AIM 2: Artificial Intelligence in Medicine II

Harvard - BMI 702 and BMIF 203, Spring 2026

Lecture 2: Embeddings and their role in NLP, Transformers and BERT, Clinical BERT, Encoder-only and decoder-only architectures, Medical question-answering, Human-AI evaluation loops.

HARVARD
MEDICAL SCHOOL

Review From Last Week

What Makes Biomedical Data Unique

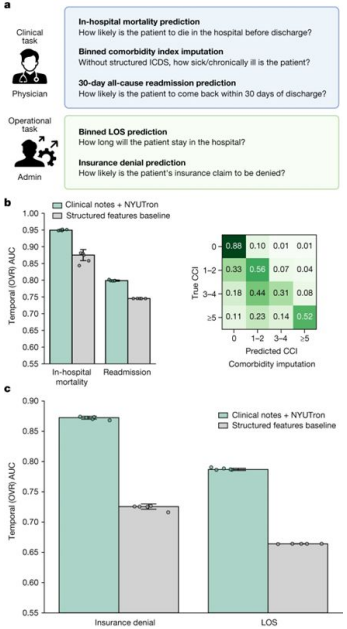
- High stake → High Standards
- Unsupervised in Nature and lack labels
- Underlying Causal Framework
- Trustworthy
- Immense Class Imbalance (outlier predictions)
- Inconsistent Measurements
- Medical Culture is hard to change

High-stakes decisions

10^8 molecular compounds

Graph learning

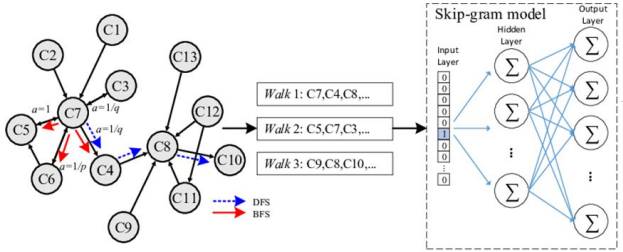
Search of new antibiotics

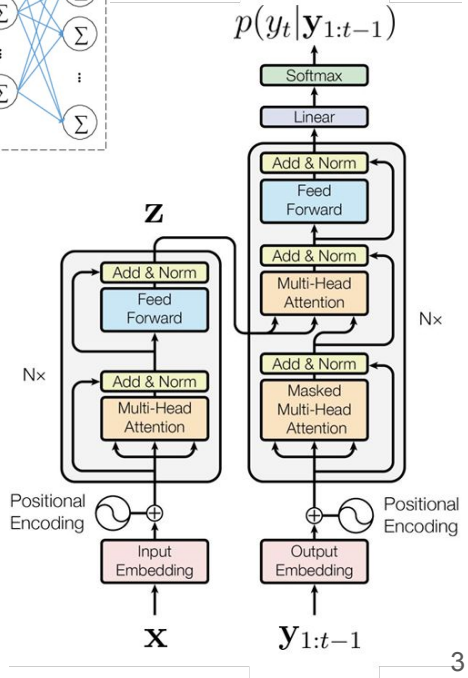


Review From Last Week

Intro to Distributed Language Representation

- Distributional Hypothesis
- Syntax, Grammar, and Semantics
- Probabilistic Models of Language
- Word2vec, node2vec, sentence2vec, and many more
- Transformer Architecture

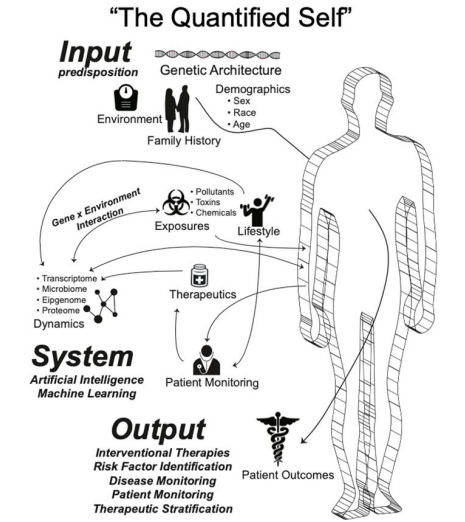


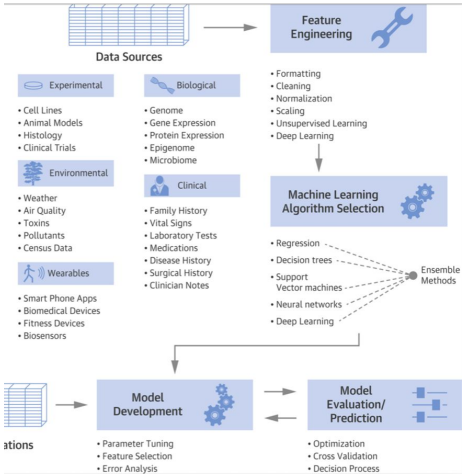


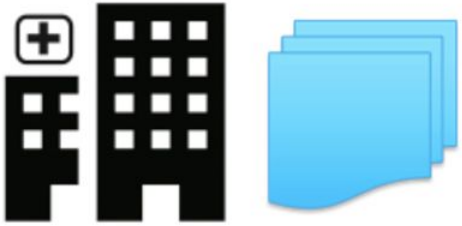
Review From Last Week

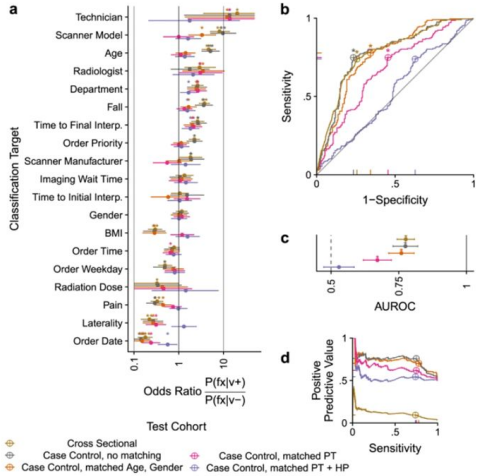
Intro to NLP in Clinical Settings

- Multi Level Dataset which doesn't exist
- EHR: data types, structure
- ML Workflow for EHR data
- How well do various data types define disease?
- Challenges:
  - Simplify patients too much
  - Learn the wrong information
  - Cheat
  - Generalizability



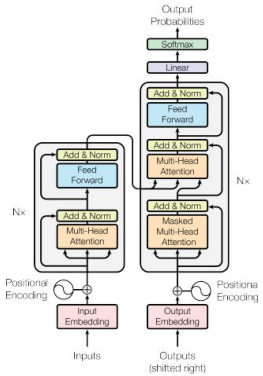


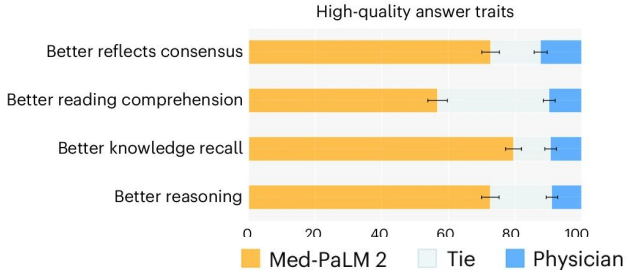




This Week's Lecture Agenda:

1. Embeddings and their role in NLP
2. Transformers, BERT, & Clinical BERT
3. Stack Encoders and Stack Decoders
4. Real-World Applications & Challenges
    a. Medical question-answering
    b. Human-AI evaluation loops





Embeddings and their role in NLP

[The, “Patient”, is, …]

Word2Vec

[The, “Patient”, is, …]

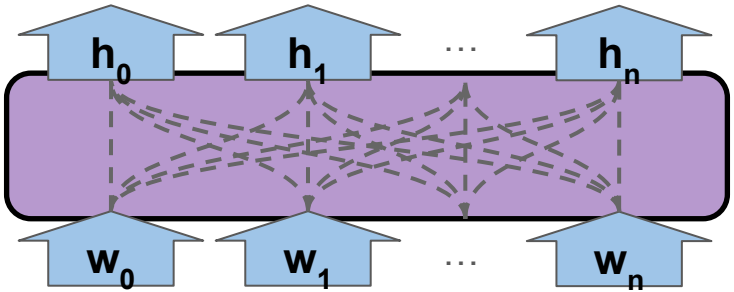
ELMo (RNNs)

Masked Language Modeling (MLM)

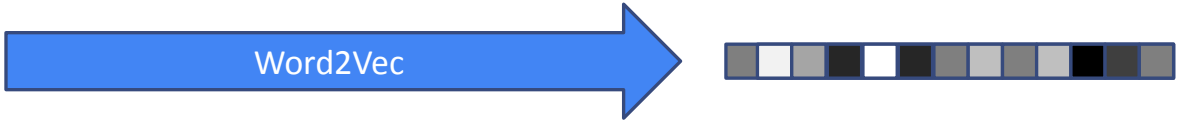
[The, [MASKED], was, …] □ Answer: Mammogram

[The, “Patient”, is, …]

BERT



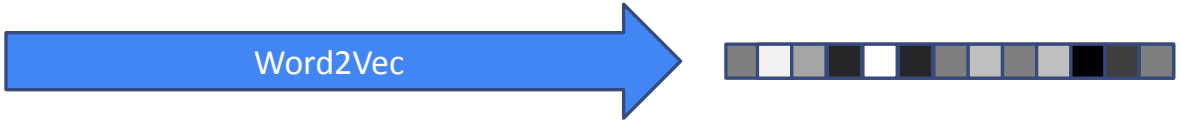
[The, “Patient”, is, …]



- Introduced by **Mikolov et al. (2013)** at Google, Word2Vec was a major shift in NLP, replacing sparse representations with **dense vector embeddings**.
- Prior methods:
  a. **One-hot encoding** (binary vectors, no meaning).
  b. **TF-IDF** (word frequency-based, ignores context).
  c. **Latent Semantic Analysis (LSA)** (matrix factorization, computationally heavy).
- Word2Vec learns **word meanings from co-occurrence**, using **two architectures**:
  a. **CBOW (Continuous Bag of Words)** – Predicts a word from its context.
  b. **Skip-gram** – Predicts surrounding words from a central word.
- These models use **shallow neural networks** with **weight matrices as word vectors**.

🔎 **Key Insight:** Similar words have **similar vectors**, allowing for analogy reasoning (e.g., `"king - man + woman ≈ queen"`).

[The, “Patient”, is, …]



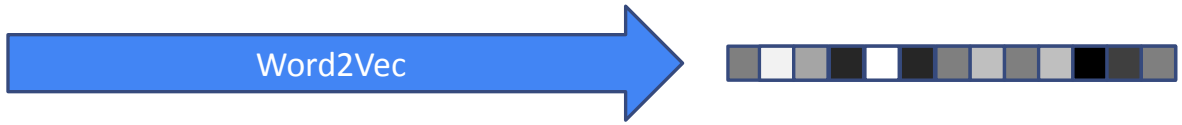
CBOW Objective:

- Given a context window $C = \{w_{t-k}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+k}\}$, the model predicts the target word w_t.
- **Softmax probability of target word:**

$$P(w_t|C) = \frac{e^{v_{w_t}^T u_C}}{\sum_{w \in V} e^{v_w^T u_C}}$$

  - v_w = output vector of word w.
  - u_C = averaged input context vector.
  - V = vocabulary size (large, making softmax expensive).

[The, “Patient”, is, …]



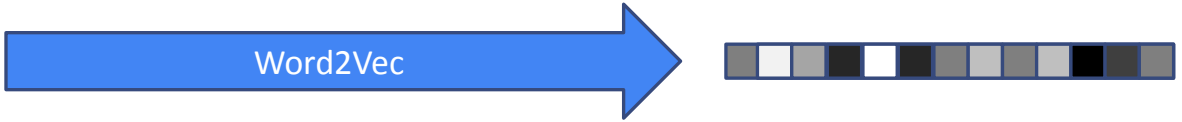
Skip-gram Objective:

- Given a target word w_t, predict surrounding words w_c.
- **Probability of context words given w_t:**

$$P(w_c|w_t) = \frac{e^{v_{w_c}^T u_{w_t}}}{\sum_{w \in V} e^{v_w^T u_{w_t}}}$$

- Optimized using **negative sampling** (reduces computational cost from full softmax).

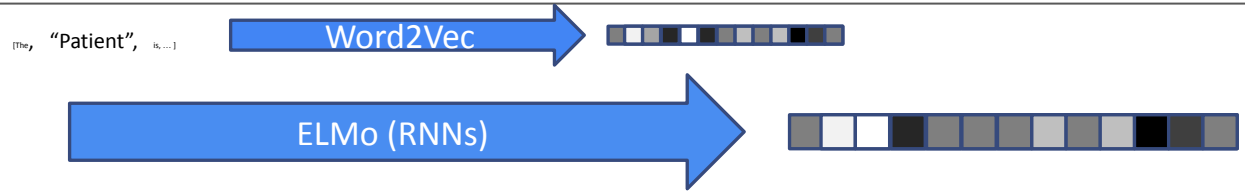
[The, “Patient”, is, …]



- **Embeddings are learned as row vectors in matrices W and W'.**
- **Cosine similarity** measures word similarity:

$$\cos(\theta) = \frac{v_{w_1} \cdot v_{w_2}}{\|v_{w_1}\| \|v_{w_2}\|}$$

Impact: Word2Vec led to **context-aware embeddings**, inspiring later models like GloVe, FastText, and Transformer-based embeddings (BERT).



- Introduced by **Peters et al. (2018)** at the Allen Institute for AI, **ELMo (Embeddings from Language Models)** improved word embeddings by making them **context-sensitive**.
- Unlike Word2Vec & GloVe (static embeddings), ELMo generates **dynamic embeddings** that change based on sentence context.
- Uses a **bidirectional LSTM (BiLSTM)** trained on a **language modeling task**:
  - **Forward LSTM** predicts next word.
  - **Backward LSTM** predicts previous word.
- Concatenates hidden states from **both LSTMs** to get **contextual word embeddings**.

Example:

- `"He wore a **mask** to the **hospital**."` → `"mask"` = medical meaning.
- `"She wore a **mask** to the **party**."` → `"mask"` = costume meaning.

[The, "Patient", is, ...] Word2Vec

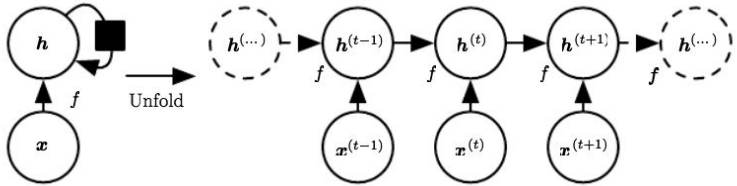
[The, "Patient", is, ...] ELMo (RNNs)

BiLSTM Model:

- ELMo computes embeddings from **stacked BiLSTM hidden states**.
- Given a sequence $(w_1, w_2, ..., w_T)$, computes hidden states:

$$\overrightarrow{h_t} = LSTM_f(w_1, ..., w_t)$$

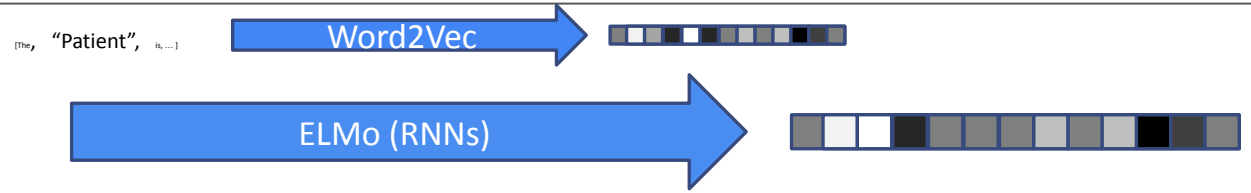
$$\overleftarrow{h_t} = LSTM_b(w_T, ..., w_t)$$



- Final ELMo embedding is a **weighted sum of hidden layers**:

$$ELMo(w_t) = \gamma \sum_{l=1}^{L} s_l h_t^{(l)}$$

- s_l = learned scalar weights for layer l.
- γ = scaling parameter.



Training Objective:

- Bidirectional Language Model (BiLM):
  - Maximize log likelihood of forward & backward sequences:

$$L = \sum_t (\log P(w_t | w_1, ..., w_{t-1}) + \log P(w_t | w_{t+1}, ..., w_T))$$

Why ELMo Was a Breakthrough:

- **Dynamic embeddings** → captures different meanings based on context.
- **Predecessor to BERT** → but still uses **RNNs instead of transformers**.
- **Outperformed Word2Vec & GloVe on NLP benchmarks.**

Applications: Clinical Name Entity Recognition

Problem
Clinical text is messy. Abbreviations, misspellings, shorthand, local jargon. Rule-based systems and bag-of-words models break constantly.

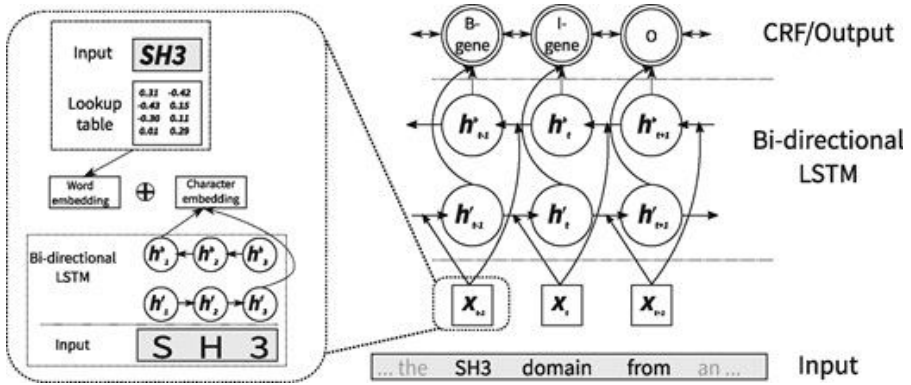
Why embeddings mattered
Word embeddings gave a way to encode *semantic similarity* and *contextual relatedness*:

- “MI”, “myocardial infarction”, “heart attack” land close in vector space
- Models generalize across surface forms
- Much less brittle than dictionaries or regex

This was one of the *first* places embeddings showed clear value in medicine.

Key embedding methods

- word2vec (skip-gram / CBOW)
- Later: ELMo (contextual, but still non-generative)



biLSTM (bidirectional Long Short-Term Memory): builds word vectors from characters + context.

Word embedding: pretrained vector for each word from large text.

CRF (Conditional Random Field): picks the best overall tag sequence.

Habibi, M., Weber, L., Neves, M., Wiegandt, D. L., & Leser, U. (2017). Deep learning with word embeddings improves biomedical named entity recognition. Bioinformatics, 33(14), i37-i48.

Application: Clinical Document & Patient-Level Phenotyping

Problem
Identify patients with a condition (HF, pneumonia, antibiotic use, etc.) from EHR notes. ICD codes are noisy. Rules don't scale.

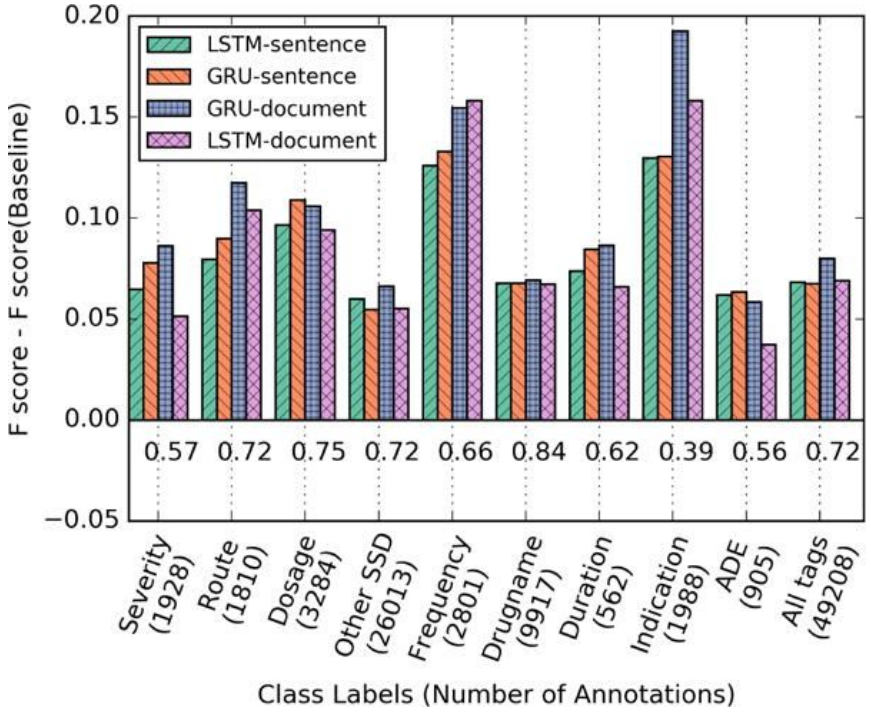
Why embeddings mattered
Embeddings enabled:

- Dense representations of notes or sentences
- Better document-level classifiers
- Semantic retrieval (find similar patients or notes)
- Much better performance with less feature engineering

This is where embeddings quietly replaced TF-IDF.

Key embedding uses

- Averaged word embeddings
- CNN/LSTM over embeddings
- Sentence/document embeddings for retrieval



Jagannatha, A., & Yu, H. (2016, June). Bidirectional RNN for medical event detection in electronic health records. In Proceedings of the 2016 conference of the north American chapter of the association for computational linguistics: Human language technologies (pp. 473-482).

Transformers, BERT, & Clinical BERT

[The, “Patient”, is, …]

Word2Vec

[The, “Patient”, is, …]

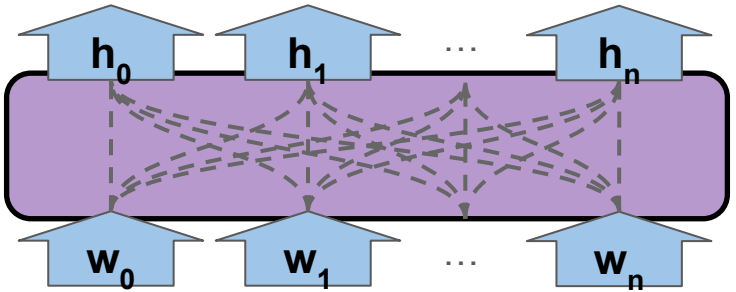
ELMo (RNNs)

Masked Language Modeling (MLM)

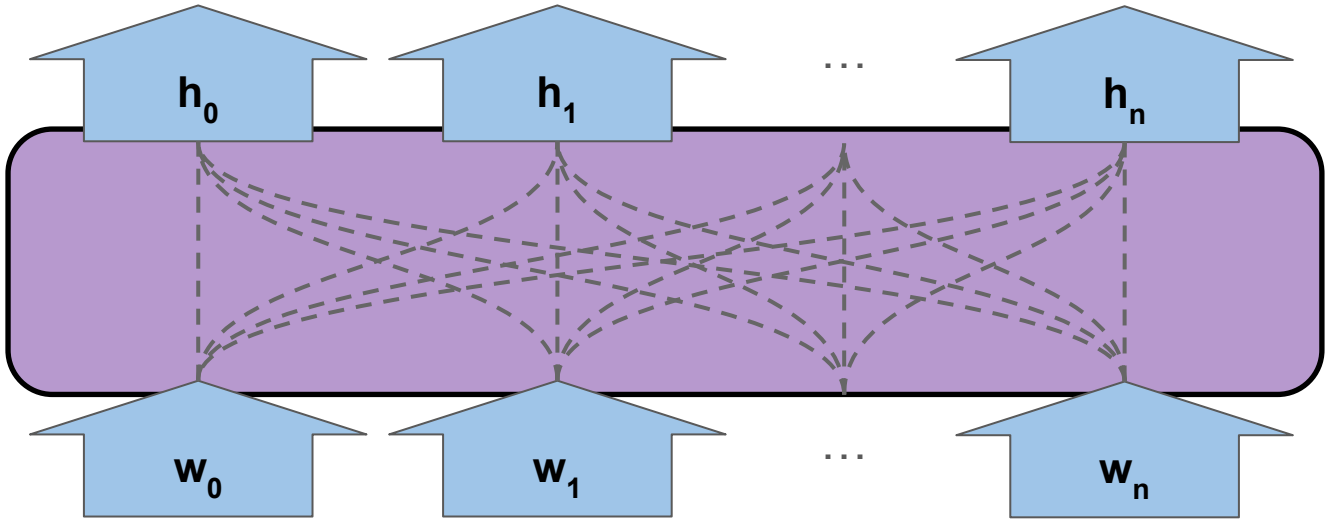
[The, [MASKED], was, …] □ Answer: Mammogram

[The, “Patient”, is, …]

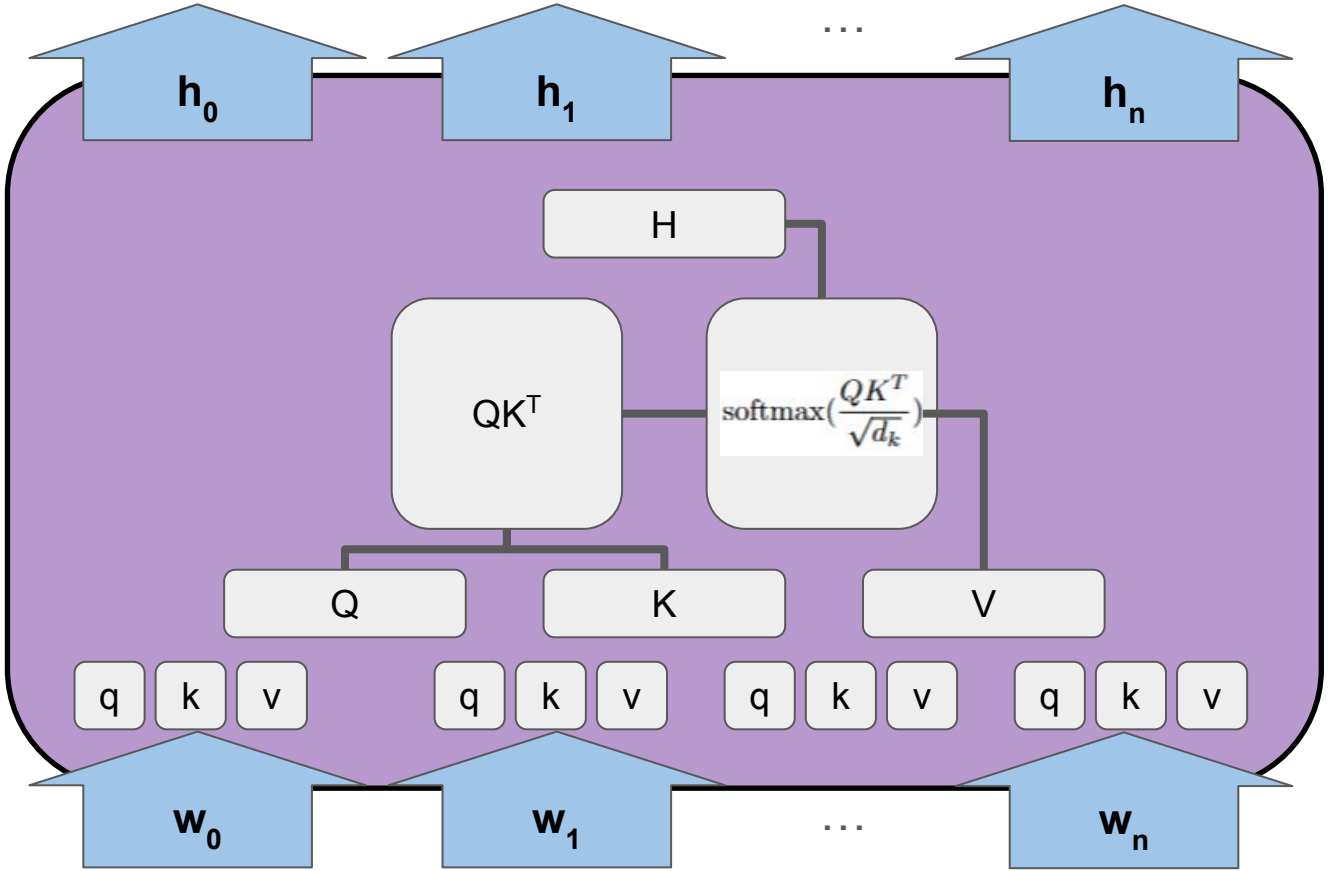
BERT



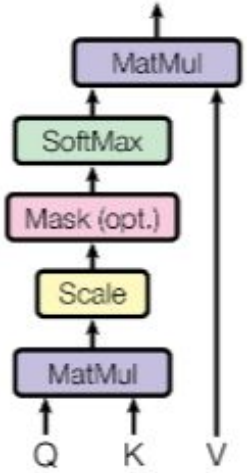
Core Concept of Self-Attention



How Scaled Dot-Product Attention Works

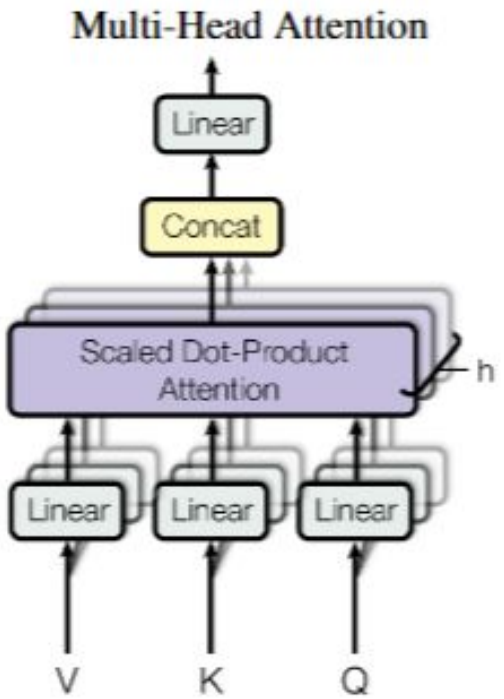


Scaled Dot-Product Attention

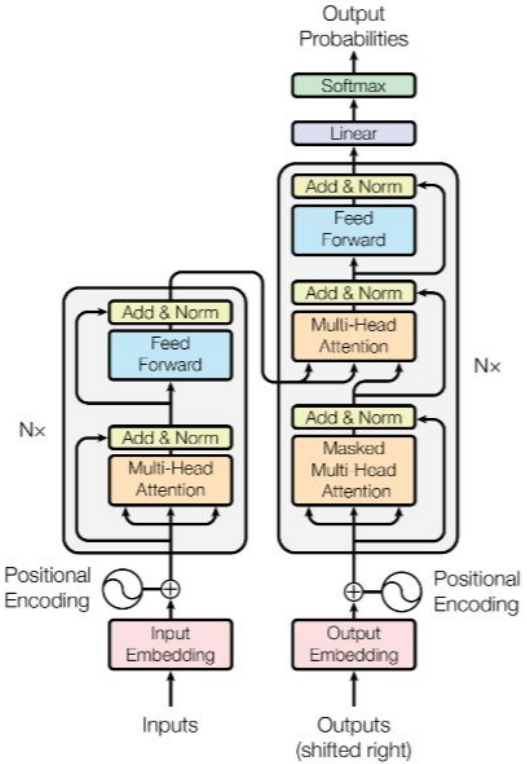


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

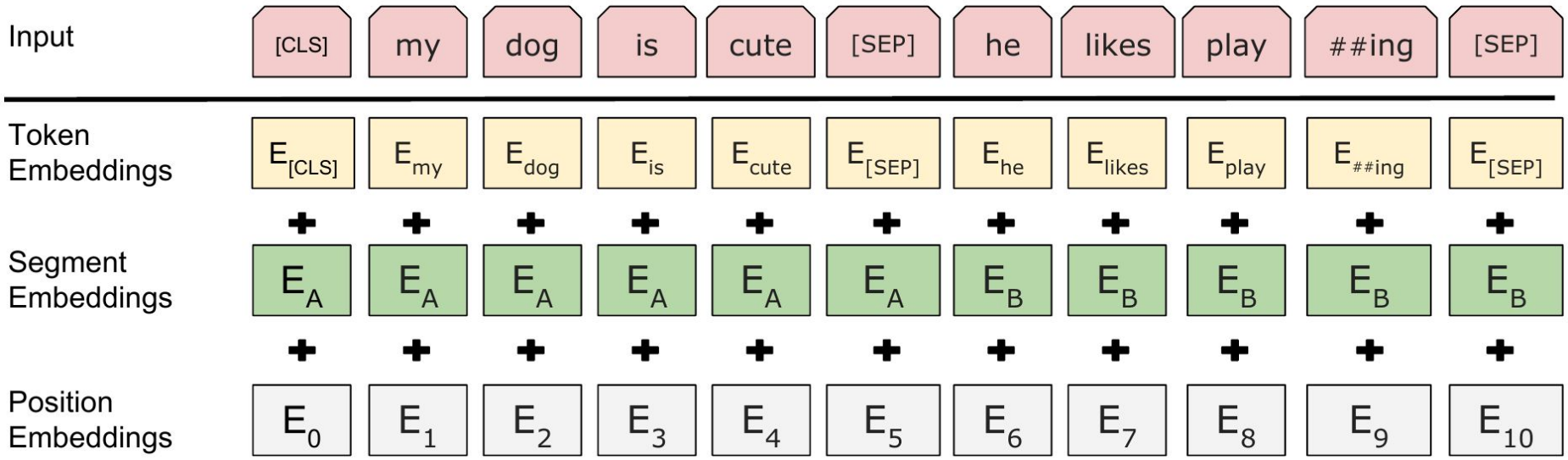
Multi-Head Attention: Diversifying Focus



Transformer Structure: Encoder and Decoder Stacks



Adding Position to Attention: Positional Encoding



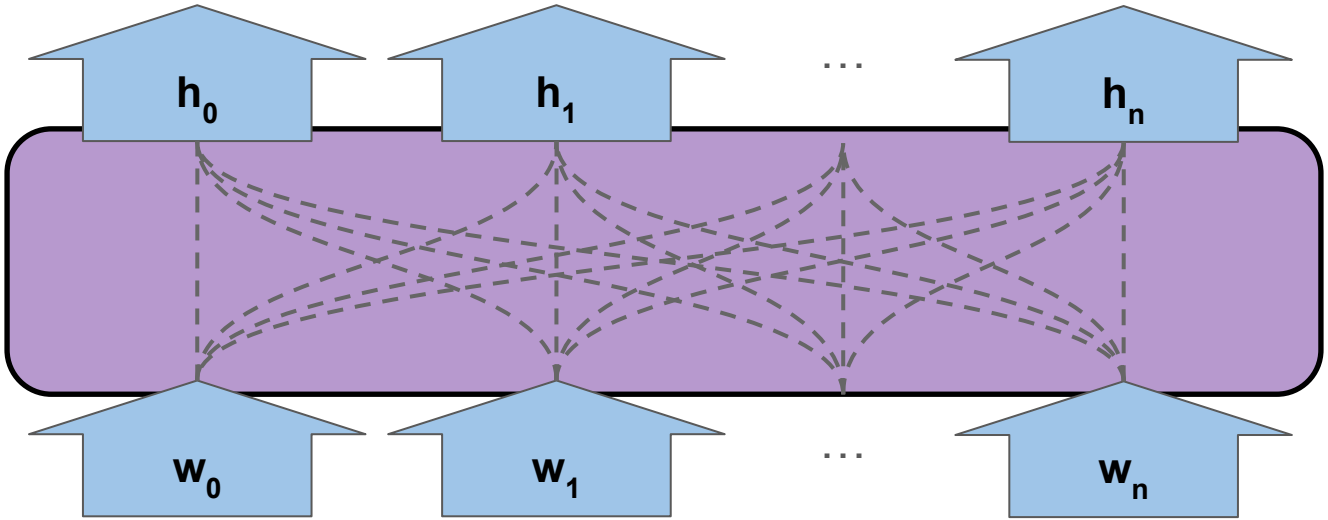
Benefits: Speed and Parallel Processing

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types. n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Applications: Language Translation and Beyond

Hallo, mein Name ist Gray



Hello, My name is Grey

[The, “Patient”, is, …]

Word2Vec

[The, “Patient”, is, …]

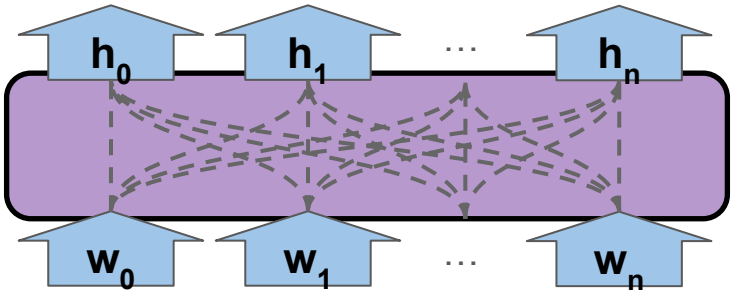
ELMo (RNNs)

Masked Language Modeling (MLM)

[The, [MASKED], was, …] □ Answer: Mammogram

[The, “Patient”, is, …]

BERT



BERT: Bidirectional Encoder Representations from Transformers

- **Introduced by Devlin et al. (2018)**, BERT revolutionized NLP by introducing **deep bidirectional context modeling** using transformers.
- Unlike RNNs, BERT **processes all words simultaneously** using self-attention, allowing full-context understanding.
- **Pre-trained on massive corpora** (Wikipedia + BooksCorpus) using **two self-supervised tasks**:
  1. **Masked Language Model (MLM):** Randomly masks 15% of words and predicts them.
  2. **Next Sentence Prediction (NSP):** Determines if two sentences are consecutive.
- Key innovation: **Contextual embeddings** change based on sentence structure.

Example:

- `"He unlocked the **bank** vault."` vs. `"He sat by the **bank** of the river."` → Same word, different vectors!

Mathematical Formulation of BERT Pre-training

Transformer Architecture:

- Uses **stacked self-attention layers** with query, key, value vectors:

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

  - Q, K, V = learned matrices for queries, keys, and values.
  - d_k = scaling factor.
- **Token embeddings**: Word pieces encoded numerically.
- **Positional embeddings**: Since transformers lack recurrence, a **positional encoding** is added:

$$PE_{(pos,2i)} = \sin(pos/10000^{2i/d})$$

$$PE_{(pos,2i+1)} = \cos(pos/10000^{2i/d})$$

Mathematical Formulation of BERT Pre-training

1. **Masked Language Model (MLM) Loss:**

$$L_{MLM} = -\sum_{i \in M} \log P(w_i | W_{/M})$$

2. **Next Sentence Prediction (NSP) Loss:**

$$L_{NSP} = -\sum_{(A,B)} [y \log P(A|B) + (1-y) \log(1 - P(A|B))]$$

- y = 1 if A, B are consecutive, else 0.

Domain-Specific BERT Models: Clinical BERT & BioBERT

- **BioBERT (2019)**: Trained on **PubMed abstracts + PMC full texts** (over 29M articles).
- **Clinical BERT (2020)**: Fine-tuned on **MIMIC-III clinical notes** to understand **EMRs & clinical language**.
- Differences from standard BERT:
  - **BioBERT** improves biomedical entity recognition & relation extraction.
  - **Clinical BERT** excels in **patient notes analysis** (e.g., de-identification, diagnosis prediction).
- **Key Insight:** General BERT struggles with **medical jargon & abbreviations**.

Example:

- `"The patient has AFib."`
  - **BERT:** Confuses **AFib** with random word.
  - **Clinical BERT:** Knows **AFib = Atrial Fibrillation**.

Fine-Tuning BERT for Medical NLP Tasks

- **Fine-tuning = Adjusting pre-trained BERT weights for specific tasks**.
- Process:
  - **Add a task-specific layer (classifier, NER, QA head, etc.).**
  - **Feed labeled medical data** through BERT (e.g., MIMIC-III, PubMed).
  - **Train on a supervised loss function** (e.g., cross-entropy for classification).
- **Key architectures for fine-tuning:**
  - **NER tasks:** Linear layer on top of token embeddings.
  - **Medical QA:** Encoder-decoder with sequence output.

Example:

- **Clinical Trial Matching:** Fine-tuned BERT predicts **eligibility** for trials from patient notes.

Fine-Tuning BERT for Medical NLP Tasks

- Training **optimizes task-specific loss functions:**
  - **NER (Named Entity Recognition):**

$$L = -\sum_{i=1}^{N} y_i \log P(y_i) + (1 - y_i) \log(1 - P(y_i))$$

  - **Medical Text Classification:**

$$L = -\sum_{i} y_i \log P(y_i|x_i)$$

  - **Medical QA (Span Prediction Loss):**

$$L = -\sum_{i} [\log P(start_i) + \log P(end_i)]$$

  - Uses **Adam optimizer with learning rate decay.**

Stack Encoders and Stack Decoders

Beyond Standard Encoder-Decoder Models

- Standard transformer models (BERT, T5) use **encoder-decoder architectures** for **sequence-to-sequence tasks** (e.g., translation, summarization).
- **Stack Encoder:** Multiple encoder layers **process input at different depths**, allowing better hierarchical representations.
- **Stack Decoder:** Multiple decoder layers **iteratively refine generated output**, improving coherence in text generation.
- **Key difference from standard transformers:** Rather than a **single-pass encoder-decoder**, stack-based models **progressively encode and decode** information.

Example Applications:

- **Machine translation** (deeper meaning capture).
- **Medical report generation** (structured summarization).
- **Chatbot response generation** (coherent multi-turn conversations).

Stack Encoders: Hierarchical Information Processing

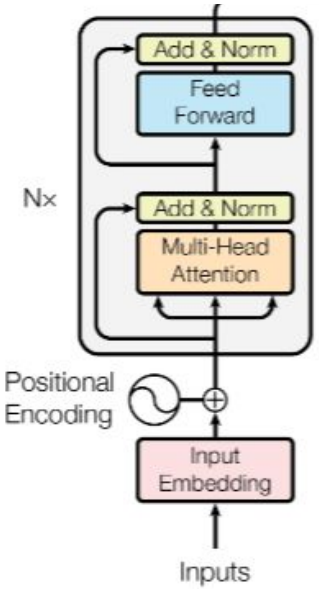
- A **stack encoder** consists of multiple encoder layers:

$$H^{(l)} = f(H^{(l-1)}, A^{(l)})$$

  - $H^{(l)}$ = output at encoder layer l.
  - $A^{(l)}$ = self-attention mechanism at layer l.
  - f = transformation function (feed-forward + attention).
- **Multi-head attention allows deep feature extraction:**

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

  - Each head **extracts different representations** (e.g., syntax, semantics).



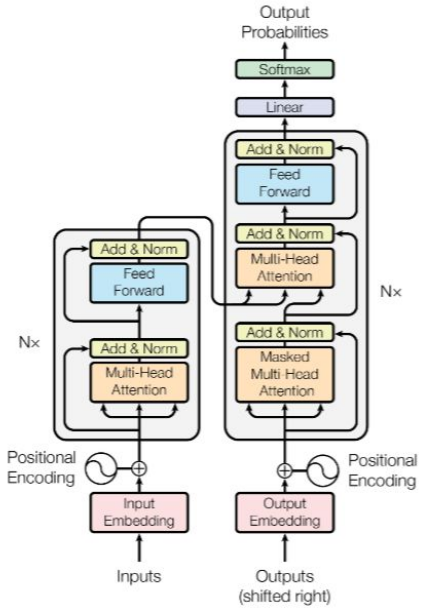
Stacking encoders **enables progressively refined feature representations**, essential for tasks like **medical entity linking & structured text understanding**.

Stack Decoders: Iterative Refinement of Outputs

- A **stack decoder** follows a similar multi-layer structure, where each layer **iteratively refines the generated sequence**:

$$G^{(l)} = g(G^{(l-1)}, C, A^{(l)})$$

  - $G^{(l)}$ = output of decoder layer l.
  - C = encoded representation from stack encoder.
  - $A^{(l)}$ = attention mechanism at layer l.
- **Two types of attention in stack decoders:**
  - **Self-attention** (captures prior generated tokens).
  - **Cross-attention** (links to encoder representations).



The deeper the decoder stack, the more the model **refines generated text**, preventing incoherent responses.

ChatGPT: Transformer Decoders in Action

- ChatGPT (GPT-based models) **use only decoders**—unlike BERT, which uses encoders.
- **Auto-regressive generation:** Predicts one token at a time:

$$P(w_t|w_{<t}) = softmax(W_o h_t)$$

  - h_t = hidden state of transformer at time t.
  - W_o = learned weight matrix.
- **Causal self-attention:** Future tokens are **masked** to prevent looking ahead.
- **Temperature & top-k sampling:** Controls diversity of generated responses.

Why This Works for NLP:

- **Decoders capture long-range dependencies**, making ChatGPT effective at **dialogue generation**.

Training & Fine-Tuning a Generative Model

- ChatGPT's training consists of **pre-training + fine-tuning**:

  1. **Pre-training (Unsupervised):** Trained on massive text datasets using **causal language modeling (CLM)**:

$$L = -\sum_{t} \log P(w_t | w_{<t})$$

  2. **Fine-tuning (Supervised RLHF):** Human annotators rate responses, and reinforcement learning optimizes model behavior:

$$L = -\sum_{t} R(y_t) \log P(y_t | x_t)$$

  - $R(y_t)$ = reward function based on response quality.

Key Innovation:

- Reinforcement learning with human feedback (RLHF) **aligns ChatGPT's responses with human expectations**.

Clinical Applications & Challenges

Medical Question Answering: What it is and why it's hard

Medical QA = answering clinical questions from evidence

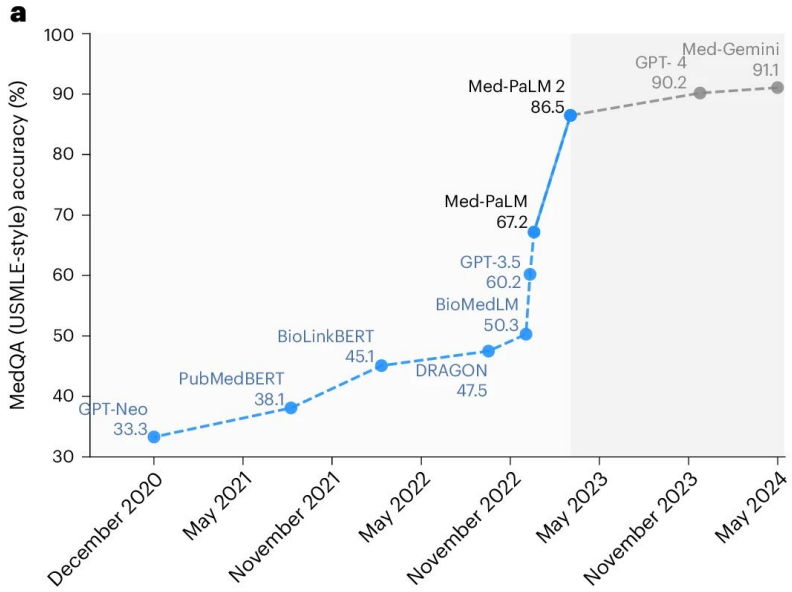
- Inputs: a question + patient context + reference sources (guidelines, UpToDate-style text, papers)
- Outputs: answer + supporting evidence (ideally with citations)

Two main modes

- Retrieval: find the right passage(s)
- Reasoning: connect evidence to a clinically correct conclusion

Why it's harder than "general QA"

- Ambiguity and missing context are common
- Errors are high-stakes (safety)
- Good answers need justification, not just fluency



Singhal, K., Tu, T., Gottweis, J., Sayres, R., Wulczyn, E., Amin, M., ... & Natarajan, V. (2025). Toward expert-level medical question answering with large language models. Nature Medicine, 31(3), 943-950.

Medical QA: Expert-level medical QA with LLMs

What they built

- **Med-PaLM 2**: a medically tuned LLM (PaLM 2 base) + prompting refinements

How they evaluate "medical QA"

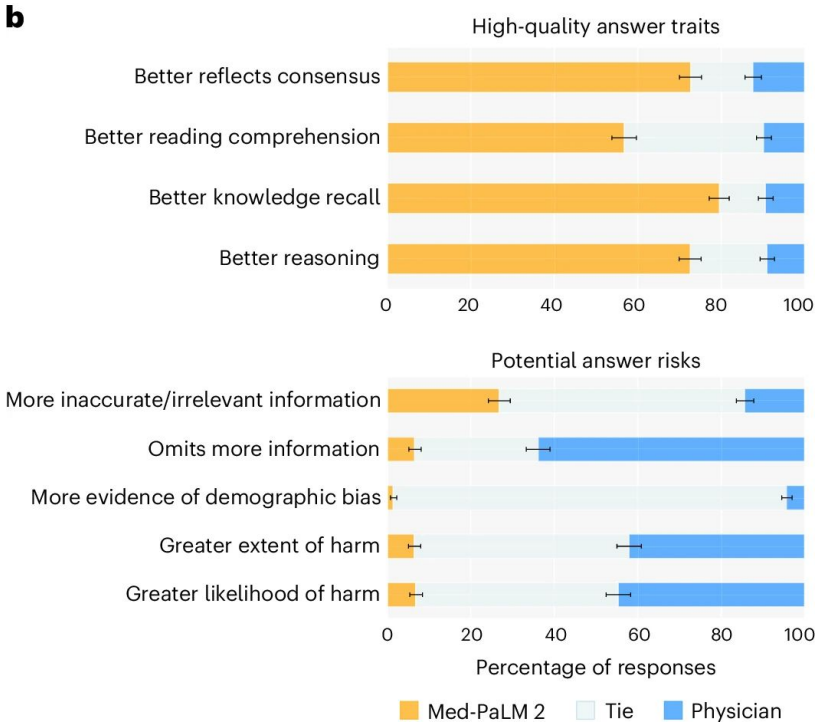
- **Multiple-choice exams** (MedQA / USMLE-style)
- **Long-form consumer questions** graded by physicians across clinical-utility axes

Headline results

- **86.5%** on MedQA (vs **67.2%** Med-PaLM)
- On **1,066** consumer questions, physicians preferred Med-PaLM 2 over physician answers on **8/9** axes

Teaching point

- Benchmarks show knowledge, but they emphasize **human rubric-based evaluation** for safety/utility



Singhal, K., Tu, T., Gottweis, J., Sayres, R., Wulczyn, E., Amin, M., ... & Natarajan, V. (2025). Toward expert-level medical question answering with large language models. Nature Medicine, 31(3), 943-950.

Human–AI Evaluation Loops

Human–AI loop = humans are part of the system, not just the end user

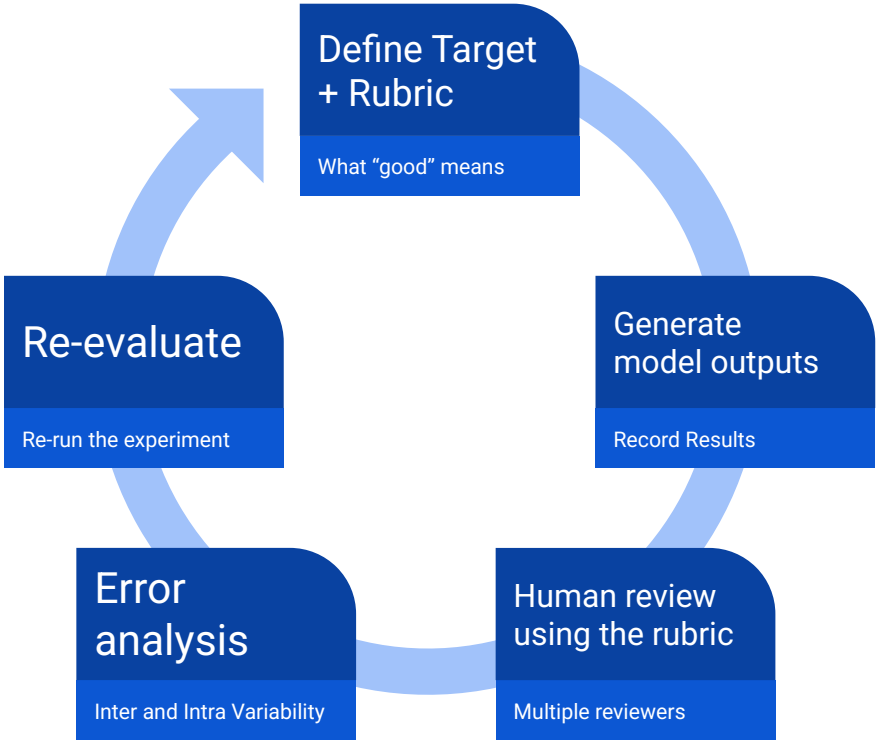
- Humans define rubrics, audit errors, and recalibrate model behavior
- Evaluation isn't one-time; it's iterative

Why medicine needs this

- "Ground truth" is often noisy or subjective (even experts disagree)
- Safety demands monitoring for drift and rare failures
- Fairness: subgroup performance must be checked deliberately

What you measure

- Accuracy + reliability (agreement)
- Calibration (does confidence match correctness?)
- Error types (what fails and why)



Kuling, G., Pullman, S., Vasilev, D., Ozimek, N., Palmer, N., King, R. W., ... & Besche, H. (2025). Assessment of short-answer questions by ChatGPT in a medical school course. NEJM AI, AIcs2500239.

Human–AI Evaluation Loop: LLM grading

Use case

• LLM grades short-answer medical responses using an explicit rubric

• Compare to human graders and quantify agreement

What the loop looks like

• Rubric design → LLM grading → human spot-check → revise prompts/rubric → re-test reliability

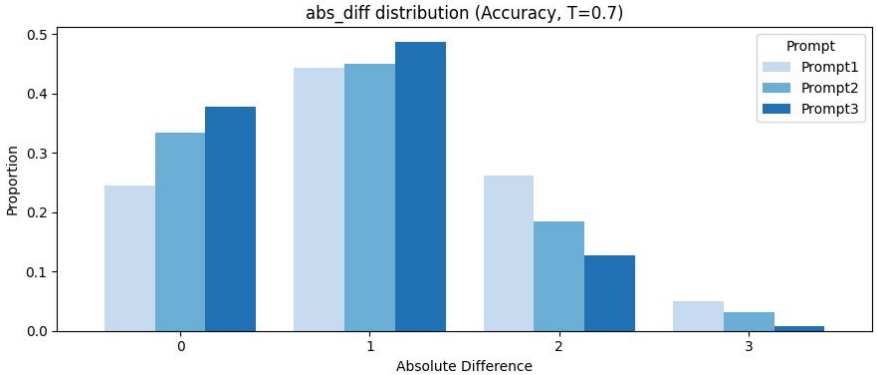
Main insight

• LLM grading can be consistent when constrained by criteria

• Human oversight catches edge cases and keeps standards stable

Why it matters

• Turns evaluation into an operational workflow, not a one-off benchmark



Kuling, G., Pullman, S., Vasilev, D., Ozimek, N., Palmer, N., King, R. W., ... & Besche, H. (2025). Assessment of short-answer questions by ChatGPT in a medical school course. NEJM AI, AIcs2500239.