Foundations of Statistical and Machine Learning for Actuaries

Artificial Intelligence, Natural Language Processing, and ChatGPT

Edward (Jed) Frees, University of Wisconsin - Madison
Andrés Villegas Ramirez, University of New South Wales

July 2025

Schedule

Day and Time	Presenter	Topics	Notebooks for Participant Activity
Monday Morning	Jed	Welcome and Foundations Hello to Google Colab	Auto Liability Claims
	Jed	Classical Regression Modeling	Medical Expenditures (MEPS)
Monday Afternoon	Andrés	Regularization, Resampling, Cross-Validation	Seattle House Sales
	Andrés	Classification	Victoria road crash data
Tuesday Morning	Andrés	Trees, Boosting, Bagging	
	Jed	Big Data, Dimension Reduction and Non-Supervised Learning	Big Data, Dimension Reduction, and Non-Supervised Learning
Tuesday Afternoon	Jed	Neural Networks	Seattle House Prices Claim Counts
	Jed	Graphic Data Neural Networks	MNIST Digits Data
Tuesday 4 pm	Fei	Fei Huang Thoughts on Ethics	
Wednesday Morning	Jed	Recurrent Neural Networks, Text Data	Insurer Stock Returns
	Jed	Artificial Intelligence, Natural Language Processing, and ChatGPT	
Wednesday After Lunch	Dani	Dani Bauer Insights	
Wednesday Afternoon	Andrés	Applications and Wrap-Up	

Wednesday Morning 5B. Artificial Intelligence, Natural Language Processing, and ChatGPT

Machine Learning

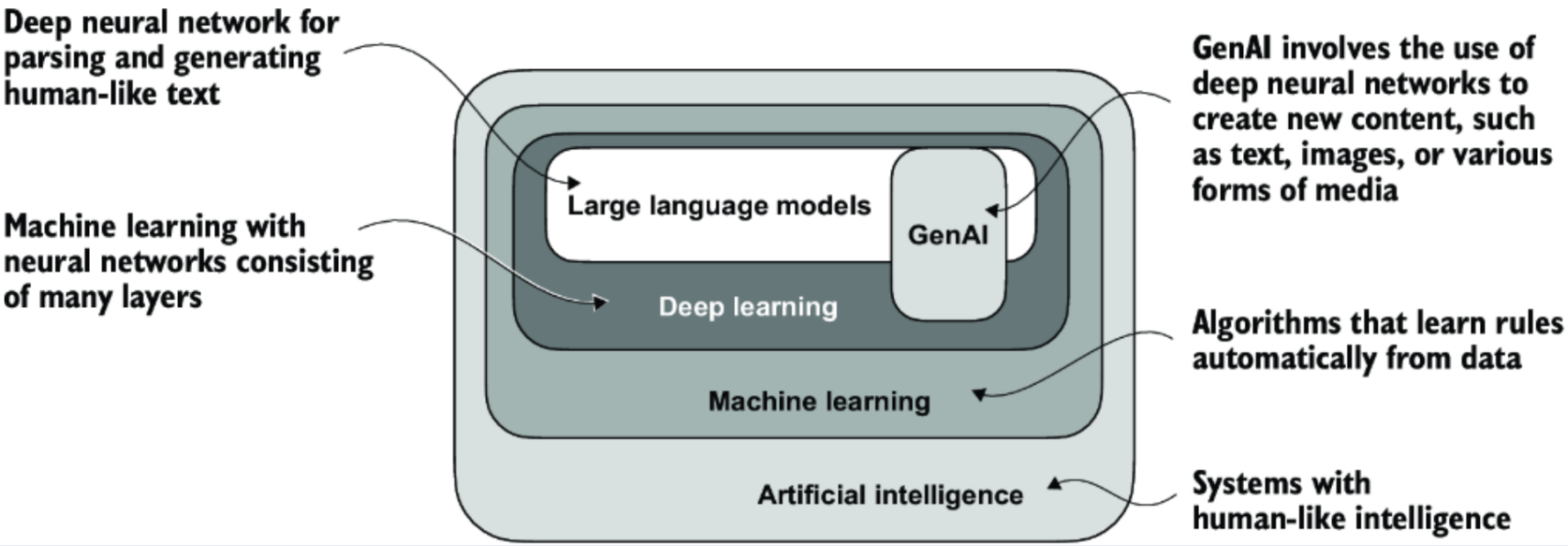
- **Machine learning** involves the development of algorithms that can learn from and make predictions or decisions based on data without being explicitly programmed.
  - To illustrate this, imagine a spam filter as a practical application of machine learning.
  - Instead of manually writing rules to identify spam emails, a machine learning algorithm is fed examples of emails labeled as spam and legitimate emails.
  - By minimizing the error in its predictions on a training dataset, the model then learns to recognize patterns and characteristics indicative of spam, enabling it to classify new emails as either spam or not spam.

Deep Learning

- **Deep learning** is a subset of machine learning that focuses on utilizing neural networks with three or more layers (also called deep neural networks) to model complex patterns and abstractions in data.
  - In contrast to deep learning, traditional machine learning requires manual feature extraction. This means that human experts need to identify and select the most relevant features for the model.

The Scope of Artificial Intelligence

- While the field of **AI** is now dominated by machine learning and deep learning, it also includes other approaches—for example, using rule-based systems, genetic algorithms, expert systems, fuzzy logic, or symbolic reasoning.



{ *Credit*: - Build a Large Language Model (From Scratch) }

A Short History of Large Language Models

- 2017: **Transformer architecture** is proposed based on the **self-attention mechanism**, which would then become the de facto architecture for most of the subsequent DL systems (Vaswani et al., 2017);
- 2018: GPT-1 (**G**enerative **P**re-Trained **T**ransformer) for natural language processing with 117 million parameters, starting a series of advances in the so-called **large language models (LLMs)**;
  - 2019: GPT-2 with 1.5 billion parameters;
  - 2020: GPT-3 with 175 billion parameters;
- 2022: ChatGPT: a popular chatbot built on GPT-3, astonished the general public, sparking numerous discussions and initiatives centered on AI safety;
- 2023: GPT-4 with ca. 1 trillion parameters (OpenAI, 2023), which allegedly already shows some sparks of artificial general intelligence (AGI) (Bubeck et al., 2023).

{ *Source:* Portfolio Optimization: Theory and Applications, 2025, by Daniel P. Palomar }

Disclaimer

- I use ChatGPT as an example of an AI system simply because it is well known.
  - There are other great tools available
- In addition, I note that the University of Wisconsin endorses other tools.
  - They prefer Microsoft 365 **Copilot Chat** and **Google Gemini**
  - They also recommend meeting tools such as *Webex AI Assistant* and *Zoom AI Companion*
- Part of their rational is that, unlike public AI services, these tools prevent your data from being used to train AI models while providing secure support for writing, research, and administrative tasks.

Large Language Models

- As we have seen, the phrase *natural language processing* (NLP) refers to broad field of computer science and linguistics focused on how machines process and understand human language.
- I now wish to focus on a subset of NLP, a *Large Language Model* (LLM).
  - This is a large-scale, deep learning-based model (e.g., GPT, BERT) that is trained on massive text corpora.
  - Its purpose is to **predict** or **generate** language.

Large Language Models (LLMs)

- Modern LLMs are trained in two main steps:
  - They are pretrained on a large corpus of unlabeled text by using the prediction of the next word in a sentence as a label.
  - They are fine-tuned on a smaller, labeled target dataset to follow instructions or perform classification tasks.

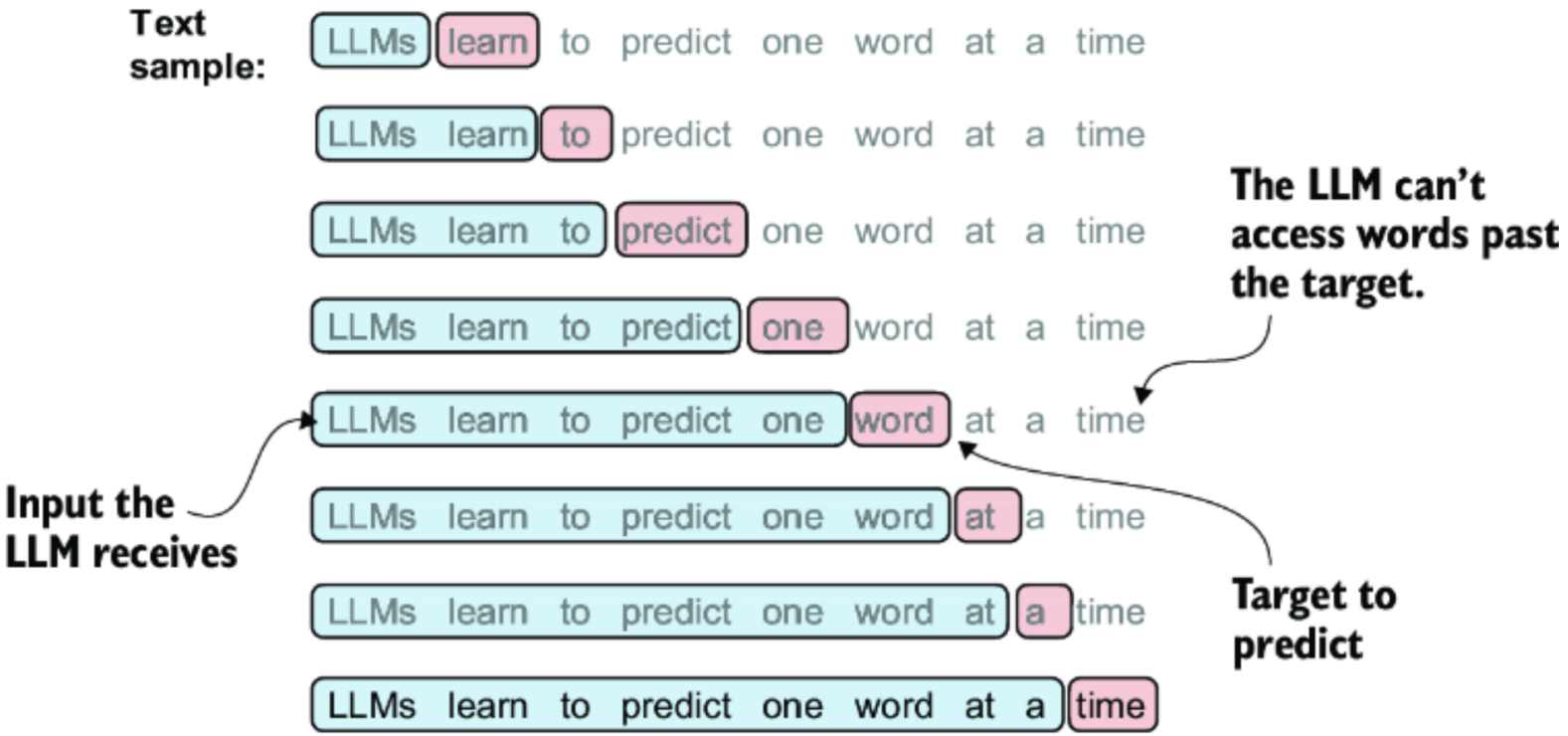


Figure 2.12 Given a text sample, extract input blocks as subsamples that serve as input to the LLM, and the LLM's prediction task during training is to

Word Embedding and Large Language Models

- **Pre-training models**
  - One popular method for training word embeddings is Word2Vec, which uses a neural network to predict the surrounding words of a target word in a given context.
  - Another: GloVe (Global Vectors for Word Representation), which leverages global statistics to create embeddings.
- The embedding size refers to the dimensionality of the model's hidden states.
  - It is a tradeoff between performance and efficiency.
  - The smallest GPT-2 models (117M parameters) use an embedding size of 768 dimensions to provide concrete examples.
  - The largest GPT-3 model (175B parameters) uses an embedding size of 12,288 dimensions.
- The byte pair encoding (BPE) tokenizer used for LLMs like GPT-2 and GPT-3 can efficiently handle unknown words by breaking them down into subword units or individual characters.

Features of LLMs

- An LLM is a neural network designed to understand, generate, and respond to human-like text.
- The "large" in "large language model" refers to both the model's size in terms of parameters and the immense dataset on which it's trained.
- LLMs have remarkable capabilities to understand, generate, and interpret human language.
  - They can process and generate text in ways that appear coherent and contextually relevant
  - They **do not** possess human-like consciousness or comprehension.
- LLMs are trained on vast quantities of text data.
  - This allows LLMs to capture deeper contextual information and subtleties of human language compared to previous approaches.

Insurance Examples of Text Data

- **1. Claims Processing and Triage**
  - Use Cases:
    - Automatically extract and interpret **incident descriptions** from claim forms.
    - Assign priority, flag for fraud, or route to the appropriate handler.
  - Example Tasks:
    - **Named Entity Recognition** to find people, dates, locations in free-text descriptions.
    - **Text classification** to categorize claims (e.g., fire, theft, flood).
  - References:
    - Chen et al. (2020) – *"Automated Claims Triage using Natural Language Processing"*, Applied AI in Insurance (Zurich Insurance).
    - McKinsey (2021) – *"The next frontier in claims automation"* emphasizes NLP's role in triage and fraud detection.

Insurance Examples of Text Data 2

- **2. Underwriting Support from Unstructured Documents**
  - Use Case: Extract key information from **insurance applications, medical reports, or inspection documents**.
  - Example Tasks:
    - Optical Character Recognition (OCR) + NLP to process PDFs.
    - Extracting risk factors from physician notes or building inspections.
  - References:
    - Accenture (2020) – Reported insurers can reduce underwriting time by 30–50% by incorporating AI/NLP.
    - IEEE Access, 2022: *"Text Mining for Underwriting Automation in Life Insurance"* — explores risk evaluation from unstructured EHRs.
- **3. Customer Sentiment and Service Feedback**
  - Use Case: Analyze **customer emails, chat logs, and surveys** to detect dissatisfaction or emerging issues.
  - Reference: Eling & Lehmann (2018): *"The Impact of Digital Transformation on the Insurance Industry"*, *Geneva Papers on*

Insurance Examples of Text Data 3

- **4. Fraud Detection from Narrative Discrepancies**
  - Use Case: Compare **free-text statements** from claimants, police reports, and witnesses to identify inconsistencies.
  - Example Tasks:
    - Semantic similarity
    - Stylometry or linguistic anomaly detection
  - References:
    - ABI (UK Association of British Insurers, 2022) – Emphasizes increasing use of NLP for fraud flagging.
    - IEEE Transactions on Knowledge and Data Engineering, 2021 – *"Deep Learning for Insurance Fraud Detection from Claims Text"*.

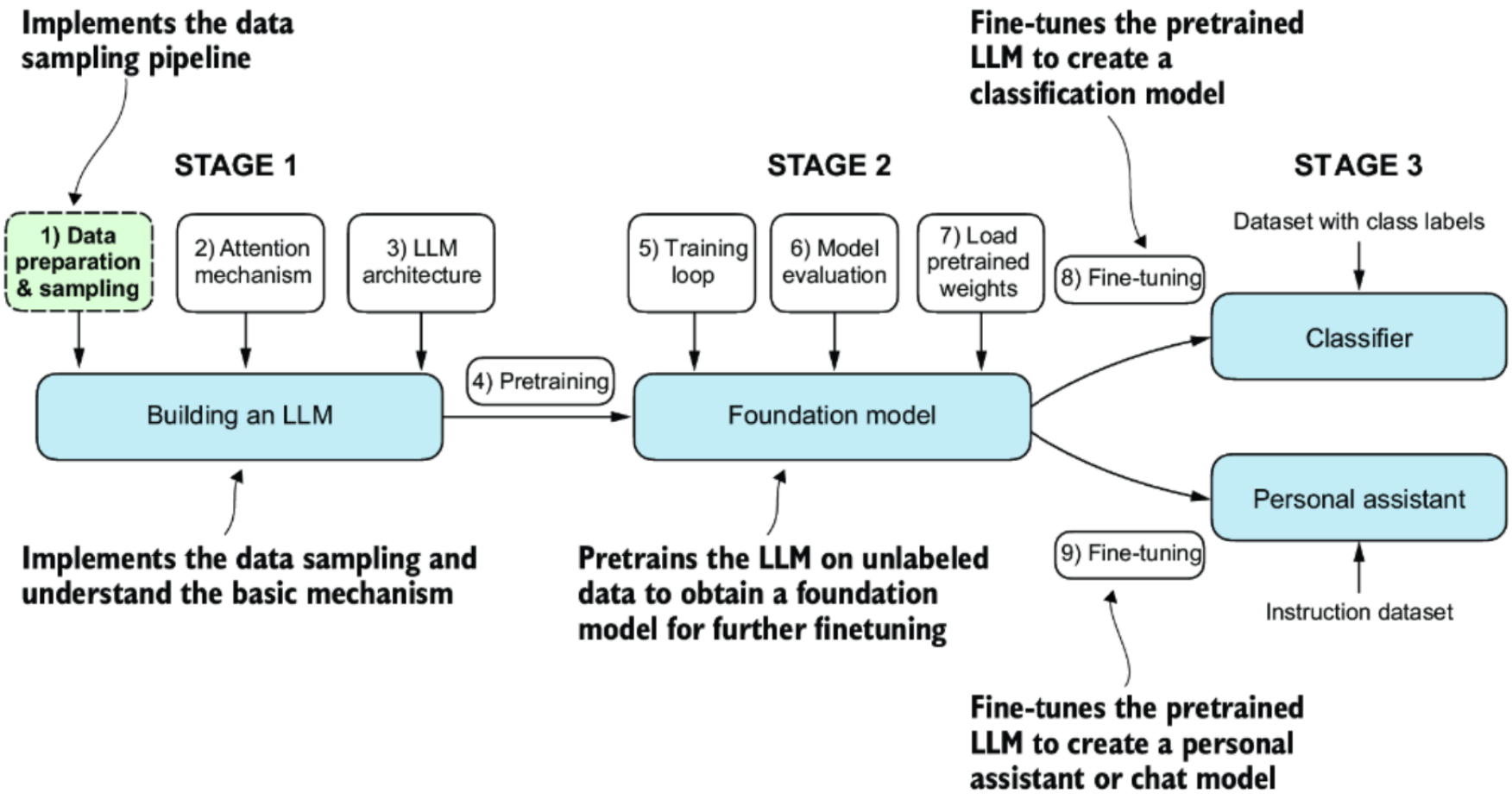
Insurance Examples of Text Data 4

- **5. Policy Recommendations and Chatbots**
  - Use Case: Power **intelligent virtual assistants** that understand user intent in natural language.
  - Example Tasks:
    - Intent classification ("I want to insure my house")
    - Dialogue systems (multi-turn policy recommendation)
  - References:
    - **Lemonade Insurance** uses NLP-based bots ("Maya") to issue policies and process claims.
    - EY Report (2021) – *"AI in insurance: Closing the customer expectation gap"*.
- **6. Litigation and Legal Document Review**
  - Use Case: Analyze legal documents, policy clauses, and court cases related to **coverage disputes** or litigation.
  - Example Tasks:
    - Clause extraction
    - Document classification (e.g., denial vs approval)
  - References:
    - Insurance Law NLP Project (Stanford) – Applied BERT

Caveats

- These sources were generated by **ChatGPT**
- However, a check later by another Large Language Model, Microsoft's *Copilot*, found that the use cases were
  - Technically feasible with current NLP tools.
  - Supported by credible references (academic, industry, or real-world implementations).
  - Relevant to actuarial and insurance domains.
- However, **not all** the references were real and accurate
  - Some were "likely illustrative", "Plausible, but not directly verifiable", and "likely inspired by real work."
- In large language models (LLMs), "hallucinations" refer to the phenomenon where the model generates information that is false, misleading, or nonsensical, even if it appears coherent and plausible.
  - See, for example, A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions - widely cited!

- Here is plan for building a LLM.



{ *Credit*: - Build a Large Language Model (From Scratch) }

Autoencoding

- To understand the transformer architecture, it is helpful to introduce *encoder* and *decoder* concepts.
  - To do so, let us also introduce the idea of autoencoding.
- An **autoencoder** is a type of neural network trained to reconstruct its input. It learns a compressed (encoded) representation and then reconstructs the original data from that encoding.
- It consists of two parts:
  - **Encoder**: Compresses the input into a lower-dimensional representation

$$\mathbf{z} = f_{enc}(\mathbf{x})$$

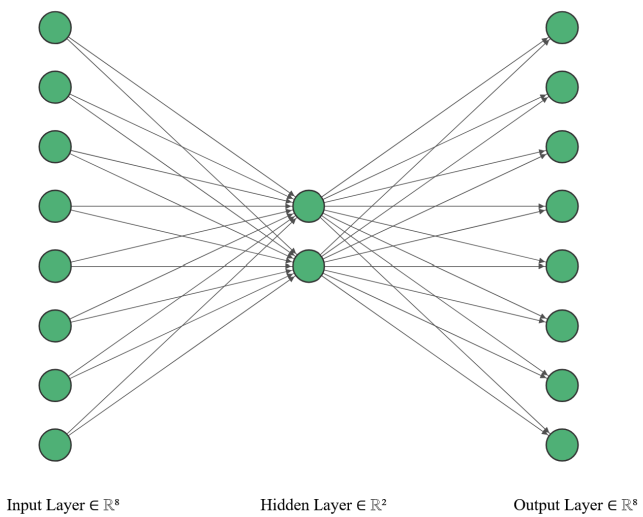
  - **Decoder**: Reconstructs the input from the latent code

$$\hat{\mathbf{x}} = f_{dec}(\mathbf{z})$$

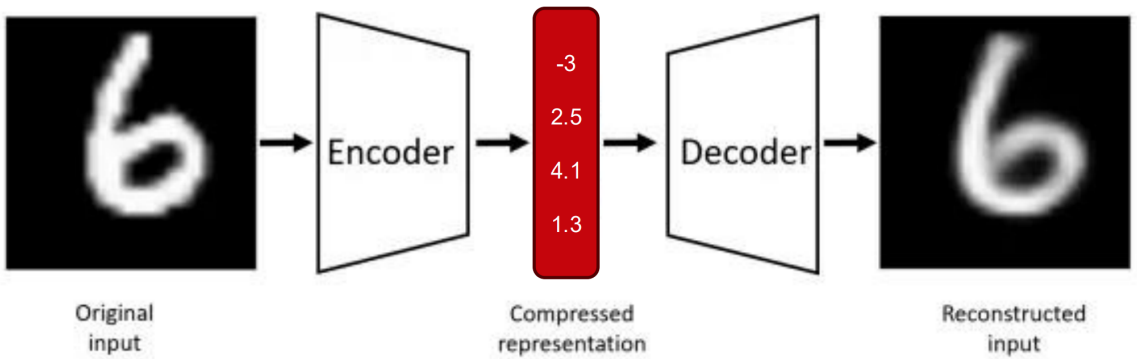
- The network is trained to minimize:

$$||\mathbf{x} - \hat{\mathbf{x}}||^2$$

- How can autoencoding be used?
  - Data compression
  - Feature extraction
  - Nonlinear PCA (in fact, autoencoders are often viewed as nonlinear generalizations of PCA)
  - Denoising representations
  - Low-dimensional embeddings for visualization



- Auto-Encoders are neural networks used in unsupervised learning
  - In this way, we can compress image data (recall more traditional methods like principal components analysis)
- Since we condense information (compression!), the predictions generally won't be perfect
  - One can represent the image information via a (limited) set of numbers and thus compare image by similarity of those numbers



If you have time and interest, check out this terrific tutorial on auto-encoders

Transformer Architecture

- LLMs utilize an architecture called the *transformer*
  - This allows them to pay selective attention to different parts of the input when making predictions
  - It makes them especially adept at handling the nuances and complexities of human language.
- A **transformer** is a deep learning model architecture designed for handling sequences (like language) using a mechanism called **self-attention**.
  - It replaces recurrent models like LSTMs and GRUs with multi-head attention and feedforward layers, enabling fast, scalable training.
- The input embeddings consist of word embeddings and **positional encoders**.

Positional Encoders

- **Positional encoding** is added to capture word order (since the model lacks recurrence). Transformers process all tokens in parallel (via self-attention), so unlike RNNs or CNNs, there is no inherent sequence or position.
  - The positional encoder is a vector added to the word embedding of each token to indicate its position in the sequence.
  - It is calculated using a deterministic formula involving sine and cosine functions.
  - Sines and cosines are use because the can correspond to a different frequencies and are smooth, continuous, and differentiable patterns.

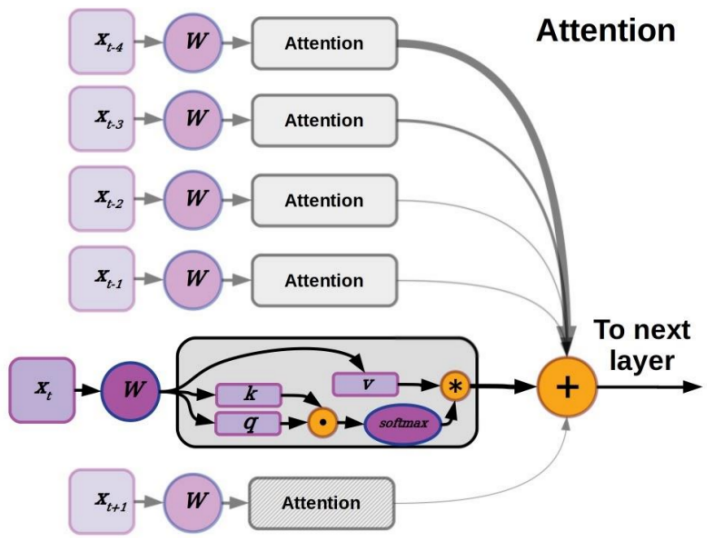
- Just for fun, here are some formulas:

$$\begin{aligned} PE_{(pos,2i)} &= \sin\left(\frac{pos}{10000^{2i/d}}\right) \\ PE_{(pos,2i+1)} &= \cos\left(\frac{pos}{10000^{2i/d}}\right) \end{aligned}$$

- with pos = position in the sequence, i = dimension index, d = total number of dimensions (e.g., 300).

Attention and Transformers

Transformers are an architecture that pays "attention" to the entire past, and figures out what the important elements are



{ *Credit*: Dani Bauer Lecture Notes }

Self-attention mechanism

- A key component of transformers and LLMs is the **self-attention mechanism**
  - It allows the model to weigh the importance of different words or tokens in a sequence relative to each other.
  - This mechanism enables the model to capture long-range dependencies and contextual relationships within the input data, enhancing its ability to generate coherent and contextually relevant output.
  - The attention mechanism gives the LLM selective access to the whole input sequence when generating the output one word at a time.
  - In contrast, with a RNN, one only retains a *summary* of the history in a "memory cell". This means one can loose information. Not a problem for short, concise sentences but can be an issue for longer, more complex sentences.

Attention Mechanism

- Here is an application, translating German to English
- To predict the second token ("you"), the transformer can rely upon previous English words and *all* of the German words.
  - Each word/token receives a so-called "attention weight".
  - Note that the German word "du" has a large black dot, indicating its score is high...

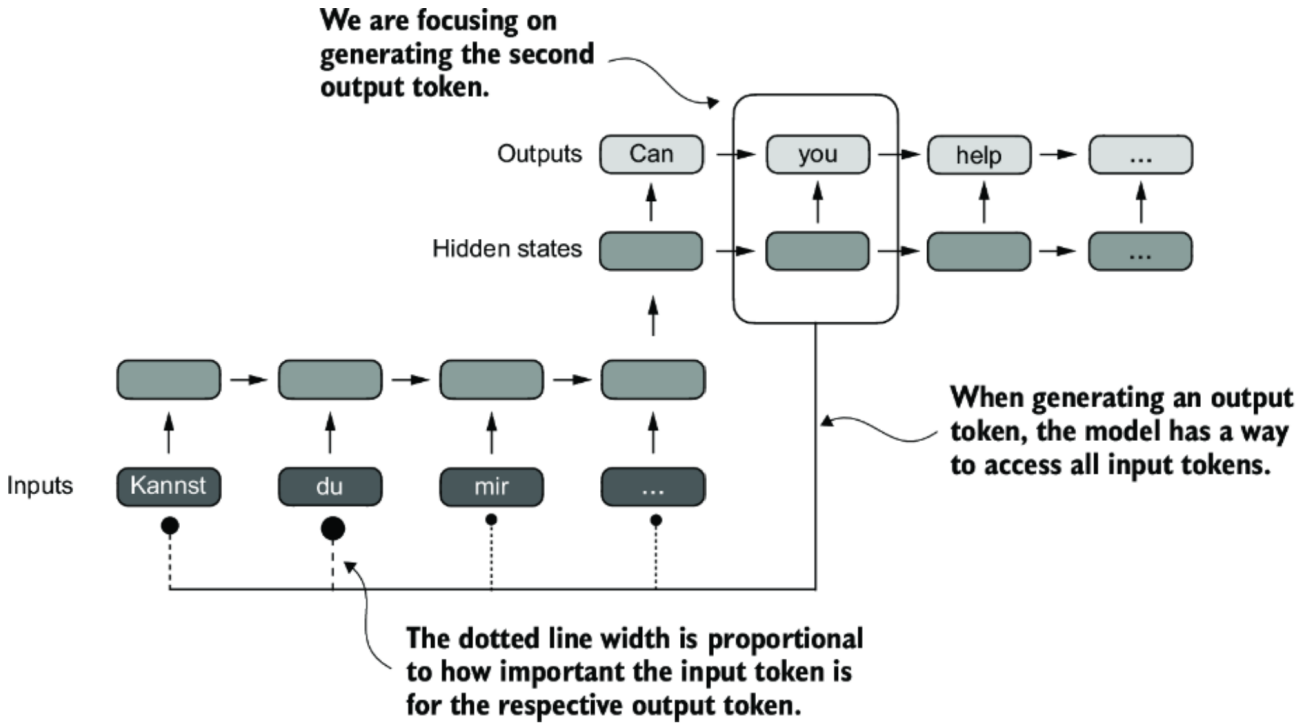
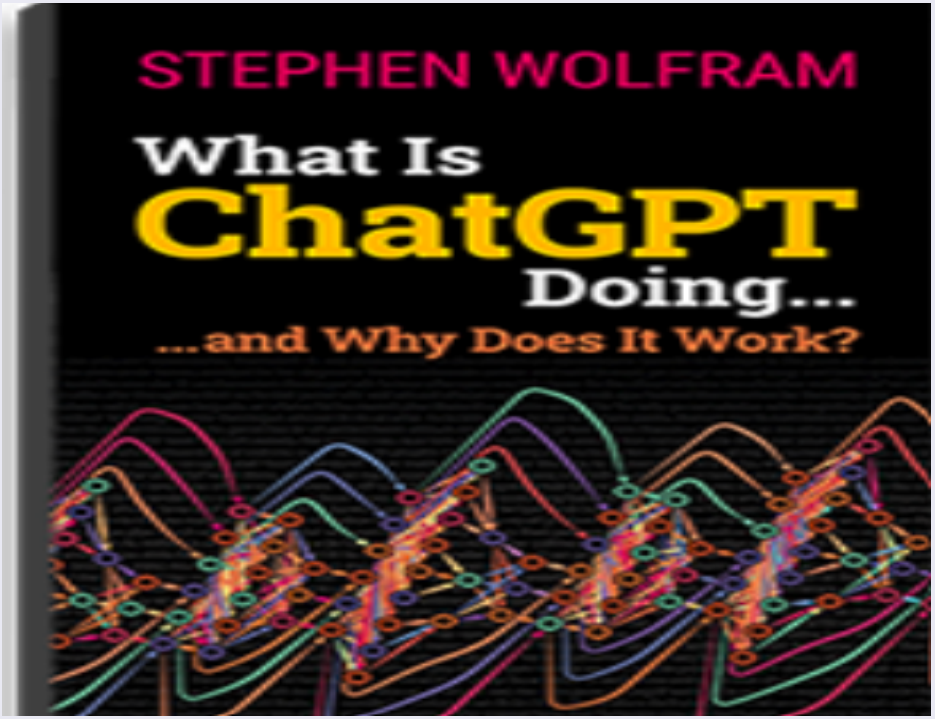


Figure 3.5 Using an attention mechanism, the text-generating decoder part of

What Is ChatGPT Doing - and Why Does It Work? by Wolfram 2023



What ChatGPT is always fundamentally trying to do is to produce a "reasonable continuation" of whatever text it's got so far, where by "reasonable" we mean "what one might expect someone to write after seeing what people have written on billions of webpages, etc."

What Is ChatGPT Doing - and Why Does It Work?

The idea of transformers is to do something at least somewhat similar for sequences of tokens that make up a piece of text. But instead of just defining a fixed region in the sequence over which there can be connections, transformers instead introduce the notion of "attention" — and the idea of "paying attention" more to some parts of the sequence than others.

... even given all that training data, it's certainly not obvious that a neural net would be able to successfully produce "human-like" text. And, once again, there seem to be detailed pieces of engineering needed to make that happen. But the big surprise—and discovery—of ChatGPT is that it's possible at all.

A Success Story

- The Spanish translation of my regression book was done by *ChatGPT* (over the course of several months)
- It is currently being reviewed by a team of experts for readability.

Modelado de Regresión con Aplicaciones Actuariales y Financieras

Edward (Jed) Frees, University of Wisconsin - Madison, Australian National University

Prefacio

Prólogo

Los actuarios y otros analistas financieros cuantifican situaciones usando datos; somos personas de 'números'. Muchos de nuestros enfoques y modelos son estilizados, basados en años de experiencia e investigaciones realizadas por legiones de analistas. Sin embargo, el mundo financiero y de gestión de riesgos evoluciona rápidamente. Muchos analistas se enfrentan a

Generative AI

Generative AI is a field within AI focused on creating machines capable of performing tasks that previously required human intelligence.

- Traditional AI: Primarily excels at classification and prediction tasks. Examples include identifying objects in an image (e.g., identifying a "cat").
- Generative AI: Goes beyond classification, excelling at content creation, such as generating new text, images, or music.

Use of deep learning to augment creative activities such as writing, music and art, to generate new things.

Some applications: text generation, deep dreaming, neural style transfer, variational autoencoders and generative adversarial networks.

The following list outlines free tools that are used in specific sub-domains of GenAI:

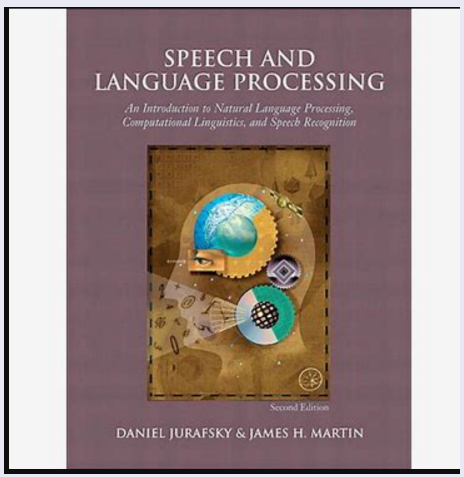
- **Text**: ChatGPT (Free), Claude, Gemini
- **Images**: Bing Image Creator, Ideogram, Leonardo AI (free tokens)
- **Video**: RunwayML (basic tier), Pika Labs (free credits)
- **PDFs**: ChatPDF, Humata
- **Product mockups**: Kittl, Canva AI

{ *Source:* https://www.analyticsvidhya.com/blog/2025/05/getting-into-gen-ai/

See also https://www.linkedin.com/pulse/what-generative-ai-llm-luis-escalante. }

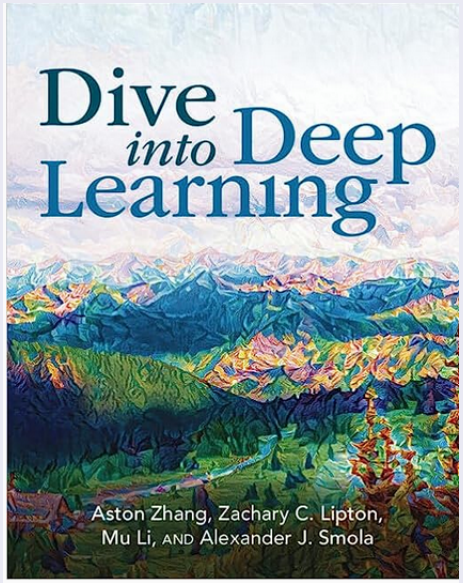
Resources For Future Studies

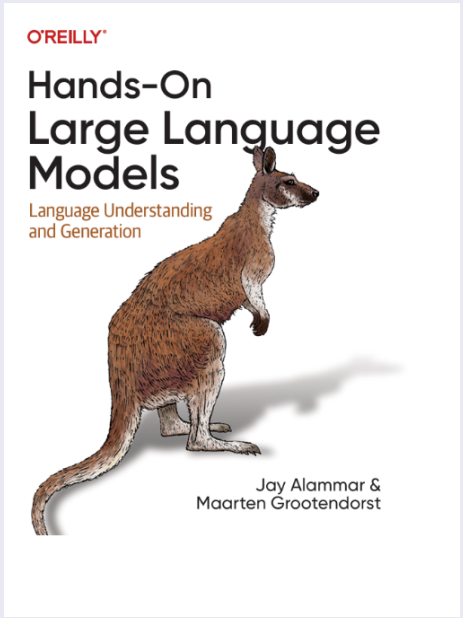
- Sebastian Raschka Build a Large Language Model (From Scratch)
  - Raschka Teaching Site
- "Speech and Language Processing" by Dan Jurafsky and James H. Martin
  - *Course:* Stanford CS224n - Natural Language Processing with Deep Learning - YouTube Lectures



More Resources For Future Studies

- Dive Into Deep Learning
- Hands on LLMs





Appendix

Attention Mechanism

- The networks we have focused on have been a well-defined size, such as 28×28 grid of pixels for graphic data.
- In contrast, natural language processing inputs tend to be variable sized, processing sequentially one token at a time.
- To handle this, it has become traditional to think in terms of database concepts including **queries**, **keys**, and **values**, so let's review these ideas first.
- For instance, a database might consist of tuples {("Zhang", "Aston"), ("Lipton", "Zachary"), ("Li", "Mu"), ("Smola", "Alex"), ("Hu", "Rachel"), ("Werness", "Brent")} with the last name being the **key** and the first name being the **value**. Then a **query** such as "Li" would return the value of "Mu".

We can design queries **q** that operate on **(k v,)** pairs in such a manner as to be valid regardless of the database size. Define the database

$$D = \{(\mathbf{k}_1, \mathbf{v}_1), \ldots, (\mathbf{k}_m, \mathbf{v}_m), \}$$

We can define the *attention* over D to be

$$\text{Attention}(\mathbf{q}, D) = \sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$$

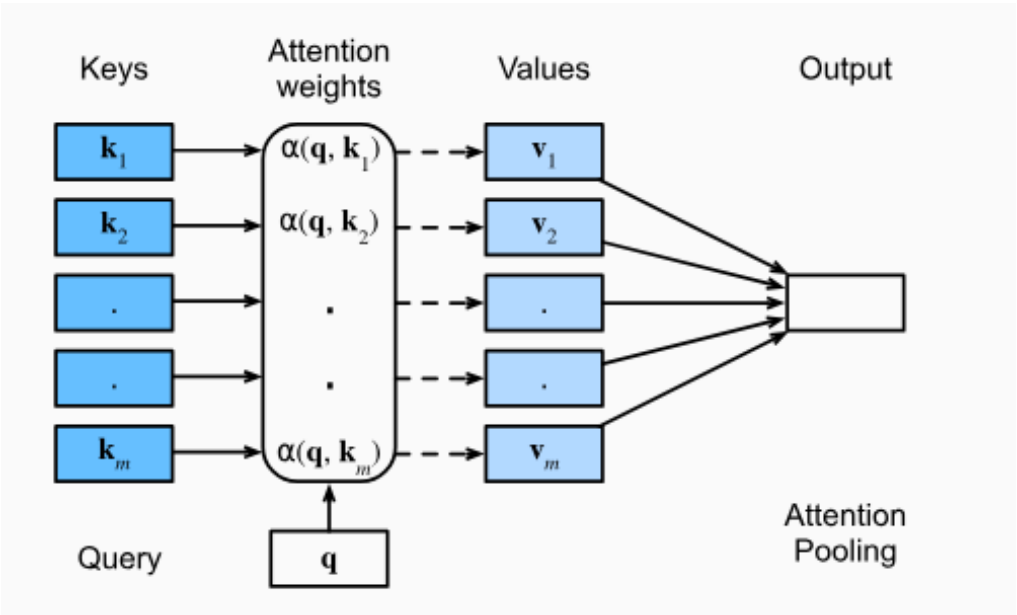
where $\alpha(\mathbf{q}.\mathbf{k}_i)$ are scalar attention weights.

- For interpretation, suppose instead we use $\mathbf{x}_i = \mathbf{k}_i$ for a set of features, the query is a specific point in the feature space, $\mathbf{x} = \mathbf{q}$, and the values $y_i = v_i$ are labeled responses.
- Then, we can interpret $\text{Attention}(\mathbf{x}, D) = \sum_{i=1}^{m} \alpha(\mathbf{x}, \mathbf{x}_i)y_i$ to be a local (nonparametric) regression approximation of a conditional expected value.
- In this case, we interpret $\alpha(\cdot)$ to be a "kernel" function from classical statistics.

For natural language process applications, it is common to use a *softmax* function

$$\alpha(\mathbf{q}, \mathbf{k}_i) = \frac{\exp(a(\mathbf{q}, \mathbf{k}_i))}{\sum_j \exp(a(\mathbf{q}, \mathbf{k}_j))}$$

and choose the *inner product* for $a(\cdot)$ to represent *similarity* between the query and each key. This choice means the weights sum to 1. Further, it is differentiable and its gradient never vanishes, all of which are desirable properties in a model.



Attention Scoring Functions

- The queries and keys may not be of the same length in which case one uses a so-call *masked softmax operation*, that is, instead of $\sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$ use $\sum_{i=1}^{l} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i$ when the length of the query $l < m$.
- *Batch matrix multiplication* is defined so we can do a set of queries at the same time.
- The dot product is typically rescaled by $1/\sqrt{(d)}$, where d is the dimension of the query and keys.
- **Additive Attention**. When the queries and keys are of different length, one can use and *additive attention* score of the form:

$$a(\mathbf{q}, \mathbf{k}) = \mathbf{w}_v' \text{tanh}(\mathbf{W}_q\mathbf{q} + \mathbf{W}_k\mathbf{k})$$

where $\{\mathbf{W}_q, \mathbf{W}_k\}$ are learnable parameters.