Deep Learning (DL) Architectures and Transfer Learning for Tabular, Unstructured, Sequential and Time-Dependent Data

Ryan Paul Lafler

Premier Analytics Consulting, LLC

Premier Analytics Consulting, LLC

Deep Learning (DL) Fundamentals and Architectures

Understanding neural networks for tabular, unstructured, and time-dependent data.

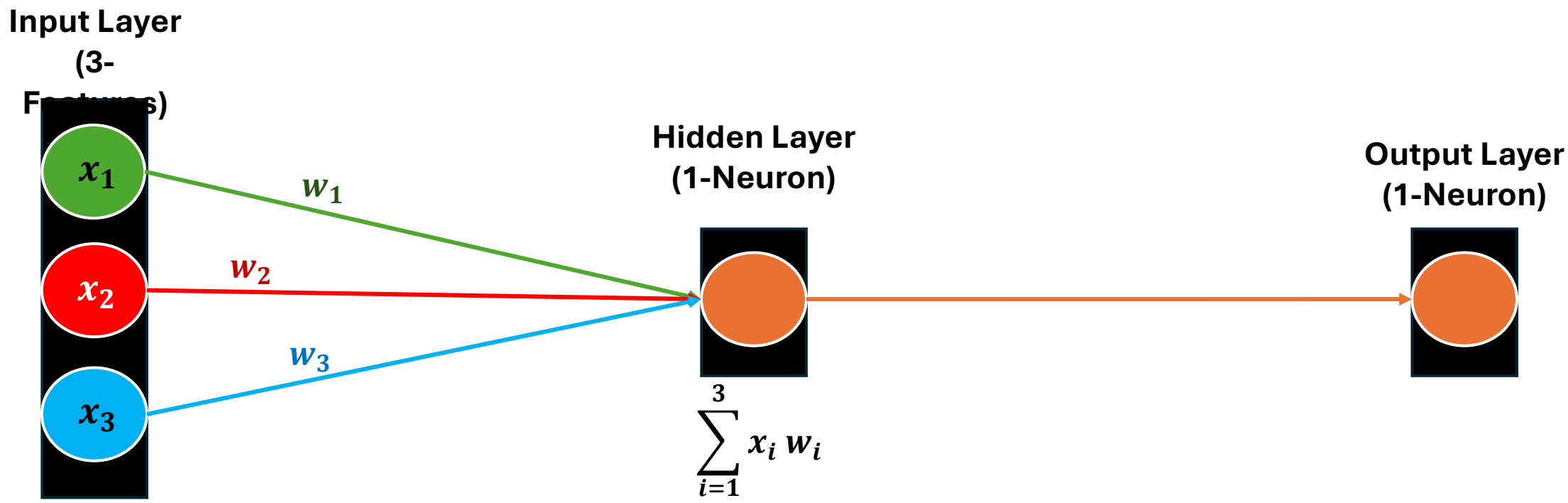
What is Deep Learning (DL)?

- **Deep Learning** leverages layers of neurons that form connections, transform data, and extract insights for regression, classification, ranking, forecasting, risk analysis, data mining, data compression, and generative AI
- Consists of several layers including:
  - **Input:** Receives raw data (tabular, audio, videos, images, and text)
  - **Hidden:** Each layer transforms data into meaningful representations understood by the network (called **feature maps** or **embeddings**)
  - **Output**: Gives final output(s) to user
- **Weights** are connecting neurons between layers, update through **backpropagation**
  - Starts as random number, then optimized during training (epochs) to better values

Neural Networks for Predictive Analytics

- Neural Networks can be designed for *regression* or *classification* by **output layer function**
  - **Regression Network** → Output function: Linear activation function
  - **Classification Network** → Output function: *Sigmoid* (binary classification) or *Softmax* (multiple classification) activation function
- Networks require significant amounts of labeled data to correctly map results and *generalize* to unseen data
- Feature engineering is automatically handled by the network → layers transform data for you
- Networks can estimate non-linear, multi-dimensional decision boundaries and prediction functions

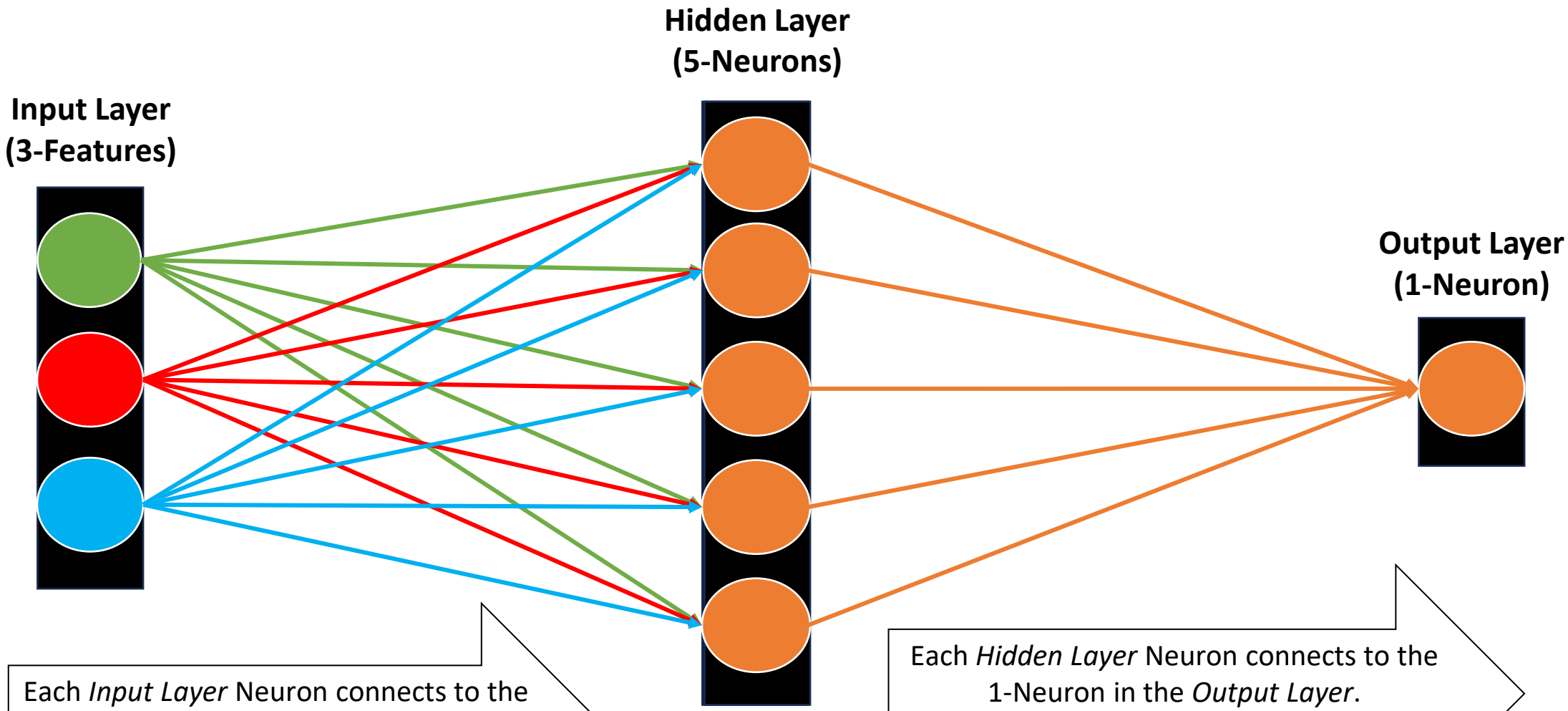
Visualizing a Simple Dense Network



Each *Input Layer* Neuron connects to the 5-Neurons in the *Hidden Layer*.
$3 \times 1 = 3$ total connections.

Each *Hidden Layer* Neuron connects to the 1-Neuron in the *Output Layer*.
$1 \times 1 = 1$ total connection.

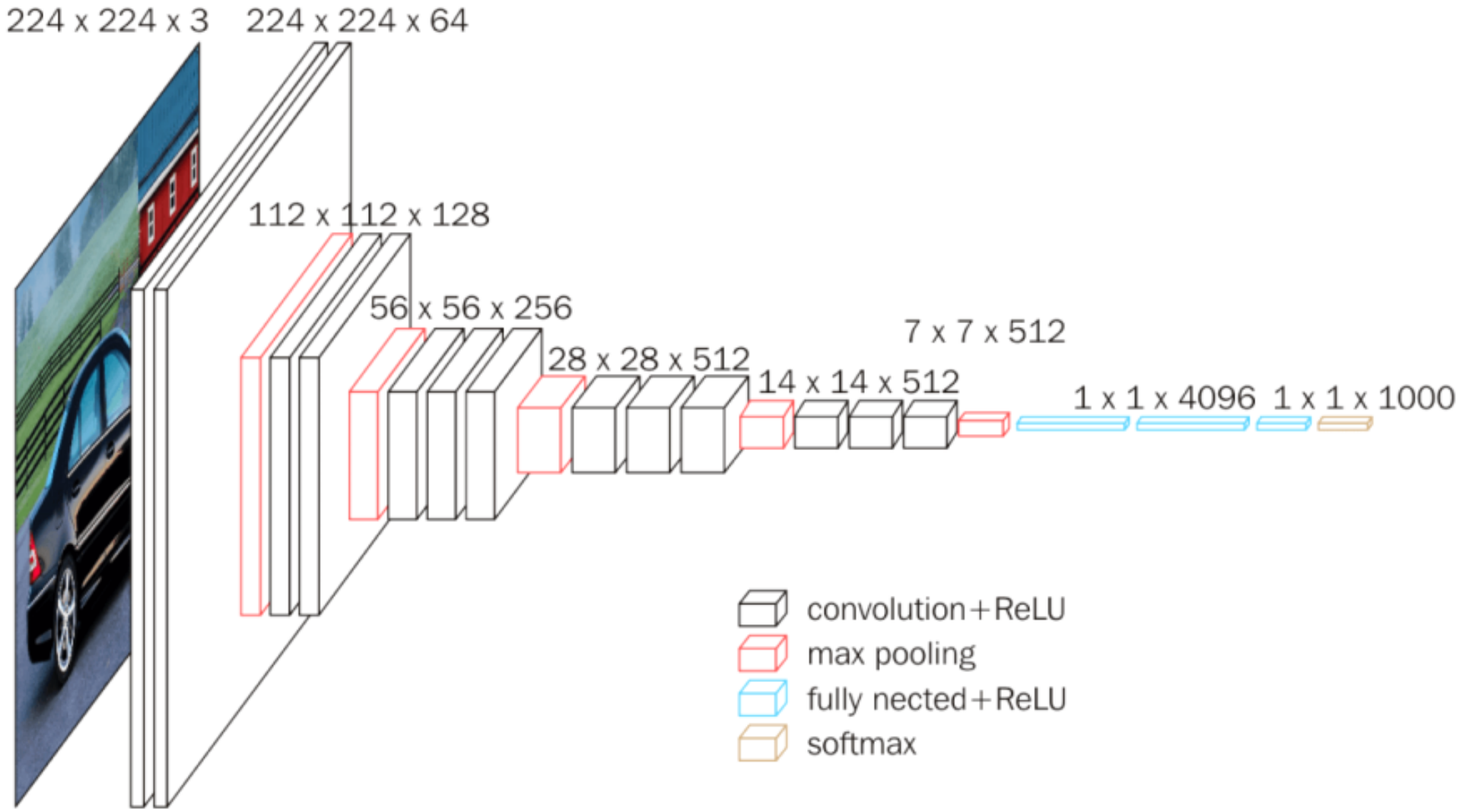
Visualizing a Hidden Layer with Multiple Neurons



Neural Networks for Content Generation

- Convolutional Neural Networks (CNNs) extract features, characteristics, and edges from images
  - Preserves information from nearby pixels in images → **spatial associations**
  - Useful for **object detection**, **image segmentation**, and **classification** tasks
- Neural Networks are also powerful generative models
  - **Variational Autoencoders** (**VAEs**) learn the *distribution* that training data come from and attempt to generate new observations from *learned distribution*
  - **Generative Adversarial Networks** (**GANs**) involve a *feedback-Network* that rates the quality of generated observations from the generative Network
    - Network continues training until it produces output deemed high quality

Visualizing a Convolutional Neural Network

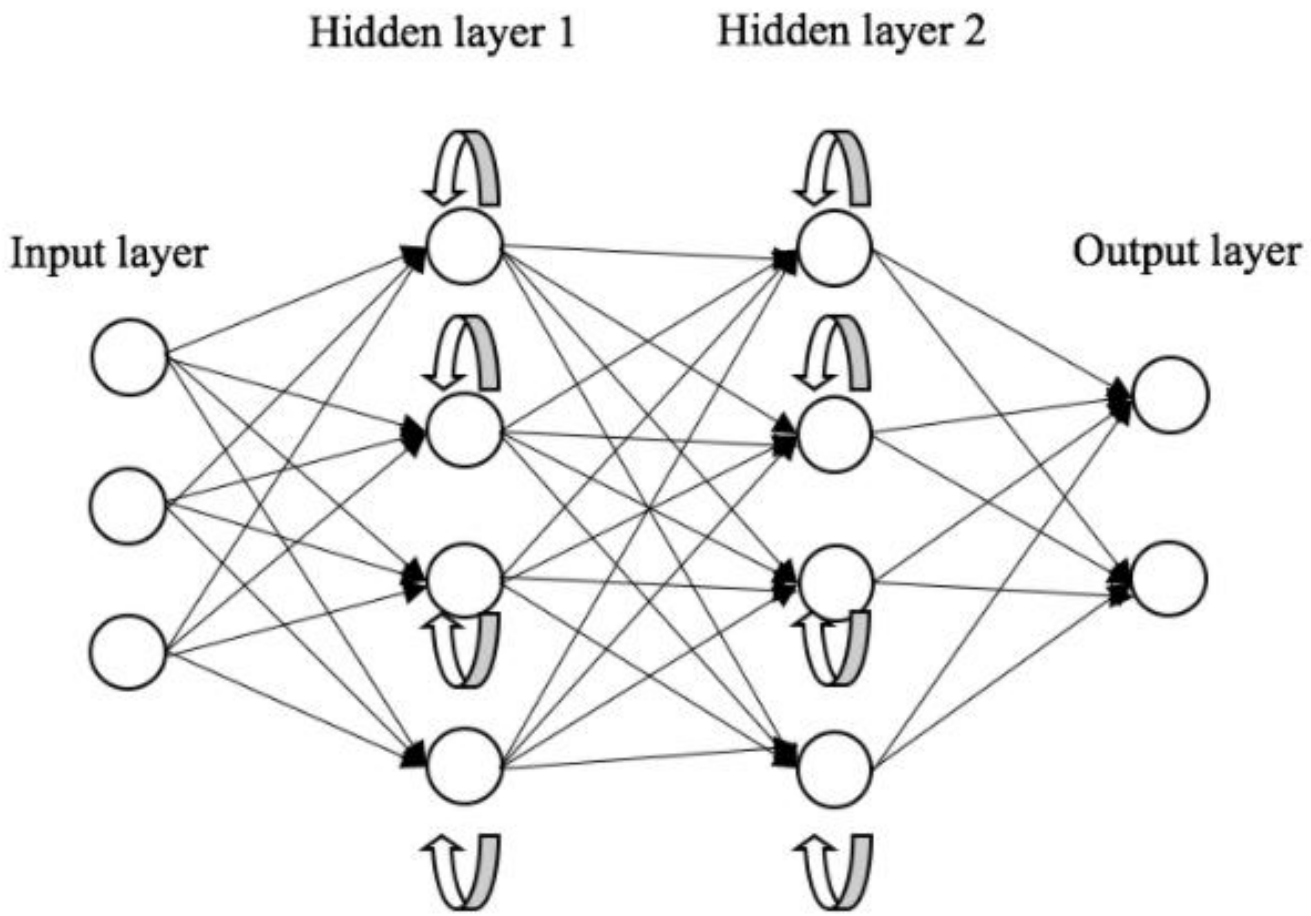


Convolutional Neural Networks (CNNs)

- Examines relationships between features using *moving windows* that shift across dataset
  - Kernels (also called filters) scan vectors (columns) or multi-dimensional arrays (i.e., images, matrices) to identify **signals**, **patterns**, and **edges**
  - **Naturally handles unstructured data types** → automatic feature extraction
  - Base layers identify key characteristics / patterns, middle layers target finer-details, and last layers focus on fine-tuning to specific task / industry / objective
- Designed to recognize patterns, extract meaningful identifiers, and parse data to create **feature maps** from original data
- Does not maintain memory of previous inputs → each input assumed **independent**

How Feature Mapping Works

Visualizing a Recurrent Neural Network



General Form of RNNs

Recurrent Neural Networks (RNNs)

- Inputs are assumed dependent on each other → nearby time steps, or words in a sequence, are assumed associated (**not independent**)
- **RNNs maintain memory** and **move one-by-one across time steps / words in sequence**
  - RNN takes new input and combines next input with memory (hidden state)
  - Gives prediction and loop continues, **sequentially**, until all inputs are exhausted
- **Suffers from vanishing gradient problem** → long-term memory is forgotten and information from early hidden states not used
  - **Long Short-Term Memory (LSTM)** introduce gates (what info to remember / forget)
  - **Gated Recurrent Units (GRU)** are simpler memory managers than LSTMs, faster but less complex

Premier Analytics Consulting, LLC

Exploring the Next Frontier with Generative AI Workflows

Harnessing encoder, decoder, and hybrid transformer models to create content, capture context, and enable reasoning with information retrieval.

Generative AI Architectures

- **Transformers** key to generative AI workflows → parallelize **attention** layers in neural networks and replace sequential networks such as RNNs and LSTMs
- **Encoder-Only architectures** excel at representation learning, embeddings, and retrieval; power search, classification, and semantic similarity
  - Create contextual representations (vector embeddings) from input tokens
  - Examples include **BERT** and **RoBERTa**
- **Decoder-Only architectures** provide autoregressive generation for text, code, simulations, and creative tasks; optimized for long-form output
  - Each output token *only* focuses on previous token(s), does not see ahead of current token → examples include GPT family & LLaMa family

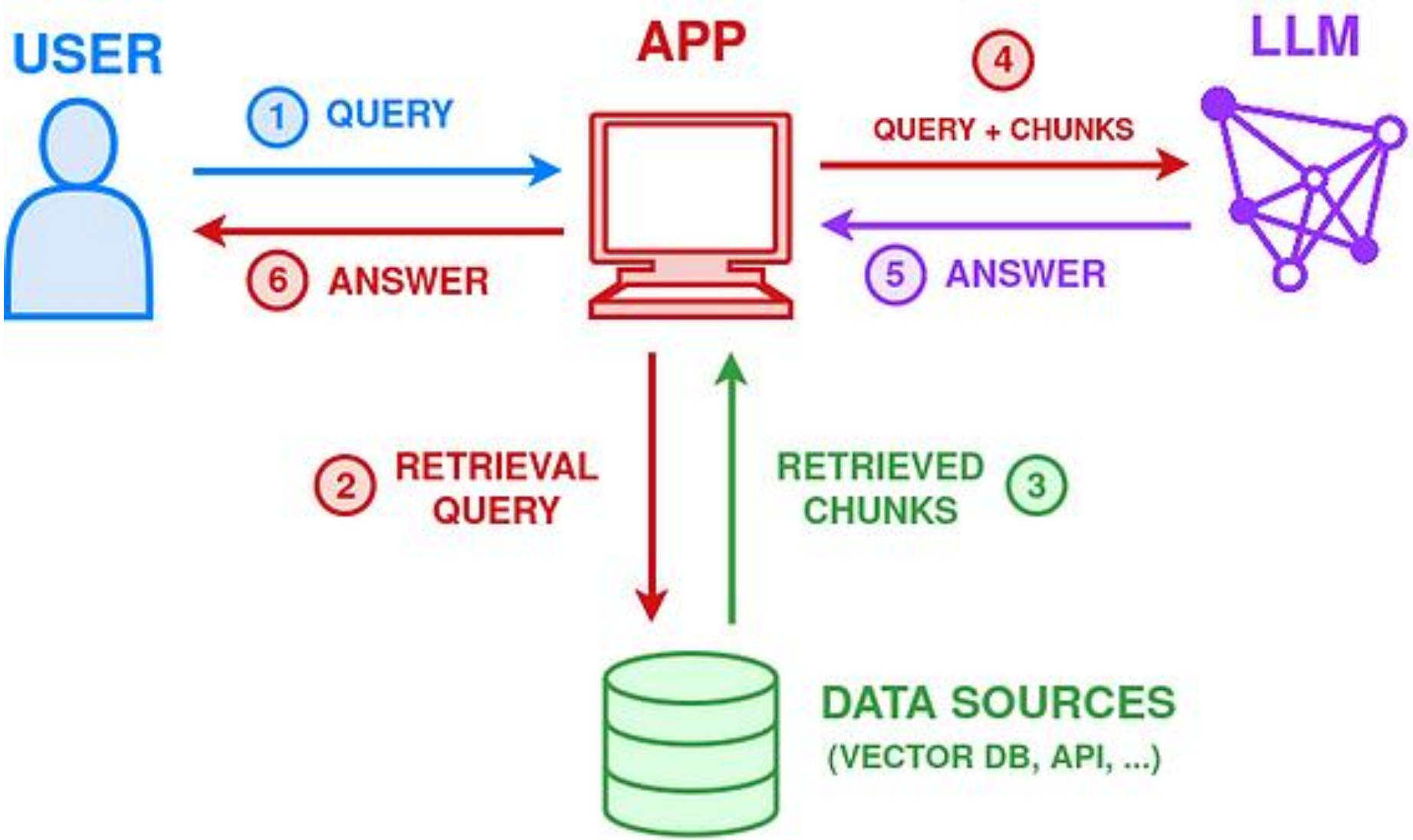
Generative AI Architectures

- **Encoder-Decoder architectures** are ideal for translation, summarization, multimodal reasoning; integrate comprehension + generation
  - Encoder-part transforms input sequence to vector embeddings
  - Decoder-part consumes vector embeddings and generates output sequence
    - Encoder-part builds a rich representation by reading entire input sequence and transforming into latent (vector) space
    - Decoder generates tokens **one-at-a-time** using **casual attention** to maintain sequence order
  - Unlike Decoder-Only architectures, Encoder-Decoder transformers excel at summarization and translation tasks by focusing generation guided by input context

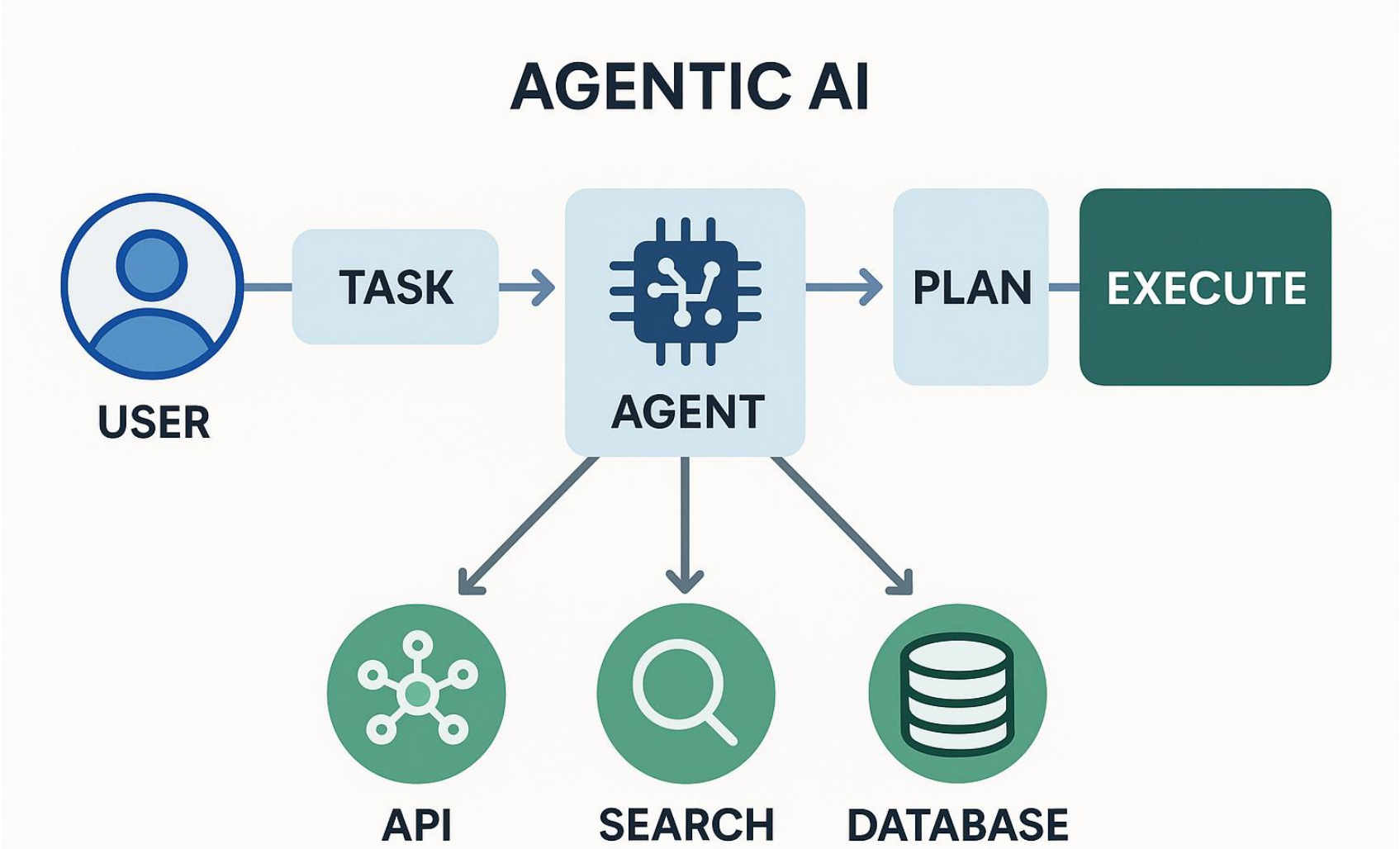
Retrieval-Augmented Generation (RAG)

- **Bridges search + generation:** Combines external knowledge retrieval with large language models (LLMs) to improve factual accuracy
- **Retriever module:** Fetches relevant documents or embeddings from a vector database or search index based on the query
- **Vector Databases:** Purpose-built for storing and searching vector embeddings
  - Enables fast similarity search and semantic retrieval to power RAG, recommendations, and multimodal AI
- **Generator module:** Conditions its response on both the input query and retrieved content, ensuring grounded outputs
- Good for **Q/A systems**, **chatbots**, & **technical documentation search**

Retrieval-Augmented Generation (RAG)



Agentic AI Workflow



Deep Learning (DL) Architectures and Transfer Learning for Tabular, Unstructured, Sequential and Time-Dependent Data

Ryan Paul Lafler

Premier Analytics Consulting, LLC