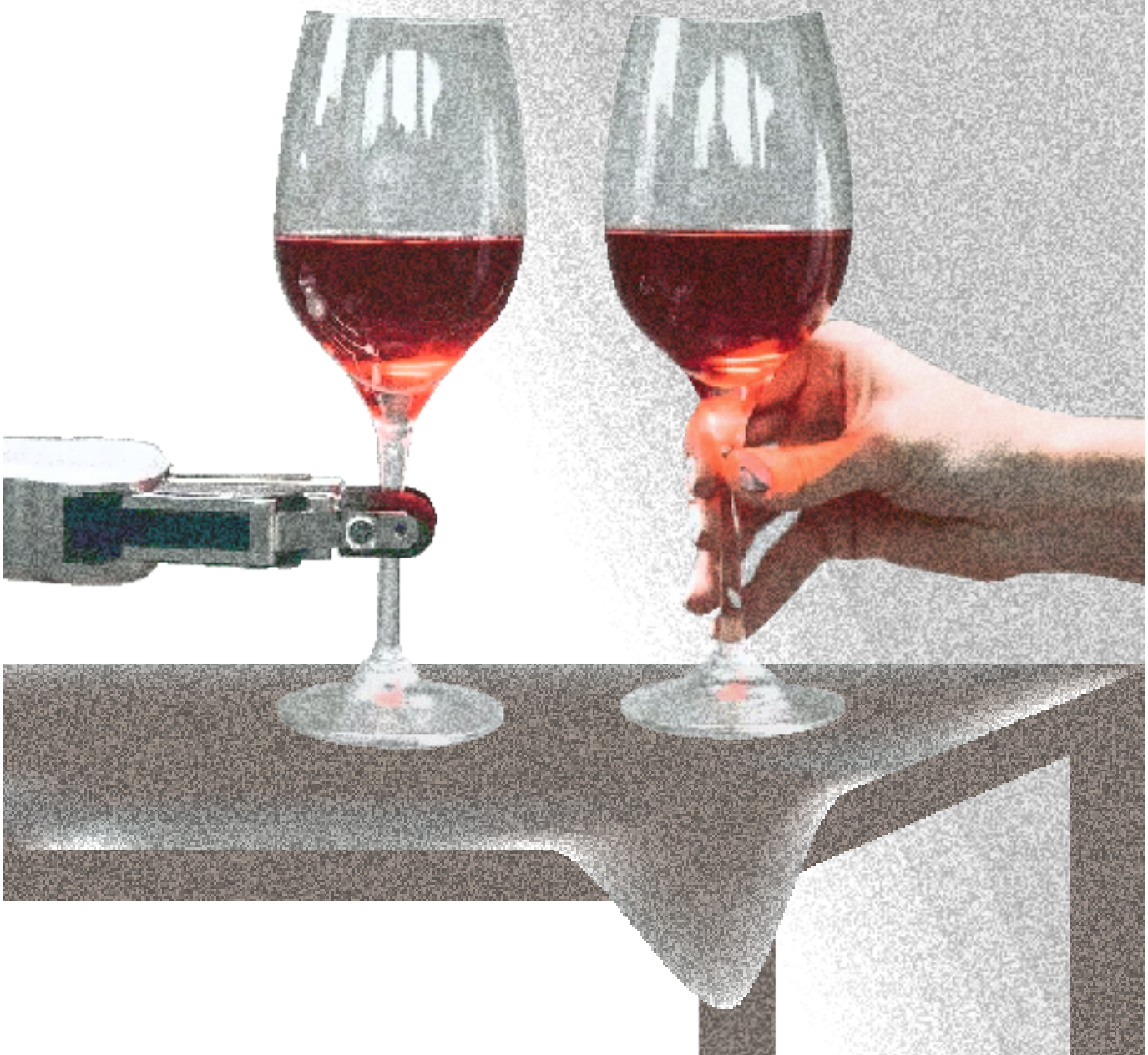
Full Self-Driving, Skynet and other AI myths

Brad Flaugher



Full Self-Driving, Skynet and Other Artificial Intelligence Myths

Modern decision making with deep machine learning models

Brad Flaugher

February 7, 2023

We have now accumulated sufficient evidence to see that whatever language the central nervous system is using, it is characterized by less logical and arithmetical depth than what we are normally used to.

– John von Neumann, 1958 [1]

Preface

This book is an attempt to organize my brain on paper. I have been thinking about decision making with computers, and the computer's relationship to the brain for too long now, and before I get dementia (I'm 35) I would like to put a stake in the ground and tell everyone where they can stick their thoughts about the future.

I am simultaneously very hopeful for the future and horrified by the way that people talk about it. I think many so-called "Data Scientists" are full of shit when they talk, and some of them know it but would like to keep their jobs. I think that many investors are snowed by fancy language and toy demos that they extrapolate into a magical future. Users of digital products are woefully ignorant about how the products work and how they fit into a larger ecosystem. I'm amused, annoyed and would like to clear things up for everyone, including myself.

I am a self-employed programmer and financially secure, these are my credentials. I have no corporate master and I am not raising money to create the future of AI, so have a limited bias towards AI boosterism. I write programs close enough to the bleeding-edge that I can see the present as fast as almost anyone. Because I am simultaneously a contractor (`https://inoxoft.com/`), investor (`https://ventures.nextfab.com/about/`) and teacher (`https://bradflaugher.com/bootcamp.html`) I get the opportunity to see many projects, and I hope I can effectively comment on some common misconceptions and errors in thinking I see in many businesspeople, developers and investors.

So let's go on this journey together, I'll try and explain to you (with words, code and examples) how modern AI works, and when you should be scared (self-driving cars without the appropriate infrastructure) or when you should not be scared (please shut up about Skynet becoming self-aware). I'll also try to do case studies in as many arenas that I think are interesting. If you have notes for me you can reach me at brad@bradflaugher.com.

Please skim this book, please skip around and read some stories. I think the first three chapters are important for everyone to know. The rest of the stories can be chosen a la carte. Enjoy.

Brad Flaugher

PS Thank you Angela Altamirano for the cover wonderful cover art. Thank you to my early supporters, William Dickson, Daniel Hoevel, Patrick Maloney, Jerrod Howlett and Jeff Zinser. A special thank you to Louis Cid and the Harvard Club of New York for giving me an early forum. Finally thank you to Dr. Péter Érdi for putting me under his wing many years ago and Dr. Elisa Esposito for her constant support.

PPS This book uses the wonderful Free and Open Source Kaobook LaTeX project (`https://github.com/fmarotta/kaobook`) and many Free and Open Source AI models for text and image generation like Stable Diffusion (`https://huggingface.co/spaces/stabilityai/stable-diffusion`) and Mann-E (`https://huggingface.co/mann-e/mann-e_4_rev-0-1`) and some closed models like GPT-3, ChatGPT and Dall-E from OpenAI (`https://openai.com/`).

Contents

History: The End of Good Old-Fashioned Artificial Intelligence

1

"Programming will be obsolete. I believe the conventional idea of "writing a program" is headed for extinction, and indeed, for all but very specialized applications, most software, as we know it, will be replaced by AI systems that are trained rather than programmed. In situations where one needs a "simple" program (after all, not everything should require a model of hundreds of billions of parameters running on a cluster of GPUs), those programs will, themselves, be generated by an AI rather than coded by hand." Matt Welsh, 2023 [2]

1.1 The AI Textbook in 1997

Dr. Elaine Rich's textbook on artificial intelligence (AI), published in the 1980s, was a groundbreaking work that helped to establish many of the core concepts and techniques in the field of AI. However, the rapid advancements in AI over the past few decades have led to many of the chapters in this textbook becoming obsolete.

One of the main reasons for this is the prevalence of deep learning, big data, and large-scale statistical models in modern AI. These techniques have largely replaced the symbolic, rule-based approach to AI that was emphasized in the textbook, making many of the chapters on knowledge representation and expert systems less relevant.

Additionally, the explosion of data and the availability of powerful computing resources have made it possible to apply machine learning techniques at a scale that was previously unimaginable. This has led to the development of highly effective machine learning models that can handle complex tasks such as image and speech recognition with a high degree of accuracy, making many of the chapters on simpler machine learning techniques such as decision trees[1] and linear regression less relevant. [3] [2]

We'll discuss this history and a few examples from the "early days" of AI to help us understand where we are headed. We'll start with machine translation, then discuss chess and finally neural networks, which will be the focus of the rest of this book.

1: Although mathematically neural networks are decision trees https://arxiv.org/abs/2210.05189

[3]: Rich et al. (2009), *Artificial Intelligence*

2: the book is now in its third edition and unlikely to be updated as Dr. Rich as retired https://www.cs.utexas.edu/~ear/

1.2 Does AI Need to Know Grammar to Translate?

Noam Chomsky is a linguist and philosopher who has made significant contributions to the field of linguistics with his theory of universal grammar. Chomsky believes that all human languages share a common underlying structure, and that this structure is innate to humans. He proposes that this innate structure is the result of a "language acquisition device" present in the human brain, which allows us to learn and produce language. Chomsky

also argues that the structure of language is largely independent of its content, and that the ability to produce and understand language is a fundamental aspect of human nature. His theory has been influential in the field of linguistics and has sparked much debate and research on the nature of language and its relationship to the human mind.

For English speakers or anyone who has learned English as a second language you'll have many examples of special cases, irregular verbs, bad English and former street slang that became good and proper over time. For programmers this is a nightmare, how can we codify human knowledge in a timely fashion? If we tried to write the rules of the English language in code (which many have tried to do) the rules themselves might change before we were finished writing them.

Explicitly translating languages through code is a difficult task because it requires a thorough understanding of the grammar, vocabulary, and syntax of both languages, as well as the nuances and subtleties of their respective cultures[3]. Simply coding rules for how to translate words or phrases from one language to another is not sufficient, as there are often multiple valid translations for a given phrase depending on the context in which it is used.

3: For programmers this is a nightmare, how can we codify human knowledge in a timely fashion? If we tried to write the rules of the English language in code (which many have tried to do) the rules themselves might change before we were finished writing them.

A more effective approach to translation is to use statistical techniques that rely on a large corpus of translated data, such as Canadian laws[4]. This type of data-driven approach involves training a machine learning model on a large dataset of translations, allowing it to learn the patterns and relationships between the languages. The model can then use this knowledge to make educated translations of new phrases or sentences, taking into account the context in which they are used.

4: They're in French AND English, which is useful data that we can use to correlate phrases and transform English to French and vice-versa.

While this approach is not perfect, it has proven to be highly effective in machine translation and can produce accurate translations even for languages that are very different from each other. The use of a large dataset of translations also allows the model to learn from the mistakes and variations present in real-world translations, further improving its accuracy.

1.3 Explicit Rules and Codified Human Knowledge

When we "teach" a computer to perform a task by explicitly writing down all of the rules of that task, we are really codifying human understanding.[5] When we codify human understanding we write down every rule that we know explicitly. For small tasks we can do this with 100 percent accuracy, and only minor headache on the part of the software developer.

5: Programming this way makes some software development totally boring, I almost switched my major in college to math after considering what a life would look like manually writing rules for handling "edge cases" for the rest of my natural life.

For example, let's write a boring function to tell you the number of days for a given month.

```
def days_in_month(year, month):
  if month in [1, 3, 5, 7, 8, 10, 12]:
    return 31
  elif month in [4, 6, 9, 11]:
    return 30
  elif month == 2:
    if (year % 4 == 0 and year % 100 != 0) or year % 400 == 0:
      return 29
    else:
      return 28
  else:
    return "Invalid month"
```

Writing code can be a tedious and repetitive task, especially when it comes to debugging and testing. It can be especially frustrating when you're working on a large project and you're trying to track down a specific bug that's causing the program to crash. Testing code can also be boring, as it often involves running the same tests over and over again to ensure that the code is working correctly.

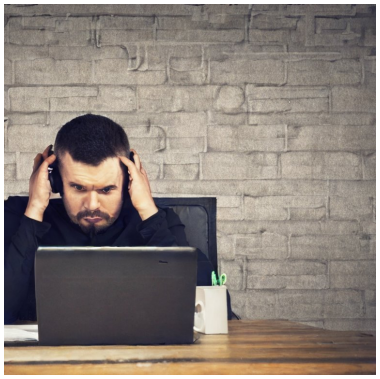


Figure 1.1: "a frustrated programmer writing boring rules on his computer" made with Stable Diffusion 2.1

Additionally, writing code can be boring because it requires a lot of concentration and focus. It can be easy to get lost in the details and lose track of time, especially if you're working on a complex problem. It can also be challenging to come up with creative solutions to problems, and it can be frustrating when your code doesn't work as expected.

While writing and testing code can be rewarding and fulfilling, it can also be a tedious and boring process. It requires a lot of patience, persistence, and attention to detail, and it can be easy to get frustrated and lose motivation. However, with practice and perseverance, it is possible to overcome these challenges and find enjoyment in the process of writing and testing code.

AI has traditionally operated by explicitly codifying human knowledge into machine-readable formats by doing the boring job of coding. This approach, which I'm calling "codified human knowledge" relies on humans to carefully structure and organize information in a way that can be understood by the AI system. The AI system then uses this structured knowledge to make decisions and perform tasks.[6]

6: Some smart people think that AI spells the "End of Programming" https://cacm.acm.org/magazines/2023/1/267976-the-end-of-programming/fulltext, I think the headline is clickbait but the general idea that programming is changing along with advances in AI is true. We are all data scientists now!

However, recent advances in AI have largely ignored the knowledge representation problem and instead have focused on using statistical techniques and neural networks to automatically learn patterns and relationships in data. This approach, known as "deep learning," involves training large neural networks on vast amounts of data, allowing the AI system to make educated classifications and transformations of data without explicit human guidance.

Deep learning has proven to be highly effective in a variety of applications, such as image and speech recognition, and has contributed to the rapid progress we have seen in AI in recent years. However, the reliance on large amounts of data and the lack of transparency in these models can make it difficult to understand how they are making decisions, which can be a concern in certain applications (hence the title of this book).

1.4 IBM Tries Every Possible Chess Move

Deep Blue was a revolutionary computer developed by IBM that was specifically designed to play chess at the highest level. It was programmed with a vast database of chess knowledge and was able to analyze millions of positions per second.

Garry Kasparov was the reigning world chess champion at the time, and he was considered to be one of the greatest players in history. He was beaten by Deep Blue, which used rules based GOFAI (Good Old Fashioned AI) to essentially calculate every possible move and project that move to the end of the game, then choose the best position by brute force. Despite Kasparov's best efforts, he was no match for Deep Blue's computational power. In the end, Deep Blue emerged victorious, defeating Kasparov in a historic match that changed the world of chess forever.

Deep Blue was a turning point in the development of AI, but Deep Blue's methods (namely calculating every possible outcome of a Chess game to determine the best move) was not suitable for many of the world's problems. It turns out that Chess is fun, but the world is not like chess.[7] The "real"[8] future of AI was being developed elsewhere, using statistics and a toy model of the brain to solve a very practical problem for banks.

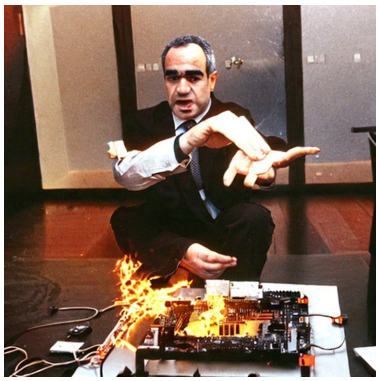


Figure 1.2: "Garry Kasparov setting a computer on fire" made with Stable Diffusion 2.1

7: if you would like to read an example of the simplest chess engine that I could imagine that is written in a similar "codified" way as Deep Blue check out https://github.com/thomasahle/sunfish/blob/master/sunfish.py

8: We might go back, and try again to more explicitly code everything up, and in some cases we still need to, but from the author's perspective, we live in a deep learning/neural network world.

1.5 Statistical Analysis of Handwriting is the Way of the Future

It was the early 1990s and Yann LeCun was a researcher at Bell Labs in New Jersey. At the time, the process of reading and processing checks was a tedious and time-consuming task that was done manually by bank employees. LeCun saw the potential for using artificial intelligence to automate this process, and he began experimenting with using convolutional neural networks (CNNs) to recognize patterns in images of checks.

At the time, CNNs were a relatively new type of neural network that had been developed in the 1980s for image recognition tasks. They were inspired by the structure of the human visual system, and were able to process images in a way that was similar to how the human brain does.

LeCun's work was groundbreaking, and he was able to achieve impressive results using CNNs to process checks. By 1993, he had developed a system that was able to read and process checks with a high degree of accuracy, significantly reducing the amount of time and effort that was required to process checks manually.[9]

LeCun's work on using CNNs for check processing was a major milestone in the field of artificial intelligence, and it laid the foundation for the development of many other applications of CNNs in the years that followed. Today, CNNs are widely used in a variety of applications, including facial recognition, image classification, and natural language processing. [10]

9: CNN digit OCR models are frequently featured in beginner training tutorials for deep learning libraries like PyTorch and Tensorflow, check one out https://github.com/jonkrohn/DLTFpT/blob/master/notebooks/alexnet_in_tensorflow.ipynb

10: check out Yann LeCun domonstrating a convolutional neural network in 1993 at https://www.youtube.com/watch?v=FwFduRA_L6Q.

1.6 Less Programmer Intelligence and More Data Intelligence

I think it's useful to separate the knowledge in the AI problem-space into two groups. The data and the programmer together make the programs that we use every day, and for the rest of this book I'll try and separate the discussion of the smarts of each to help us better understand the world. [11]

Figure 1.3: "a group of computer programmers striking outside of Microsoft's offices with placards saying 'rule-based programming is boring'" made with Dall-E 2

Early AI relied heavily on a human programmer to design, write, and debug computer programs. Good programmers needed domain expertise, problem-solving skills, logical thinking, and the ability to learn and adapt to new programming languages and technologies.

We are now in the age of big data, and everyone knows that "data is gold". Statistical AI methods that are now most prevalent rely on extracting meaningful insights and knowledge from large datasets. This involves using statistical and analytical methods to discover patterns and trends in data, and using this information to inform business decisions or solve problems.

Throughout this book I'll discuss the interaction between programmer and data, and what can go wrong. Working with big data and statistics at a large scale has given AI tremendous ability, but has made understanding and testing models infinitely more difficult. It is your authors belief that understanding the nuance of this interaction between programmer and data is essential to understanding modern AI.

1.7 From Explicit Rules to a Black Box, and Beyond

Artificial intelligence (AI) has come a long way since its inception, and the way that it makes decisions has changed significantly over time. In the early days of AI, explicit rules were used to tell the AI system what to do in certain situations. These rules were often written by humans and encoded into the system, and the AI would follow them to make decisions.

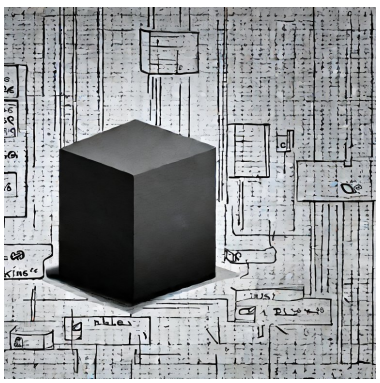


Figure 1.4: "from explicit rules, to a black box and beyond" made with Stable Diffusion 2.1

However, with the advent of deep learning, we have started to rely on a statistical understanding of the truth for AI to make decisions. Deep learning is a type of machine learning that involves training artificial neural networks on large datasets. These neural networks are able to learn patterns and relationships in the data, and can use this knowledge to make decisions.

The use of deep learning has led to the development of powerful AI that is able to perform tasks that were previously thought to be impossible for a machine to handle. For example, deep learning has led to the development of AI systems that are able to recognize faces, translate languages, and even beat humans at complex games like chess and Go.

While deep learning has led to significant advances in AI, it has also made it harder to debug and understand how the AI system is making its decisions. With explicit rules, it was relatively easy to understand why the AI made a particular decision. However, with deep learning, it is often difficult to understand exactly how the AI arrived at its decision. This can make it challenging to troubleshoot problems with the AI system and to ensure that it is making decisions that are fair and unbiased.

AI has come a long way since its early days, and the way that it makes decisions has changed significantly over time. While explicit rules were once used to tell the AI what to do, we now rely on a statistical understanding of the truth for AI to make decisions. This has led to the development of powerful AI that is able to perform a wide range of tasks, but it has also made it harder to debug and understand how the AI is making its decisions.[12]

12: From here on out I'll try and use "GOFAI" (Good Old-Fashioned Artificial Intelligence) to describe the rules-based approach, and use "Deep Learning" or "Machine Learning" to talk about modern neural network and statistics-based techniques

1.8 Key Takeaways

- **AI is a misleading term**, we should instead talk about rules-based programming (GOFAI) and deep learning so we don't confuse ourselves, our partners and our users.[13]
- Good Old-Fashioned AI **(rules-based AI) is hard to create and maintain**. We used to use it for chess engines (like Deep Blue) and translation, but now programmers favor using statistical machine learning techniques.
- Good Old-Fashioned AI is not well suited to many problems, like machine translation and handwriting detection.
- Modern Deep Learning techniques use data to train models instead of humans explicitly writing rules, and the **deep learning models are often as good as the data they are trained with**.[14]

13: Avoid the imprecise words "Algorithms" and "AI" and instead consider the more precise "rules-based AI" AKA "GOFAI" to describe situations where programmers explicitly write rules that they design and "deep learning" or "machine learning" where programmers create models using a dataset and let the machine figure out the rules via a neural network.

14: More *intelligent* data and maybe less intelligent programmers, or maybe programmers who are better trained in statistics and complex systems theory instead of "classical" computer science.

The Regression Theory of Everything

2

"AI Scientists disagree as to whether these language networks possess true knowledge or are just mimicking humans by remembering the statistics of millions of words. I don't believe any kind of deep learning network will achieve the goal of AGI [Artificial General Intelligence] if the network doesn't model the world the way the brain does. Deep learning networks work well, but not because they solved the knowledge representation problem. They work well because they avoided it completely, relying on statistics and lots of data instead. How deep learning networks work is clever, their performance impressive, and they are commercially valuable. I am only pointing out that they don't possess knowledge and, therefore, are not on the path to having the ability of a five-year-old child." Jeff Hawkins, 2022 [4]

2.1 Let's Avoid Knowledge Representation!

The knowledge representation problem in AI is the challenge of how to formally represent knowledge in a way that a computer can understand and reason about. This typically involves creating a set of symbols, rules, and structures that can be used to represent concepts, relationships, and other types of information. The goal is to create a representation that is both expressive enough to capture all relevant aspects of the domain, and computationally tractable enough to allow for efficient reasoning and inference. There are many different approaches to knowledge representation, including logic-based, semantic networks, frames, and ontologies, each with their own strengths and weaknesses.

Deep learning techniques handle knowledge representation differently than traditional symbolic AI methods. Unlike symbolic AI, which relies on explicit and hand-coded representations of knowledge, deep learning techniques learn to represent knowledge implicitly through the use of neural networks.

In deep learning, knowledge is represented in the form of the weights of the neural network. These weights are learned through training on a large dataset and they capture the underlying relationships and patterns in the data. The neural network can then use these learned weights to make predictions, classifications or generate new data.

Deep learning models can handle large and complex datasets, and can automatically extract features from the data without the need for manual feature engineering. This makes them particularly well-suited for tasks such as image and speech recognition, natural language processing, and other areas where large amounts of data are available. However, they are not as good at explicating how they arrived at a decision, which can be a disadvantage.

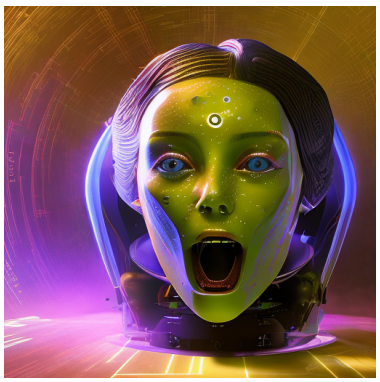


Figure 2.1: "mdjrny-v4 a disembodied computer being force-fed data like a duck looking somewhat like a sci-fi character 8k" made with Mann-E

In summary, deep learning techniques handle knowledge representation by learning the underlying patterns and relationships in the data through the use of neural networks, which can then be used for prediction, classification, and generation tasks. In GOFAI, knowledge is held by the programmer and explicitly coded into rules, deep learning methods instead use data to guess the best rules from the relationships peresent in the dataset.

2.2 A Simple Neural Network is also a Linear Regression

A neural network can be mathematically equivalent to a regression or a decision tree under certain conditions.

A neural network is a machine learning model composed of layers of interconnected artificial neurons, which are designed to process and analyze data. They can be used for a wide range of tasks, such as image and speech recognition, natural language processing, and prediction.

A regression is a statistical method used to predict a continuous variable based on one or more input features. A linear regression, for example, is a simple neural network with one input layer, one output layer and no hidden layers. In this case, the weights of the network are the coefficients of the linear equation and the network is equivalent to a linear regression model.

A decision tree is a tree-based model used for classification and prediction tasks. It consists of a series of if-then rules that are used to make decisions based on the input data. A neural network with one input layer, one output layer and one hidden layer with the ReLU activation [1] function is equivalent to a decision tree. This is because the ReLU activation function allows the network to implement a piecewise linear function which can represent the decision boundaries of a decision tree.

In summary, under certain conditions, a neural network can be mathematically equivalent to a linear regression or a decision tree. These conditions include having one input and one output layers, and having a specific activation function in the case of a decision tree.[5]

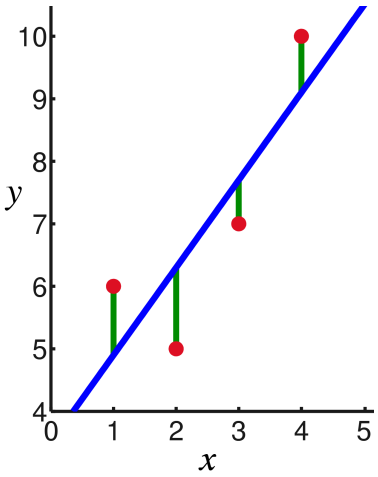


Figure 2.2: A simple linear regression, the red points are the training data, and the blue line is the regression line. If you don't understand this please read https://en.wikipedia.org/wiki/Regression_analysis

1: Neurons in a neural network can have different activation functions, if you don't know what it is and don't want to read about it on wikipedia, that's OK. https://en.wikipedia.org/wiki/Activation_function

[5]: Aytekin (2022), *Neural Networks are Decision Trees*

2.3 Dummy Variables for Dummies (Wonkish)

Chapter Summary: It's all numbers, man. Machine learning techniques require that we turn everything (images, text, sound) into numbers and shove them into the model in the same way we use dummy variables in a simple regression. If you are satisfied with this, please skip this section. If you would like to learn a bit about the details and see some code examples, please keep reading. This section is necessarily technical, but should be approachable for anyone who has taken a college statistics class. [2]

2: Thanks to Paul Krugman for popularizing (to me at least) the *wonkish* classifier, I just mean it's a little nerdy. But it's important.

Dummy variables are used in regression analysis to include categorical variables in a model. Categorical variables are variables that take on a finite number of distinct values, such as "red", "green", "blue" or "yes", "no". Since these variables cannot be directly included in a regression model, as they are not numerical, they need to be transformed into numerical variables.

The process of creating dummy variables is also known as one-hot encoding. It involves creating a new binary variable for each category of the original variable. For example, if you have a categorical variable "color" with three categories: "red", "green", "blue", you would create three binary variables: "color_red", "color_green", "color_blue". Each binary variable would take a value of 1 if the original variable is equal to the category, and 0 otherwise.

When using dummy variables in a regression, it is important to remember to include only n-1 binary variables, where n is the number of categories in the original variable. This is because including all n binary variables would result in perfect multicollinearity, which is when two or more independent variables are perfectly correlated. One of the binary variables can be dropped to avoid this problem.

Dummy variables are used in regression analysis to include categorical variables in a model. The process of creating dummy variables involves creating a new binary variable for each category of the original variable and one-hot encoding it. It is important to remember to include only n-1 binary variables, to avoid perfect multicollinearity.

The creation of dummy variables in a regression is analogous to preprocessing image, text and other data for a neural network for deep learning. This preprocessing is important as it ensures that the data is in a format that can be easily understood and processed by the network. The preprocessing steps for numbers, text, and images are slightly different.

For numbers:

- Normalization: It is common to normalize the input data by scaling it to have a mean of 0 and a standard deviation of 1. This helps to ensure that all input features have similar scales and prevents any one feature from dominating the network's computations.
- Imputation: Handling missing data is important, as it can negatively impact the model's performance. Common imputation techniques include replacing missing values with the mean, median, or mode of the feature.

For text:

- Tokenization: Text data must first be converted into a numerical format that can be understood by the network. This is typically done by tokenizing the text into individual words or n-grams and then encoding them as integers or real-valued vectors. A one-hot encoding exactly like the dummy variable method used in regression is also frequently used. [3] [4]
- Stop-words removal: The most common words in any language like "a", "an", "the", etc. that do not contain much meaning are called

Listing 2.1: Mapping text to numbers.

```
'movies': 99,
'after': 100,
'think': 101,
'characters': 102,
'watch': 103,
'two': 104,
'films': 105,
'character': 106,
'seen': 107,
'many': 108,
'being': 109
```

3: Sometimes text is just mapped to a number! Shocking, but it works. See how it is taught in the TensorFlow tutorials https://www.tensorflow.org/text/guide/word_embeddings

4: GPT-3 uses byte-level Byte Pair Encoding (BPE) tokenization and has a vocabulary size of 50,257.

stop-words, they are often removed to reduce the dimensionality of the data.
- Stemming/Lemmatization: Words that have the same meaning can be stemmed or lemmatized to reduce the vocabulary size and increase the chances of generalization.
- Vocabulary Size: Each model must choose a vocabulary size or the maximum number of tokens that it will analyze. This may cause misspellings, slang or typos to be discarded in analysis.

For images:

- Converting to RGB or Greyscale: Each image is analyzed by its pixel color value, every point on an image will either have 3 color values (red, green, blue) or one single value (on a white/black scale) if the image is analyzed in greyscale.
- Convolutions[5]: Pixel values are analyzed in groups that are defined by the model, since individual pixel values are only colors (or greyness) they must be combined together by the model to detect patterns like faces and stop signs. The method of convolution is defined by the model itself.
- Resizing: neural network can only accept images of a fixed size, so resizing the image to match the network's requirements is important.
- Normalization: It is common to normalize the pixel values to be in the range of 0-1 or -1 to 1. This will help the model converge faster.
- Data Augmentation: To increase the amount of data and prevent overfitting, common data augmentation techniques such as flipping, rotation, and cropping can be applied to the images.

In summary, preprocessing is an important step in training a neural network, as it ensures that the data is in a format that can be easily understood and processed by the network. The preprocessing steps for numbers, text, and images involve normalization, imputation, tokenization, stop-words removal, stemming/lemmatization, resizing and data augmentation.

5: If you would prefer a visualization of a neural network check out this excellent video by Dennis Dmitriev https://www.youtube.com/watch?v=3JQ3hYko51Y.

2.4 Try That Again With 33,640,010 Parameters

In the first section of this chapter we introduced the idea that a simple neural network is mathematically equivalent to a regression. This is true and a useful way to think about neural networks and deep learning. [6]

In practice, neural networks frequently trained with millions of parameters. Here is an example of the trainable parameters of a simple image classifier. This example is a dense neural network to classify handwritten digits, and is not yet as sophisticated as the one used by Yann LeCun and company in the 1990s

In these next few examples, don't worry if you don't understand the layer types or know what batch nomalization means. The point I would like to make is that neural networks are often created with millions of trainable parameters, once you agree with me regarding this point we will discuss the implications of this in more depth.

6:

$$y = Wx + b \tag{2.1}$$

There's your simple formula for a single-cell neural network and regression, in practice we'll change W, x and b to matrices and introduce nonlinear activation function f so,

$$y = f(Wx + b) \tag{2.2}$$

if you please. If equations scare you, don't worry about it for now.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 64)                50240
_________________________________________________________________
batch_normalization (BatchNo (None, 64)                256
_________________________________________________________________
dense_1 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_1 (Batch (None, 64)                256
_________________________________________________________________
dense_2 (Dense)              (None, 64)                4160
_________________________________________________________________
batch_normalization_2 (Batch (None, 64)                256
_________________________________________________________________
dropout (Dropout)            (None, 64)                0
_________________________________________________________________
dense_3 (Dense)              (None, 10)                650
=================================================================
Total params: 59,978
Trainable params: 59,594
Non-trainable params: 384
_________________________________________________________________
```

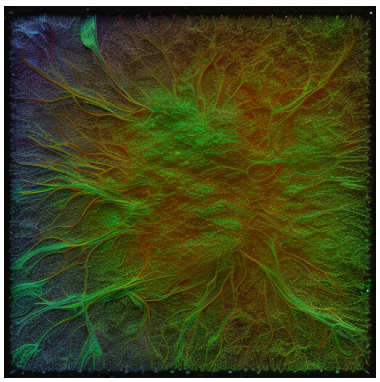


Figure 2.3: "mdjrny-v4 layers and layers of neurons looking like a gooey lasagna, but the data has green blood 8k" made with Mann-E

59,594 parameters! That's not too bad for a simple neural network, but in practice the networks often get even bigger. Here is a *real* example that will make your brain a little hot.

```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 28, 28, 64)        640
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 28, 28, 64)        36928
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 14, 14, 64)        0
_________________________________________________________________
batch_normalization (BatchNo (None, 14, 14, 64)        256
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 14, 14, 128)       73856
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 14, 14, 128)       147584
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 7, 7, 128)         0
_________________________________________________________________
batch_normalization_1 (Batch (None, 7, 7, 128)         512
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 7, 7, 256)         295168
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 7, 7, 256)         590080
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 7, 7, 256)         590080
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 3, 3, 256)         0
_________________________________________________________________
batch_normalization_2 (Batch (None, 3, 3, 256)         1024
```

```
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 3, 3, 512)         1180160
_________________________________________________________________
conv2d_8 (Conv2D)            (None, 3, 3, 512)         2359808
_________________________________________________________________
conv2d_9 (Conv2D)            (None, 3, 3, 512)         2359808
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 2, 2, 512)         0
_________________________________________________________________
batch_normalization_3 (Batch (None, 2, 2, 512)         2048
_________________________________________________________________
conv2d_10 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
conv2d_11 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
conv2d_12 (Conv2D)           (None, 2, 2, 512)         2359808
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 1, 1, 512)         0
_________________________________________________________________
batch_normalization_4 (Batch (None, 1, 1, 512)         2048
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 4096)              2101248
_________________________________________________________________
dropout (Dropout)            (None, 4096)              0
_________________________________________________________________
dense_1 (Dense)              (None, 4096)              16781312
_________________________________________________________________
dropout_1 (Dropout)          (None, 4096)              0
_________________________________________________________________
dense_2 (Dense)              (None, 10)                40970
=================================================================
Total params: 33,642,954
Trainable params: 33,640,010
Non-trainable params: 2,944
_________________________________________________________________
```

When a neural network has millions of trainable parameters and deep layers of neurons[7], it can be difficult to explain to a human what each of those parameters represents or how they contribute to the overall function of the network. This is because the interactions between the different layers and neurons can be complex and non-linear, making it challenging to understand the specific role of each parameter. The parameters model complex interactions between the inputs, making it difficult to understand their specific function. Furthermore, in deep neural networks, the high number of layers can lead to high level of abstraction, meaning that the individual neurons and their weights have little interpretability.

When a neural network has millions of interacting parameters, it can lead to mathematical chaos [8], which is a phenomenon where small changes in the initial conditions of the network can lead to vastly different outputs. This is because the interactions between the large number of parameters can create non-linear relationships that are sensitive to small changes. This can

7: In case you are wondering, the human brain has about 86 billion neurons that do the "thinking". Physical neurons are more complicated than the neurons used in deep learning and have supporting structures like astrocytes that do some computation as well. https://en.wikipedia.org/wiki/Astrocyte

8: Mathematical chaos is a real thing, it refers to the behavior of certain dynamic systems that are highly sensitive to initial conditions. This sensitivity leads to seemingly random and unpredictable behavior, even though the underlying equations governing the system are deterministic (meaning they are transparent and known to us). https://en.wikipedia.org/wiki/Chaos_theory

make it difficult to predict the behavior of the network, as small changes in the input or the parameters can lead to unexpected and seemingly random outputs.[9]

9: Chaotic outputs in a large deterministic system are OK in many domains, and there are controls that can be put in place to keep AI "safe" but deep learning by itself will not produce those controls.

2.5 Multicolinearity and the End of Science

Data science is a horrible term because it implies that data scientists are scientists and that the work they do is scientific when in reality data scientists are not scientists and the work they do is not scientific. Data scientists use mathematics, statistics, and computer science to analyze data, but it is not scientific in the traditional sense. Data scientists do not use the scientific method and do not conduct experiments or develop theories. Data science is more akin to engineering or data analytics than actual scientific research.

The main goal of a scientist is to gain knowledge and understanding of the natural world through research, experimentation, and data analysis. Scientists make models of the world to test and explain. So-called data scientists make models too but a model with layers of interacting parameters, trained under chaotic conditions is almost impossible to explain. Multicolinearity[10] is just one issue, but deep learning techniques are practically incompatible with the scientific method. With deep learning methods it can be difficult to determine which variable had the most impact on the outcome of the model. Coefficients of the model to be unstable and unreliable, which can make it difficult to explain the model in a meaningful way.

10: "Multicolinearity is when two or more variables in a study are closely related to each other. Think of it like dating: imagine you are trying to figure out what makes someone a good partner. You might look at things like how much they make, how attractive they are, and how kind they are. But, these things are all related to each other. For example, attractive people might make more money, and kind people might be more attractive. So, it's hard to know which one is really important for being a good partner, because they are all related. That's like multicolinearity in a study." By ChatGPT given the prompt "explain multicolinearity to a teenager using a dating example"

Explaining the inner-workings of a neural network with millions of interacting parameters is difficult for many reasons. Most of the complexity is due to the number of parameters, which can make it difficult to understand how the model works and why it makes certain decisions. The interactions between the parameters are often non-linear and can be difficult to understand, as the model can be sensitive to small changes in the parameters, which is the definition of mathematical chaos. This can make it difficult to explain why the model is making certain decisions, as the interactions between the parameters are not always clear.

2.6 Let's Test Some Random Inputs! Feature Importance and Explainability

With a huge complex system of neurons that combine inputs in novel ways, it becomes vary hard to understand which inputs are the most important for the system as a whole. To better understand deep learning models, data scientists use randomized or averaged inputs to a model to test the feature importance of a deep learning neural network. This type of testing is done to determine which features are most influential in the overall output of the network. This is done by randomly or averaging the input values for each feature and then running the model to see how the output is affected. For example, if a neural network is used to detect objects in an image, the data

scientist may randomize the color of the objects to see how the model's accuracy is affected. If the accuracy drops significantly, they can infer that color is an important feature in the network.

Randomized or averaged inputs to a model can also be used to determine if a particular feature is necessary for the network to function correctly. For example, if the output of the network is not as accurate when a particular feature is randomized or averaged, then the data scientist can infer that this feature is important for the network's performance. By using this method, data scientists can gain insights into which features are important and which can be removed from the model to improve performance.

2.7 The Universal Machine Learning Workflow

The Universal Machine Learning Workflow[6] is an important chapter in a technical guide for data scientists by the author of the most popular machine learning framework in the world[11] that outlines the *Universal* workflow for machine learning projects. I think this chapter should be understood by everyone using, investing in and creating machine learning models.

[6]: Chollet (2022), *Deep learning with python, second edition*

11: It's called TensorFlow and was one of the most popular and "Most Loved" Technologies of 2022 https://survey.stackoverflow.co/2022/#most-popular-technologies-misc-tech

Before Chollet gets into the details of model building, he chooses to begin with a note on ethics:

"You may sometimes be offered ethically dubious projects, such as "building an AI that rates the trustworthiness of someone from a picture of their face." First of all, the validity of the project is in doubt: it isn't clear why trustworthiness would be reflected on someone's face. Second, such a task opens the door to all kinds of ethical problems. Collecting a dataset for this task would amount to recording the biases and prejudices of the people who label the pictures. The models you would train on such data would merely encode these same biases into a black-box algorithm that would give them a thin veneer of legitimacy. In a largely tech-illiterate society like ours, "the AI algorithm said this person cannot be trusted" strangely appears to carry more weight and objectivity than "John Smith said this person cannot be trusted," despite the former being a learned approximation of the latter. Your model would be laundering and operationalizing at scale the worst aspects of human judgement, with negative effects on the lives of real people.

Figure 2.4: "the face of a trustworthy person" made with Stable Diffusion 2.1. (Hey, it's a white lady!)

Technology is never neutral. If your work has any impact on the world, this impact has a moral direction: technical choices are also ethical choices. Always be deliberate about the values you want your work to support."[6]

Chollet uses the outline below to explain the *Universal* workflow. I'll summarize the workflow and my own notes for a nontechnical audience here:

1. Define the Task
    a) Collect a Dataset [12]
    b) Understand Your Data
    c) Choose a Measure of Success [13]
2. Develop a Model

12: This is often the hardest part. Data is often labeled by hand or stolen from another domain. See https://en.wikipedia.org/wiki/Transfer_learning

13: Accuracy isn't everything! We might prefer a less accurate model that avoids false-positives in some scenarios.

a) Prepare the Data [14]
b) Choose an Evaluation Protocol
c) Beat a Baseline [15]
d) Develop a model that overfits
e) Regularize and Tune Your Model

3. Deploy the Model

a) Explain Your Work to Your Stakeholders and Set Expectations [16]
b) Ship an Inference Model
c) Monitor Your Model in the Wild
d) Maintain Your Model

Of all of the steps in the workflow, "Defining The Task" and "Explaining The Work To Shareholders and Setting Expectations" are where the most miscommunication occurs.

In defining the task, machine learning engineers are often given impossible problems to solve, and because they want to keep paying their mortgage they solve another related problem instead and allow stakeholders to jump to their own conclusions. By clearly understanding what a model is predicting and how the data is collected some of this misunderstanding can be avoided.

I will discuss Large Language Models later, but they all do the same thing. They predict the next words in a sentence, just like the keyboard on your iPhone. They come up with amazing text but by fundamentally understanding what they are predicting you gain insight into their limitations. They are also all trained on the text publicly available on the internet, this includes scientific sources, but also fan-fiction and anime, so the idea that a model trained on this data could be relied on to predict anything truthful is preposterous.

Many models work like magic and many users assume models are predicting the future, when they are really correlating based on their past data. Even simple models of creditworthiness might have as similar problem. While building a creditworthiness model for a bank, due to lack of data a machine learning engineer might [17]create a model that predicts whether someone is a smoker instead of whether they are creditworthy. Because smoking and poverty are correlated, maybe the model "works", but it doesn't do what stakeholders think it does.

Setting expectations is another hellscape of misaligned incentives.[18] Good machine learning engineers and data scientists are supposed to educate and advise stakeholders about the limitations of their own models. Read Chollet's advice on this topic below:

The expectations of non-specialists towards AI systems are often unrealistic. For example, they might expect that the system "understands" its task and is capable of exercising human-like common sense in the context of the task. To address this, you should consider showing some examples of the failure modes of your model (for instance, show what incorrectly classified samples look like, especially those for which the misclassification seems surprising).

14: This is discussed in section 2.3 above, everything gets turned into numbers.

15: Does your model predict better than a totally random model and/or coin-flip?

16: This is almost never done. And as Chollet explains this is both difficult and requires mutual understanding and domain knowledge from the part of "the business" and the Machine Learning Engineer.

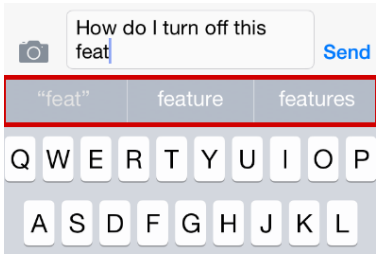


Figure 2.5: Fundamentally LLMs are using the same techniques as a predictive keyboard on your phone. This is why Yann Lecun says even though they are impressive some like ChatGPT are not particularly innovative. https://www.youtube.com/watch?v=ULbpPHjiSBg

17: On purpose or inadvertently.

18: The CEO of OpenAI wrote this and still raised ten billion dollars: *"ChatGPT is incredibly limited, but good enough at some things to create a misleading impression of greatness. it's a mistake to be relying on it for anything important right now. it's a preview of progress; we have lots of work to do on robustness and truthfulness."* https://twitter.com/sama/status/1601731295792414720

They might also expect human-level performance, especially for processes that were previously handled by people. Most machine learning models, because they are (imperfectly) trained to approximate human-generated labels, do not nearly get there. You should clearly convey model performance expectations. Avoid using abstract statements like "The model has 98 percent accuracy" (which most people mentally round up to 100 percent), and prefer talking, for instance, about false negative rates and false positive rates. You could say, "With these settings, the fraud detection model would have a 5 percent false negative rate and a 2.5 percent false positive rate. Every day, an average of 200 valid transactions would be flagged as fraudulent and sent for manual review, and an average of 14 fraudulent transactions would be missed. An average of 266 fraudulent transactions would be correctly caught." Clearly relate the model's performance metrics to business goals.

You should also make sure to discuss with stakeholders the choice of key launch parameters—for instance, the probability threshold at which a transaction should be flagged (different thresholds will produce different false negative and false positive rates). Such decisions involve trade-offs that can only be handled with a deep understanding of the business context.[6]

One does not need to be a psychologist to understand that these conversations almost never happen. The workflow of machine learning is universal, we are all doing fancy regressions and we all deal with the same wild expectations, shitty data and misalignment with business.

Figure 2.6: "mdjrny-v4 a handsome businessperson explaining the business context to a scientist wearing a white coat over coffee 8k" made with Mann-E

2.8 A Machine Learning Engineer is a Data Janitor

Since the machine learning workflow is *universal*, it is fairly straightforward to automate. There are a plethora of offerings from companies big and small who offer AutoML tools that anyone with basic Excel skills can use[19]. Since modern AI is "all a regression" these tools do the regression for you, users just need to feed them the data and the AutoML tool finds out the relationship. On the surface it seems like the job of the machine learning engineer has become obsolete before it even became a proper discipline.

19: See this awesome-AutoML github project for an up-to-date list of commercial AutoML tools `https://github.com/windmaple/awesome-AutoML#commercial-products`

Despite the existence of AutoML tools, machine learning engineers (and data scientists) do some actual work. A 2016[20] Crowd Flower survey of 16,000 data scientists showed that on average we spend our time doing the following:

20: I know this study is old, but I have found that the findings still hold when talking to data scientists and/or machine learning engineers that I train and employ.

- **60 percent of a machine learning engineer's time is spent cleaning and organizing data**, we are data janitors! If you had great data ready in an Excel file you would have been able to fit your own model without us. We need to organize the mess of data that exists in the world so it is in a nice format to be fit in a model. For example it is well known that almost all large language models use the same common crawl dataset (`https://commoncrawl.org/`), but how it is organized and weighted can change the quality of the outputs dramatically. The organization of the same dataset can lead to the same dataset either

producing a nice predictive keyboard or something approaching ChatGPT.

- **19 percent of a machine learning engineer's time is spent collecting data**, we often start making models without a huge dataset, so data needs to be collected in order to make the first model, and after that first model (to match people for our new dating app, let's say) is deployed we rely on users to provide us with data for future models. Collecting new data is a creative endeavour sometimes, and often involves human effort. Amazon's Mechanical Turk[21] service has been leveraged for this purpose for years, and OpenAI has successfully deployed outsourced talent to solve some of the trickiest problems facing large language models[22].
- **9 percent of a machine learning engineer's time is spent fitting models**, we need to fit models, there is some art to choosing the right architecture and modeling techniques. An AutoML tool can do this reasonably well too, but considering some fancy training servers cost $28 per hour or more[23], sometimes it is useful to have an expert guess the best model architecture and save time in training, even if that expert is getting paid $249,000 yer year[24].
- **4 percent of a machine learning engineer's time is spent refining algorithms** many machine learning engineers spend almost no time doing this. Other researchers (corporate-academic types who write scientific papers) spend a lot of time doing this, it averages out to four percent, but doesn't represent the average day of a machine learning engineer.

Overall, the main job of most machine learning engineers is to clean up and collect a ton of data, and then feed that data into the same machine learning model that everyone else is using, and then rinse and repeat.[25]

21: "Getting the right training data was another challenge. ImageNet was a collection of one hundred thousand labeled images that required significant human effort to generate, mostly by grad students and Amazon Mechanical Turk workers." https://arstechnica.com

22: Read more about how OpenAI used Kenyan workers to help label toxic content https://time.com/6247678/openai-chatgpt-kenya-workers/.

23: See the latest pricing from AWS here https://aws.amazon.com/sagemaker/pricing/

24: See the latest machine learning engineer salaries at levels.fyi https://www.levels.fyi/t/software-engineer/focus/ml-ai

25: Here is a discussion on reddit asking "When was the last time you wrote a custom neural net?" and supports my understanding of the world https://www.reddit.com/r/MachineLearning/comments/yto34q/d_when_was_the_last_time_you_wrote_a_custom/

2.9 Key Takeaways

- **Deep learning models are fundamentally large unscientific regressions** they are trained to create a function that maps input data to output data.
- **Deep learning models are chaotic systems containing millions of interacting parameters** they are not designed to be explained or created in a way that their weights can be used for scientific analysis. They find reasonable answers and don't care how they get there. Multicolinearity (understanding the relationship of an input and output) and feature importance (understanding which inputs are most important) are only understandable with a high level of statistical error.
- **Small changes in inputs of a deep learning model may dramatically change the outputs** deep learning models are complex deterministic systems that can exhibit chaotic behavior. Their inner workings are functionally unknowable and practically impossible to test.
- **Machine learning engineers spend most of their time collecting and organizing data**, because deep learning models often share

common architecture, getting good data to train with is the best thing that an engineer can do to train good models. In practice only a handful of corporate-academic types are experimenting with new and exciting architectures.

- **Deep learning models can have impressive and useful outputs, but the creators of models should be encouraged to highlight their failures and limitations.** Machine learning engineers might be more keen to highlight failures and limitations if they are encouraged to do so by their users, managers and investors.

Creativity and Decision Making with Deep Learning Models

3

"AI Policy

I expect you to use AI (ChatGPT and image generation tools, at a minimum), in this class. In fact, some assignments will require it. Learning to use AI is an emerging skill, and I provide tutorials in Canvas about how to use them. I am happy to meet and help with these tools during office hours or after class

Beware of the limits of ChatGPT:

- *If you provide minimum effort prompts, you will get low quality results. You will need to refine you prompts to get good outcomes. This will take work.*
- *Don't trust anything it says. If it gives you a number or a fact, assume it is wrong unless you either know the answer or can check in with another source. You will be responsible for any errors or omissions provided by the tool. It works best for topics you understand.*
- *AI is a tool, but one that you need to acknowledge using. Please include a paragraph at the end of any assignment that uses AI explaining what you used the AI for and what prompts you used to get the results. Failure to do so is in violation of academic honesty policies*
- *Be thoughtful about when this tool is useful. Don't use it if it isn't appropriate for the case or circumstance."*

Dr. Ethan Mollick, 2023 - Syllabus for class at The Wharton School at the University of Pennsylvania

3.1 Theories of Creativity

Machine learning models use data (from the past) to discover rules and make classifications. Because of the way they are constructed these classifications, suggestions, artworks are by definition derivative or "having parts that originate from another source". I won't get into a philosophical discussion on what the nature of creativity is, but it's worth considering how using deep learning AI models biases us towards the past, but also could give us insights from other domains.

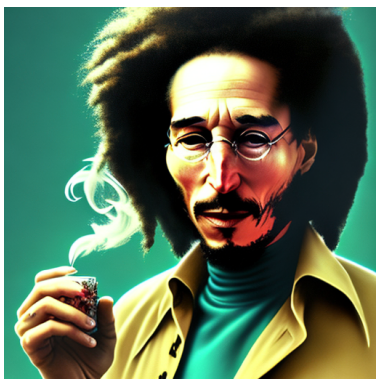


Figure 3.1: "mdjrny-v4 Steve Jobs Smoking weed with Bob Marley 8k" made with Mann-E. It's 100% derivative, but it's art too (I guess).

It can be argued that creativity is simply the combination of existing works. This is because many new ideas and innovations are often inspired by and built upon existing concepts. For example, a new form of music may be created by combining elements from different genres. Similarly, a new technology may be created by combining and improving upon existing technologies.

It can also be argued that creativity involves much more than just combining existing works. Creativity is not just about recombining existing ideas, but also about coming up with completely new and original concepts. This

requires a unique perspective and a deep understanding of the subject matter, as well as the ability to think outside the box.

AI models of speech, when heavily used, may slow down the evolution of language. AI art models may slow down "progress" in art, whatever that means. AI models of disease trained on data from 1980 may be irrelevant to today's diseases[1]. That said these same models may give us interesting insights in new domains, models trained on beautiful paintings may be put to use in a new domain (like desiging beautiful interiors) and that model could give new insight to interior designers, models of the interaction of ants could be put to use in designing cities and so on and so forth. AI cuts many ways, it makes us faster but makes us more reliant on the past, models can be used across domains but should be used intentionally and transparently when possible. Each use opens up a new world of possibilites for users, and sometimes a new headache for intellectual property lawyers. We'll discuss all of these topics in this chapter.

1: We'll discuss "concept drift" later, don't worry.

3.2 Creative Uses of Power Tools

Power tools, even simple ones like a drill[2] are complex machines that take human input and transform it using static rules. The pressure the user of a drill puts on the bit and the speed at which they pull the trigger all deterministically affect the output. Just because a tool is a deterministic machine doesn't mean it is unable to produce creatve works.

2: something like these `https://www.dewalt.com/products/power-tools/drills`

What is happening as we use generative tools like ChatGPT or image-to-textmodels is that the "creative act" has been relocated. The creative act is now the prompt you give the tool, the users input. And someone can still be good at using AI, just like someone can be good at using any piece of software.

Modern AI is a complex system of algorithms, data, and analytics that can be used to solve complex problems. AI systems can learn from data, identify patterns, and make predictions about the future. AI systems are typically used to automate and assist human decision making. AI systems are programmed with specific objectives and goals, and the user input decides the output. For example, an AI system could be programmed to solve a mathematical problem and the user input would determine the parameters of the problem and the output would be the solution.

Figure 3.2: "mdjrny-v4 a mikita drill being used in an artist's studio to make a colorful artwork 8k" made with Mann-E

A power drill is also a complex deterministic system. The user input is limited to the type of drill bit, the speed of the drill, and the direction of the drill. The output is determined by these inputs, as the drill will only drill in the direction and speed determined by the user. The user also has to choose the correct drill bit to ensure the drill can do the job correctly and safely.

Overall, both modern AI and a power drill are complex but ultimately deterministic systems. The user input, however limited it may be, ultimately controls the output of the system. Both systems require a user to understand how to use them and to make the correct input to get the desired output.

3.3 Garbage In, Garbage Out

The "Power Tool" of AI is a deterministic [3] system and the rules of that system are determined by the data that the model is shown. If a model is trained on shitty[4] data, it will produce shitty results. End of book...

... maybe not. It's worth thinking about this for a moment. It is often said that modern AI can "generalize and make informed inferences given new data". If deep learning models are really just a big regression, these models will always come up with an answer, but if the world changes, these models will still be projecting complicated averages of their past data into the future.[5]

So, let's separate AI's decision making into two extremes; *Creative Decision Making* and *Critical Decision Making*. The stakes are very low in a world of *Creative Decision Making* and who gives a shit if the AI is all a regression, and it just mushes together the limited data that it's seen. In a creative context you can also ask an AI interesting questions, so long as you don't soley rely on it's output without checking the facts first[6]. Even if "Garbage In, Garbage Out" holds, garbage can still be helpful for a creative process.

Note that this book is called "Full-Self Driving, Skynet..", a self-driving car and a nuclear-bomb-equipped all-seeing AI are clearly not engaging in any *Creative Decision Making*. I'll spend lot's of time diving into this but this distintion is super important.

Modern AI, particularly deep learning models, will never be reliable enough to make critical decisions by themselves. These models will transform the nature of work and jobs and do a lot of things, but because of the nature of how these models are created, they should **never** be relied on to make critical decisions by themselves. If you think this point is obvoius and pedantic, just stop reading I guess. But the general public seems to be confused about this point, so I'm going to discuss it in depth for many chapters. I'll say here that there are also many ways to reign in deep learning models and allow them to participate in critical decision making without having the final say, most of them involve humans getting a vote, others involve putting the deep learning model 'on rails' and either programming explicit rules via GOFAI or by physical systems that limit the deep learning model's decision making capacity[7].

3: and static or unchanging

4: scientific term here

5: I'm going to talk about concept drift in a few paragraphs, I promise

6: see the syllabus note at the beginning of this chapter

3.4 Garbage In, New Perspective Out?

For creative tasks, it generally doesn't matter that a deep learning model is unscientific or trained on a lopsided dataset. An informed user of AI knows this and can account for that in their decision making, especially when engaging in creative decision making. The situation becomes problematic once we allow deep learning models to engage in critical decision making by themselves.

Generative models will come up with amazing (but derivative) works of art, But, they will never "change the game". A model of sculpture trained

Figure 3.3: Marcel Duchamp's "Fountain". A urinal that blew peoples minds https://www.tate.org.uk/art/artworks/duchamp-fountain-t07573.

on past sculptures in 1917 will never come up with Marcel Duchamp's "Fountain". There is a term that machine learning engineers use for this, when the rules of the game are changed, it's called "Concept Drift".

3.5 Concept Drift and the End of Usefulness

Concept drift refers to the phenomenon where the distribution of data changes over time, causing the performance of AI models built on historical data to degrade. The model becomes less useful because it is trained to make predictions based on the relationship between the inputs and outputs in the data it was trained on, and if this relationship changes, the model may start making incorrect predictions. This is particularly problematic in real-world applications, where the data is constantly evolving and the relationship between inputs and outputs is subject to change. To mitigate the effects of concept drift, it is often necessary to continually retrain AI models on updated data.

Figure 3.4: "Plastic surgery gone wrong" made with Stable Diffusion. Imagine a model that classifies images as "human face" or "not human face", and imagine that model was trained on images of human faces before 1900, maybe you would not be surprised if you gave it a picture of a human face that had a lot of cosmetic surgery done to it, and that model might say "this is not a human face", the idea of what a human face is has changed over time this is called "concept drift".

The frequency with which an AI model needs to be retrained to mitigate the effects of concept drift depends on several factors, including the rate at which the data distribution changes, the size and complexity of the model, and the availability of computational resources.

For some applications with relatively stable data distributions, retraining the model once every few months or even once per year may be sufficient. However, in other applications where the data is changing rapidly, it may be necessary to retrain the model more frequently, such as once per week or even once per day.

Ultimately, the frequency of retraining will depend on the specific requirements of the application, and the trade-off between the cost of retraining and the potential cost of incorrect predictions. In general, it's recommended to monitor the performance of the model over time and to retrain it as needed to ensure that it remains accurate and relevant.

As an example, think about facial recognition models. Facial recogniton models can recognize faces they've seen before, and to have a "perfect" facial recognition model for everyone in the world, you would need to retrain the model on every new face as fast as people were born, or as fast as peoples faces change due to aging or horrific motorcycle accidents. The rate at which you need to retrain the model to maintain it's accuracy depends on how fast new faces are added to the population.[8]

8: This "facial recognition" model is different than the plastic surgery example above, this model is the model that takes a face and says "this is a picture of Brad Pitt" or "this is a picture of Joe Biden" but if you want the model to "recognize" every one in the world, it needs to see everyones face at least once. That is my pedantic but important point I'm making here.

If a deployed AI model is not monitored, there are several risks that can emerge:

- Accuracy degradation: As the data distribution changes over time, the model may become less accurate, leading to incorrect predictions. This can result in financial losses, reduced customer satisfaction, or even harm to individuals.

- Bias amplification: AI models can be biased, and if this bias is not monitored and addressed, it can be amplified over time as the model continues to make incorrect predictions. This can result in discriminatory outcomes, such as unequal treatment of individuals based on protected characteristics such as race, gender, or age.
- Legal liability: In some cases, incorrect predictions made by AI models can result in legal liability, particularly if the model is being used to make decisions that have significant consequences, such as in the criminal justice system or in medical diagnosis.
- Reputational damage: If an AI model is making incorrect predictions, it can damage the reputation of the organization deploying the model, potentially leading to a loss of customers or investors.

Therefore, it is important to monitor AI models once they are deployed, and to take action to address any issues that arise, such as retraining the model or adjusting its parameters, in order to mitigate these risks and ensure that the model continues to perform well over time.

3.6 The Impossibility of Fairness

Achieving fairness in AI models can be challenging, and to some extent it may be impossible to completely eliminate all forms of bias. This is because AI models are trained on historical data, which may contain biases and disparities that are perpetuated in the model's predictions. One can attempt to "fix the training set"but in practice models will continue to bias their prediction to past data, or their creators careful curation of the past.[7]

[7]: Christian (2020), *The Alignment Problem: Machine Learning and Human Values*

However, it is possible to reduce the impact of bias in AI models through careful design and monitoring of the model's performance. This may include techniques such as fairness constraints, algorithmic transparency, and regular auditing of the model's predictions to identify and address any issues of bias.

It's important to note that fairness is a complex and multi-faceted concept, and different definitions of fairness may be appropriate for different applications. For example, some definitions of fairness may prioritize equal treatment of all individuals, while others may prioritize proportional representation or equal opportunities.

Ultimately, the extent to which fairness is achievable in AI models will depend on the specific requirements of the application and the level of effort that is put into designing and monitoring the model to ensure that it is making fair and unbiased predictions.

3.7 Transfer Learning Everywhere

Transfer learning is a machine learning technique that involves transferring knowledge from one model trained on a task to another model trained on a related task. The idea behind transfer learning is that a model that has

been trained on one task can be fine-tuned for another task, reducing the amount of labeled data required to train the new model.

For example, imagine that you have a large convolutional neural network (CNN) that has been trained to recognize objects in natural images. You can use the knowledge learned by this model as a starting point to train a new model that recognizes objects in medical images, such as X-rays or MRI scans. The new model can start with the weights of the pre-trained CNN and fine-tune them on the new task, using a much smaller amount of labeled data than would be required to train the model from scratch.[9]

9: Transfer learning is taking data from a different domain, and pointing the model at a new task where data is hard to come by. Think if you tried to "catch bigfoot" using AI, since you have no data on bigfoot you would maybe use a human face detector and stick in on cameras in the woods to try and find a human-like face out there. It could work wonderfully and you could catch bigfoot, or it could turn out that you made a logical leap and bigfoots face doesn't look anything like a humans, so your use of human data to catch bigfoot was incorrect.

Transfer learning can be useful in many different applications, particularly when labeled data is scarce or expensive to obtain. By leveraging knowledge from a pre-trained model, transfer learning can help to improve the performance of new models, reduce the amount of data required for training, and accelerate the development of new machine learning applications.

If the past is not like the future, you are doing transfer learning. Most models steal data from other sources so are doing transfer learning too. This is fine, but do we know that we are doing this? A model that is deployed in a domain experiencing concept drift and continues to make predictions without being retrained can be considered to be doing a harmful form of transfer learning. This is because the model has been trained on a different distribution of data (the past), and is being applied to a new domain with a different distribution (the present and future).

In traditional transfer learning, the goal is to transfer knowledge from one domain to another related domain, where the data distributions are similar enough to enable the model to make accurate predictions. However, in the case of concept drift, the data distributions are changing over time, and the model is becoming less accurate as a result. [10]

10: Think about your life and if you "transferred" your model of thinking from when you were 12 years old, to "today" when you are 35 years old, that is a bad model to be operating on! That model needs to be updated. We'll explore many examples of where this is and is not a problem in the next chapter."

By continuing to make predictions without being retrained, the model is essentially "transferring" its knowledge from a historical data distribution to a new, changed data distribution, which may not be a valid assumption. This can result in incorrect predictions and other negative outcomes, such as harm to individuals or organizations.

3.8 Industrial-Scale Plagarism

Aside from regurgitating the past and predictions from other domains, deep learning models can also enable plagarism on an industrial scale. Early text generation models could be made to write entire sections of *Harry Potter* when fed the opening lines of a chapter. Even as models grow large and more sophisticated, users, researchers and lawyers are still able to "extract the training data" from large models[8], causing headaches for their creators and adding to the work of intellectual property lawyers.

[8]: Carlini et al. (2023), *Extracting Training Data from Diffusion Models*

To avoid this creators of deep learning models have the following tools at their disposal:

- Best practices in machine learning: data preprocessing, augmentation, reuglarization, model architeture choices, early stopping and validation are all things that we are taught to do to prevent overfitting.
- Contractual agreements: Microsoft (owner of github and cocreator of the Github Copilot code generation model) has a special contract for anyone submitting code to github that essentially says that "we are allowed to use this code to train our models, and if our models regenerate your copyrighted code, you can go fuck yourself"
- Release the kraken: StabilityAI's Stable Diffusion model was released as open source software, they still managed to get sued, but they basically said, "this model was trained on all copyrighted and copylefted images on the web, sometimes it'll generate stuff that violates copyright law, but we are in Germany and will give this thing away for free to the public, and see how the photographers and graphic designers of the world handle it... OKbye!".

Figure 3.5: "Harry Potter and Batman High-fiving" made with Stable Diffusion. I don't think Warner Brothers likes this...

These techniques can help to reduce the risk of having AI models reproduce the training data and cause intellectual property disputes, but there is no way to completely eliminate these risks, they are simply side-effects of the method of machine learning that we are doing nowadays. If you are an IP lawyer and need an expert witness, I'm your man brad@bradflaugher.com.

3.9 Humans Love Computers

It turns out the most potent combination is a smart AI paired with a smart person. [11]

Once upon a time, Garry Kasparov, a legendary chess player, organized a tournament where humans and AI could compete together. The tournament was designed to showcase the strengths of both humans and AI, and how they could work together to achieve incredible results.

The tournament consisted of several rounds, with each round featuring a human player paired with an AI chess program. The human player would make the first move, and then the AI program would make its move, with the pair working together to win the game.

At the start of the tournament, the human players were skeptical of their AI partners, thinking that the machines would just take over and make all the decisions. But as the tournament progressed, they quickly realized that their AI partners were able to provide valuable insights and help them make better moves.

One team in particular, made up of a seasoned chess player and a cutting-edge AI program, stood out from the rest. The combination of the player's experience and the AI's computational power allowed them to anticipate their opponents' moves and make strategic decisions that no human could have made on their own.

11: Funnily enough at Wharton I recently took a class where Dr. Cade Massey said basically "If you want to bring people along and get them to use your model, let them play with the ouput a bit, allowing for even the slightest bit (2%) adjustment in the outputs will get users to adopt your model much faster

In the final round, the team faced off against the reigning champions, a pair of world-class chess players. The game was intense, with both sides making bold moves and calculating intricate strategies. In the end, the human-AI team emerged victorious, much to the surprise of the crowd.

The tournament was a huge success, and showed that when humans and AI work together, they can achieve amazing things. The tournament participants learned that AI is not just a tool, but a valuable partner, and that by combining their strengths, they could achieve results that neither could have accomplished on their own.

The tournament inspired many people to explore the potential of human-AI collaboration, and showed that by embracing technology, we can create a brighter future for all.

AI is an amazing partner, but we need to think for ourselves too. We can't blindly trust AI, but we can use it to inspire and challenge us.[9]

[9]: Mansharamani (2020), *Think for yourself: Restoring common sense in an age of experts and artificial intelligence*

3.10 Key Takeaways

- **Models can puke out their training data** cool!

Model Cards and Case Studies 4

modify huggingface model cards https://huggingface.co/docs/hub/main/model-cards

add a model of physical systems early, something like fluid dynamics

4.1 Lightning Round of Model Analysis

1. **A text generation model trained on Shakespeare's plays** This could be used to spruce up dialoge for a screenplay, write a book, or write a fancy-sounding email. It should be obvious to the user what it is doing, and this would probably delight users and not cause many problems.
2. **A classifer to suggest promising lithium mining sites**
3. **A generative model to discover drugs that might be useful in preventing liver disease**
4. **A in image classifier for a dating app, trained on images of famous couples** Someone wants to start their own dating app, but they don't have any of their own data. So they base their "compatability score" off of photos of potential couple and classify them as either "looking like a celebrity couple, and therefore compatable" or "not looking like a celebrity couple, and therefore incompatable". This model sounds funny and interesting, but could certainly change lives. I still would say this is a creative use of a model, and probably fine.
5. **A hate speech classifier for a social media site**
6. **A personal danger classifier for your smartwatch** see instapaper
7. **A threat classifier traned on images of war** Wars might change from place to place, if the model was trained in the middle east it might not be as useful in Ukraine, but maybe it could be helpful to augment a security guard looking over thousands of cameras at Grand Central Station. It is not clear to me whether classifying threats falls under creative or critical decision making. Is this a useful classifier for an understaffed security unit, or a bias-reinforcing tool for police abuse? I'm not sure, but I'm sure this model is out there already.[1]
8. **A text generation model claiming to be Artificial General Intelligence**[2] A model using deep learning techniques, and trained on the entire internet, but claiming to have general purpose answers to everything[3]. This could absolutely be used for creative endeavors and generate new perspectives but if it is given critical tasks and relied on to "tell the truth", this will probably disappoint. Creating the model isn't a problem in itself, but how it is used or misused might cause a problem.[4]
9. **A fully self-driving car, one without a steering wheel** The car was trained on millions of miles of road-trials, and maybe a sophisticated simulation environment where it drove trillions of virtual miles.[5]

1: if you would like a deeper discussion of ethics please read Reid Blackman's "Ethical Machines"[10], he is pretty good for a Philosophy PhD.

2: Let's say something like ChatGPT

3: ChatGPT does not make these claims

4: What problems, you ask? Well, give me a few chapters.

5: Let's assume the model is only updated once per month here, I'll talk about this in more depth soon.

This model is not giving you a new perspective or funny perspective on driving because all of the training happened in the past. So, as soon as the self-driving model is released, it becomes stale. Any significant changes in the envoronment might cause it to behave strangely and dangerously.

Self-Driving With Statistics | 5

- *First Law: A robot may not injure a human being or, through inaction, allow a human being to come to harm.*
- *Second Law: A robot must obey the orders given it by human beings except where such orders would conflict with the First Law.*
- *Third Law: A robot must protect its own existence as long as such protection does not conflict with the First or Second Law.*

Asimov's Three Laws of Robotics

[11] [12] [13] [5]

5.1 Semiautonomy is Stupid

Talk about the levels 0-4, deep learning is a subsystem of these levels.

Semiautonomy is dumb because it feels autonomous, but the user is expected to take over at any time. This is the worst of both worlds. [14]

and this article Tesla Crash Illustrates Problem With Semi-Automated Driving and comments

5.2 Trolley Problems

5.3 Concept Drift (Reprise)

5.4 Multicolinearity (Reprise)

[1]

1: You can't get your self-driving car dirty either, it'll mess up those sensors.

5.5 See You In Court, Asshole!

5.6 A Train is a Self-Driving Car, Right?

Discuss how the problem space has changed in warehousing, we don't actually have self-driving forklifts, we have moving shelves.

Maybe talk about elon musk and manufacturing.

Unplugging Skynet 6

"The second requirement of goal-misalignment risk is that an intelligent machine can commandeer the Earth's resources to pursue its goals, or in other ways prevent us from stopping it... We have similar concerns with humans. This is why no single person or entity can control the entire internet and why we require multiple people to launch a nuclear missile. Intelligent machines will not develop misaligned goals unless we go to great lengths to endow them with that ability. Even if they did, no machine can commandeer the world's resources unless we let it. We don't let a single human, or even a small number of humans, control the world's resources. We need to be similarly careful with machines." Jeff Hawkins, 2022 [4]

6.1 AI and Human Safety

6.2 Useful Incompatability

It's a feature not a bug that no single comupter can control all others, AKA Security by Obscurity

6.3 Checks and Balances

6.4 Free-Rider Problems

I can steal your AI quite easily from outputs

6.5 When You Can't Tell The Difference

Talk about Taleb's aphorisms "Another definition of modernity: conversations can be more and more completely reconstructed with clips from other conversations taking place at the same time on the planet.", "You are alive in inverse proportion to the density of cliches in your writing."

6.6 Sucker Traps

"(Traditional) search engines are databases, organized collections of data that can be stored, updated, and retrieved at will. (Traditional) search engines are indexes. a form of database, that connect things like keywords to URLs; they can be swiftly updated, incrementally, bit by bit (as when you update a phone number in the database that holds your contacts).

Large language models do something very different: they are not databases; they are text predictors, turbocharged versions of autocomplete. Fundamentally, what they learn are relationships between bits of text, like words, phrases, even whole sentences. And they use those relationships to predict other bits of text. And then they do something almost magical: they paraphrase those bits of texts, almost like a thesaurus but much much better. But as they do so, as they glom stuff together, something often gets lost in translation: which bits of text do and do not truly belong together." Gary Marcus, 2023 [15]

"The sucker's trap is when you focus on what you know and what others don't know, rather than the reverse."

6.7 Dead Inside

"If you know, in the morning, what your day looks like with any precision you are a little bit dead - the more precision the more dead you are."

Revolutionary for Whom? 7

"The inhabitant of London could order by telephone, sipping his morning tea in bed, the various products of the whole earth – he could at the same time and by the same means adventure his wealth in the natural resources and new enterprise of any quarter of the world – he could secure forthwith, if he wished, cheap and comfortable means of transit to any country or climate without passport or other formality." John Maynard Keynes, 1920 [16]

7.1 Self-Driving Horses

7.2 The Battle of the Assistants

Butler vs Indian Virtual Assistant vs Siri

7.3 Employees That Are Better Than You

Respond directly to Jon Krohn's TED talk about monkeys being dumber than us... what about construction equipmenth that's stronger than us physically, or racism/eugenics people that are dumber than us [17]

[17]: (2022), *Jon Krohn*

7.4 Who is Helped the Most?

7.5 Who is Hurt the Most?

7.6 How to Respond

7.7 This Book is a Case Study

Errors and Omissions 8

Bibliography

Here are the references in citation order.

[1] John von Neumann and Ray Kurzweil. *The Computer and the Brain (The Silliman Memorial Lectures Series)*. New Haven, CT, USA: Yale University Press, Aug. 2012 (cited on page ii).

[2] Matt Welsh. *The end of programming*. Jan. 2023. URL: https://cacm.acm.org/magazines/2023/1/267976-the-end-of-programming/fulltext (cited on page 1).

[3] Elaine Rich, Kevin Knight, and Shivashankar B. Nair. *Artificial Intelligence*. Tata McGraw-Hill, 2009 (cited on page 1).

[4] Jeff Hawkins. *A thousand brains: A new theory of intelligence*. Basic Books, 2022 (cited on pages 7, 30).

[5] Caglar Aytekin. 'Neural Networks are Decision Trees'. In: (2022). DOI: 10.48550/ARXIV.2210.05189 (cited on pages 8, 29).

[6] Franc ois Chollet. *Deep learning with python, second edition*. Manning Publications, 2022 (cited on pages 14, 16).

[7] Brian Christian. *The Alignment Problem: Machine Learning and Human Values*. New York, NY, USA: W. W. Norton & Company, Oct. 2020 (cited on page 23).

[8] Nicholas Carlini et al. *Extracting Training Data from Diffusion Models*. 2023. DOI: 10.48550/ARXIV.2301.13188. URL: https://arxiv.org/abs/2301.13188 (cited on page 24).

[9] Vikram Mansharamani. *Think for yourself: Restoring common sense in an age of experts and artificial intelligence*. Harvard Business Review Press, 2020 (cited on page 26).

[10] Reid Blackman. *Ethical Machines: Your Concise Guide to Totally Unbiased, Transparent, and Respectful AI*. Harvard Business Review Press, July 2022 (cited on page 27).

[11] MacAskill2022. 'The Case for Longtermism'. In: *The New York Times* (Aug. 5, 2022). (Visited on 08/05/2021) (cited on page 29).

[12] Cade Metz. 'The Long Road to Driverless Trucks'. In: *N.Y. Times* (Sept. 2022) (cited on page 29).

[13] Cade Metz. 'Stuck on the Streets of San Francisco in a Driverless Car'. In: *N.Y. Times* (Sept. 2022) (cited on page 29).

[14] Jason Torchinsky and Beau Boeckmann. *Robot, take the wheel: The road to autonomous cars and the lost art of driving*. Apollo Publishers, 2019 (cited on page 29).

[15] Gary Marcus. *Is chatgpt really a 'code red' for google search?* Jan. 2023. URL: https://cacm.acm.org/blogs/blog-cacm/268376-is-chatgpt-really-a-code-red-for-google-search/fulltext (cited on page 31).

[16] John Maynard Keynes, Elizabeth Johnson, and Donald Moggridge. *The Collected Writings of John Maynard Keynes (Volume 5)*. Cambridge, England, UK: Cambridge University Press, Dec. 2012 (cited on page 32).

[17] *Jon Krohn*. [Online; accessed 18. Oct. 2022]. Oct. 2022. URL: https://www.jonkrohn.com/posts/2022/10/7/tedx-talk-how-neuroscience-inspires-ai-breakthroughs-that-will-change-the-world (cited on page 32).