Machine Learning Review

CS 4277 Deep Learning

Kennesaw State University

What is machine learning?

Definition by Tom Mitchell (1998):

Machine Learning is the study of algorithms that

- improve their performance `P`
- at some task `T`
- with experience `E`.

A well-defined learning task is given by `<P, T, E>`.

Examples of Machine Learning Tasks [1]

Improve on task T, with performance P, given experience E

T: Playing checkers

- P: Percentage of games won against an arbitrary opponent
- E: Playing practice games against itself

T: Recognizing hand-written words

- P: Percentage of words correctly classified
- E: Database of human-labeled images of handwritten words

T: Driving on four-lane highways using vision sensors

- P: Average distance traveled before a human-judged error
- E: A sequence of images and steering commands recorded while observing a human driver.

T: Categorize email messages as spam or legitimate.

- P: Percentage of email messages correctly classified.
- E: Database of emails, some with human-given labels

[1] Slide credit: Ray Mooney

Elements of Machine Learning Problems

Every machine learning problem includes

- data from which to learn,
- a model that takes input data and produces an "answer",
- an error function that quantifies the badness of our model, and
- an algorithm that adjusts the model's parameters to minimize a loss function.
  - A loss function is a surrogate of the error function used by the algorithm. It may be the error function itself, but is often some closely related function with desirable mathematical properties.

Our model, or hypothesis, comes from a model/hypothesis class. Once the parameters are learned, we have an instance of the hypothesis class tuned to our particular machine learning problem.

Kinds of Machine Learning Tasks

- Classification: identify the correct label for an instance
  - Does this image contain a dog/peron on no-fly list/lung tumor?
  - Which radio emitted the signal we received?
  - Will this customer respond to this advertisement?
- Clustering: identify the groups into which instances fall
  - What are the discernible groups of ... customers, cars, colors in an images
  - Is this operating state similar to known operating states, or is it an anomaly?
- Agent behavior
  - Given the state, which action should the agent take to maximize its goal attainment?
- Generation
  - Given a prompt, generate an image/video/poem/love letter.

Kinds of Machine Learning Algorithms

- Supervised
  - Learn from a training set of labeled data – the supervisor
  - Generalize to unseen instances
- Unsupervised
  - Learn from a set of unlabeled data
  - Place an unseen instance into appropriate group
  - Infer rules describing the groups
- Reinforcement learning
  - Learn from a history of trial-and-error exploration
  - Output is a *policy* – a mapping from states to actions (or probability distributions over actions)

Classification using supervised learning methods makes up the lion's share of machine learning.

KENNESAW STATE UNIVERSITY

Supervised Learning Problem Setup

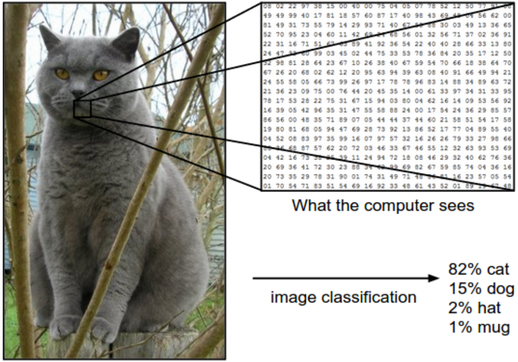
Every machine learning problem contains the following elements:

- An input $\vec{x}$
- An unkown target function $f : \mathcal{X} \to \mathcal{Y}$
- A data set $\mathcal{D}$
- A learning model, which consists of
  - a hypothesis class $\mathcal{H}$, and
  - a learning algorithm.

A learning algorithm uses elements of $\mathcal{D}$ to estimate parameters of of a particular $h(\vec{x})$ from $\mathcal{H}$ which maps every $\vec{x}$ to an element of $\mathcal{Y}$.

Classification Example

Classification is a supervised learning task in which the target function maps feature vectors in $\mathbb{R}^d$ to one of a defined set of classes.



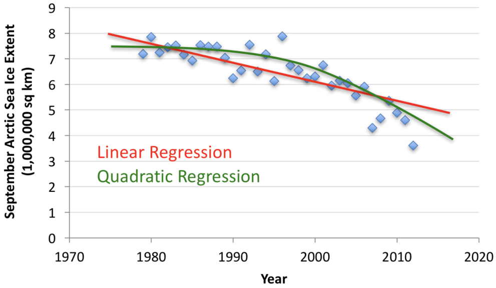
[2]

- In linear classification we assume that there are lines that separates classes acceptably well.
  - Binary: two classes
  - Multiclass: more than two classes

[2] https://cs231n.github.io/classification/

Regression Example

Regression is a kind of supervised learning in which the target function maps feature vectors in $\mathbb{R}^d$ to arbitrary real values.



[3]

- In linear regression we assume that there is a line that fits the data acceptably well.
  - Simple regression: one input variable, e.g., $f(x;\vec{\theta}) = wx + b$, where $\vec{\theta} = (w, b)$
  - Multiple regression: multiple input variables
  - Multivariate regression: multiple output variables

[3] https://www.cc.gatech.edu/~bboots3/CS4641-Fall2018/Lecture3/03_LinearRegression.pdf

Example: Credit Scoring

Let's create a credit score based on two variables: age and income (in thousands), which we'll say are real numbers.

- An input $\vec{x}$ is a vector in $\mathbb{R}^2$. For example, a 25 year-old person making $60,000 would be represented by the vector $(24, 60)$.
- "Credit score" $= \sum_{i=1}^{d} w_i x_i$

The weights w_i represent the importance of corresponding features of input instances.

From that "score" we take a decision:

- Approve credit if $\sum_{i=1}^{d} w_i x_i \geq$ threshold
- Deny credit if $\sum_{i=1}^{d} w_i x_i <$ threshold

KENNESAW STATE UNIVERSITY

Data Sets

Consider a hypothetical data set, $\mathcal{D}$, for the credit scoring problem.

- Each data point represents a previous customer
- Since this is is supervised learning, every data point has an associated label: $+1$ for a customer off whom the bank made money, -1 for a customer off whom the bank lost money

```
   age  income  approve
0   64      90        1
1   78      92        1
2   38      80        1
3   29      66       -1
4   94      79        1
```

Data in this form is often called a *design matrix*, an $N \times D$ matrix in which

- each of the N rows represent an example, and
- each of the D rows represents a *feature* (or *covariate* or *predictor*) of the data examples.

Tabular Data vs. Unstructured Data

The credit data is an example of *tabular* or *structured* data.

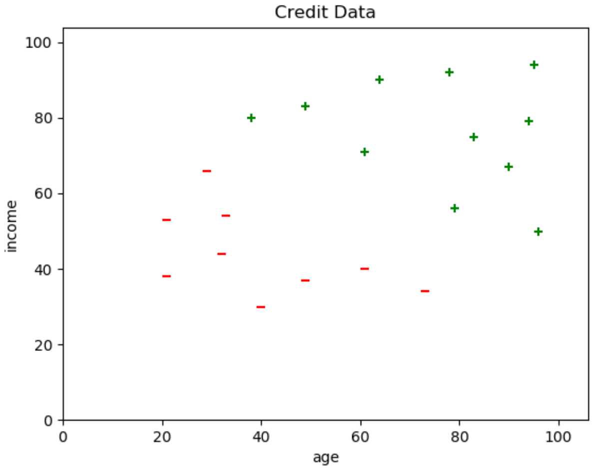
```
    age  income  approve
0    64      90        1
1    78      92        1
2    38      80        1
3    29      66       -1
4    94      79        1
```

We say it is structured because we impose the structure on it. There is nothing inherent in the data that requires `age` to come before`income`, but we must choose some order and stick with it.

Unstructured data is data whose structure is inherent in the data, not imposed by us. Examples include images and natural langauge text.

Credit Data Plot

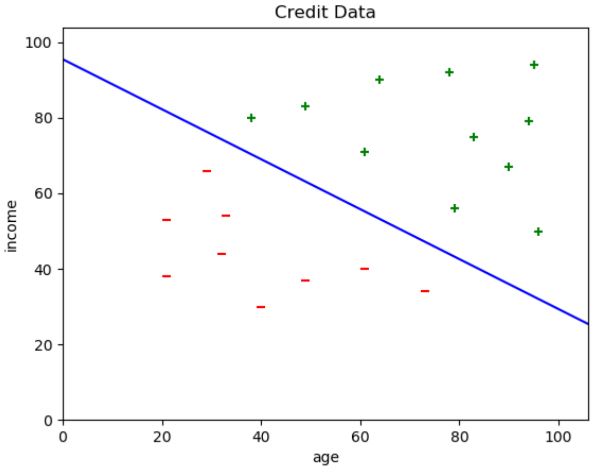
A scatter plot gives us intuition about the structure of the data.



Is there a line that separates the +s from the -s?

A Linear Separator

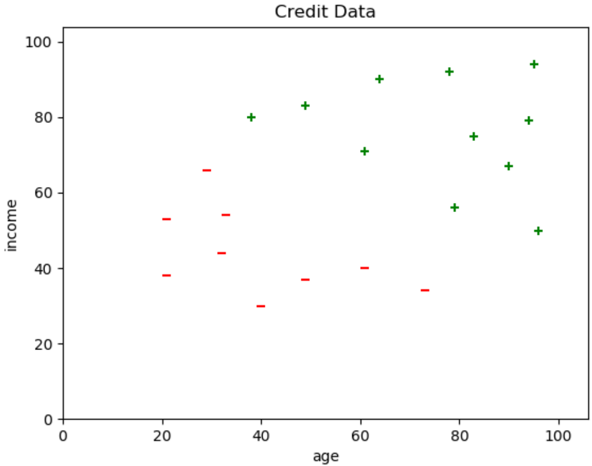
Here's a line that separates the data into classes.



Are there other lines? How many of these lines are there?

Version Spaces

A *version space* is the set of all h in $\mathcal{H}$ consistent with our training data. For our credit data, it's the set of all lines that separate the classes.



Machine learning algorithms find these lines algorithmically.

A Practical Example: Iris Classification

It's a rite of passage to apply supervised learning to the Iris data set. The canonical source for the Iris data set is the UCI Machine Learning Repository. Download iris.data.

Iris Data

The data set contains 150 instance of Iris flowers with

- 4 features:
  - sepal_length
  - sepal_width
  - petal_length
  - petal_width

and

- 3 classes:
  - Iris-setosa
  - Iris-versicolour
  - Iris-virginica

Scikit-learn Recipe

1. Set up feature matrix and target array
2. Separate data into training set and test set
3. Choose (import) model class
4. Set model parameters via arguments to model constructor
5. Fit model to data
6. Apply model to new data

Let's apply this recipe to a data set.

Scikit-learn Data Representation

The basic supervised learning setup in Scikit-learn is:

- Feature Matrix
  - Rows are instances
  - Columns are features
- Target array
  - An array of len(rows) containing the training labels for each instance

We can easily obtain these with a Pandas DataFrame.

Step 1: Iris feature matix and target array

From the description on the Iris Data Set page we know that the Iris instances have four features – (sepal_length, sepal_width, petal_length, petal_width) – and three classes – (Iris-setosa, Iris-versicolour, Iris-virginica). We can read these into a DataFrame with [4]

```
import pandas as pd
iris = pd.read_csv(
    "iris.data",
    names=["sepal_length", "sepal_width", "petal_length", "petal_width", "species"])
iris.head()
   sepal_length sepal_width petal_length petal_width    species
0           5.1         3.5          1.4         0.2  Iris-setosa
1           4.9         3.0          1.4         0.2  Iris-setosa
2           4.7         3.2          1.3         0.2  Iris-setosa
3           4.6         3.1          1.5         0.2  Iris-setosa
4           5.0         3.6          1.4         0.2  Iris-setosa
```

This DataFrame (in tidy format) contains an $N \times D$ design matrix in the first four columns.

[4] You can get this data set through Scikit-learn's datasets module or the ucimlrepo package, but I want to show the use of Pandas for general data manipulation.

Step 1.1: Scikit-learn Input Data

For Scikit-learn we need a feature matrix `X` and target array `y`:

```
X_iris = iris.drop("species", axis=1)
y_iris = iris["species"]
```

We can check that the number of samples in the feature matrix equals the number of labels in the target array with

```
X_iris.shape[0] == y_iris.shape[0] # True
```

There are 150 samples and 150 target labels.

KENNESAW STATE UNIVERSITY

Step 2: Separate Data into Training and Test Sets

We want to separate our data into non-overlapping training and test subsets. Since the data in our data set are arranged in a neat order, we should randomize the samples and split in a way that represents each class equally in the training and test sets. Scikit-learn provides a library functoin to do this:

```
import sklearn
from sklearn.model_selection import train_test_split

X_iris_train, X_iris_test, y_iris_train, y_iris_test = \
    train_test_split(X_iris,
                     y_iris,
                     random_state=1)
```

You will often want to create a third set, a *validation* set, which you use to tune hyperparameters.

> *Set aside your test set at the beginning of the process and don't use it for anything but testing!*

Step 3.1: Choose a model

In your machine learning class you'll learn that no hypothesis class (aka model class, aka hypothesis class, aka algorithm, aka estimator) is best for all data [5]. You must choose your model class based on the data. Things to consider:

- What's the dimensinalty of your data?
- Are your features linearly separable?
- Are your features numeric or categorical?

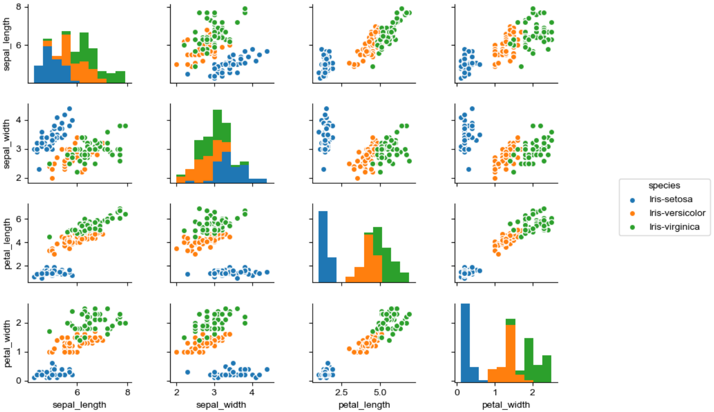
Scikit-learn calls models *estimators*.

[5] Wolpert and Macready, *No Free Lunch Theorems for Optimization*

Step 3: Visualizing the Iris data

You can begin to explore your data with a pairplot:

```
import seaborn as sns
sns.pairplot(X_iris_train, hue="species", size=1.5)
```



These look linearly separable, so we'll try a linear discriminant classifier, SVM.

KENNESAW STATE UNIVERSITY

Step 4: Set algorithm hyperparameters

Hyperparameters are parameters of the algorithm, as opposed to the parameters of the model.

```
from sklearn import svm
model = svm.SVC(kernel="linear")
```

Most parameters are optional, with reasonable default values. Beacuse we know the Iris data set is so well-suited to liner classifiers we choose a `linear` kernel (deafult is `rbf` – radial basis function)

KENNESAW STATE UNIVERSITY

Step 5: Fit model to data

The General Form of Learning Algorithms is:

1. Initialize a model's parameters to some initial values.
2. Until some stopping criterion is reached (e.g., error within bounds)
   - Evaluate the model on some subset of the data $\mathcal{D}$
   - If error is present, update the model's parameters to reduce the error
     - The magnitude of the correction is often captured in a "learning rate" hyperparameter, often represented by η or α

When the algorithm is finished, you have a model, a particular $h \in \mathcal{H}$, that "fits" the training data. In Scikit-learn the learning algorithm is encapsulated in the model's `fit` method:

```
model.fit(X_iris_train, y_iris_train)
```

Step 6: Apply model to new data

To apply the trained model to new (unseen) data, pass an array of instances to predict:

```
y_iris_model = model.predict(X_iris_test)
```

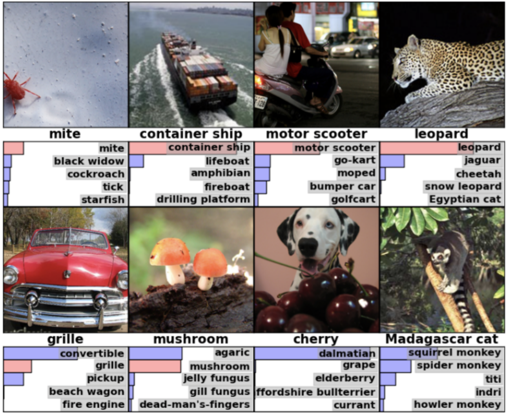
We can test the generalization error (how well the classifier performs on unseen data) using the built-in accuracy score:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_iris_test, y_iris_model)
1.0
```

As you can see, a linear SVM classifier works perfectly on the Iris data. Try out different classifiers to see how well they perform.

A Scikit-learn estimator (model/hypothesis) is an object that has `fit` (train) and `predict` (test) methods.

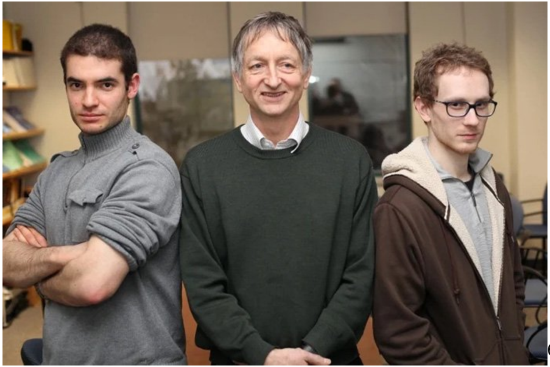
The Deep Learning Revolution



In 2010, Fei-Fei Li and her team published a data set of tens of millions of labeled photographs – ImageNet – and launched the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

In the first two years traditional approaches, characterized by complex feature engineering, continued to win.

AlexNet

[6]

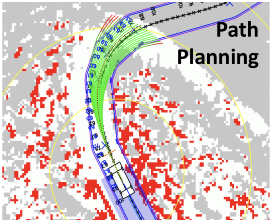
In 2012, Alex Krizhevsky and Ilya Sutskever from Geoff Hinton's lab at the University of Toronto **dominated** the ILSVRC with what is now called AlexNet

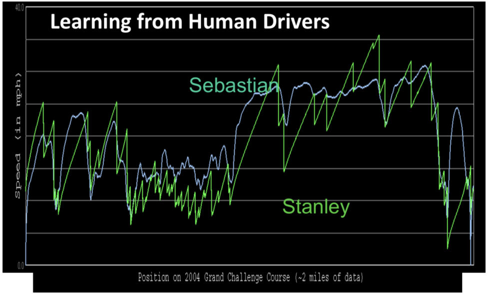
Since then, deep learning has practically taken over AI.

[6] https://web.cs.toronto.edu/news-events/news/congratulations-pour-in-for-geoffrey-hinton-after-nobel-win

Autonomous Cars

Autonomous cars are teeming with deep learning models.





Images and movies taken from Sebastian Thrun's multimedia website.

KENNESAW STATE UNIVERSITY

Scene Labeling

Convolutional deep neural networks (CNNs), or ConvNets, are widely used in vision applications.

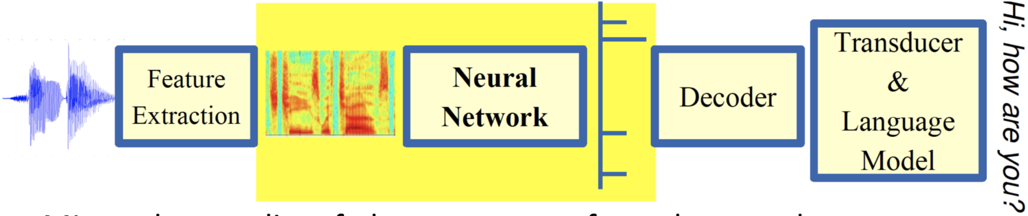
[7]

[7]Farabet et al. ICML 2012, PAMI 2013

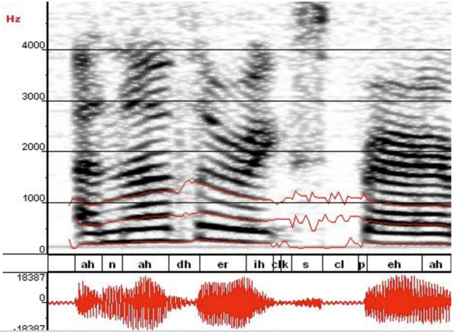
KENNESAW STATE UNIVERSITY

Speech Recognition

One of the early projects that popularized the uses of ReLU activation functions in deep neural networks.



ML used to predict of phoneme states from the sound spectrogram



Deep learning has state-of-the-art results

# Hidden Layers	1	2	4	8	10	12
Word Error Rate %	16.0	12.8	11.4	10.9	11.0	11.1

Baseline GMM performance = 15.4%

[Zeiler et al. "On rectified linear units for speech recognition" ICASSP 2013]

Generative AI

What is "generative AI?"

- Two types of supervised machine learning models: discriminative and generative
  - Discriminative, $p(y|x)$: learn a function that discriminates between classes
  - Generative, $p(x, y)$: learn a joint probability distribution over data
    - Enables generative models to both discriminate and *generate samples*
- Modern GenAI based on deep learning
- Gained attention with Ian Goodfellow's generative adversarial networks (GANs) in 2014, now auto-regressive transformer models all the rage, e.g., large language models (LLMs) like ChatGPT

Thanks to GANs, everyone can dance!

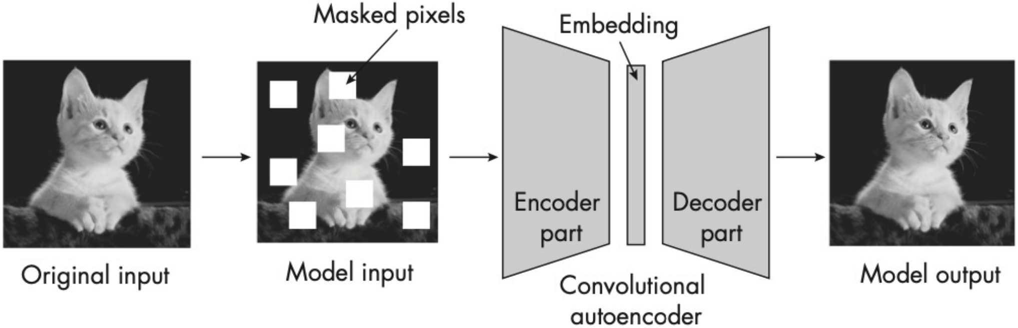

[8]

- https://www.youtube.com/watch?v=PCBTZh41Ris&t=173s

[8] https://arxiv.org/pdf/1808.07371.pdf

You've been using GenAI for years.

E.g., automatically fixing images.



[9]

This is an example of *self-supervised* learning: the data examples themselves are the labels. We remove parts of the examples and train a model to reconstruct the examples.

[9] https://nostarch.com/machine-learning-q-and-ai

KENNESAW STATE UNIVERSITY

You've even been using LLMs for years.

- Modern LLMs based on transformer architecture developed for machine translation.
  - $\{en, fr, de, \ldots\} \rightarrow \{en, fr, de, \ldots\}$
- Google had a chatbot for years but didn't release it publicly because due to a lack of clear business applications.
  - Remember when Blake Lemoine grabbed his 15 minutes of fame?

Figure 1: It's alive!

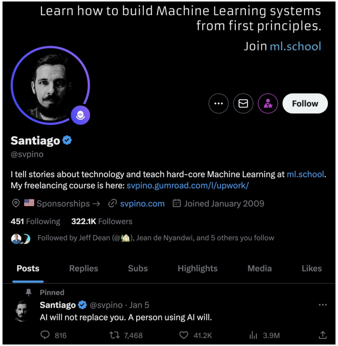
  - The Blake Lemoine kerfuffle once again raised the issue of the Turing Test.

KENNESAW STATE UNIVERSITY

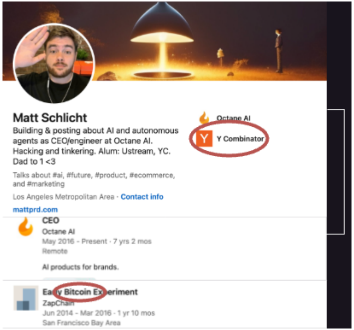
Generative AI Hype Cycle

Unfortunately, the public discourse is filled with noise from various grifters:

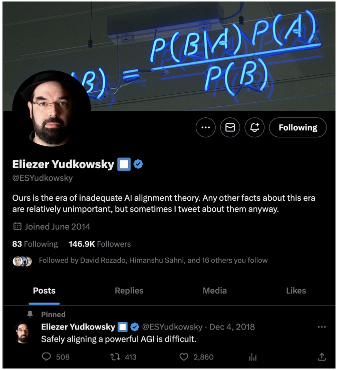
hypers,



charlatans, and



doomers.



But that doesn't mean LLMs are useless.

KENNESAW STATE UNIVERSITY

Whale Detection using Anthropic's Claude Opus

With Claude, the research team achieved:

- 89.4% accuracy in detecting minke whales, compared to 76.5% with traditional methods
- Real-time analysis of data that previously took two weeks to process manually
- Coverage of thousands of kilometers of North American coastline
- Analysis of hundreds of thousands of acoustic recordings annually

Approach:

- Turned whale recordings into visual representations called spectrograms.
- Used both ConvNets and Claude on the spectograms.
- Used *transfer learning* to speed up training and reduce need for training data.
  - Start with foundation model, like Claude/GPT/etc, trained on massive general data set.
  - Fine-tune foundation model by training on domain-specific data.

Impact:

- Create protected zones along major migration routes and breeding grounds
- Redirect shipping traffic away from areas where whales are present
- Pause or modify drilling operations when whales are detected nearby
- Adjust fishing zones and enforce stricter regulations to prevent gear entanglement

https://www.anthropic.com/customers/university-of-sydney

Ethics

- Bias and fairness
- Explainability
- Weaponizing AI
- Concentrating power
- Existential risk

Closing Thoughts

Deep learning dominates AI today.

- That doesn't mean other kinds of AI and ML are irrelevant.
- Deep learning itself was written off decades ago.
- Researchers should learn the right lessons from the story of deep learning.

We'll learn that story, full of winters and springs, in the next lecture.