World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(RESEARCH ARTICLE)

Check for updates

Deep Learning

Ganesh Viswanathan [1,*], Gaurav Samdani [1], Yawal Dixit [2] and Ranjith Gopalan [2]

[1] *Department of Data science and Business Analytics, UNC Charlotte, Charlotte, NC, United states.*
[2] *Principal Consultant, Cognizant Technologies Corp, Charlotte, NC, United states.*

World Journal of Advanced Engineering Technology and Sciences, 2025, 14(03), 512-527

Publication history: Received on 11 February 2025; revised on 28 March 2025; accepted on 30 March 2025

Article DOI: https://doi.org/10.30574/wjaets.2025.14.3.0149

Abstract

Deep learning has revolutionized artificial intelligence by enabling machines to learn complex patterns from vast amounts of data. This white paper explores the fundamental principles of deep learning, including neural network architectures, training methodologies, and key advancements such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models. We discuss applications across various domains, including computer vision, natural language processing, healthcare, and finance, highlighting real-world use cases and breakthroughs. Additionally, we examine the challenges of deep learning, such as data requirements, model interpretability, and computational constraints, along with emerging trends in model efficiency and responsible AI. This paper aims to provide insights into the current state of deep learning and its future trajectory, helping researchers and industry professionals navigate the rapidly evolving AI landscape.

Keywords: Deep Learning; Neural Networks; Convolutional Neural Networks (CNNs); Recurrent Neural Networks (RNNs); Transformer Models; Natural Language Processing (NLP); Computer Vision; Model Interpretability; Computational Constraints; Model Efficiency; Responsible AI; Training Methodologies

1. Introduction

Deep learning is a subset of machine learning that utilizes artificial neural networks to model and solve complex problems. Unlike traditional machine learning algorithms, deep learning models leverage multiple layers of abstraction to learn from large datasets. The rise of deep learning has been fueled by advances in computational power, availability of massive datasets, and improvements in algorithms. This paper aims to provide a comprehensive understanding of deep learning, focusing on its core principles, architectures, training strategies, applications, and challenges.

1.1. History and Evolution of Deep Learning

AI, machine learning and deep learning are frequently associated but have distinct differences. AI is designed so that it behaves more or less like a human being, capable of solving problems like people. Meanwhile, ML is a subset of AI that uses statistical algorithms to train computer systems so they can make independent predictions and decisions based on historical data when exposed to new inputs. DL is a subset of machine learning and AI(Fig 1).

While ML and AI encompass a wide range of algorithms and techniques for training computer systems, deep learning is a specific approach to machine learning that is based on artificial neural networks. It is widely used for understanding texts and images for natural language processing (NLP) and computer vision-related tasks.

* Corresponding author: Ganesh Viswanathan

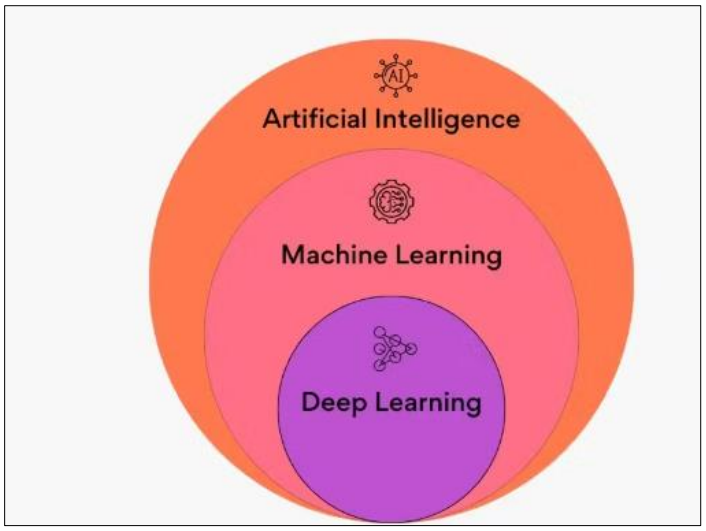

Figure 1 Deep learning a subset of AI

The origins of deep learning can be traced back to the 1940s and 1950s, when early computational models of neural activity were proposed. The Perceptron, introduced by Frank Rosenblatt in 1958, was one of the first artificial neural networks designed to mimic the learning process of the human brain. However, due to its limitations in solving non-linearly separable problems, interest in neural networks diminished for a few decades.

The resurgence of deep learning began in the 1980s with the development of backpropagation, a key algorithm that allows neural networks to learn by adjusting weights iteratively. During the 1990s and early 2000s, computational power and data availability continued to increase, leading to advancements in neural network architectures. The rise of deep learning as a dominant AI paradigm can be attributed to breakthroughs in hardware acceleration (such as GPUs), the availability of large-scale labeled datasets, and improvements in training algorithms.

2. Key Components of Deep Learning

Deep learning systems comprise several key components:

- **Neural Networks**: The fundamental building blocks of deep learning models, consisting of interconnected nodes or "neurons" that process information.

2.1. Understanding neural networks

- A neural network is a mathematical function that takes in inputs and produces outputs.
- A neural network is a computational graph through which multidimensional arrays flow.
- A neural network is made up of layers, each of which can be thought of as having several “neurons.”
- A neural network is a universal function approximator that can in theory represent the solution to any supervised learning problem.

- **Training Data**: Large amounts of labeled or unlabeled data that enable deep learning models to generalize patterns and make predictions(Fig 2).

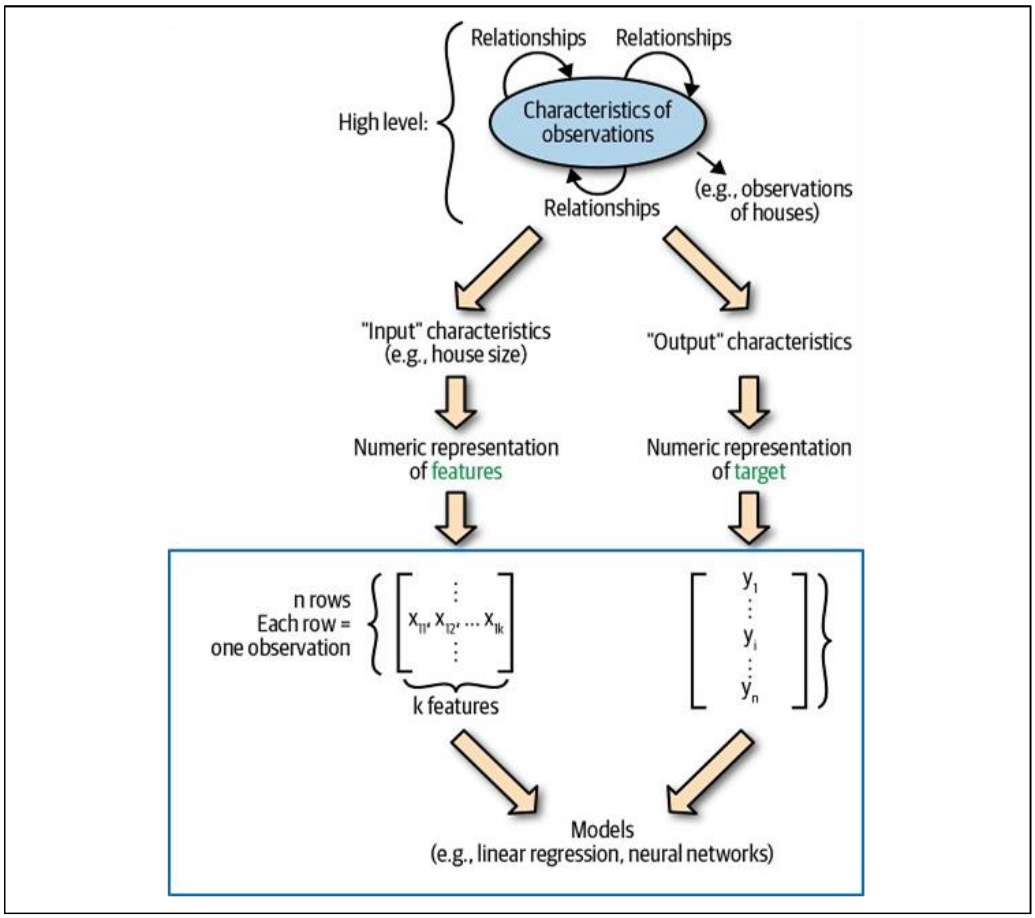

Figure 2 Role data plays in ML models

- **Activation Functions and weights**: Non-linear functions such as ReLU and sigmoid that introduce complexity into neural networks, allowing them to model intricate relationships in data(Fib 3).

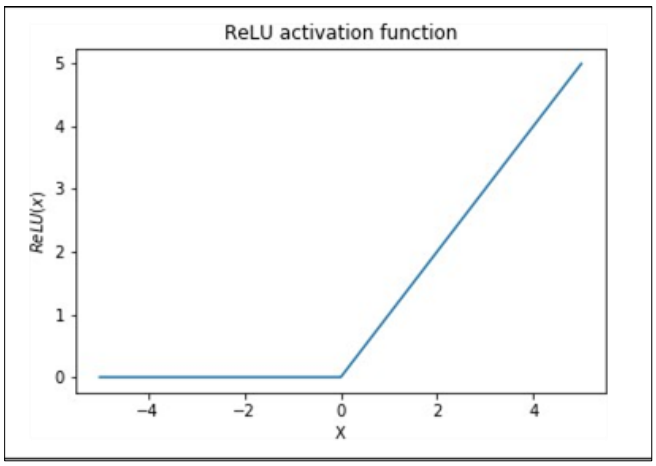

Figure 3 Building relations in data

Activation functions define the activation of a neuron in a neural network and combination of several activation functions define a neural network

- **Loss Functions**: Measures of how far the model's predictions deviate from actual values, guiding the training process.

3. Depictions of Neural Networks and deep learning

Neural networks are the backbone of deep learning. A neural network has the following fundamentals.

3.1. Functions

A function is a mathematical concept that takes an input and produces an output based on a specific rule. For example, the function (f(x) = x^2) takes an input (x) and produces an output (x^2). Functions can be represented in various ways, including mathematical notation, diagrams, and code. In deep learning, functions are used to transform data and perform calculations on it.

Math

Here are two examples of functions, described in mathematical notation:

$f_1(x) = x^2$

$f_2(x) = max(x, 0)$

This notation says that the functions, which we arbitrarily call f_1 and f_2, take in a number x as input and transform it into either x^2 (in the first case) or $max(x, 0)$ (in the second case).

Diagrams

One way of depicting functions is to:

- Draw an x-y plane (where x refers to the horizontal axis and y refers to the vertical axis).
- Plot a bunch of points, where the x-coordinates of the points are (usually evenly spaced) inputs of the function over some range, and the y-coordinates are the outputs of the function over that range.
- Connect these plotted points.

This was first done by the French philosopher René Descartes, and it is extremely useful in many areas of mathematics, in particular calculus. Figure 4 shows the plot of these two functions.

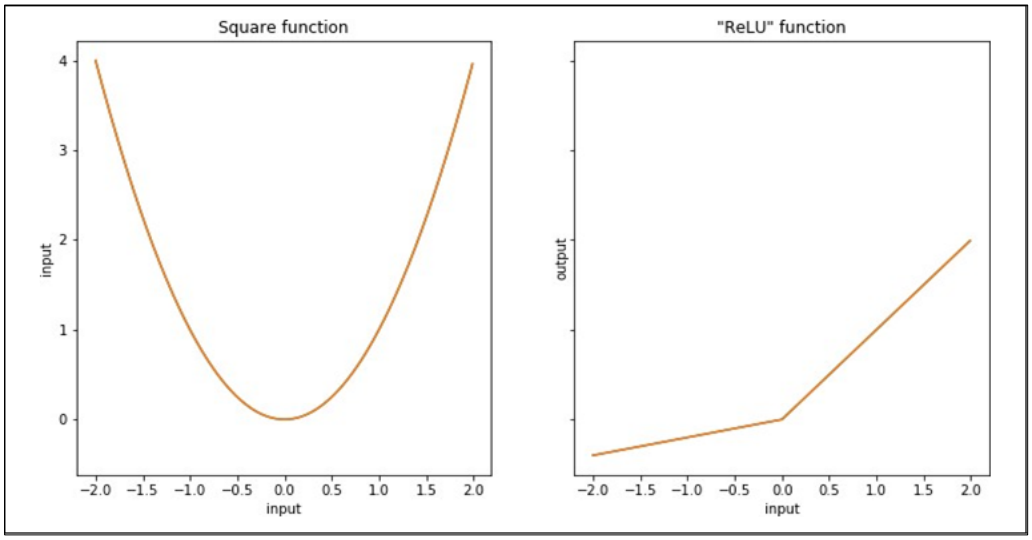

Figure 4 Two continuous, mostly differentiable functions

However, there is another way to depict functions that aren't as useful when learning calculus but that will be very useful for us when thinking about deep learning models. We can think of functions as boxes that take in numbers as

input and produce numbers as output, like mini factories that have their own internal rules for what happens to the input. Figure 5 shows both these functions described as general rules and how they operate on specific inputs.

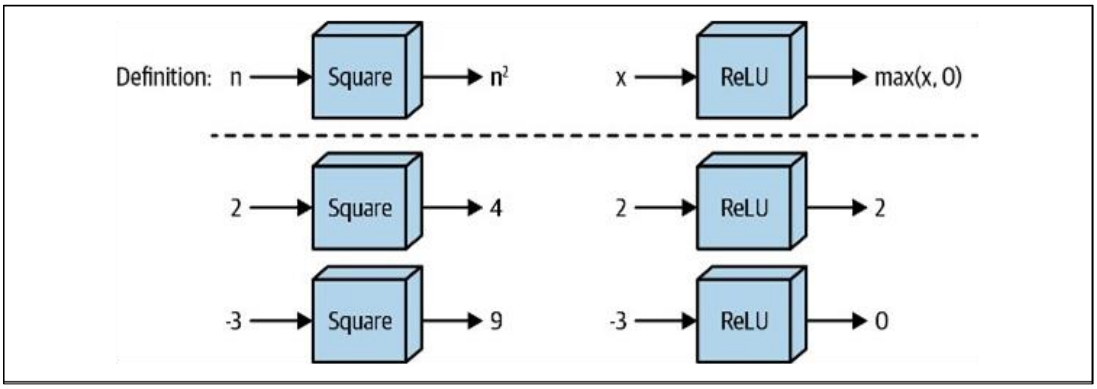

Figure 5 Another way of looking at these functions

3.2. Derivatives

Derivatives are a fundamental concept in calculus that measure the rate of change of a function's output with respect to its input. The derivative of a function at a point is the slope of the tangent line to the function at that point. Mathematically, the derivative of a function (f) at a point (a) can be approximated as:

[f'(a) \approx \frac{f(a + \Delta) - f(a - \Delta)}{2\Delta}]

where (\Delta) is a small value. Derivatives are crucial in deep learning for optimizing neural networks, as they allow us to adjust the weights of the network to minimize the error in its predictions.

3.2.1. Diagrams

First, the familiar way: if we simply draw a tangent line to the Cartesian representation of the function *f*, the derivative of *f* at a point *a* is just the slope of this line at *a*. As with the mathematical descriptions in the prior subsection, there are two ways we can actually calculate the slope of this line. The first would be to use calculus to actually calculate the limit. The second would be to just take the slope of the line connecting *f* at $a - 0.001$ and $a + 0.001$. The latter method is depicted in Figure 6 and should be familiar to anyone who has taken calculus.

3.2.2. Derivatives

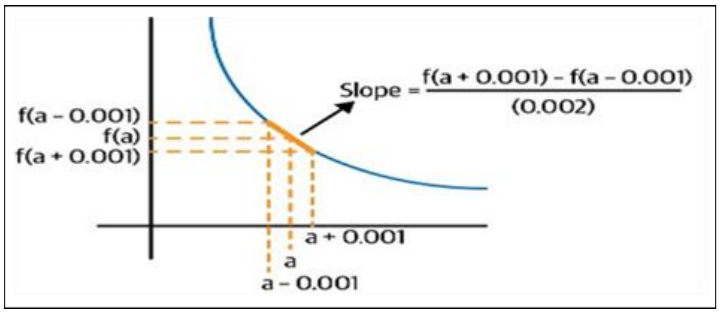

Figure 6 Derivatives as slopes

As we saw in the prior section, another way of thinking of functions is as minifactories. Now think of the inputs to those factories being connected to the outputs by a string. The derivative is equal to the answer to this question: if we pull up on the input to the function *a* by some very small amount—or, to account for the fact that the function may be asymmetric at *a*, pull down on *a* by some small amount—by what multiple of this small amount will the output change, given the inner workings of the factory? This is depicted in figure 7

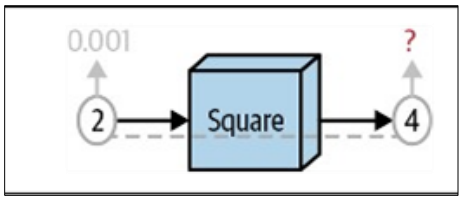

Figure 7 Another way of visualizing derivatives

This second representation will turn out to be more important than the first one for understanding deep learning.

3.3. Nested Functions

Nested functions are functions that are combined in such a way that the output of one function becomes the input to another. This creates a composite function, where multiple functions are "nested" within each other. Mathematically, if we have two functions (f_1) and (f_2), a nested function can be represented as (f_2(f_1(x))). This means that we first apply (f_1) to (x), and then apply (f_2) to the result of (f_1(x)).

3.3.1. Diagram

The most natural way to represent a nested function is with the “minifactory” or “box” representation Fig 8 shows, an input goes into the first function, gets transformed, and comes out; then it goes into the second function and gets transformed again, and we get our final output.

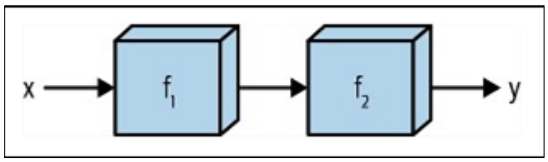

Figure 8 Nested functions, naturally

3.4. The Chain Rule

The chain rule is a fundamental theorem in calculus that allows us to compute the derivative of a composite function. In deep learning, this is particularly important because neural networks are essentially composite functions made up of multiple layers. The chain rule helps us understand how changes in the input of a function affect the output, which is crucial for training neural networks.

Mathematically, the chain rule states that for two functions (f) and (g), the derivative of the composite function (f(g(x))) with respect to (x) is given by:

$$[\frac{d}{dx} f(g(x)) = f'(g(x)) \cdot g'(x)]$$

In other words, we first take the derivative of the outer function (f) with respect to its input (g(x)), and then multiply it by the derivative of the inner function (g) with respect to (x).

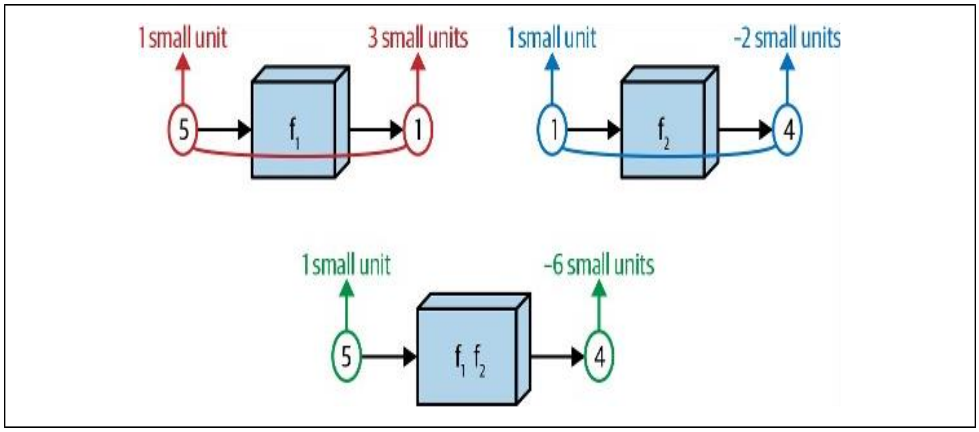

Figure 9 One dimensional Representation of a neural network

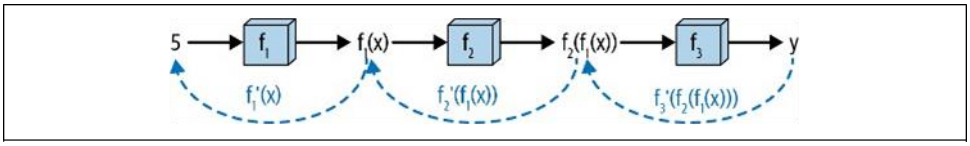

Figure 10 Chaining of one-dimensional neural network

The above diagrams just provide a one dimensional representation of a neural network. However, in practical applications all the inputs are n dimensional vectors representing several thousand data points. Thus a neural network with weights can be represented as

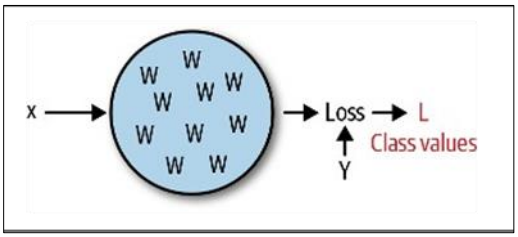

Figure 11 Neural Network with weights

A simple way to think of a neural network with weights or a neuron

The neural network thus consists of layers of interconnected neurons that process input data and generate predictions.

- **Input Layer**: Accepts raw data.
- **Hidden Layers**: Extract hierarchical features.
- **Output Layer**: Produces final predictions.

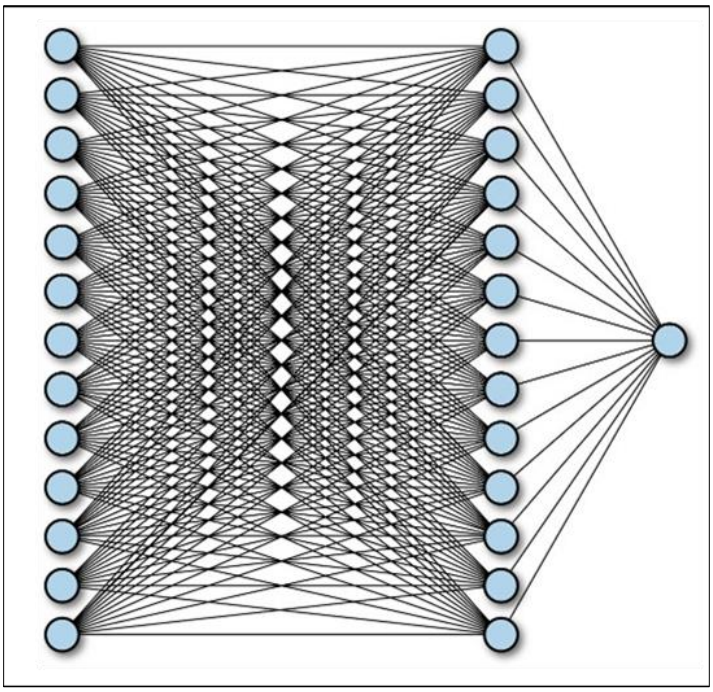

Figure 12 Pictorial representation of a neural network with several inputs, hidden layers and output layers

Neural Networks depicted in form of layers

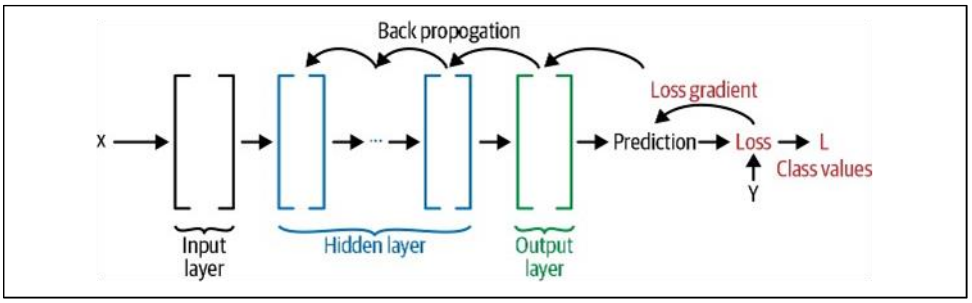

Figure 13 Deep learning models are neural networks with more than one hidden layer

4. Training Neural Networks

Training involves optimizing the weights of connections using backpropagation and gradient descent:

- **Forward Pass**: Computes outputs from inputs.
- **Loss Calculation**: Measures prediction error.
- **Backward Pass**: Adjusts weights using derivatives.
- **Optimization**: Updates parameters using techniques like stochastic gradient descent (SGD).

5. Types of neural network

5.1. Advanced Architectures

Neural networks have advanced leaps and bounds over the past few years due to increase in capability of underlying hardware. This has led to the development of several advanced architectures as below.

Convolutional Neural Networks (CNNs)

CNNs are designed for image processing. They use:

Convolutional Layers: Apply filters to detect features.

Pooling Layers: Reduce dimensionality.

Fully Connected Layers: Combine extracted features for classification.

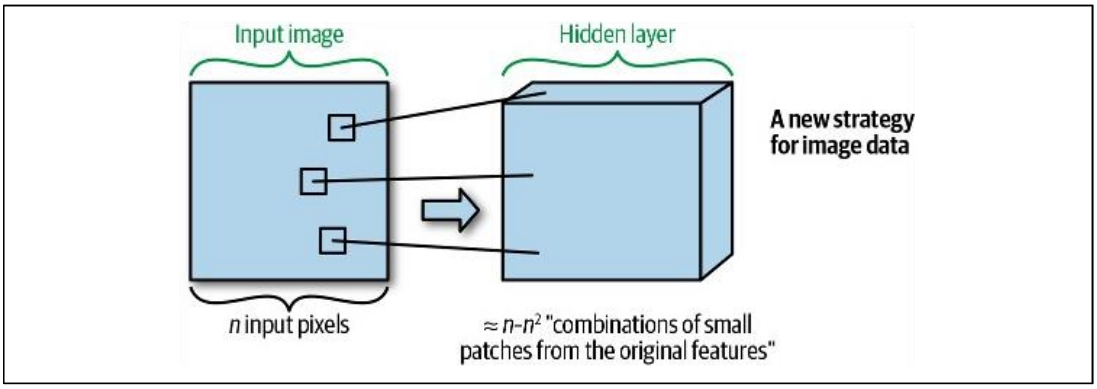

Figure 14 With image data we can define each learned feature to be a function of a small patch of data

Convolutional neural networks differ from regular neural networks in that they create an order of magnitude more features, and in that each feature is a function of just a small patch from the input image(refer to fig 13). Now we can get more specific: starting with *n* input pixels, the convolution operation just described will create *n* output features, one for each location in the input image. What actually happens in a convolutional Layer in a neural network goes one step further: there, we'll create *f sets* of *n* features, *each* with a corresponding (initially random) set of weights defining a visual pattern whose detection at each location in the input image will be captured in the feature map. These *f* feature maps will be created via *f* convolution operations refer to illustrations as below

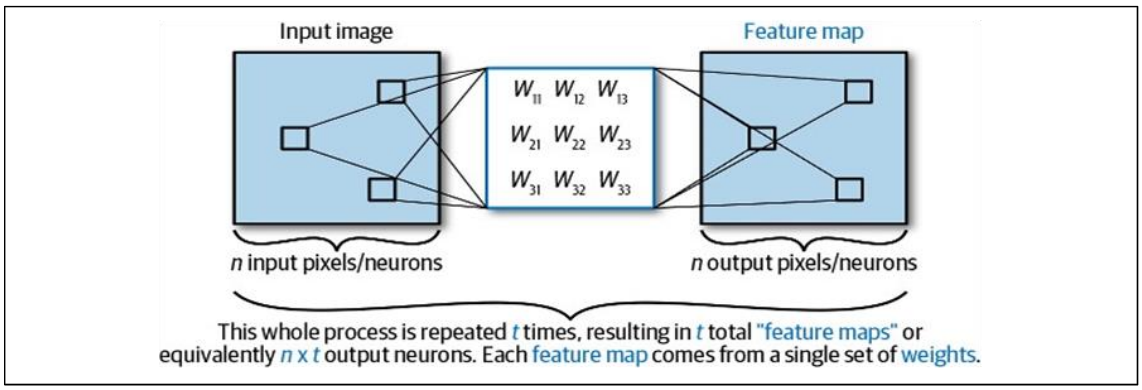

Figure 15 Feature map from single set of weights

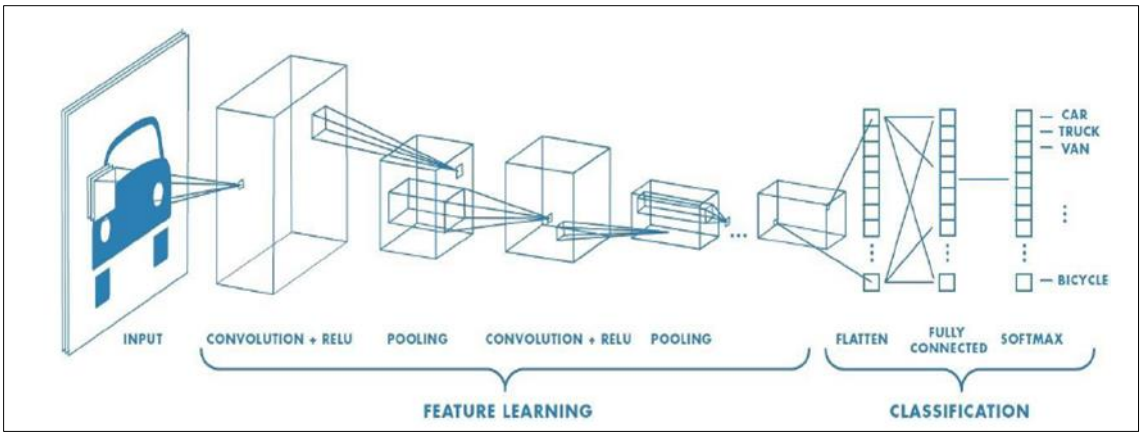

Figure 16 Practical applications of Neural network

5.2. Recurrent Neural Networks (RNNs)

While conventional neural networks are great at retaining patterns where adjacent data points are independent observations they don't do well against ordered or dependent observations. This can happen commonly while dealing with time series or language data. Dealing with ordered or sequential data has lead to the rapid development of another class of neural network called as recurrent neural networks.

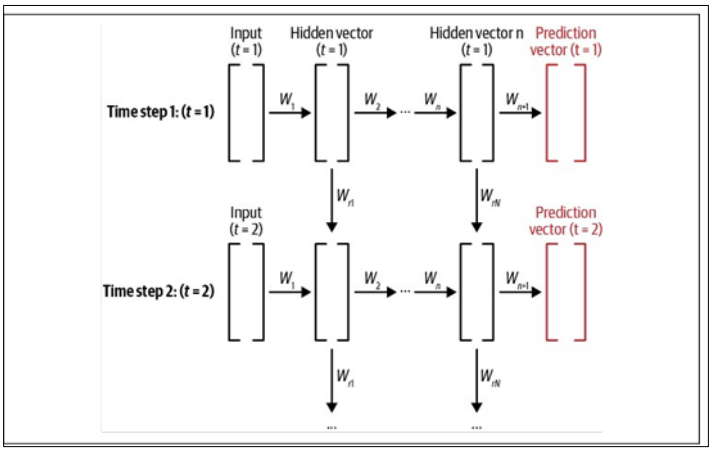

Figure 17 Recurrent neural Networks pass the representations of each layer forward into the next time step as shown below

RNNs handle sequential data and maintain memory through recurrent connections. Variants include:

- **Long Short-Term Memory (LSTM)**: Addresses vanishing gradient problems.
- **Gated Recurrent Units (GRU)**: Simplified LSTM alternative.

Optimization Techniques

- **Momentum**: Accelerates gradient descent.
- **Learning Rate Decay**: Adjusts learning rates dynamically.
- **Dropout**: Prevents overfitting by randomly disabling neurons.

5.3. Transformers

In 2017, the Google Research team published a paper called *"Attention Is All You Need"*, which presented the Transformer architecture and was a paradigm shift in Machine Learning, especially in Deep Learning and the field of natural language processing. The Transformer, with its parallel processing capabilities, allowed for more efficient and scalable models, making it easier to train them on large datasets. It also demonstrated superior performance in several NLP tasks, such as sentiment analysis and text generation tasks. This is considered as a giant leap in the field of deep

learning which has been pivotal in leading to large language models and several other neural network advancements which have revolutionized the industry.

5.3.1. The transformer architecture consists of the following parts

- **Encoder**: The encoder consists of a stack of identical layers, each containing two main components: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. Each of these components is followed by layer normalization and residual connections. The multi-head attention mechanism enables the model to focus on various parts of the input sequence simultaneously, capturing different aspects of the data. The feed-forward network processes each position independently and identically.
- **Decoder**: The decoder is similar to the encoder but includes an additional multi-head attention mechanism that attends to the encoder's output. This allows the decoder to focus on relevant parts of the input sequence while generating the output. The decoder also includes masking in its self-attention mechanism to prevent positions from attending to subsequent positions, ensuring that the predictions for a position depend only on the known outputs at positions less than the current position.
- **Input Embeddings**: Both the encoder and decoder start with input embeddings, which convert the input tokens into high-dimensional vectors. These embeddings are then combined with positional encodings to provide the model with information about the relative or absolute position of the tokens in the sequence.
- **Positional Encoding**: Positional encodings are added to the input embeddings to give the model information about the position of each token in the sequence. These encodings use sine and cosine functions of different frequencies.
- **Layer Normalization**: Layer normalization is applied after each sub-layer (attention and feed-forward) to stabilize and accelerate the training process.
- **Feed-Forward Network**: The feed-forward network consists of two linear transformations with a ReLU activation in between. This network is applied to each position separately and identically.
- **Multi-Head Attention**: The multi-head attention mechanism allows the model to jointly attend to information from different representation subspaces at different positions. It consists of several attention heads, each of which performs a scaled dot-product attention operation.
- **Residual Connections**: Residual connections are used around each sub-layer to help with the flow of gradients during training, facilitating the training of deep networks.
- **Building the Transformer**: The Transformer model is constructed by stacking multiple encoder and decoder layers. The encoder processes the input sequence, and the decoder generates the output sequence, one token at a time, using the encoder's output as context.
- **Training and Inference**: During training, the model is trained to predict the next token in the sequence given the previous tokens. During inference, the model generates the output sequence one token at a time, using the previously generated tokens as input.

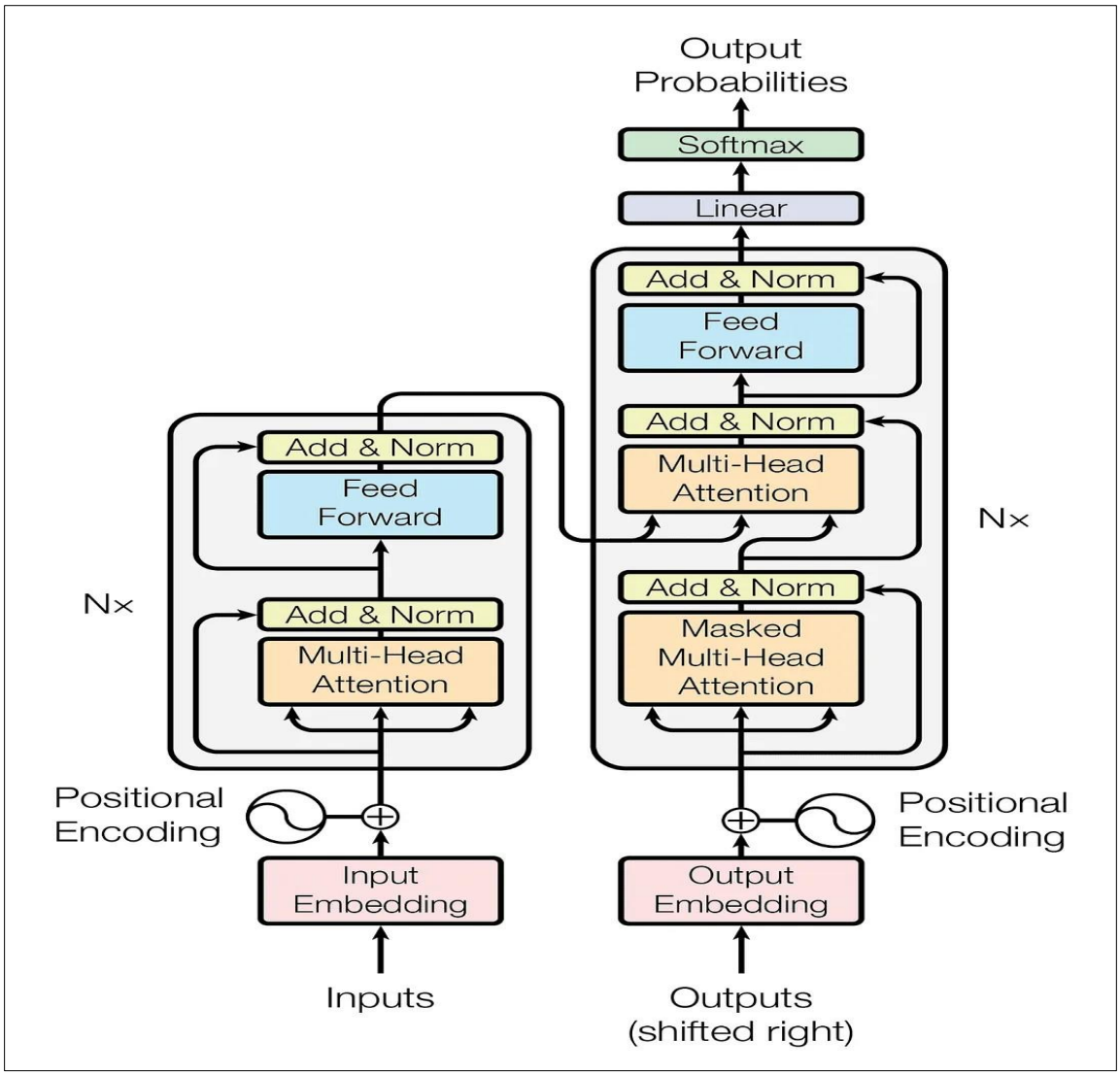

Figure 18 General architecture of transformers

5.3.2. Variants of Transformers:

- **BERT (Bidirectional Encoder Representations from Transformers)**: BERT is designed to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers. This allows the model to understand the context of a word based on its surroundings1.
- **GPT (Generative Pre-trained Transformer)**: GPT is an autoregressive model that uses a transformer decoder to generate text. It is trained to predict the next word in a sentence, making it effective for text generation tasks1.
- **T5 (Text-To-Text Transfer Transformer)**: T5 treats every NLP problem as a text-to-text problem. This means that both the input and output are always text strings, which allows the model to be applied to a wide range of tasks1.
- **Transformer-XL**: This variant introduces a segment-level recurrence mechanism and a novel positional encoding scheme, which enables learning dependency beyond a fixed length without disrupting temporal coherence1.
- **XLNet**: XLNet is a generalized autoregressive pretraining method that enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order1.
- **RoBERTa (Robustly optimized BERT approach)**: RoBERTa is a robustly optimized version of BERT that modifies key hyperparameters, removes the next sentence prediction objective, and trains with much larger mini-batches and learning rates1.

5.3.3. Optimization Techniques

- **Layer Normalization**: This technique is applied after each sub-layer (attention and feed-forward) to stabilize and accelerate the training process1.
- **Residual Connections**: These connections are used around each sub-layer to help with the flow of gradients during training, making it easier to train deep networks1.
- **Dropout**: Dropout is used to prevent overfitting by randomly setting a fraction of input units to zero at each update during training time1.
- **Learning Rate Scheduling**: Adjusting the learning rate during training can help the model converge faster and avoid local minima1.
- **Gradient Clipping**: This technique involves clipping the gradients during backpropagation to prevent the gradients from exploding, which can destabilize the training process1.
- **Mixed Precision Training**: Using mixed precision (16-bit and 32-bit floating-point numbers) can speed up training and reduce memory usage without sacrificing model performance1.

6. Applications for Deep Learning

Deep learning is used across industries:

6.1. Healthcare: Deep learning has made significant strides in healthcare, with various models applied to medical imaging, diagnostics, drug discovery, and patient care. Here are some of the most popular deep learning models used in healthcare and their applications.

- ResNet, DenseNet, EfficientNet: Used in analyzing X-rays, MRIs, CT scans, and histopathology images to detect diseases like pneumonia, brain tumors, and cancer.
- U-Net: A popular segmentation model for identifying regions of interest in medical images (e.g., tumor segmentation in MRI scans).
- CheXNet: Developed by Stanford, it detects pneumonia in chest X-rays with performance comparable to radiologists.
- LSTM Networks: Used for predicting patient deterioration, sepsis onset, and mortality risk based on time-series patient data.
- Bi-LSTM: Helps in clinical text analysis by processing patient notes, prescriptions, and medical histories.
- BioBERT & Med-BERT: Fine-tuned versions of BERT for biomedical text processing, used for clinical note summarization, question answering, and literature search.
- GPT-based Models: Used for generating medical reports and improving doctor-patient interactions via AI-driven chatbots.
- DeepMind's AlphaFold: Predicts protein structures, aiding in drug discovery.
- IBM Watson Health: Uses NLP and deep learning for clinical decision support.
- Google's DeepVariant: Identifies genetic variants in DNA sequencing.

6.2. Finance: Deep learning is widely used in finance for fraud detection, algorithmic trading, risk assessment, and customer service. Below are some popular deep learning models used in finance and their applications:

- Fraud Detection: CNNs analyze transaction sequences as images to detect fraudulent behavior.
- Stock Price Prediction: CNNs are used to detect trends and patterns in financial charts and time-series data.
- Portfolio Optimization: CNN-based models analyze correlation heatmaps to optimize asset allocation.
- LSTM Networks: Used to predict stock prices, forex movements, and commodity trends based on historical data.
- Risk Assessment: LSTMs analyze sequential customer data for credit scoring and loan default prediction.
- Market Sentiment Analysis: NLP-powered LSTMs analyze news articles, financial reports, and social media to determine market sentiment.
- FinBERT: A finance-specific version of BERT, used for sentiment analysis in financial reports and earnings call transcripts.
- GPT-based Models: Generate financial summaries, analyze contracts, and automate regulatory compliance.
- AI-Powered Chatbots: GPT models assist in customer service, investment advisory, and financial planning.
- Unsupervised Fraud Detection: Autoencoders detect unusual transaction patterns that may indicate fraud.
- Market Anomaly Detection: Identify irregular trading patterns in stock markets.
- J.P. Morgan's LOXM: Uses deep learning for automated trading and portfolio management.
- Visa & Mastercard: Use deep learning for real-time fraud detection.
- BloombergGPT: A finance-specific AI model used for summarizing financial reports and analyzing sentiment.

- DeepMind's AlphaPortfolio: Applies reinforcement learning to optimize investment portfolios.

6.3. Autonomous Vehicles: Autonomous vehicles (AVs) rely on deep learning for perception, decision-making, and control. Here are some of the most popular deep learning models used in autonomous driving and their applications:

- YOLO (You Only Look Once): Real-time object detection for recognizing pedestrians, vehicles, traffic signs, and obstacles.
- Faster R-CNN: Used for high-precision object detection and localization.
- SSD (Single Shot MultiBox Detector): Balances speed and accuracy in detecting road objects.
- U-Net & SegNet: Semantic segmentation models used for lane marking detection and drivable area segmentation.
- LSTM Networks: Used for predicting the trajectory of surrounding vehicles and pedestrians.
- GRUs (Gated Recurrent Units): Process time-series data from sensors (e.g., LiDAR, radar) to enhance decision-making.
- ViTs (Vision Transformers): Used in object recognition and scene understanding, competing with CNNs.
- BERT-based Models: NLP-based models assist in understanding traffic signs, road signs, and voice commands.
- Tesla Autopilot & FSD: Uses CNNs, transformers, and RL for lane following, traffic sign recognition, and navigation.
- Waymo's Self-Driving Stack: Uses deep learning for sensor fusion, object tracking, and trajectory prediction.
- NVIDIA DRIVE: A complete AI platform that integrates CNNs and reinforcement learning for autonomous vehicle control.
- Uber ATG & Aurora: Focuses on deep learning-based perception and planning for auomous ride-hailing.

6.4. Natural Language Processing: Chatbots and language translation.

- Vanilla RNNs: Used for simple text processing tasks like sentence classification and next-word prediction.
- LSTM Networks: Used in chatbots, text summarization, and language modeling.
- GRUs: A computationally efficient alternative to LSTMs, commonly used in speech recognition and predictive text input.
- TextCNN: A CNN-based model for text classification (e.g., spam detection, sentiment analysis).
- Character-level CNNs: Used for morphological text processing, such as misspelled word corrections.
- RoBERTa: An optimized version of BERT for improved text understanding.
- DistilBERT: A lightweight BERT model for faster inference.
- ALBERT: A more efficient, parameter-reduced version of BERT.
- Google Search (BERT-based NLP): Improves search query understanding.
- ChatGPT (GPT-4-based NLP): AI-powered conversation assistant.
- Amazon Alexa & Google Assistant: Use transformer-based models for voice and text interactions.
- Google Translate: Uses Seq2Seq models with transformers.
- Grammarly & QuillBot: Use transformers for text correction and rephrasing.

7. Conclusion

A McKinsey survey showed that while AI adoption has been adding value to organizations, the use of deep learning in business is still at an early stage. However, the advancements of this approach to AI are very promising. It is already transforming several industries, including healthcare, finance, and automotive.

Companies have massive amounts of data, known as big data, that need to be analyzed and understood. Deep learning is capable of handling it and analyzing it very accurately. Classic machine learning algorithms like Naïve Bayes, decision trees, and support vector machines can't directly be applied to raw data. Typically, a preprocessing step called feature extraction is necessary to convert the data into a format that algorithms can understand so as to classify it. The process is complex and time-consuming.

In contrast, deep learning eliminates the need for manual feature extraction, with neural networks capable of drawing features out of raw data automatically. Multiple processing layers produce increasingly complex representations of data, culminating in the best possible input data.

Deep learning enables transfer learning, a technique where a pre-trained neural network can be reused for a new but related problem. The input and middle layers of the pre-trained model, which detect generic features, are transferred to another network. Only the later layers need to be retrained to recognize more specific features. This approach is ideal when sufficient data is challenging to obtain or when faster training is required. Several public and third-party models and toolkits are available for fine-tuning for different use cases.

These advantages of deep learning techniques make them useful and increase the scope of being used for implementing new models for new use cases. In the future of deep learning, there will be more advancements in terms of algorithms. For instance, FAIR has developed state-of-the-art deep learning models like Detectron2 for object detection problems which outperform many other previously released algorithms like VGG16 and ResNet models.

Deep learning continues to gain popularity owing to its highly accurate results, ability to make predictions on unstructured datasets, and provide useful insights about the datasets. While often confused with machine learning and artificial intelligence, it is specifically based on artificial neural networks. And although its use in business is still at a nascent stage, it holds much promise as it is already transforming several industries. It's likely that deep learning will continue to evolve and have a significant impact on our society in the future

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

[2] Schmidhuber, J. (2015). Deep learning in neural networks: An overview. Neural Networks, 61, 85-117.

[3] Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press.

[4] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. Advances in Neural Information Processing Systems, 25, 1097-1105.

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 770-778.

[6] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30, 5998-6008.

[7] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780.

[8] Radford, A., Narasimhan, K., Salimans, T., & Sutskever, I. (2018). Improving language understanding by generative pre-training. OpenAI.

[9] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[10] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. arXiv preprint arXiv:2005.14165.

[11] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., ... & Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587), 484-489.

[12] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

[13] Kingma, D. P., & Welling, M. (2014). Auto-encoding variational Bayes. arXiv preprint arXiv:1312.6114.

[14] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. Proceedings of the International Conference on Machine Learning, 448-456.

[15] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1-9.

[16] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster R-CNN: Towards real-time object detection with region proposal networks. Advances in Neural Information Processing Systems, 28, 91-99.

[17] Girshick, R. (2015). Fast R-CNN. Proceedings of the IEEE International Conference on Computer Vision, 1440-1448.

[18] Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[19] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[20] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI.

[21] Chen, T., Kornblith, S., Norouzi, M., & Hinton, G. (2020). A simple framework for contrastive learning of visual representations. Proceedings of the International Conference on Machine Learning, 1597-1607.

[22] He, K., Fan, H., Wu, Y., Xie, S., & Girshick, R. (2020). Momentum contrast for unsupervised visual representation learning. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 9729-9738.

[23] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. Proceedings of the European Conference on Computer Vision, 213-229.

[24] Deep learning from scratch by Seth Weidman

[25] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[26] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. arXiv preprint arXiv:2005.14165