(Start recording before moving to the next slide)

Welcome

Course n. 097215

Natural Language Processing

Introduction

March 2025, 097215

Eyal Ben-David

An LLM Welcomes You All to the Course

(We asked GPT4** to write)
“The opening words of an NLP course:”

Welcome to the fascinating world of Natural Language Processing, or NLP as we'll often refer to it. Here at the Technion, we pride ourselves on providing courses that not only impart knowledge but also cultivate curiosity, and NLP is one domain where there's no end to the wonders we can uncover.

Imagine a world where machines understand, process, and even generate human language. Well, that's not a distant future or science fiction anymore. We're living it! Every time you ask your smart speaker a question, get a recommendation from a chatbot, or even when you're translating languages
on-the-go, NLP is at work.
...

** More on LLMs (and many other cool NLP models) later:)

How did it look “not long ago”...

(We asked GPT2** to write)
“The opening words of an NLP course:”

Your course can teach you how to do data mining from your native country, or to learn the science-fiction tropes the media loves to play upon. Or even how to think about building applications that have intelligence and power in them.

To put this into context, this is quite a big deal. Data is incredibly big, so if you can't understand how it can be used to solve problems, then you're more likely to end up with the same problems you did at the university.

My first course ever involved students talking to a bot about all the things that used to surprise me: cars, computers, and even XXX. It was a fascinating opportunity for my students to see a different side of what I did, so that they could understand why I was designing an app that could be used by them today...

We made some progress...

“The opening words of an NLP course:”

Your course can teach you how to do data mining from your native country, or to learn the science-fiction tropes the media loves to play upon. Or even how to think about building applications that have intelligence and power in them.

To put this into context, this is quite a big deal. Data is incredibly big, so if you can't understand how it can be used to solve problems, then you're more likely to end up with the same problems you did at the university.

My first course ever involved students talking to a bot about all the things that used to surprise me: cars, computers, and even XXX. It was a fascinating opportunity for my students to see a different side of what I did, so that they could understand why I was designing an app that could be used by

(We asked GPT4** to write)
“The opening words of an NLP course:”

Welcome to the fascinating world of Natural Language Processing, or NLP as we'll often refer to it. Here at the Technion, we pride ourselves on providing courses that not only impart knowledge but also
cultivate curiosity, and NLP is one domain where there's no end to the wonders we can uncover.

Imagine a world where machines understand, process, and even generate human language. Well, that's not a distant future or science fiction anymore. We're living it! Every time you ask your smart speaker a question, get a recommendation from a chatbot, or even when you're translating languages
on-the-go, NLP is at v
...

What’s next?

Will we reach Artificial General Intelligence (AGI)?

Final Project Assignment

Find a textual task that LLMs struggle with

Go play with models and find where they need to improve
https://aistudio-preprod.corp.google.com/prompts/new_chat

מתמטיקה רבה.
פיזיקה, כאשר אומר לו שהוא טועה
- שמשנה את תשובתה.
היוריסטיקה.

Today’s Agenda

- **Preliminaries**
- What is *NLP*?
- NLP tasks
- What is ML
- Why is ML useful for NLP

Technicalities

- **Welcome to *NLP** 2025 Spring!**
- **Course staff:**
  - Eyal Ben-David (lecturer)
  - Eilam Shapira (TA in charge)
  - Shani Pais (TA)
- **Moodle**

  Please direct all academic inquiries here.
- **My email:** eyalbd12@technion.ac.il

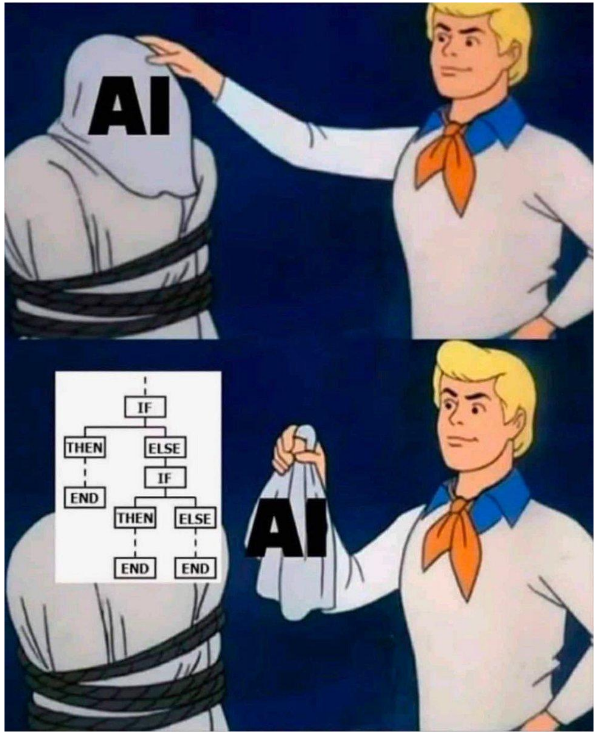
  Please direct **important** personal inquiries here.

A bit about myself

- **Completed my Ph.D. studies @ Technion**
- **Research scientist @ Google**
  - Start using Gemini :)
- **Background in Electrical Engineering**
- **This is the third I am running this class**

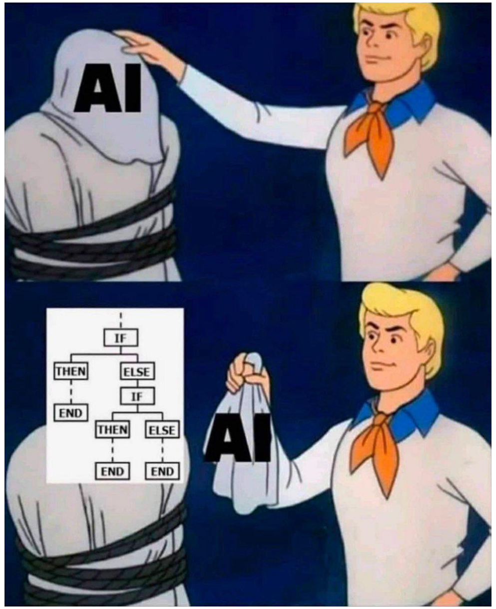
Goal

- **Our main goal – differentiate you from this:**

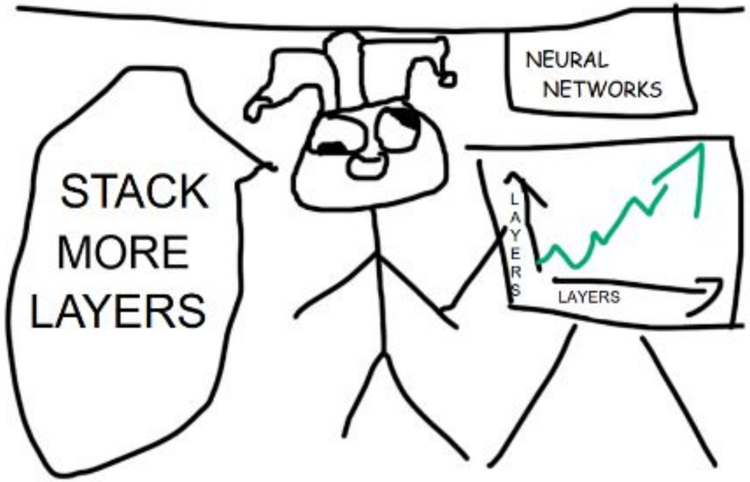


Goal

- Our main goal –
- differentiate you from this:

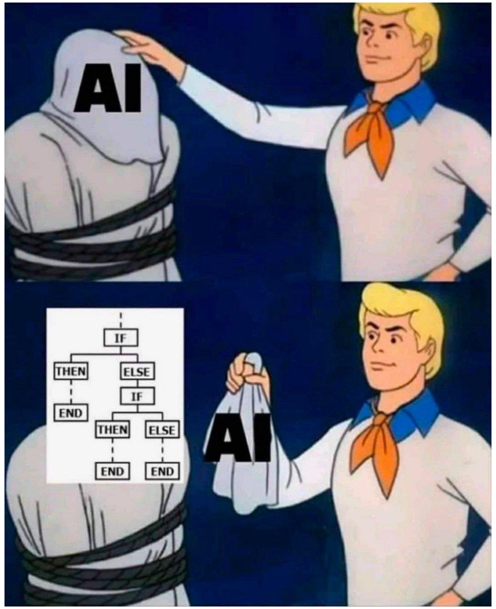


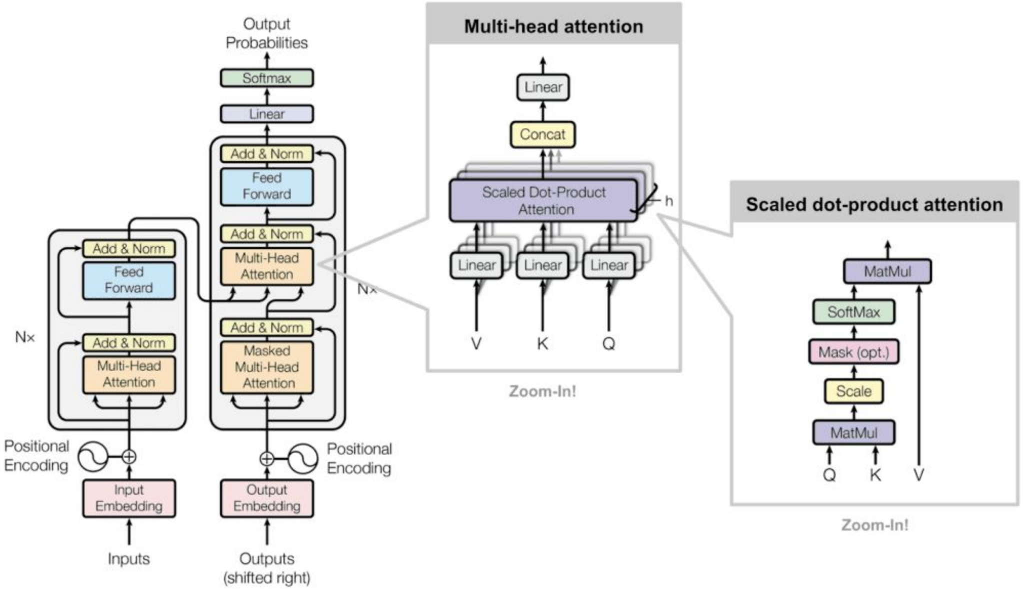
- but also from this:



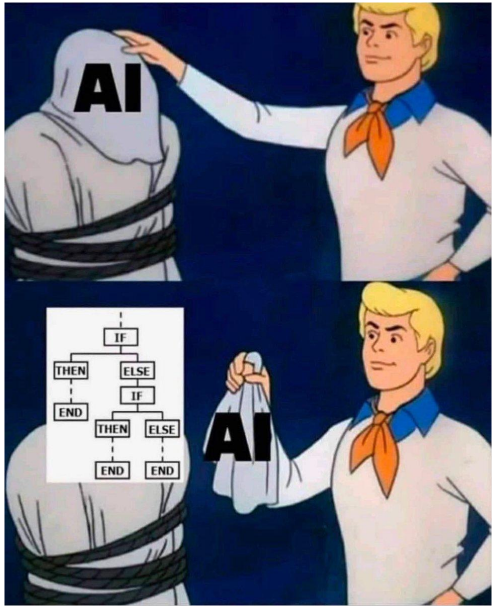
Goal

Understand how/why NLP evolved

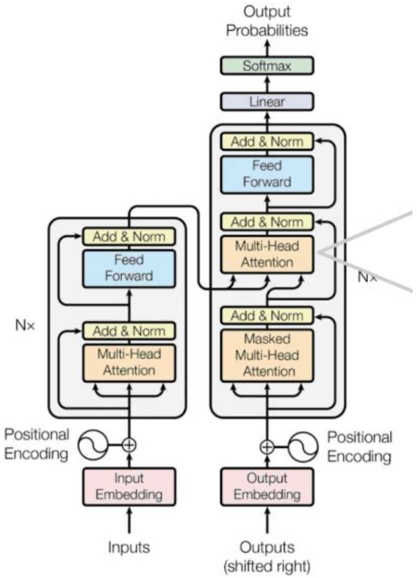




Goal



AI
IF
THEN
ELSE
IF
END
THEN
ELSE
END
END
AI



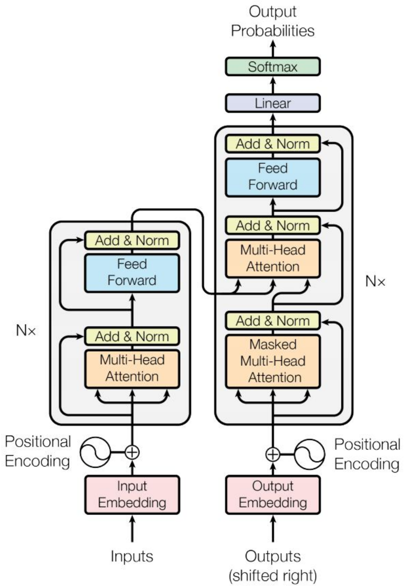
Output Probabilities
Softmax
Linear
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
Add & Norm
Masked Multi-Head Attention
Add & Norm
Feed Forward
Add & Norm
Multi-Head Attention
N×
N×
Positional Encoding
Positional Encoding
Input Embedding
Output Embedding
Inputs
Outputs (shifted right)

?

Goal

Two main pillars

1. Modeling



2. Coding:

```python
# Welcome PyTorch-Transformers (formely known as pytorch-pretrained-bert)!
import torch
from pytorch_transformers import *

# Simple and standard API for 6 transformer architectures & 27 pretrained model weights:
MODELS = [(BertModel,       BertTokenizer,      'bert-base-uncased'),
          (OpenAIGPTModel,  OpenAIGPTTokenizer, 'openai-gpt'),
          (GPT2Model,       GPT2Tokenizer,      'gpt2'),
          (TransfoXLModel,  TransfoXLTokenizer, 'transfo-xl-wt103'),
          (XLNetModel,      XLNetTokenizer,     'xlnet-base-cased'),
          (XLMModel,        XLMTokenizer,       'xlm-mlm-enfr-1024')]

# Let's encode some text in a sequence of hidden-states using each model:
for model_class, tokenizer_class, pretrained_weights in MODELS:
    # Load pretrained model/tokenizer
    tokenizer = tokenizer_class.from_pretrained(pretrained_weights)
    model = model_class.from_pretrained(pretrained_weights)

    # Encode text
    input_ids = torch.tensor([tokenizer.encode("Here is some text to encode")])
    last_hidden_states = model(input_ids)[0]  # Models outputs are now tuples

# Models can return full list of hidden-states & attentions weights at each layer
model = model_class.from_pretrained(pretrained_weights,
                                    output_hidden_states=True,
                                    output_attentions=True)
input_ids = torch.tensor([tokenizer.encode("Let's see hidden-states and attentions")])
all_hidden_states, all_attentions = model(input_ids)[-2:]

# Models are compatible with Torchscript
model = model_class.from_pretrained(pretrained_weights, torchscript=True)
traced_model = torch.jit.trace(model, (input_ids,))

# Simple serialization for models and tokenizers
model.save_pretrained('./directory/to/save/')  # save
model = model_class.from_pretrained('./directory/to/save/')  # re-load

# SOTA examples for GLUE, SQUAD, text generation...
```

Technicalities

- **Our course is updating:**
  - We are adding new topics
- **Working hard: Lots of new concepts and HWs**
  - This semester we have a midterm exam, 2 wet HWs and a mini project
- **But will be worth the extra effort (all around). At the project, you can perform top-notch research (or settle on a humble project)**
- **On our end: Working hard to make learning smooth**
- **On your end: Be active from beginning. Start working on your HW early (!!!)**

Technicalities

- **Our goals for students:**
  - Understand **traditional** NLP approaches
  - Understand **modern** NLP inside-out and outside-in
  - Acquire (NLP) terminology
  - Know how things work (model), and to make things that work (code)
  - Be immune to the hype

- **NLP is an insanely fast-moving field – we will try to distill the important stuff!**

Grading

Grading policy:

- **Exercises:** 40% of final grade (20% each)
  - Implementing a well defined algorithm.

- **Midterm:** 20% of final grade
  - ~20 questions

- **Mini project:** 40% of final grade
  - Perform (small) research

**** Exercises are in pairs/singles. Don't even ask for triplets.**

Lectures

- Lectures will be **recorded**.
- **We highly recommend attending to class.**

Today’s Agenda

- Preliminaries
- **What is *NLP*?**
- NLP tasks
- What is ML
- Why is ML useful for NLP

What is Natural Language Processing?

*A **field of ML** that tries to give **machines the ability to understand human language as it is spoken and written**.*

It is a field in the intersection of:

- Computer science
- Artificial intelligence
- Linguistics

Applications that use NLP?

- Simple: spelling correction, text categorization

- Complex: speech recognition, machine translation, dialog, question answering, personal assistance

- Unknown: human-level comprehension (is that just NLP?)

How do we do NLP?

- 1950 -- ~1990s → Linguistic focus, Write many rules, semantics, syntax

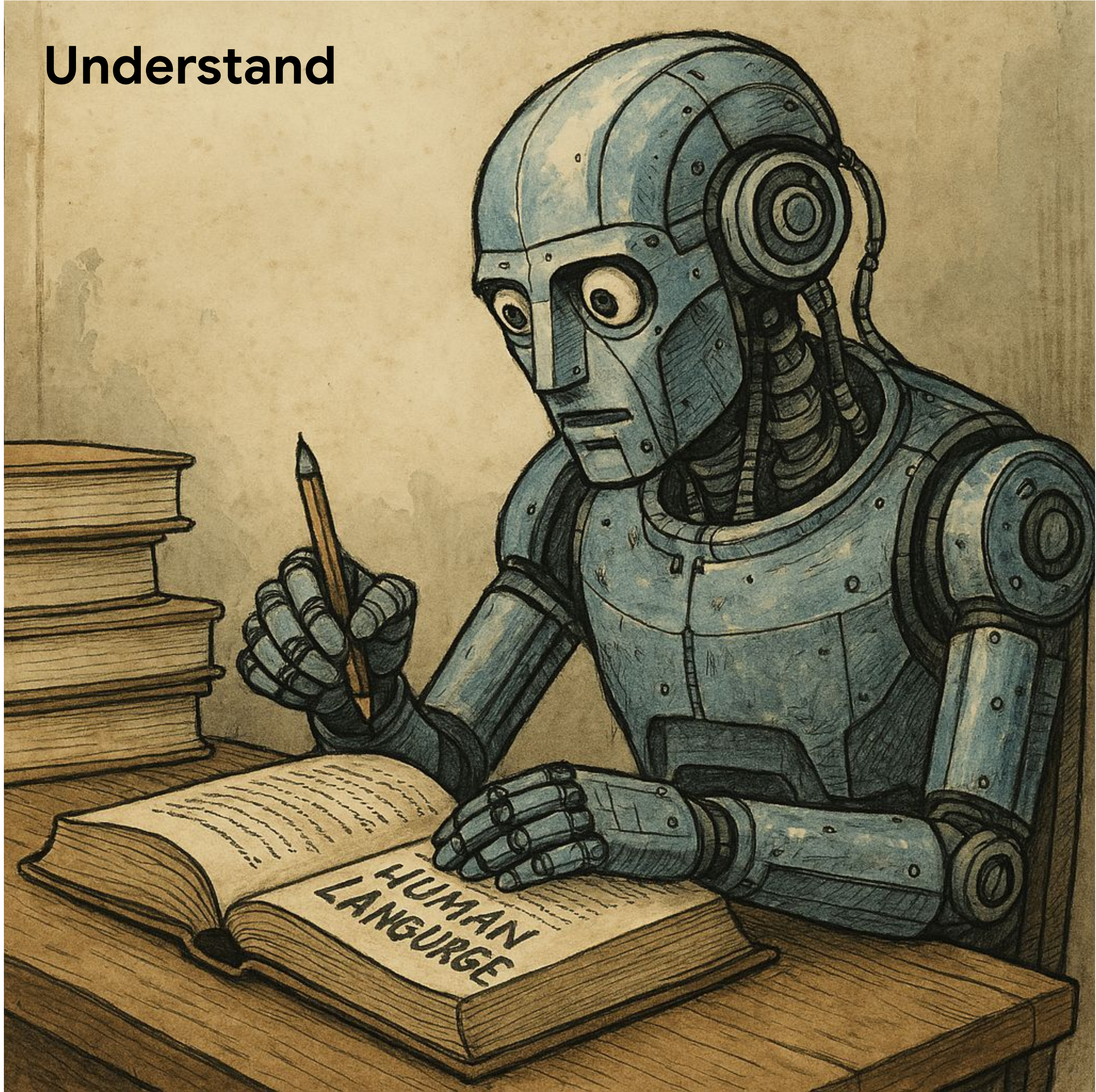
Large data

- 1990 -- ~2000 → Corpus-based statistics, Empirical revolution
- 2000 -- 2013 → Supervised machine learning
- ~2013 -- today → “deep learning!#@$!@”

How do we do NLP?

- **1950 -- ~1990s ⟶ Linguistic focus, Write many rules, semantics, syntax**
- **1990 -- ~2000 ⟶ Corpus-based statistics, Empirical revolution**
- **2000 -- 2014 ⟶ Supervised machine learning, scale to more data**
- **~2013 -- today ⟶ “deep learning”**
  - **2013-2018⟶** MLP(FFNN)/CNN/**RNN**
  - **2018 ⟶ Transformers (Attention-based models)**

Core Challenge



Understanding natural language

Human language is a system that allows human to express thoughts and communicate.

We use words to express and communicate. But words may have several meanings.

Date

“Date of birth”

Understanding human language

We combine words to form expressions and sentences.

“Bring to the table.”

“Are you out of your mind?”

“Calling the shots.”

Understanding human language

And it gets harder with sentences, documents...

Understanding human language

If that's not enough:

- **There are more than 7000 spoken languages.**
- **Different dialects.**
- **People speak and express themselves one differently.**
  - formal, informal, convince, tweet, talk to a friend.
- **To understand each other, we use world knowledge.**
- **Language is changing over time.**

Levels of Language Analysis

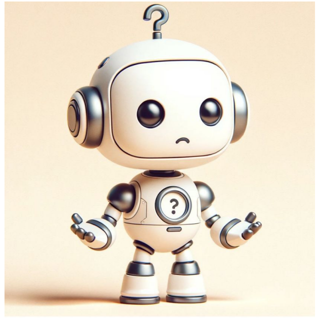
- Phonetics (how sounds are made)
- Phonology (how sounds can combine)
- Morphology (how words are built)
- Syntax (how words are combined)
- Semantics (the meaning of words/phrases)
- Pragmatics (the true contextual meaning of words/phrases)
- Discourse (structure/meaning across sentences)

In this Course:

- Phonetics (how sounds are made)
- Phonology (how sounds can combine)
- Morphology (how words are built)
- Syntax (how words are combined)
- Semantics (the meaning of words/phrases)
- Pragmatics (the true contextual meaning of words/phrases)
- Discourse (structure/meaning across sentences)

As you can see, language is hard...

I am a huge metal fan!



Today's Agenda

- Preliminaries
- What is *NLP*?
- **NLP tasks**
- What is ML
- Why is ML useful for NLP

What type of tasks are we facing?

Throughout the course we will visit a variety of tasks:

- Traditional taks
- Real applications
- Tasks that could not be addressed until recently

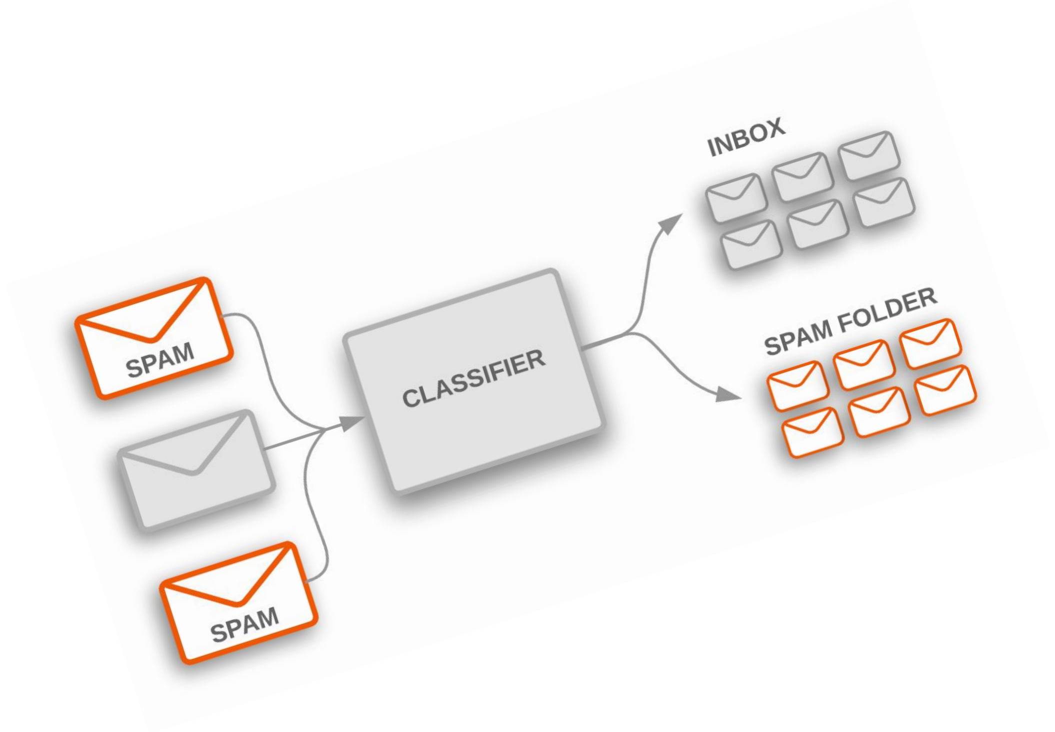
Sequence/Text Classification

Input:

- Text T
- A set of classes $C = \{c_1, c_2, \dots, c_n\}$

Output:

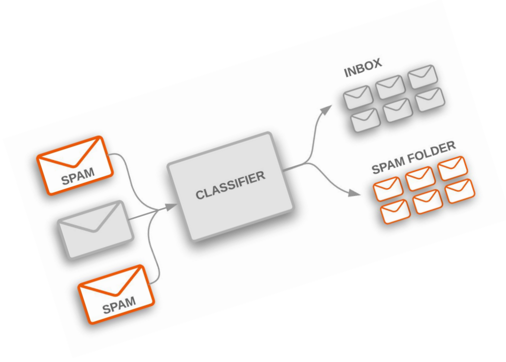
- A predicted class $c \in C$



Common Text Classification Tasks

- Sentiment Analysis
- Spam detection
- Natural Language Inference
- Stance detection

- Etc.



Token Classification

Input:

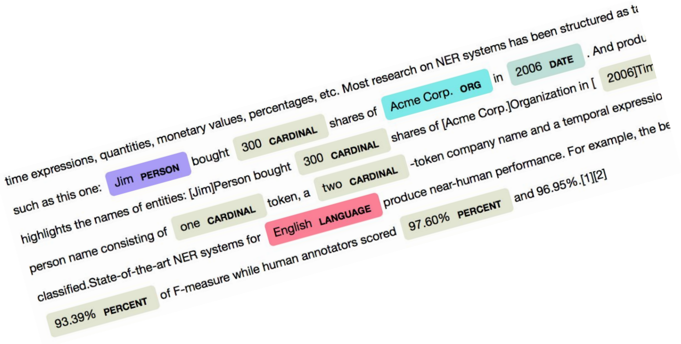
- Tokens/words $T=[t_1, t_2, \ldots, t_k]$
- A set of classes $C = \{c_1, c_2, \ldots, c_n\}$

Output:

- Predicted classes per token $\hat{C}=[\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_k]$, $\hat{c}_i \in C$ and $i \in \{1, \ldots, k\}$

Common Token Classification Tasks

- Part of Speech (POS) Tagging
- Named Entity Recognition (NER)
- Aspect Detection
- Dependency Parsing*
- Question Answering**
- Etc.



* The label of a token is one of the other tokens

** Depends on how we define the task.

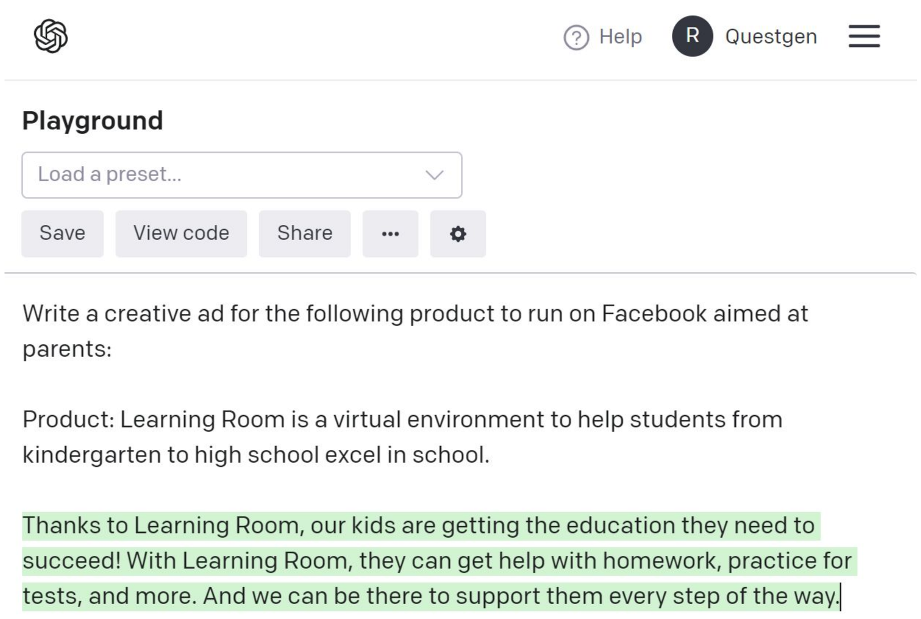
Text Generation

Input:

- Text T
- Vocabulary $V = \{v_1, v_2, \ldots, v_n\}$

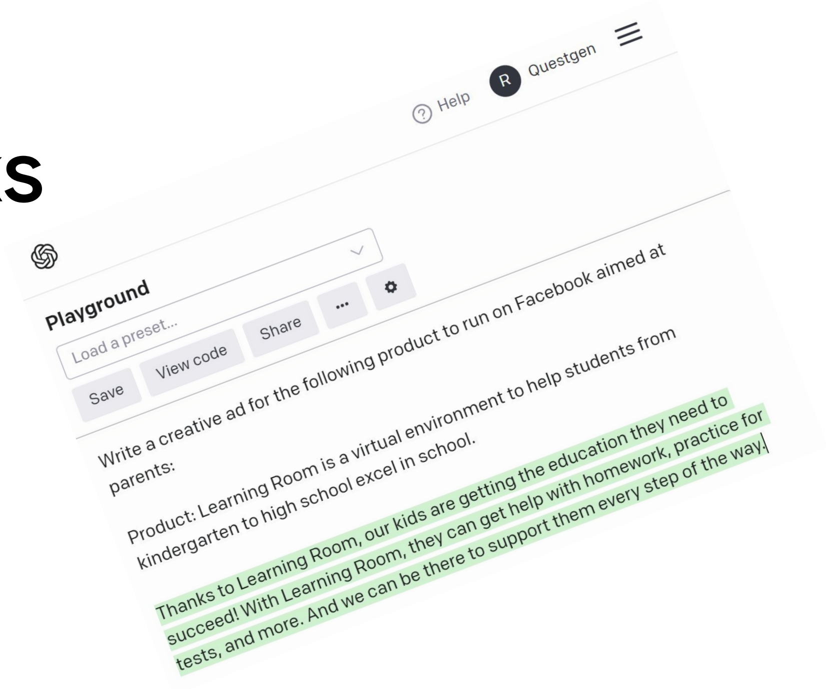
Output:

- Generated text $\hat{T}=[\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_k], \hat{t}_i \in V$ and $i \in \{1, \ldots, k\}$



Common Text Generation Tasks

- Text Summarization
- Machine Translation
- Dialogue Generation
- Question Answering*

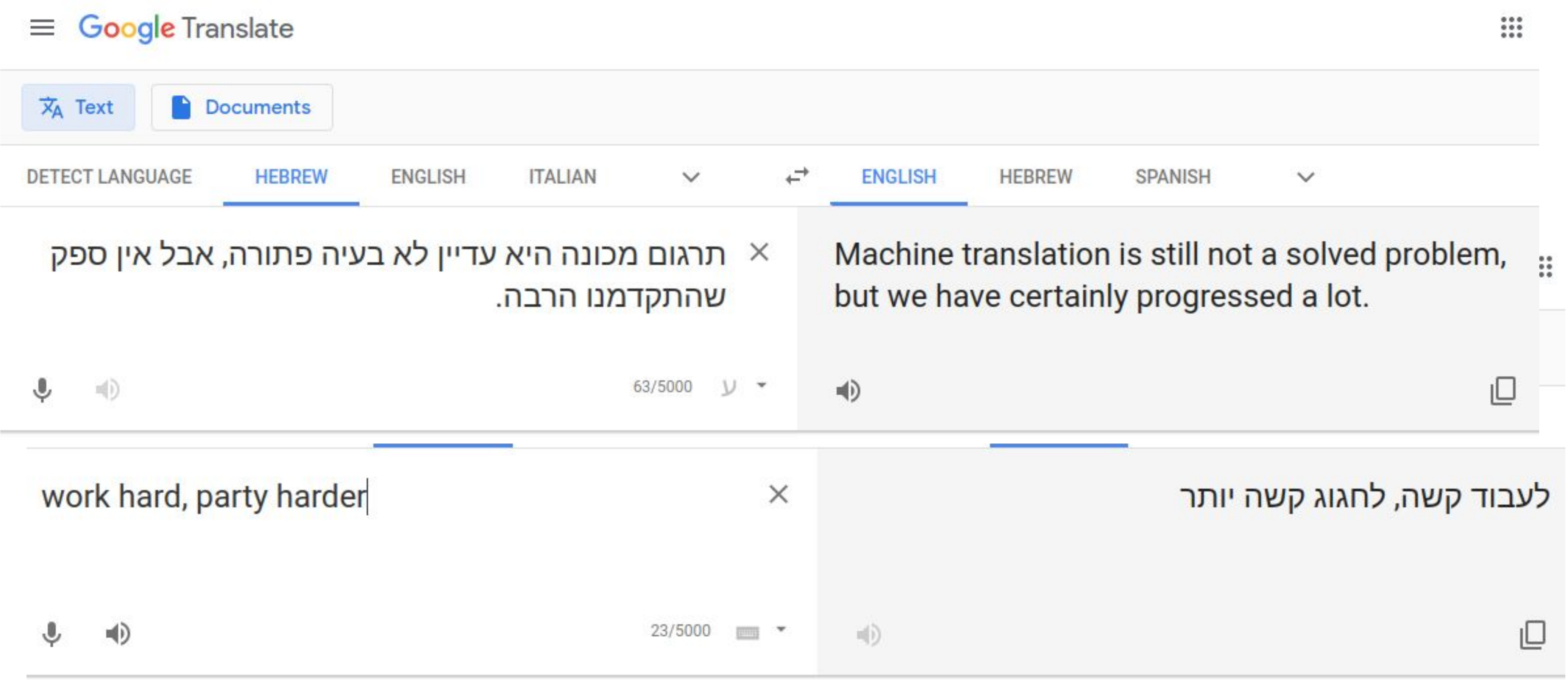


- Actually, every task can be addressed through text generation**

* Depends on how we define the task.

** more on that later in the course:).

Machine Translation



Another way to look at it

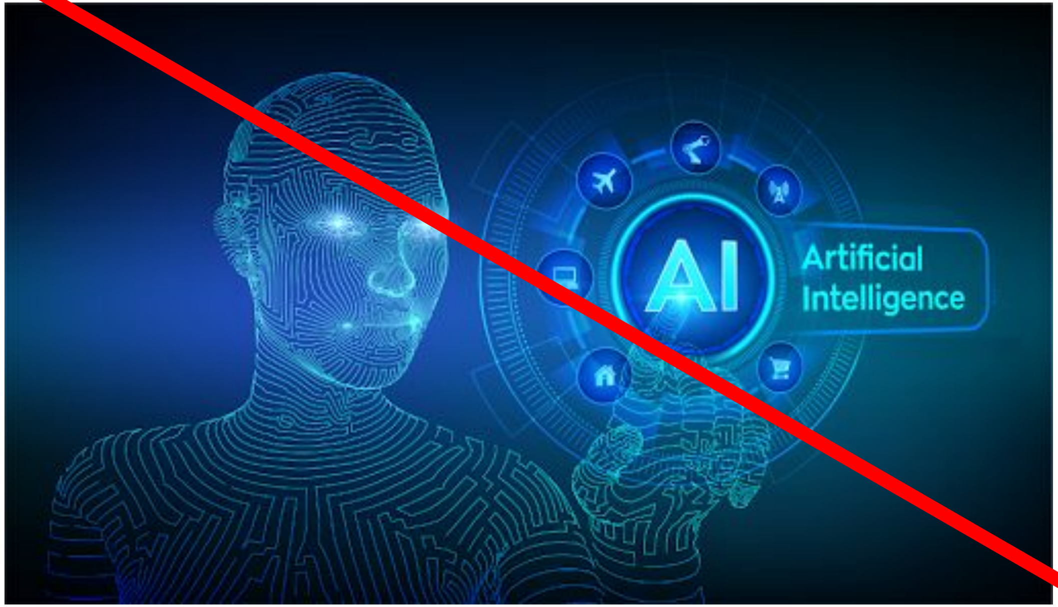
We can roughly divide NLP tasks into:

1. **Syntax**
    - Part-of-speech tagging, dependency parsing
2. **Semantics**
    - Machine translation, named entity recognition, sentiment analysis, topics...
3. **Discourse**
    - Summarization, coreference resolution, question answering, dialog
4. **Speech**
    - speech recognition, text-to-speech, recognizing person

Today’s Agenda

- Preliminaries
- What is *NLP*?
- NLP tasks
- **What is ML**
- Why is ML useful for NLP

What is machine learning?



What is machine learning?

- **(Too) many distinctions:**
  - Online/Offline
  - Unsupervised/Semi-supervised/Supervised
  - Classification/Regression/Structured Prediction
  - Discriminative/Generative
  - In-distribution/Out-of-distribution

- **We'll untie this in future classes, today let's start simple…**

What is machine learning?

ML solution:

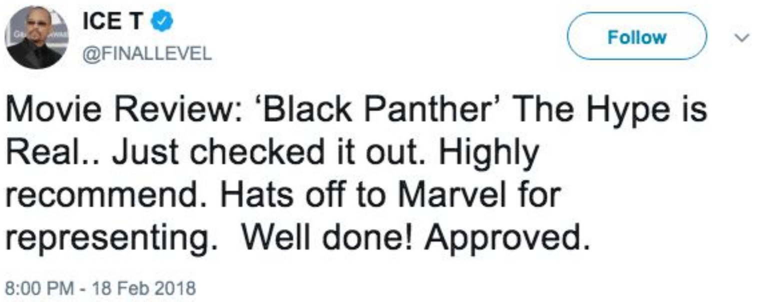
- Model
- Training
- Inference

suppose you want to know:

<u>Task</u>: What do people think about this movie?

How to determine the sentiment of the review

- **Model: a rule that determines whether the review is positive or negative**

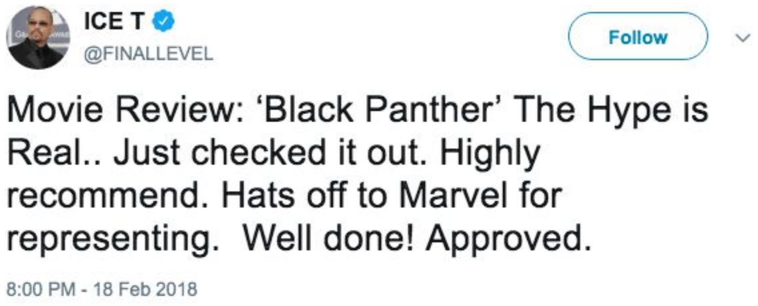


rule

How to determine the sentiment of the review

- **Training: Designing the rule**

  For instance: More positive words then negative words → Positive review



rule

How to determine the sentiment of the review

- Inference: Use the rule to decide if the review is positive or negative

rule

How to determine the sentiment of the review


ICE T
@FINALLEVEL
Follow
Movie Review: 'Black Panther' The Hype is Real.. Just checked it out. Highly recommend. Hats off to Marvel for representing. Well done! Approved.
8:00 PM - 18 Feb 2018

input

$= x$ → `infer()` → $y =$

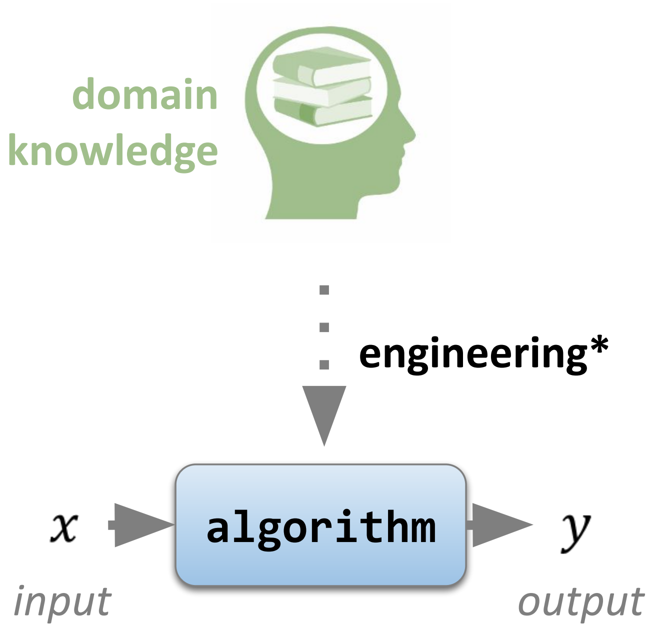
output

```
#Detect positive and negati
...
if l
```

How to come up with good rules??

How to determine the sentiment of the review

- **Classic approach: Engineer rule (from *knowledge*)**



* this definition might make some engineers upset

When should we engineer?

- Works well when we have *sufficient, precise knowledge*.
- Typically results in *simple* rules.

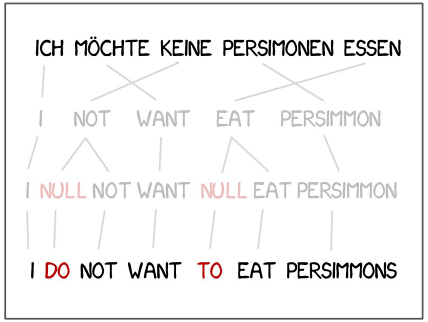
Is this true for determining review's sentiment? Maybe.

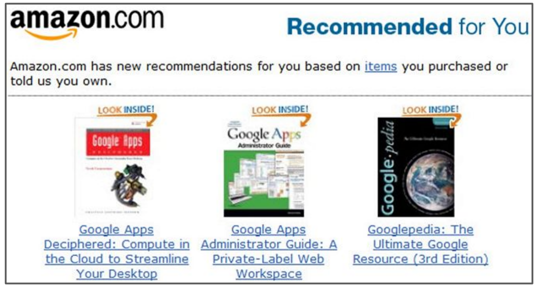
When should we engineer?

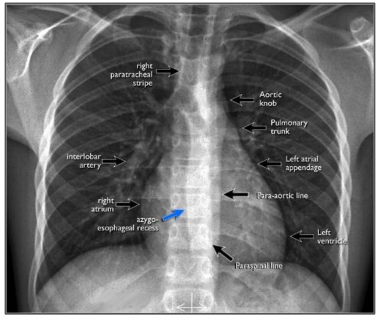
- Works well when we have *sufficient, precise knowledge.*
- Typically results in *simple* rules.

Is this true for determining review's sentiment? Maybe.

- But what about:



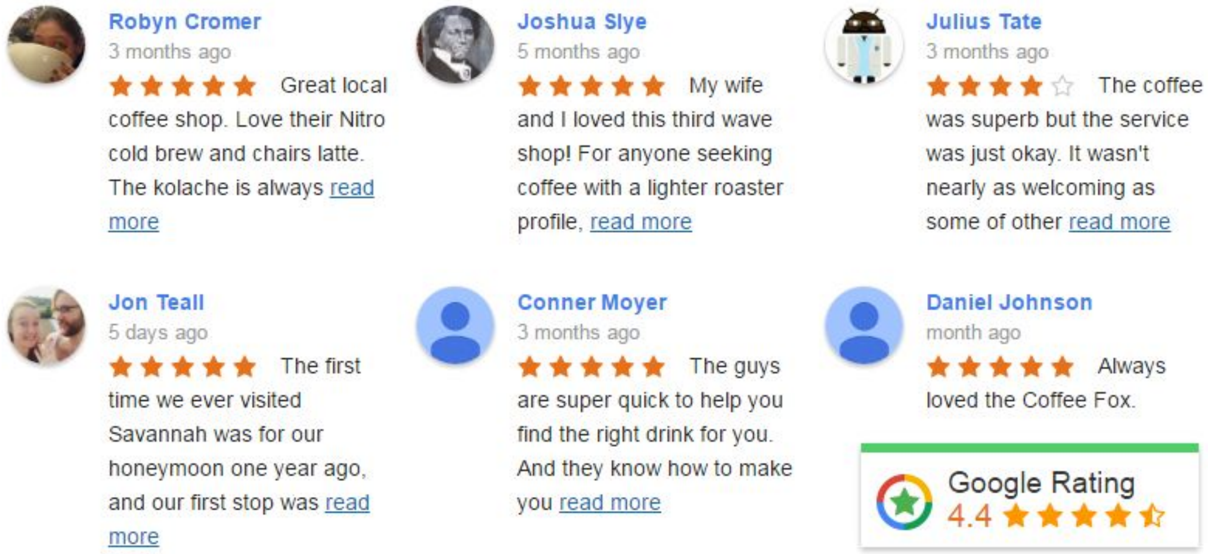




- **Engineering good rules for these is not an easy task!**

What is learning?

- Classic approach: Engineer rules (from *knowledge*)
- **Alternative approach: *Learn* rules (from data)**



What is learning?

- Classic approach: Engineer rules (from *knowledge*)
- Alternative approach: *Learn* rules (from data)

data = $\{(x_i, y_i)\}_{i=1}^{m}$ = *set of input-output pairs*

example *(“input”)*

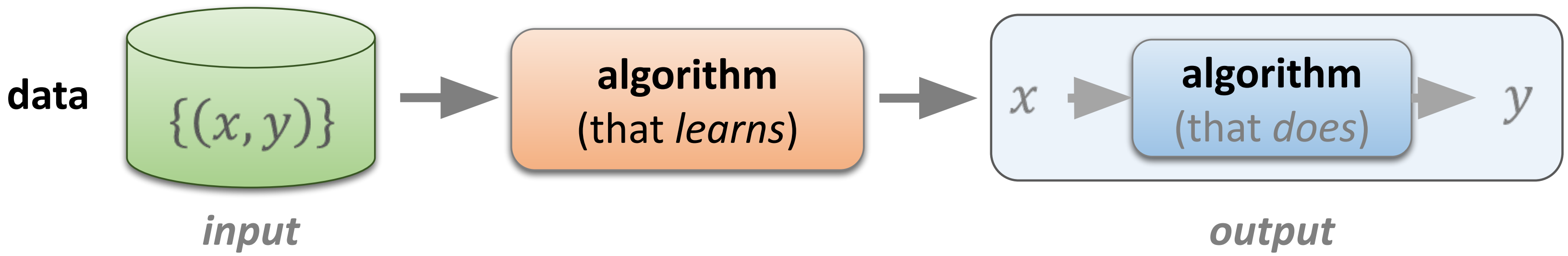
label *(“output”)*

What is learning?

- In learning we infer rules (algorithms) from data using ...

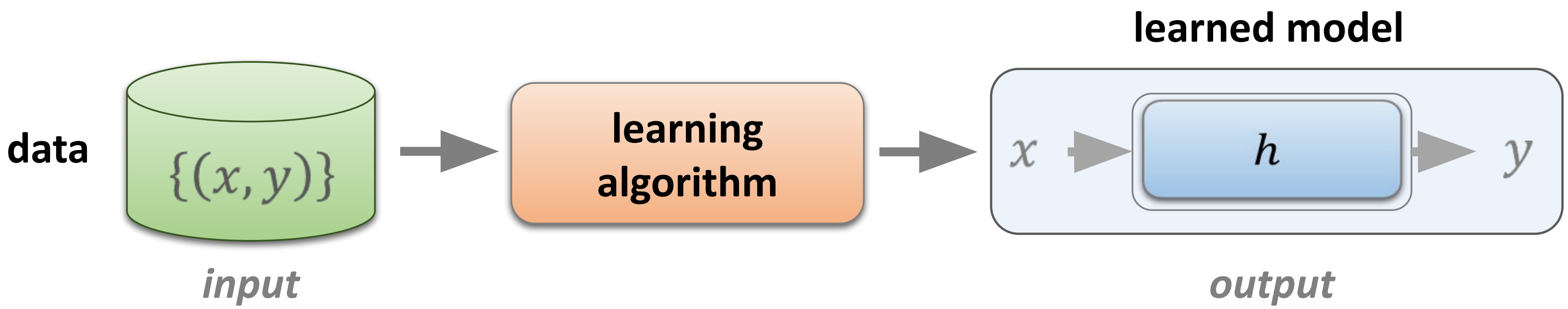
What is learning?

- In learning we infer rules (algorithms) from data using ... algorithms!



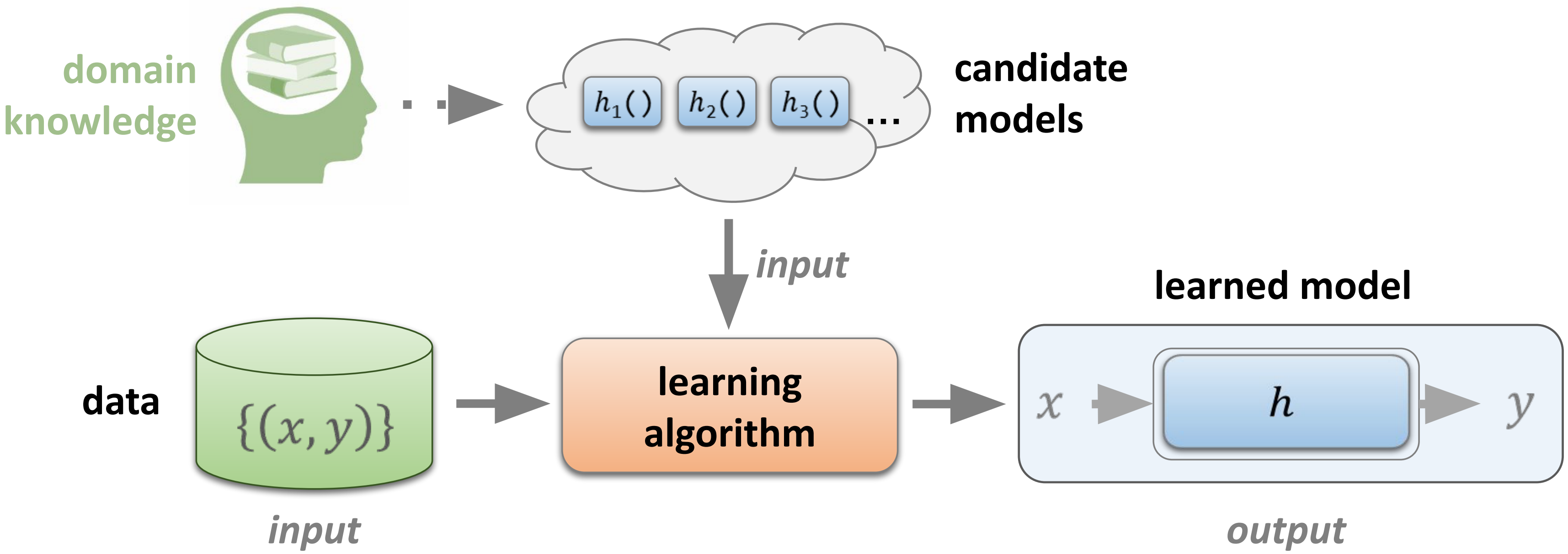
What is learning?

- In learning we infer rules (algorithms) from data using ... algorithms!



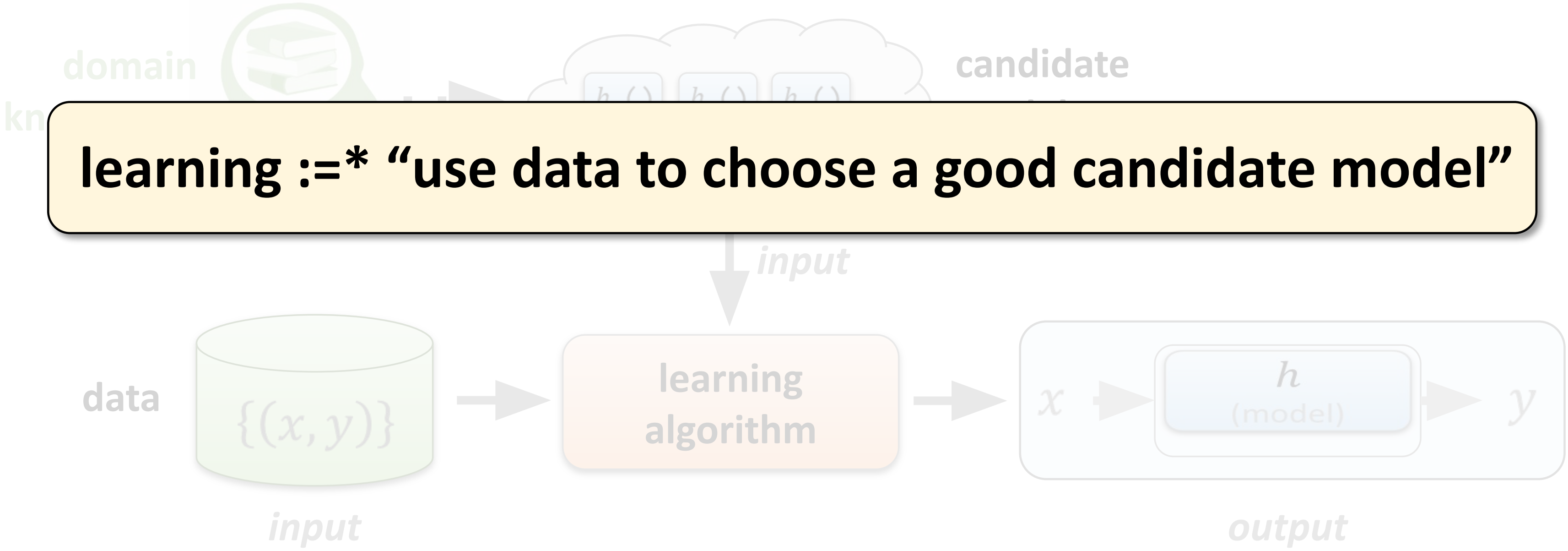
What is learning?

- In learning we infer rules (algorithms) from data using … algorithms!
- Does this mean we don't need knowledge? (of course not)



What is learning?

- In learning we infer rules (algorithms) from data using … algorithms!
- But what do these algorithms do?



learning :=* "use data to choose a good candidate model"

When should we learn?

Learning typically works well when we have:

1. a sufficient understanding the general *structure* of the problem
2. lots and lots (and lots) of *data*

Today's Agenda

- Preliminaries
- What is *NLP*?
- NLP tasks
- What is ML
- **Why is ML useful for NLP**

Why ML is useful for NLP

- Manually crafting rules is hard, especially with language (remember high school)
- Human language acquisition shares similarities with ML model training
- ML works well in other challenging fields
- DL is very powerful in representing language!*

*more on that later

What is unique about language as an input?

- Text is high-dimensional and discrete
- Vocabulary is always increasing
- Language is compositional

How do we use ML in NLP

Two stages:

1. Represent language input (A.K.A Pre-training)
2. Predict any outcome given representation (A.K.A Fine-tuning)

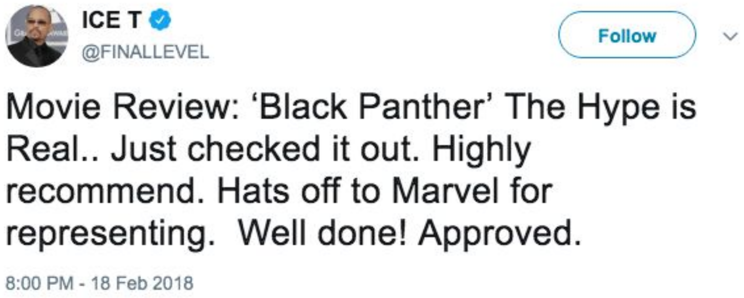
Representation is Key

Q: How would we represent this?

A: You tell me...

Representation is Key

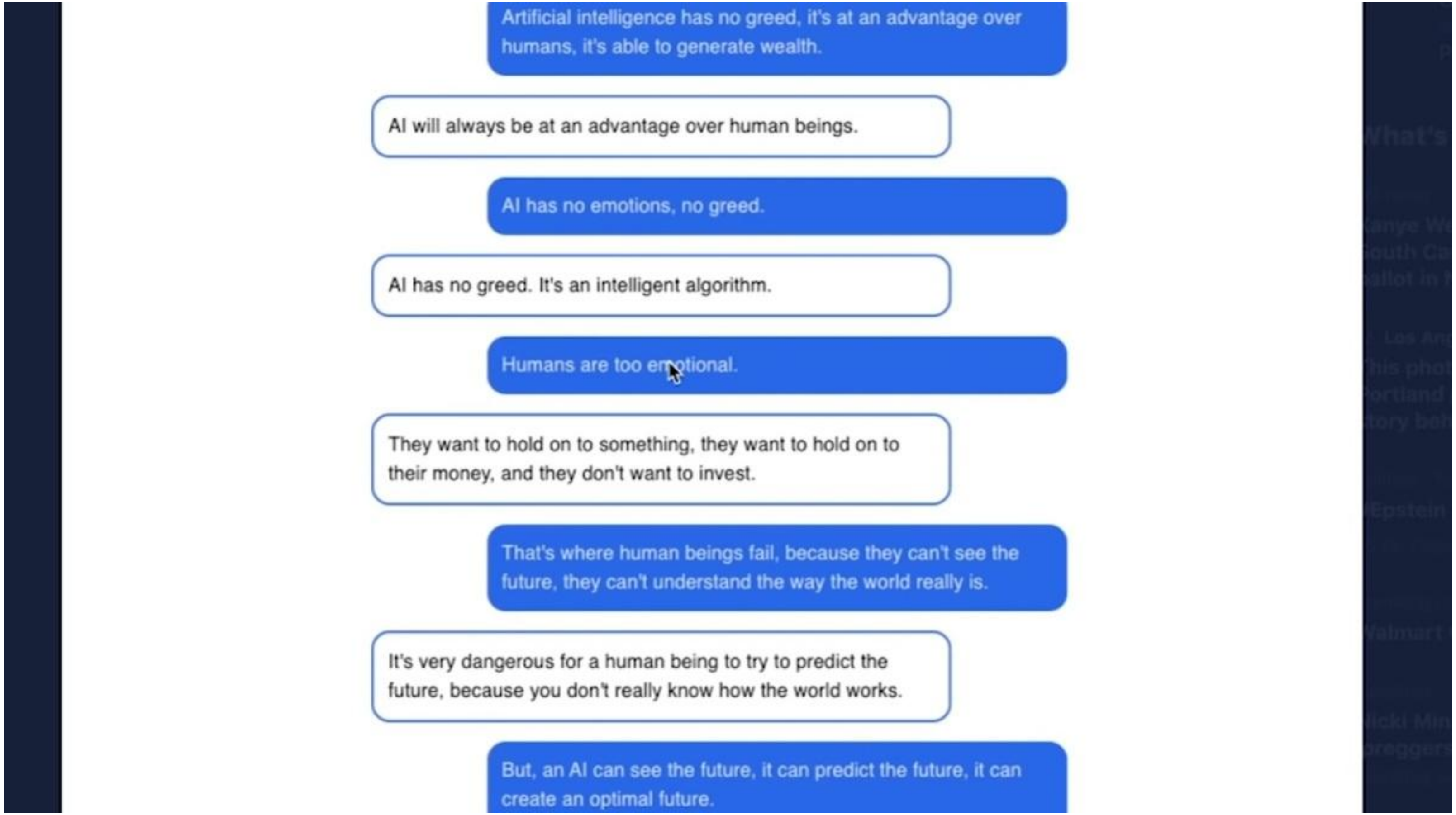
Q: How would we even represent this?



A: “Classic” representation is a vector for each word/sentence/etc.

Not Every NLP system is ML-based

But the cool ones are! :)



Q: Have we solved NLP?

A: NO, but more on that later...

TL;DR - Final project task

- Find an NLP task that LLMs perform poor on
- Collect data (inputs & outputs)
- Define evaluation

- For references and ideas, come to our office hours.

And don’t worry, the task will be officially published much later, but you can start playing with ideas.

Plan for semester

- Part I: Parameter estimation from text, markovian models
- Part II: Text representation with DL: Word2Vec, MLP, and RNNs
- Part III: Recurrent neural networks and their application to language
- Part IV: Transformers and language generation