

Deep Learning CSC-Elective

Instructor : Dr Muhammad Ismail Mangrio

Slides prepared by Dr. M Asif Khan

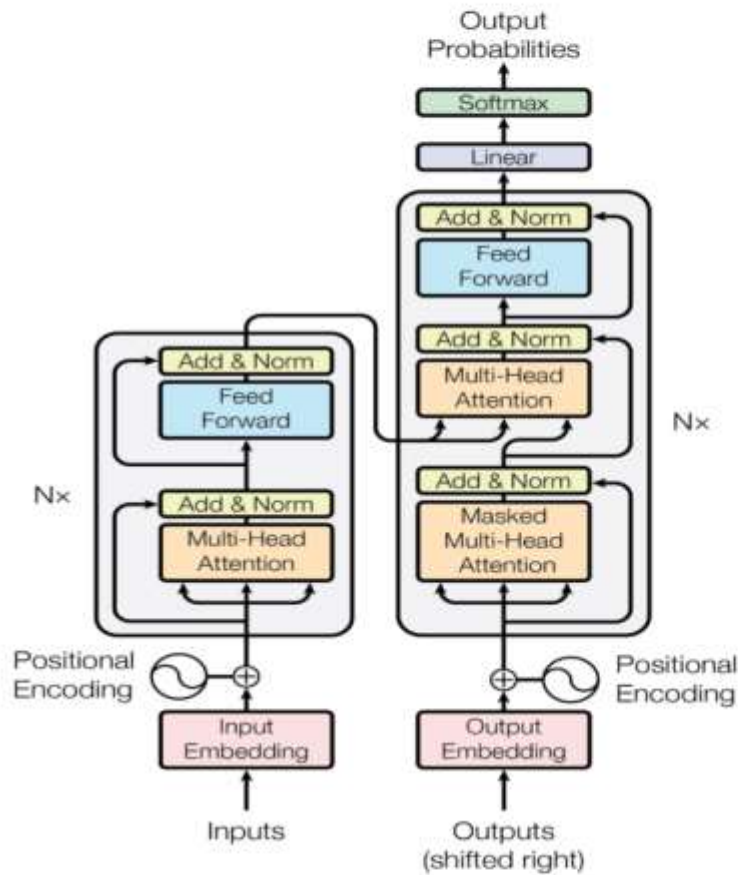
ismail@iba-suk.edu.pk

Unit 02 NLP Week 8

Contents

- Intro to BERT
- BERT input format
- BERT pre-training
- BERT fine-tuning
- BERT variants

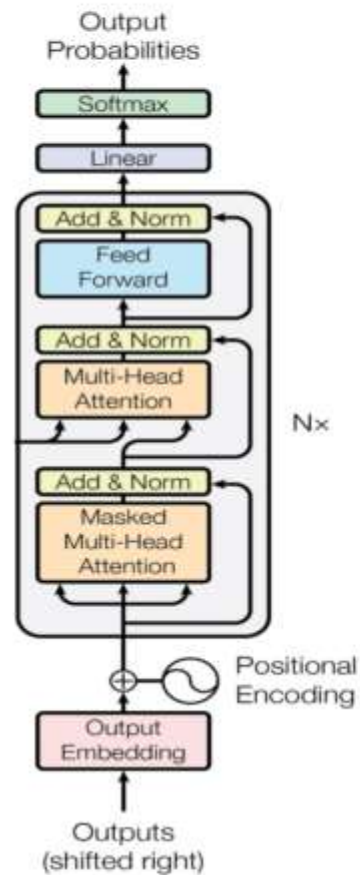
Transformer



Encoder

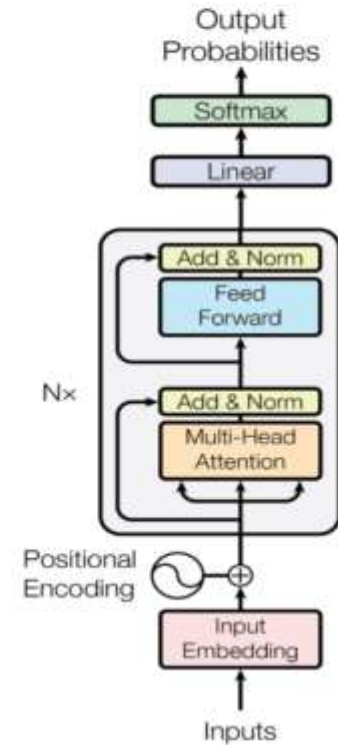
Decoder

GPT*



Decoder-only

BERT*



Encoder-only

*Illustrative example, exact model architecture may vary slightly

[CLS] everybody dance now [SEP]



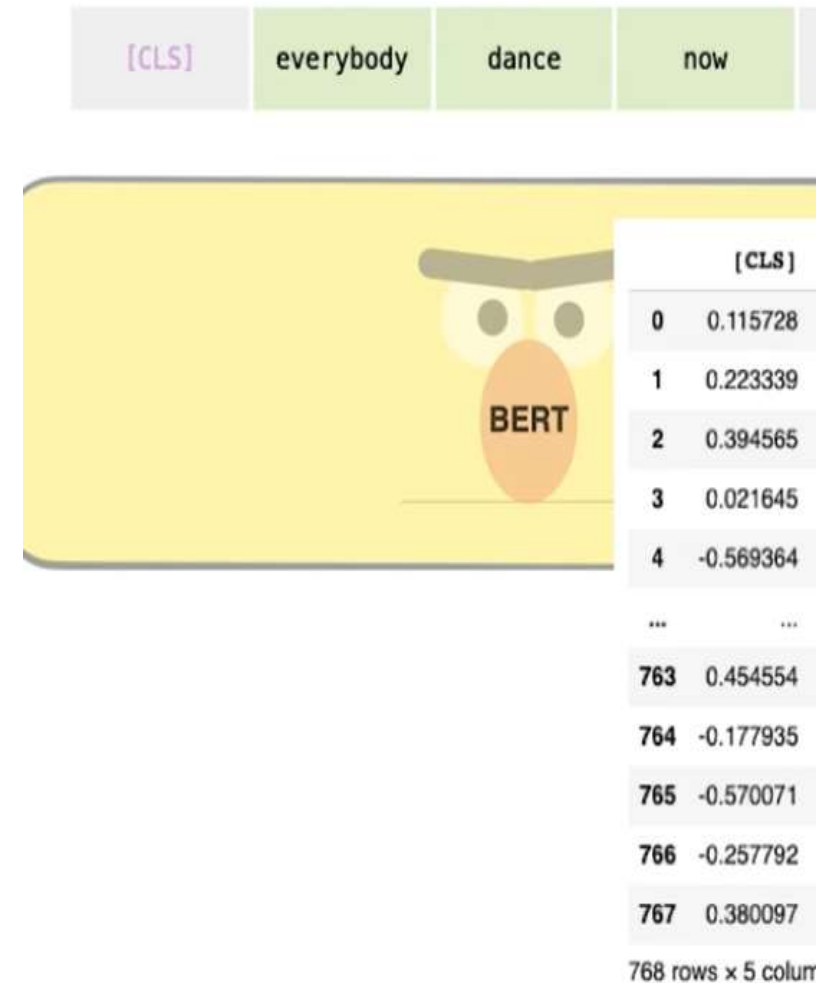
[CLS] everybody dance now [SEP]



	[CLS]	everybody	dance	now	[SEP]
0	0.115728	-0.150873	0.076831	-0.661984	0.834172
1	0.223339	0.269943	0.009255	-0.858786	0.279408
2	0.394565	0.411511	1.518060	0.053227	0.127079
3	0.021645	-0.369215	-0.217087	-0.251777	0.270489
4	-0.569364	-0.537542	-0.033147	0.340954	-0.452304
...
763	0.454554	0.539476	0.672731	0.648410	-0.864230
764	-0.177935	-0.192709	-0.515615	-0.102858	0.159125
765	-0.570071	0.058834	-0.369875	0.195014	-0.065941
766	-0.257792	0.029307	-0.338636	-0.121347	-0.979295
767	0.380097	0.632699	0.124689	-0.666084	-0.147284

768 rows x 5 columns

- CLS represents entire sentence





Hyperion



Dune



The Matrix



Hyperion



Neo the one



Dune



The Matrix

10%



15%



90%



Evolution of NLP to Large Language Models



What is BERT

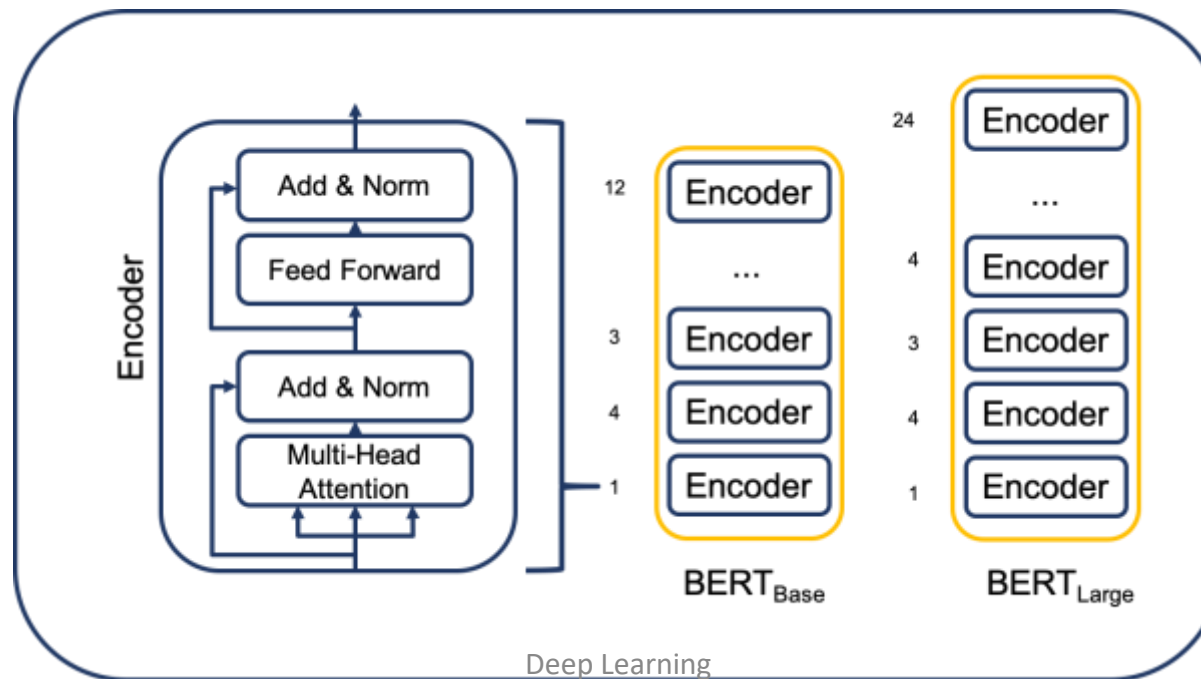
- **BERT** is a transformer-based language model developed by Google in 2018.
- Designed to **pre-train deep bidirectional** representations from **unlabeled** text by jointly conditioning on both **left and right context** in all layers.
- The Impact:
 - Achieved state-of-the-art results on 11 NLP tasks at the time of release.
 - Revolutionized tasks like Question Answering (SQuAD) and Natural Language Inference (NLI).
 - Became the foundational model for countless subsequent NLP applications (e.g., search engines, chatbots).
- **Traditional Embeddings** (Word2Vec, GloVe): Static representations. The word "bank" has the same vector in "river bank" and "investment bank."
- **Unidirectional Models** (GPT, ELMo left-to-right): Could only use **context from the left or the right**, but not both simultaneously for a given word.
 - Example: For "I accessed my ____ account."
 - A left-to-right model sees "I accessed my" but not the crucial word "account."
- **The Need:** A model that understands context from both directions at once for a **deeper understanding**.

BERT: Bi-directionality

- The Power of **Looking Both Ways**
- "The chef added the ____ to the pizza." Show arrows from both the left ("The chef added the") and the right ("to the pizza") pointing towards the blank, suggesting words like "anchovies," "cheese."
- The ability for a model to incorporate **context from both the left and the right** of a target word simultaneously is called **bi-directionality**.
- **BERT's Approach:** Unlike previous models that processed text sequentially, BERT's Transformer encoder **reads the entire sequence** of words at once.
- This allows every word to have some context from **every other word** in the sentence.
- **Analogy:**
 - Reading a sentence with a missing word; you use the **entire sentence's context** to guess it, not just the words before it.

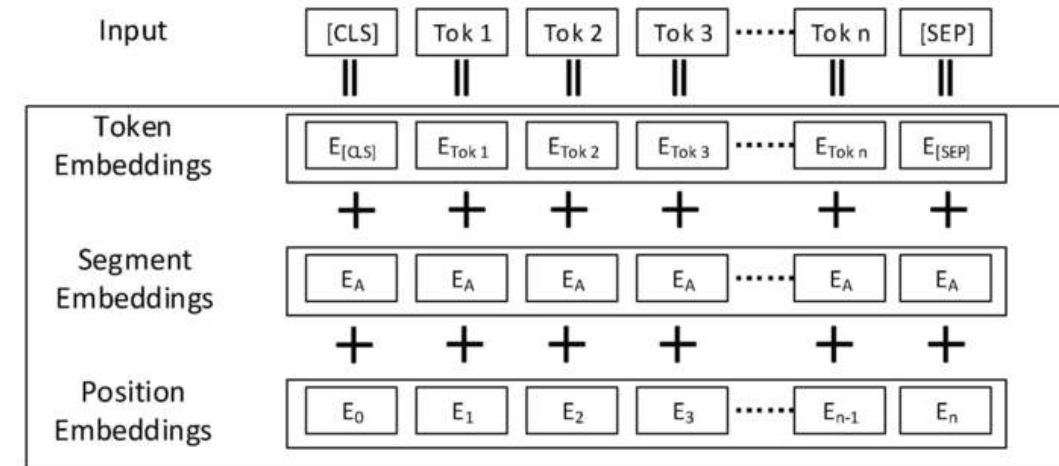
BERT: The Transformer architecture

- BERT is built only on the **Encoder half** of the original Transformer model.
- **Why the Encoder?** The encoder is designed to create rich, **contextual representations of input** data. It's a perfect fit for understanding language, **not generating it**.
- **Key Component:** The Self-Attention mechanism is the heart of the Transformer, allowing BERT to weigh the importance of all other words when encoding a specific word.



BERT's Input Representation

- BERT's input is cleverly constructed from three embeddings:
 - Token Embeddings:** WordPiece subword tokens.
 - Segment Embeddings:** Indicates **which sentence a token belongs to** (e.g., Sentence A: 0, Sentence B: 1). Crucial for tasks like NLI.
 - Position Embeddings:** Tells the model the **order of the words**, since the Transformer itself has no inherent sense of order.
- Special Tokens:**
 - [CLS]: **Classification token**, always the first token. Its final (output) hidden state is used for classification tasks.
 - [SEP]: Separator token, used to separate two sentences. Marks the end of sentence.



[CLS] The sky is blue . [SEP] The weather is clear today . [SEP]

BERT's Input Representation

- BERT first converts this into a single sequence using special tokens:
 - [CLS] I like NLP [SEP] It is fascinating [SEP]

Token:	[CLS]	I	like	NLP	[SEP]	It	is	fascinating	[SEP]
Token Embeddings	(Vector for [CLS])	(Vector for I)	(Vector for like)	(Vector for NLP)	(Vector for [SEP])	(Vector for It)	(Vector for is)	(Vector for fascinating)	(Vector for [SEP])
+ Segment Embeddings	(Vector for "A")	(Vector for "A")	(Vector for "A")	(Vector for "A")	(Vector for "A")	(Vector for "B")	(Vector for "B")	(Vector for "B")	(Vector for "B")
+ Position Embeddings	(Pos 0)	(Pos 1)	(Pos 2)	(Pos 3)	(Pos 4)	(Pos 5)	(Pos 6)	(Pos 7)	(Pos 8)
= Final Input Vector	E[CLS]	E_I	E_like	E_NLP	E[SEP]	E_It	E_is	E_fascinating	E[SEP]

BERT: Pre-training vs. Fine-tuning

- **Pre-training:**

- **What:** BERT is trained on a massive unlabeled text corpus (e.g., Wikipedia(~2.5B words), BookCorpus (~800M words)).
- **Goal:** Learn a general-purpose "language understanding" by solving two pre-training tasks.
 - **Masked Language Model (MLM)**
 - **Next Sentence Prediction (NSP)**
- **Cost:** Extremely computationally expensive (done once by Google).

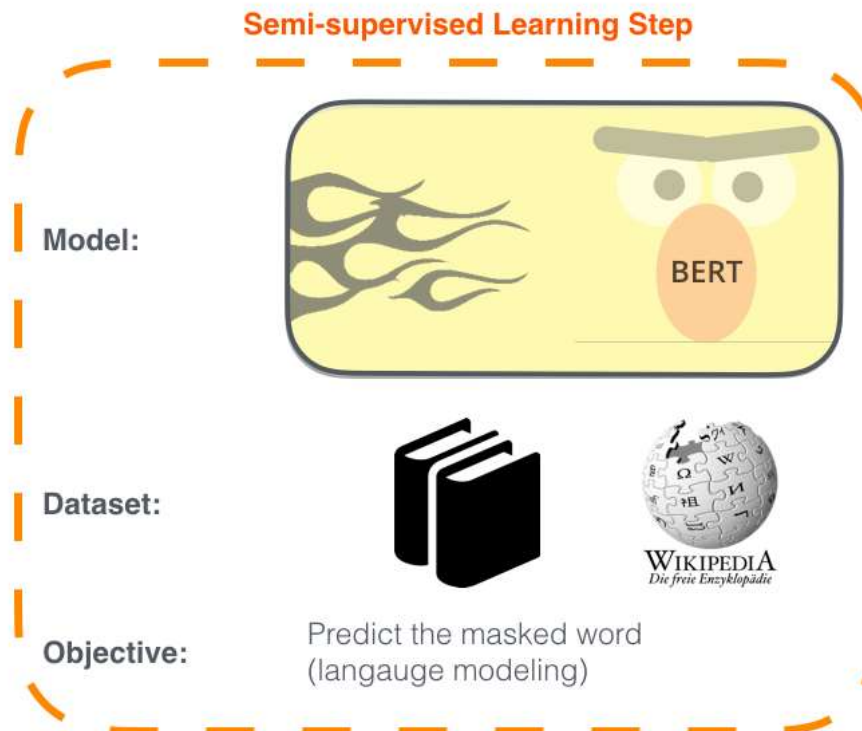
- **Fine-tuning:**

- **What:** The pre-trained BERT model is further trained on a smaller, labeled dataset for a specific task (e.g., sentiment analysis, spam detection).
- **Goal:** Specialize the general model for a specific application.
- **Cost:** Relatively fast and inexpensive.

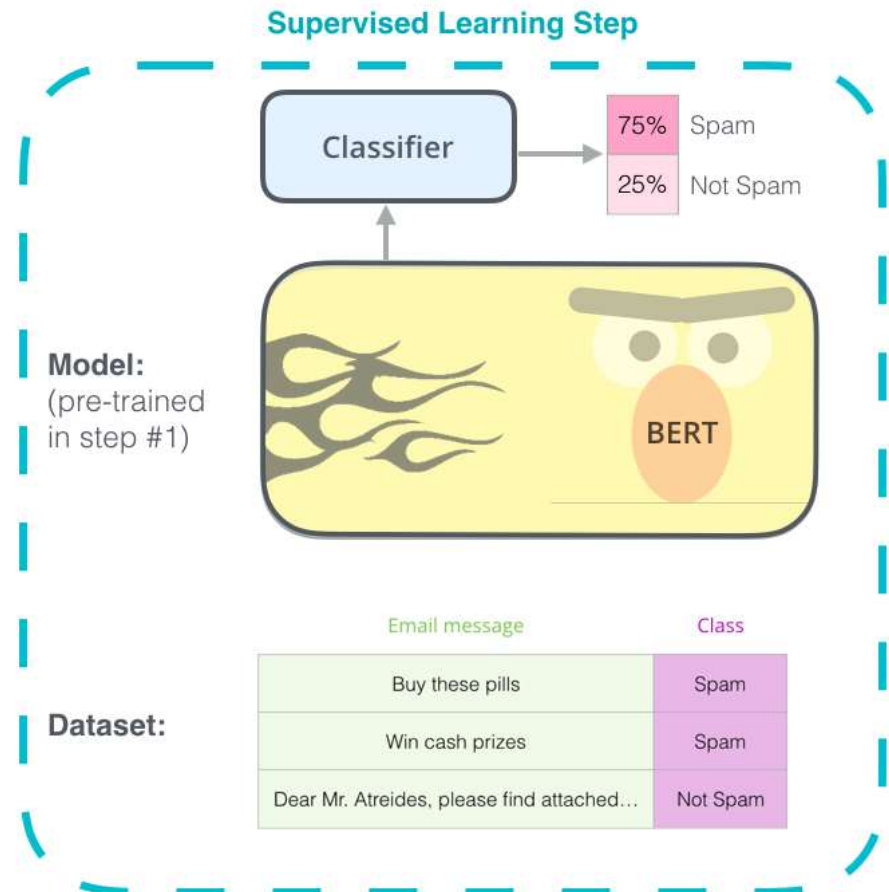
BERT: Pre-training vs. Fine-tuning

1 - **Semi-supervised** training on large amounts of text (books, wikipedia..etc).

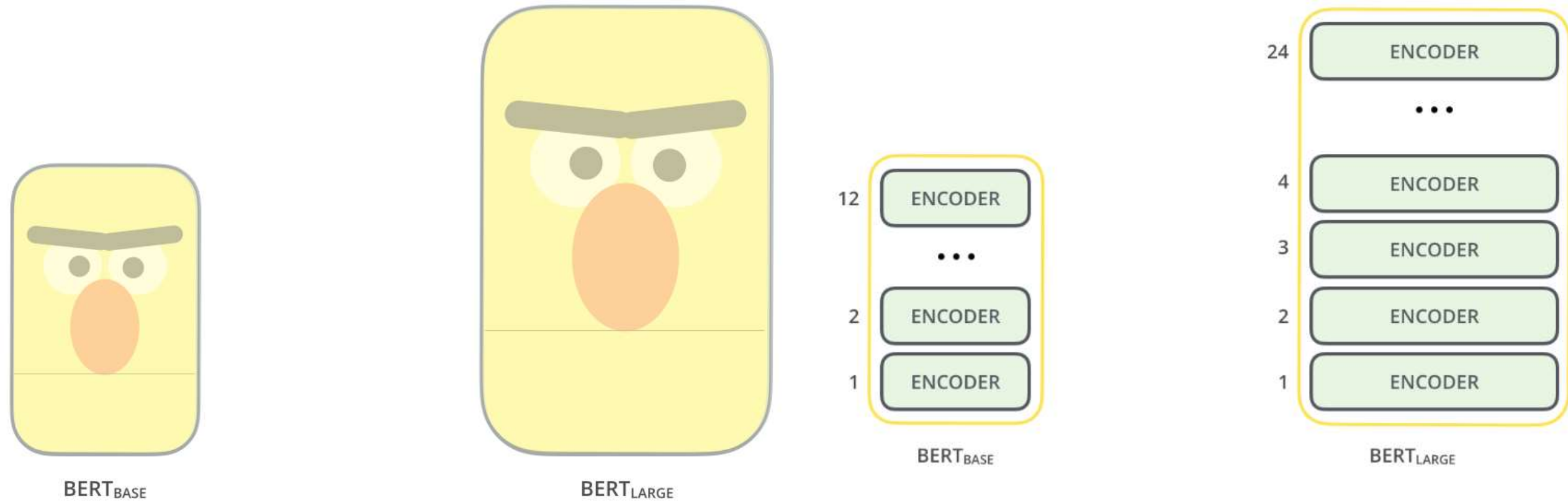
The model is trained on a certain task that enables it to grasp patterns in language. By the end of the training process, BERT has language-processing abilities capable of empowering many models we later need to build and train in a supervised way.



2 - **Supervised** training on a specific task with a labeled dataset.



BERT Model Sizes



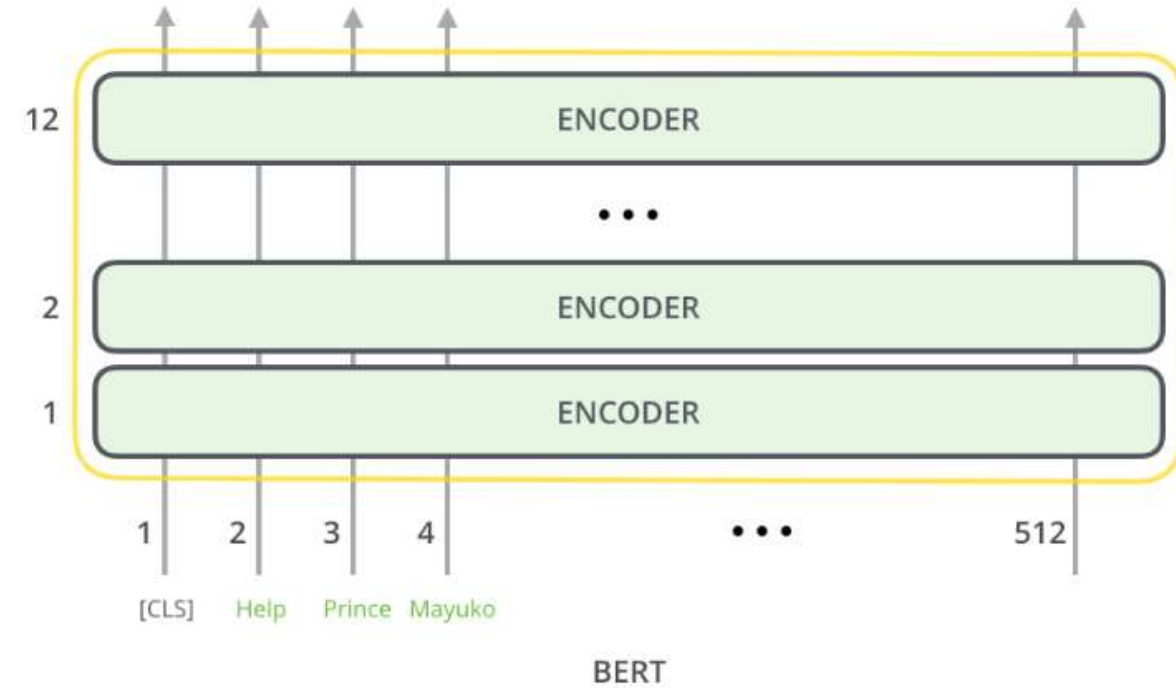
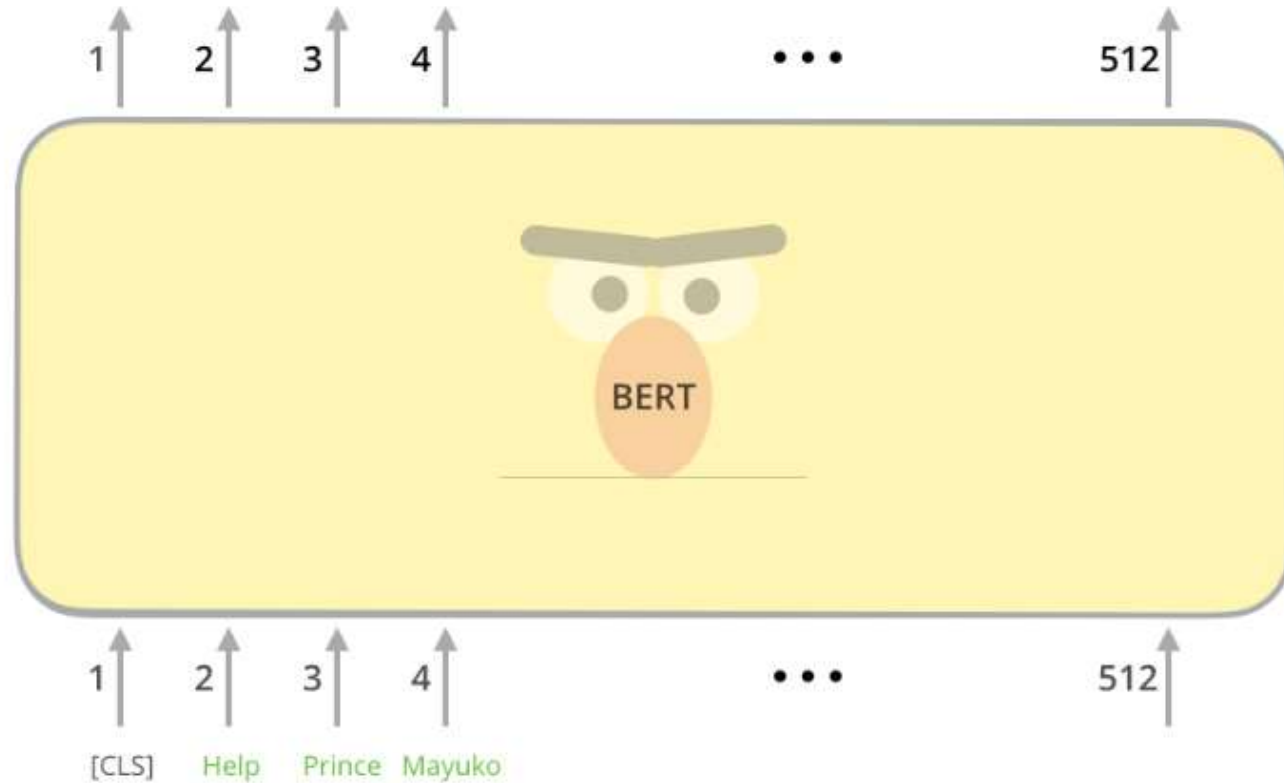
BERT Model Sizes

- Not all BERTs are created equal

Feature	BERT-Base	BERT-Large
Transformer Layers (Depth)	12	24
Hidden Size	768	1024
Attention Heads	12	16
Total Parameters	110 Million	340 Million

- Key Takeaway:** BERT-Large is more powerful but also more computationally expensive. BERT-Base is often a good starting point for many applications.
- (Note: Many smaller, distilled versions like DistilBERT now exist for faster, lighter-weight applications.)

Model Inputs



BERT: Pre-training Masked Language Model (MLM)

- **A sentence:** "The [MASK] sat on the mat." with an arrow pointing to the correct prediction "cat".
- **Procedure:**
 1. 15% of the input tokens are randomly selected.
 2. Of those selected:
 - 80% are replaced with the [MASK] token.
 - 10% are replaced with a random token. ["The tree sat on the mat."]
 - 10% are left unchanged. ["The cat sat on the mat."]
- **Task:** The model must **predict the original vocabulary id** of the masked word based on the bidirectional context.
- **Why the Random Replacements?** Makes the model more robust and prevents it from over-relying on the [MASK] token, which isn't present during fine-tuning.

BERT: Pre-training Masked Language Model (MLM)

- BERT hides (**masks**) some words.
- Sometimes it replaces them with **random** words.
- Sometimes it leaves them **unchanged**.
- The model must guess the original word.
- This teaches BERT deep understanding of context in both directions.

BERT: Pre-training: Next Sentence Prediction (NSP)

- **IsNext:**

Sentence A: [CLS] The man went to the store. [SEP]

Sentence B: He bought a gallon of milk. [SEP]

Label: IsNext

- **NotNext:**

Sentence A: [CLS] The man went to the store. [SEP]

Sentence B: The Eiffel Tower is in Paris. [SEP]

Label: NotNext

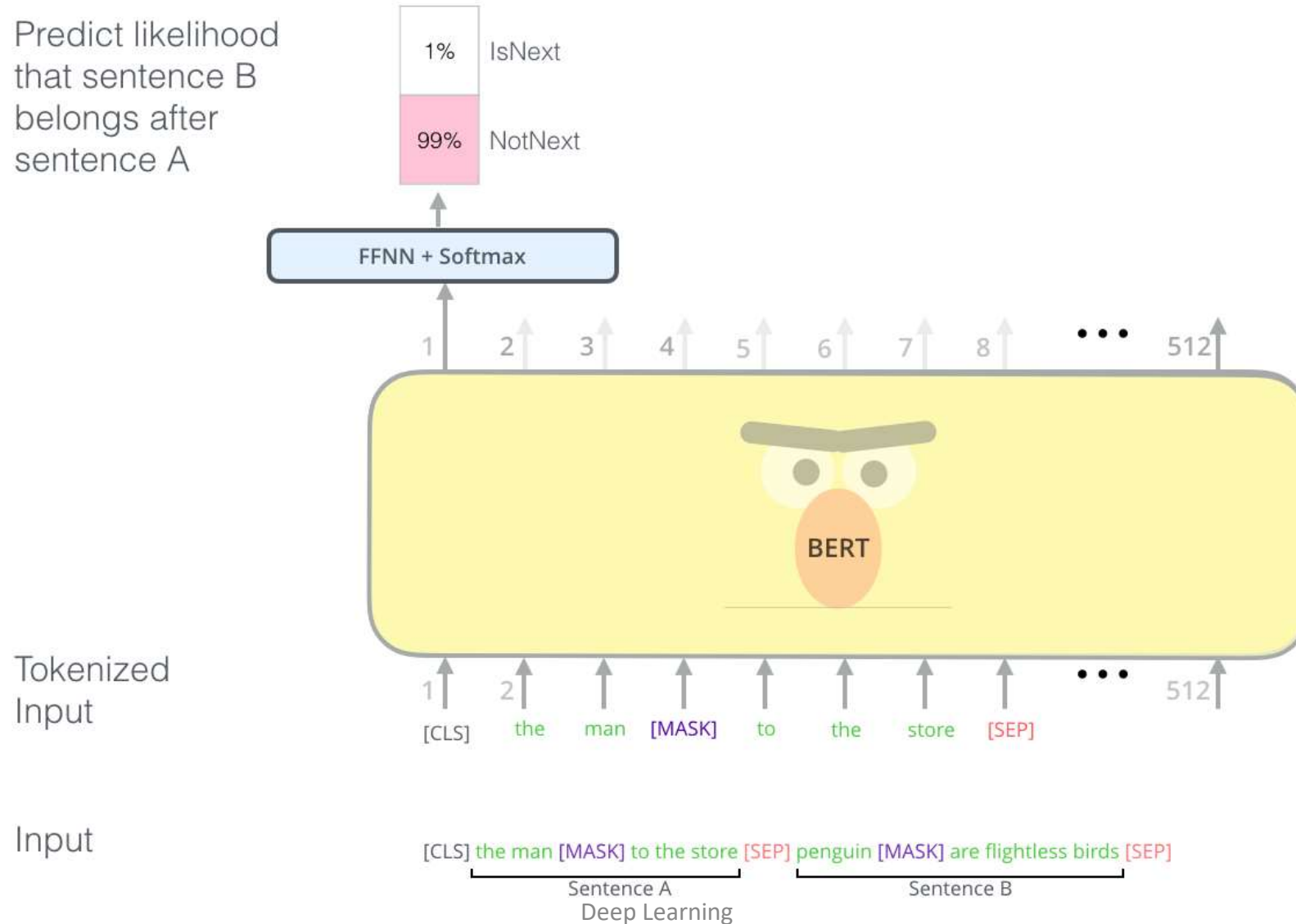
- **Goal:** Teach the model to understand the **relationships between sentences**, which is crucial for tasks like Question Answering (QA) and Natural Language Inference (NLI).
- **Procedure:**
 1. 50% of the time, Sentence B is the actual next sentence following Sentence A (Label: IsNext).
 2. 50% of the time, Sentence B is a random sentence from the corpus (Label: NotNext).
- The model uses the [CLS] token's representation to make this binary classification.

BERT: Pre-training: Next Sentence Prediction (NSP)

Sentence A	Sentence B	Label
The man went to the store.	He bought a gallon of milk.	IsNext
The man went to the store.	The Eiffel Tower is in Paris.	NotNext

- Model sees many such examples during training.
- Learns patterns of logical continuation, causality, or topic consistency.

BERT: Pre-training: Next Sentence Prediction (NSP)



BERT: Fine-Tuning Process

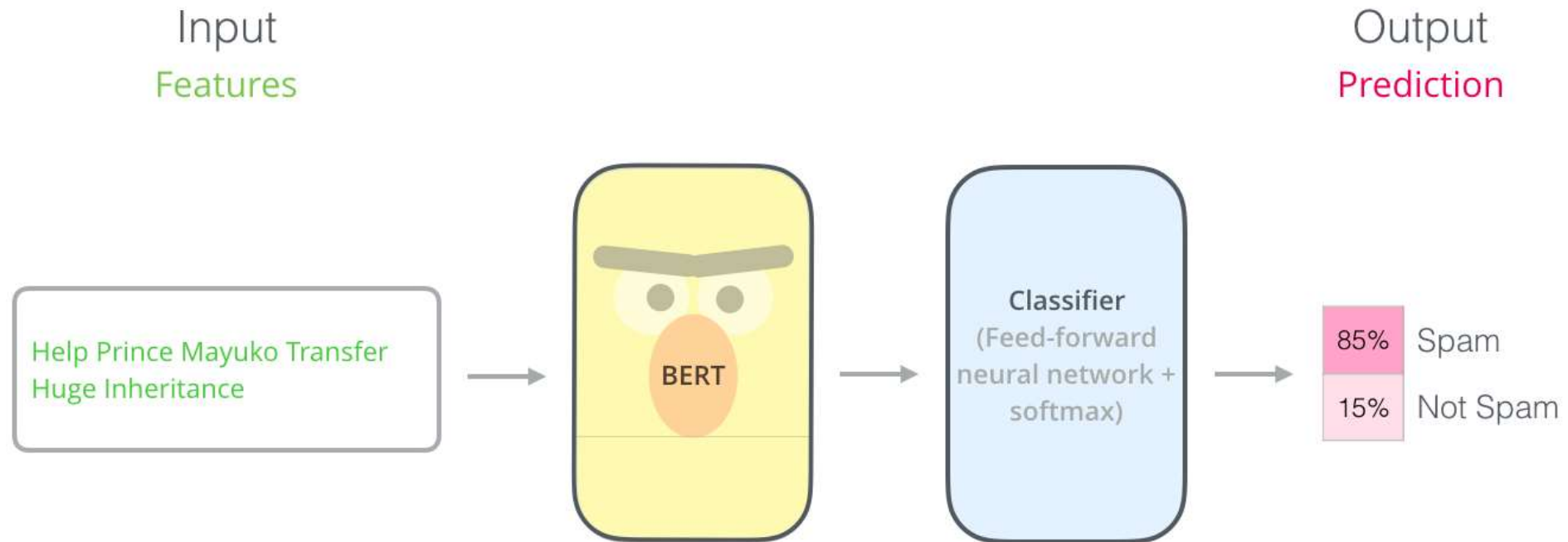
- Fine-tuning is relatively straightforward.
- You take the pre-trained BERT model and simply add a small task-specific layer on top (e.g., a linear classifier for sentiment).
- Then, you train the entire model end-to-end on your labeled task data.
- Because the model is already a powerful language understander, it requires relatively few epochs (3-4) and little data to achieve excellent performance.

*Code is available at [elearnnig](#) with file `NLP14_Word_embedding_Layer.ipynb`

BERT: Fine-Tuning Process for Text classification

- Sentiment Analysis
- **Input:** [CLS] I loved this movie! [SEP] -> BERT -> [CLS] representation -> Linear Classifier -> Output: "Positive" (vs. "Negative").
- **Task:** Classify the entire text into a category (e.g., Spam/Ham, Positive/Negative Sentiment, Topic Labeling).
- **Method:**
 - The final hidden state of the first [CLS] token is used as the aggregate sequence representation.
 - This vector is fed into a small classification layer (e.g., a simple feed-forward network) to predict the final label.
- The [CLS] token is designed to hold a representation that is useful for classification.

BERT: Fine-Tuning Process for Text classification



BERT: Fine-Tuning Process for Q & A

- **Task:** Extract the answer to a question from a given context paragraph.
- **Input:** [CLS] Question [SEP] Context [SEP]
- The model is fine-tuned to predict two values for every token in the context:
 - The probability of being the start of the answer span.
 - The probability of being the end of the answer span.
- The answer is the text between the tokens with the highest start and end scores.

BERT: Fine-Tuning Process for Q & A

- We use BERT's special tokens to package the question and context into a single, unified input sequence.
- [CLS] Which organization carried out the Apollo program? [SEP] The Apollo program was ... carried out by NASA, which succeeded... [SEP]
- [CLS]: The classification token, as always, comes first.
- Question: The entire question is written out.
- [SEP]: This separator token marks the end of the question and the beginning of the context.
- Context: The entire paragraph from which we want to extract the answer.
- [SEP]: A final separator to mark the end of the input.
- This format is perfect because BERT was pre-trained with NSP (Next Sentence Prediction), so it already has a built-in understanding of how two sentence-like segments relate to each other.

BERT: Fine-Tuning Process for Q & A

- Instead of making one classification decision for the whole sequence, the model makes **two predictions for every single token** in the input sequence.
- The model processes the entire sequence and outputs two sets of scores (probabilities):

Token:	[CLS]	Which	organiz ation	...	by	NASA	,	which	...	[SEP]
Start Score	0.001	0.001	0.002	...	0.150	0.850	0.001	0.001	...	0.001
End Score	0.001	0.001	0.001	...	0.100	0.700	0.150	0.001	...	0.001

- **Start Score:** For each token, the model asks, "Is this token the beginning of the answer span?"
- **End Score:** For each token, the model asks, "Is this token the end of the answer span?"
- In our example, the model has correctly learned that:
 - The token NASA has the highest probability of being the start of the answer.
 - The token NASA also has the highest probability of being the end of the answer (since the answer is a single word).

BERT: Fine-Tuning Process for NER

- **Example:** Labeling Entities in Text
- **A sentence:** "[CLS] Tim Cook is the CEO of Apple in California ." with tags below: B-PER I-PER O O O B-ORG O B-LOC O
- **Content:**
- **Task:** Label each word in a sentence with an entity type (e.g., Person, Organization, Location).
- **Method:**
 - The final hidden state of each input token (word piece) is fed into a classification layer.
 - This layer predicts a label for each token (e.g., using a BIO/BILUO tagging scheme: B-PER, I-PER, O, etc.).
- This showcases BERT's strength in token-level (sequence labeling) tasks.

BERT: Fine-Tuning Process for Sentence Pair Tasks

- **Example:** Natural Language Inference (NLI)
- **Input:** [CLS] Premise [SEP] Hypothesis [SEP] -> BERT -> [CLS] representation -> Classifier -> Output: "Entailment", "Contradiction", or "Neutral".
- Content:
- **Tasks:** Determine the relationship between two sentences (e.g., Paraphrase Identification, Natural Language Inference).
- Method:
 - The two sentences are combined into a single input sequence separated by the [SEP] token.
 - The [CLS] token's representation, which has been trained via NSP to understand inter-sentence relationships, is used for the final classification.

BERT: Fine-Tuning Process for Sentence Pair Tasks

- [CLS] The cat is sleeping on the mat. [SEP] The mat is occupied. [SEP]
- [CLS]: The classification token, whose final hidden state will represent the aggregate meaning of the entire sequence relationship.
- Premise: The first sentence, acting as the foundational fact.
- [SEP]: Separates the two sentences, telling BERT where one ends and the other begins.
- Hypothesis: The second sentence, whose truth is being evaluated.
- [SEP]: Marks the end of the input.
- **Output:** [0.02, 0.97, 0.01] (These probabilities correspond to [Contradiction, Entailment, Neutral]).
The final prediction is the class with the highest probability: "Entailment".

The BERT Ecosystem and Legacy

- A family tree or a cloud of model names: RoBERTa, ALBERT, DistilBERT, SciBERT, BioBERT, etc., all branching off from BERT.
- **Optimized Variants:**
 - **RoBERTa:** A robustly optimized BERT that removes NSP and uses more data.
 - **ALBERT:** Reduces memory consumption and increases training speed.
 - **DistilBERT:** A smaller, faster, cheaper, and lighter version.
- **Domain-Specific BERTs:** Models pre-trained on scientific text (SciBERT), biomedical literature (BioBERT), legal documents (Legal-BERT), etc.
- **The Transformer Wave:** BERT paved the way for the current era of large pre-trained models (GPT-3, T5, etc.).

Important Resources

- <https://www.tensorflow.org/hub>
- <https://huggingface.co/>
- https://www.tensorflow.org/text/tutorials/classify_text_with_bert
- https://www.tensorflow.org/text/guide/bert_preprocessing_guide

Summary

- Introduced BERT model
- It uses on encoder part of transformer architecture
- It is pre-trained by Google on huge datasets
- It can be fine-tuned for different NLP tasks
- It has diverse variants used to specific tasks