AI - Past, Present & Future

Jaideep Ganguly, ScD (MIT)

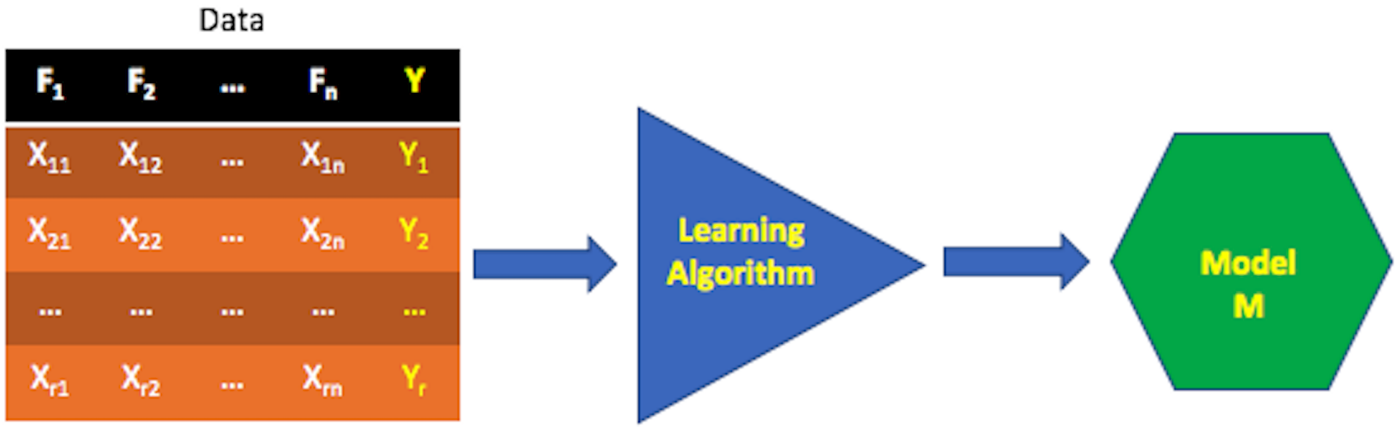
March 31, 2024

About Jaideep

1. VP of Software Development, Salesforce
2. Director of Software Development, Amazon [2010 - 2019].
3. Director of Engineering, Microsoft [2006 - 2010].
4. Doctor of Science, Massachusetts Institute of Technology (MIT).
5. Master of Science, Massachusetts Institute of Technology (MIT).
6. Bachelor of Technology (Honours), Indian Institute of Technology, Kharagpur.

What is Machine Learning (ML)? Classification!

1. In 1959, Arthur Samuel (MIT), defined ML, a subset of AI, as a *"field of study that gives computers the ability to learn without being explicitly programmed".*



2. ML is effective for complex tasks where deterministic solution don't suffice, e.g., speech recognition, handwriting recognition, spam, fraud detection, etc. These cannot be solved manually in a large scale.

Can a Computer Think?

1. The human brain is a remarkable organ. It has enabled us understand science and advance mankind.
2. The idea of mimicking the human brain or even improving the human cognitive functions is an alluring one and is an objective of Artificial Intelligence research.
3. But we are not even close in-spite of a century of research. However, it continues to have a major hold on our imagination given the potential of the rewards.
4. *The question of whether a computer can think is no more interesting than the question of whether a submarine can swim" - Dijkstra*
5. It is more interesting to understand the evolution of Machine Learning - *How did it start, here are we today and where do we go from here.*

Knowledge Based (KB) Expert Systems

1. The field of AI was defined as computers performing tasks that were specifically thought of as something only humans can do.
2. In the 1980s, the expert systems were of great interest and focused on knowledge and inference mechanisms. They did a good job in their domains but were narrow in specialization and were difficult to scale.
3. Once these systems worked, they were no longer considered to be AI! For example, today the best chess players are routinely defeated by computers but chess playing is no longer really considered as AI! McCarthy referred to as the "AI effect". IBM's Watson is one such program at a level such as that of a human expert.
4. Fifty years ago Jim Slagle's (MIT) symbolic integration program (MACSYMA) was a tremendous achievement.
5. It is very hard to build a program that has "common sense" and not just narrow domains of knowledge.

From Perceptrons to Deep Learning

1. Rosenblatt (Cornell) is credited with the concept of Perceptrons, "a machine which responds like the human mind" as early as in 1957.
2. In a critical book written in 1969, Marvin Minsky and Seymour Papert showed that Rosenblatt's original system was blind to simple XOR. That was incorrect and the field of "Neural Networks" disappeared!
3. In 2006, Hinton developed Deep Learning which extends earlier important work by Yann LeCun (New York Univ).
4. Deep learning's important innovation is to have models learn categories incrementally, attempting to nail down lower-level categories (like letters) before attempting to acquire higher-level categories (like words).

Deep Learning (DL)

1. DL powers Amazon's Alexa, Google's search, Facebook's news feed and Netflix's recommendation engine. Instead of manually encoding 1,000's of rules, the DL automatically extracts the rules from data.
2. In 2012, during the ImageNet competition in computer vision, Hinton achieved the best accuracy in image recognition by an astonishing margin of more than 10% using DL and sparked renewed research.
3. Every decade has seen focus on a different technique: Neural Networks in the late '50s and '60s, various symbolic approaches in the '70s, KB systems in the '80s, Bayesian Networks in the '90s, Support Vector machines in the '00s, and Neural Networks again in the '10s.
4. And now GPT, LLM in the '20s.

Linear Regression

1. Regression is a statistical approach to find the relationship between variables between X_i and Y_i.
2. A common function used to model training data is a linear regression model. The model or the hypothesis is given by:

$$y_i = \theta_0 + \theta_1 x_{i1} + \theta_2 x_{i2} \cdots + \theta_n x_{in} \tag{1}$$

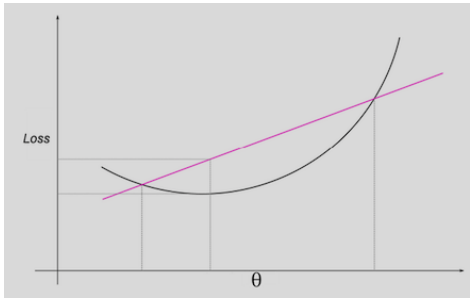
where x_{ij} is the observed value of the feature x_j, y_i is the predicted value of the outcome and θ_i are constants the values of which need to be determined. For now, we limit the the values of the features x_{ij} to numbers, positive or negative. Later on, we will study techniques on how to deal with the situation where the feature value is a string.

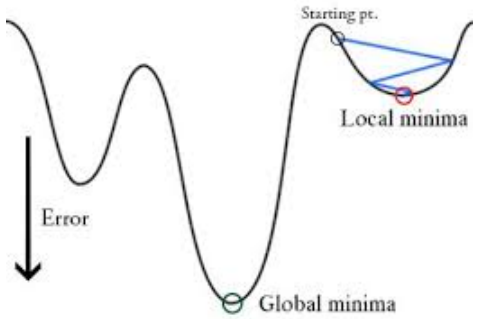
$$\text{Squared Error Loss (L)} = \frac{1}{2} \times \sum_{i=1}^{r} (\hat{y}_i - y_i)^2 \tag{2}$$

This loss function is always positive, there is a minimum, is monotonically increasing from that minimum value in both positive and negative directions. Such a function is called a convex function.

3. We need to minimize some loss function over the training data.

Gradient Descent





1. Minimize by setting the partial derivative of the loss wrt θ_j to zero.

$$\frac{\partial L}{\partial \theta_0} = \sum_{i=1}^{r} (\hat{y}_i - y_i) = 0 \quad \frac{\partial L}{\partial \theta_j} = \sum_{i=1}^{r} (\hat{y}_i - y_i)\, x_{ij} = 0 \tag{3}$$

2. Randomly assigning values to θ_j. For a convex function it does not matter what the initial weights are as it will always converge. $\hat{y}$ and x_{ij} are observed and y_i is computed. This means the slope of the loss function can be readily computed.

Logistic Regression

In many cases the value of y_i need to be bounded between 0 and 1. For logistic regression, Least Squared Error will result in a *non-convex* graph with local minimums and hence is not feasible. In such cases, we use the logistic regression model as given below:

$$y_i = \frac{1}{1 + e^{-z}} \tag{4}$$

$$z_i = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \cdots + \theta_n x_n \tag{5}$$

- for z is large +ve number, $\frac{1}{e^z} = 0$; $y_i = 1$
- for z $= 0$, $y_i = 0.5$
- for z is large -ve number, $y_i = 0$

Hence, the value of y_i is bounded between 0 and 1 for z between -∞ and ∞.

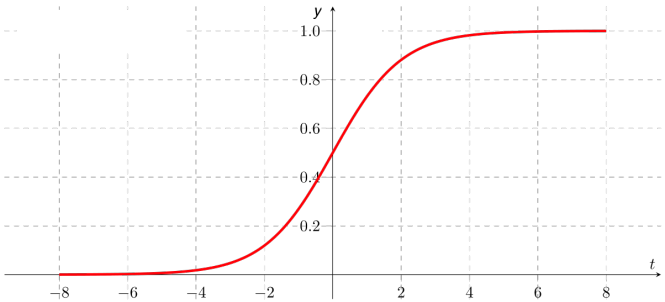
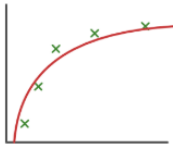
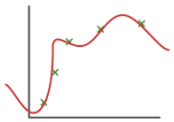


Figure: Sigmoid Curve

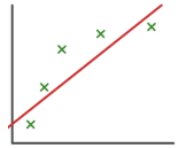
Predictions & Errors



(a) Correct Fit



(b) Over fit



(c) Under fit

Predictions from the model will have differences or errors. Overfitting will occur when an excessive number of features are used than required. Underfitting occurs when an insufficient number of features are used than required.

1. Bias is the difference between average model prediction and the true target value and variance is the variation in predictions across different training data samples. Simple models with small number of features have high bias and low variance whereas complex models with large number of features have low bias and high variance.
2. Regularization is a technique used to avoid problem of overfitting. It prevents overfitting in linear models by a penalty term that penalizes large weight values.

Information & Uncertainty

1. In Claude Shannon's (MIT) information theory, one bit of information reduces the uncertainty by 2. Similarly, if 3 bits of information are sent, then the reduction in uncertainly by 2^3, i.e., 8. This is intuitive. With 3 bits, there could be 8 possible values and so if a particular set of bits are transmitted, 8 possibilities are eliminated with 1 certainty.
2. Information Content When the information is probabilistic, the self-information I_x, or Information Content of *measuring a random variable X as outcome x is defined as:*

$$I_x = log\left(\frac{1}{p(x)}\right) = -log(p(x)) \tag{6}$$

where $p(x)$ is probability mass function.

3. Shannon Entropy of the random variable X is defined as:

$$H(X) = \sum_x -p(x)log(p(x)) = E(I_x) \tag{7}$$

It is the *expected information content* of the measurement of X.

Cross Entropy - Kullback–Leibler (KL) divegence

1. Cross-Entropy is defined as:

$$H(p,q) = -\sum_i p(i) log_2 q(i) \tag{8}$$

where p is the true distribution and q is the predicted distribution. If the predictions are perfect, then the cross-entropy is same as the entropy. If the prediction differs, then there is a divergence which is known as *Kullback – Leibler (KL) divergence*. Hence,

$$Cross\ Entropy = Entropy + KL\ Divergence$$

$$KL\ Divergence = H(p,q) - H(p)$$

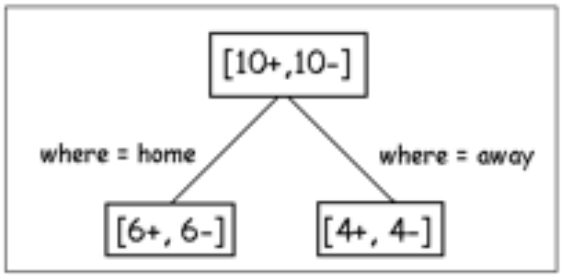
Hence, if the predicted distribution is closer to true distribution when KL divergence is low.

Decision Tree

1. Decision Tree is a supervised machine learning algorithm where You try to separate your data and group the samples together in the classes they belong to. You maximize the purity of the groups as much as possible each time you create a new node of the tree.
2. At each step, each branching, you want to decrease the entropy, so this quantity is computed before the cut and after the cut. If it decreases, the split is validated and we can proceed to the next step, otherwise, we must try to split with another feature or stop this branch.
3. XGBoost is a machine learning algorithm that belongs to the ensemble learning category, specifically the gradient boosting framework. It utilizes decision trees as base learners and employs regularization techniques to enhance model generalization.

Example

Consider the following table for win/loss of soccer games at home and away.



The entropy is:

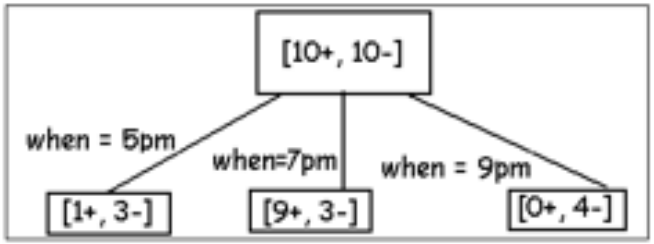
$$H\left(\frac{6}{12},\frac{6}{12}\right) = -\frac{6}{12}log_2\left(\frac{6}{12}\right) - \frac{6}{12}log_2\left(\frac{6}{12}\right) = 0.69$$

$$H\left(\frac{4}{8},\frac{4}{8}\right) = -\frac{4}{8}log_2\left(\frac{4}{8}\right) - \frac{4}{8}log_2\left(\frac{4}{8}\right) = 0.69$$

$$H = -\frac{12}{20} \times H\left(\frac{6}{12},\frac{6}{12}\right) - \frac{8}{20}H\left(\frac{4}{8},\frac{4}{8}\right) = 0.69$$

Example

Partitioning by when, we have:



$$H(5pm) = H\left(\frac{1}{4}, \frac{3}{4}\right) = -\frac{1}{4}log_2\left(\frac{1}{4}\right) - \frac{3}{4}log_2\left(\frac{3}{4}\right) = 0.56$$

$$H(7pm) = H\left(\frac{9}{12}, \frac{3}{12}\right) = -\frac{9}{12}log_2\left(\frac{9}{12}\right) - \frac{3}{12}log_2\left(\frac{3}{12}\right) = 0.56$$

$$H(9pm) = H\left(\frac{0}{4}, \frac{4}{4}\right) = -\frac{0}{4}log_2\left(\frac{0}{4}\right) - \frac{4}{4}log_2\left(\frac{4}{4}\right) = 0.00$$

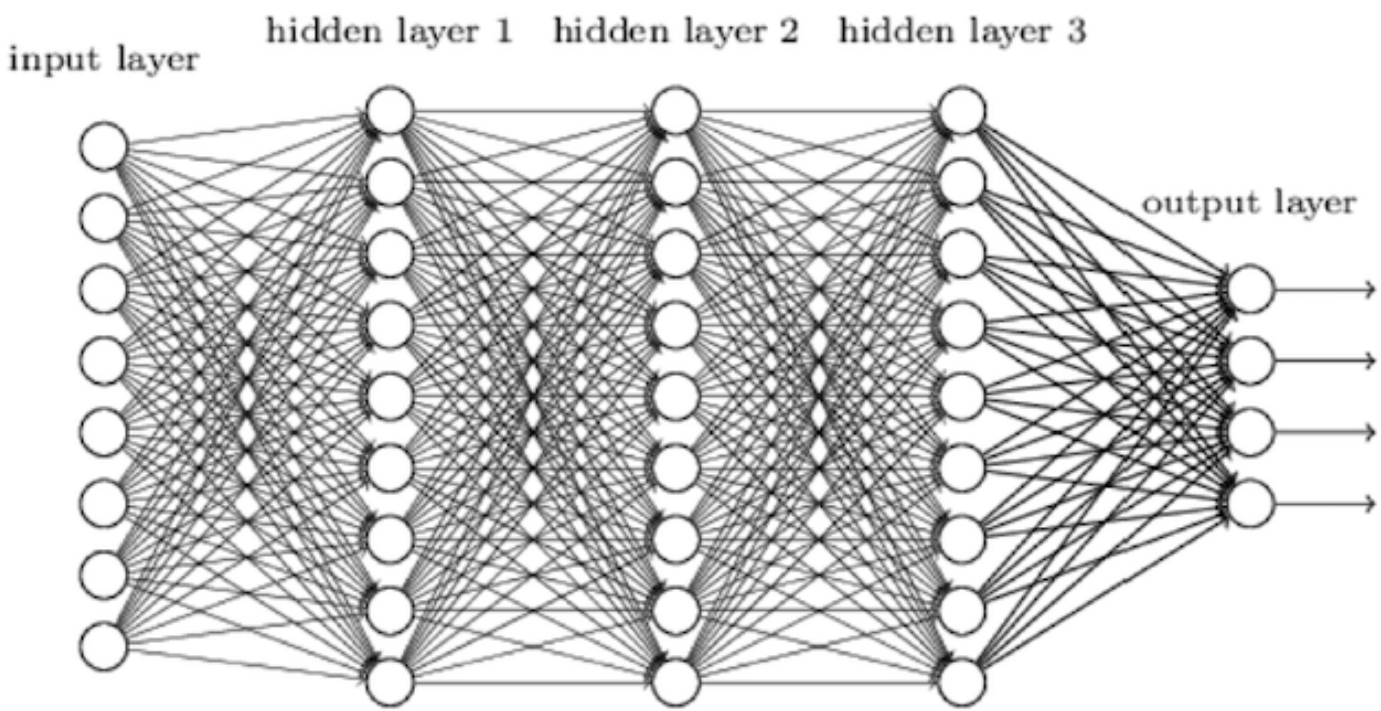
Entropy after partition is:

$$H = -\frac{4}{20}H\left(\frac{1}{4}, \frac{3}{4}\right) - \frac{12}{20}H\left(\frac{12}{20}, \frac{3}{4}\right) - \frac{4}{20}H\left(\frac{4}{20}, \frac{3}{4}\right) = 0.45$$

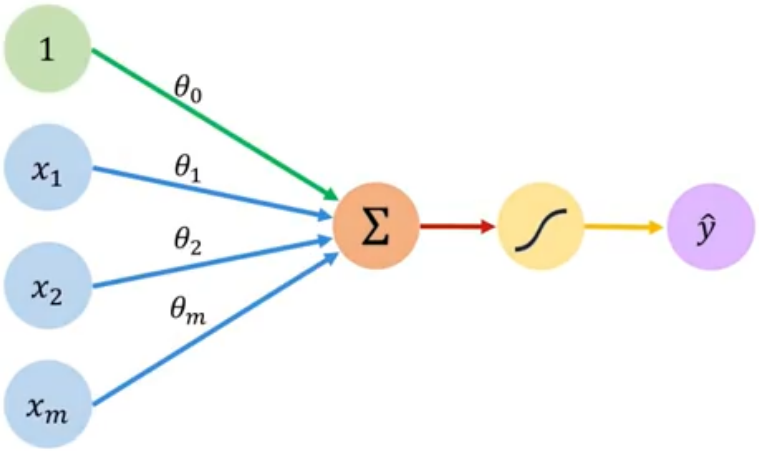
Hence, the Information gain is 0.69 - 0.45 = 0.24.

Deep Neural Net

1. Deep Learning Success Stories - Image Recognition, Speech Comprehension, Chatbot.
2. DNNs are suitable where the raw underlying features are not individually interpretable. This success is attributed to their ability to learn hierarchical representations, unlike traditional methods that rely upon hand-engineered features.



Deep Neural Net



Inputs Weights Sum Non-Linearity Output

Output

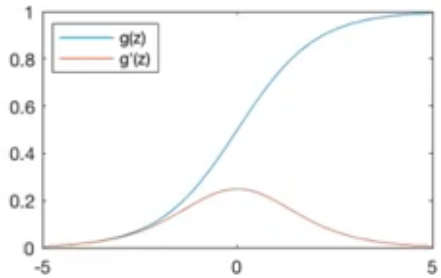
Linear combination of inputs

$$\hat{y} = g\left(\theta_0 + \sum_{i=1}^{m} x_i\ \theta_i\right)$$

Non-linear activation function

Bias

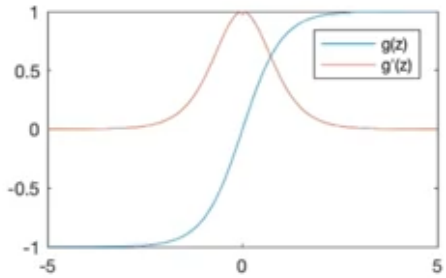
Sigmoid Function



$$g(z) = \frac{1}{1 + e^{-z}}$$

$$g'(z) = g(z)(1 - g(z))$$

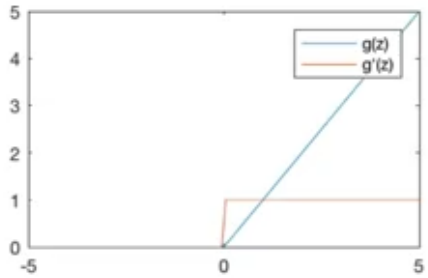
Hyperbolic Tangent



$$g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

$$g'(z) = 1 - g(z)^2$$

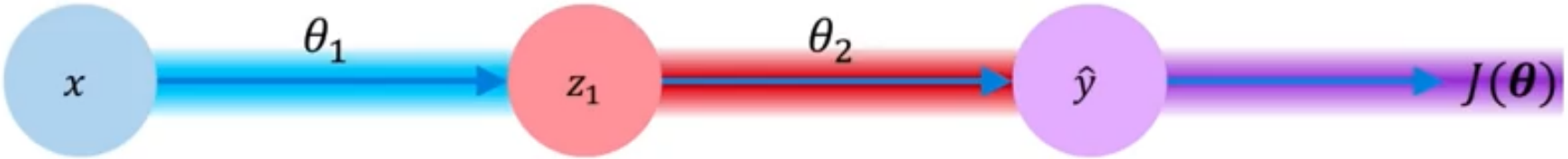
Rectified Linear Unit (ReLU)



$$g(z) = \max(0, z)$$

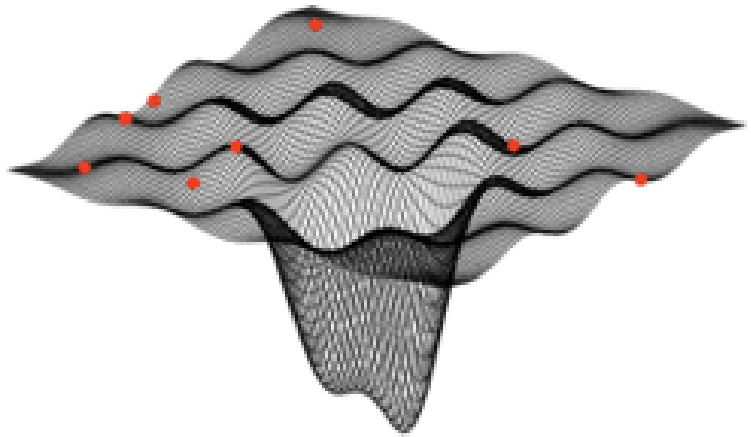
$$g'(z) = \begin{cases} 1, & z > 0 \\ 0, & \text{otherwise} \end{cases}$$

Back Propagation



$$\frac{\partial J(\boldsymbol{\theta})}{\partial \theta_1} = \frac{\partial J(\boldsymbol{\theta})}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1} * \frac{\partial z_1}{\partial \theta_1}$$

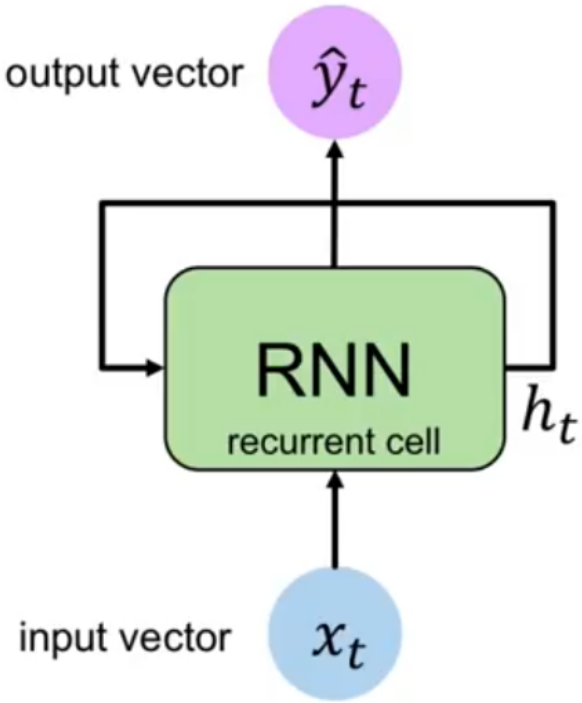
Figure: Loss Minimization



Recurrent Neural Network (RNN)

1. Dialogue systems, self-driving cars, robotic surgery, speech, among others require an explicit model of sequentiality or time – a combination of classifiers or regressors cannot provide these.
2. SVM, logistic regression, feedforward networks have proved very useful without explicitly modeling time. But the assumption of independence precludes modeling long range dependencies. Frames from video, snippets of audio, and words pulled from sentences – the independence assumption fails.
3. RNNs are connectionist models with the ability to selectively pass information across sequence steps while processing sequential data one element at a time. They can model input and/or output consisting of sequences of elements that are not independent. In other words can handle variable length sequences, track long term dependencies, maintain information about order, etc.

RNN



Apply a **recurrence relation** at every time step to process a sequence:

$$h_t = f_W(h_{t-1}, x_t)$$

cell state — function parameterized by W — old state

Figure: Same function and set of parameters are used in every step

Source - MIT 6.S191- Recurrent Neural Networks

RNN

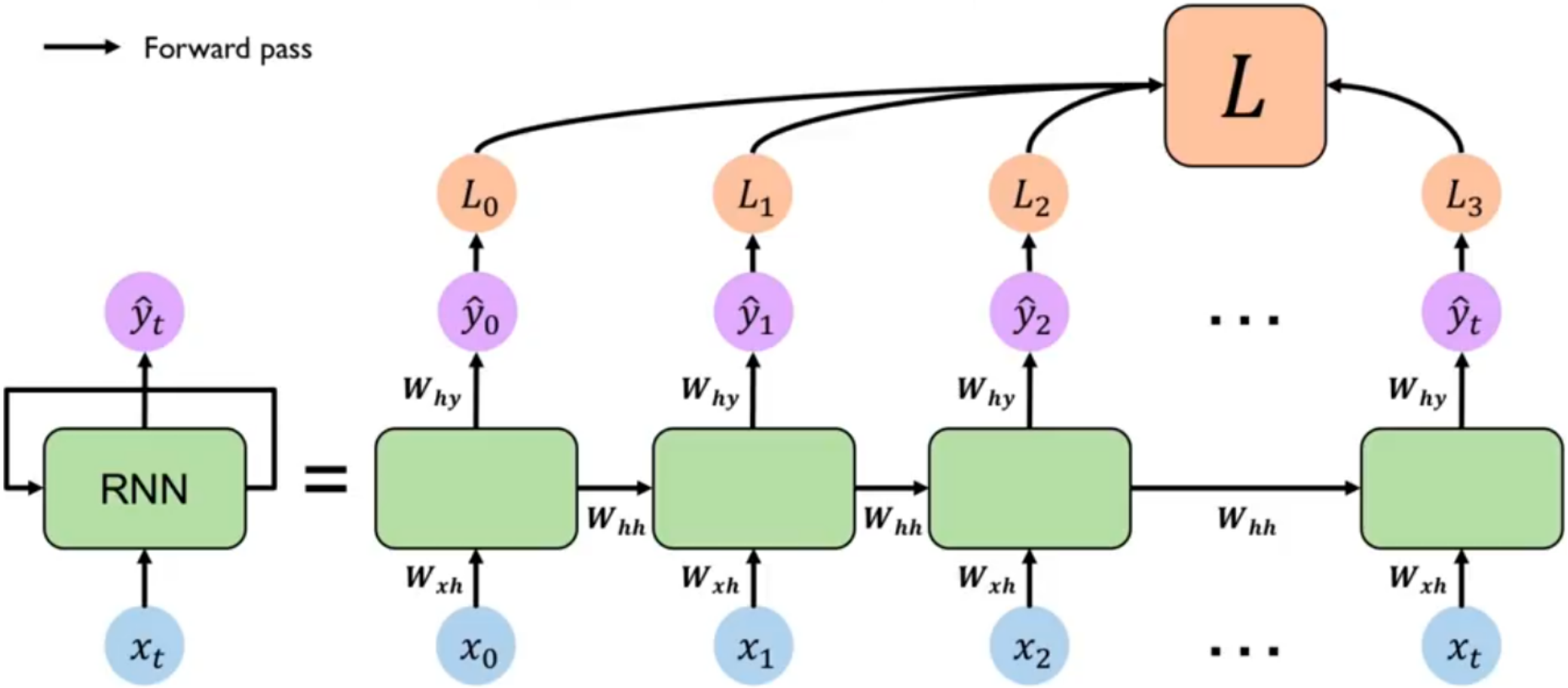


Figure: Computational Graph

RNN

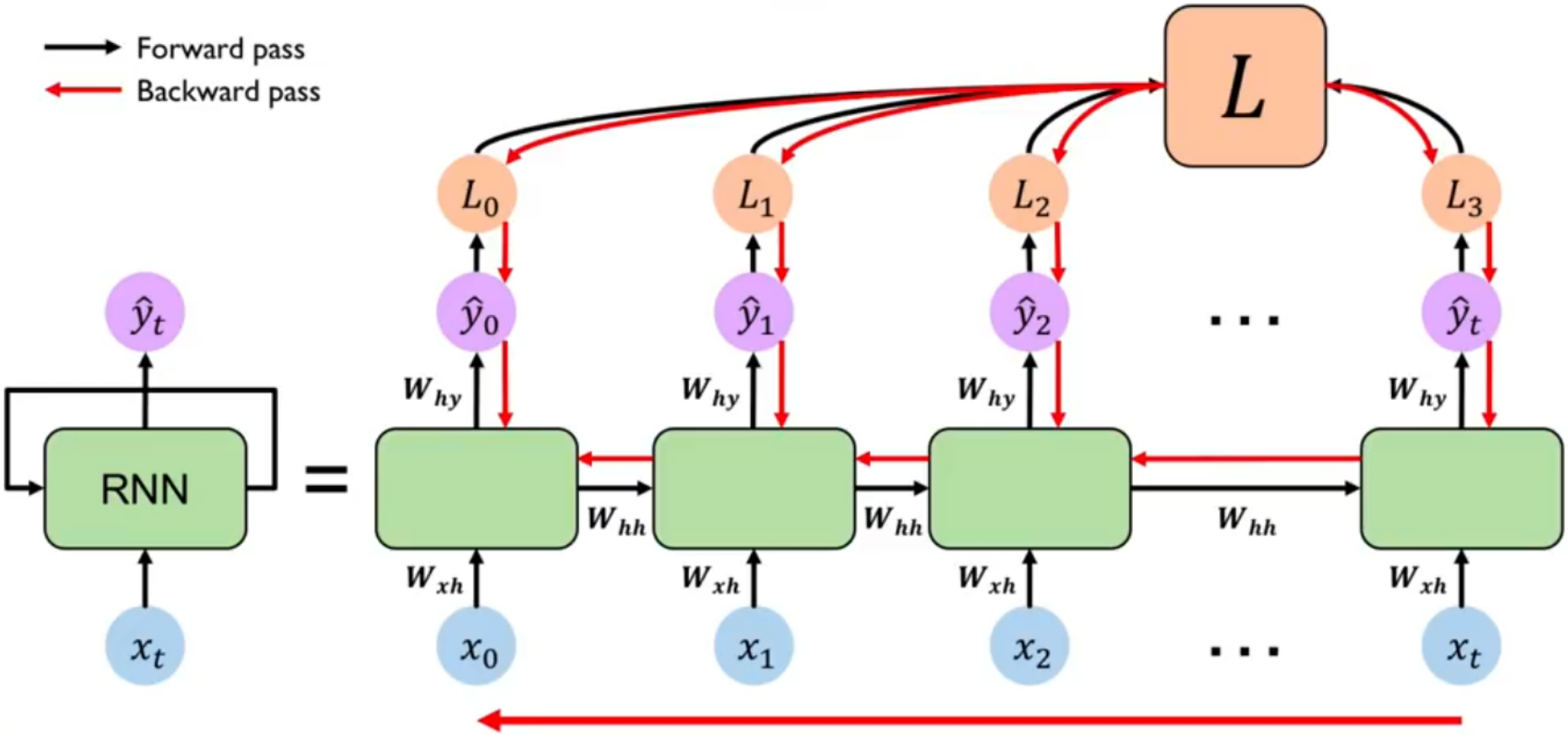


Figure: Backpropagation over time

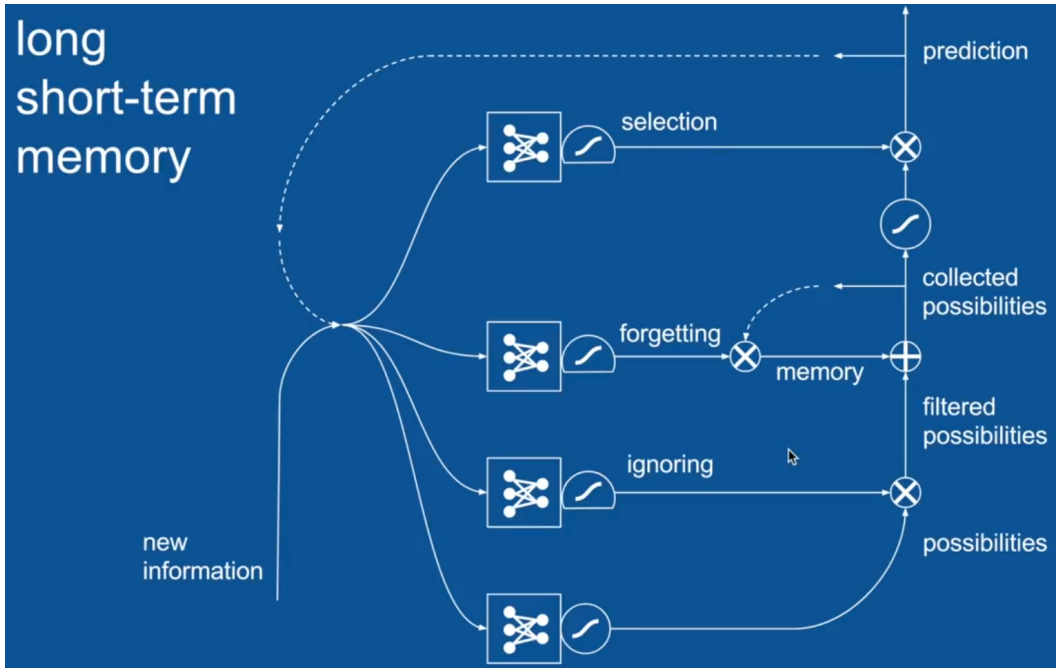
The Problem of Vanishing / Exploding Gradients

1. Learning with recurrent networks is challenging due to the difficulty of learning long-range dependencies, [Bengio et al. 1994], [Hochreiter et al, 2001].
2. Problems of vanishing and exploding gradients occur when backpropagating errors across many time steps.
3. Which of the two phenomena occurs depends on whether the weight of the recurrent edge $|wjj| > 1$ or $|wjj| < 1$ and on the activation function in the hidden node. For a sigmoid activation function, the vanishing gradient problem is more pressing, but with a rectified linear unit $max(0, x)$, it is easier to imagine the exploding gradient.

Long Short Term Memory (LSTM) - Hochreiter & Schmidhuber

A deep learning, sequential neural net that allows information to persist. It is a special type of Recurrent Neural Network which is capable of handling the vanishing gradient problem faced by traditional RNN.

1. Maintain a separate cell state
2. Use gates to (a) Forget (b) Selectively update and (c) Output
3. Backpropagation does not require matrix multiplication



Reinforcement Learning

1. $\hat{P}$ is sampled and is NOT predicted.
2. P_i is predicted and is a function of the weights and is differentiable with respect to the weights.
3. Cross Entropy Loss has to be minimized, this will lead to determining the weights.
4. Policy Gradients
   1. Discounted Rewards for the i^{th} move. R_i reduces geometrically over previous steps.
   $$loss_i = -R_i log(P_i^{chosen}) \tag{9}$$
   2. Normalized Rewards across a batch of moves
   $$R_i = \frac{R_i - \bar{R}}{stdev(R)} \tag{10}$$

Generalised Pretrained Transformer (GPT) - 2017

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

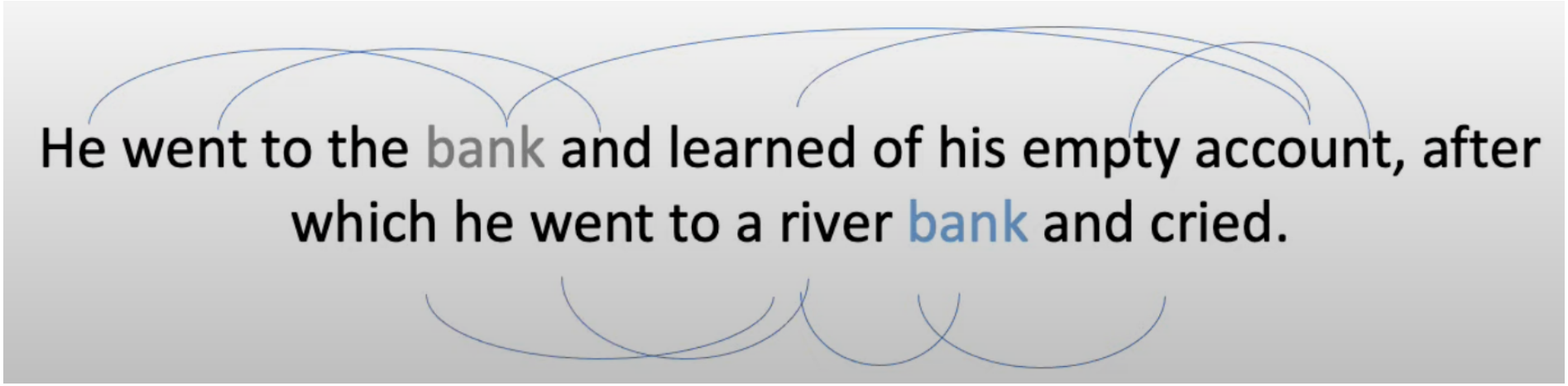
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com

"Neural Machine Translation by Jointly Learning to Align and Translate" by Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, 2014. Introduced concept of attention mechanism and laid the foundation for subsequent developments in NLP and DL, including the transformer architecture introduced in "Attention Is All You Need."

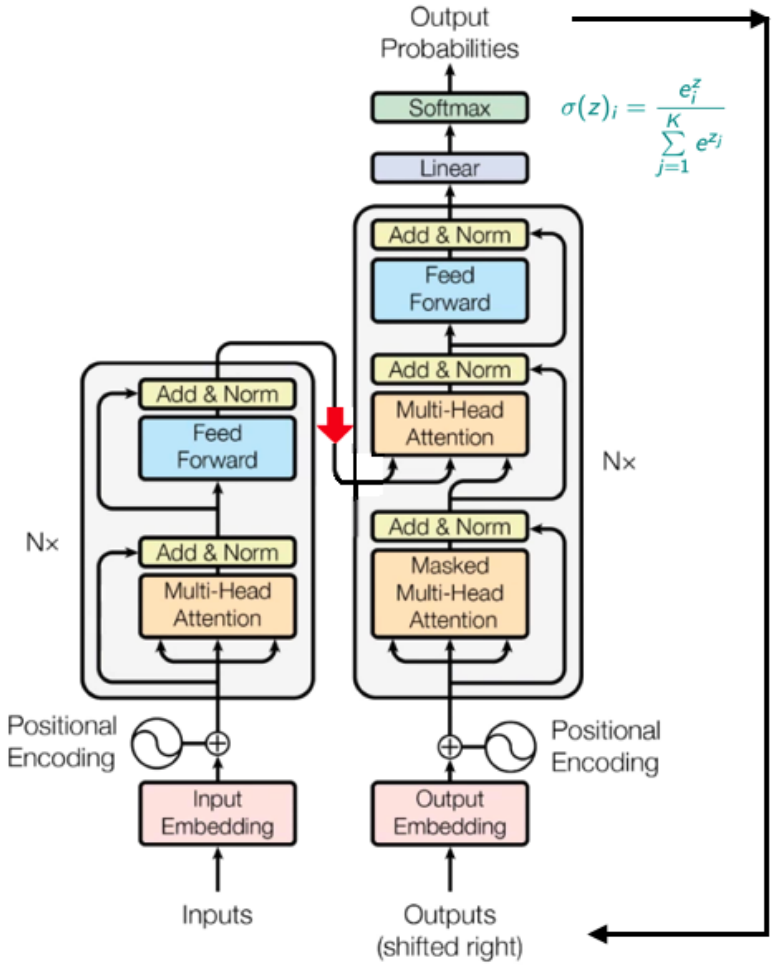
Attention



1. Attention mechanism has an infinite reference window
2. In contrast, Recurring Neural Network (RNN) has a short reference window, Long Short Term Memory (LSTM) has a longer window. RNN does not work, LSTM has limited capability.

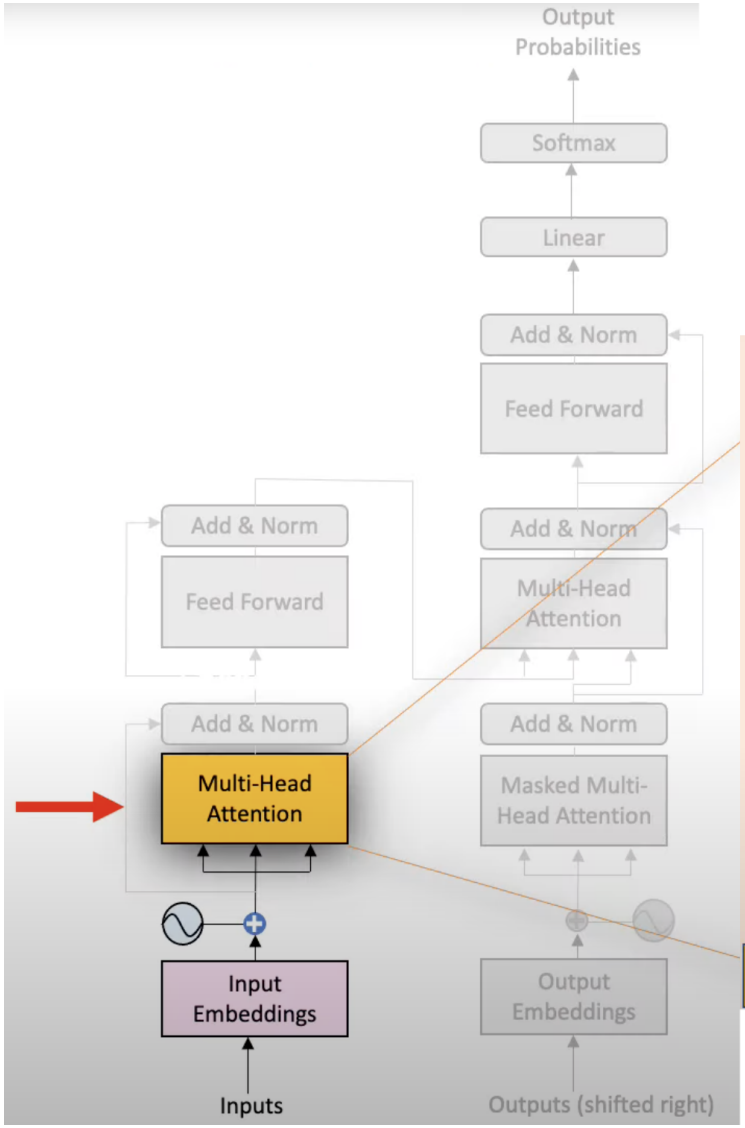
Key Concepts

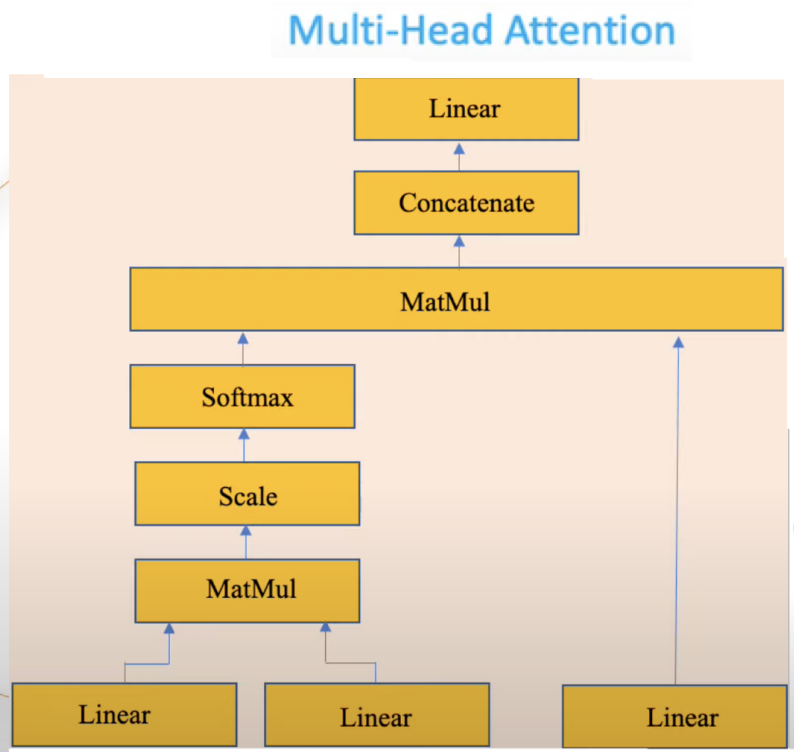
1. Encode, Positional encoding
2. Key, Query and Value $\rightarrow$ Attention Filter
3. Multi Head Attention
4. Information preservation, normalisation
5. Decode, Masked Attention
6. Potential Innovations
7. Conclusion

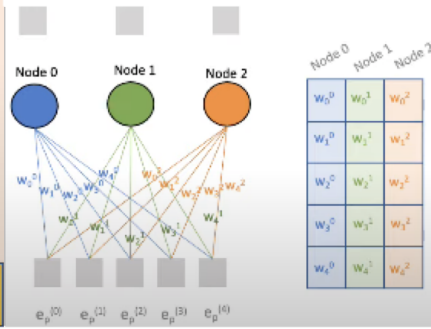


Decoders are autoregressive models; They are trained to predict the next token after reading the preceding ones

Multi Head Attention

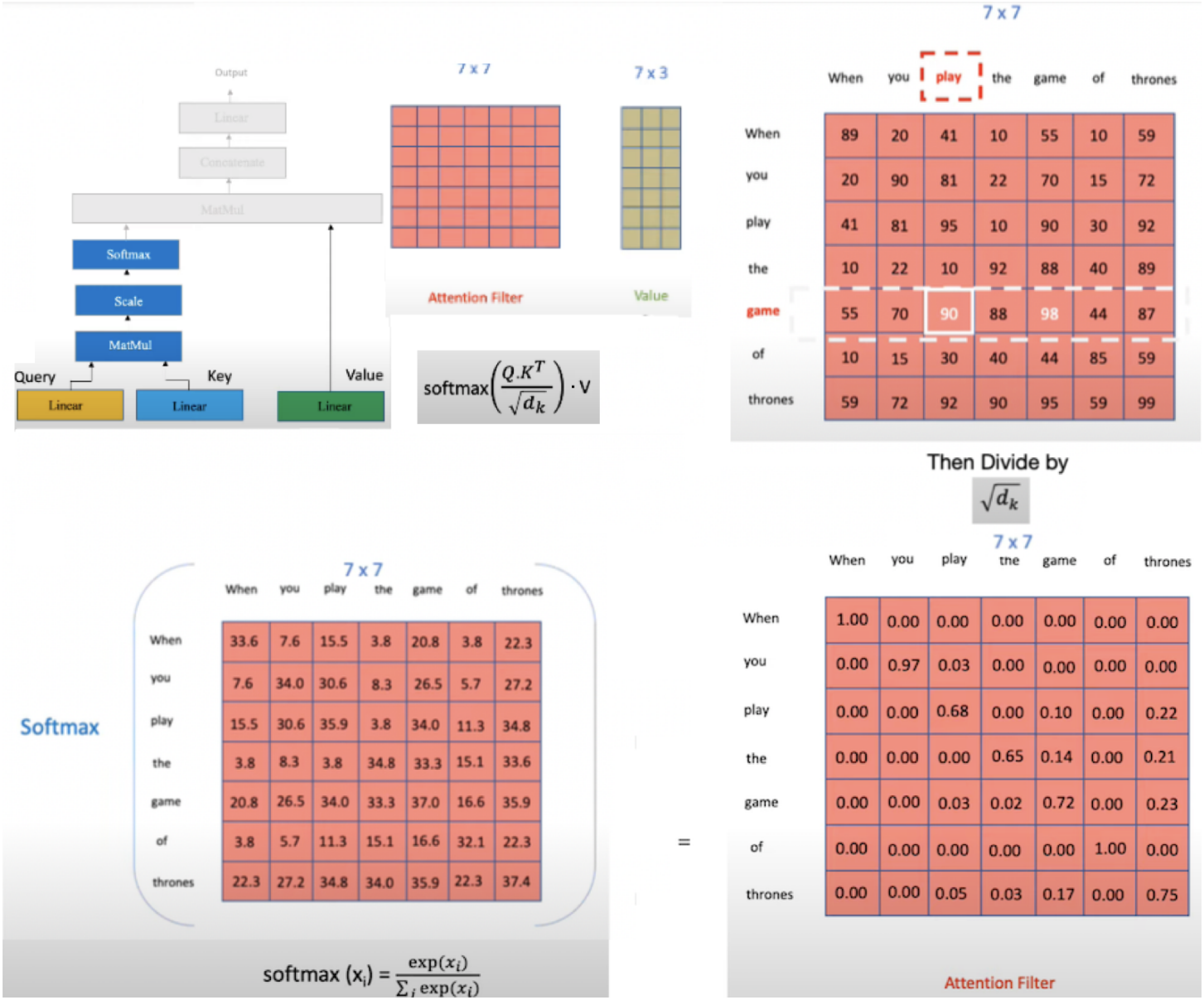




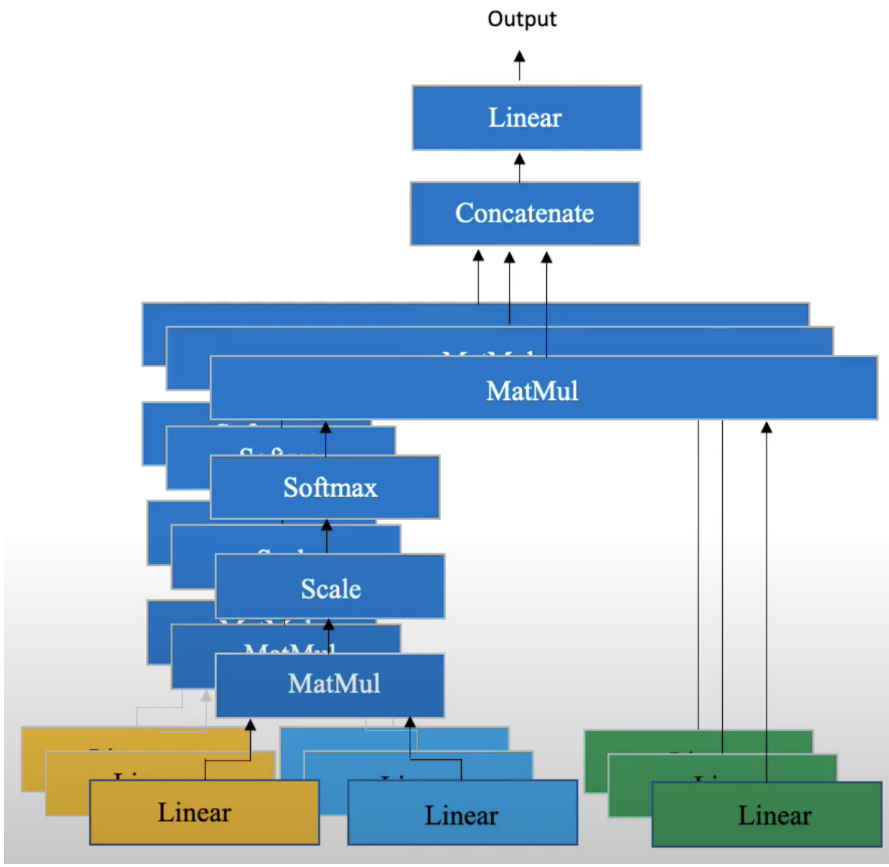


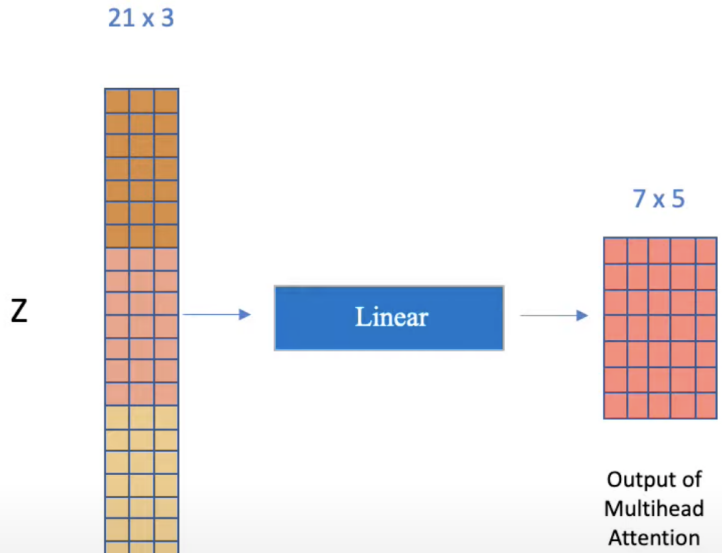
Linear Layer - No Activation Function

Attention Filter - filter out unnecessary information (noise)

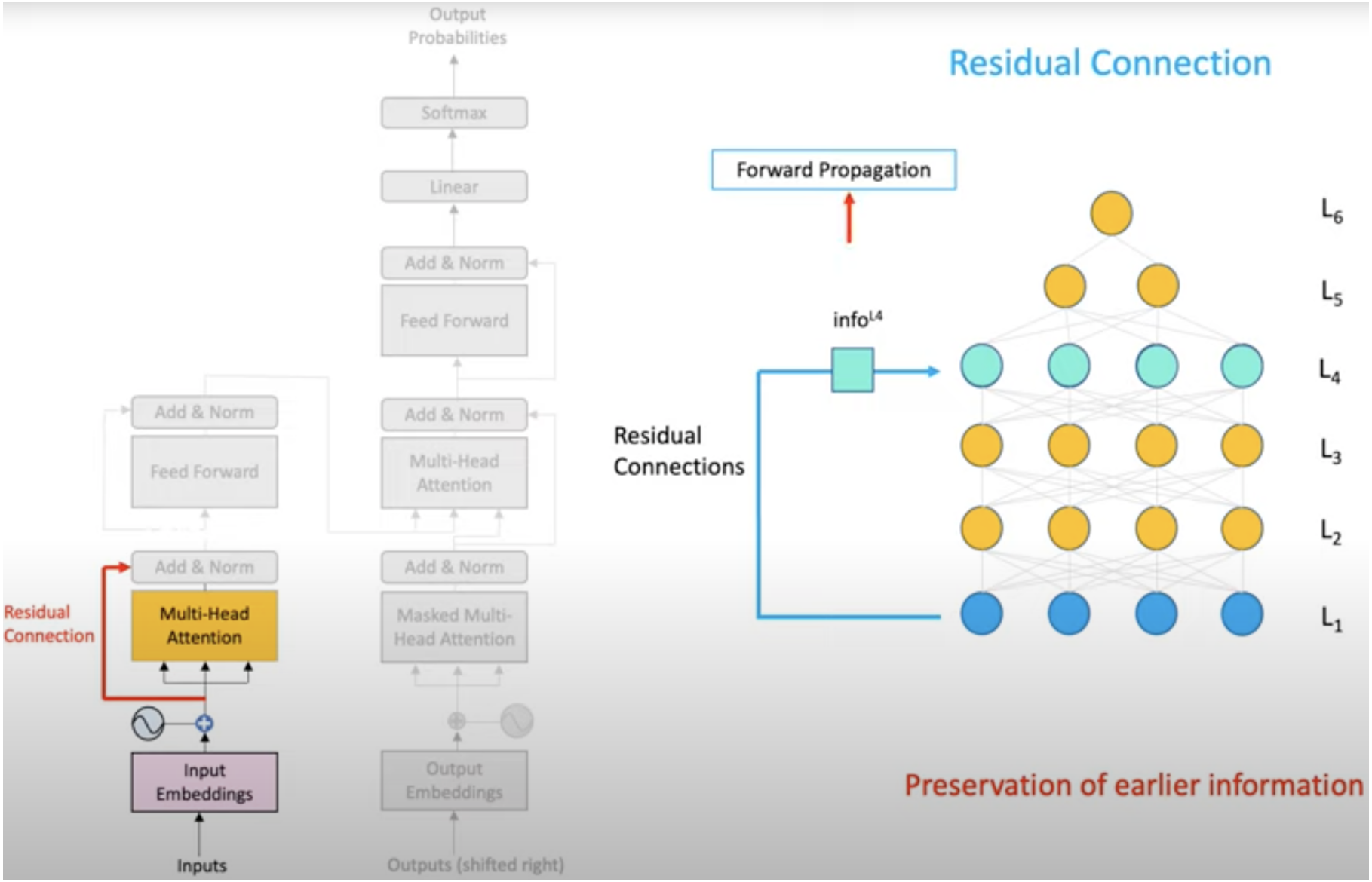


Multi Head Attention - Concatenation

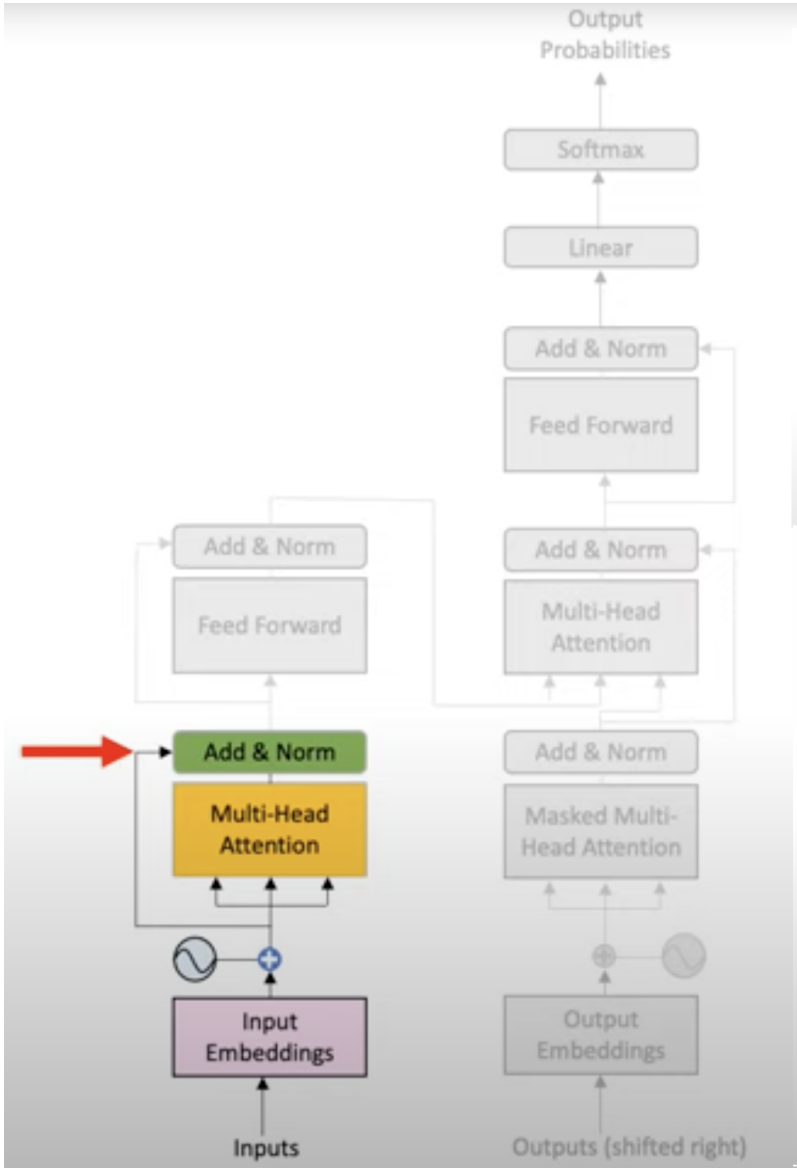


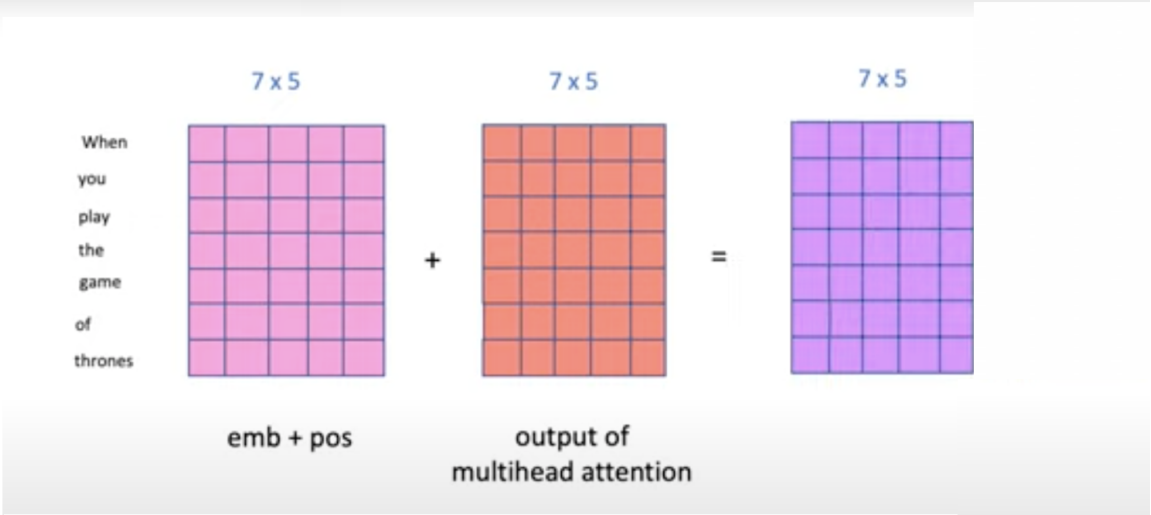


Information Preservation



Add (Preserve Information) & Normalisation



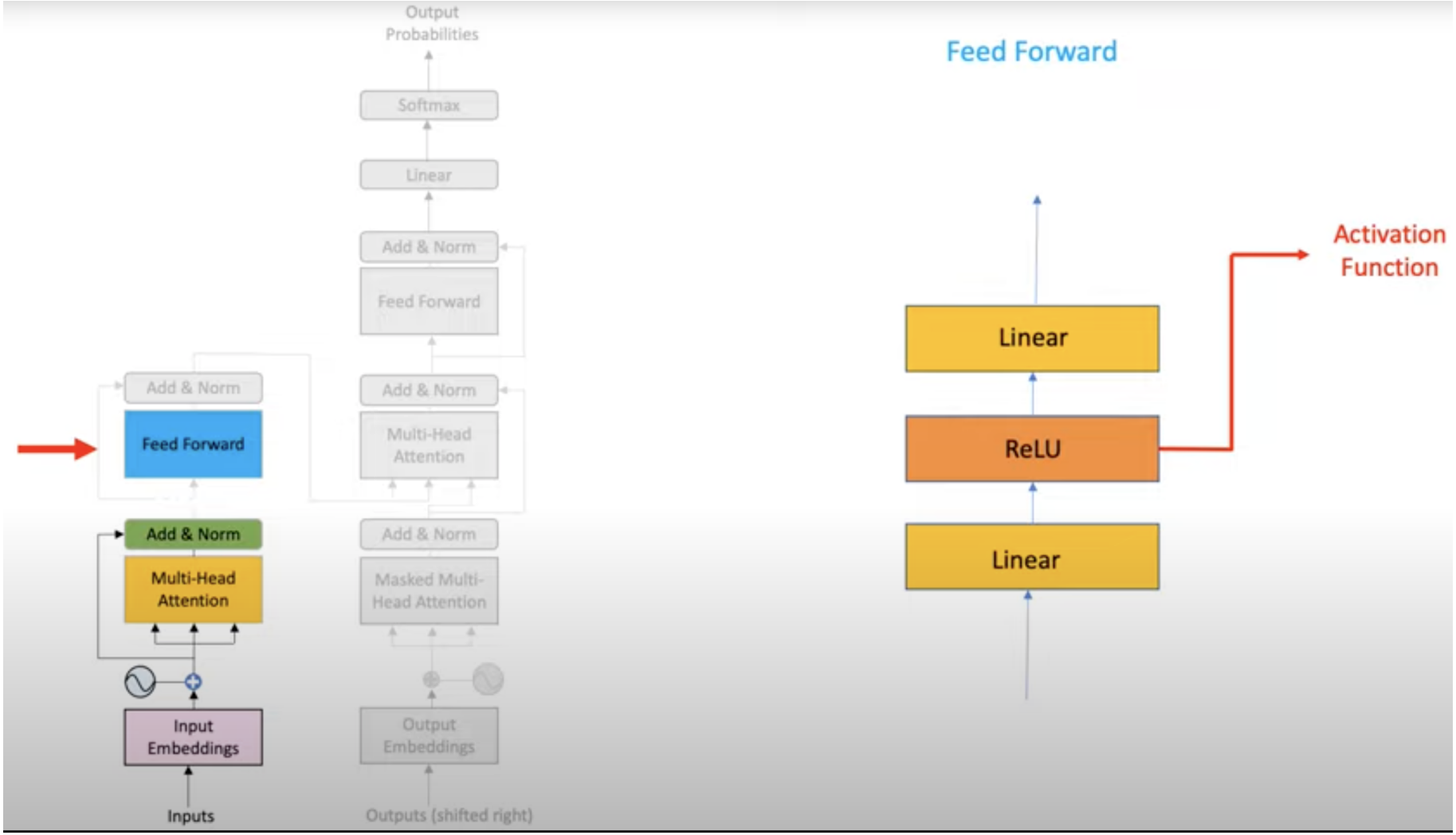


7 x 3

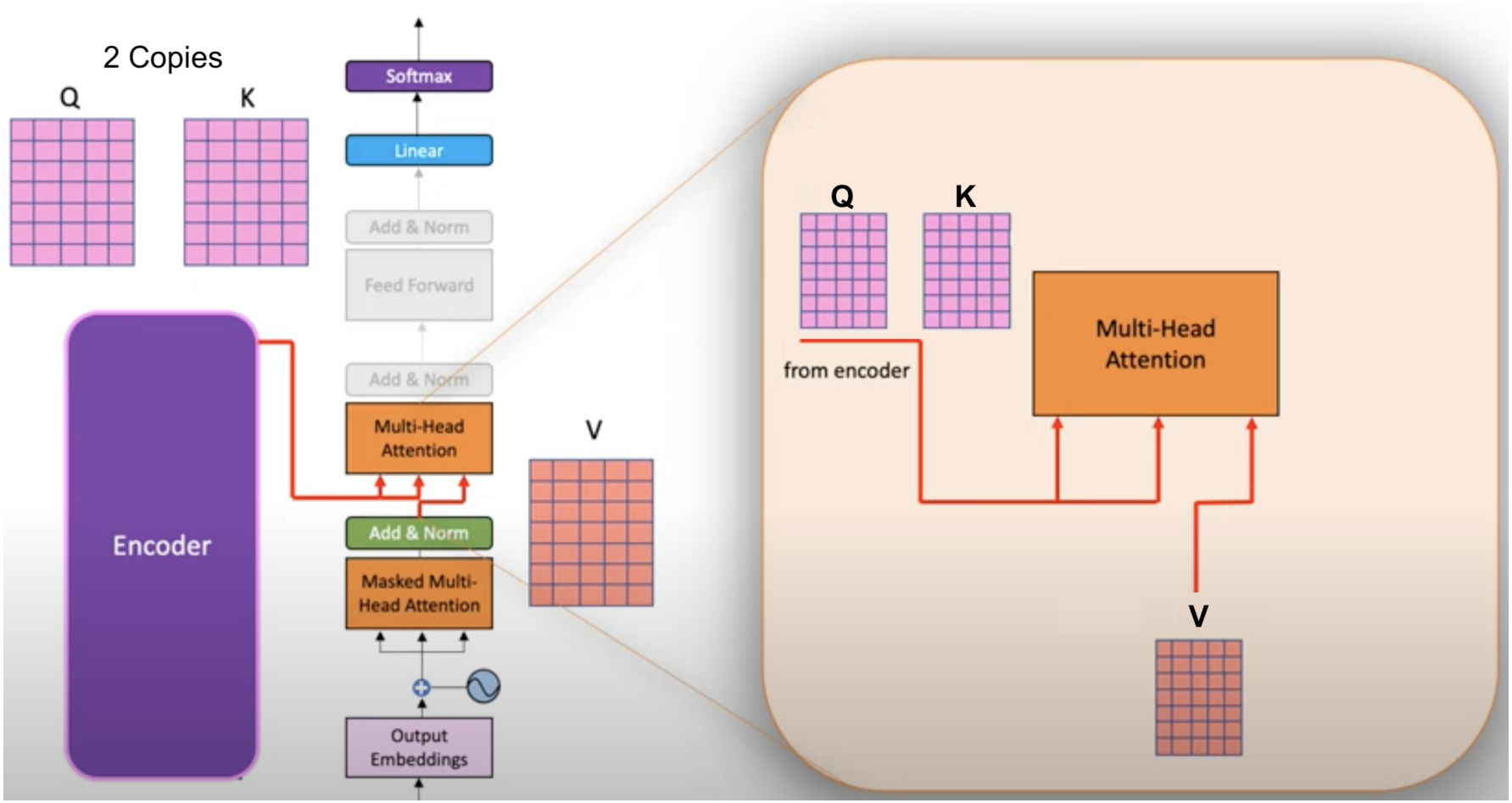
						mean (μ)	std (σ)
x_0 = When	0.98	1.28	0.41	0.27	0.41	0.67	0.44
x_1 = you	0.52	0.01	2.06	0.27	0.33	0.64	0.82
x_2 = play	2.22	0.27	0.10	0.41	2.06	1.01	1.04
x_3 = the	0.99	1.00	0.11	0.27	0.33	0.54	0.42
x_4 = game	0.52	0.01	0.33	2.06	0.52	0.69	0.79
x_5 = of	0.10	2.06	0.73	0.27	0.41	0.71	0.79
x_6 = thrones	0.33	0.01	0.13	0.27	1.28	0.40	0.51
	f_0	f_1	f_2	f_3	f_4		

$$x_0 = \frac{0.98 - 0.67}{\sqrt{0.44^2 + 0.0001}}$$

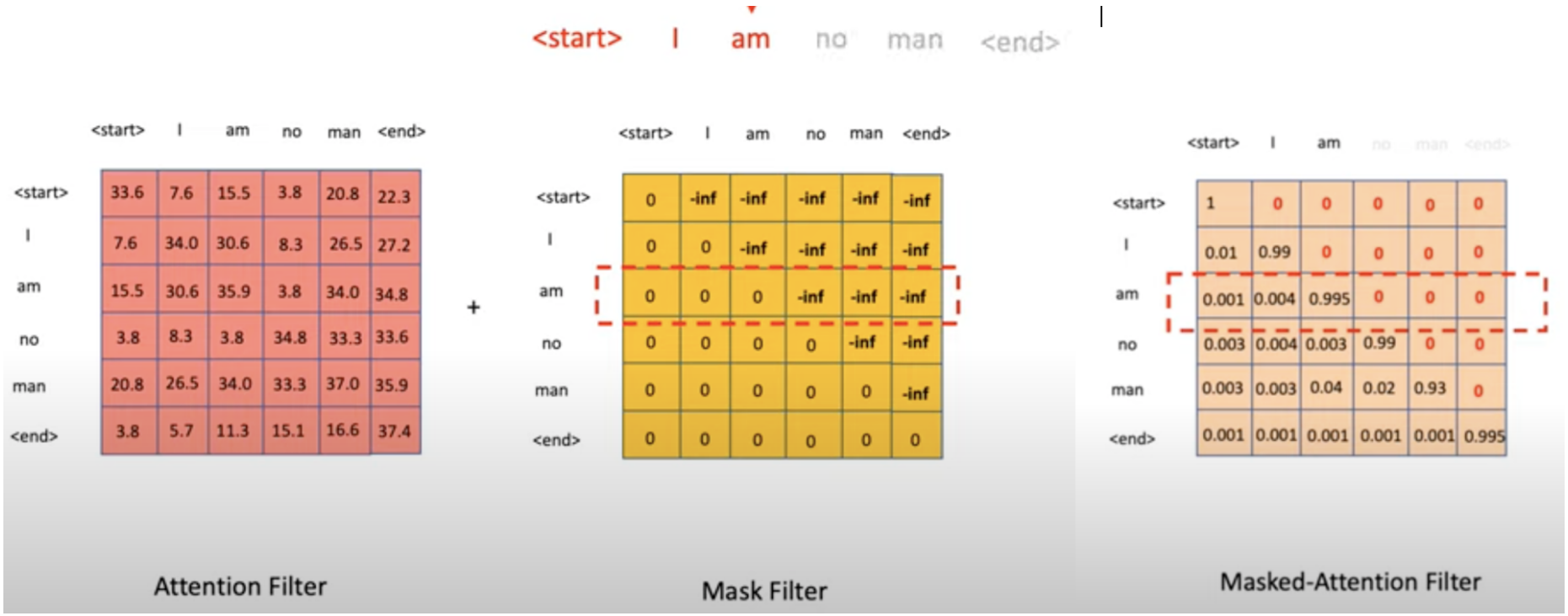
Feed Forward



Decoder Layer



Masking


<start> I am no man <end>
<start> I am no man <end>
<start> 33.6 7.6 15.5 3.8 20.8 22.3
I 7.6 34.0 30.6 8.3 26.5 27.2
am 15.5 30.6 35.9 3.8 34.0 34.8
no 3.8 8.3 3.8 34.8 33.3 33.6
man 20.8 26.5 34.0 33.3 37.0 35.9
<end> 3.8 5.7 11.3 15.1 16.6 37.4
+
<start> I am no man <end>
<start> 0 -inf -inf -inf -inf -inf
I 0 0 -inf -inf -inf -inf
am 0 0 0 -inf -inf -inf
no 0 0 0 0 -inf -inf
man 0 0 0 0 0 -inf
<end> 0 0 0 0 0 0
<start> I am no man <end>
<start> 1 0 0 0 0 0
I 0.01 0.99 0 0 0 0
am 0.001 0.004 0.995 0 0 0
no 0.003 0.004 0.003 0.99 0 0
man 0.003 0.003 0.04 0.02 0.93 0
<end> 0.001 0.001 0.001 0.001 0.001 0.995
Attention Filter
Mask Filter
Masked-Attention Filter

Conclusion

1. GPT does not replicate human writing or speaking processes. Although they imitate human writing, any apparent cleverness primarily arises from our inclination to attribute human characteristics to non-human entities (anthropomorphization).
2. LLMs are essentially establishing statistical connections among vectors representing words and more extended grammatical structures. Each word within a sentence is linked to the subsequent word in the sequence with an associated probability.
3. This diverges significantly from human cognitive processes, where we employ word meanings to construct intricate and precise structures of "meanings" and definitions. For LLMs, definitions are confined to statistical interrelations among intricate vectors encompassing words, sentences, and more extensive grammatical constructs. LLMs cannot innovate, not just yet.
4. That said, LLMs are impressive and have massive use cases.

LLM pitfalls?

1. Large language models can do jaw-dropping things. But nobody knows exactly why.
2. Figuring it out is one of the biggest scientific puzzles of our time and a crucial step towards controlling more powerful future models.
3. Burda and Edwards teamed up with colleagues to study the phenomenon. They found that in certain cases, models could seemingly fail to learn a task and then all of a sudden just get it, as if a lightbulb had switched on. This wasn't how deep learning was supposed to work. They called the behavior.
4. According to classical statistics, the bigger a model gets, the more prone it is to overfitting. Double descent is a phenomenon observed in machine learning where the performance of a model initially decreases, then increases, and finally decreases again as the model complexity or dataset size increases. This phenomenon contradicts the traditional bias-variance tradeoff, where increasing model complexity typically leads to a decrease in bias and an increase in

Apps powered by LLMs

1. The basic building block of LangChain is the LLM, which takes in text and generates more text. Lots of LLM providers ... OpenAI, Anthropic, Cohere, Hugging Face, Llama-cpp, etc.
2. LLM applications do not pass user input directly into an LLM. Usually they will add the user input to a larger piece of text, called a prompt template, that provides additional context on the specific task at hand. LangChain facilitates prompt management and optimization.
3. Combine LLMs and Prompts in multi-step workflows.
4. Combine LLM with your own text data. Large number of document loaders available - JSON, AWS S3, Azure Blob Storage File, Confluence, CSV, Email, PDF, Excel, MS Word/Excel, G Drive, ...
5. Agents involve an LLM making decisions about which Actions to take, taking that action, seeing an Observation, and repeating that until done. Zero-shot means the agent functions on the current action only - it has no memory. It uses the ReAct framework to decide which tool to use, based solely on the tool's description.

Questions?

Thank You!