WELCOME

BERT

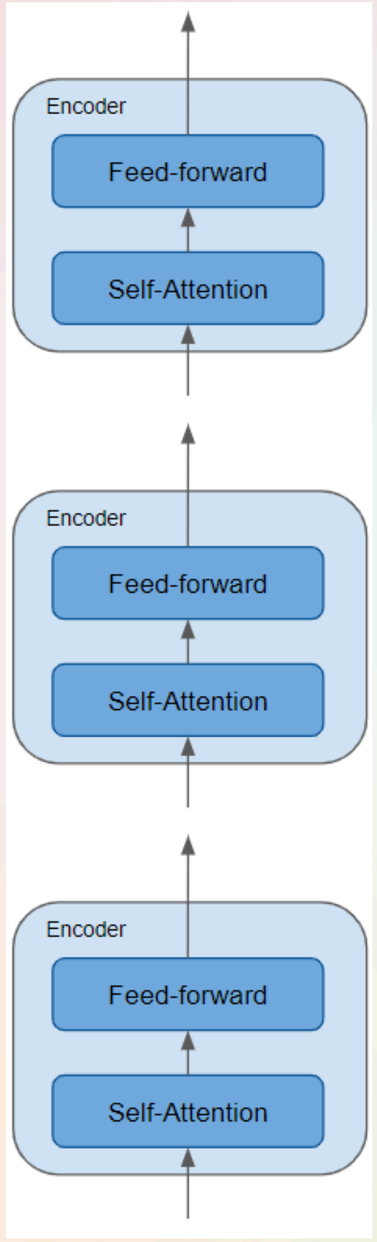
BERT

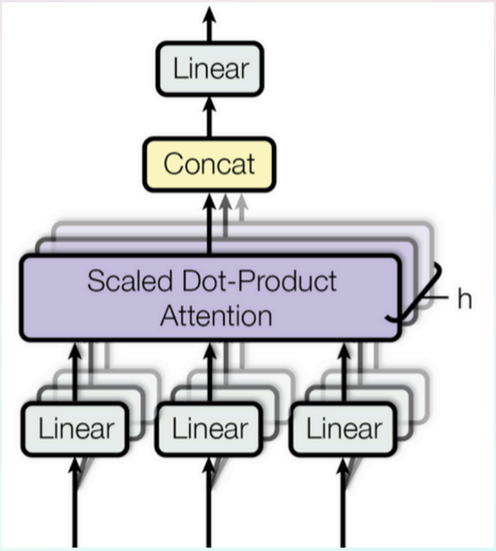
Pre-training of Deep Bidirectional Transformers for Language Understanding

Model's Architecture

BERT-base (L=12, H=768, A=12,110M)
BERT-large(L=24, H=1024, A=16, 340M).

- **L (Layers)**: This represents the number of Transformer blocks in the model
- **A (Self-Attention Heads)**: This is the number of self-attention heads in each self-attention layer within a Transformer block.
- **H (Hidden Size)**: It is the dimensionality of the embeddings used in the model.

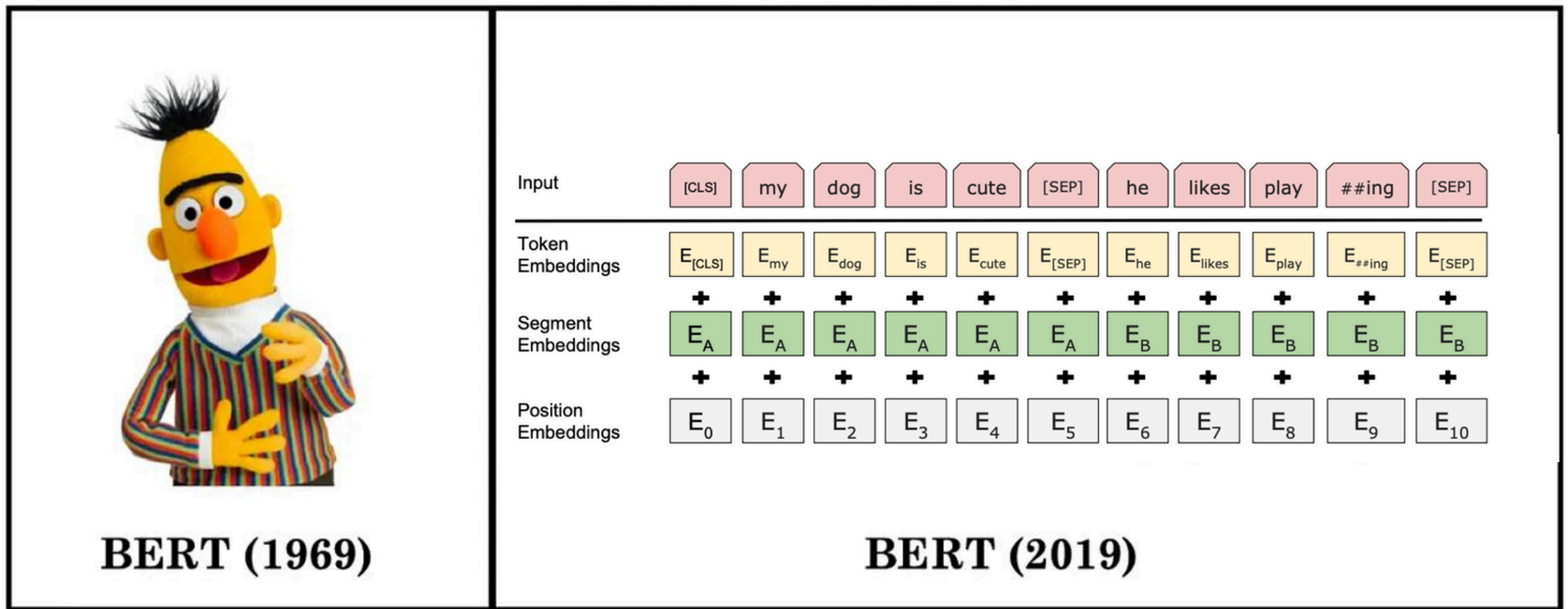




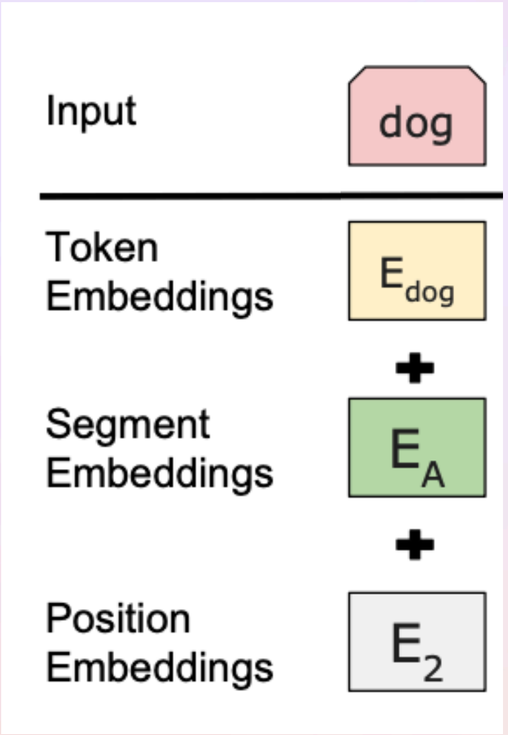
$$x = (x_1, x_2, x_3, \ldots, x_{768})$$

Input/Output representations

- **Tokens** represent the smallest pieces of meaningful data in the text. They can be words or sub units of them.
- **A sentence** is just a span of contiguous text, not necessarily a grammatically correct one.
- **A sequence** is the input of tokens that are fed into the system, might be a sentence or multiple ones.

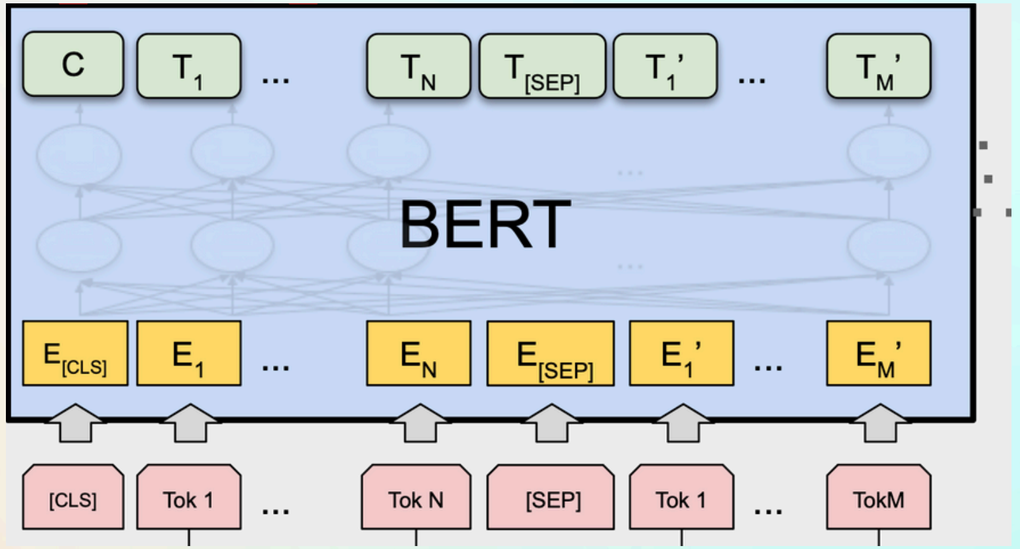


Input/Output representations



- **Token embeddings** are vectors capturing semantic meaning.
- **Segment embeddings** identify the sentence where the token belongs to.
- **Position embeddings** keep track of the position of the element in the sentence, as transofrmers aren't able to do it.

[CLS] and **[SEP]** are special "start" and "separating" vectors. Note that [CLS] is mapped to the final hidden vector **C**, while every token i is mapped to the **vector Ti** .



Pre-training

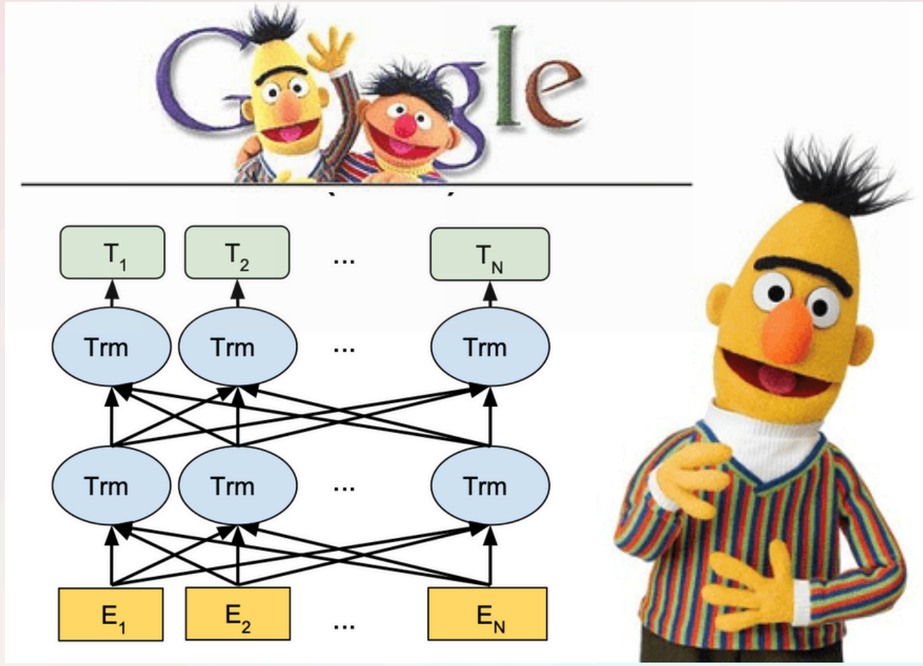
BERT is a deep bidirectional neural network, and what makes it “deep” is that at every layer, every word is compared to the ones in the sequence.

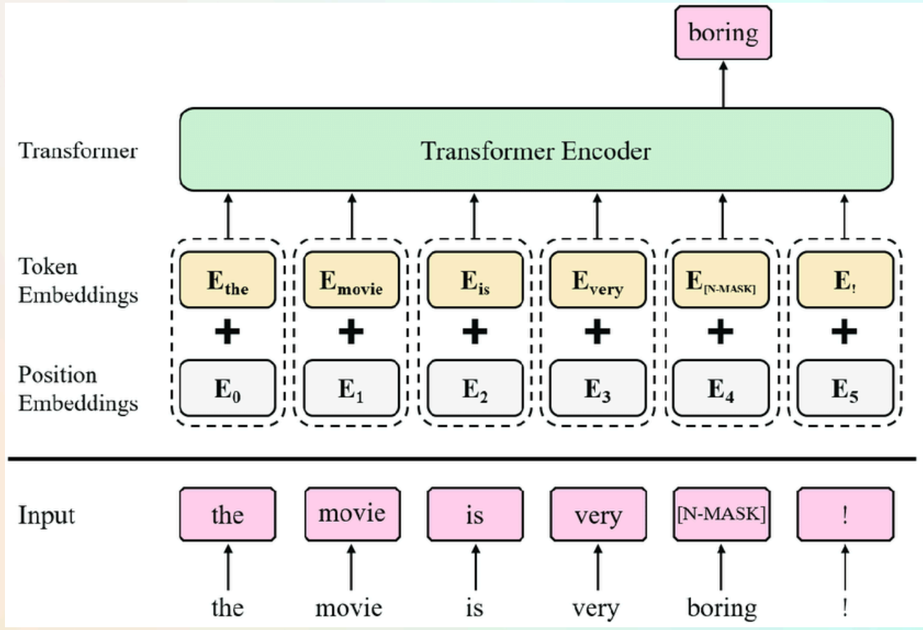
MLM

Masked Language Modeling : consist in masking some tokens (15%) and let the model predict them.

To not harm fine tuning with the masking procedure actually of the 15% of the randomly selected tokens:

- 80% of them are masked.
- 10% are replaced with a random token.
- 10% use the same word.





Pre-training

Ablation on MLM

Ablation is the study of what happens when we remove or modify some components of a model.
Here we show that the amount of randomly replaced words (1.5%) is not harmful for the model.

Masking Rates			Dev Set Results		
MASK	SAME	RND	MNLI	NER	
			Fine-tune	Fine-tune	Feature-based
80%	10%	10%	84.2	95.4	94.9
100%	0%	0%	84.3	94.9	94.0
80%	0%	20%	84.1	95.2	94.6
80%	20%	0%	84.4	95.2	94.7
0%	20%	80%	83.7	94.8	94.6
0%	0%	100%	83.6	94.9	94.6

Table 8: Ablation over different masking strategies.

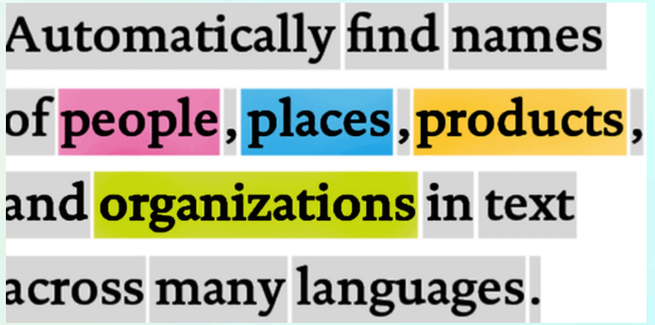
MNLI

MNLI (Multi-Genre Natural Language Inference) is a task where given a **premise** sentence the goal is to determine if another one called **hypothesis**:

- logically follows from it (*entailment*)
- contradicts it (*contradiction*)
- can't conclude it(*neutral*).

NER

(Named Entity Recognition) is a task that involves identifying and **classifying named entities** (such as people, organizations, locations, dates, etc.) in a text.

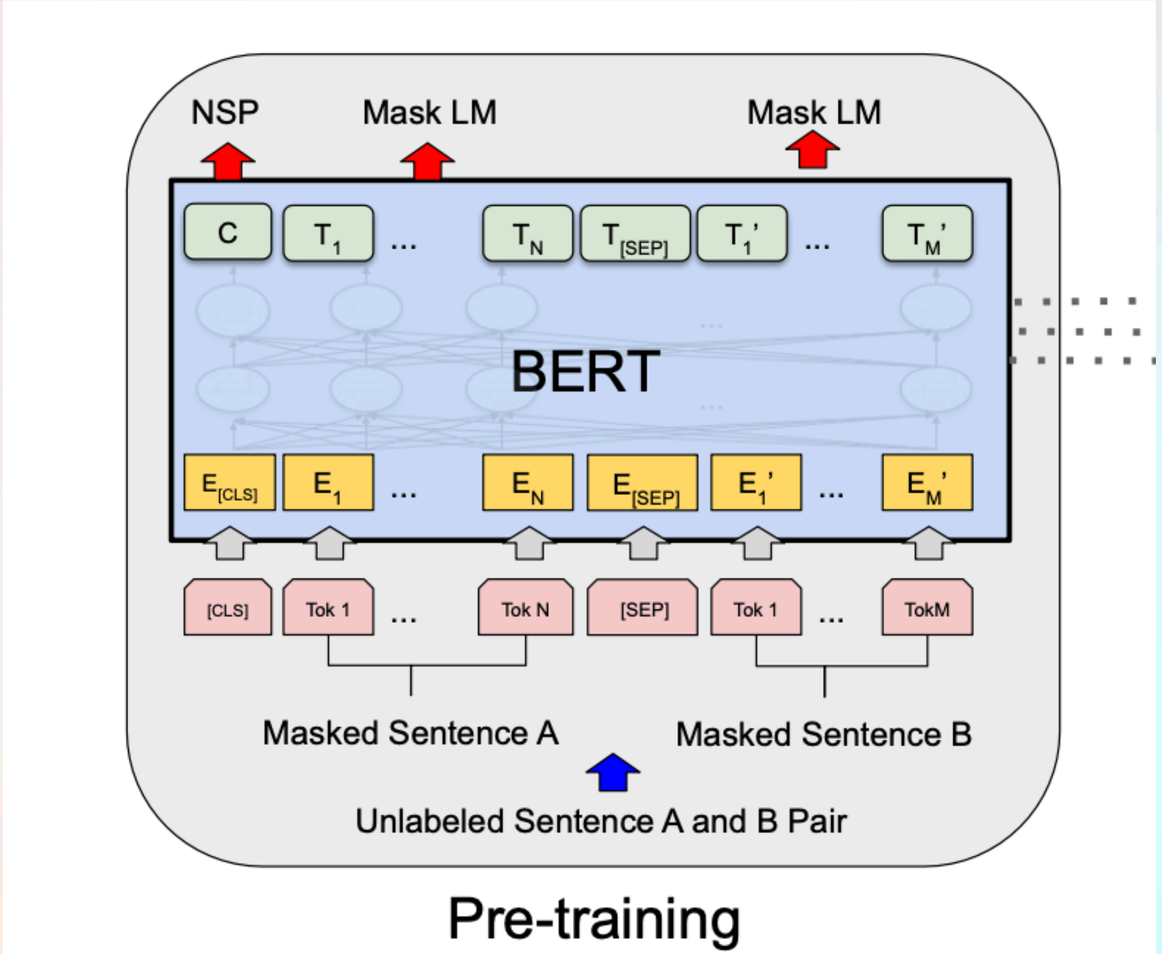


Pre-training

NSP

NSP (**Next Sentence Prediction**) is used as a pre-training task because it helps the model understand the **relation between sentences**, which is crucial in some NLP tasks.

Two random sentences are chosen from the corpus.
The first sentence receives the **A embedding** and the second receives the **B embedding**.
50% of the time B is the actual next sentence that follows A and 50% of the time it is a random sentence.



C (the hidden vector corresponding to the [CLS] token) is used for next sentence prediction (NSP)

Pre-training

Procedure

We need a **loss function** to minimize, and in this case it is the sum of the mean masked LM likelihood and the mean next sentence prediction likelihood.

$$L_{MLM} = -\frac{1}{|M|} \sum_{i \in M} log\left(P_\theta\left(t_i \left| X_{\setminus M}\right.\right)\right)$$

- **M** = set of masked words in the correct order of appearance.
- **X\M** is the set of input tokens X minus the set of masked ones M, which are replaced with the token **[MASK]**.
- The **probability** in the function is the one the model has found in the process. It represents the probability that the i-th word is at position i.

Pre-training

Procedure

$$L_{NSP} = -\frac{1}{|N|}\sum_{j=1}^{N} y_j \log P_\theta\left(y_j = 1 \,|A_j, B_j\right) + (1 - y_j) \log P_\theta\left(y_j = 0 \,|A_j, B_j\right)$$

- **N** = total number of sentence pairs in the batch.
- **yj** is a **boolean variable**, for each index j it has value 0 if Bj does not follow Aj, it has value 1 if Bj follows Aj.
- The **probability** in the function is the one the model has found in the pre-training NSP process. It represents the probability that the the j-th pair (A,B) is made by two consecutive sentences.

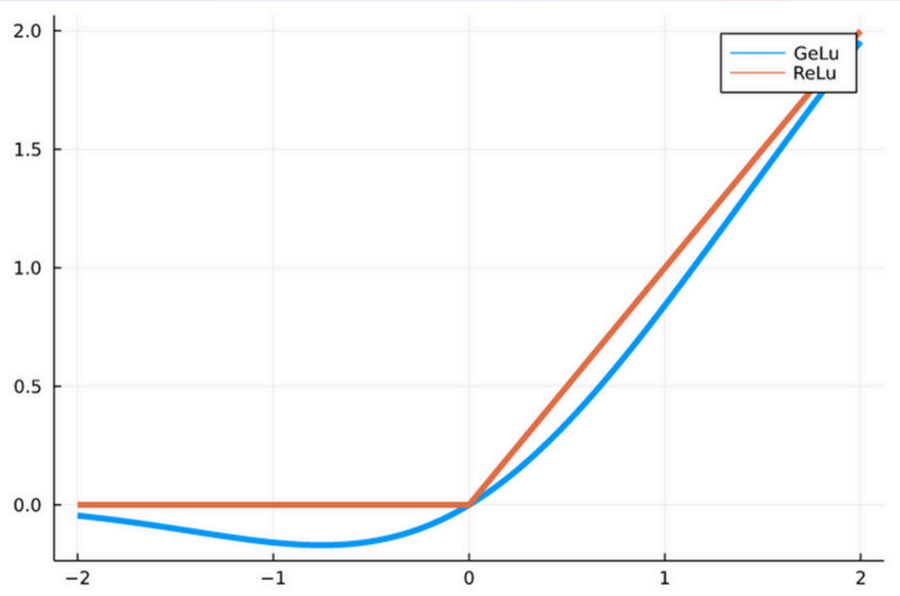
The final loss function is:

$$L = L_{MLM} + L_{NSP}$$

Pre-training

Optimization

The optimization algorithm used is called **Adam**(Adaptive Moment Estimation), and it is considered as a default choice for many ML models.
There is one main difference from the usual approach, which is the **gelu** function instead of the classic *relu* one.



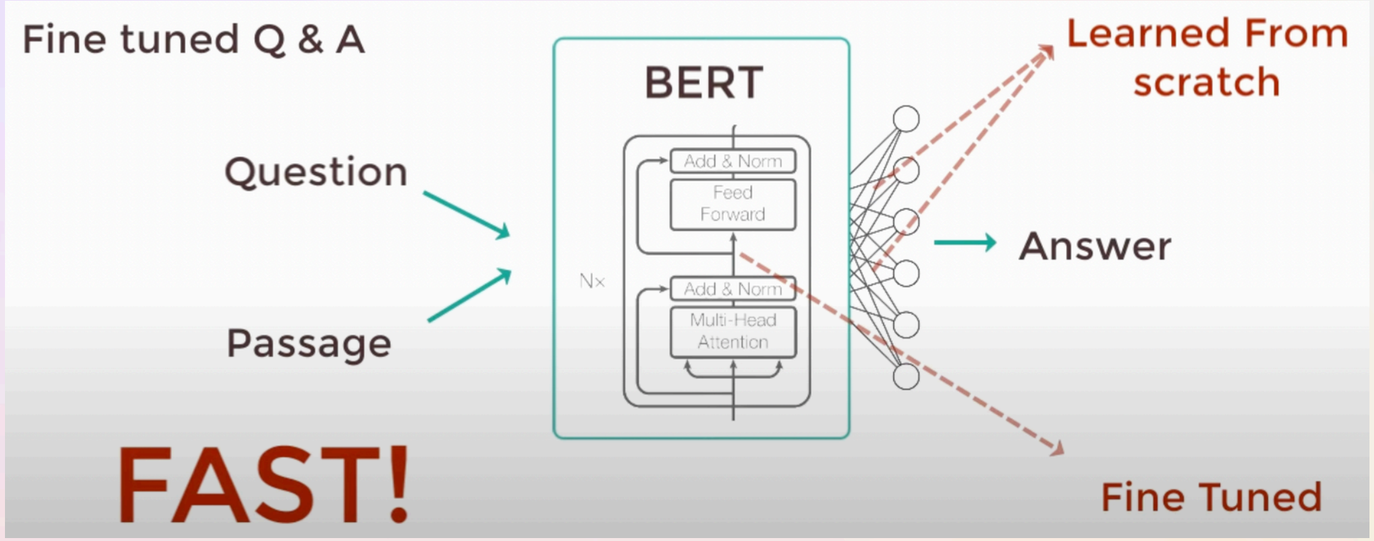
$$GELU = x \times \Phi\left(x\right)$$

Why it is better:

- it provides smoother, more stable gradient flow.
- helps avoid dead neurons.
- Its probabilistic nature makes it more suitable for transformer-based architectures.

Fine Tuning

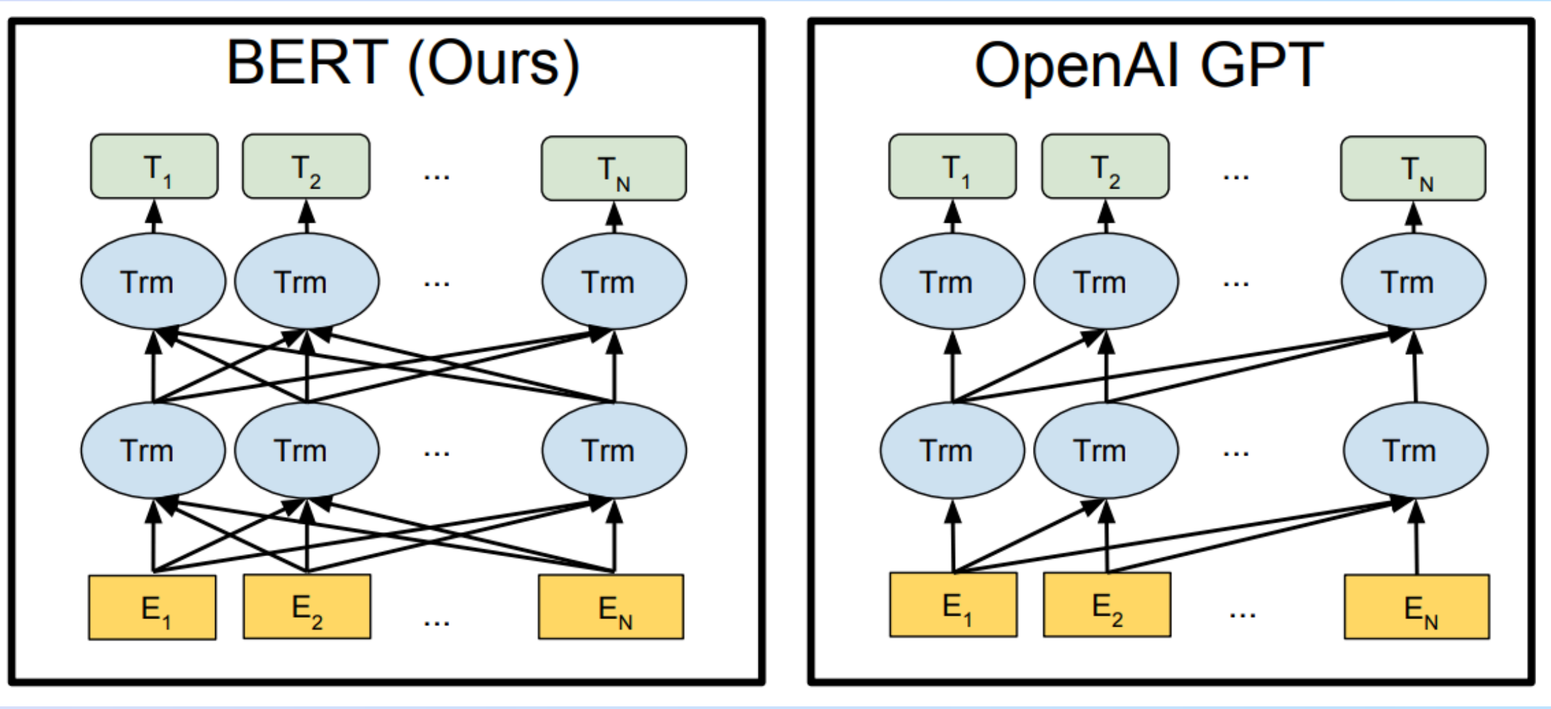
After BERT's pre-trained layers, a **task-specific output layer is added**. This layer contains few parameters that have to be learned from scratch, while it performs just a fine tuning on the pre-learned ones of BERT.

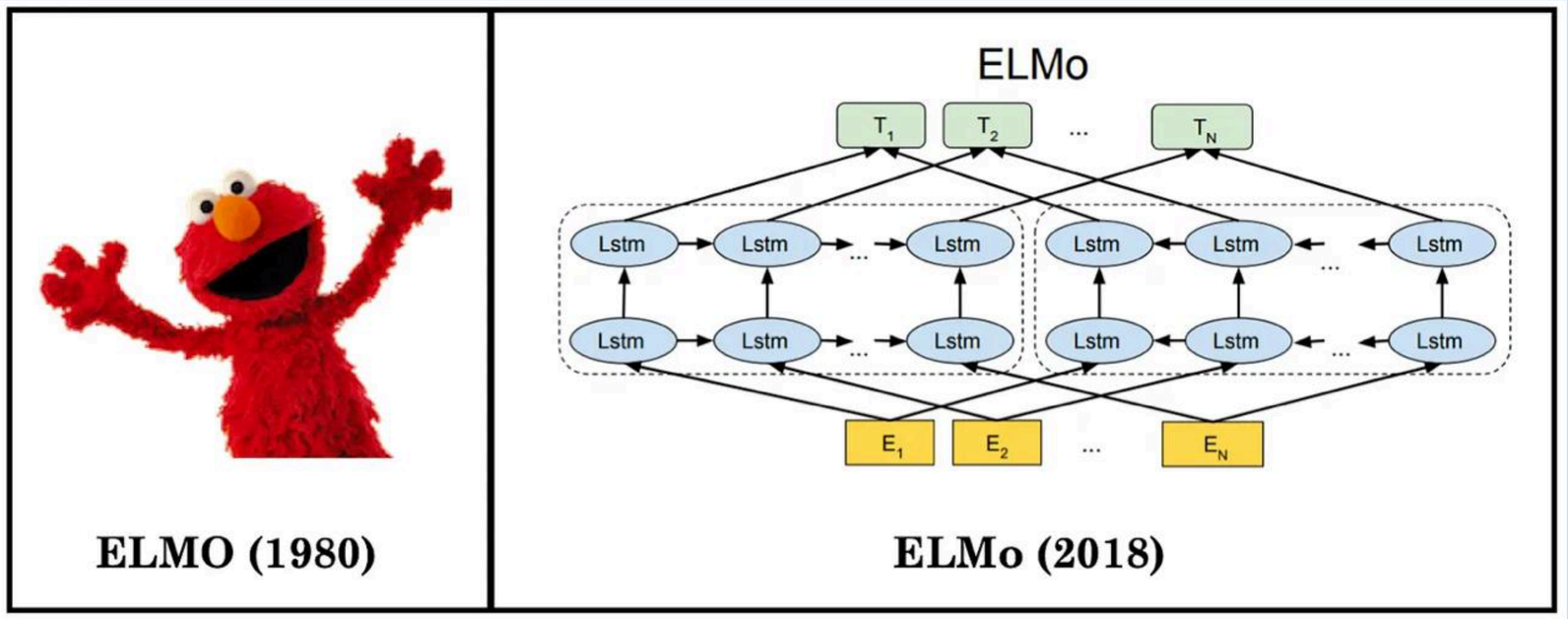


BERT improved **text pair tasks** by unifying encoding and self-attention. Instead of separately encoding the two strings, BERT concatenates them and applies bidirectional cross-attention between the sentences.

Pre-training model architectures

BERT is deeply bidirectional, OpenAI GPT is unidirectional, and ELMo is shallowly bidirectional.





Benchmarks

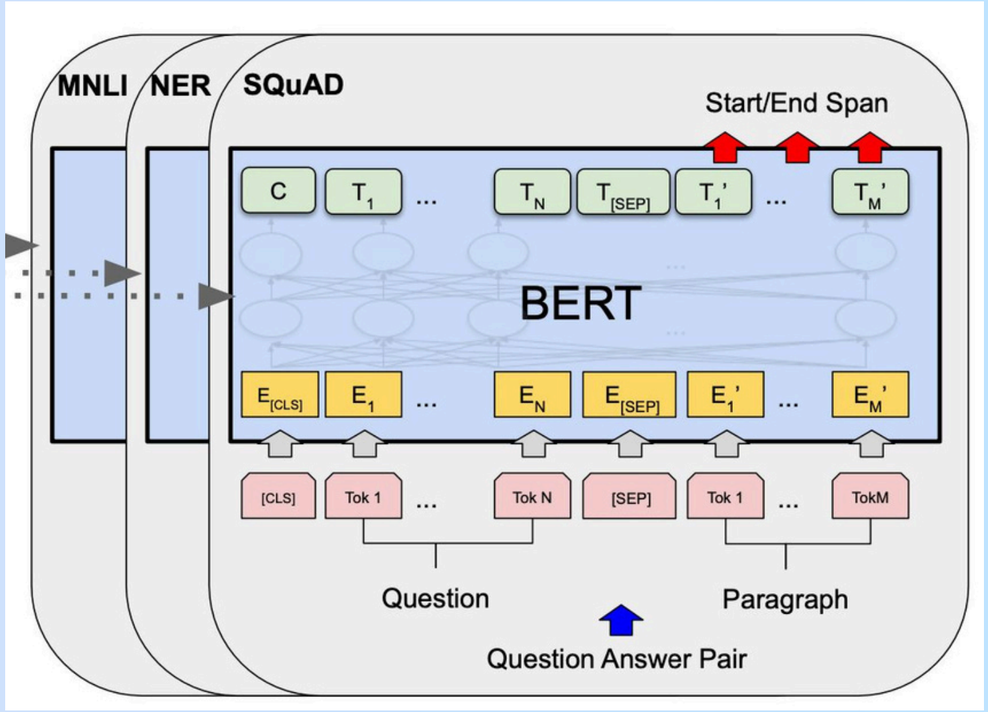
- General Language Understanding Evaluation (GLUE)
- Stanford Question Answering Dataset (SQuAD v1.1)
- SQuAD v2.0
- Situations With Adversarial Generations (SWAG)

SQuAD 1.1

Given a question and a passage containing the answer, predict the answer text span.

- New parameters: **Start vector S**, **End vector E**
- **Probability** of word i being the start of the answer:

$$P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$$ ← Softmax function



- **Score** of a candidate span from position i to j: $S \cdot T_i + E \cdot T_j$

- **Max scoring span** → our prediction
- Training objective: **maximize** the **sum of** the **log-likelihoods** of the correct start and end positions

SQuAD 2.0

- **Extends SQuAD 1.1**
- Allows for **no short answer** to exist in the provided paragraph
- For questions without an answer, **[CLS] token** is both the start and the end of the answer span, so the score is:

$$s_{\text{null}} = S \cdot C + E \cdot C$$

- Score of the best non-null span: $s_{i,j} = \max_{j \geq i}(S \cdot T_i + E \cdot T_j)$
- Predict a non-null answer when:

$$s_{i,j} > s_{\text{null}} + \tau$$

Error term

SWAG dataset

- Given a sentence, the task is to choose the **most plausible continuation among 4 choices**.
- BERT_LARGE outperformed the authors' baseline ELMo system by **+27.1%** and OpenAI GPT by **8.3%**.

A waiter brings a fork. The waiter **a) starts to step away. (74.76%)** b) adds spaghetti to the table. (21.57%) c) brings a bunch of pie to the food (2.67%) d) drinks from the mug in the bowl. (0.98%)	He is up a tree. Someone **a) stands underneath the tree. (97.44%)** b) is at a pool table holding a cup. (1.14%) c) grabs a flower from a paper. (0.96%) d) is eating some cereal. (0.45%)
An old man rides a small bumper car. Several people a) get in the parking lot. (76.58%) b) wait in the car. (15.28%) **c) get stuck with other bumper cars. (6.75%)** d) are running down the road. (1.39%)	He pours the raw egg batter into the pan. He a) drops the tiny pan onto a plate. (93.48%) **b) lifts the pan and moves it around to shuffle the eggs. (4.94%)** c) stirs the dough into a kite. (1.53%) d) swirls the stir under the adhesive. (0.05%)

Ablation studies

Tasks	MNLI-m (Acc)	SQuAD (F1)
$\text{BERT}_{\text{BASE}}$	84.4	88.5
No NSP	83.9	87.9
LTR & No NSP	82.1	77.8

Effect of **Pre-training Tasks**:

- **No NSP** : Bidirectional model trained using the “masked LM” but without the “next sentence prediction” task has a significantly worse performance on **MNLI** and **SQuAD 1.1**.
- **LTR & No NSP** (like OpenAI GPT): Bidirectional representations outperform left-to-right models on all tasks.
- Training **separate left-to-right and right-to-left models** (like ELMo) is less efficient and less powerful.

Ablation studies

Effect of **Number of Training Steps**:

- MLM pre-training **converges slightly slower** than LTR pre-training, **BUT** it **outperforms** the LTR model **in absolute accuracy** almost immediately.
- For high fine-tuning accuracy, a large amount of pre-training (128,000words/batch * 1M steps = **128B words processed**) is necessary for BERT.

Ablation studies

Effect of **Model Size**:

- **Larger models lead to a strict accuracy improvement**.
- Scaling to extreme model sizes leads to large improvements not only on large-scale, but **also very small scale tasks**.
- BERT_LARGE contains **340M parameters**, while the newest models like GPT-4 and PaLM have scaled up to **hundreds of billions of parameters**.

Ablation studies

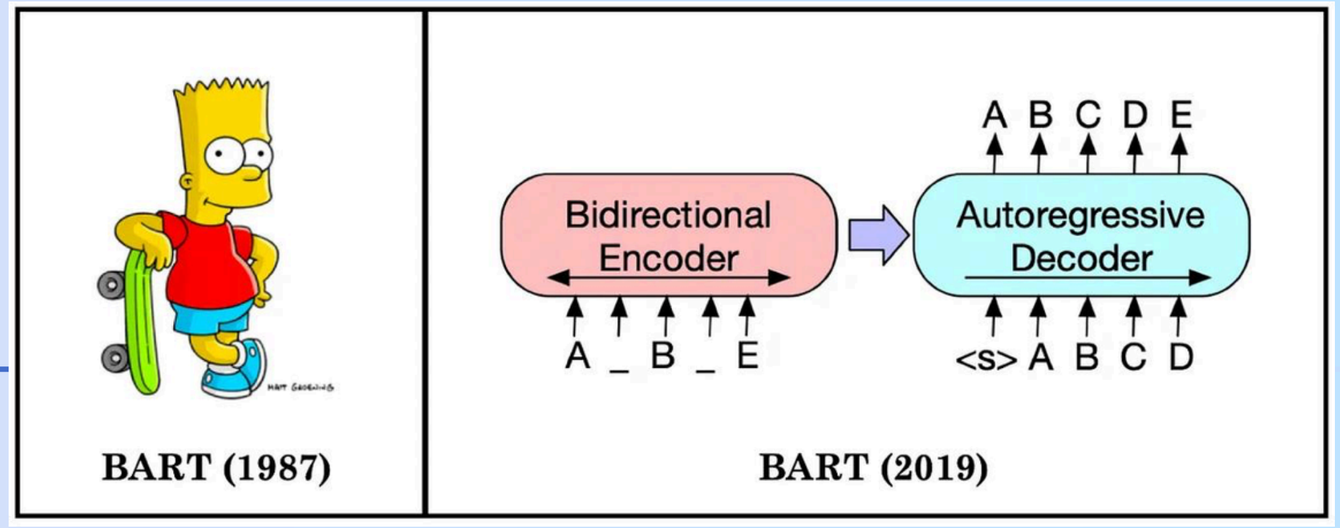
Feature-based Approach with BERT:

- Not all tasks can be represented by a Transformer encoder architecture → we need a task-specific model architecture.
- Pre-computing an expensive representation of the training data once and running many experiments with cheaper models on top of it is computationally beneficial.
- **BERT is effective for both fine-tuning and feature-based approach.**

Applications of BERT

1. **Text Classification**:
    a. **Sentiment Analysis** for customer reviews and social media posts
    b. **Spam Detection** for emails.
2. **Question Answering**: used in search engines (Google Search) and virtual assistants.
3. **Named Entity Recognition**: useful in information extraction, legal document processing, and content categorization.
4. **Paraphrase Detection**: duplicate question detection on customer service platforms.

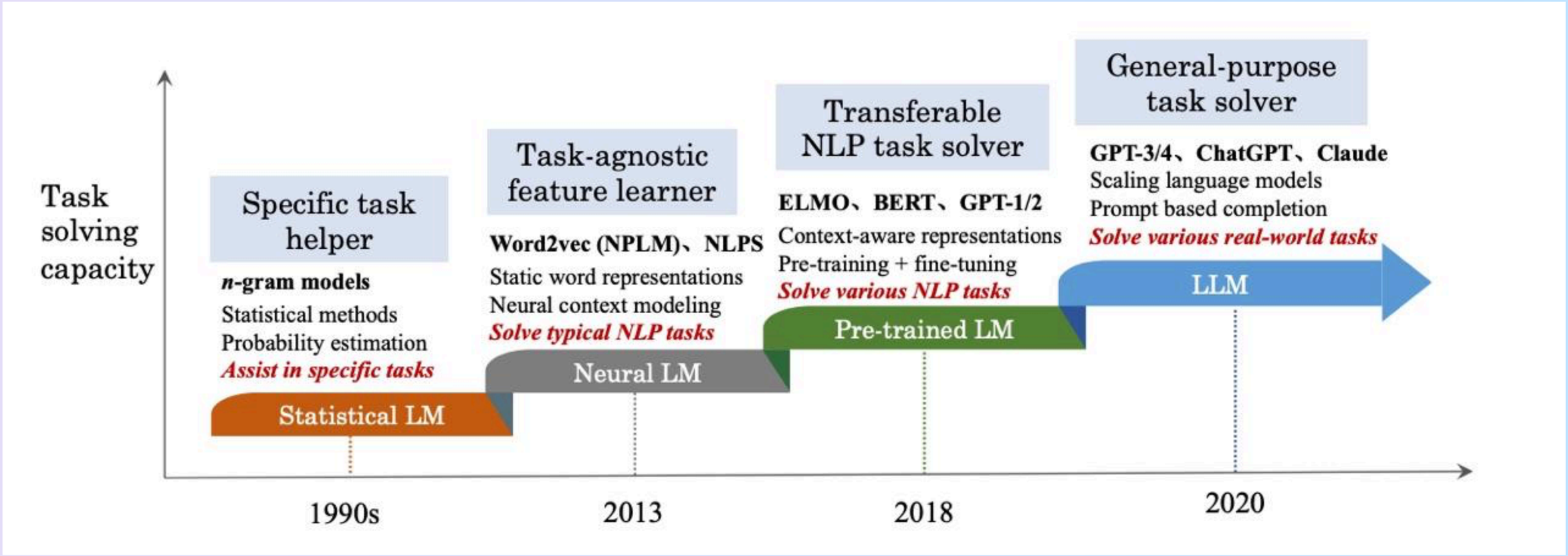
Beyond BERT



1. **Text Summarisation** (BERTSUM)
2. Biomedical and Scientific **Research** NLP (BioBERT and SciBERT)
3. **Multilingual** and Cross-Lingual Applications (mBERT and XLM-R): translation, multilingual text classification, and cross-lingual search
4. **Text Generation**: *Google*'s T5 and *Meta*'s BART combine BERT's bidirectional architecture with decoder models to enable text generation.

Descendant Models

- **ALBERT**
  - **less parameters**
  - shares weights across layers
  - no NSP: **Sentence Order Prediction**
- **DistilBERT**
  - a smaller, faster model
  - **knowledge distillation:** training a smaller “student” model to mimic a larger “teacher” model
  - **reduced layers** & dynamic masking
- **RoBERTa:**
  - training on **larger data**
  - longer training times
  - **no NSP**
  - **dynamic masking**


Task solving capacity
Specific task helper
n-gram models
Statistical methods
Probability estimation
Assist in specific tasks
Statistical LM
Task-agnostic feature learner
Word2vec (NPLM)、NLPS
Static word representations
Neural context modeling
Solve typical NLP tasks
Neural LM
Transferable NLP task solver
ELMO、BERT、GPT-1/2
Context-aware representations
Pre-training + fine-tuning
Solve various NLP tasks
Pre-trained LM
General-purpose task solver
GPT-3/4、ChatGPT、Claude
Scaling language models
Prompt based completion
Solve various real-world tasks
LLM
1990s
2013
2018
2020

Thank You!