NLP

(Natural Language Processing)

using

Transformers

Sandeep Arora : Principal Architect
https://www.linkedin.com/in/sandeeparora/

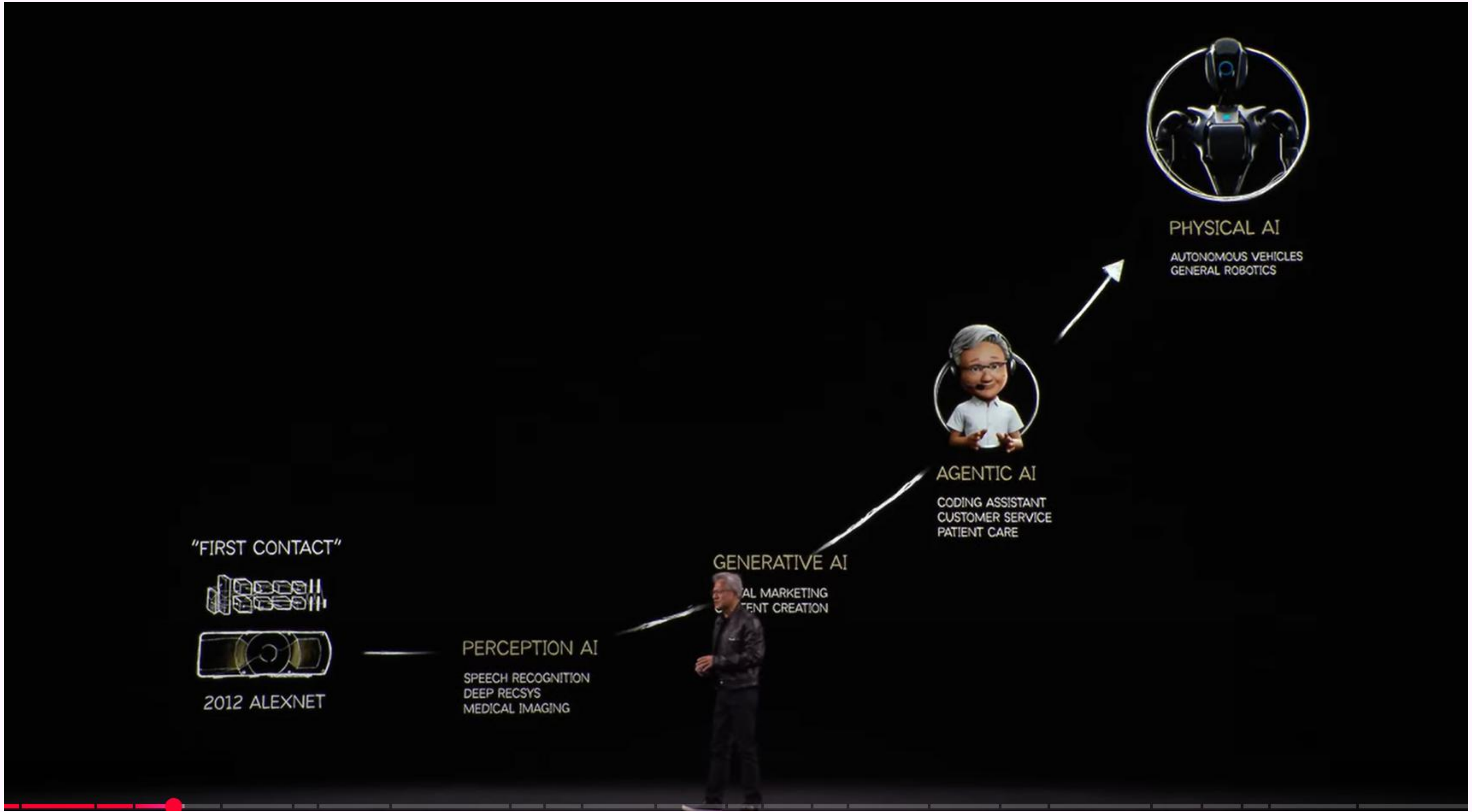
What is NLP

NLP is a field of **linguistics** and **machine learning** focused on understanding human language. The aim of NLP is not only to understand single words individually, but to be able to understand the context of those words. Below are a few common tasks

- **Sentence-level classification**:
Determine sentiment, detect spam, check grammar, or assess logical relationships between sentences.
- **Word-level classification**: Identify parts of speech (e.g., noun, verb) and recognize named entities like **people**, **places**, or **organizations**.
- **Text generation**: Auto-complete prompts, fill in blanks, or generate new text based on a given input.
- **Text-to-text transformation**: Extract answers from context or perform tasks like **translation** and **summarization**.

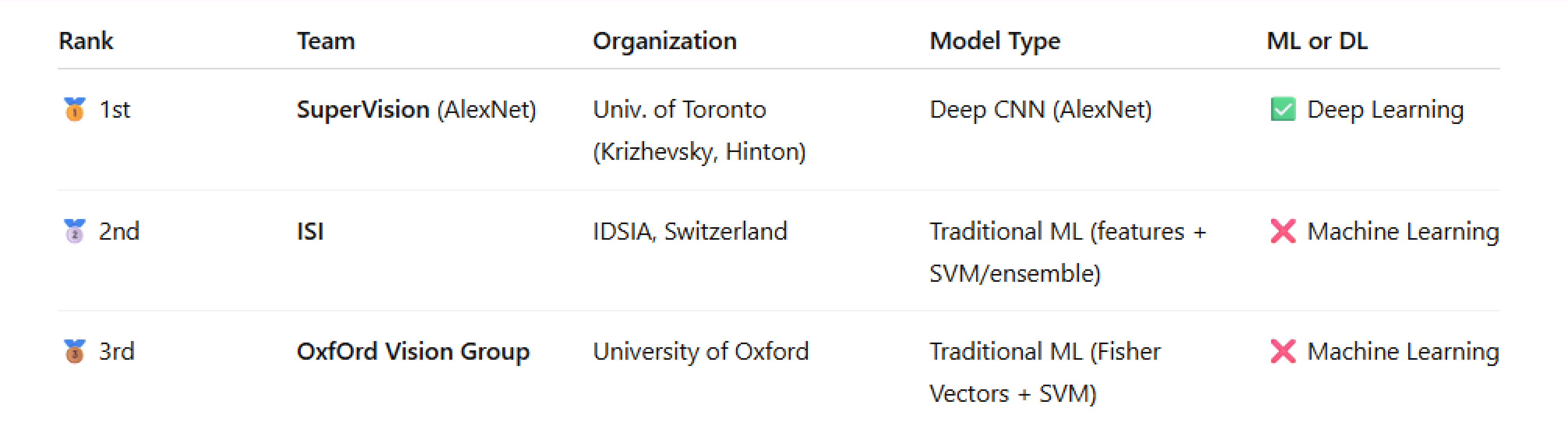
Lets take a step back. 2012 was the birth of Deep Learning Revolution

- You can say, 2012 was the birth of Deep Learning.
- AlexNet won the competition. A CNN (type of Neural Network in Deep Learning) was trained to algorithm automatically learnt features of image and was able to predict images with high accuracy.
- After that scientists started researching on using Neural networks for NLP



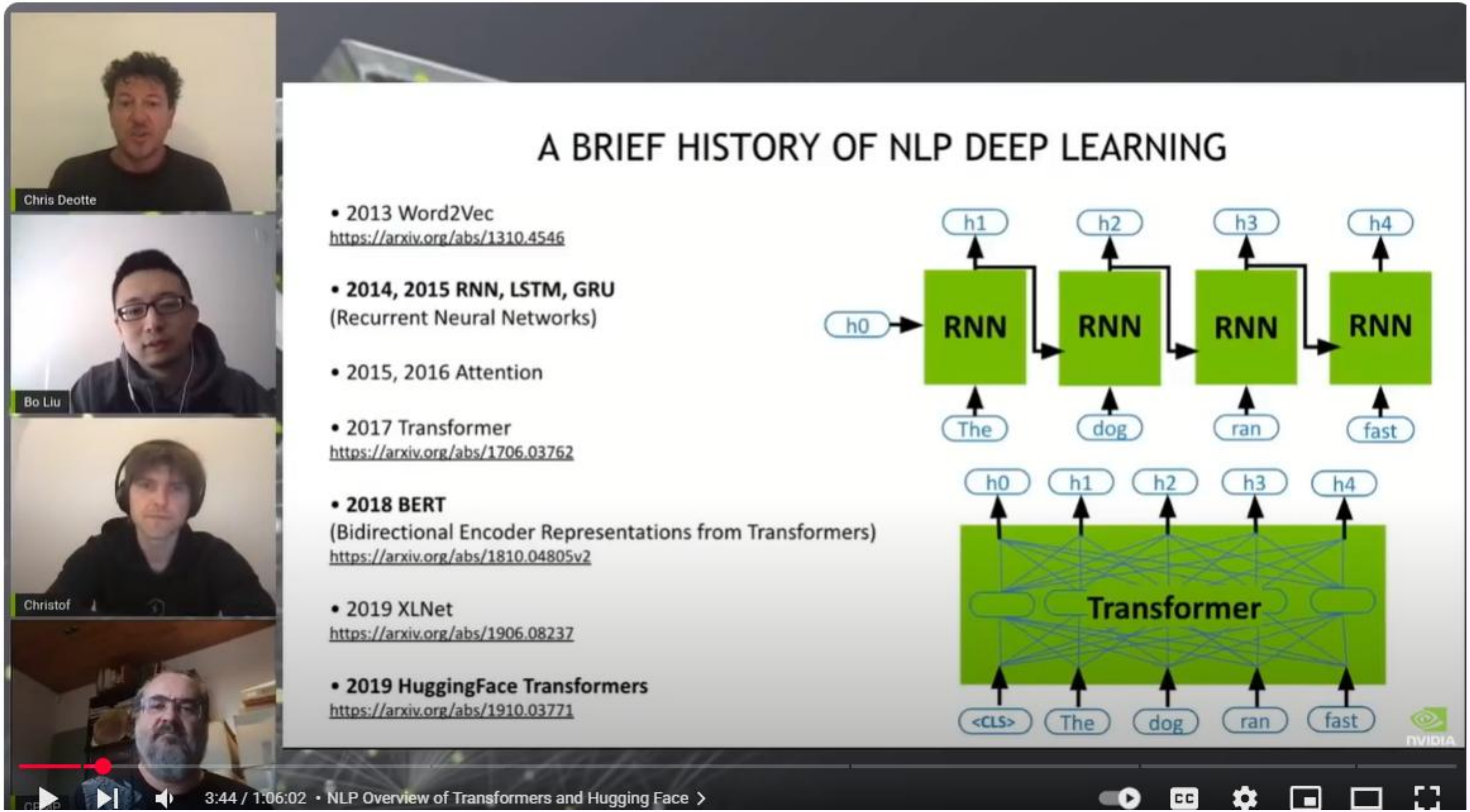
https://youtu.be/X9cHONwKkn4?t=8440

2012 was the birth of Deep Learning Revolution



Rank	Team	Organization	Model Type	ML or DL
🥇 1st	**SuperVision** (AlexNet)	Univ. of Toronto (Krizhevsky, Hinton)	Deep CNN (AlexNet)	✅ Deep Learning
🥈 2nd	**ISI**	IDSIA, Switzerland	Traditional ML (features + SVM/ensemble)	❌ Machine Learning
🥉 3rd	**OxfOrd Vision Group**	University of Oxford	Traditional ML (Fisher Vectors + SVM)	❌ Machine Learning

NLP – brief history



A BRIEF HISTORY OF NLP DEEP LEARNING

- 2013 Word2Vec
https://arxiv.org/abs/1310.4546
- **2014, 2015 RNN, LSTM, GRU**
(Recurrent Neural Networks)
- 2015, 2016 Attention
- 2017 Transformer
https://arxiv.org/abs/1706.03762
- **2018 BERT**
(Bidirectional Encoder Representations from Transformers)
https://arxiv.org/abs/1810.04805v2
- 2019 XLNet
https://arxiv.org/abs/1906.08237
- **2019 HuggingFace Transformers**
https://arxiv.org/abs/1910.03771

https://www.youtube.com/watch?v=PXc_SlnT2g0

Word2Vec – (2013 created by Tomas Mikolov, Google. It is open sourced.)

A neural network model that turns words into meaningful **vectors**. Learns **word meanings** by placing similar words **closer** in vector space.

Architecture

Word2Vec uses a shallow neural network (1 hidden layer), so it's an early form of Deep Learning — but very lightweight.

How is Word2Vec Model trained

You train it on your own business data using one of the below methods

- CBOW: Predicts a word given its context (surrounding words).
- Skip-gram: Predicts the surrounding words given a single input word

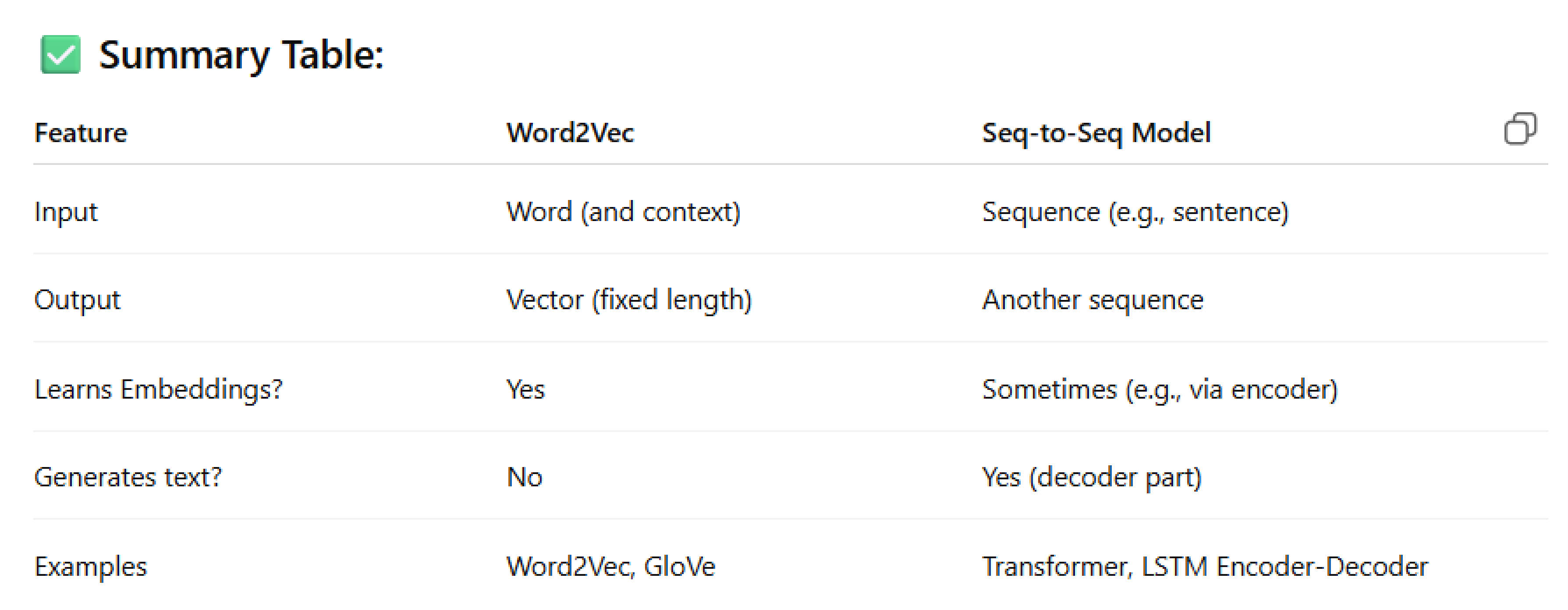
Is it a Sequence-to-Sequence Model ?

- No, Word2Vec is not a **sequence-to-sequence** (seq-to-seq) model
- A **seq-to-seq model** takes in a **sequence** (e.g., a sentence) and **outputs another sequence** — often of different length.
- Word2Vec

Word2Vec – (2013 created by Tomas Mikolov, Google. It is open sourced.)

Is it a Sequence-to-Sequence Model ?

- No, Word2Vec is not a sequence-to-sequence (seq-to-seq) model
- A **seq-to-seq model** takes in a **sequence** (e.g., a sentence) and **outputs another sequence** — often of different length.



✅ Summary Table:

Feature	Word2Vec	Seq-to-Seq Model
Input	Word (and context)	Sequence (e.g., sentence)
Output	Vector (fixed length)	Another sequence
Learns Embeddings?	Yes	Sometimes (e.g., via encoder)
Generates text?	No	Yes (decoder part)
Examples	Word2Vec, GloVe	Transformer, LSTM Encoder-Decoder

GloVe is another word embedding model that improves on Word2Vec by using global word co-occurrence. It comes with pre-trained vectors from large datasets like Wikipedia.

Word2Vec – (2013 created by Tomas Mikolov, Google. It is open sourced.)

A neural network model that turns words into meaningful **vectors**. Learns **word meanings** by placing similar words **closer** in vector space.

Used for

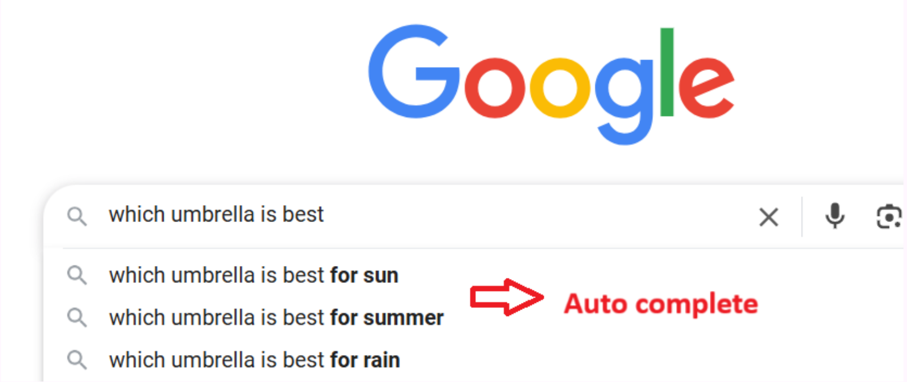
- Search relevance like search engines
- Text classification like email classification
- Sentiment analysis
- Word similarity
- Recommendation systems

Limitations

- Ignores word order and grammar.
- Cannot capture context of sentence.
- Requires large text data to train well.
- Cannot handle unseen (OOV) words.
- Does not capture sentence or document meaning.
- Shallow architecture with limited expressiveness.

RNN – Recuring Neural Networks

You must have seen many tools where when you enter a few words, it completes it for you. Here sequence is important.



Google

which umbrella is best

which umbrella is best **for sun**

which umbrella is best **for summer**

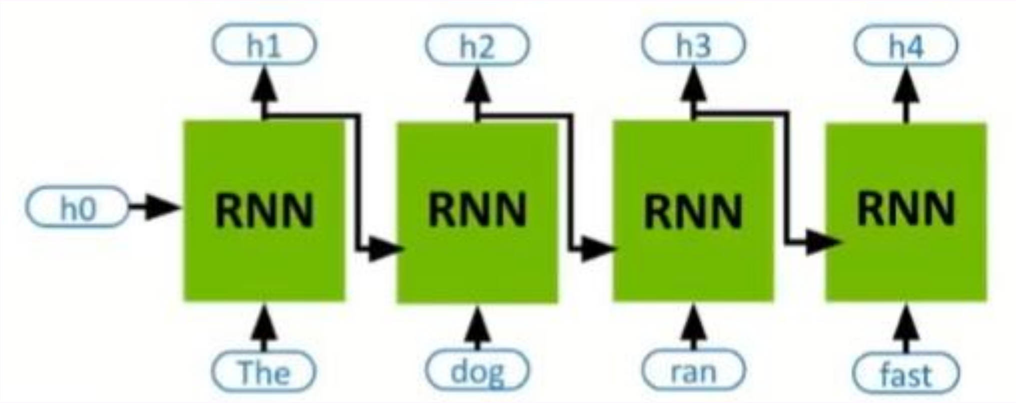
which umbrella is best **for rain**

Auto complete

What is an RNN

RNN (Recurrent Neural Network) is a type of neural network designed to handle sequential data, like natural language or time-series. It maintains memory by passing hidden states from one time step to the next.

.



Source: https://www.youtube.com/watch?v=PXc_SlnT2g0

RNN – Recuring Neural Networks

Few important concepts about RNNs

1. RNNs are not **Seq-to-seq** models because they take in one word at a time
2. RNNs are **auto-regressive**, meaning they generate one word at a time

Limitations

- Short memory: Can forget long-term context (solved by LSTM/GRU)
- Vanishing gradients during training: so quality decreases as length of sentence increases
- Slow computation: since you pass one word at a time. Sequential processing is less parallelizable
- Less effective than Transformers on long texts

LSTM – Long Short-Term Memory

LSTM (Long Short-Term Memory) is a special type of RNN designed to remember long-term dependencies.
It uses gates (input, forget, output) to control what to keep, forget, and output.

Benefits

- Solves vanishing gradient problem.
- Remembers long-term context in sequences.
- Works better than simple RNNs on long texts, speech, time series.

Limitations

- Complex architecture (more parameters, slower training)
- Still less parallelizable than Transformers
- Not ideal for very long sequences compared to modern models

Enter Transformers

Encoder

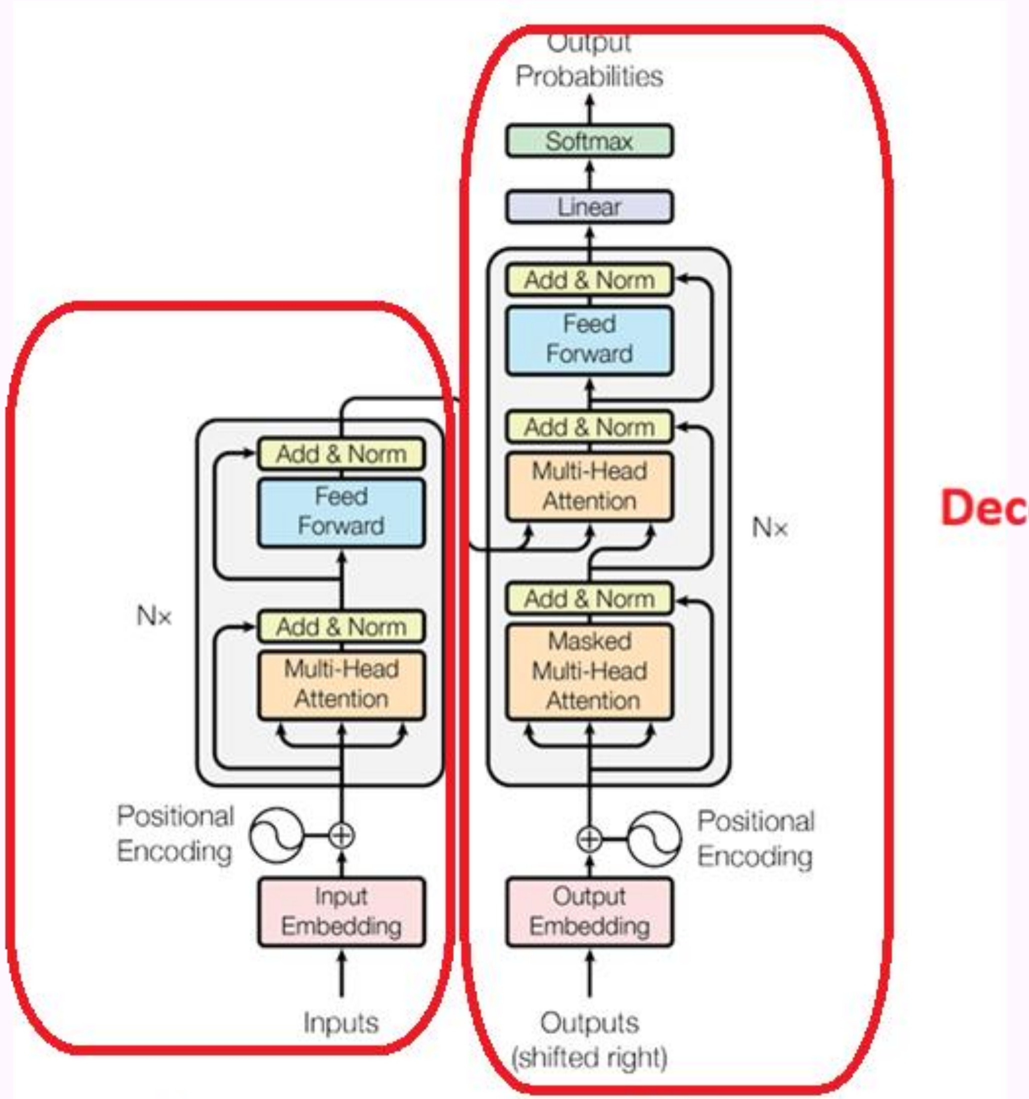
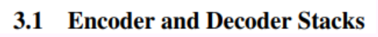


Figure 1: The Transformer - model architecture.

Decoder



3.1 Encoder and Decoder Stacks

https://arxiv.org/pdf/1706.003762

Enter Transformers

- The Transformer architecture was introduced in June 2017.
- The labs started using transformers for NLP
- (Important) Read my presentation on Transformers

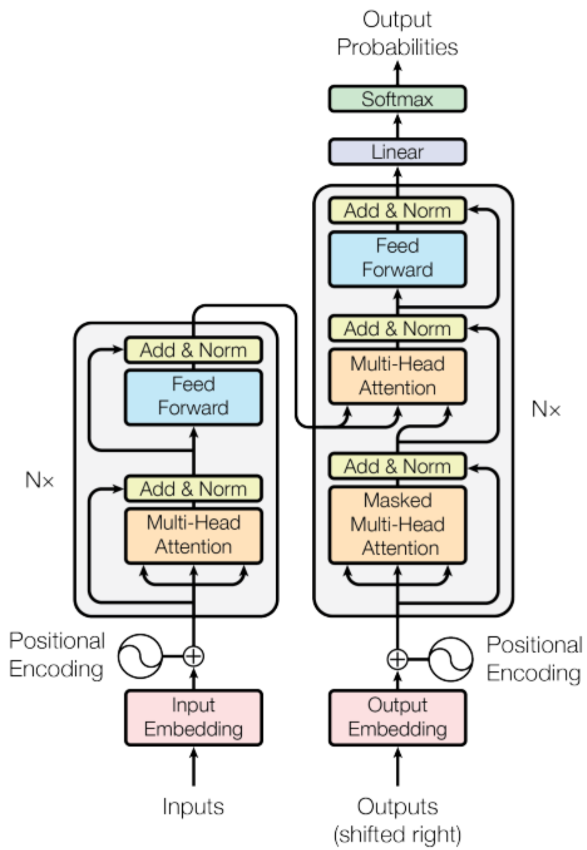


Figure 1: The Transformer - model architecture.

3.1 Encoder and Decoder Stacks

https://arxiv.org/pdf/1706.03762

The model has two main parts:

•**Encoder**: Understands the input and creates feature representations.

•**Decoder**: Uses those features to generate the output sequence.

Types of Transformer Models

Encoder-only (e.g., BERT):
Auto-encoder models that read text bidirectionally for deep understanding. Ideal for tasks like classification, NER, and question answering.

Decoder-only (e.g., GPT, LLaMA):
Auto-regressive models that generate text left-to-right. Best for open-ended generation like writing, coding, or completing prompts.

Encoder-decoder (e.g., T5, BART):
Seq-to-seq models that encode input and decode output. Great for tasks like translation, summarization, and generative Q&A.

BERT developed by Google in 2018

What is BERT?

BERT (Bidirectional Encoder Representations from Transformers) is a language model that deeply understands text by looking at words in both directions.

Architecture

- Based on the Transformer architecture
- Only uses the **encoder**, No decoder
- **Not** a **seq-to-seq** model; it is for **understanding**, not generating.

Use of BERT

- Google uses the BERT model in Google Search to better understand natural language queries.
- It understands the context of the entire query, not just individual keywords.
- It improves search ranking by better matching the user’s intent with relevant results.

GPT3 developed by OpenAI in 2020

What is GPT?

GPT-3 (Generative Pre-trained Transformer 3) is a large language model that generates human-like text by predicting the next word in a sentence based on prior context.

Architecture

- Based on the Transformer architecture
- Uses only the **decoder**, no **encoder**
- It is a seq-to-seq model; designed for generating, not just understanding

Use of GPT3

- GPT-3 powers AI applications like chatbots, writing assistants, and coding helpers.
- It generates text by understanding patterns in large volumes of training data.
- It can answer questions, write essays, translate text, and more—all from plain language prompts.

Summary

Language models in the Transformers library fall into three main categories:

Encoder-only models (e.g., **BERT**): Use a bidirectional approach to understand context. Best for classification, named entity recognition (NER), and question answering.

Decoder-only models (e.g., GPT, LLaMA): Process text left to right. Used for text generation, essay writing, and code generation.

Encoder-decoder models (e.g., T5, BART): Combine both approaches. These are seq-to-seq models. Used for translation, summarization, and question answering.