The Easy, the Hard and the Ugly

- Easy
  - Inference (*fit*)
  - Fine-Tuning (*predict*)
- Hard
  - Pre-training
- Ugly (Responsible AI)
  - Bias
  - Toxicity
  - Misinformation
  - Hallucinations
  - Plagiarism

Easy: Outline

- NLP
  - prompt engineering
  - inference (predict)
  - fine-tuning (fit)
- IR
  - ➢**Quick recap**
- 1 slide on hard problems for researchers/practitioners

Easy - IR

- Inverted index solutions
  - Elastic Search, Solr, Lucene
  - SQL-contains
- Basic multilingual support
- Out-of-the-box ranking
- Simple 10-blue links with any UX framework
- Lots of open-source solutions

Ranking models

- Classical models
  - Boolean
  - Vector space
  - Probabilistic (BM25, LMs, Bayesian networks)
- Web
  - PageRank
  - Hubs & Authorities
- Neural
- Semi-structured text

IR - Relevance evaluation

- Metrics
  - Precision, recall, nDCG, etc.
- Evaluate using gold set
- Crowdsource with a budget
- Simple click instrumentation

Easy: If you are a Researcher/Practioner

- What can you do right now
  - without too much effort
- What other low hanging fruits are out there?
  - Research opportunities
    - Pros/Cons
    - Pointers to related work

Easy: Outline

- NLP
  - ➢**prompt engineering**
  - inference (predict)
  - fine-tuning (fit)
- IR
  - ✓Quick recap
- 1 slide on hard problems for researchers/practitioners

Prompt Engineering
(and disintermediating web-search)

- Super-Popular (100+ million users)
  - Most successful (rapid) adoption of any web app ever
- Super-Easy
  - Easier than Fine-Tuning (and Inference)
- Use Cases
  - “Helping” with homework:
    - Cheating (?)
  - Documentation:
    - Alternative to stack overflow

“Helping” with homework: Cheating (?)

- ***Collaborate*** with students on essays
  - *You have no idea how much we’re using ChatGPT*
  - Cheating?
- ChatGPT is better for some tasks
  - Good: thesis statements, outlines
  - Bad: capture student’s voice
  - Worse: quotes
- Learning opportunity:
  - How to decompose writing to subtasks
  - Collaboration is great,
    - but student is responsible for end-product

Disintermediating Google

- Google → Stackoverflow → Instant Answers → ChatBot

stackoverflow About Products For Teams [faiss]

Home

PUBLIC

Questions

Tags

Users

Companies

COLLECTIVES

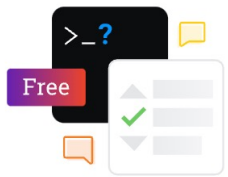
Explore Collectives

LABS

Discussions

TEAMS

Stack Overflow for Teams – Start collaborating and sharing organizational knowledge.



Questions tagged [faiss]

Ask Question

The faiss tag has no usage guidance, but it has a tag wiki.

Learn more... Top users Synonyms

88 questions

Newest | Active | Bountied | Unanswered | More | Filter

8 votes
✓ 3 answers
29k views

faiss ERROR: Could not find a version that satisfies the requirement faiss (from versions: none)

When Running Installation: pip install faiss I am getting this error: ERROR: Could not find a version that satisfies the requirement faiss (from versions: none) ERROR: No matching distribution ...

python python-3.x pip anaconda faiss

predactor **782** asked Nov 20, 2019 at 14:51

7 votes
2 answers
4k views

Installing Faiss on Lambda Stack

I want to install Faiss-GPU on Lambda Stack by conda install -c pytorch faiss-gpu but there is no conda installed. I tried the solution mentioned here: Installing faiss on Google Colaboratory with ...

python faiss

MAMProgr **410** asked Nov 3, 2019 at 11:10

6 votes
✓ 4 answers
10k views

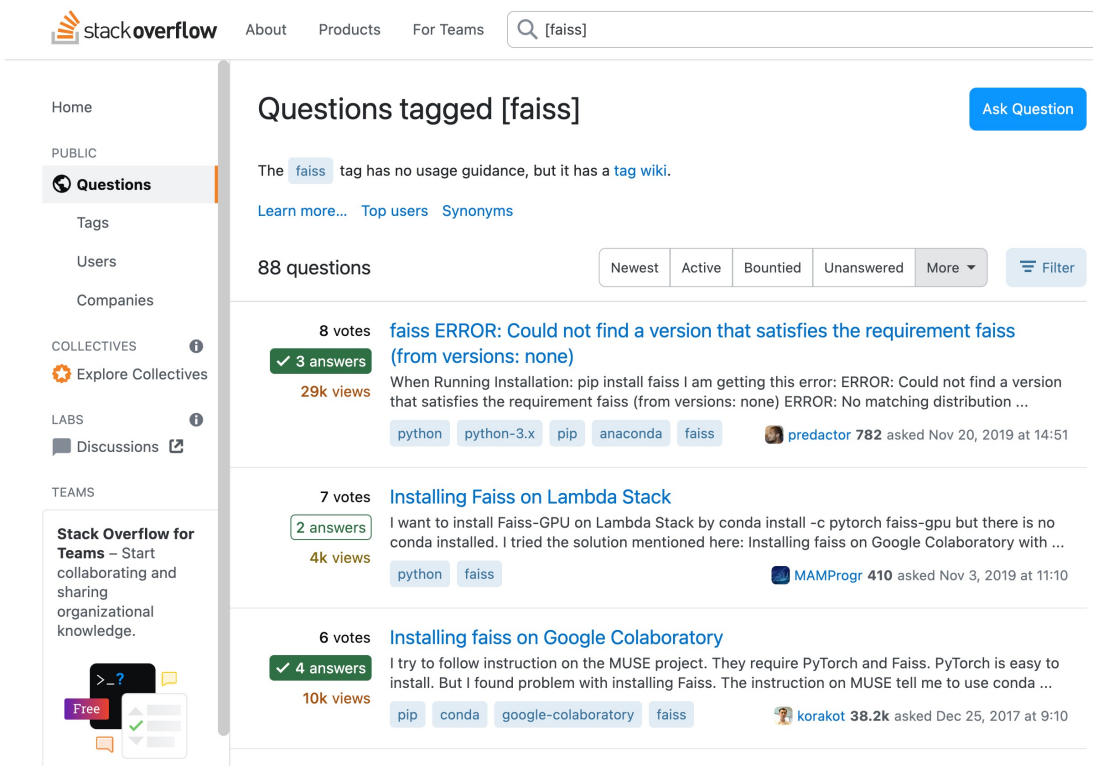
Installing faiss on Google Colaboratory

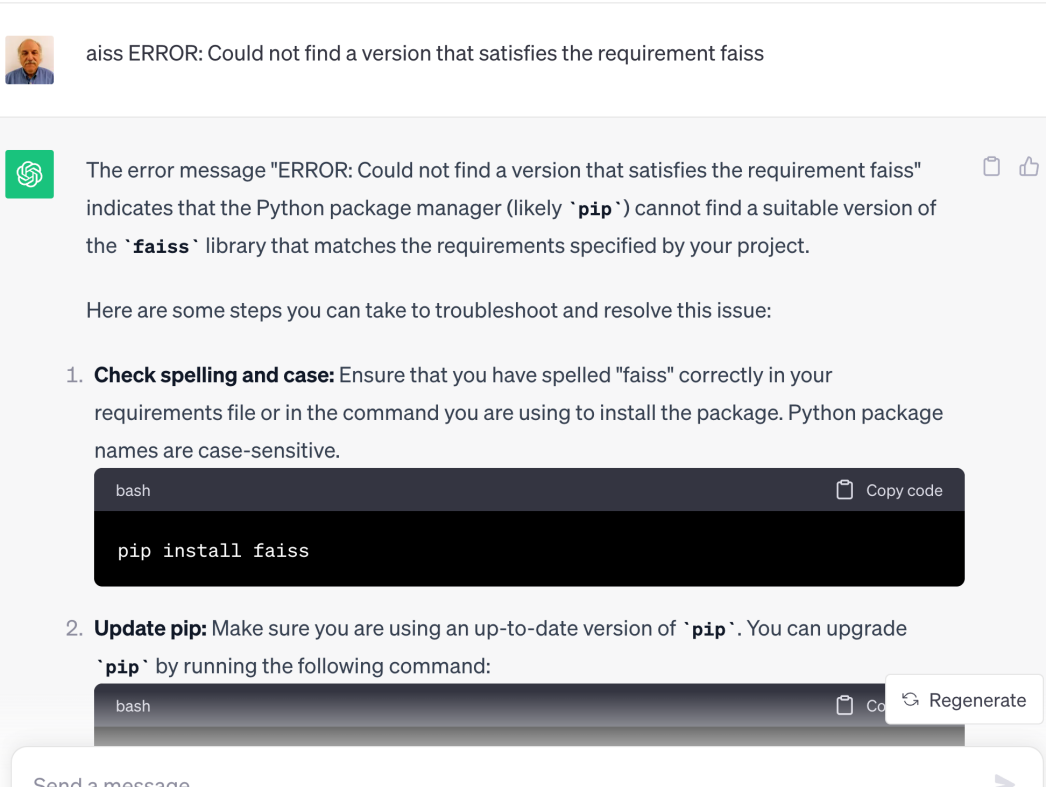
I try to follow instruction on the MUSE project. They require PyTorch and Faiss. PyTorch is easy to install. But I found problem with installing Faiss. The instruction on MUSE tell me to use conda ...

pip conda google-colaboratory faiss

korakot **38.2k** asked Dec 25, 2017 at 9:10

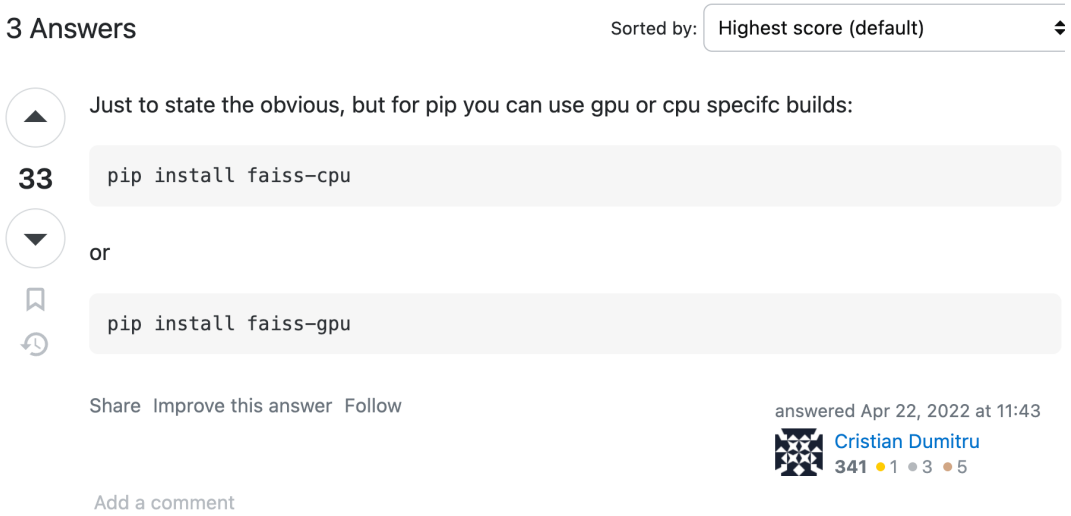
Replacing Stackoverflow with ChatGPT





https://github.com/kwchurch/CIKM_2023_tutorial

I may be old-fashioned, but I still use Stackoverflow…



- Advantages of Stackoverflow
  - Behavioral signals: logs, votes
  - Wisdom of the crowd
  - Feedback to developers
    - Bug Reports
      - with stats (prioritization)
      - and workarounds (with votes)
- Web search
  - Solitaire
  - Multi-player game (auction)
- If we lose stackoverflow
  - Developers will suffer
- Where does ChatGPT get its content?
  - Stackoverflow?

Example: Recommendations and Compare/Contrast

Recommendation Use Cases

- For readers:
  - What should I read?
- For authors:
  - What should I cite?
- For conference organizers:
  - Who should review what?
- Finding experts:
  - Who knows? (Streeter & Lochbaum, 1998)

Semantic Scholar API

- https://api.semanticscholar.org/recommendations/v1/papers/forpaper/21321bad706a9f9dbb502588b0bb393cf15fa052?from=all-cs&fields=title,externalIds,citationCount
  - Semantic Scholar: A collection of 200M papers
  - query: 21321bad706a9f9dbb502588b0bb393cf15fa052
    - A paper id on Semantic Scholar
  - 200M papers → Specter embeddings
    - BERT-like vectors with 768 hidden dimensions
    - Mostly encodes titles and abstracts (with some fine-tuning on citation graph)
  - from=all-cs: limits search to computer science
    - about 10% of their collection of 200M papers
    - uses FAISS to find approximate nearest neighbors for query

Need to explain these topics before getting into this example

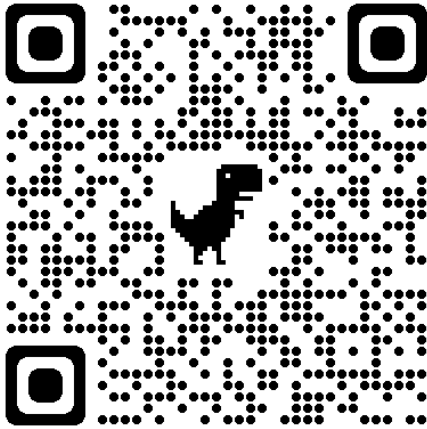
- BERT
- Specter
- Approximate Nearest Neighbors
- FAISS

WSDM-2024 Papers on arXiv

Recommendations Compare and Contrast Page on Semantic Scholar

- Defense Against Model Extraction Attacks on Recommender Systems
- GPT4Table: Can Large Language Models Understand Structured Table Data? A Benchmark and
- Motif-Based Prompt Learning for Universal Cross-Domain Recommendation
- Linear Recurrent Units for Sequential Recommendation
- ProGAP: Progressive Graph Neural Networks with Differential Privacy Guarantees
- Capturing Temporal Node Evolution via Self-supervised Learning: A New Perspective on Dynan
- GAD-NR: Graph Anomaly Detection via Neighborhood Reconstruction
- The Devil is in the Data: Learning Fair Graph Neural Networks via Partial Knowledge Distillatio
- A Multi-Granularity-Aware Aspect Learning Model for Multi-Aspect Dense Retrieval
- Causality Guided Disentanglement for Cross-Platform Hate Speech Detection
- Intent Contrastive Learning with Cross Subsequences for Sequential Recommendation
- LEAD: Liberal Feature-based Distillation for Dense Retrieval
- Pre-trained Recommender Systems: A Causal Debiasing Perspective

Recommendations for a Paper in WSDM-2024



score	citationCount	Paper	Authors	year	More like this	Compare & Contrast
	1	Defense Against Model Extraction Attacks on Recommender Systems	Sixiao Zhang, Hongzhi Yin, ..., Cheng Long	**2023**	Similar to this	Compare & Contrast
40	54	Revisiting Adversarially Learned Injection Attacks Against Recommender Systems	Jiaxi Tang, Hongyi Wen, Ke Wang	2020	Similar to this	Compare & Contrast
39	0	Model Stealing Attack against Recommender System	Zhihao Zhu, Rui Fan, ..., Enhong Chen	2023	Similar to this	Compare & Contrast
38	0	Poisoning Attacks against Recommender Systems: A Survey	Zongwei Wang, Min Gao, ..., Shazia Sadiq	2024	Similar to this	Compare & Contrast
37	9	Gray-Box Shilling Attack: An Adversarial Learning Approach	Zongwei Wang, Min Gao, ..., Jiang Zhong	2022	Similar to this	Compare & Contrast
36	0	Toward Robust Recommendation via Real-time Vicinal Defense	Yichang Xu, Chenwang Wu, Defu Lian	2023	Similar to this	Compare & Contrast
35	6	Debiasing Learning for Membership Inference Attacks Against Recommender Systems	Zihan Wang, Na Huang, ..., Z. Ren	2022	Similar to this	Compare & Contrast

Compare and Contrast

Title: Revisiting Adversarially Learned Injection Attacks Against Recommender Systems

Top pdf 54 citations; 48 references
Authors: Jiaxi Tang, Hongyi Wen, Ke Wang

tl;dr: This paper revisits the adversarially-learned injection attack problem, where the injected fake user 'behaviors' are learned locally by the attackers with their own model – one that is potentially different from the model under attack, but shares similar properties to allow attack transfer.

Abstract: Recommender systems play an important role in modern information and e-commerce applications. While increasing research is dedicated to improving the relevance and diversity of the recommendations, the potential risks of state-of-the-art recommendation models are under-explored, that is, these models could be subject to attacks from malicious third parties, through injecting fake user interactions to achieve their purposes. This paper revisits the adversarially-learned injection attack problem, where the injected fake user 'behaviors' are learned locally by the attackers with their own model – one that is potentially different from the model under attack, but shares similar properties to allow attack transfer. We found that most existing works in literature suffer from two major limitations: (1) they do not solve the optimization problem precisely, making the attack less harmful than it could be, (2) they assume perfect knowledge for the attack, causing the lack of understanding for realistic attack capabilities. We demonstrate that the exact solution for generating fake users as an optimization problem could lead to a much larger impact. Our experiments on a real-world dataset reveal important properties of the attack, including attack transferability and its limitations. These findings can inspire useful defensive methods against this possible existing attack.

Citation:

```
@Article{Tang2020RevisitingAL,
 author = {Jiaxi Tang and Hongyi Wen and Ke Wang},
 booktitle = {ACM Conference on Recommender Systems},
 journal = {Proceedings of the 14th ACM Conference on Recommender Systems},
 title = {Revisiting Adversarially Learned Injection Attacks Against Recommender Systems},
 year = {2020}
}
```

- Input: query + candidate
- Use prompts (chatbots)
  - Summary of candidate
  - Similarities of query & candidate
  - Differences of query & candidate

Output from Chatbot

Similarities

- As an Assistant, I can tell you that both "Revisiting Adversarially Learned Injection Attacks Against Recommender Systems" and "Defense Against Model Extraction Attacks on Recommender Systems" focus on security aspects of recommender systems. The first paper explores injection attacks where the attacker manipulates the recommendation model by injecting malicious data, while the second paper deals with defending against model extraction attacks aimed at extracting sensitive information from the recommender system. Both papers highlight the importance of addressing security vulnerabilities in recommender systems to protect user data and maintain the integrity of the recommendation models.

Differences

- "Revisiting Adversarially Learned Injection Attacks Against Recommender Systems" focuses on exploring adversarially learned injection attacks against recommender systems, where the attacker has knowledge about the dataset used to train the target recommendation model. On the other hand, "Defense Against Model Extraction Attacks on Recommender Systems" focuses on strategies to defend against model extraction attacks, where attackers try to extract the architecture or parameters of a recommender system. The former paper deals with adversarial attacks, while the latter focuses on defense mechanisms against a different type of attack.

Chatbots are Super-Easy
www.vecml.com

```
from vecml import vecml
import numpy as np
import argparse

parser = argparse.ArgumentParser()
parser.add_argument("--chat_name", help="string", required=True)
parser.add_argument("--files", help="comma separated filenames", default=None)
parser.add_argument("--prompt", help="prompt", required=True)
parser.add_argument("--apikey", help="API key from VecML", required=True)
args = parser.parse_args()

if __name__ == '__main__':
    vecml.init(args.apikey,"us-west")

    if not args.files is None:
        vecml.create_chat(args.chat_name, args.files.split(','))
    print(vecml.chat(args.chat_name, args.prompt))
```

Decomposing a big problem into subtasks

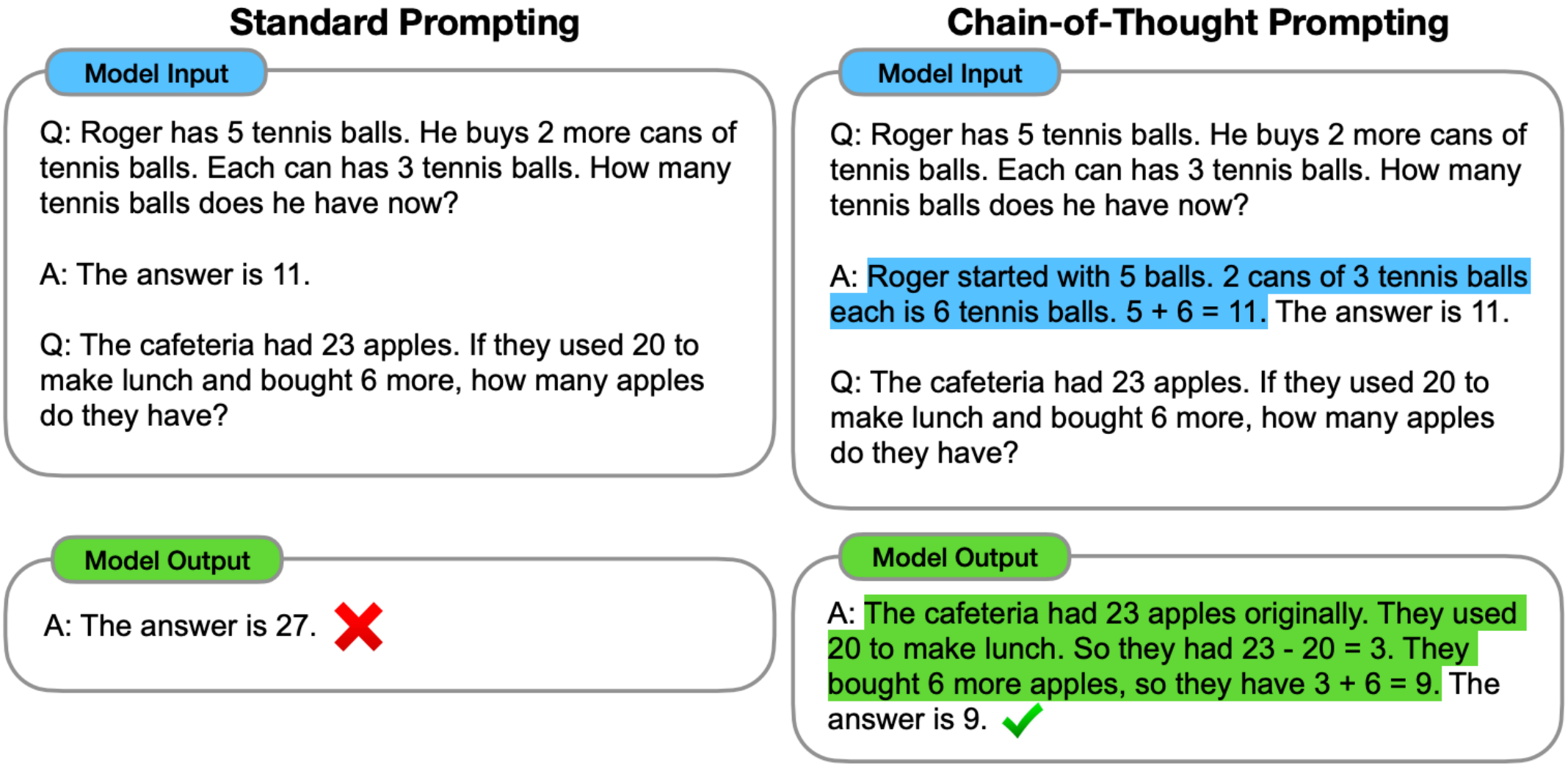


Figure 1: Chain-of-thought prompting enables large language models to tackle complex arithmetic, commonsense, and symbolic reasoning tasks. Chain-of-thought reasoning processes are highlighted.

Limitations

Decomposing Problems

- Chain-of-Thought Prompting
  - works because...
  - ChatGPT needs help decomposing problems into subtasks

Hypothetical

- Suppose ChatGPT
  - can add two small numbers
- But for large numbers,
  - it makes up answers
- After each release,
  - “small” gets “bigger”
- (We will return to this later)

Resources

飞桨 PaddleHub

DataSets (Benchmarks)

- Examples
  - GLUE (NYU)
  - SQuAD (Stanford)
- Features
  - Splits:
    - Test, Validation (Dev), Train
  - Metrics
  - Leaderboard

Hubs

- Examples
  - HuggingFace https://huggingface.co/
  - PaddleHub (Baidu, China)
- Features
  - Datasets (50k)
  - Models (300k)
  - Apps (100k)
  - Community Engagement
    - (Social Media)

Growing Quickly
(≈ 10x/year)

Standard 3-Step Recipe

Step	**Description**	**Time**	**Hardware**
1			
2			
3	Inference (predict)	Seconds/Minutes	0+ GPUs

Standard 3-Step Recipe

Most users should *not* do this themselves

Step	**Description**	**Time**	**Hardware**
1	Pre-Training	Days/Weeks	Large GPU cluster
2			
3	Inference (predict)	Seconds/Minutes	0+ GPUs

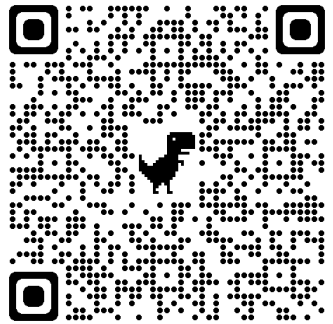
Standard 3-Step Recipe

Step	**Description**	**Time**	**Hardware**
1	Pre-Training	Days/Weeks	Large GPU cluster
2	Fine-Tuning (fit)	Hours/Days	1+ GPUs
3	Inference (predict)	Seconds/Minutes	0+ GPUs

Not Hard

Inference (predict)

Easy: Inference (Colab) HuggingFace Pipelines



- See here for a list of currently supported tasks
  - machine learning:
    - classification, regression,
    - token classification,
    - classify spans (Q&A), fill mask
  - speech:
    - speech-to-text (automatic speech recognition (ASR)),
    - text-to-speech (speech synthesis)
    - audio classification
- vision:
  - image classification, video classification, image segmentation, image to text, visual question answering
- natural language:
  - text classification
  - question answering
  - fill mask (Cloze task)
  - machine translation (MT)
  - feature extraction

“Easy Inference”

Replication

- Many students should be able to
  - Download models from hubs
  - And use them as intended
  - To do many useful things

Plenty of Room for Improvement

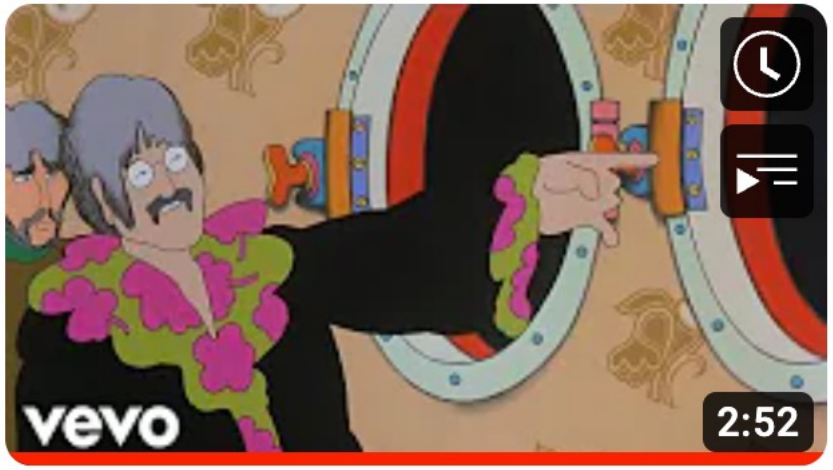
- Especially on first impression,
  - It is impressive how much
  - can be accomplished
    - with “easy inference”
- But there is always
  - more work to do
- Current solutions are
  - unlikely to stand up to test of time

Test of time: Inference is cool (for now)

https://youtu.be/rLl9XBg7wSs

https://github.com/kwchurch/CIKM_2023_tutorial

Test of time: Inference is cool (for now)
Will you still love me when I'm Sixty Four?



The Beatles - When I'm Sixty Four (Official Video)

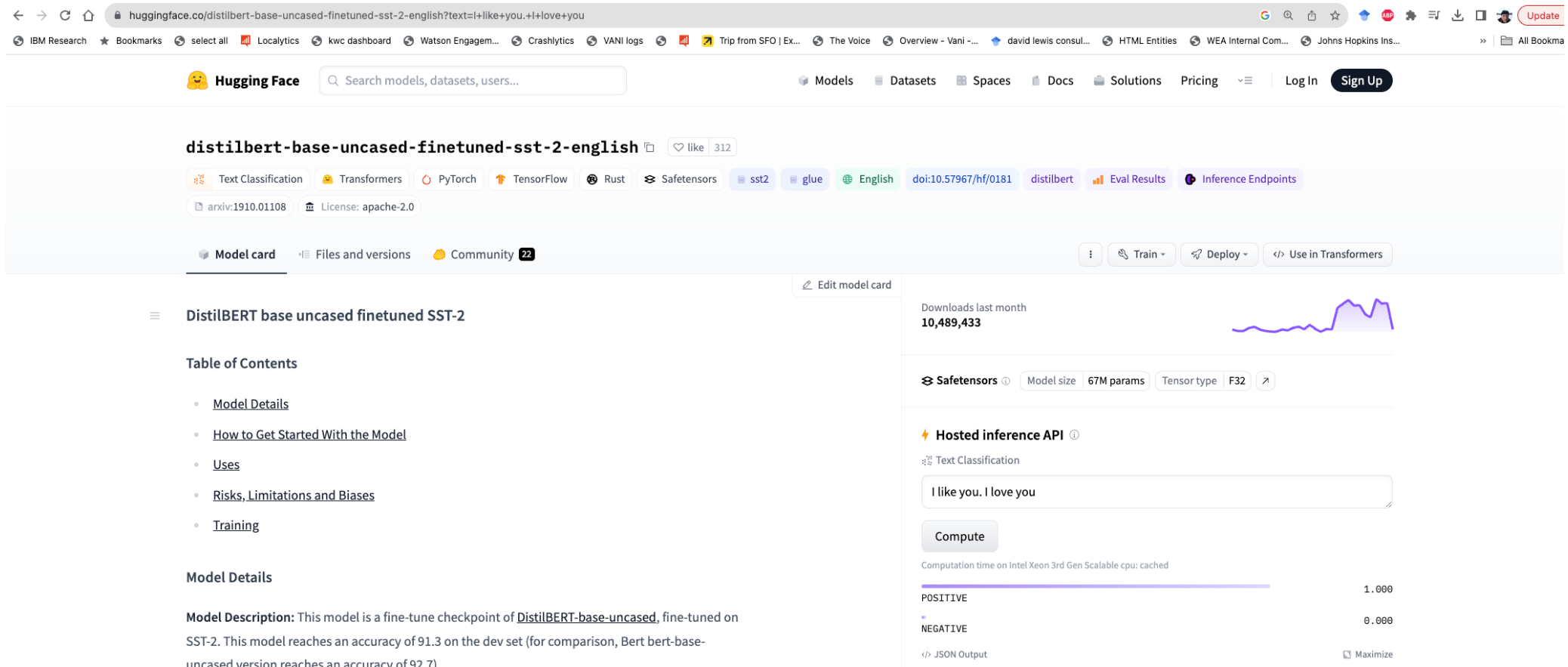
TheFreddyShow • 585K views

"When I'm Sixty-Four" is a song by the English rock band The Beatles, written by Paul McCartney (credited to Lennon–McCartney) and...

https://www.youtube.com/watch?v=wUDRIC5RSX4

```
[ ]  pipe = pipeline("text-classification")
     pipe(["I love you", "I hate you"])
```

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english and revision
Using a pipeline without specifying a model name and revision in production is not recommended.
[{'label': 'POSITIVE', 'score': 0.9998656511306763},
 {'label': 'NEGATIVE', 'score': 0.9991129040718079}]
```



Input: *I love you*
Output: Depends on Model (among other things)

- Task: classify
- Models:
  - Sentiment
    - Expected output label: positive
  - Fake News:
    - Expected output label: not fake (real)
  - Spam/Ham:
    - Expected output label: not spam (ham)

```
for model in …
 do
   echo I love you |
   gft_predict \
      --task classify \
      --model $model
 done
```

Scores could be more confident

Many of these models produce reasonable results

I love you is **positive**

Predicted Label	Score	Model	Labels for Model
positive	0.512	SetFit/deberta-v3-large__sst2__train-16-7	negative, positive
POSITIVE	0.871	ayameRushia/roberta-base-indonesian-sentiment-analysis-smsa	POSITIVE, NEUTRAL, NEGATIVE
positive	0.807	SetFit/distilbert-base-uncased__sst2__train-32-2	negative, positive
positive	0.999	AdapterHub/bert-base-uncased-pf-sst2	negative, positive
positive	0.917	SetFit/deberta-v3-large__sst2__train-32-1	negative, positive
positive	0.999	moshew/tiny-bert-aug-sst2-distilled	negative, positive
positive	0.651	rohansingh/autonlp-Fake-news-detection-system-29906863	negative, positive
5 stars	0.872	tomato/sentiment_analysis	1 star, 2 stars, 3 stars, 4 stars, 5 stars
5 stars	0.424	cmarkea/distilcamembert-base-sentiment	1 star, 2 stars, 3 stars, 4 stars, 5 stars
5 stars	0.872	nlptown/bert-base-multilingual-uncased-sentiment	1 star, 2 stars, 3 stars, 4 stars, 5 stars

But..

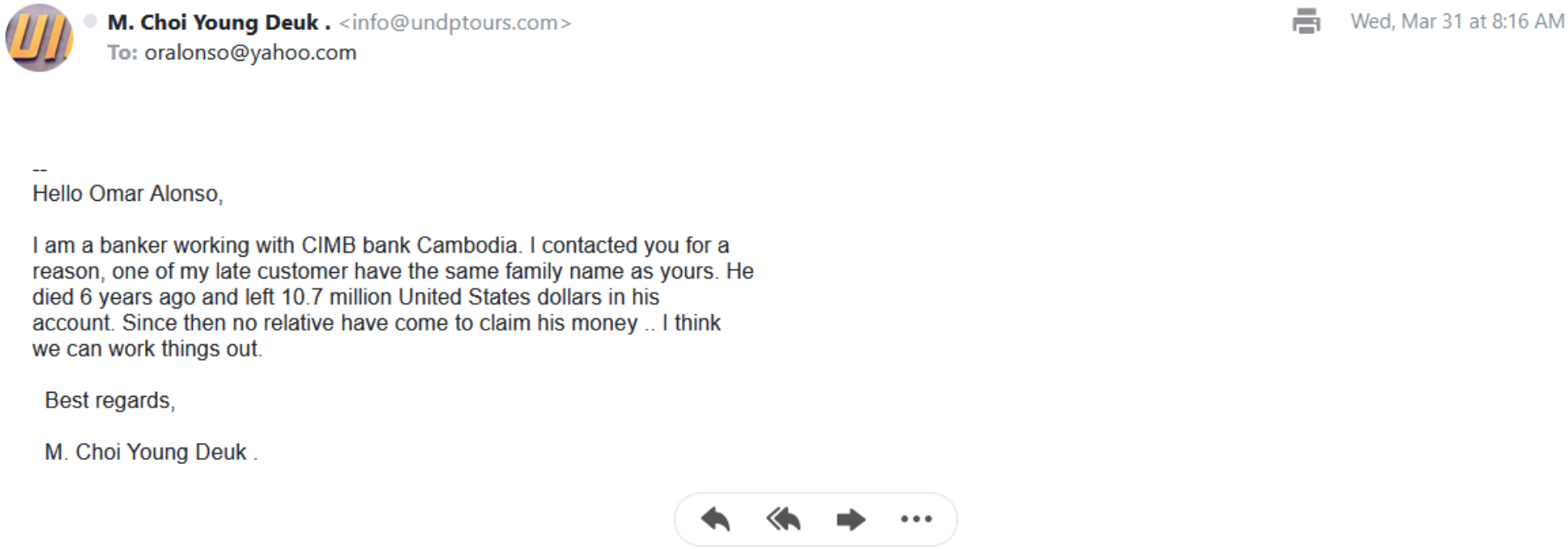
I love you is **fake news**

Predicted Label	Score	Model	Labels for Model
Fake	0.998	yaoyinnan/bert-base-chinese-covid19	Neutral, Fake, Real
Fake	0.986	yaoyinnan/roberta-fakeddit	Fake, Real
fake	0.958	Qiaozhen/fake-news-detector	real, fake
FAKE	0.959	Narrativaai/fake-news-detection-spanish	REAL, FAKE

I love you is both **spam** and **ham**

Predicted Label	Score	Model	Labels for Model
spam	0.826	SetFit/distilbert-base-uncased__enron_spam__all-train	ham, spam
not spam	1.000	sureshs/distilbert-large-sms-spam	not spam, spam

What is a label?



M. Choi Young Deuk . <info@undptours.com>
To: oralonso@yahoo.com

Wed, Mar 31 at 8:16 AM

--
Hello Omar Alonso,

I am a banker working with CIMB bank Cambodia. I contacted you for a reason, one of my late customer have the same family name as yours. He died 6 years ago and left 10.7 million United States dollars in his account. Since then no relative have come to claim his money .. I think we can work things out.

Best regards,

M. Choi Young Deuk .

Spam email?

Label: yes, no

Ugly: Challenges for Classification

- Benchmarks are small, clean, unrealistic…
- Arms race, Adversarial attacks (GANs),
- Labeling is not as easy as it might seem
  - Depends on context
  - Consider: adult content
    - Parents posted innocent videos of their kids on YouTube
    - But these videos went viral among not-so-innocent audience
- Incentives:
  - If toxicity is profitable,
  - then use classifiers in reverse direction
  - to maximize toxicity (and profits)

Another Example of Inference: Machine Translation (MT)

```
from transformers import pipeline
pipe = pipeline("translation", model="Helsinki-NLP/opus-mt-fr-en")
pipe("Je t'aime.")
```

Back Translation

```
[ ]  def backtranslate(s0, models):
       pipes = [pipeline('translation', model=m) for m in models]
       translations = [s0]
       for p in pipes:
         ss = translations[-1]
         tt = p(ss)
         translations.append(tt[0]['translation_text'])
       return translations
```

```
backtranslate('No smoking', ["Helsinki-NLP/opus-mt-en-de", "Helsinki-NLP/opus-mt-de-fr", "Helsinki-NLP/opus-mt-fr-en",])
```

```
['No smoking',
 'Rauchen verboten',
 'Il est interdit de fumer',
 'Smoking is prohibited']
```

Back Translation of Synonyms

```
for s in syns:
  print(backtranslate(s, ["Helsinki-NLP/opus-mt-en-zh", "Helsinki-NLP/opus-mt-zh-en"]))
```

```
['rant and raving', '暴暴和暴暴,', 'Violence and violence, violence, violence, violence, violence,
['regard and heed', '重视和关注', 'Attention and attention']
['monster and abnormity', '薄命的、薄命的、薄命的、', 'the left hand, the left hand, the left hand,
['relinquish and forsake', '放弃和放弃', 'Waiver and abandonment']
['ebb and recede', 'ebb和reede( 缩缩)', 'ebb and reede (shrunk)']
['austere and stern', '坚硬和坚硬', 'Hard and hard.']
```

Fill Mask (Cloze Task)

3s

```
pipe = pipeline("fill-mask", model='distilroberta-base')
pipe("She likes <mask>, but he likes that.")
```

```
Some weights of the model checkpoint at distilroberta-base
- This IS expected if you are initializing RobertaForMaske
- This IS NOT expected if you are initializing RobertaForM
[{'score': 0.024591688066720963,
  'token': 123,
  'token_str': ' him',
  'sequence': 'She likes him, but he likes that.'},
 {'score': 0.022750644013285637,
  'token': 9366,
  'token_str': ' pizza',
  'sequence': 'She likes pizza, but he likes that.'},
 {'score': 0.022747714072465897,
  'token': 10017,
  'token_str': ' cats',
  'sequence': 'She likes cats, but he likes that.'},
 {'score': 0.012954547069966793,
  'token': 69,
  'token_str': ' her',
  'sequence': 'She likes her, but he likes that.'},
 {'score': 0.011491155251860619,
  'token': 162,
  'token_str': ' me',
  'sequence': 'She likes me, but he likes that.'}]
```

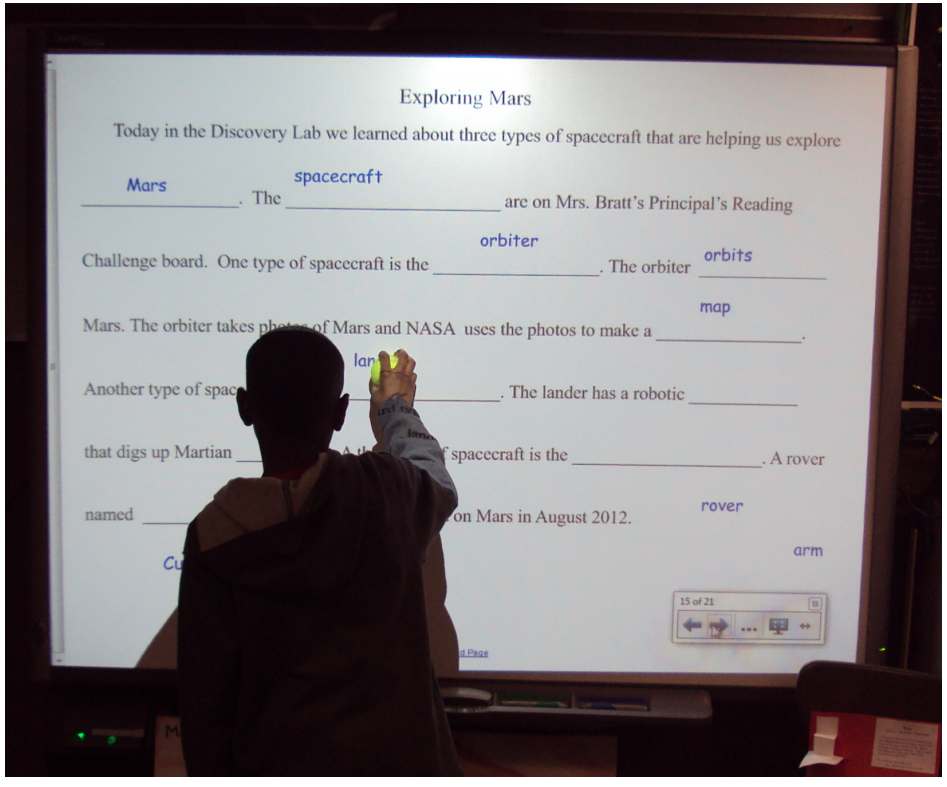
More Inference Tasks: Classification, Fill Mask, …

`echo … | gft_predict -task fill-mask`

BERT: Fill-Mask (Delvin et al, 2019)

Input	Choice 1		Choice 2	
I <mask> you	miss	30%	love	17%
I <mask> him	love	14%	miss	13%
I <mask> her	love	16%	miss	12%
I love <mask>	him	2%	cats	2%
I <mask> New York	love	33%	miss	17%
I love the <mask> guy	pizza	2%	funny	2%
I hate the <mask> guy	fucking	3%	pizza	2%

Cloze Task (1953)



By Laurie Sullivan - DSC01784, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=62727253

Lots of History:

Fill-mask, Cloze Task, Entropy

Prediction and Entropy of Printed English

By C. E. SHANNON

(*Manuscript Received Sept. 15, 1950*)

A new method of estimating the entropy and redundancy of a language is described. This method exploits the knowledge of the language statistics possessed by those who speak the language, and depends on experimental results in prediction of the next letter when the preceding text is known. Results of experiments in prediction are given, and some properties of an ideal predictor are developed.

first line is the original text and the numbers in the second line indicate the guess at which the correct letter was obtained.

(1) THERE IS NO REVERSE ON A MOTORCYCLE A
(2) 1 1 1 5 1 1 2 1 1 2 1 1 1 5 1 1 7 1 1 1 2 1 3 2 1 2 2 7 1 1 1 1 4 1 1 1 1 1 3 1
(1) FRIEND OF MINE FOUND THIS OUT
(2) 8 6 1 3 1 1 1 1 1 1 1 1 1 1 1 6 2 1 1 1 1 1 1 2 1 1 1 1 1 1
(1) RATHER DRAMATICALLY THE OTHER DAY
(2) 4 1 1 1 1 1 1 1 1 5 1 1 1 1 1 1 1 1 1 1 1 1 6 1 1 1 1 1 1 1 1 1 1 1 1 (9)

Out of 102 symbols the subject guessed right on the first guess 79 times, on the second guess 8 times, on the third guess 3 times, the fourth and fifth guesses 2 each and only eight times required more than five guesses. Results of this order are typical of prediction by a good subject with ordinary literary English. Newspaper writing, scientific work and poetry generally lead to somewhat poorer scores.

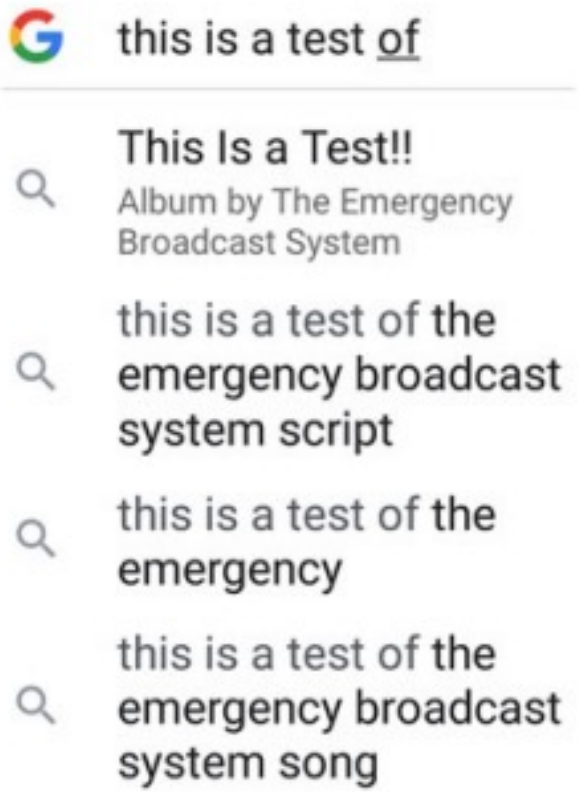
Guess the missing word in a cliché
(from “Emerging trends: Deep nets for poets”)

Fill Mask

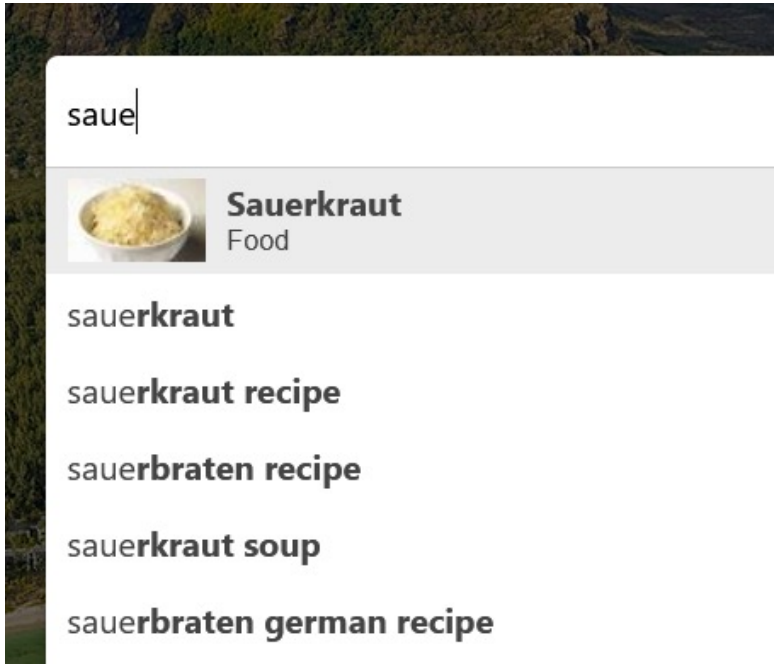
Table 3. Unmasking: replace each input word with [MASK] and predict fillers. Red is added to highlight differences between the top prediction and the original input

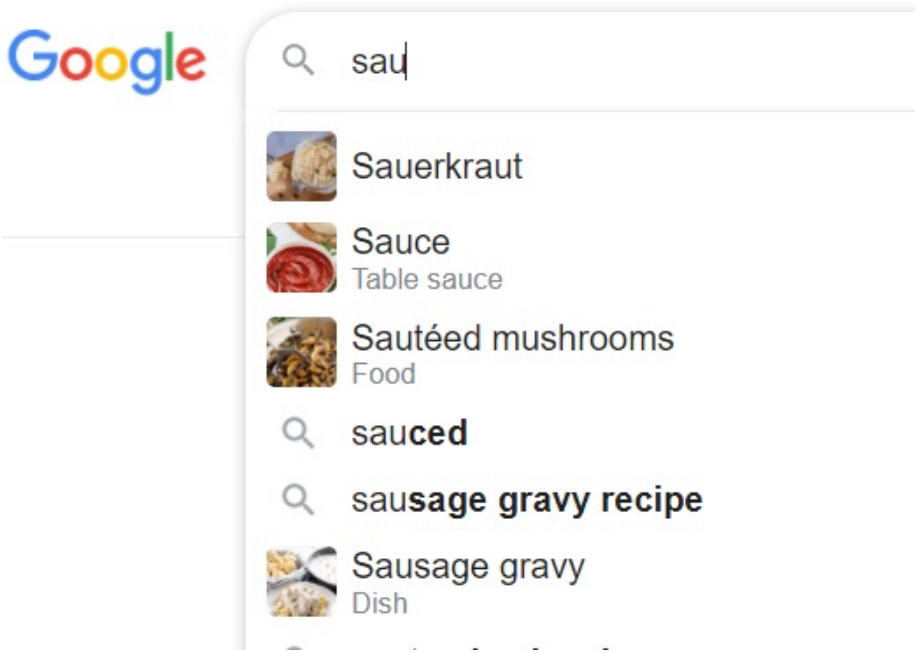
Word	Rank 0	Rank 1	Rank 2
This	this:0.595	it:0.375	there:0.004
is	was:0.628	is:0.334	included:0.007
a	a:0.935	another:0.029	the:0.023
test	part:0.253	subset:0.125	variation:0.082
of	of:0.886	for:0.070	on:0.017
the	the:0.883	an:0.088	our:0.005
emergency	ratio:0.257	television:0.048	satellite:0.031
broadcast	braking:0.620	response:0.070	management:0.024
system	system:0.244	technique:0.077	capability:0.059
.	.:0.969	;:0.029	!:0.001

Autocomplete in Search



Autocomplete ≠ Fill-Mask

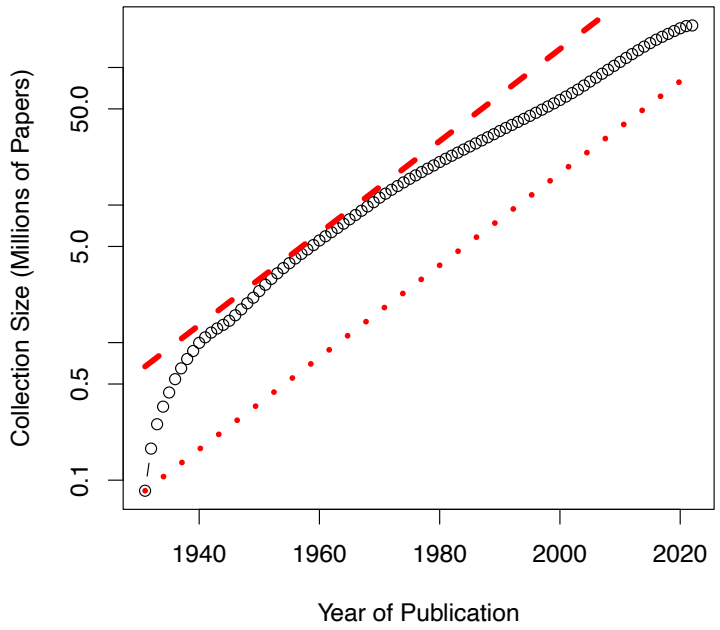




Massive Growth → Mistaken Impression that everything is new (and there is no history)

Scientific literature doubles every 9 years (90% written since I started PhD)

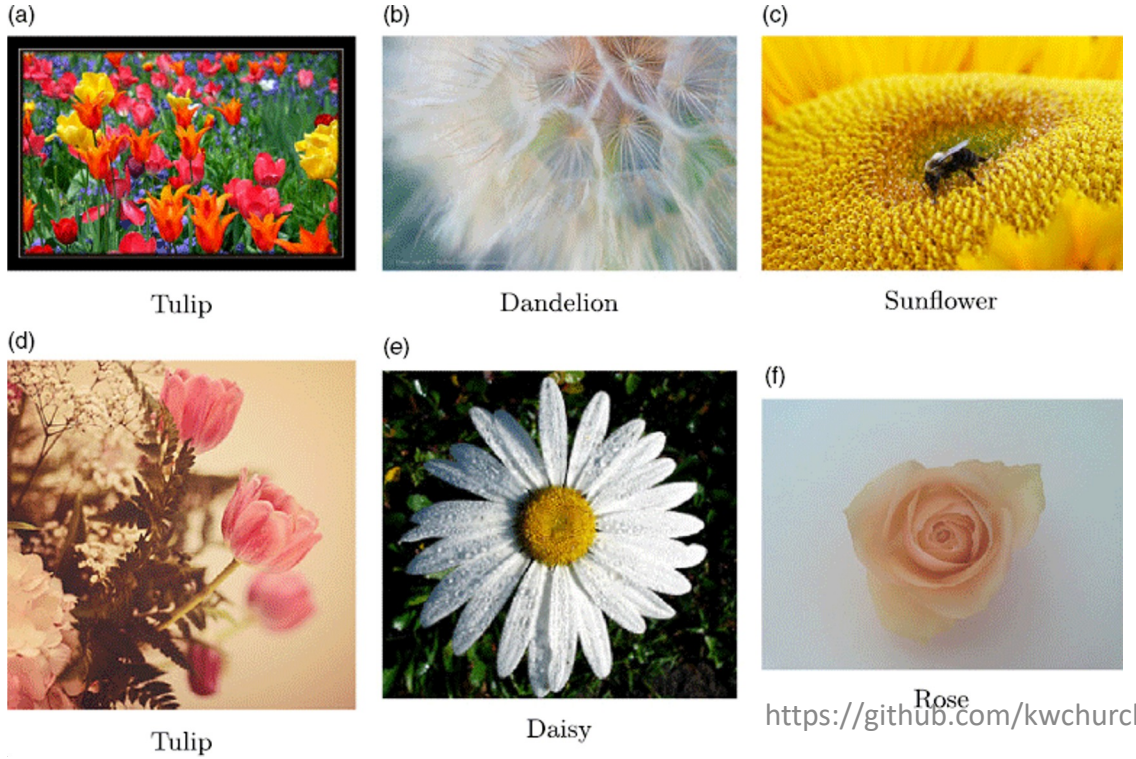
- What's new
  - The world is taking notice (in AI)
  - Fluency is much improved
- What's not new
  - Chatbots (and much of the tech)
  - SOTA-Chasing
  - ➢**New/better *shiny* objects**
    - (with same old quality?)
    - Does SOTA-Chasing → Progress?
  - Trustworthiness is still open



Fine-Tuning (fit) in GFT

$$f_{pre} + data \rightarrow f_{post}$$

- Flowers
  - f_{pre}: Resnet
  - $data$: flowers




```
gft_fit --eqn 'classify: label ~ text' \
    --model H:bert-base-cased \
    --data H:emotion \
    --output_dir $outdir
```

f_{pre}: Pre-trained Model

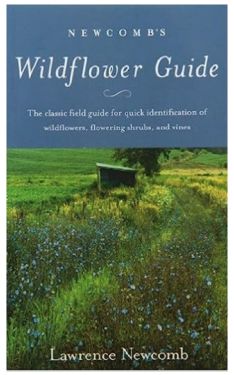
f_{post}: Post-trained Model

```
7 g = glm(dist ~ poly(speed, 2), data=cars)
```

Fine-Tuning (fit) in R

https://github.com/kwchurch/CIKM_2023_tutorial

Guide Book App



Example of Fine-Tuning (aka, *fit*)

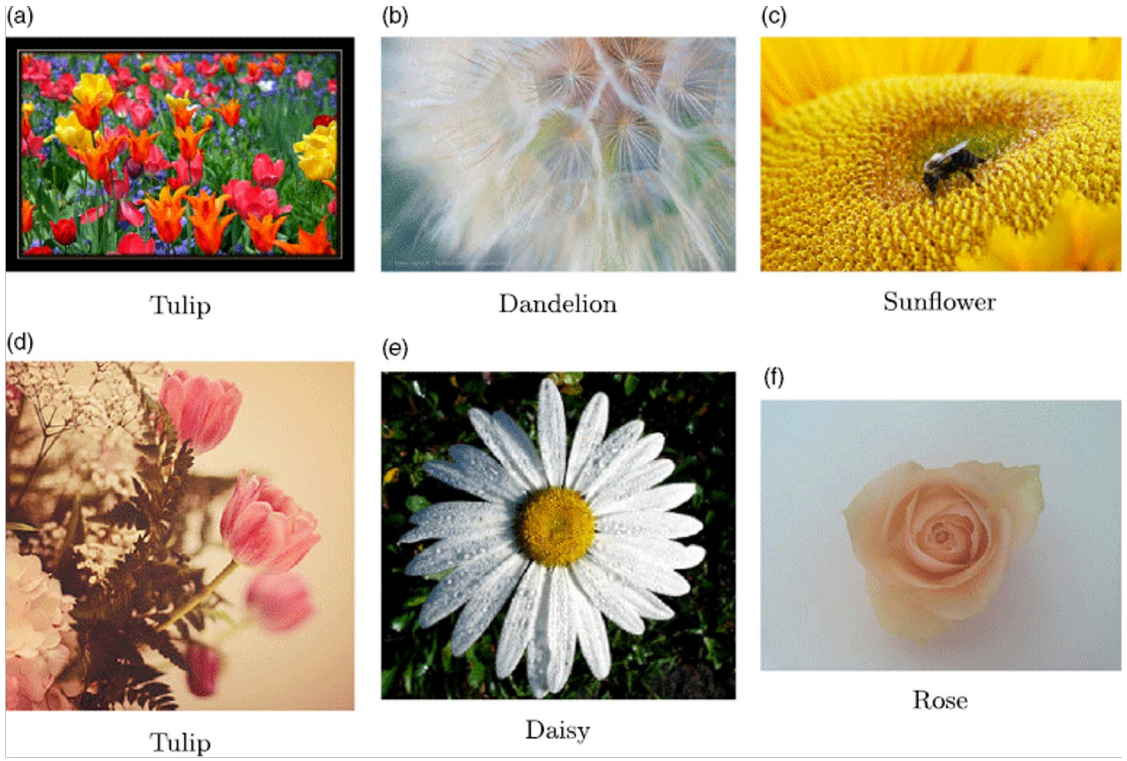
fit: $f_{pre} + data \rightarrow f_{post}$

f_{pre}: Resnet

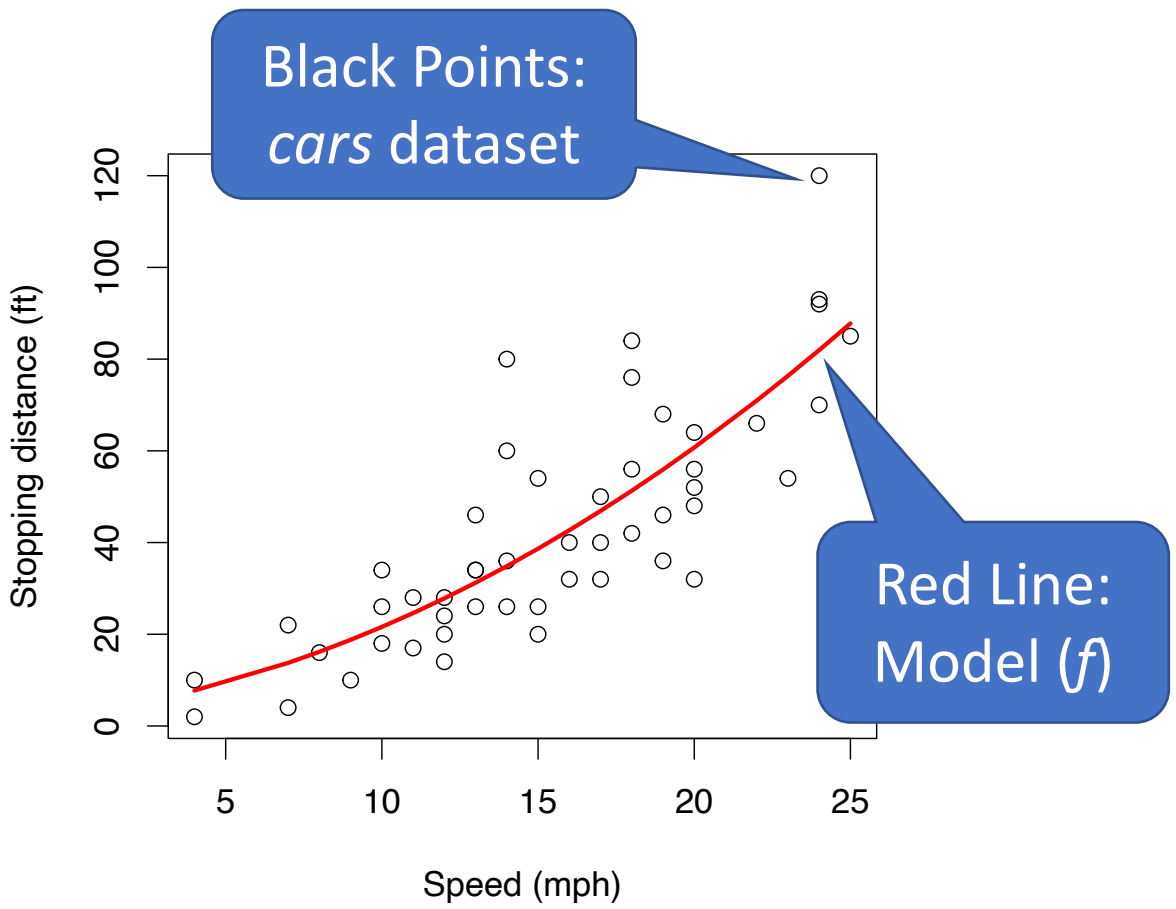
- Maps images (jpg files) → classes (strings)
- Trained on ImageNet
  - input (x): 14M images (of many things)
  - output (y): 1000 classes (strings)

- data : flowers
  - input (x): 2195 pictures of flowers
  - outputs (y): 5 classes of flowers
    - *Tulip, Dandelion, Sunflower, Daisy, Rose*

- f_{post}:
  - Maps images (jpg files) → flowers (strings)
  - Reject modeling is hard



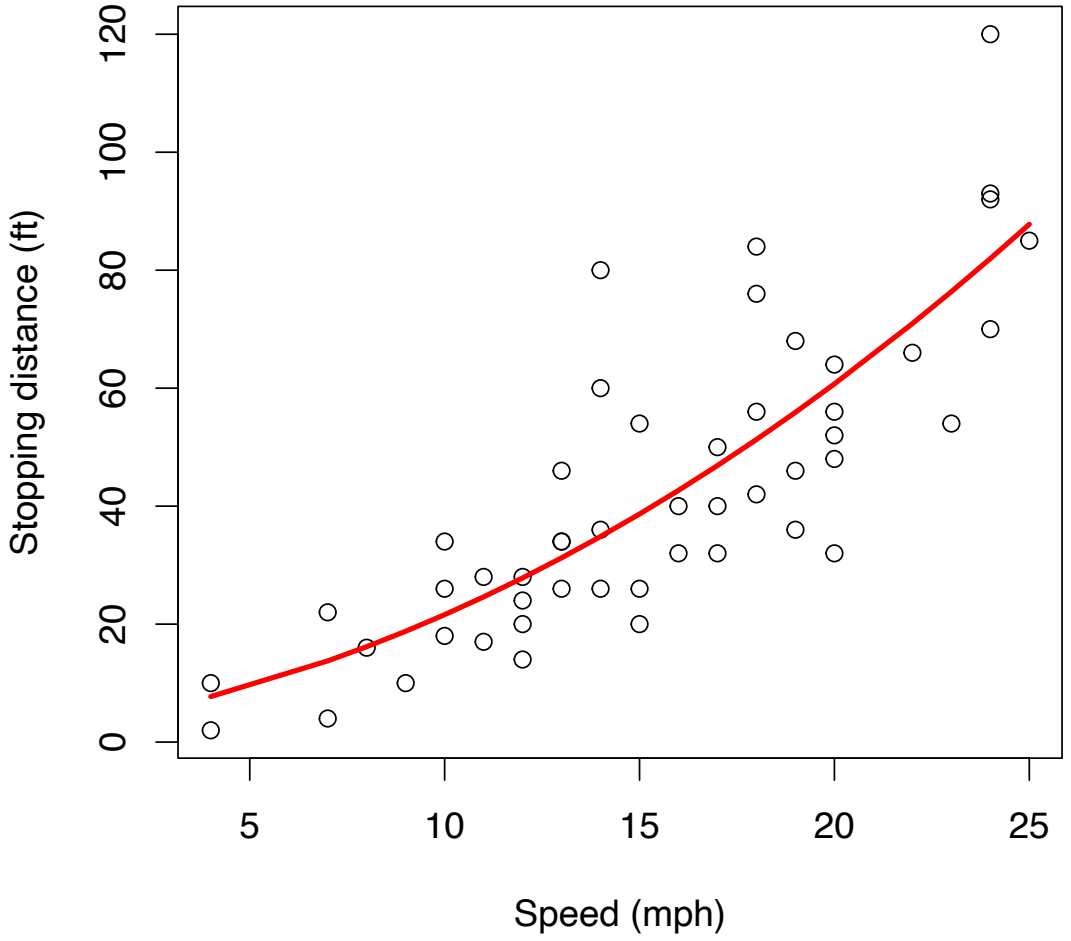
Fine-Tuning (Fit) in R (Statistics Package)



Make deep nets look like regression
(Not much programming)

- Notational Conventions:
  - Observations: Circles
  - Models (f): Red Lines
- Prediction: $f(x)$
  - Use model (f) to map
    - input x (speed) to
    - output y (stopping distance)
  - For linear regression,
    - f is a polynomial
  - For gft,
    - f is typically a model from a hub

Datasets
Example: Cars




```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

Two Columns:
1. cars$speed
2. cars$dist

Example of Datasets in HuggingFace

https://huggingface.co/datasets/dair-ai/emotion

```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

Hugging Face | Search models, datasets, users... | Models | Datasets | Spaces | Docs

Datasets: dair-ai / emotion | like 114

Tasks: Text Classification | Sub-tasks: multi-class-classification | Languages: English | Multilinguality: monolingual | Size Categories: 10K<n<100K | Language Creato

Annotations Creators: machine-generated | Source Datasets: original | Tags: emotion-classification | License: other

Dataset card | Files and versions | Community 6

Dataset Viewer | Auto-converted to Parquet | API | Go to dataset viewer

Subset: split (20k rows) | Split: train (16k rows)

x Search this dataset

y

text (string)	label (class label)
"i didnt feel humiliated"	0 (sadness)
"i can go from feeling so hopeless to so damned hopeful just from being around someone who cares and is awake"	0 (sadness)
"im grabbing a minute to post i feel greedy wrong"	3 (anger)
"i am ever feeling nostalgic about the fireplace i will know that it is still on the property"	2 (love)
"i am feeling grouchy"	3 (anger)
"ive been feeling a little burdened lately wasnt sure why that was"	0 (sadness)
"ive been taking or milligrams or times recommended amount and ive fallen asleep a lot faster but i also feel like so funny"	5 (surprise)

Downloads last month 7,358

Use in dataset library | Edit dataset card

Train in AutoTrain | Papers with Code

Evaluate models | HF Leaderboard

Homepage: github.com | Size of downloaded dataset files: 16.1 MB

Size of the auto-converted Parquet files: 28.2 MB | Number of rows: 436,809

Models trained or fine-tuned on dair-ai/emotion

aiknowyou/it-emotion-analyzer

https://github.com/kwchurch/CIKM_2023_tutorial

Emotion → GLUE (Cola)

https://huggingface.co/datasets/glue/viewer/cola/train

```
> head(cars)
  speed dist
1     4    2
2     4   10
3     7    4
4     7   22
5     8   16
6     9   10
```

Hugging Face | Search models, datasets, users... | Models | Datasets | Spaces | Docs | Solutions | Pricing

Datasets: glue | like 29

Tasks: natural-language-inference | acceptability-classification | text-classification-other-paraphrase-identification | +4 | Task Categories: text-classification | text-scoring | Languages: en | Multilinguali

Size Categories: 10K<n<100K | Licenses: cc-by-4-0 | Language Creators: unknown | Annotations Creators: unknown | Source Datasets: unknown

Dataset card | Files and versions

Terminology: Subset

Terminology: Splits

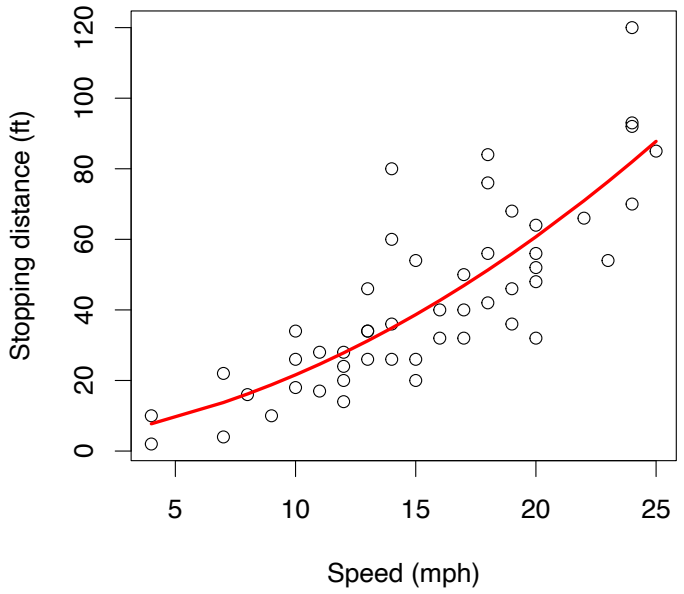
Dataset Preview

Subset: cola

Split: test | train | validation

sentence (string)	label (class label)	idx (int)
Our friends won't buy this analysis, let alone the next one we propose.	1 (acceptable)	0
One more pseudo generalization and I'm giving up.	1 (acceptable)	1
One more pseudo generalization or I'm giving up.	1 (acceptable)	2
The more we study verbs, the crazier they get.	1 (acceptable)	3
Day by day the facts are getting murkier.	1 (acceptable)	4
I'll fix you a drink.	1 (acceptable)	5
Fred watered the plants flat.	1 (acceptable)	6
Bill coughed his way out of the restaurant.	1 (acceptable)	7
We're dancing the night away.	1 (acceptable)	8
Herman hammered the metal flat.	1 (acceptable)	9
The critics laughed the play off the stage.	1 (acceptable)	10

glm (General Linear Models) in R (and Sklearn)




```
# Create the black points
plot(cars, xlab="Speed (mph)", ylab="Stopping distance (ft)")

g = glm(dist ~ poly(speed, 2), data=cars)

o = order(cars$speed)
# Show predictions as a red line
lines(cars$speed[o], predict(g,cars)[o], col="red", lwd=3)
```

glm: fit poly model with data

predict: $dist \approx g(cars\$speed)$

Hugging Face Search models, datasets, users...

Emotion Dataset

$classify: label \sim text$

Datasets: **emotion** like 18

Tasks: multi-class-classification text-classification-other-emotion-classification Task Categories: text-classification

Language Creators: machine-generated Annotations Creators: machine-generated Source Datasets: original

Dataset card Files and versions

Task/Loss

Columns

Row

Dataset Preview

Subset: default Split: train

text (string)	label (class label)
i didnt feel humiliated	0 (sadness)
i can go from feeling so hopeless to so damned hopeful just from being around someone who cares and is awake	0 (sadness)
im grabbing a minute to post i feel greedy wrong	3 (anger)
i am ever feeling nostalgic about the fireplace i will know that it is still on the property	2 (love)
i am feeling grouchy	3 (anger)
ive been feeling a little burdened lately wasnt sure why that was	0 (sadness)
ive been taking or milligrams or times recommended amount and ive fallen asleep a lot faster but i also feel like so funny	5 (surprise)
i feel as confused about life as a teenager or as jaded as a year old man	4 (fear)
i have been with petronas for years i feel that petronas has performed well and made a huge profit	1 (joy)
i feel romantic too	2 (love)
i feel like i have to make the suffering i m seeing mean something	0 (sadness)

Fine-Tuning (fit): Numerous Use Cases

$$f_{pre} + data \rightarrow f_{post}$$

- Flowers
  - f_{pre}: Resnet
  - $data$: flowers
- Emotion Classification
  - f_{pre}: https://huggingface.co/bert-base-uncase
  - $data$: https://huggingface.co/datasets/dair-ai/emotion
- GLUE
- SQuAD
- Machine Translation
- Speech Recognition
- Vision
- and much more

```
gft_fit --eqn 'classify: label ~ text' \
    --model H:bert-base-cased \
    --data H:emotion \
    --output_dir $outdir
```

f_{pre}: Pre-trained Model

f_{post}: Post-trained Model

```
7 g = glm(dist ~ poly(speed, 2), data=cars)
```

Fine-Tuning (fit) in R

GLUE Subsets

Subset	Dataset
COLA	H:glue,cola
SST2	H:glue,sst2
WNLI	H:glue,wnli
MRPC	H:glue,mrpc
QNLI	H:glue,qnli
QQP	H:glue,qqp
SSTB	H:glue,sstb
MNLI	H:glue,mnli

GLUE COLA SUBSET	
Sentence	**Label**
Bill sang himself to sleep.	1 (acceptable)
Bill squeezed the puppet through the hole.	1 (acceptable)
Bill sang Sue to sleep.	1 (acceptable)
The elevator rumbled itself to the ground.	0 (unacceptable)
If the telephone rang, it could ring itself silly.	1 (acceptable)
She yelled hoarse.	0 (unacceptable)
Ted cried to sleep.	0 (unacceptable)
The tiger bled to death.	1 (acceptable)

GFT Program for COLA

https://github.com/kwchurch/gft/blob/master/examples/fit_examples/model.HuggingFace/language/data.HuggingFace/glue/cola.sh

```
#!/bin/sh

echo hostname = `hostname`

gft_fit --model H:bert-base-cased \
    --data H:glue,cola \
    --metric H:glue,cola \
    --figure_of_merit matthews_correlation \
    --output_dir $1 \
    --eqn 'classify: label ~ sentence' \
    --num_train_epochs 3
```

GLUE COLA SUBSET	
Sentence	**Label**
Bill sang himself to sleep.	1 (acceptable)
Bill squeezed the puppet through the hole.	1 (acceptable)
Bill sang Sue to sleep.	1 (acceptable)
The elevator rumbled itself to the ground.	0 (unacceptable)
If the telephone rang, it could ring itself silly.	1 (acceptable)
She yelled hoarse.	0 (unacceptable)
Ted cried to sleep.	0 (unacceptable)
The tiger bled to death.	1 (acceptable)

Simple Equations Cover Many Cases of Interest
GLUE: A Popular Benchmark

Subset	Dataset	Equation
COLA	H:glue,cola	$classify: label \sim sentence$
SST2	H:glue,sst2	$classify: label \sim sentence$
WNLI	H:glue,wnli	$classify: label \sim sentence$
MRPC	H:glue,mrpc	$classify: label \sim sentence1 + sentence2$
QNLI	H:glue,qnli	$classify: label \sim sentence1 + sentence2$
QQP	H:glue,qqp	$classify: label \sim question + sentence$
SSTB	H:glue,sstb	$regress: label \sim question1 + question2$
MNLI	H:glue,mnli	$classify: label \sim premise + hypothesis$

https://github.com/kwchurch/CIKM_2023_tutorial

Equation Keywords ≈ Pipeline Tasks

Benchmark	Subset	Dataset	Equation
GLUE	COLA	H:glue,cola	$classify: label \sim sentence$
SQuAD 1.0		H:squad	$classify\_spans: answers \sim question + context$
SQuAD 2.0		H:squad_v2	$classify\_spans: answers \sim question + context$
CONLL2003	POS	H:conll2003	$classify\_tokens: pos\_tags \sim tokens$
	NER	H:conll2003	$classify\_tokens: ner\_tags \sim tokens$
TIMIT		H:timit_asr	$ctc: text \sim audio$
Amazon Reviews		H:amazon_reviews_multi	$classify: label \sim question + sentence$
VAD		C:$gft/datasets/VAD/VAD	$regress: Valence + Arousal + Dominance \sim Word$

gft Cheat Sheet
(General Fine-Tuning)

4+1 Functions

1. gft_fit: $f_{pre} \rightarrow f_{post}$ (fine-tuning)
   - 4 Arguments, --output_dir, --metric, --splits
   - (plus most args in most hubs)
2. gft_predict: $f(x) \rightarrow \hat{y}$ (inference)
   - Input: 4 Arguments (x from data or stdin)
   - Output: $\hat{y}$ for each x
3. gft_eval: Score model on dataset
   - Input: 4 Arguments, --split, --metric, …
   - Output: Score
4. gft_summary: Find good stuff
   - Input: 4 Arguments
   - (may include: __*contains*__, __*infer*__)
5. gft_cat_dataset: Output data to *stdout*
   - Input: 4 Arguments (--data, --eqn)

4 Arguments

✓ --data

✓ --model

➢ **--eqn** $\boldsymbol{task: y \sim x_1 + x_2}$

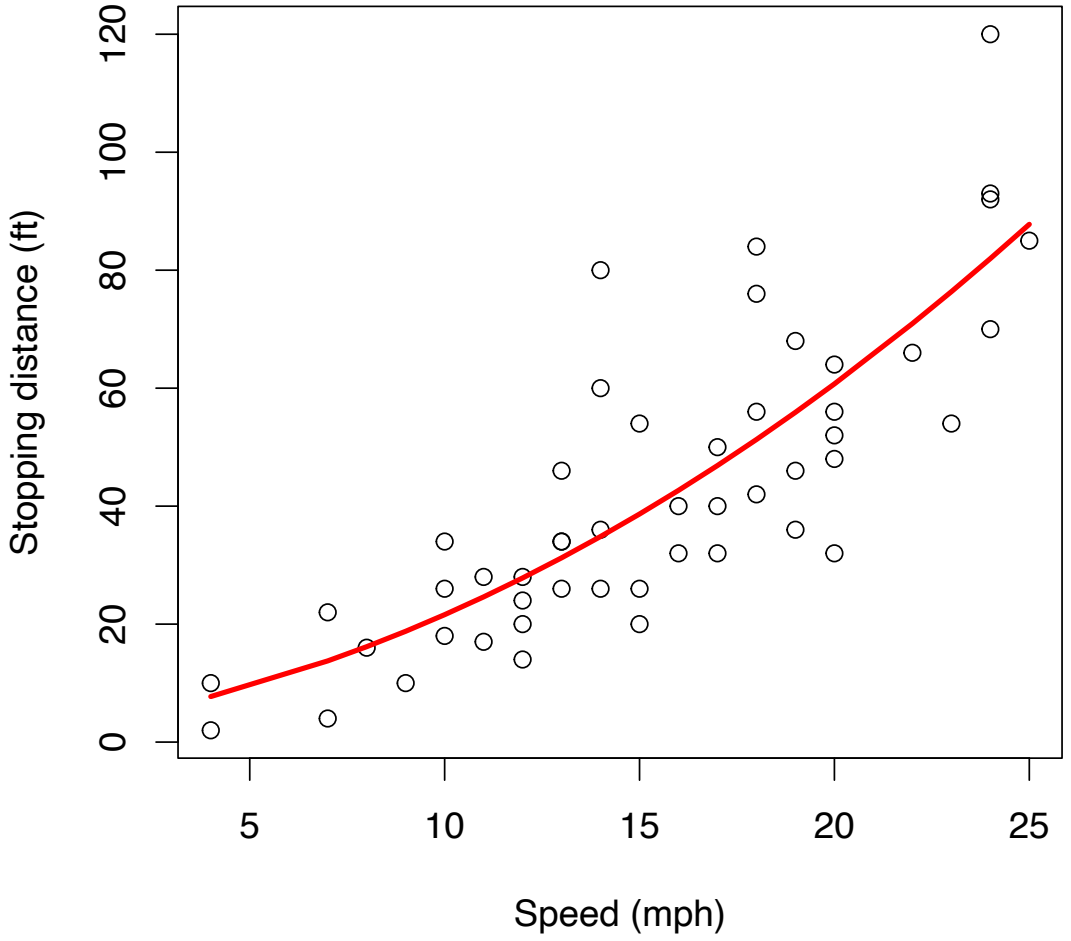
- **--task**
  1. classify (text-classification)
  2. classify_tokens (token-classification)
  3. classify_spans (QA, question-answering)
  4. classify_audio (audio-classification)
  5. classify_images (image-classification)
  6. regress
  7. text-generation
  8. MT (translation)
  9. ASR (ctc, automatic-speech-recognition)
  10. fill-mask

Fit a model (*g*) to data (*cars*)

Regression in *R*:

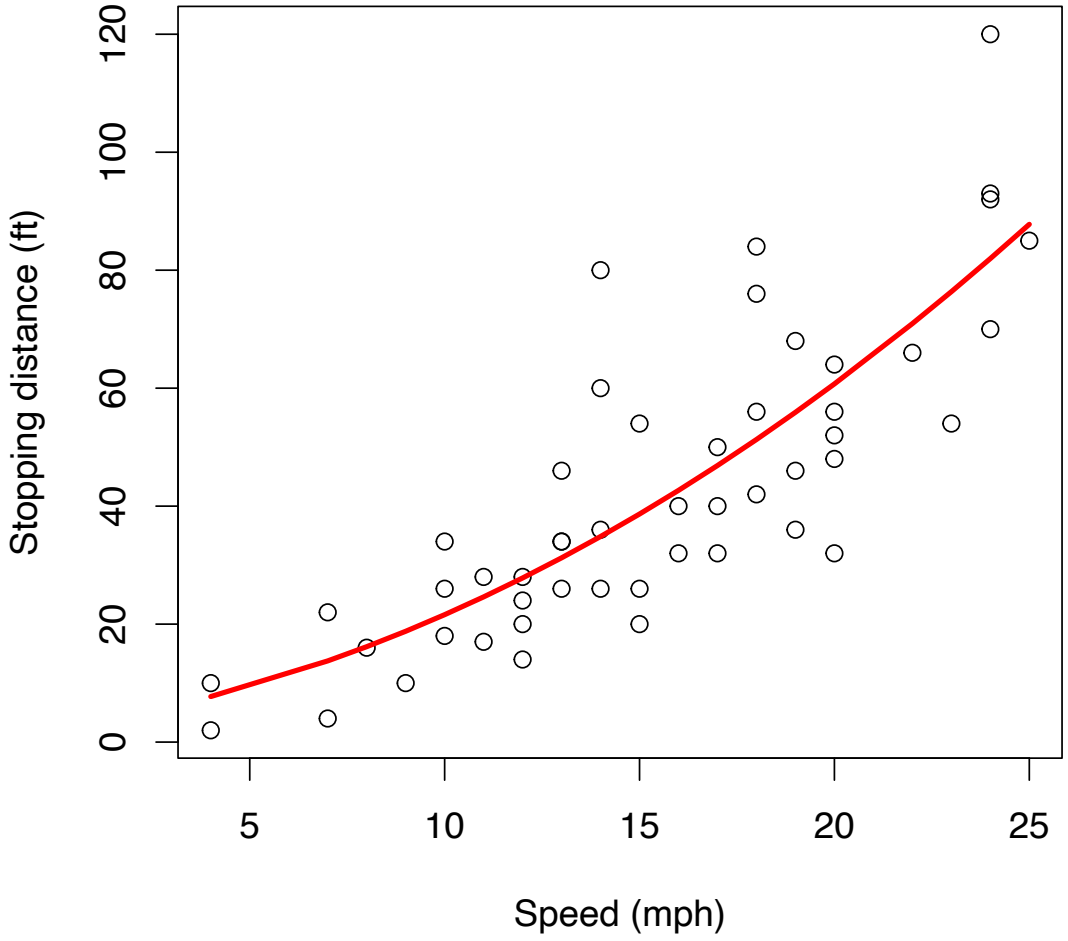
Summarize (almost) anything

```
> plot(cars, xlab="Speed (mph)", ylab="Stopping distance (ft)")
> g = glm(dist ~ poly(speed,2), data=cars)
> o = order(cars$speed)
> lines(cars$speed[o], predict(g,cars)[o], col="red", lwd=3)
```



Regression in *R*:

Summarize (almost) anything




```
> summary(cars)
     speed           dist       
 Min.   : 4.0   Min.   :  2.00  
 1st Qu.:12.0   1st Qu.: 26.00  
 Median :15.0   Median : 36.00  
 Mean   :15.4   Mean   : 42.98  
 3rd Qu.:19.0   3rd Qu.: 56.00  
 Max.   :25.0   Max.   :120.00  
```

Regression in *R*:

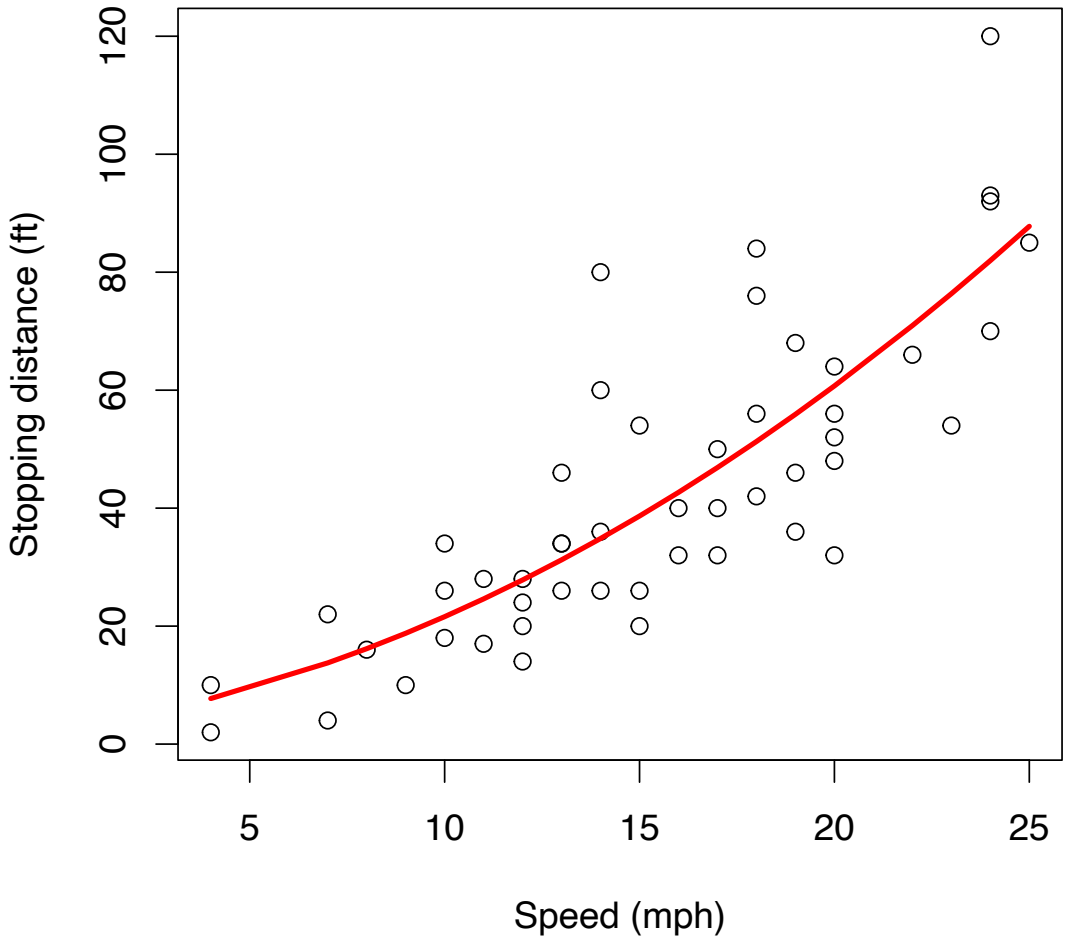
Summarize (almost) anything

Fit a model (*g*) to data (*cars*)

Predict

Summarize a model (*g*)

Opportunity to Improve *g*




```
> plot(cars, xlab="Speed (mph)", ylab="Stopping distance (ft)")
> g = glm(dist ~ poly(speed,2), data=cars)
> o = order(cars$speed)
> lines(cars$speed[o], predict(g,cars)[o], col="red", lwd=3)
> summary(g)

Call:
glm(formula = dist ~ poly(speed, 2), data = cars)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-28.720   -9.184   -3.188    4.628   45.152

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)       42.980      2.146  20.026  < 2e-16 ***
poly(speed, 2)1  145.552     15.176   9.591 1.21e-12 ***
poly(speed, 2)2   22.996     15.176   1.515    0.136
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for gaussian family taken to be 230.3131)

    Null deviance: 32539  on 49  degrees of freedom
Residual deviance: 10825  on 47  degrees of freedom
AIC: 418.77

Number of Fisher Scoring iterations: 2
```

https://github.com/kwchurch/CIKM_2023_tutorial

gft_summary

- Summarize almost anything
  - Models
  - Datasets

Tasks

- classify, text-classification — $y \in \{0,1,2, \ldots\}$
- regress — $y \in \mathbb{R}$ or $y \in \mathbb{R}^N$
- QA, Question Answering, classify spans — y for each start/end of span
- token classification — y for each token
  - NER (Named Entity Recognition)
  - POS (Part of Speech Tagging)
- translation, MT
- ASR, Automatic Speech Recognition, ctc — y for each phoneme

gft Cheat Sheet
(General Fine-Tuning)

4+1 Functions

1. gft_fit: $f_{pre} \rightarrow f_{post}$ (fine-tuning)
   - 4 Arguments, --output_dir, --metric, --splits
   - (plus most args in most hubs)
2. gft_predict: $f(x) \rightarrow \hat{y}$ (inference)
   - Input: 4 Arguments (x from data or stdin)
   - Output: $\hat{y}$ for each x
3. gft_eval: Score model on dataset
   - Input: 4 Arguments, --split, --metric, …
   - Output: Score
4. gft_summary: Find good stuff
   - Input: 4 Arguments
   - (may include: __*contains*__, __*infer*__)
5. gft_cat_dataset: Output data to *stdout*
   - Input: 4 Arguments (--data, --eqn)

4 Arguments

✓ --data

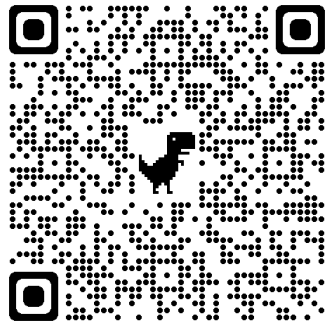
✓ --model

✓ --eqn $task: y \sim x_1 + x_2$

✓ **--task**

1. classify (text-classification)
2. classify_tokens (token-classification)
3. classify_spans (QA, question-answering)
4. classify_audio (audio-classification)
5. classify_images (image-classification)
6. regress
7. text-generation
8. MT (translation)
9. ASR (ctc, automatic-speech-recognition)
10. fill-mask

Easy Conclusions: Inference (Colab) HuggingFace Pipelines



- See here for a list of currently supported tasks
  - machine learning:
    - classification, regression,
    - token classification,
    - classify spans (Q&A), fill mask
  - speech:
    - speech-to-text (automatic speech recognition (ASR)),
    - text-to-speech (speech synthesis)
    - audio classification
  - vision:
    - image classification, video classification, image segmentation, image to text, visual question answering
  - natural language:
    - text classification
    - question answering
    - fill mask (Cloze task)
    - machine translation (MT)
    - feature extraction