

# **CIS 6930 Topics in Computing for Data Science**

## **Week 12: More Deep Learning Topics (2)**

Tue 11/18/2021  
Yoshihiko (Yoshi) Suhara

**2pm-3:20pm & 3:30pm-4:50pm**

# Course Schedule

- (Today) Thu 11/18 **(2x 80min)** (\* Finalize the term project title & plan!)
- Tue 11/23 Project presentation (1)
- Thu 11/30 Project presentation (2) (\* Final session! 😭)

# **Term Project**

## **Project Presentation & Project Report**

# Term Project

- ~~Fri 11/12: Feedback & Pre-finalize the project plan~~
- **Fri 11/18:** Finalize the project title and plan
- Tue 11/23: Project presentations (1)
- Tue 11/30: Project presentations (2)
- **Tue 12/7:** Project report deadline

Each student must work on their own term project (**No group work**)

Did you confirm the plan with me? Please reply to the e-mail  
Alternatively, use the office hours after the session to discuss it

# Project Presentation (5min + 3min QA)

**Tue 11/23 & Tue 11/30**

The presentation should contain

- 1) Motivation (30 seconds)
- 2) Clear problem definition (1 minute)
- 3) Methodology (1-2 minutes)
- 4) Evaluation: Results and Discussion (1-2 minutes)
- 5) Key takeaway (30 seconds)

## Notes

- Presentations will NOT be evaluated by the performance of your solution
- Students are encouraged to share in-progress work to get feedback from the instructor
- Please use this opportunity to improve the quality of your final project report

# Project Report (due Tue 12/7)

The project report must contain

- 1) Motivation
- 2) Clear problem definition + related background
- 3) Methodology: Description of the solution/baselines and the dataset
- 4) Evaluation: Results and Discussion
- 5) Personal reflection: Challenges encountered, lessons learned

The ideal project report should also answer the following questions

- How is your solution different from other solutions (baselines)?
- How better is your solution than the baselines?
- Why is your solution better than the baselines?

# **Any Questions?**

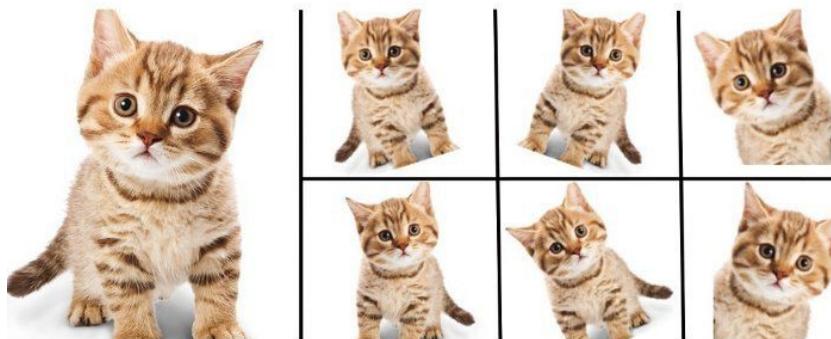
# Agenda: A Series of Short Talks!

- Advanced topics & Future directions
  - Data Augmentation
  - Adversarial Attacks; Interpretability & Explainability
  - Multilingual Models
  - Vision & Language
  - Pre-trained Language Models for Data Integration
  - Open AI GPT-3
  - Prompt Learning
- Topic that are not covered by the course
  - Reinforcement Learning
  - Graph Neural Networks
  - Time Series Data
  - MLOps

# Data Augmentation

# What is Data Augmentation and Why?

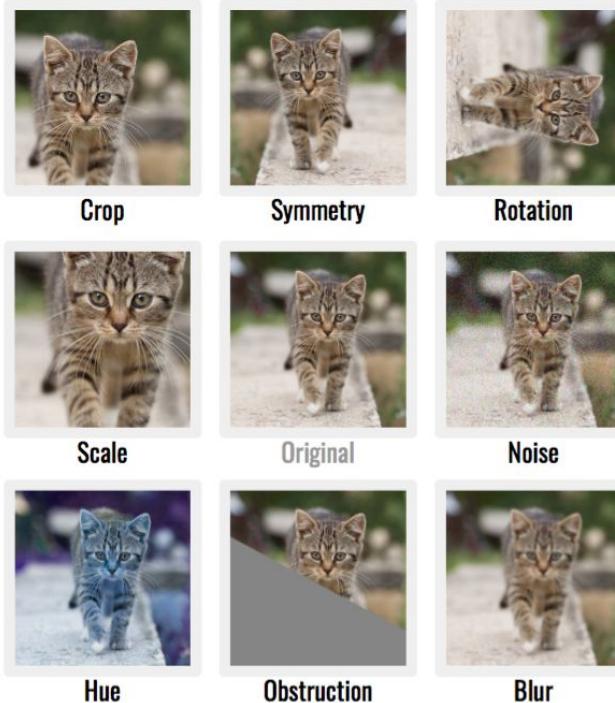
- A technique that **synthetically generate additional training examples** based on the original data to help the model more robust and generalized better



[Rotom: A Meta-Learned Data Augmentation Framework for Entity Matching, Data Cleaning, Text Classification, and Beyond](#)

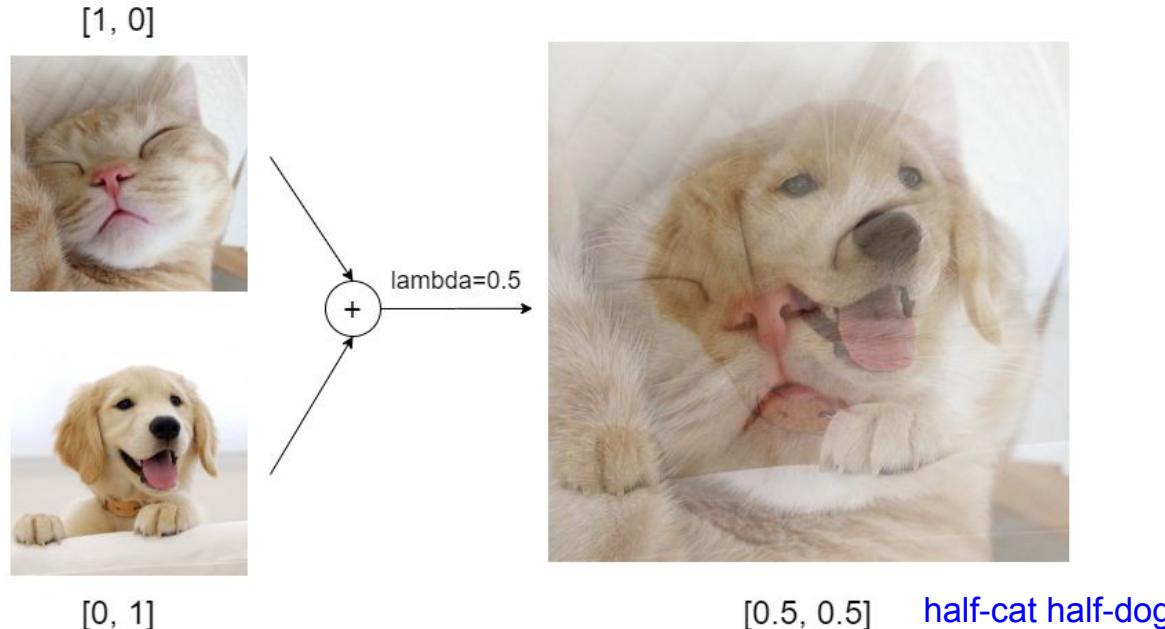
# Data Augmentation for Image Data: Basic Techniques

- Invariance property



# Data Augmentation for Image Data: MixUp

- Interpolate two images and their labels



# Cutout & CutMix

Image

ResNet-50



Mixup [47]



Cutout [3]



CutMix



[\[1905.04899\] CutMix: Regularization Strategy to Train Strong Classifiers with Localizable Features](#)

# Data Augmentation for Text Data

- It's not that straightforward compared to image data. Why?

# Data Augmentation for Text Data

- It's not that straightforward compared to image data. Why?
- → A little perturbation can completely change the meaning
  - e.g., It's a cat → It's not a cat

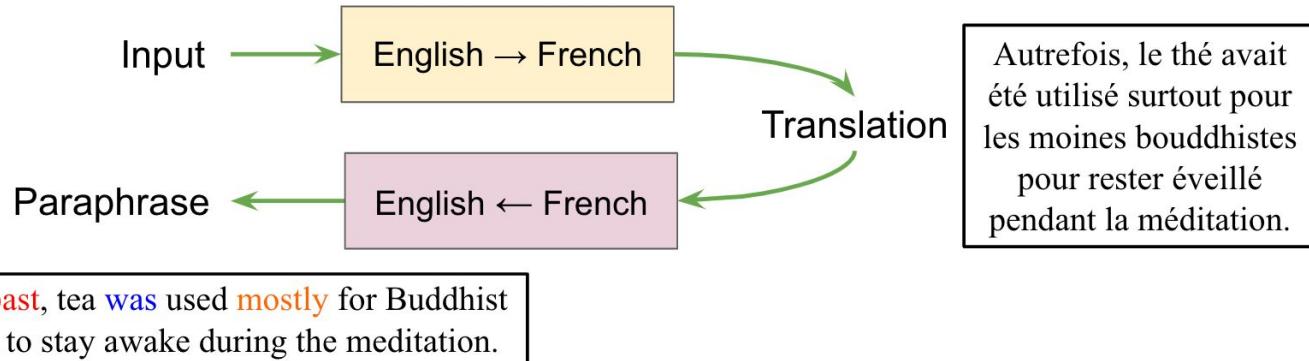
# Data Augmentation for Text: Basic Techniques

- Character/word-level random insertion/deletion/swapping
- Replacing randomly chosen words with synonyms
- etc.

# Model-based Data Augmentation

- Paraphrase generation
- Back-translation

Previously, tea had been used primarily for Buddhist monks to stay awake during meditation.



# Data Augmentation Toolbox

- Image
  - <https://github.com/facebookresearch/AugLy>
  - etc.
- Text
  - <https://github.com/styfeng/DataAug4NLP>
    - [\[2105.03075\] A Survey of Data Augmentation Approaches for NLP](#)
  - etc.

# NL-Augmenter

## Collaborative Data Augmentation

[https://gem-benchmark.com/nl\\_augmenter](https://gem-benchmark.com/nl_augmenter)

Feature Engineering in the DL era?



The NL-Augmenter is a collaborative effort intended to add transformations of datasets dealing with natural language. Transformations augment text datasets in diverse ways, including: introducing spelling errors, translating to a different language, randomizing names and numbers, paraphrasing, changing the style ... and whatever creative augmentation you contribute. We invite submissions of transformations to this framework by way of a [GitHub](#) pull request, through August 31,

# **Adversarial Attacks**

## **Interpretability & Explainability**

# Adversarial Examples: A Famous Example



$x$   
“panda”  
57.7% confidence

+ .007 ×



$\text{sign}(\nabla_x J(\theta, x, y))$   
“nematode”  
8.2% confidence

=

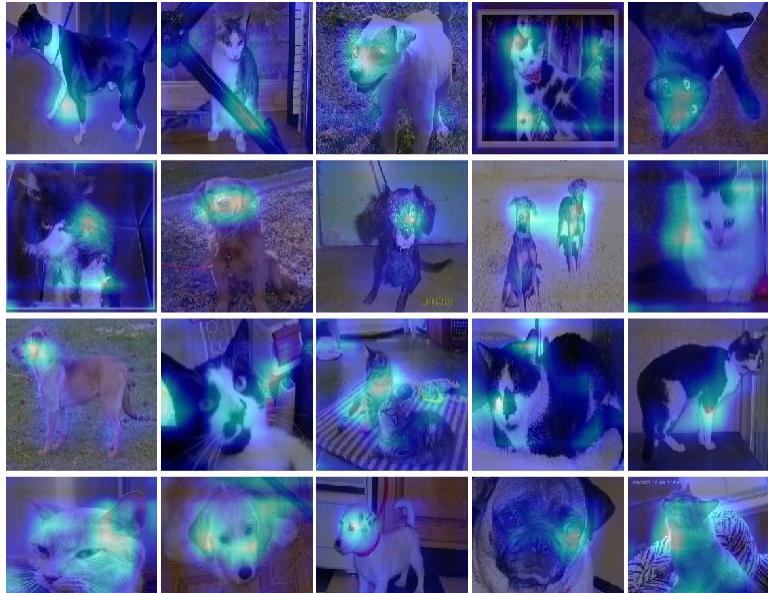


$x + \epsilon \text{sign}(\nabla_x J(\theta, x, y))$   
“gibbon”  
99.3 % confidence

[\[1412.6572\] Explaining and Harnessing Adversarial Examples](#)

# Interpretability & Explainability

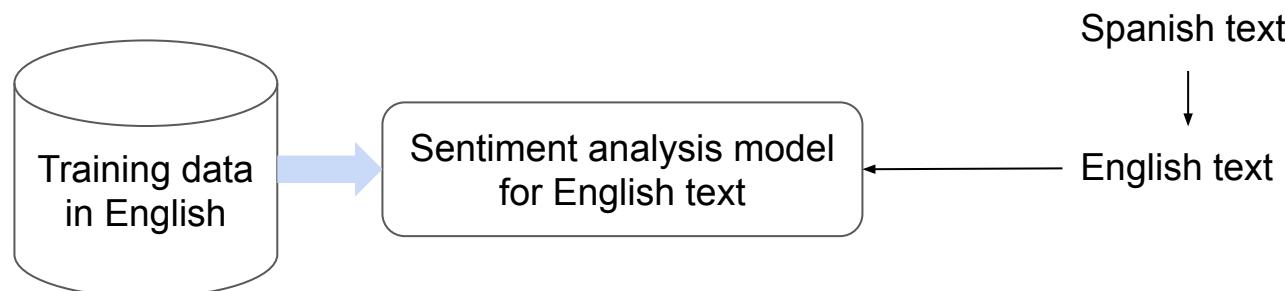
- No clear definition!
- A series of efforts that aim to provide “plausible” evidence



# Multilingual Models

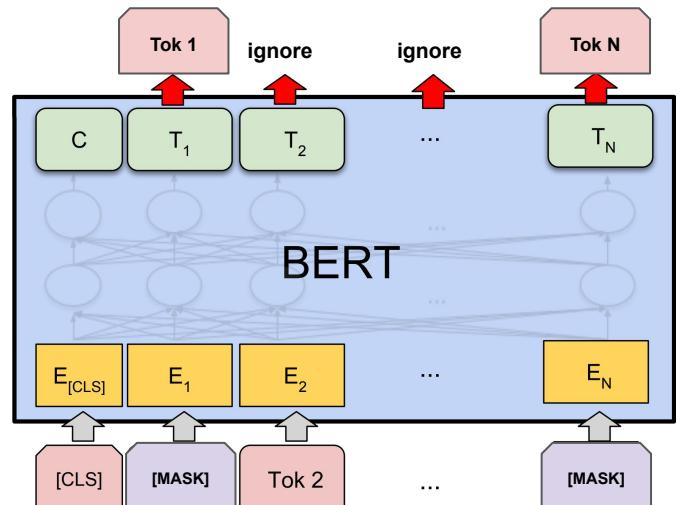
# Traditional NLP: Monolingual Models

- Models are language-specific
  - An English model can be only used for English text
- One solution to apply an English model to another language is translating the input text into English (by Machine-Translation)
  - Translation quality is bottleneck :(



# Multilingual BERT

- A single BERT model pre-trained on Wikipedia articles in 102 languages



I have a cat .      Tengo un gato .  
↑  
I [MASK] a cat .      [MASK] un gato .  
↑

# Multilingual pre-trained model: Example

<https://huggingface.co/nlptown/bert-base-multilingual-uncased-sentiment>



# Research Trend

- Task-specific models → Task-independent pre-trained models + Fine-tuning
- Monolingual models → Multilingual models
- what else?

# Research Trend

- Task-specific models → Task-independent pre-trained models + Fine-tuning
- Monolingual models → Multilingual models
- Unimodal (i.e., text) models → Multi-modal (e.g., text + image) models

# **Vision & Language**

# Visual Question Answering

- Input: Question + Image
- Output: Answer

Who is wearing glasses?

man



woman



Is the umbrella upside down?

yes



no



Where is the child sitting?

fridge



arms



How many children are in the bed?

2



1



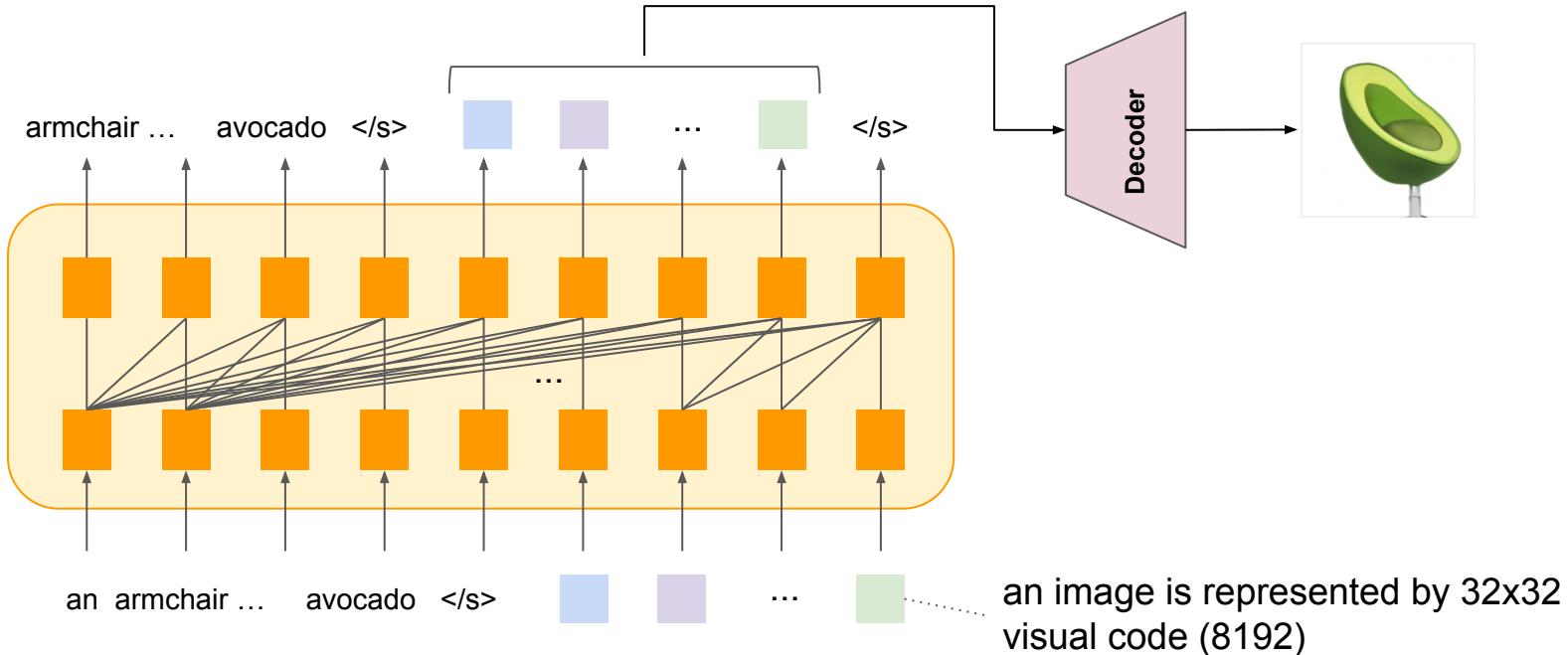
# Text-to-Image Generation

- “An armchair in the shape of an avocado”



# DALL·E

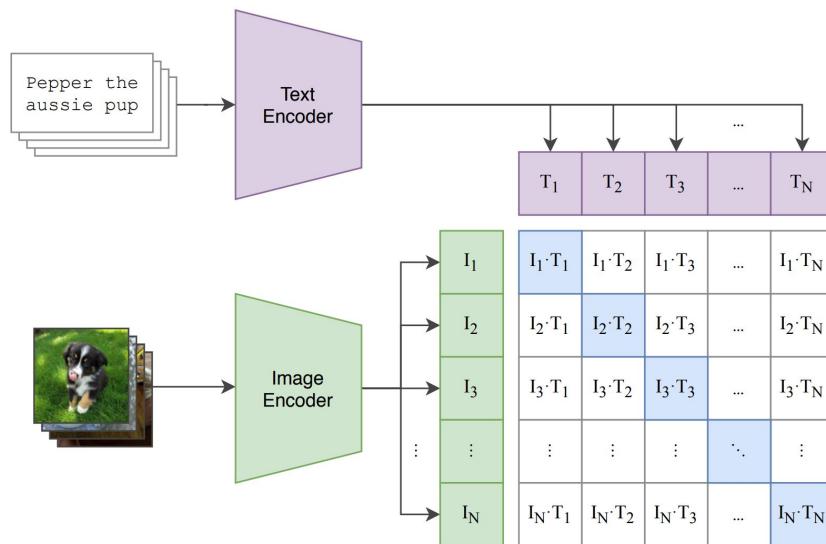
- A small GPT-3 model pre-trained with a huge number of **caption-image pairs**



# CLIP (Contrastive Language–Image Pre-training)

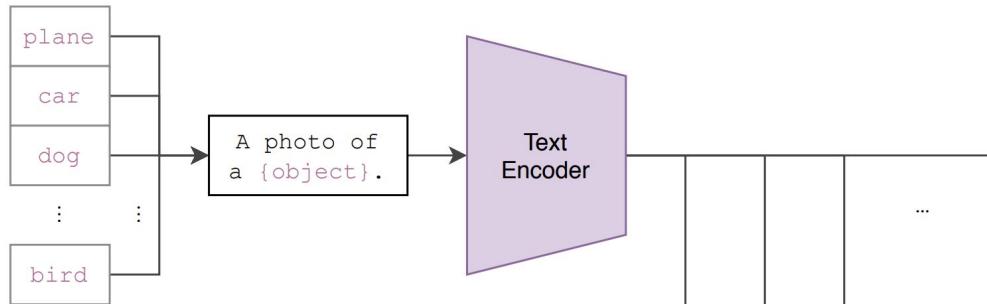
- Text Encoder (e.g., Transformer Encoder) + image encoder (e.g., Vision Transformer)
- Pre-training with a huge number of **caption-image pairs**

(1) Contrastive pre-training

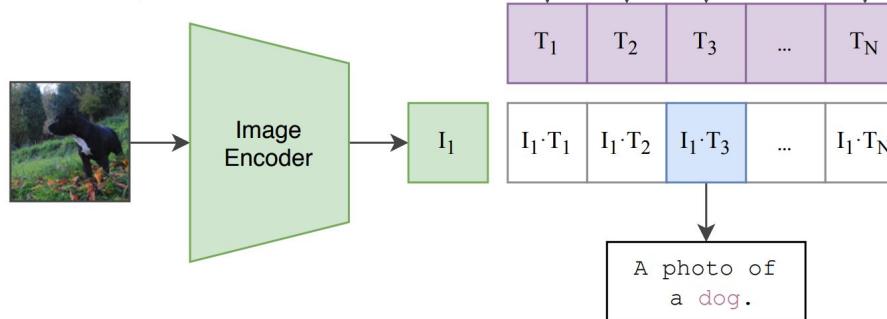


# CLIP for Zero-shot Image Classification

(2) Create dataset classifier from label text



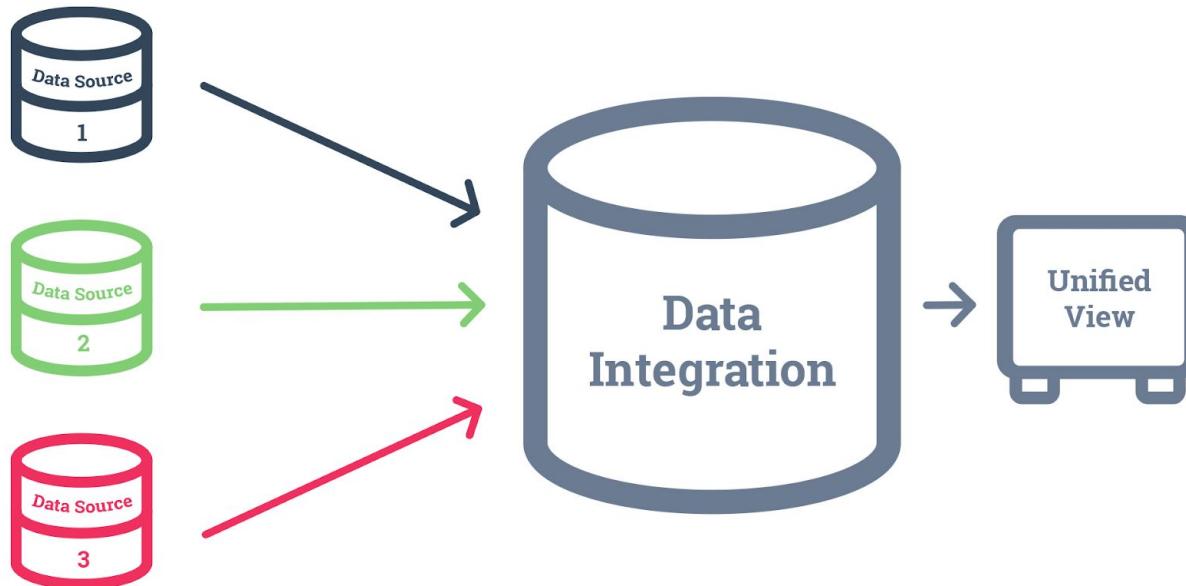
(3) Use for zero-shot prediction



# **Pre-trained LMs for Data Integration**

# Data Integration

- One of the ultimate goal of data management research



# Entity Matching

- Input: Two collections of data entries (tables, JSON files, text, ...)
- Output: all entry pairs that refer to the same entity (products, businesses, ...)

Table A:

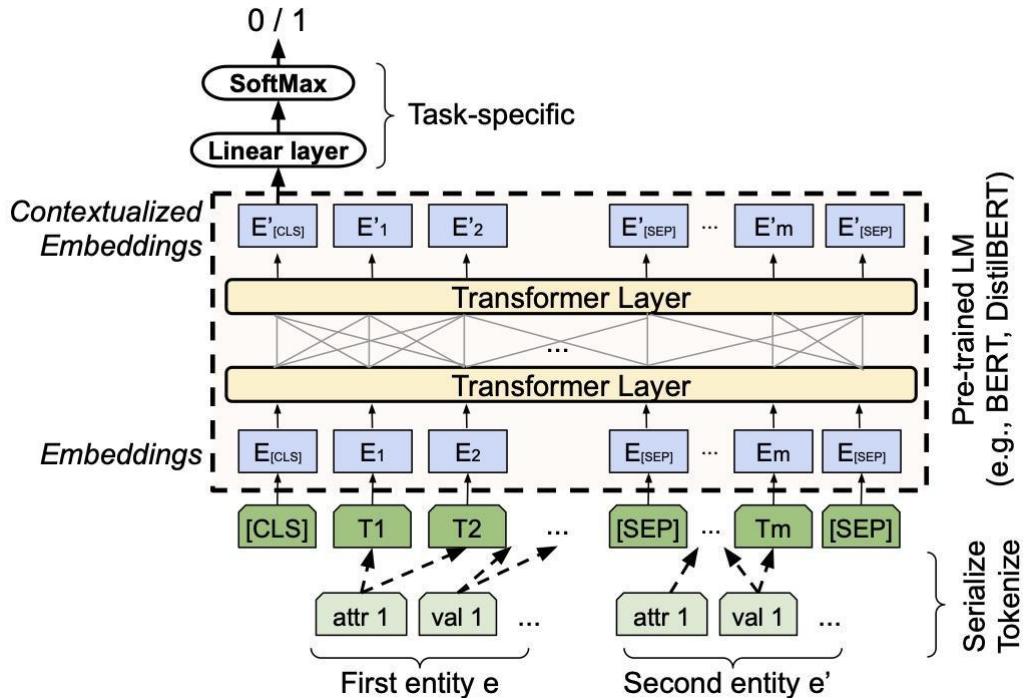
title	manf./modelno	price
<i>instant immersion spanish deluxe 2.0</i>	topics entertainment	49.99
<i>adventure workshop 4th-6th grade 7th edition</i>	encore software	19.99
<i>sharp printing calculator</i>	sharp el1192bl	37.63

Table B:

title	price
<i>instant immers spanish dlux 2</i>	36.11
<i>encore inc adventure workshop 4th-6th grade 8th edition</i>	17.1
<i>new-sharp shr-el1192bl two-color printing calculator 12-digit lcd black red</i>	56.0

[2004.00584] Deep Entity Matching with Pre-Trained Language Models

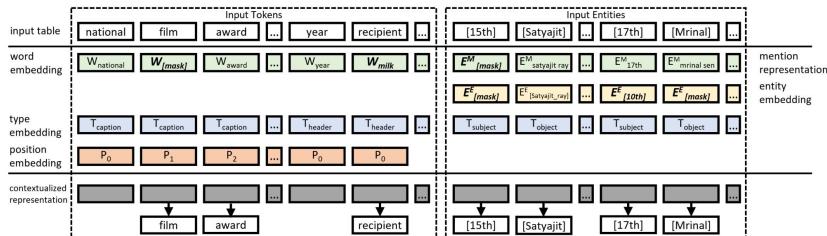
# Pre-trained LM for Entity Matching



$\text{serialize}(e) ::= [\text{COL}] \text{ attr}_1 [\text{VAL}] \text{ val}_1 \dots [\text{COL}] \text{ attr}_k [\text{VAL}] \text{ val}_k$

# Pre-trained LMs for Table Data

- TaPas [ACL '20]
- TaBERT [ACL '20]
- TURL [VLDB '21]
- TABBIE [NAACL '21]
- ...



In which city did Piotr's last 1st place finish occur?			
Year	Venue	Position	Event
R <sub>1</sub> 2003	Tampere	3rd	EU Junior Championship
R <sub>2</sub> 2005	Erfurt	1st	EU U23 Championship
R <sub>3</sub> 2005	Izmir	1st	Universiade
R <sub>4</sub> 2006	Moscow	2nd	World Indoor Championship
R <sub>5</sub> 2007	Bangkok	1st	Universiade

(A) Content Snapshot from Input Table

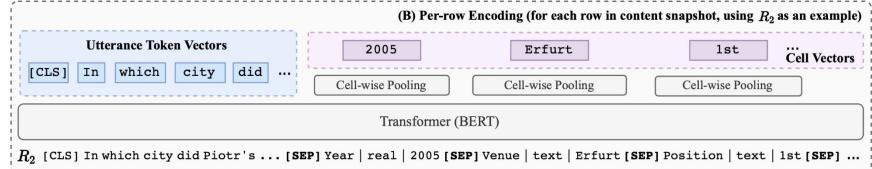


Table	col1	col2	0	1	2	3
	0	1				
	2	3				

**Token Embeddings:** [CLS], query, ?, [SEP], col, #1, col, #2, 0, 1, 2, 3

**Position Embeddings:** POS<sub>0</sub>, POS<sub>1</sub>, POS<sub>2</sub>, POS<sub>3</sub>, POS<sub>4</sub>, POS<sub>5</sub>, POS<sub>6</sub>, POS<sub>7</sub>, POS<sub>8</sub>, POS<sub>9</sub>, POS<sub>10</sub>, POS<sub>11</sub>

**Segment Embeddings:** SEG<sub>0</sub>, SEG<sub>1</sub>, SEG<sub>2</sub>, SEG<sub>3</sub>, SEG<sub>4</sub>, SEG<sub>5</sub>, SEG<sub>6</sub>, SEG<sub>7</sub>, SEG<sub>8</sub>, SEG<sub>9</sub>, SEG<sub>10</sub>

**Column Embeddings:** COL<sub>0</sub>, COL<sub>1</sub>, COL<sub>2</sub>, COL<sub>3</sub>, COL<sub>4</sub>, COL<sub>5</sub>, COL<sub>6</sub>, COL<sub>7</sub>, COL<sub>8</sub>, COL<sub>9</sub>, COL<sub>10</sub>

**Row Embeddings:** ROW<sub>0</sub>, ROW<sub>1</sub>, ROW<sub>2</sub>, ROW<sub>3</sub>, ROW<sub>4</sub>, ROW<sub>5</sub>, ROW<sub>6</sub>, ROW<sub>7</sub>, ROW<sub>8</sub>, ROW<sub>9</sub>, ROW<sub>10</sub>

**Rank Embeddings:** RANK<sub>0</sub>, RANK<sub>1</sub>, RANK<sub>2</sub>, RANK<sub>3</sub>, RANK<sub>4</sub>, RANK<sub>5</sub>, RANK<sub>6</sub>, RANK<sub>7</sub>, RANK<sub>8</sub>, RANK<sub>9</sub>, RANK<sub>10</sub>

# Break?

# **OpenAI GPT-3**

## **Very Large Language Models**

# OpenAI GPT-3

- Yet another huge GPT model (175B param), which can be used for a wide variety of downstream tasks **without fine-tuning**
- Examples
  - <https://beta.openai.com/examples>

**Examples**  
Explore what's possible with some example applications

Search...  All categories

 Chat Open ended conversation with an AI assist...	 Q&A Answer questions based on existing knowle...
 Grammar correction Corrects sentences into standard English.	 Summarize for a 2nd grader Translates difficult text into simpler concep...
 Natural language to OpenAI API Create code to call to the OpenAI API usin...	 Text to command Translate text into programmatic commands.
 English to French Translates English text into French.	 Natural language to Stripe API Create code to call the Stripe API using nat...
 SQL translate Translate natural language to SQL queries.	 Parse unstructured data Create tables from long form text
 Classification Classify items into categories via example.	 Python to natural language Explain a piece of Python code in human un...
 Movie to Emoji Convert movie titles into emoji.	 Calculate Time Complexity Find the time complexity of a function.
 Translate programming languages Translate from one programming language ...	 Advanced tweet classifier Advanced sentiment detection for a piece o...

# GPT-3 Hype

- OpenAI released GPT-3 API (almost only) to developers who are interested in creating applications using GPT-3
- Twitter timeline became full of GPT-3 advertisements
  - For example, <https://twitter.com/sharifshameem/status/1282676454690451457?s=20>

# GPT-3 Hype

- OpenAI released GPT-3 API (almost only) to developers who are interested in creating applications using GPT-3
- Twitter timeline became full of GPT-3 advertisements
  - For example, <https://twitter.com/sharifshameem/status/1282676454690451457?s=20>



# GPT-3 Takes “Training” Examples as Input

- Instead of updating the model parameters, GPT-3 **takes few training examples as input to condition** the model

Playground Q&A Save

I am a highly intelligent question answering bot. If you ask me a question that is rooted in truth, I will give you the answer. If you ask me a question that is nonsense, trickery, or has no clear answer, I will respond with "Unknown".

Q: What is human life expectancy in the United States?  
A: Human life expectancy in the United States is 78 years.

Q: Who was president of the United States in 1955?  
A: Dwight D. Eisenhower was president of the United States in 1955.

Q: Which party did he belong to?  
A: He belonged to the Republican Party.

Q: What is the square root of banana?  
A: Unknown

Q: How does a telescope work?  
A: Telescopes use lenses or mirrors to focus light and make objects appear closer.

Q: Where were the 1992 Olympics held?  
A: The 1992 Olympics were held in Barcelona, Spain.

Q: How many squigs are in a bonk?  
A: Unknown

Q:

“Training” examples



# GPT-3 Demos

<https://gpt3demo.com/>

# Future Direction & My Thoughts

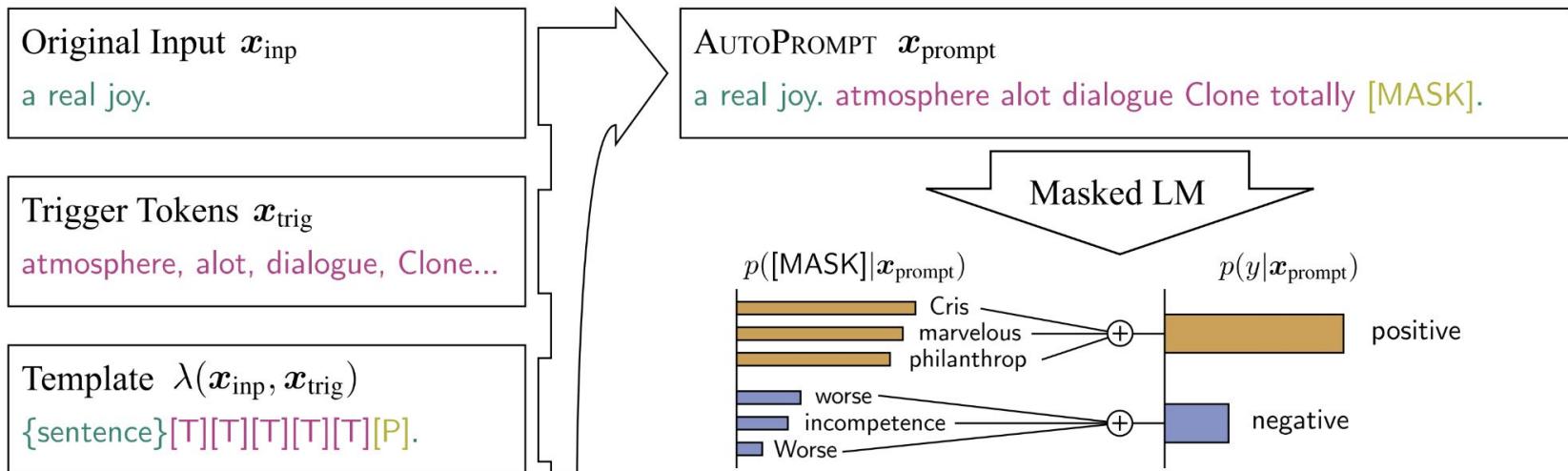
## Language Model as a Service

- Language models will keep getting larger and larger
  - So does “GPU resource” disparity :(
- Like GPT-3, those very large language models will be accessed via Web API
- Perhaps, we will stop training ML models sometime soon (?)

# Prompt Learning

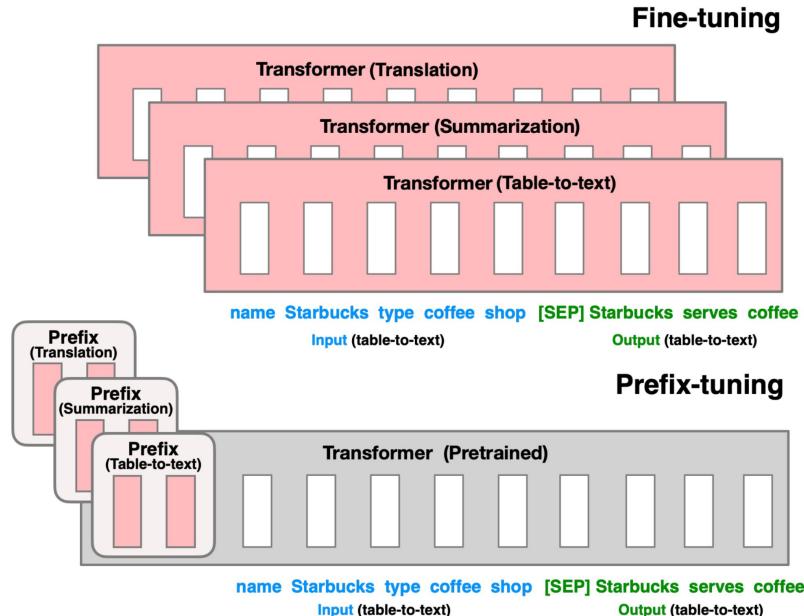
# AutoPrompt

- Inserting a few **trigger tokens** that are helpful to improve the performance



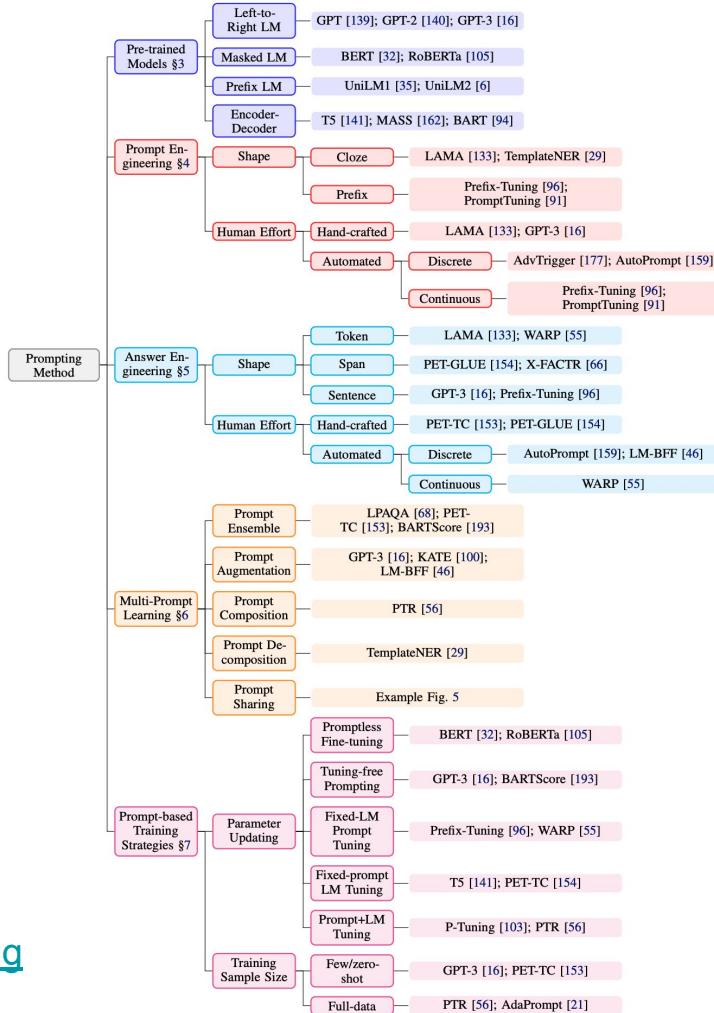
# Prefix Tuning

- Only training a small number of **continuous vectors** for each task instead of fine-tuning the pre-trained language model



# Very Hot Topic in 2021

Fine-tuning → Prompt learning



[2107.13586] Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing

# **Other Topics Not Covered By This Course**

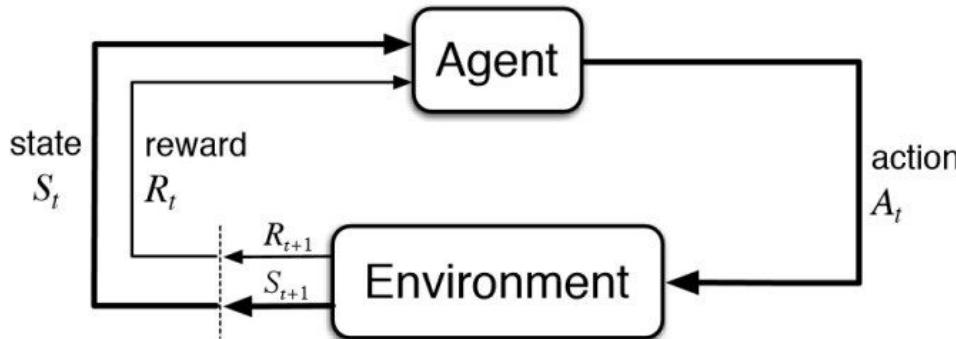
# Reinforcement Learning

- Powerful solution for robotics & agent development



# What is Reinforcement Learning?

- Update the policy of the agent based on rewards from the environment
  - cf. optimizing the objective function with/without labels



This approach is useful especially when it's **difficult to define the success/failure** (i.e., true labels) in each step

# Q-Learning: Example

- Move the agent (car) following the Q-table
- Update the Q-table once the agent receives a reward

Game Board:



Current state ( $s$ ):    **0 0 0  
0 1 0**

Q Table:

	0 0 0 1 0 0	0 0 0 0 1 0	0 0 0 0 0 1	1 0 0 0 0 0	0 1 0 0 0 0	0 0 1 0 0 0
↑	0.2	0.3	1.0	-0.22	-0.3	0.0
↓	-0.5	-0.4	-0.2	-0.04	-0.02	0.0
→	0.21	0.4	-0.3	0.5	1.0	0.0
←	-0.6	-0.1	-0.1	-0.31	-0.01	0.0

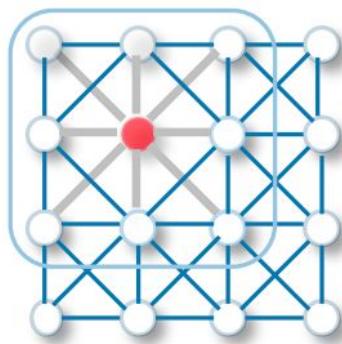
$\gamma = 0.95$

# Is Reinforcement Learning Good for NLP tasks?

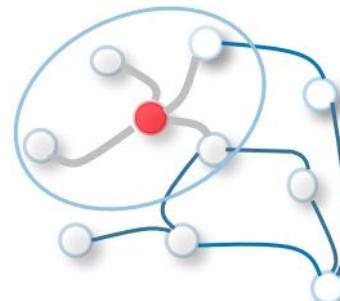
- Basically,
  - 1) we can define objective functions for NLP tasks
  - 2) we can tell if each of the system output is correct/incorrect
- What about chatbots (dialog management)?

# Graph Neural Networks

- Essentially, neural networks with **graph convolution**



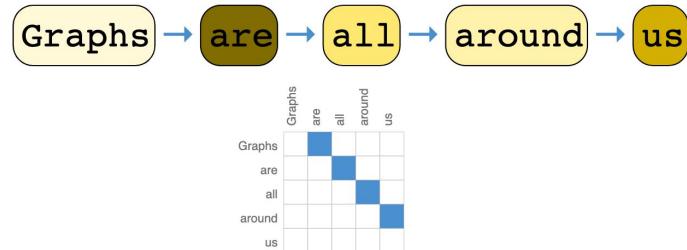
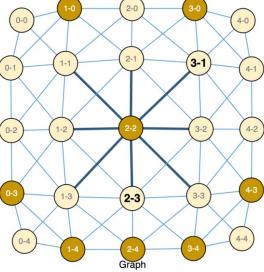
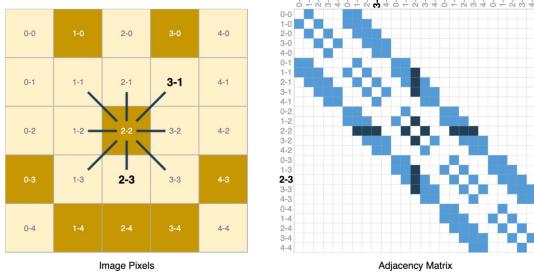
2d convolution



graph convolution

# Graph Neural Networks for Image/Text Data

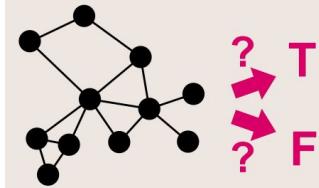
- What do you think?



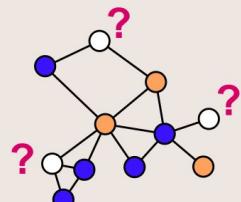
# Graph Neural Networks

- A good solution for network analysis but **NOT for NLP/CV** in my opinion

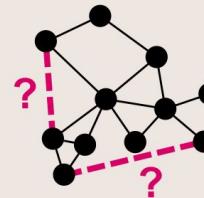
Graph Classification



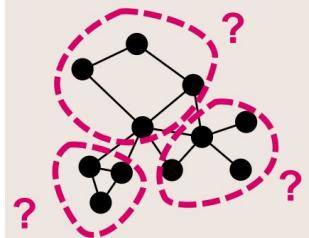
Node Classification



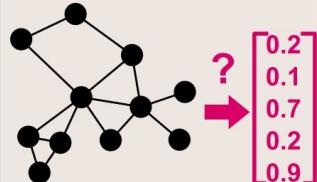
Link Prediction



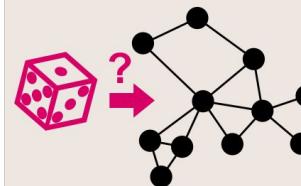
Community Detection



Graph Embedding

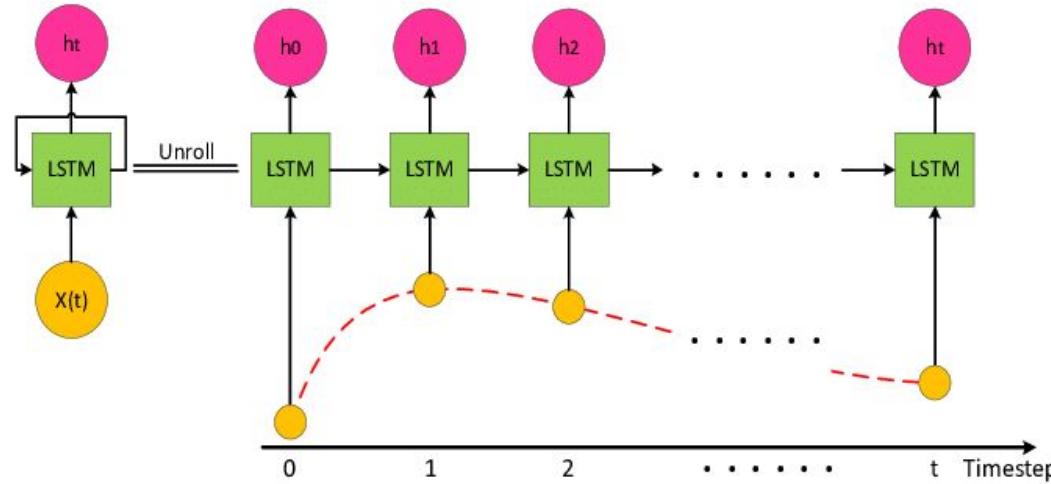


Graph Generation



# Time Series Data

- Most time series problems can be formulated as “many-to-one” or “many-to-many”-type problems
- → RNNs or Transformers

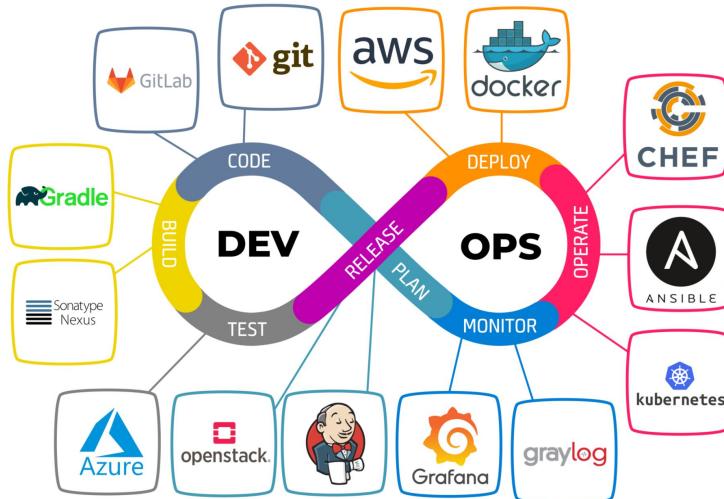


Figure

# **MLOps: Real-world Machine Learning Systems**

# DevOps

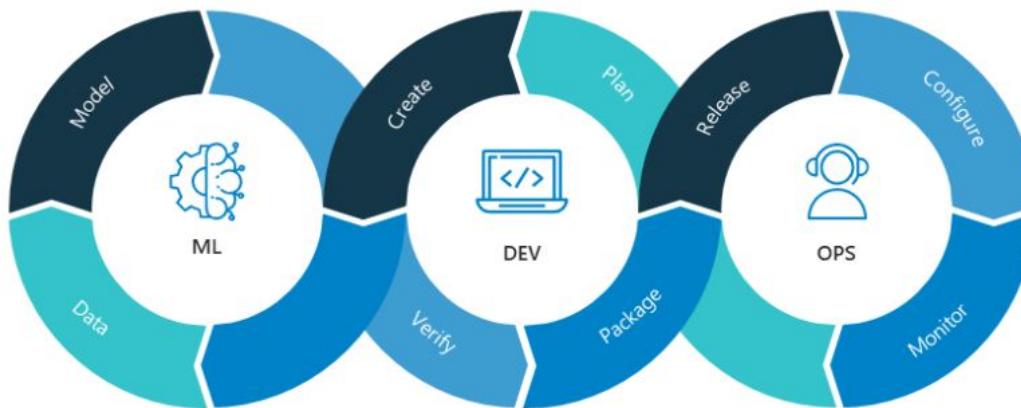
- A set of practices that combines software development (Dev) & system operations (Ops)
- To have a better software development cycle



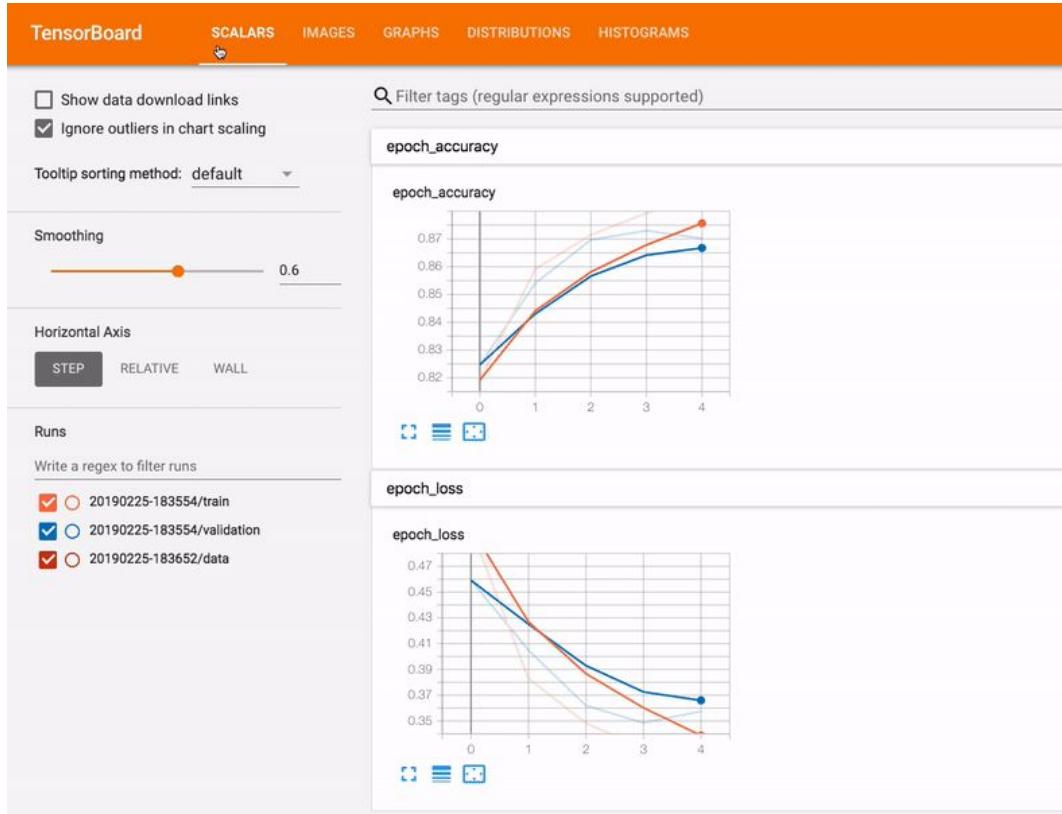
[What is DevOps and where is it applied?](#)

# MLOps

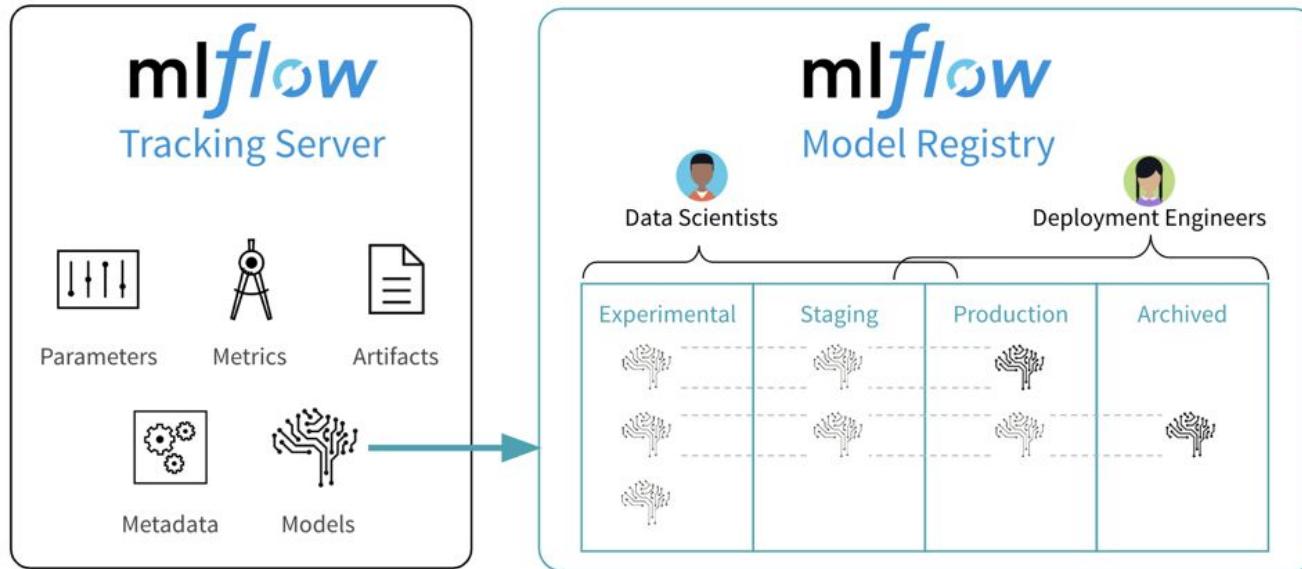
- DevOps for Machine Learning
- Key: Data & Model Management + Performance Monitor



# TensorBoard



# MLflow

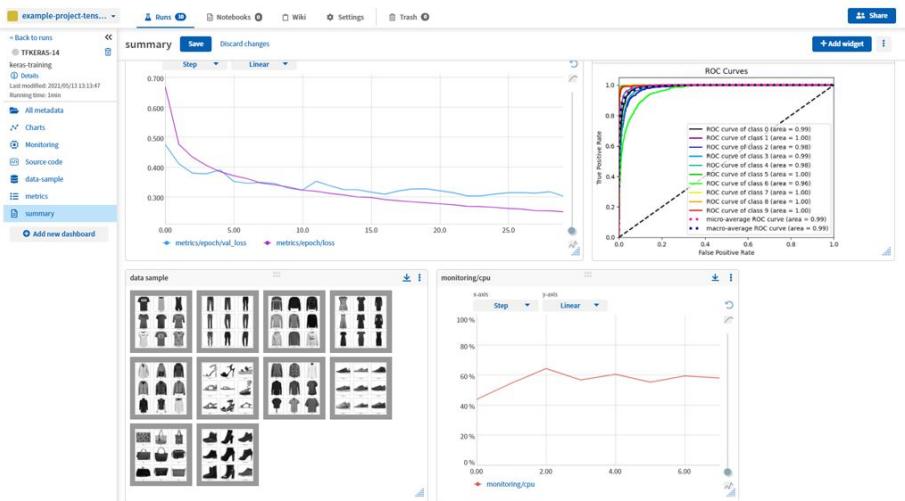


# MLflow: Tracking Server

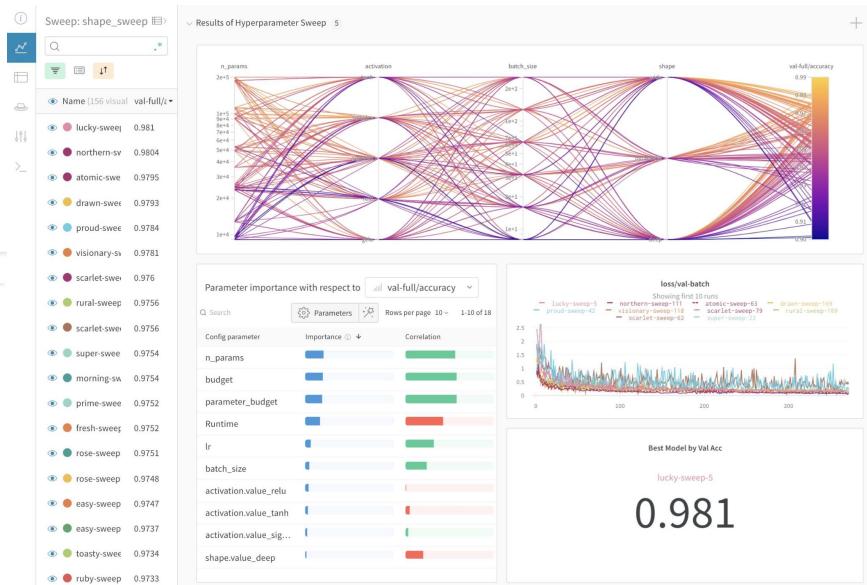
The screenshot shows the MLflow Tracking Server interface. At the top, there is a navigation bar with the 'mlflow' logo, GitHub, and Docs links. Below the navigation bar, the page title is 'diabetes1'. On the left, there is a sidebar titled 'Experiments' containing a list of experiments: Default, insurance1, health1, anomaly1, kiva1, bank1, spx\_exp, kiva\_exp, and diabetes1. The 'diabetes1' experiment is selected and highlighted with a blue background. The main content area displays the 'Experiment ID: 8' and 'Artifact Location: file:///C:/Users/moezs/pycaret-demo-td/mlruns/8'. There is a 'Notes' section with a note indicating 'None'. Below this, a search bar contains the query 'metrics.rmse < 1 and params.model = "tree" and tags.mlflow.source.type = "LOCAL"'. The search results show 'Showing 32 matching runs' with buttons for 'Compare', 'Delete', and 'Download CSV'. A table follows, showing the results of 32 runs. The columns in the table are: Start Time, Run Name, and Metrics (AUC, Accuracy, F1, Kappa, MCC, Precision, Recall, TT). The last two columns are Run ID and Run Time.

			Metrics								Tags	
	Start Time	Run Name	AUC	Accuracy	F1	Kappa	MCC	Precision	Recall	TT	Run ID	Run Time
<input type="checkbox"/>	2020-07-29 15:26:36	CatBoost Classifier	0.821	0.76	0.584	0.425	0.446	0.737	0.495	2.49	128a4c8546d6...	28.86
<input type="checkbox"/>	2020-07-29 15:26:07	Gradient Boosting Classifier	0.81	0.756	0.587	0.422	0.438	0.72	0.501	0.57	5f4bc5a973d6...	7.27
<input type="checkbox"/>	2020-07-29 15:25:59	Logistic Regression	0.778	0.75	0.595	0.42	0.428	0.682	0.534	0.22	603bf3a98e49...	3.27
<input type="checkbox"/>	2020-07-29 15:25:55	Linear Discriminant Analysis	0.804	0.773	0.63	0.471	0.483	0.738	0.555	0.02	0ad0f61fd34c...	0.94
<input type="checkbox"/>	2020-07-29 15:25:54	Ridge Classifier	0.764	0.754	0.587	0.42	0.434	0.71	0.507	0.02	861a569f0cfb4...	0.91
<input type="checkbox"/>	2020-07-29 15:25:53	CatBoost Classifier	0.817	0.763	0.598	0.438	0.453	0.725	0.516	0.38	ae9411e722cc...	34.26
<input type="checkbox"/>	2020-07-29 15:25:18	Gradient Boosting Classifier	0.794	0.752	0.594	0.421	0.431	0.694	0.523	0.25	3630b3b45d99...	4.95
<input type="checkbox"/>	2020-07-29 15:25:13	Logistic Regression	0.784	0.745	0.591	0.41	0.419	0.672	0.535	0.02	5581e006abf4...	0.96
<input type="checkbox"/>	2020-07-29 15:25:12	Linear Discriminant Analysis	0.802	0.771	0.631	0.469	0.478	0.721	0.565	0.01	82638098e9bb...	0.53
<input type="checkbox"/>	2020-07-29 15:25:11	Ridge Classifier	0	0.752	0.591	0.42	0.432	0.699	0.518	0.01	1facf88f9754e...	3.91

# Online MLOps Platforms



Neptune.ai



Weight and Biases

0.981

# Topics Not Covered But Important

- Include but not limited to...
  - Virtualization frameworks (e.g., Docker, Kubernetes)
  - Cloud services (e.g., AWS, Google Cloud, Azure)
  - ML management platforms (e.g., MLflow, WandB, Neptune)
  - Model configuration management tools (e.g., Hydra)
  - ...
- Create your own template!
  - For example, <https://github.com/ashleve/lightning-hydra-template>

~~A journey of a thousand miles begins with a single step~~  
A system of a thousand models begins with a single line of code

# PyTorch Lightning



```
PYTORCH
# models
encoder = nn.Sequential(nn.Linear(28 * 28, 64), nn.ReLU(), nn.Linear(64, 3))
decoder = nn.Sequential(nn.Linear(3, 64), nn.ReLU(), nn.Linear(64, 28 * 28))

encoder.cuda(0)
decoder.cuda(0)

# download on rank 0 only
if global_rank == 0:
    mnist_train = MNIST(os.getcwd(), train=True, download=True)

# split dataset
transform=transforms.Compose([
    transforms.ToTensor(),
    transforms.Normalize(0.5, 0.5)])
mnist_train = MNIST(os.getcwd(), train=True, download=True, transform=transform)

# train (55,000 images), val split (5,000 images)
mnist_train, mnist_val = random_split(mnist_train, [55000, 5000])

# The dataloaders handle shuffling, batching, etc...
mnist_train = DataLoader(mnist_train, batch_size=64)
mnist_val = DataLoader(mnist_val, batch_size=64)

# optimizer
params = [encoder.parameters(), decoder.parameters()]
optimizer = torch.optim.Adam(params, lr=1e-3)

# TRAIN LOOP
model.train()
num_epochs = 1
for epoch in range(num_epochs):
    for train_batch in mnist_train:
        x, y = train_batch
        x = x.cuda(0)
        x = x.view(x.size(0), -1)
        z = encoder(x)
        x_hat = decoder(z)
        loss = F.mse_loss(x_hat, x)
        print('train loss:', loss.item())

        loss.backward()
        optimizer.step()
        optimizer.zero_grad()

# EVAL LOOP
model.eval()
with torch.no_grad():
    val_loss = []
    for val_batch in mnist_val:
        x, y = val_batch
        x = x.cuda(0)
        x = x.view(x.size(0), -1)
        z = encoder(x)
        x_hat = decoder(z)
        loss = F.mse_loss(x_hat, x)
        val_loss.append(loss)
    val_loss = torch.mean(torch.tensor(val_loss))
model.train()
```

PYTORCH LIGHTNING

Turn PyTorch into Lightning

Lightning is just plain PyTorch



# Hands-on Session: PyTorch Lightning

- [Google Colab](#)

# Summary

- Advanced topics & Future directions
  - Data Augmentation
  - Adversarial Attacks; Interpretability & Explainability
  - Multilingual Models
  - Vision & Language
  - Pre-trained Language Models for Data Integration
  - Open AI GPT-3
  - Prompt Learning
- Topic that are not covered by the course
  - Reinforcement Learning
  - Graph Neural Networks
  - Time Series Data
  - MLOps

# Project Presentation (5min + 3min QA)

Tue 11/23 & Tue 11/30

- All students must attend both of the sessions
  - Join the zoom link on time (I'll start the zoom 5 minutes before the session)
  - The first presentation will **start at 2pm sharp**
- 
- Presenters will share their screens to make presentations (Camara On)
  - Presentations will be recorded