NLP and Deep Learning
MAT3399

Lecture 9:
Question Answering

Tuan Anh Nguyen @ Aimesoft
ted.nguyen95@gmail.com

Content taken from Hugging Face NLP Course

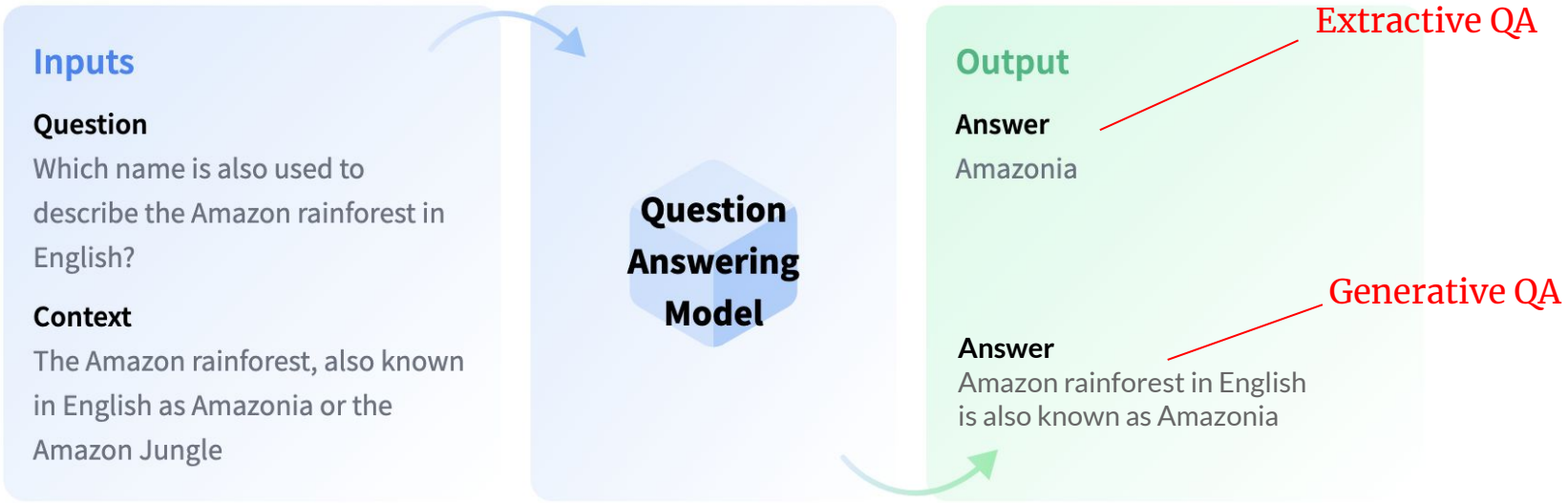
Question Answering?

Question answering (QA) is a computer science discipline within the fields of information retrieval and natural language processing that is concerned with building systems that automatically answer questions that are posed by humans in a natural language.

There are different QA variants:

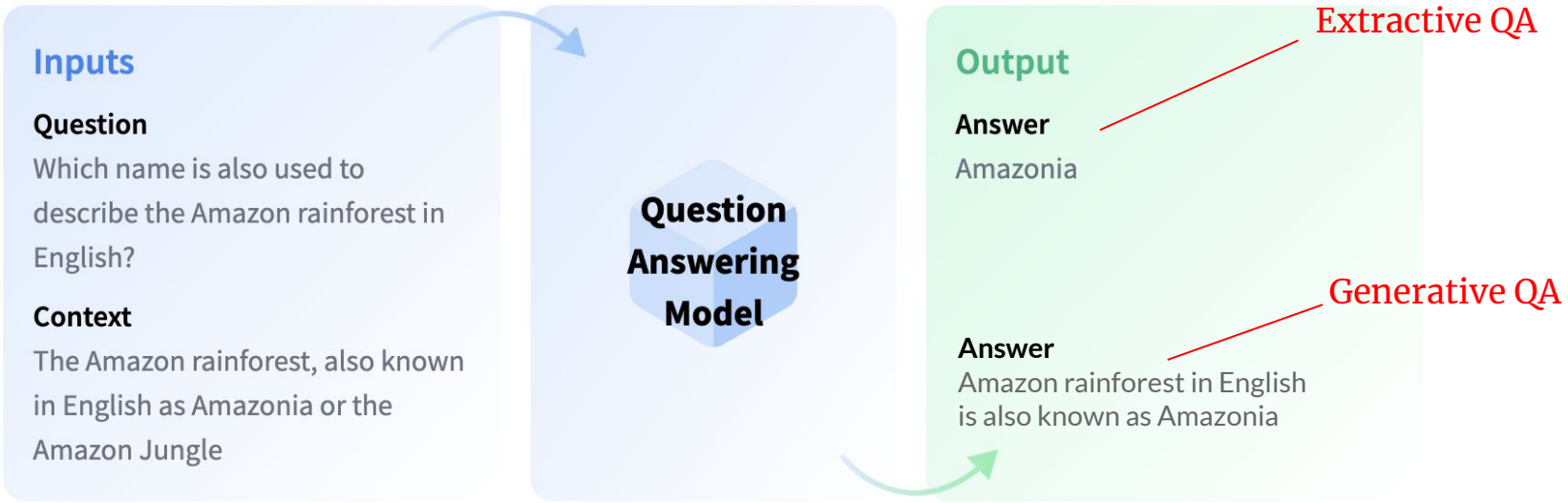
- Extractive QA: The model extracts the answer from a context.
- Generative QA: The model generates the answer in free text.
    - Open Generative QA: The model is given the question and context.
    - Closed Generative QA: The model is given the question only.

Extractive QA vs Generative QA

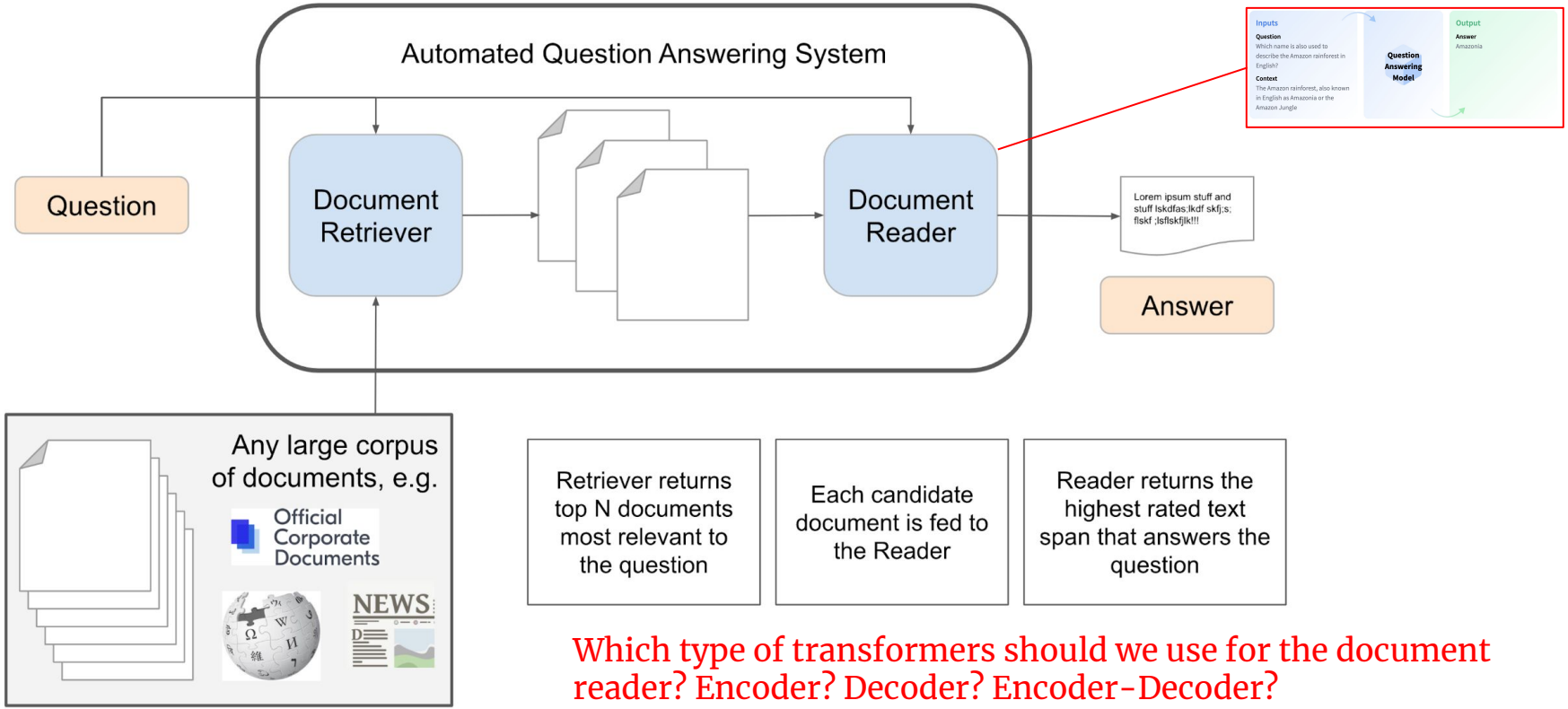


In this lecture, we mostly focus on Extractive QA.

Is this really “Question Answering”?



Architecture of a Completed QA System



Which type of transformers should we use for the document reader? Encoder? Decoder? Encoder-Decoder?

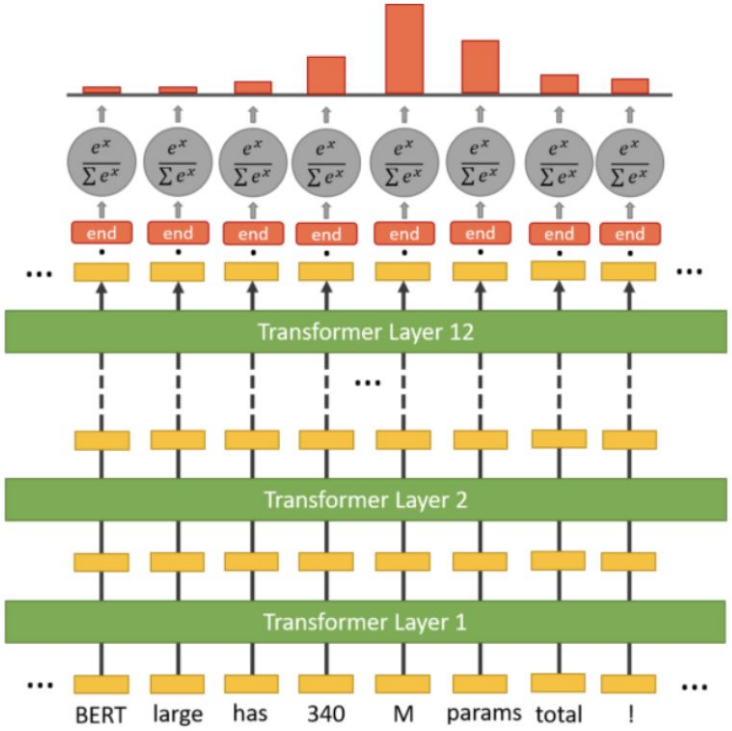
How does it work?

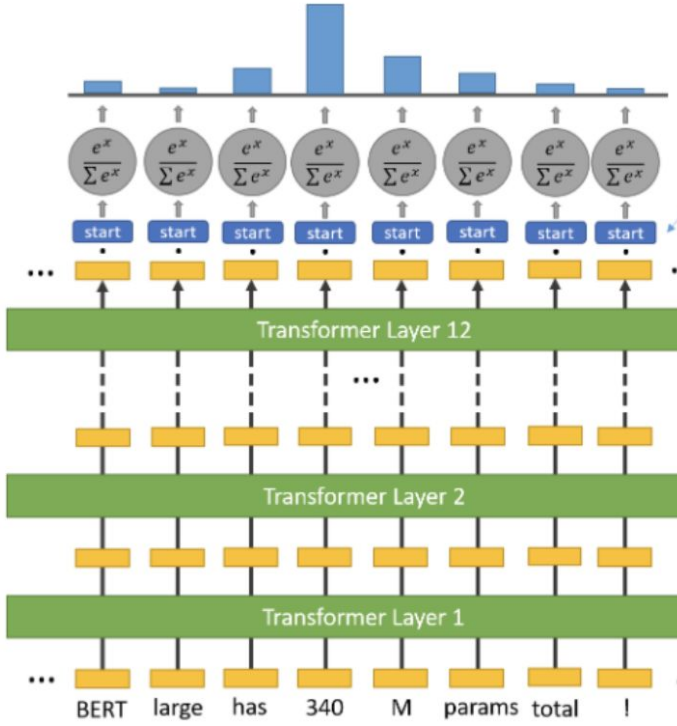
Goal: We try to predict the start/end position of the answer in the context -> Everything between the start & end token is considered the answer.

We pass the question along with the context like this:

[CLS] Which name is also used to describe the Amazon rainforest in English? [SEP] The Amazon rainforest, also known in English as Amazonia or the Amazon Jungle [SEP]

Question Answering Head for Both Start & End token





*This length 768 vector is the **weights** for the start token classifier.*
*The **same weights** are applied to **every position**.*

How to Choose the Best Start & End Token Pair?

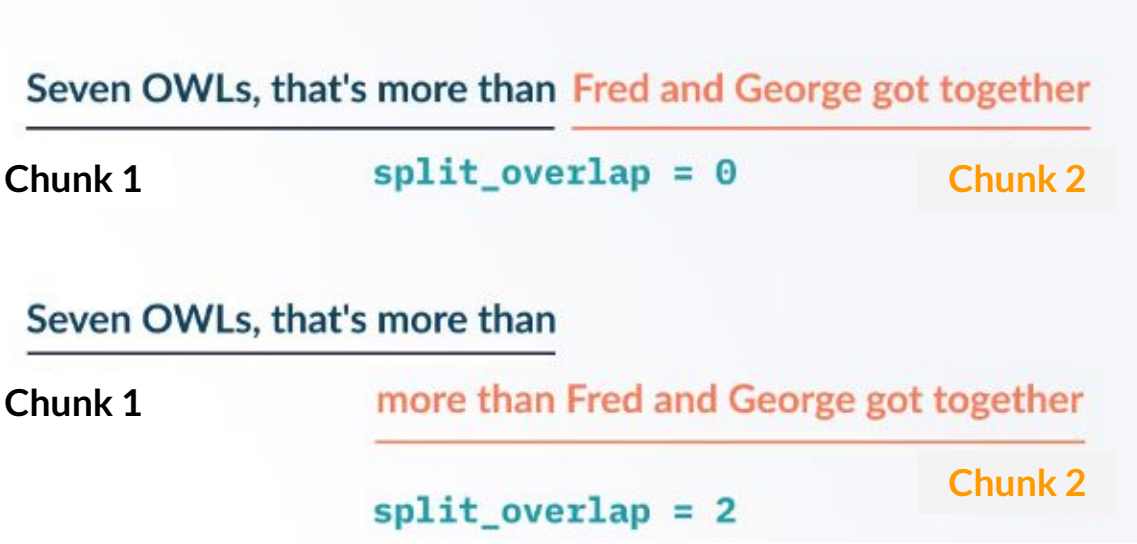
We'll look at the logit scores for the n_best start logits and end logits, excluding positions that give:

- An answer that wouldn't be inside the context
- An answer with negative length
- An answer that is too long (we limit the possibilities at max_answer_length=30)

Handle Long Context

The model for document reader has limitation on how many token you can pass to the model. How can we handle very long context?

Answer: Split context into chunks.



Which way is better?

Best Models to Use for QA

Encoder models like BERT, RoBERTA are popular for QA task.

	BERT	RoBERTa	DistilBERT	XLNet
Size (millions)	**Base**: 110 **Large**: 340	**Base**: 110 **Large**: 340	**Base:** 66	**Base**: ~110 **Large**: ~340
Training Time	**Base**: 8 x V100 x 12 days* **Large:** 64 TPU Chips x 4 days (or 280 x V100 x 1 days*)	**Large:** 1024 x V100 x 1 day; 4-5 times more than BERT.	**Base**: 8 x V100 x 3.5 days; 4 times less than BERT.	**Large:** 512 TPU Chips x 2.5 days; 5 times more than BERT.
Performance	Outperforms state-of-the-art in Oct 2018	2-20% improvement over BERT	3% degradation from BERT	2-15% improvement over BERT
Data	16 GB BERT data (Books Corpus + Wikipedia). 3.3 Billion words.	160 GB (16 GB BERT data + 144 GB additional)	16 GB BERT data. 3.3 Billion words.	**Base**: 16 GB BERT data **Large**: 113 GB (16 GB BERT data + 97 GB additional). 33 Billion words.
Method	BERT (Bidirectional Transformer with MLM and NSP)	BERT without NSP**	BERT Distillation	Bidirectional Transformer with Permutation based modeling

Evaluation

We use exact match and F1 score to evaluate QA task:

Exact match (EM): This is basically accuracy. For each question+answer pair, if the *characters* of the model's prediction exactly match the characters of (one of) the True Answer(s), EM = 1, otherwise EM = 0.

F1 score: The number of shared words between the prediction and the truth is the basis of the F1 score: precision is the ratio of the number of shared words to the total number of words in the *prediction*, and recall is the ratio of the number of shared words to the total number of words in the *ground truth*.

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Coding Exercise

Try `nguyenvulebinh/vi-mrc-base` model for Vietnamese QA.

Finetune `vinai/phobert-base` with Vietnamese SQUAD dataset. Download data here.

See how to finetune model:
https://huggingface.co/learn/nlp-course/chapter7/7