# Machine Learning HW7

## BERT - Question Answering

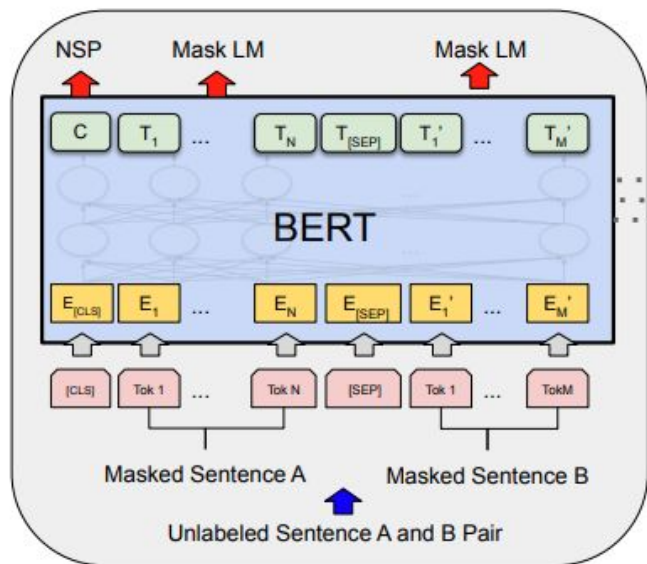ML TAs
mlta-2022-spring@googlegroups.com

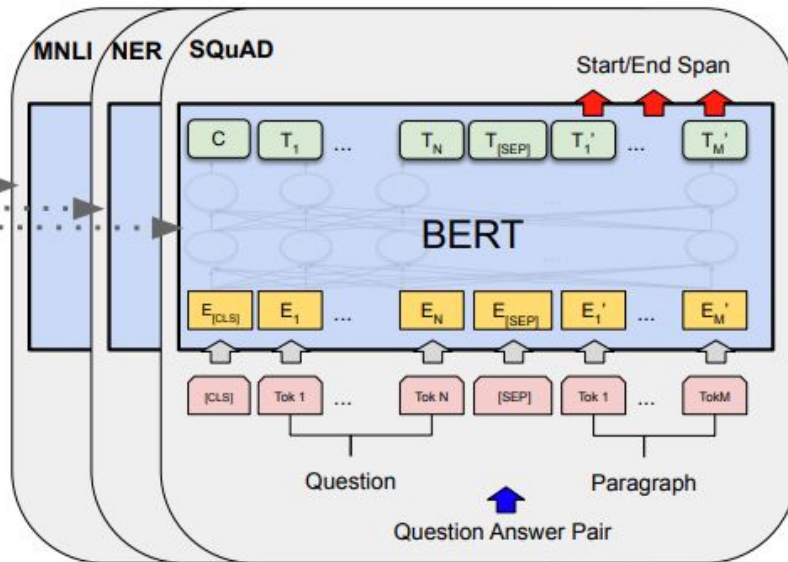# Outline

# 1. Task Introduction

# BERT



Pre-training
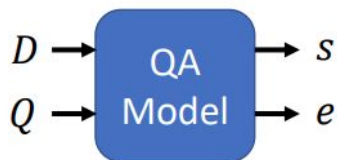
Fine-Tuning

https://arxiv.org/abs/1810.04805

# Task: Extractive Question Answering

- Extraction-based Question Answering (QA) (E.g. SQuAD)

**Document**: $D = \{d_1, d_2, \cdots, d_N\}$

**Query**: $Q = \{q_1, q_2, \cdots, q_N\}$

$D \rightarrow$ QA Model $\rightarrow s$

$Q \rightarrow$ QA Model $\rightarrow e$

output: two integers $(s, e)$

**Answer**: $A = \{q_s, \cdots, q_e\}$

In meteorology, precipitation is any product of the condensation of 17 spheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain 77 atte 79 cations are called "showers".

What causes precipitation to fall?
**gravity** $s = 17, e = 17$

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?
**graupel**

Where do water droplets collide with ice crystals to form precipitation?
**within a cloud** $s = 77, e = 79$

# Dataset: DRCD & ODSQA

DRCD: 台達閱讀理解資料集 Delta Reading Comprehension Dataset

ODSQA: Open-Domain Spoken Question Answering Dataset

- train: DRCD + DRCD-TTS
  - 10524 paragraphs, 31690 questions
- dev: DRCD + DRCD-TTS
  - 1490 paragraphs, 4131 questions
- test: DRCD + ODSQA
  - 1586 paragraphs, 4957 questions

# Dataset: Chinese Reading Comprehension

```
{
    "questions": [
        {
            "id": 0,
            "paragraph_id": 3884,
            "question_text": "羅馬教皇利奧三世在800年正式加冕誰為羅馬
            "answer_text": "查理大帝",
            "answer_start": 141,
            "answer_end": 144
        },
        {
            "id": 1,
            "paragraph_id": 4630,
            "question_text": "百濟國在哪一年建國?",
            "answer_text": "公元前18年",
            "answer_start": 266,
            "answer_end": 271
```

Testing:　　No answer!

```
"questions": [
    {
        "id": 0,
        "paragraph_id": 792,
        "question_text": "士官長的頭盔上會有何裝飾物?",
        "answer_text": null,
        "answer_start": null,
        "answer_end": null
```

```
"paragraphs": [
    "2010年引進的廣州快速公交運輸系統，屬世界第二大快速公交系統，日常載
    "廣州是京廣鐵路、廣深鐵路、廣茂鐵路、廣梅汕鐵路的終點站。2009年末，
    "廣州自古已是華南地區著名的商埠，擁有2000多年的開放貿易歷史。1970
    "廣州市內雨水充潤、土地肥沃，市區曾經有非常廣大的農業用地。兩千年前
    "古代廣州的手工業非常發達，船舶業、冶鑄和五金業、紡織業、食品加工、
    "中華人民共和國成立後工業國有化。五六十年代時，工業有所恢復，但文化
```

# Dataset: Chinese Reading Comprehension

**Paragraph**

新加坡、馬來西亞的華文學術界在 1970 年代後開始統一使用簡體中文；然而 繁體字 在媒體中普遍存在著，例如華人商店的招牌、舊告示及許多非學術類中文書籍，香港和臺灣所出版的書籍也有在市場上流動。當地許多中文報章都會使用「標題繁體字，內容簡化字」的方式讓簡繁中文並存。除此之外，馬來西亞所有本地中文報章之官方網站都是以繁體中文為主要文字。

**Question:**　　　新加坡的華文學術界在哪個年代後開始使用簡體中文？

**Answer:**　　　1970


**Question:**　　　馬來西亞的華人商店招牌主要使用什麼文字？

**Answer:**　　　繁體字

# 2. Tutorial

# Tokenization

'李宏毅教授2022機器學習'

tokenize

['李', '宏', '毅', '教', '授', '2022', '機', '器', '學', '習']

tokens_to_ids

[3330, 2131, 3675, 3136, 2956, 10550, 3582, 1690, 2119, 5424]

# Tokenization



input_ids: [CLS] $q_1$ $q_2$ [SEP] $d_1$ $d_2$ $d_3$ [SEP] [PAD] [PAD]

question      document

token_type_ids:    0    0    0    0    1    1    1    1    0    0

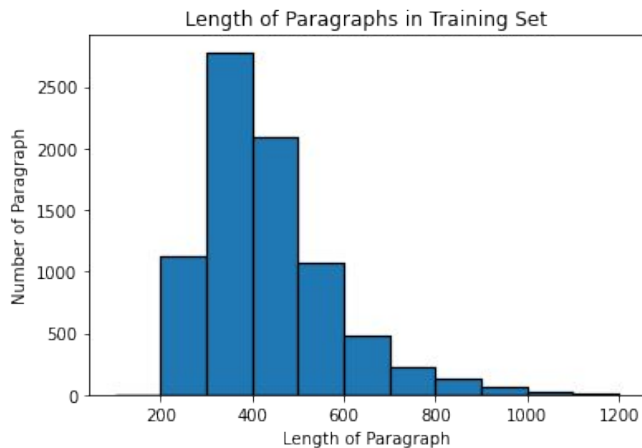attention_mask:    1    1    1    1    1    1    1    1    0    0

# Why Long Paragraph is an Issue?

Total sequence length = question length + paragraph length + 3 (special tokens)

Maximum input sequence length of BERT is restricted to 512, why?

➔    Self-Attention in transformer has $O(n^2)$ complexity


Length of Paragraphs in Training Set

Therefore, we may not be able to process the whole paragraph.

What can we do?

# Training

We know where the answer is in training!

Assumption: Info needed to answer the question can be found near the answer!

Simple solution: Just draw a window (as large as possible) around the answer!

e.g. window size = max_paragraph_len = 32

新加坡、馬來西亞的華文學術界在 **1970** 年代後開始統一使用簡體中文；然而 **繁體字** 在媒體中普遍存在著，例如華人商店的招牌、舊告示及許多非學術類中文書籍，香港和臺灣所出版的書籍也有在市場上流動 ...

**Q:** 新加坡的華文學術界在哪個年代後開始使用簡體中文？ **A:** **1970**
**Q:** 馬來西亞的華人商店招牌主要使用什麼文字？ **A:** **繁體字**

# Testing

We do not know where the answer is in testing ➡ split into windows!

e.g.   window size = max_paragraph_len = 32

新加坡、馬來西亞的華文學術界在 **1970**年代後開始統一使用簡體中文；然而 **繁體字**在媒體中普遍存在著，例如華人商店的招牌、舊告示及許多非學術類中文書籍 ......

**Q:**    新加坡的華文學術界在哪個年代後開始使用簡體中文 ？       **A:**    **1970**
**Q:**    馬來西亞的華人商店招牌主要使用什麼文字 ？       **A:**    **繁體字**

For each window, model predicts a start score and an end score ➡ take the maximum to be answer!

|  | start score | start position | end score | end position | total score |
|---|---|---|---|---|---|
| window 1 | 0.5 | 23 | 0.4 | 26 | 0.9 |
| window 2 | 0.3 | 35 | 0.7 | 37 | 1.0 |

Answer:

position 35 to 37

# 3. Hints

# Hints for beating baselines

- ❖ **Simple:**
  - ➤ Sample code
- ❖ **Medium:**
  - ➤ Apply linear learning rate decay
  - ➤ Change value of "doc_stride"
- ❖ **Strong:**
  - ➤ Improve preprocessing
  - ➤ Try other pretrained models
- ❖ **Boss:**
  - ➤ Improve postprocessing
  - ➤ Further improve the above hints

**Estimated training time**

|        | K80   | T4  | T4 (FP16) | P100 | V100 |
|--------|-------|-----|-----------|------|------|
| Simple | 40m   | 20m | 8m        | 10m  | 7m   |
| Medium | 40m   | 20m | 8m        | 10m  | 7m   |
| Strong | 2h    | 1h  | 25m       | 35m  | 20m  |
| Boss   | 12.5h | 6h  | 2.5h      | 4.5h | 2h   |

- ❖ **Training Tips (Optional):**
  - ➤ Automatic mixed precision (fp16)
  - ➤ Gradient accumulation
  - ➤ Ensemble

# Linear Learning rate decay

```
##### TODO: Apply linear learning rate decay #####
```

Method 1:	Adjust learning rate manually

- Decrement **optimizer.param_groups[0]["lr"]** by **learning_rate / total training step** per step

```python
learning_rate = 1e-4
optimizer = AdamW(model.parameters(), lr=learning_rate)
total_step = 1000
for i in range(total_step):
    optimizer.param_groups[0]["lr"] -= learning_rate / total_step
```

**This block is only an example!**
**You only need to add 1 or 2 lines**

```
optimizer.param_groups[0]["lr"]

-1.5863074217500927e-18
```

**Check whether learning rate after training is very close to zero**

**Decrement learning rate to zero linearly**
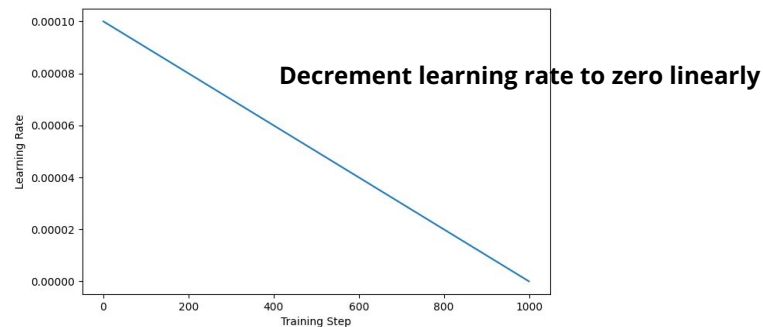
Method 2:	Adjust learning rate automatically by scheduler

- huggingface (Recommended)	Link
- pytorch	Link

```python
scheduler = ...

for i in range(total_step):
    ...
    optimizer.step()
    scheduler.step()
    ...
```
**You only need to add 3 or 4 lines**

You may also try other learning rate schedules (e.g. warmup)!

# Doc stride

```
##### TODO: Change value of doc_stride #####
self.doc_stride = 150
```

## Testing

We do not know where the answer is in testing ⟹ split into windows!

e.g. window size = max_paragraph_len = 32

新加坡、馬來西亞的華文學術界在**1970**年代後開始統一使用簡體中文，然而**繁體字**在媒體中普遍存在著，例如華人商店的招牌、舊告示及許多非學術類中文書籍......

Q: 新加坡的華文學術界在哪個年代後開始使用簡體中文？    A: 1970
Q: 馬來西亞的華人商店招牌主要使用什麼文字？    A: 繁體字

start position of 1st window

start position of 2nd window

doc_stride = distance between the start position of two consecutive windows

doc_stride is set to "max_paragraph_len" in sample code (i.e. no overlap)

What if answer is near the boundary of windows or across windows?

**Hint:**      Overlapping windows

# Preprocessing

Hint:     How to prevent model from learning something it should not learn
          during training? (i.e. answers are not always near the middle of window)

```
##### TODO: Preprocessing #####
# Hint: How to prevent model from learning something it should not learn

if self.split == "train":
        # Convert answer's start/end positions in paragraph_text to start/end positions in tokenized_paragraph
        answer_start_token = tokenized_paragraph.char_to_token(question["answer_start"])
        answer_end_token = tokenized_paragraph.char_to_token(question["answer_end"])

        # A single window is obtained by slicing the portion of paragraph containing the answer
        mid = (answer_start_token + answer_end_token) // 2
        paragraph_start = max(0, min(mid - self.max_paragraph_len // 2, len(tokenized_paragraph) - self.max_paragraph_len))
        paragraph_end = paragraph_start + self.max_paragraph_len

        # Slice question/paragraph and add special tokens (101: CLS, 102: SEP)
        input_ids_question = [101] + tokenized_question.ids[:self.max_question_len] + [102]
        input_ids_paragraph = tokenized_paragraph.ids[paragraph_start : paragraph_end] + [102]
```

# Other pretrained models

You can choose any model you like! [Link to pretrained models in huggingface]

Note 1:   You are **NOT** allowed to use pretrained models outside huggingface!
(Violation = cheating = final grade x 0.9)

Note 2:   Some models have    📦 Model card    describing details of the model

Note 3:   Changing models may lead to error message, try it solve it yourself

# Postprocessing

Hint: Open your prediction file to see what is wrong

(e.g. what if predicted end_index < predicted start_index?)

```python
def evaluate(data, output):
    ##### TODO: Postprocessing #####
    # There is a bug and room for improvement in postprocessing
    # Hint: Open your prediction file to see what is wrong

    answer = ''
    max_prob = float('-inf')
    num_of_windows = data[0].shape[1]

    for k in range(num_of_windows):
        # Obtain answer by choosing the most probable start position / end position
        start_prob, start_index = torch.max(output.start_logits[k], dim=0)
        end_prob, end_index = torch.max(output.end_logits[k], dim=0)

        # Probability of answer is calculated as sum of start_prob and end_prob
        prob = start_prob + end_prob

        # Replace answer if calculated probability is larger than previous windows
        if prob > max_prob:
            max_prob = prob
            # Convert tokens to chars (e.g. [1920, 7032] --> "大 金")
            answer = tokenizer.decode(data[0][0][k][start_index : end_index + 1])

    # Remove spaces in answer (e.g. "大 金" --> "大金")
    return answer.replace(' ','')
```

# Training Tip:    Automatic mixed precision

- PyTorch trains with 32-bit floating point (FP32) arithmetic by default
- Automatic Mixed Precision (AMP) enables automatic conversion of certain GPU operations from FP32 precision to half-precision (FP16)
- Offer about 1.5-3.0x speed up while maintaining accuracy

```
# Change "fp16_training" to True to support mixed precision training (fp16)
fp16_training = False

if fp16_training:
    !pip install accelerate==0.2.0
    from accelerate import Accelerator
    accelerator = Accelerator(fp16=True)
# Documentation for the toolkit:  https://huggingface.co/docs/accelerate/
```

```
if fp16_training:
    model, optimizer, train_loader = accelerator.prepare(model, optimizer, train_loader)
```

```
if fp16_training:
    accelerator.backward(output.loss)
else:
    output.loss.backward()
```

## Estimated training time

|        | T4  | T4 (FP16) |
|--------|-----|-----------|
| Simple | 20m | 8m        |
| Medium | 20m | 8m        |
| Strong | 1h  | 25m       |
| Boss   | 6h  | 2.5h      |

**Warning:** only work on some gpu (e.g. T4, V100)

**Reference:**     accelerate documentation
Intro to native pytorch automatic mixed precision

# Training Tip:    Gradient accumulation

Use it when gpu memory is not enough but you want to use larger batch size

- Split global batch into smaller mini-batches
- For each mini-batch:   Accumulate gradient without updating model parameters
- Update model parameters





Gradient accumulation
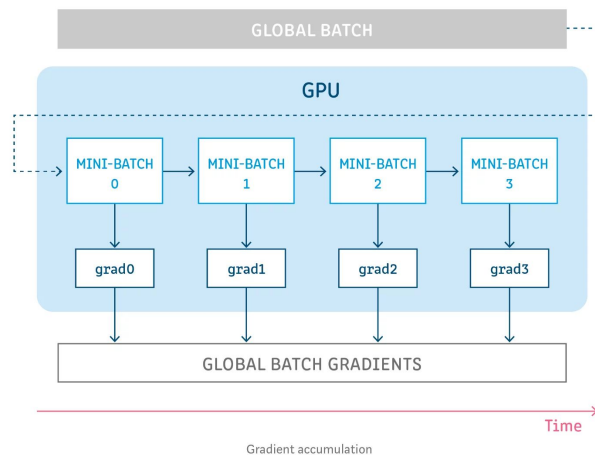
**Reference:**     Gradient Accumulation in PyTorch

# 4. Grading

# Grading

| | |
|---|---|
| Report | +4pts |
| Code Submission | +2pts |
| Simple Baseline (Public) | +0.5pt |
| Simple Baseline (Private) | +0.5pt |
| Medium Baseline (Public) | +0.5pt |
| Medium Baseline (Private) | +0.5pt |
| Strong Baseline (Public) | +0.5pt |
| Strong Baseline (Private) | +0.5pt |
| Boss Baseline (Public) | +0.5pt |
| Boss Baseline (Private) | +0.5pt |

# Kaggle (4pts)

**Deadline:**           **5/6 (Fri.) 23:59**

**Displayed name:**     \<student ID>_\<anything>

**Testing Set:**         4957 Questions (~50% public set, ~50% private set)

**Evaluation Metric:**    Accuracy (Exact Match)

| Baselines | Public Score |
|---|---|
| Simple (0.5pt + 0.5pt) | 0.45139 |
| Medium (0.5pt + 0.5pt) | 0.65792 |
| Strong (0.5pt + 0.5pt) | 0.78136 |
| Boss (0.5pt + 0.5pt) | 0.84388 |

**Submission Format: (csv)**

```
ID,Answer
0,值勤制服
1,《諸羅縣志》
2,袁晁
3,柏林陸軍學院
4,長盤勝負
5,孫銘武
6,1874年
7,版圖向西擴張
```

# Code Submission (2pts)

- **NTU COOL**
  - **Deadline: 5/6 (Fri.) 23:59**
  - Compress your code into **<student ID>_hw7.zip** (e.g. b10901118_hw7.zip)
  - We can only see your last submission.
  - **Do not submit your model or dataset.**
  - If your code is not reasonable, your semester grade x 0.9.

# Grading -- Bonus

- If your ranking in private set is top 3, you can choose to share a report to NTU COOL and get extra 0.5 pts.

- About the report
  - Your name and student_ID
  - Methods you used in code
  - Reference
  - in 200 words
  - Deadline is same as code submission
  - Please upload to NTU COOL's discussion of HW7

Report Template

# 5. Report

# Report Questions (4 pts)

1.  (2%) After your model predicts the probability of answer span start/end position, what rules did you apply to determine the final start/end position? (the rules you applied must be different from the sample code)

2.  (2%) Try another type of pretrained model which can be found in huggingface's Model Hub (e.g. BERT -> BERT-wwm-ext, or BERT -> RoBERTa ), and describe
    - the pretrained model you used
    - performance of the pretrained model you used
    - the difference between BERT and the pretrained model you used (architecture, pretraining loss, etc.)

# 6. Regulations

# Regulations

- You should NOT plagiarize, if you use any other resource, you should cite it in the reference. (＊)
- You should NOT modify your prediction files manually.
- Do NOT share codes or prediction files with any living creatures.
- Do NOT use any approaches to submit your results more than 5 times a day.
- **Do NOT search or use additional data.**
- **Do NOT use any pre-trained models outside huggingface.**
- Your **assignment will not be graded** and your **final grade x 0.9** if you violate any of the above rules.
- Prof. Lee & TAs preserve the rights to change the rules & grades.

(＊) Academic Ethics Guidelines for Researchers by the Ministry of Science and Technology

# Links

Kaggle :

https://www.kaggle.com/c/ml2022spring-hw7

Colab :

https://colab.research.google.com/drive/1QIoQ42zYwX_ETAs2GIkeh8q Fj0UjHXfH?usp=sharing

# If any questions, you can ask us via…

- NTU COOL (recommended)
  - https://cool.ntu.edu.tw/courses/11666
- Email
  - mlta-2022-spring@googlegroups.com
  - The title **should** begin with "[hw7]"
- TA hours
  - Each Friday During Class
  - Chinese TA hour: Tuesday 20:00~21:00
  - English TA hour: Friday 22:00~23:00