENT CREATION

- Image Style Transfer with Torch
- Rendered Image Denoising Using Autoencoders
- Image Super Resolution Using Autoencoders

DEEP LEARNING FOR HEALTHCARE

- Modeling Time Series Data with Recurrent Neural Networks in Keras
- Medical Image Classification Using the MedNIST Dataset
- Data Science Workflows for Deep Learning in Medical Applications
- Medical Image Segmentation with DIGITS
- Image Classification with TensorFlow: Radiomics—1p19q Chromosome Status Classification
- Medical Image Analysis with R and MXNet
- Data Augmentation and Segmentation with Generative Networks for Medical Imaging
- Coarse-to-Fine Contextual Memory for Medical Imaging

DEEP LEARNING FOR INTELLIGENT VIDEO ANALYTICS

- AI Workflows for Intelligent Video Analytics with DeepStream

CUDA - X

https://www.nvidia.com/en-in/technologies/cuda-x/

1 Computational Finance

2 Climate, Weather and Ocean Modeling

2 Data Science and Analytics

4 Artificial Intelligence
- DEEP LEARNING AND MACHINE LEARNING

8 Federal, Defense and Intelligence

10 Design for Manufacturing/Construction: CAD/CAE/CAM
- COMPUTATIONAL FLUID DYNAMICS
- COMPUTATIONAL STRUCTURAL MECHANICS
- DESIGN AND VISUALIZATION
- ELECTRONIC DESIGN AUTOMATION
- INDUSTRIAL INSPECTION

19 Media & Entertainment
- ANIMATION, MODELING AND RENDERING
- COLOR CORRECTION AND GRAIN MANAGEMENT
- COMPOSITING, FINISHING AND EFFECTS
- EDITING
- ENCODING AND DIGITAL DISTRIBUTION
- ON-AIR GRAPHICS
- ON-SET, REVIEW AND STEREO TOOLS
- WEATHER GRAPHICS

29 Medical Imaging

30 Oil and Gas

31 Research: Higher Education and Supercomputing
- COMPUTATIONAL CHEMISTRY AND BIOLOGY
- NUMERICAL ANALYTICS
- PHYSICS
- SCIENTIFIC VISUALIZATION

47 Safety and Security

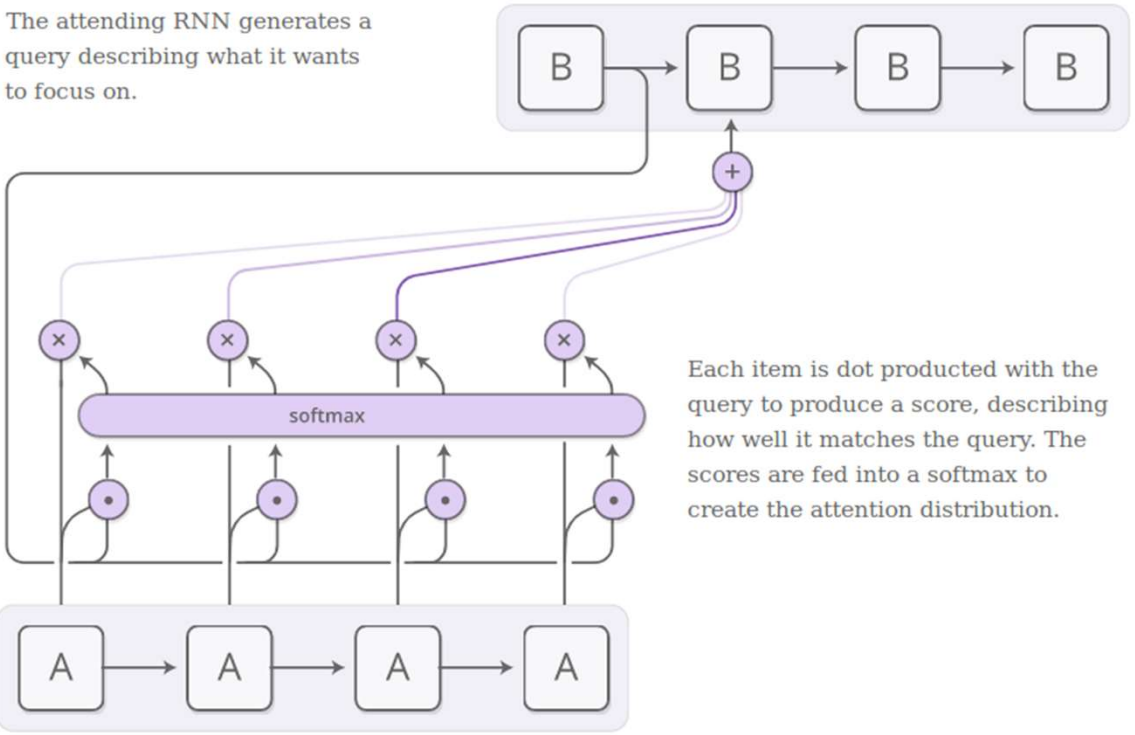
49 Tools and Management

Mixed Precision

Conclusions

- **Mixed precision training benefits:**
  - Math, memory speedups
  - Larger minibatches, larger inputs
- **Automatic Loss Scaling simplifies mixed precision training**
- **Mixed precision matches FP32 training accuracy for a variety of:**
  - Tasks: classification, regression, generation
  - Problem domains: images, language translation, language modeling, speech
  - Network architectures: feed forward, recurrent
  - Optimizers: SGD, Adagrad, Adam
- **Note on inference:**
  - Can be purely FP16: storage and math (use library calls with FP16 accumulation)
- **More details:**
  - S81012: Training Neural Newtorks with Mixed Precision: Real Examples (Thu, 9am)
  - http://docs.nvidia.com/deeplearning/sdk/mixed-precision-training/

Attention



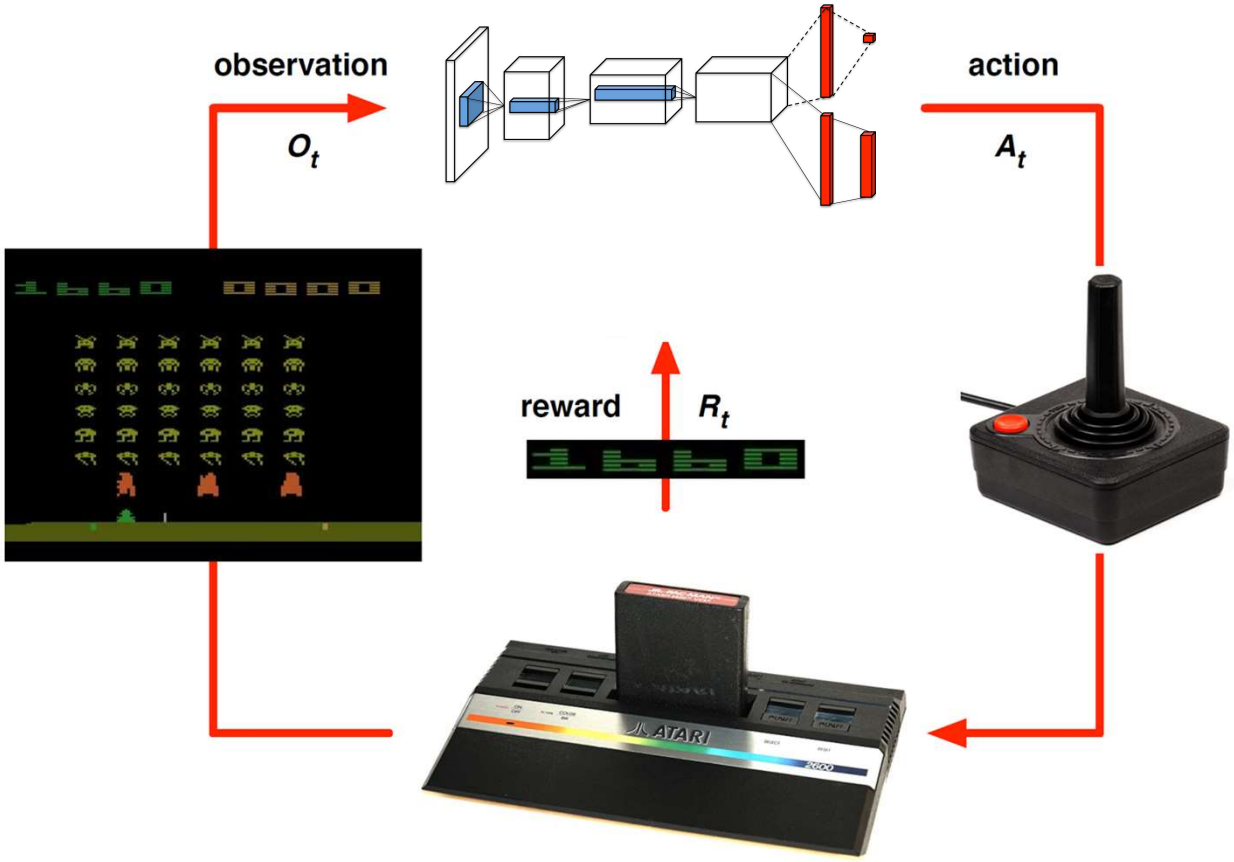
- https://distill.pub/2016/augmented-rnns/#attentional-interfaces

Reinforcement Learning

Applying RL to games

Atari Example

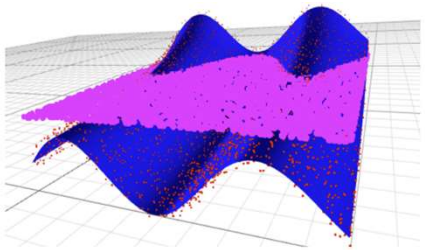
- Agent is a DL network
  - Interprets screen pixels
  - Outputs game action
  - Possible to use CNN+RNN
- Environment is Atari Emulator
  - Game AI
- Reward is game score



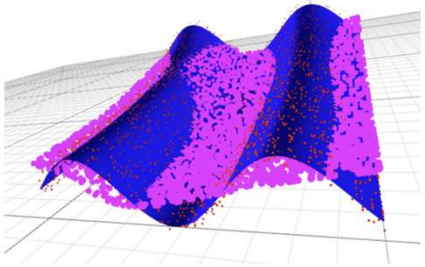
DL Intro

- Visualization & Model Training
  - Full Fidelity Visualization [vs] Graph Nodes
  - Model evolution over time

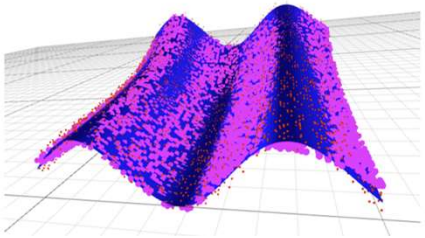
Before Training
[Random Initialization]

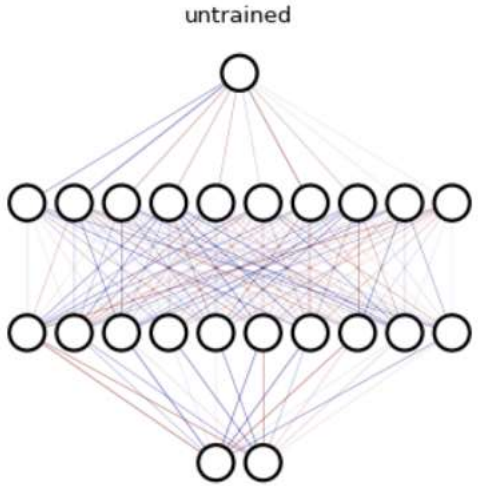


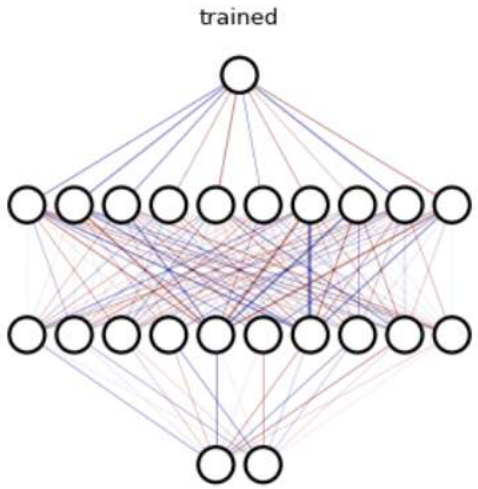
Midway
[150 epochs]



Done Training
[300 epochs]







GluonNLP Integration in ELIT

Word2Vec (Mikolov et al., 2013)
GloVe (Pennington et al., 2014)
FastText (Bojanowski et al., 2017)
ELMo (Peters et al., 2018)
BERT (Devlin et al., 2018)

← ELIT

MXNet Implementation of Contextual String Embedding model (Akbik et al., 2018)