

Configuration

Groq API Key

.....

OpenAI API Key

.....

Custom URL (Optional)

e.g., https://example.com/ar

Search Engine with Tools & Agents

This agent uses Wikipedia, DuckDuckGo Search and Arxiv. You can also provide a Custom URL in the sidebar to add your own data.



Provide a summary of 1706.03762 paper.



Paper: *Attention Is All You Need* (arXiv:1706.03762)
Authors: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin
Published: June 2017 (conference version at NIPS 2017)

1. Motivation

Sequence-to-sequence tasks (machine translation, summarization, etc.) had been dominated by **recurrent neural networks (RNNs)** and **convolutional neural networks (CNNs)**. Both rely on either sequential processing (RNNs) or limited-range local context (CNNs), making them computationally expensive to train on long sequences and difficult to parallelize.

The authors asked: *Can we dispense with recurrence and convolution altogether and build a model that captures dependencies solely through attention?*

2. Core Idea – The Transformer

The **Transformer** replaces recurrence and convolution with **self-attention** mechanisms. Its architecture consists of an **encoder** and a **decoder**, each built from stacked identical layers:

Component	Sub-layers (per layer)
Encoder	1. Multi-head self-attention 2. Position-wise fully-connected feed-forward network
Decoder	1. Masked multi-head self-attention (prevents attending to future positions) 2. Multi-head encoder-decoder attention (queries the encoder outputs) 3. Position-wise feed-forward network

Each sub-layer is wrapped with **residual connections** and **layer normalization**.

2.1 Scaled Dot-Product Attention

Given queries **Q**, keys **K**, and values **V**:

$$\text{Attention}(Q,K,V)=\text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

The scaling factor ($\sqrt{d_k}$) stabilizes gradients for large dimensionalities.

2.2 Multi-Head Attention

Instead of a single attention function, the model linearly projects **Q**, **K**, **V** into *h* different sub-spaces, applies scaled dot-product attention in each, then concatenates and projects the result. This lets the model attend to information from different representation sub-spaces simultaneously.

2.3 Positional Encoding

Since the model contains no recurrence, **positional information** is injected by adding a deterministic sinusoidal vector to each input embedding:

$$PE_{(pos,2i)}=\sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right),\quad PE_{(pos,2i+1)}=\cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right)$$

These encodings allow the model to learn relative positions and generalize to longer sequences than seen during training.

2.4 Feed-Forward Networks

Each position passes through the same two-layer fully-connected network:

$$\text{FFN}(x)=\max(0,xW_1+b_1)W_2+b_2$$

with ReLU activation and dimensionality ($d_{\text{ff}}=4d_{\text{model}}$).

3. Training Details

- Dataset:** WMT 2014 English-German (4.5 M sentence pairs) and English-French (36 M pairs).
- Optimization:** Adam optimizer ($\beta_1=0.9$, $\beta_2=0.98$, $\epsilon=10^{-9}$) with a custom learning-rate schedule

$$\text{lr}=d_{\text{model}}^{-0.5}\cdot\min(\text{step}^{-0.5},\text{step}\cdot\text{warmup_steps}^{-1.5})$$

- Regularization:** Dropout (0.1) on attention weights and residual connections, label smoothing ($\epsilon_{\text{label}}=0.1$).
- Model size (base):** ($d_{\text{model}}=512$), 8 attention heads, 6 encoder + 6 decoder layers, ($d_{\text{ff}}=2048$).

- **Training time:** ~3.5 days on 8 P100 GPUs.

4. Results

Task	Metric (BLEU)	Baseline (RNNSearch+)	Transformer (Base)
En→De (WMT 2014)	28.4	24.6	28.4
En→Fr (WMT 2014)	41.0	38.1	41.0

The Transformer **outperforms** strong recurrent baselines while being **significantly faster**:

- **Training speed:** ~3× faster per step than comparable LSTM models (thanks to parallelizable attention).
- **Inference speed:** ~10–30× faster on GPUs for long sequences, because each layer can be computed in parallel across time steps.

Ablation studies confirm that:

- Multi-head attention is crucial (more heads → better performance up to a point).
- Removing positional encodings degrades results dramatically.
- The scaled dot-product formulation stabilizes training; using raw dot-product leads to poor convergence.

5. Impact & Legacy

The Transformer introduced a **new paradigm** for sequence modeling:

- **Universal applicability:** Soon adopted for language modeling (GPT series), BERT-style pre-training, vision (ViT), speech, reinforcement learning, and many multimodal tasks.
- **Scalability:** The architecture scales efficiently to billions of parameters, enabling the current wave of large language models.
- **Research directions:** Follow-up work explored relative positional encodings, sparse attention, efficient Transformers (Longformer, Reformer, Performer), and hybrid encoder-decoder designs.

6. Key Take-aways

1. **Self-attention alone can replace recurrence and convolution** for sequence transduction, provided positional information is supplied.
2. **Multi-head attention** lets the model jointly attend to information from different representation subspaces.
3. The **parallelizable nature** of attention yields dramatic speedups in both training and inference, especially on modern hardware.
4. The Transformer set the foundation for the rapid development of large-scale pre-trained models that dominate NLP (and increasingly other domains) today.

In a nutshell: *Attention Is All You Need* demonstrates that a stack of self-attention and feed-forward layers, equipped with positional encodings, can achieve state-of-the-art translation quality while being far more parallelizable and efficient than recurrent architectures, thereby reshaping the landscape of deep learning for sequential data.



What is the difference between Generative AI and Agentic AI ?



Generative AI vs. Agentic AI – what sets them apart?

Aspect	Generative AI	Agentic AI (AI agents / “agentic AI”)
Core purpose	Produce new artefacts – text, images, audio, video, code, etc. – that resemble data it was trained on.	Act autonomously to achieve goals, make decisions, and interact with an environment (digital or physical).
Typical architecture	One-shot or few-shot generative models (e.g., diffusion models, large language models, GANs, autoregressive transformers).	A pipeline that often includes: 1. Perception (input parsing, sensor data). 2. Reasoning / planning (LLM, symbolic planner, reinforcement-learning policy). 3. Action (API calls, tool use, robot actuation). 4. Memory / state management .
Output	A static piece of content (a paragraph, an image, a piece of code).	A sequence of actions over time, possibly producing multiple pieces of content along the way.
Autonomy	Generally reactive : you give a prompt, it returns a result. No ongoing self-directed behavior.	Proactive : decides what to do next, may self-initiate queries, schedule tasks, or move in a physical space.
Goal orientation	Implicit – the “goal” is to match the distribution of the training data or follow a prompt.	Explicit – an agent is given a goal (e.g., “book a flight”, “optimize a warehouse layout”) and works toward it, often with sub-goals and feedback loops.

Aspect	Generative AI	Agentic AI (AI agents / "agentic AI")
Interaction with tools	Usually self-contained (the model generates the answer internally). Some "tool-augmented" LLMs blur the line, but the primary output is still content.	Tool-using by design: calls APIs, runs shell commands, manipulates files, controls robots, etc. The agent can chain many tools together.
Memory & state	Stateless or short-term context (the prompt window).	Can maintain long-term memory , session state, or external knowledge bases, allowing it to build on past actions.
Examples	• ChatGPT, Claude, Gemini when used only for conversation or code generation. • DALL-E, Stable Diffusion, Midjourney (image generation). • GitHub Copilot (code suggestion).	• Auto-GPT / Baby-AGI (LLM that loops through "think → act → observe"). • ReAct agents that retrieve information, reason, and execute commands. • Household robots (e.g., iRobot with planning modules). • Virtual assistants that schedule meetings, order groceries, and follow up on tasks.
Evaluation metrics	Fidelity, diversity, realism, alignment with prompt (BLEU, FID, human rating, etc.).	Task success rate, cumulative reward, number of steps to goal, safety/robustness, resource efficiency.
Research community	Generative modelling (diffusion, GANs, autoregressive LMs).	Multi-agent systems, reinforcement learning, planning, tool-use, embodied AI, "compound AI" research.

How they relate

- Agents often *use generative models* as a component.
 - An AI agent might call a language model to draft an email, then use an API to send it.
 - A robot may generate a description of a scene with a vision-to-text model, then plan a motion based on that description.
- Generative AI can be "agent-like"** when paired with external tools (e.g., "function-calling" in modern LLMs). This blurs the line, giving a generative model the ability to act, but the **agentic behavior** still comes from the orchestration layer that decides *when* and *how* to call those tools.
- Scope of ambition** – Generative AI aims at *creative output*; Agentic AI aims at *goal-directed execution* in dynamic settings.

Quick-look analogy

Analogy	Generative AI	Agentic AI
Chef	A recipe-generator that writes a dish description.	A chef that reads the recipe, gathers ingredients, cooks, tastes, and adjusts until the dish is ready.
Writer	A language model that drafts a paragraph.	An author that outlines a plot, writes chapters, edits, researches facts, and decides when to submit.

Bottom line

- Generative AI** = "content creator." It excels at producing high-quality, novel artefacts from prompts.
- Agentic AI** = "goal-oriented actor." It couples perception, reasoning, memory, and tool use to autonomously achieve tasks, often **leveraging** generative models as sub-components.

Understanding the distinction helps when designing systems: if you only need a high-quality output, a pure generative model may suffice; if you need a system that *decides, plans,* and *acts*

Ask me anything...

