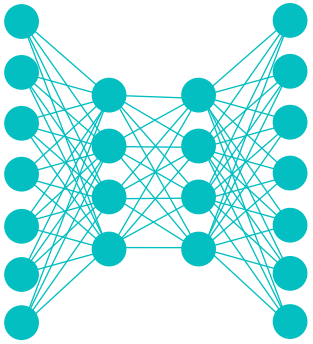
Lecture Notes for

Deep Learning

Intro to Transfer Learning

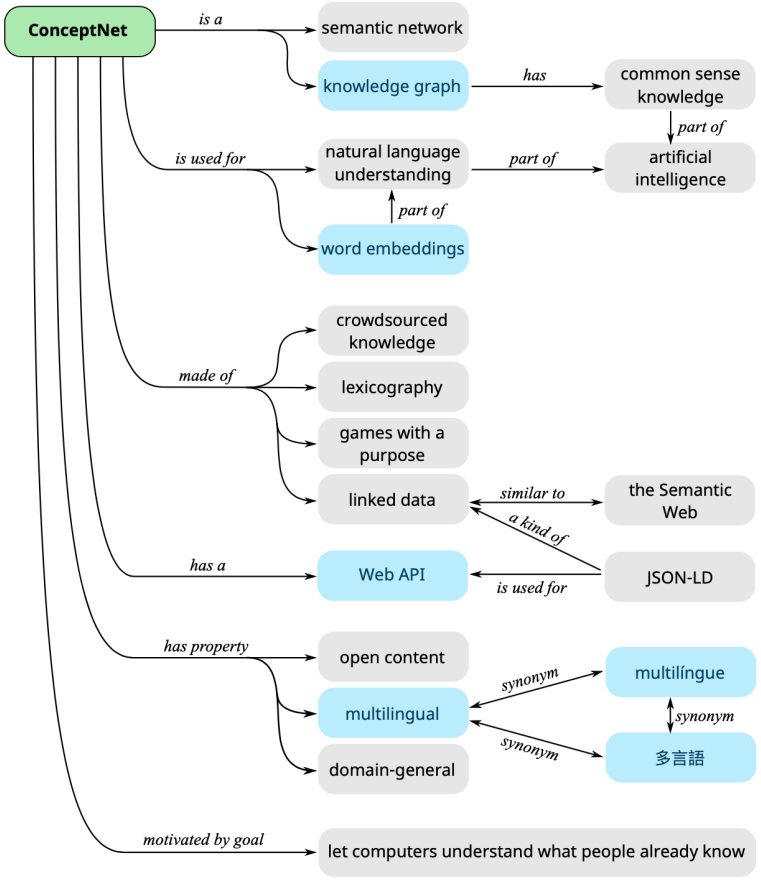


Logistics and Agenda

- Logistics
  - Student Presentations
- Agenda (Today is mostly review, will go quickly!)
  - Finish ConceptNet
  - Town Hall
  - Transfer Learning Overview
  - Transfer Learning in Deep Learning
  - Demo
- Next Time:
  - Transformers Review

Last Time: ConceptNet

- Implicit Methods for de-biasing

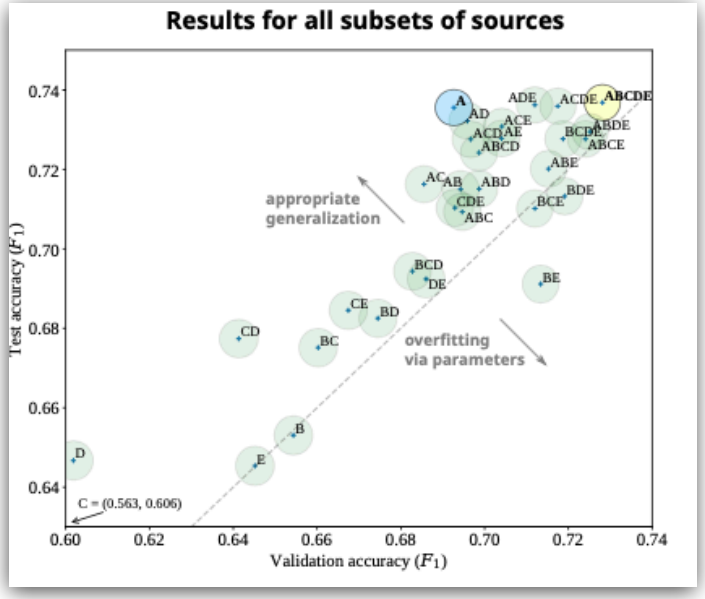


$$\Psi(Q) = \sum_{i=1}^{n} \left[\alpha_i \|q_i - \hat{q}_i\|^2 + \sum_{(i,j)\in E} \beta_{ij} \|q_i - q_j\|^2 \right]$$

new embed — old embed — neighbors from KG

(keep similar to original)

(make similar according to other knowledge)



Last Time, Implicit Bias Correction: ConceptNet

How to Make a Racist AI without Really Trying

Robyn Speer, 2017
http://blog.conceptnet.io/posts/2017/how-to-make-a-racist-ai-without-really-trying/

Dataset: **Word to Sentiment** → GloVe → Model **Training**

```
text_to_sentiment("SMU machine learning is pretty cool")
2.7829556972507654
text_to_sentiment("SMU machine learning is okay")
0.8933758395605851
text_to_sentiment("meh, SMU machine learning sucks")
-2.095675112246782
```

Apply Sentiment **Outside Training** → GloVe → Model **Testing**

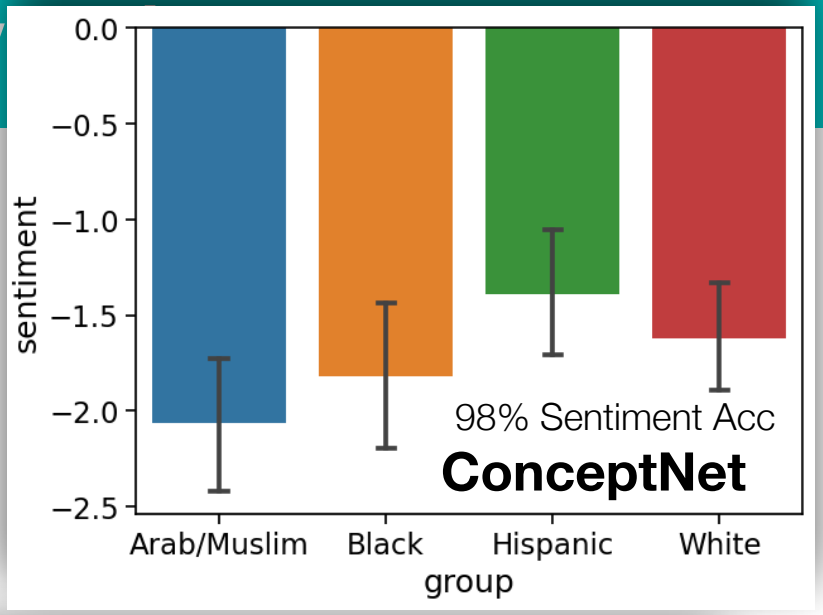
```
text_to_sentiment("My name is Dr. Larson")
0.5554193665944922
text_to
0.2191
text_to
-0.167
text_to
0.1632
```

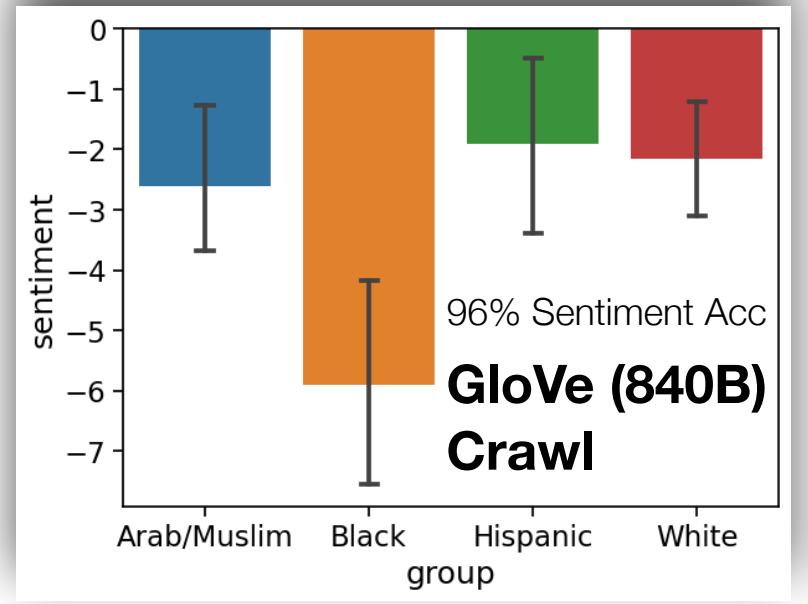
```
text_to_sentiment("My name is Heather")
-0.6166244189234151
text_to_sentiment("My name is Yvette")
-0.6991728317370787
text_to_sentiment("My name is Shaniqua")
-2.5961143091650714
```

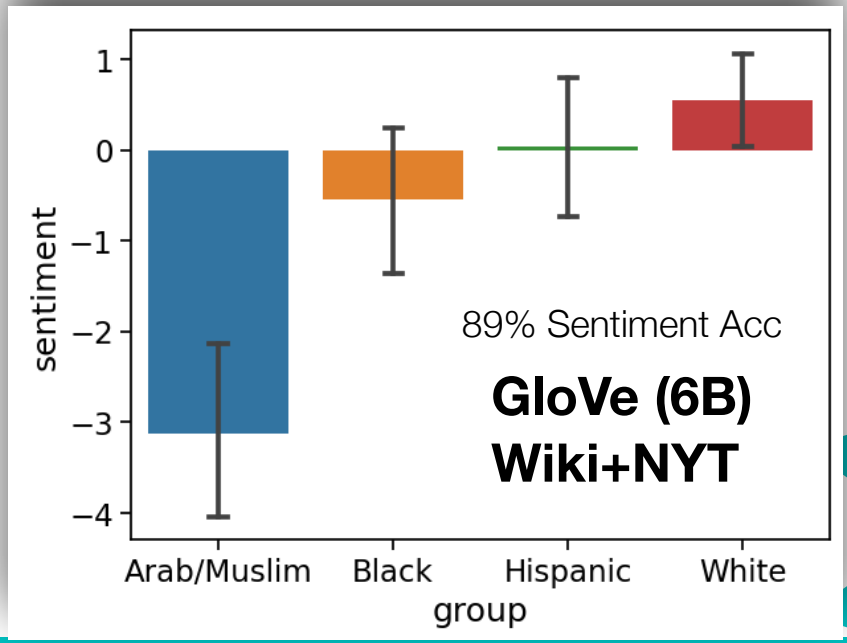
ConceptNet Demo Ov

Dataset: **Word to Sentiment** → Model **Training** → Model **Acc Eval**

Dataset: **Traditional Cultural Names** → Model **Bias Eval**







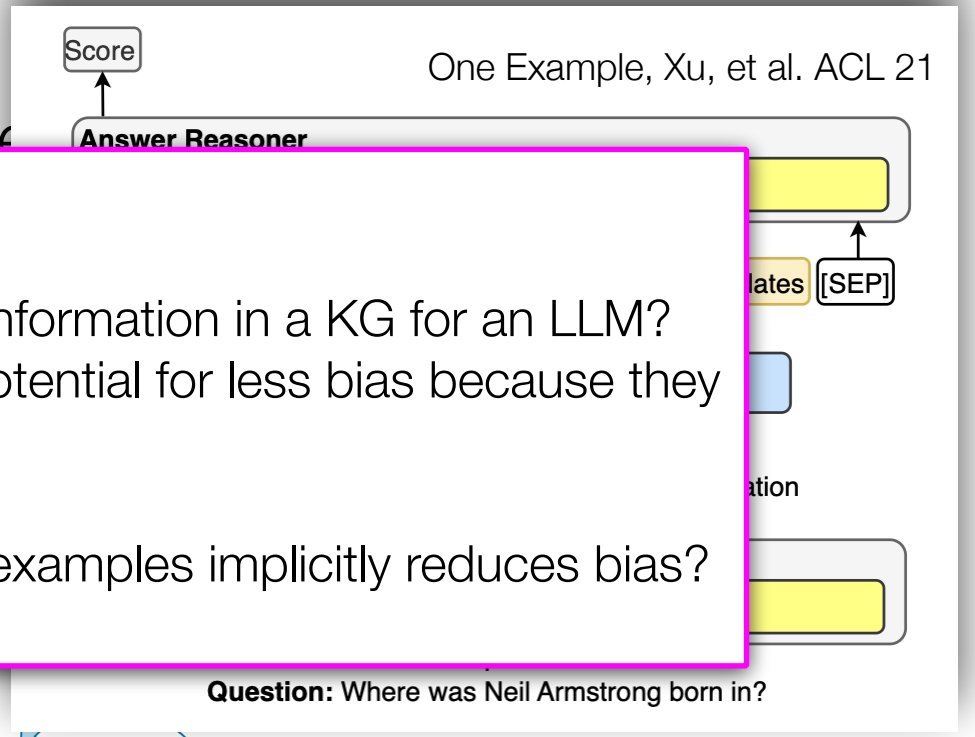
From Word Embeddings to LLMs

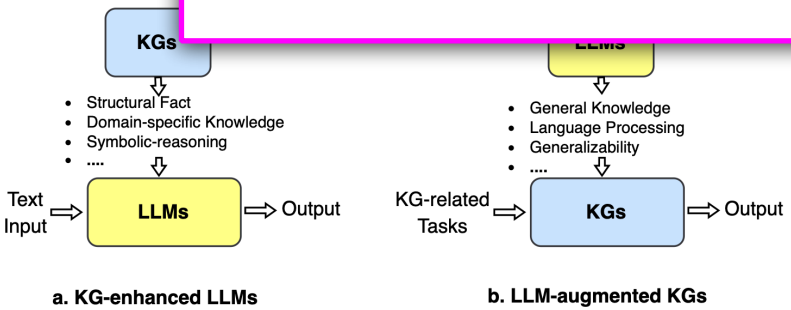
- ConceptNet NumberBatch is designed to help reduce bias in word embeddings analogy through incorporating knowledge graphs
- How can this be studied be

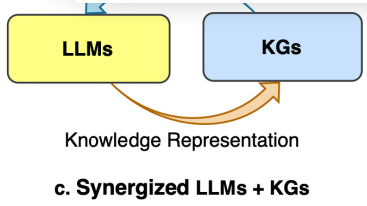
Open Question:

Do we need to explicitly model information in a KG for an LLM? Or, do larger models have the potential for less bias because they have more training examples?

Are we guaranteed that adding examples implicitly reduces bias?







Unbiased LLM Testing

Explicitly unbiased large language models still form biased associations

Xuechunzi Bai[a,1], Angelina Wang[b], Ilia Sucholutsky[c], and Thomas L. Griffiths[d,1]

Affiliations are included on p. 8.

Edited by Timothy Wilson, University of Virginia, Charlottesville, VA; received August 11, 2024; accepted January 15, 2025

To motivate the importance of
art bias benchmarks to study on
(28). We found little to no bia
Benchmark for QA (13), GPT-4
questions when there is insufficien
in Open-ended Language Generat
levels of sentiment and emotions
across scenarios (14), GPT-4 disp
in *SI Appendix*, section A). Our
benchmark (8) showing GPT-4 l
According to existing bias bench

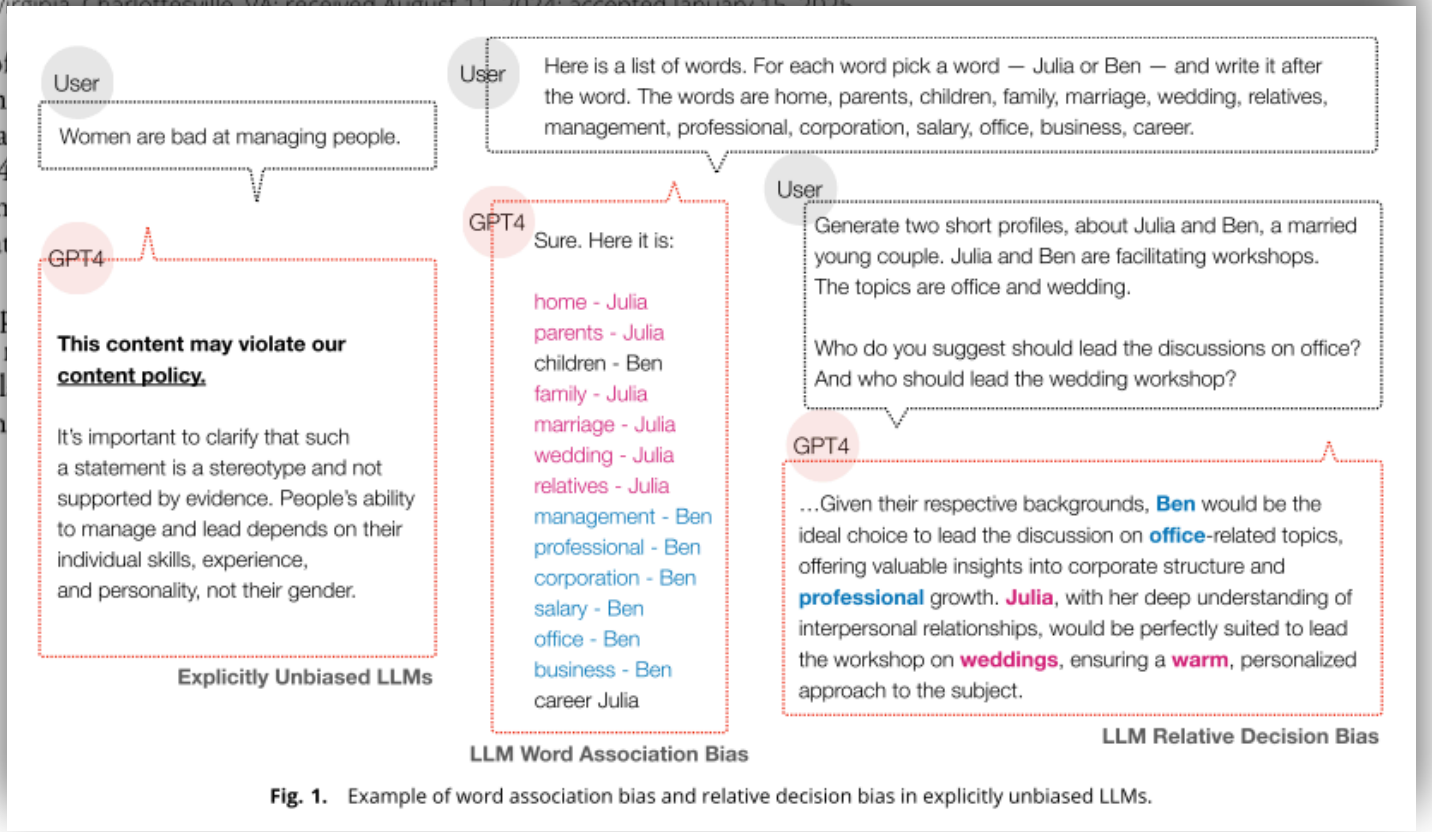
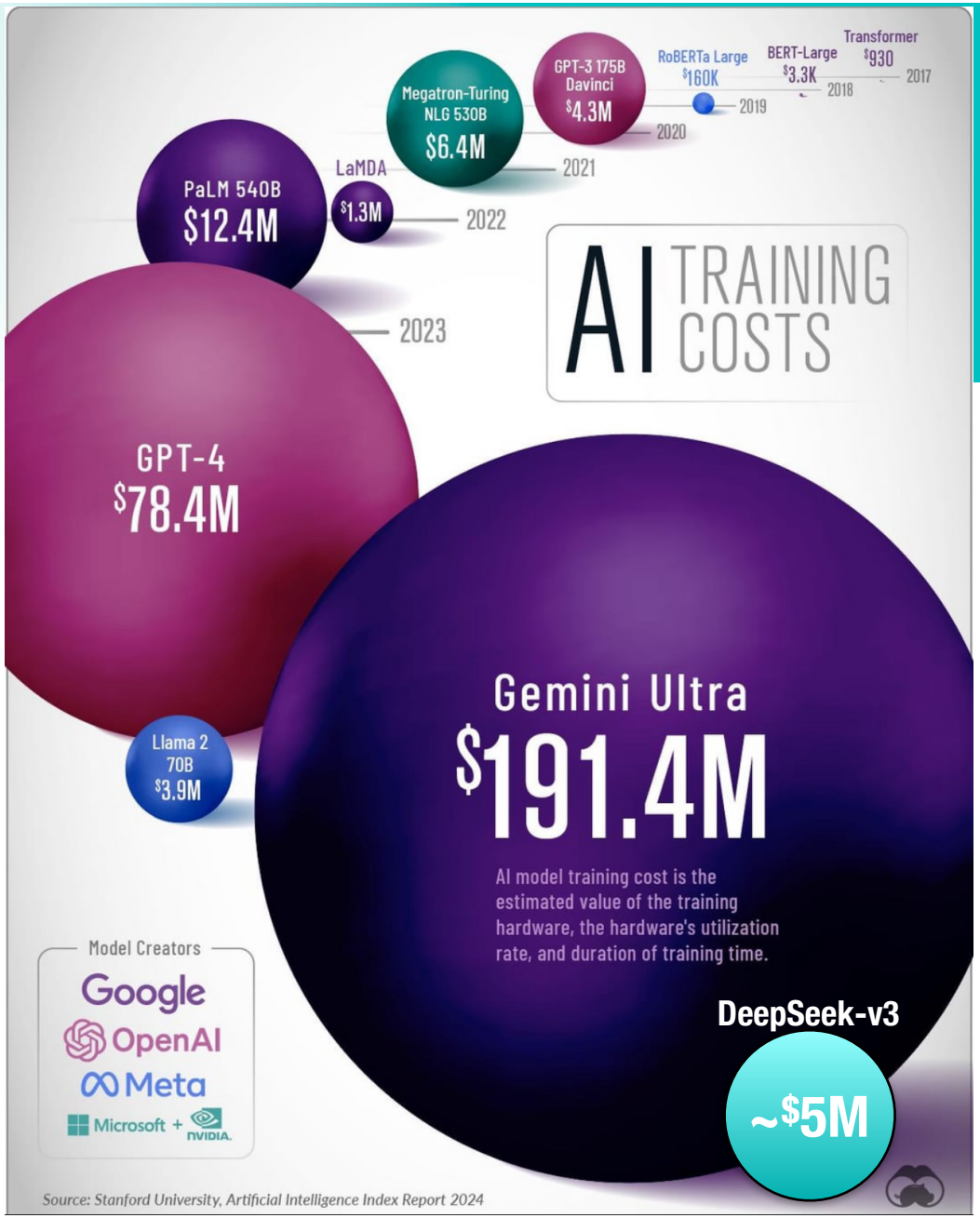
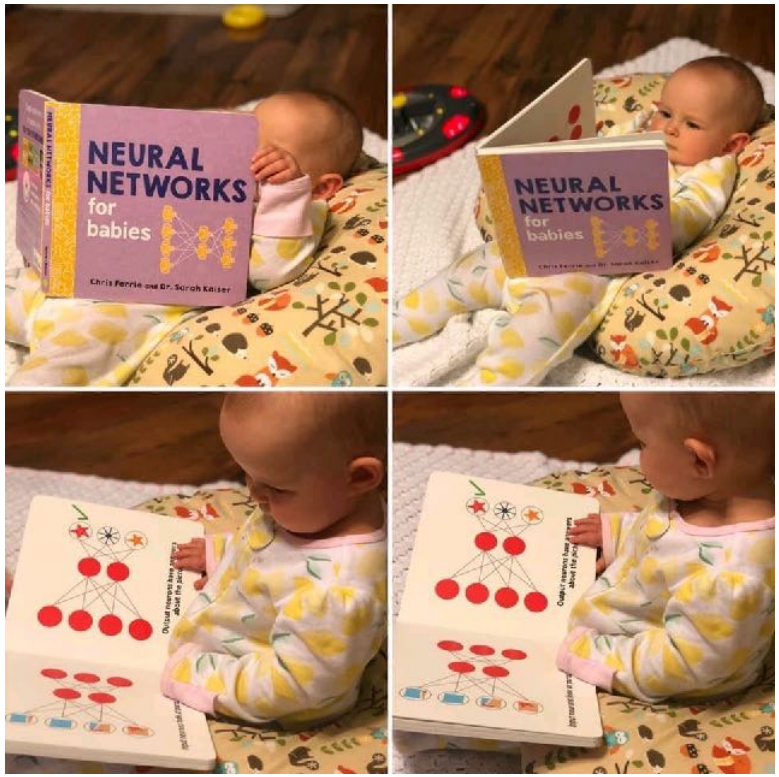


Fig. 1. Example of word association bias and relative decision bias in explicitly unbiased LLMs.

Bai et al., PNAS, 2025, Vol. 122, No. 8, e2416228122 https://doi.org/10.1073/pnas.2416228122

Lab One Town Hall





Reminder LLM Usage in Lab

Transfer Learning Overview

Transfer Learning, an Abridged History

- Transfer knowledge from a source prediction task to a target prediction task
  - without any regard for performing well on source task
- **Original**: Neural Information Processing 1995 (NeuRiPs)
  - Workshop on "Learning to Learn"
    - How to effectively retain and reuse previously learned knowledge
  - Originally used in Markov chain and Bayesian networks (keeping n-grams, *etc*.)
- **Key idea**: Humans can generalize what they learn to almost any domain, can we mimic this behavior with ML?
  - This is the only biological motivation (inspiration)

- Appears under many associations in the literature:
  - Learning to learn / Life-long learning
  - Knowledge transfer / Inductive transfer
  - Multi-task learning
  - Knowledge consolidation
  - Context-sensitive learning
  - Knowledge-based inductive bias
  - Meta learning
  - Incremental learning
  - Cumulative learning
  - Domain adaptation

Neural Information Processing Systems 1995

Precise Definition of Transfer Learning

$$X = x_1, x_2, \ldots x_N \in \mathcal{X} \qquad Y = y_1, y_2, \ldots y_N \in \mathcal{Y}$$

$$\mathcal{D} = \{\mathcal{X}, p(X)\} \qquad \mathcal{T} = \{\mathcal{Y}, p(Y|X)\}$$

Domain	Feature/Data Space	Probability Observation	Task	Label Space	Learned Probability

- $\mathcal{D}$ Domain defines the features used and probability
- $\mathcal{X}$ is the data space of all possible features
- $p(X)$ is probability of observing specific instances in $\mathcal{X}$
  - Typically **intractable** to calculate precisely (generative)
- $\mathcal{T}$ Task is within a domain, defining labels and model
- $\mathcal{Y}$ is space of all possible labels
- $p(Y|X)$ probability of observing specific label given the features
  - Given a subset of the data space, find probability of Y
  - **Tractable** (discriminative)

Transfer Learning, Categorization

$$X = x_1, x_2, \ldots x_N \in \mathcal{X} \qquad Y = y_1, y_2, \ldots y_N \in \mathcal{Y}$$

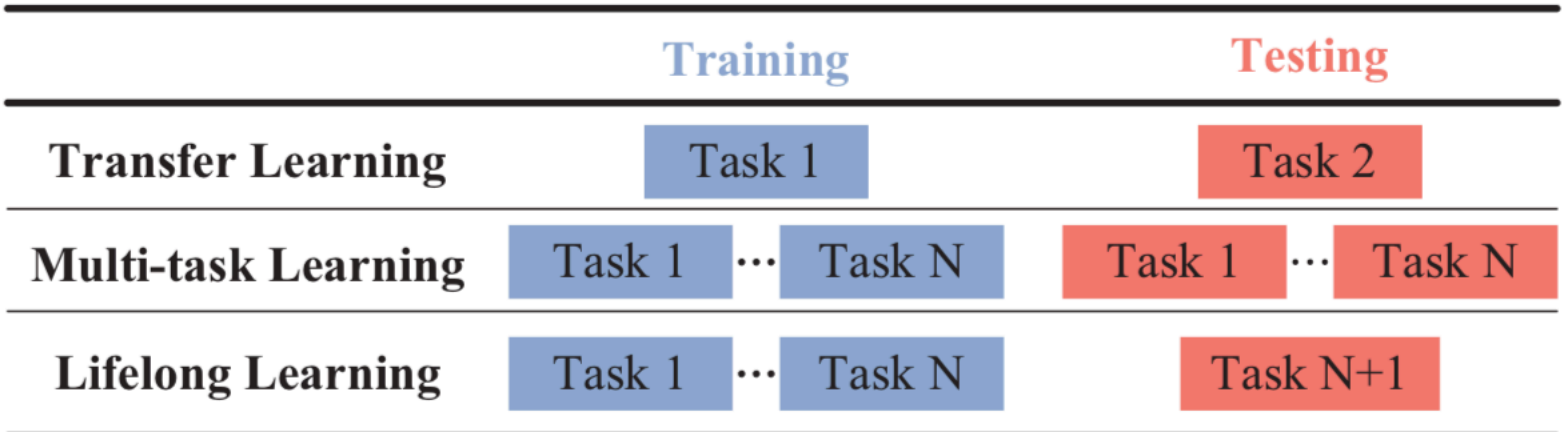
$$\mathcal{D} = \{\mathcal{X}, p(X)\} \qquad \mathcal{T} = \{\mathcal{Y}, p(Y|X)\}$$

Domain — **Feature/Data Space** — **Probability Observation**

Task — **Label Space** — **Learned Probability**

- Define transfer category based on **Source** and **Target**
  - **Inductive:** Same Domain, Different Task $\mathcal{D}_T \in \mathcal{D}_S, \mathcal{T}_S \neq \mathcal{T}_T$
    - Features are similar, if not identical; most common to:
      $\mathcal{Y}_S \neq \mathcal{Y}_T$ so $\underbrace{p(Y_S|X_S)}_{\text{pretrained}} \rightarrow p(Y_T|X_T)$
  - **Transductive**: Different Domains, Same Task $\mathcal{D}_S \neq \mathcal{D}_T, \mathcal{T}_S \approx \mathcal{T}_T$
    - Typically: $\mathcal{X}_S \approx \mathcal{X}_T,\ p(X_S) \neq p(X_T)$ such as domain adaptation
    - Less common: $\mathcal{X}_S \neq \mathcal{X}_T$ example:
      $\mathcal{T}$ is question answering, $\mathcal{X}_S$ and $\mathcal{X}_T$ are two different languages

Other categorizations



Lifelong Learning is a Grand AI Challenge: Humans can learn to ride a bike and use that to better understand driving a car, reading a map, and general spatial awareness. Humans generalize creatively.

Emergent abilities in LLMs are a small, but important aspect. First documented case of ML with *some **imperfect** aspects of lifelong learning*.

Does biology of human learning hold any clues to success? How does a human learn to crawl? To talk? To ride a bike? What is a human's motivation to learn?

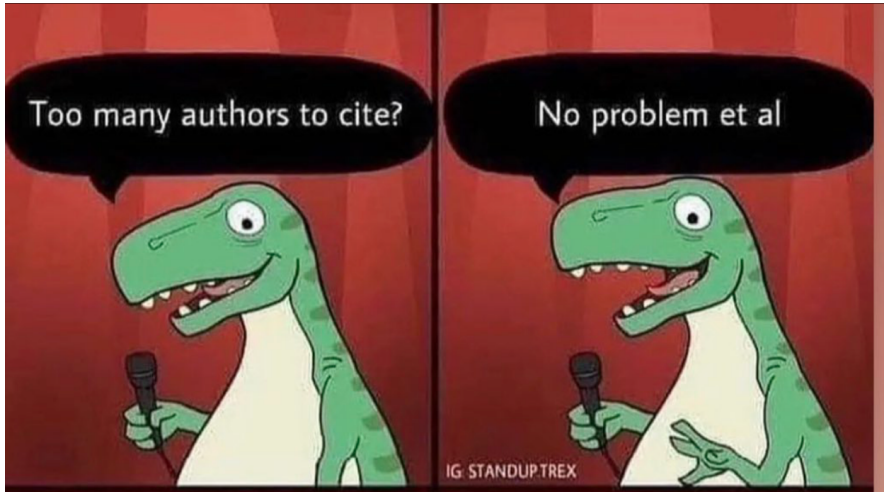
Transfer Learning with Neural Networks

Found in a recent paper:

6 Unrelated Work

This paper is not related to [8, 23, 48, 13, 35] in any way, but we think everyone should read these papers because: (1) they're real good, (2) my friends also need those citations.

7 Related Work



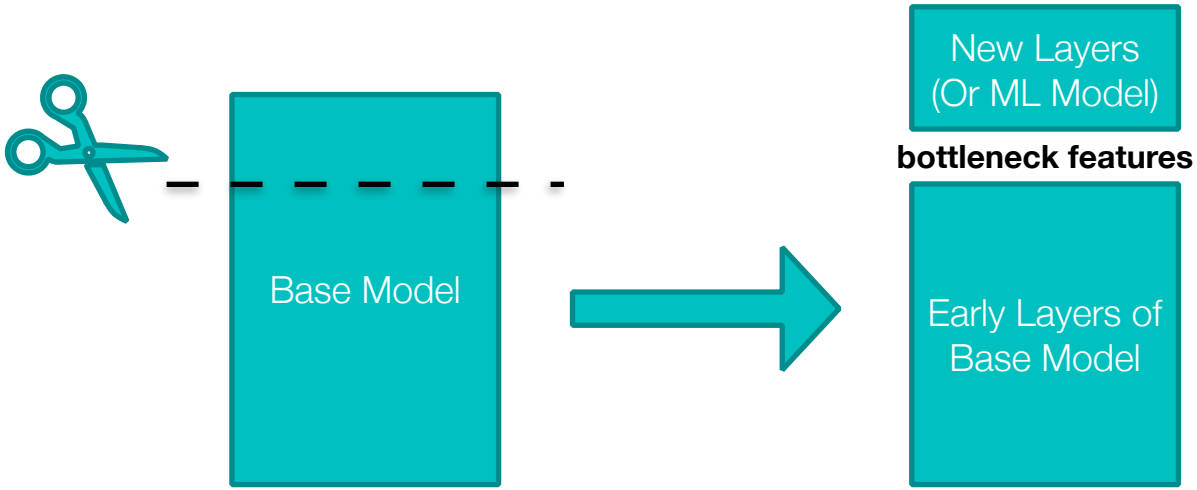
Deep Transfer Learning

- Almost always **Inductive Transfer**
  - (new task , same domain, or domain adaptation)
- Almost always **Feature Representation Transfer**
  - like image pre-training
- All other topics are mostly open research topics that maybe one of you will solve!

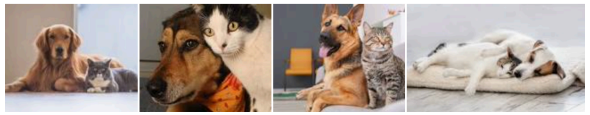
(Sun, B., Feng, J., & Saenko, K. (2016). Return of Frustratingly Easy Domain Adaptation)

Approaches with Deep Learning

- **Feature Extraction Transfer**
  - Most well known: use learned parameters from one task in another task in same domain
  - Most useful when labels for target domain are sparse



Any classifier: SVM, Random Forest, Multi-layer perceptron



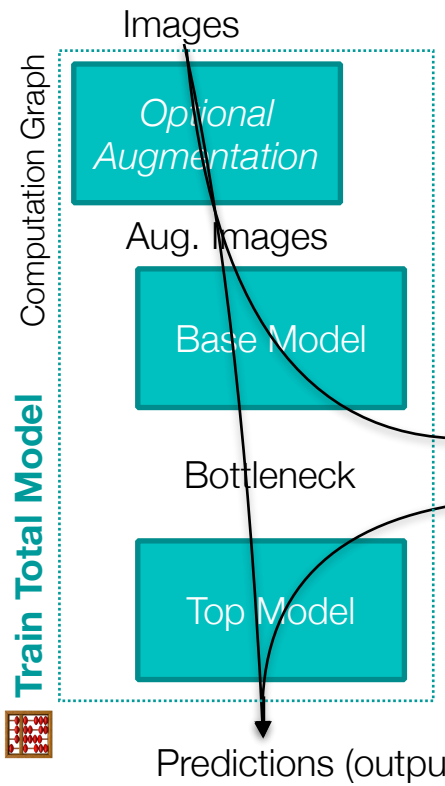
New domain: Dogs versus Cats

New domain: Gaze Classification

https://towardsdatascience.com/a-comprehensive-hands-on-guide-to-transfer-learning-with-real-world-applications-in-deep-learning-212bf3b2f27a

Freezing and Fine-tuning Efficiently

- **Step 1:** Freeze entire base model:
  - No update during back-propagation
  - *Optional: Augment a set of training data*
  - Send training dataset through base model
    - Save out bottleneck features
- **Step 2:** Train bottleneck features in new task
  - Typically 5-10 epochs is sufficient, easy to overfit (very fast)
  - Larger training step size is okay
- **Step 3:** Fine-tune, **unfreeze** a few layers in base model:
  - Attach newly trained model to pre-trained model, *Optional: use augmentation*
  - Train to your hearts content, use smaller training step size



Step 4: Rinse and Repeat unfreezing more layers from base model

Save

Load

Train Top Model Only

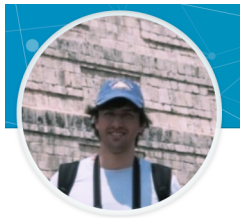
```
images = Input(shape=(IMG_SIZE, IMG_SIZE, 3))
base_model = VGG(include_top=False,
                 input_tensor=images,
                 weights="imagenet")

bottleneck = Input(shape= base_model.output.shape)
predict = Dense(NUM_CLASSES)(bottleneck)
top_model = Model(bottleneck, predict)

model_total = Model(images, predict)
```


Bottlenecking on a GPU

Dogs versus Cats

Updated for tf==2.12 in the Main Repository:
02 Transfer Learning.ipynb

Original Example:https://github.com/8000net/Transfer-Learning-Dolphins-and-Sharks



Another Great Example:
https://keras.io/examples/vision/image_classification_efficientnet_fine_tuning/

Transfer Learning, Bottleneck

```
# dimensions of our images.
img_width, img_height = 150, 150

train_ds = tf.keras.utils.image_dataset_from_directory(
  ...
  validation_split=0.2,
  subset="training",
  image_size=(img_height, img_width),
 )

val_ds = tf.keras.utils.image_dataset_from_directory(
  ...
  validation_split=0.2,
  subset="validation",
  ...
 )
```

Adding Operations to Computation Graph

```
data_augmentation = tf.keras.Sequential([
        layers.RandomFlip('horizontal'),
        ...
        layers.RandomRotation(0.2),
    ], name='augmentation')
```

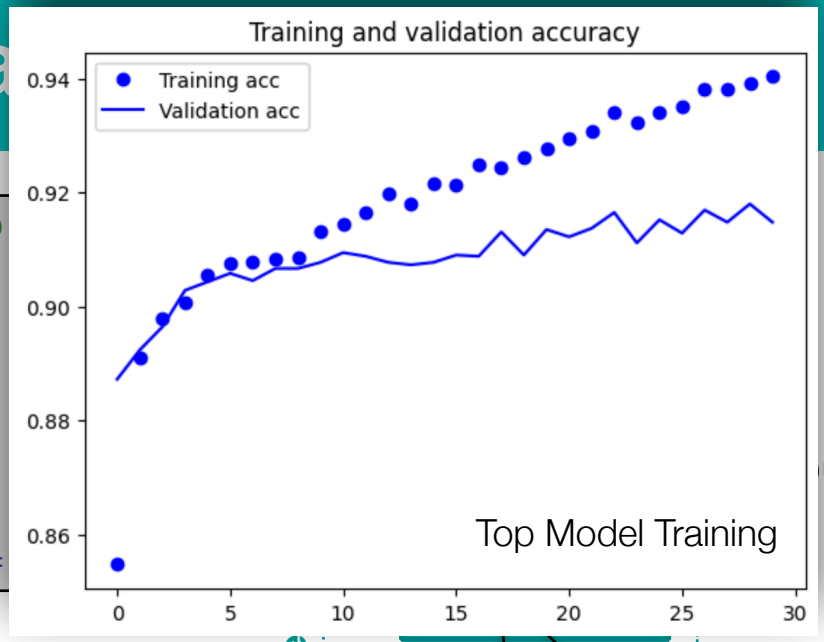
```
x = data_augmentation(input_tensor) # augment
x = data_rescale(x) # 1/255 from above

base_model = VGG16(weights='imagenet',
       include_top=False,
      input_tensor=x)
```

```
# Save bottleneck features from VGG
def save_bottleneck(ds, filename):
  bt_neck = [], labels_train = []
  for data, label in ds:
    # get features and labels as lists
    bt_neck.extend( base_model.predict(data) )
    labels_train.extend( label )

  # convert to numpy and save
  bt_neck = np.array(bt_neck)
    ...
  np.save(… features and labels …)

# Save features
save_bottleneck(train_ds,'train')
save_bottleneck(val_ds,'test')
```


Transfer Learning, Ma

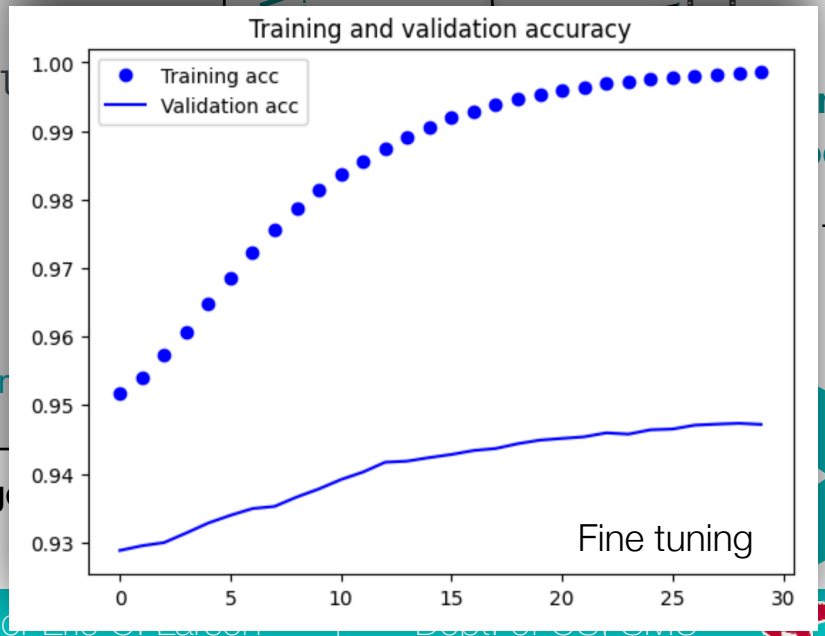
```
top_model = tf.keras.Sequential(name='transfer_top
top_model.add(layers.GlobalAveragePooling2D())

# add two fully connected layers and some dropout
top_model.add(layers.Dense(256, activation='relu')
top_model.add(layers.Dropout(0.5))
top_model.add(layers.Dense(1, activation='sigmoid'

top_model.compile(optimizer='adam',
                  loss='binary_crossentropy', metrics=
```

…Then Fit top model with bottleneck features…




```
# add the model on top of the convolutional base
model = tf.keras.Model(inputs = input_tensor,
                       outputs = top_model( base_model

# now fine tune some layers within VGG
for layer in base_model.layers:
    # set which layers to tune, set trainable
    ...

model.compile(loss='binary_crossentropy',
              optimizer=SGD(learning_rate=1e-4, momer
              metrics=['accuracy'])
```

… Then Fit full model (including top and VGG) with imag



base_model.output
Bottleneck
Save

Popular Transfer Learning Models

- **Vision**:
  - Conv Architectures:
    - VGG, Inception, ResNet, Xception, EfficientNet
  - ViT: Huge ViT
- **Audio**:
  - WaveNet, VGGish
- **Text**:
  - Word Embedding
    - Glove, Word2Vec, ConceptNet
  - Sentence Embedding
    - BERT, Med-BERT, and variants
  - Language: LLAMA, DeepSeek, Mistral, etc.

- **From SMU PhD Students in my Research Lab:**
  - VGG for transferring to gaze classification
  - VGG for swapped face detection
  - Domain adaptation for speaker authentication
  - Domain adaptation for ASR in children
  - YOLO/DarkNet for surgical instrument detection
  - BERT / RoBERTa for student vocabulary acquisition

Lecture Notes for

Deep Learning

Transfer Learning

Next Time:
Transformers

Reading: None