DLHLP - HW4-1
BERT

TA: 紀伯翰, 謝濬丞

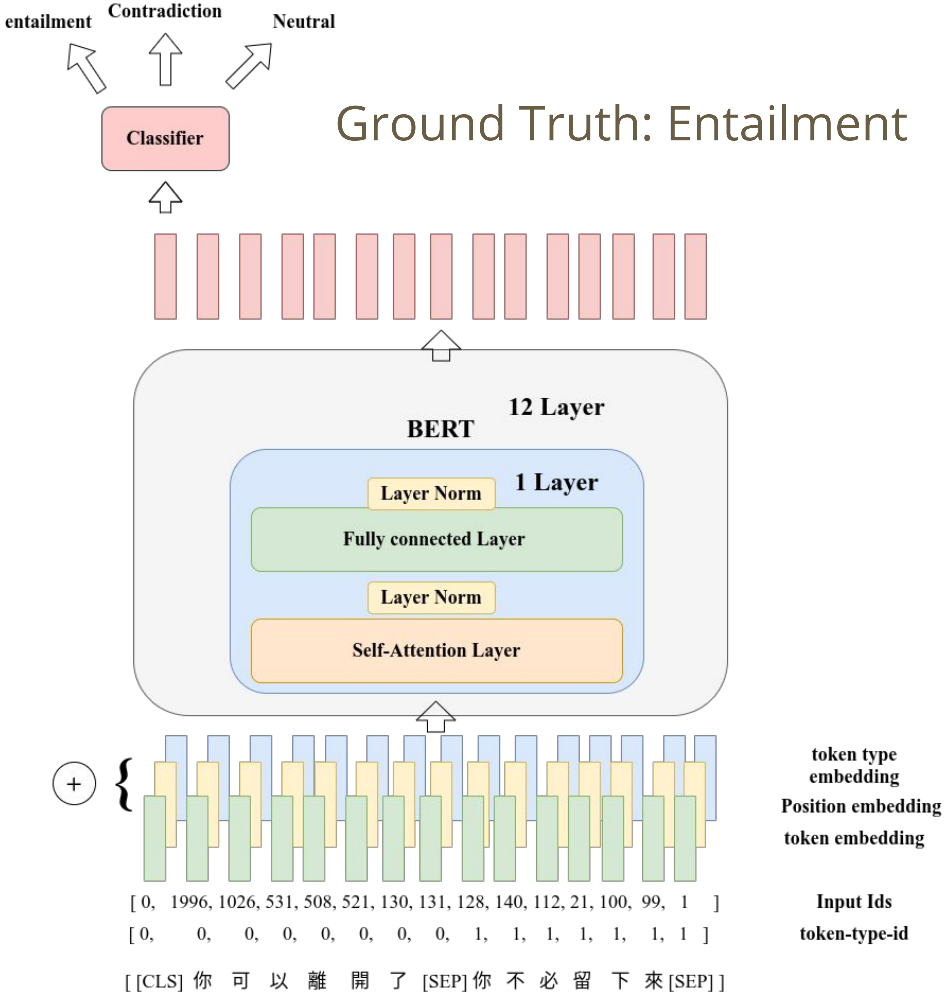
Task(s)

- Natural Language Inference Task(s) - (1)
  - XNLI-zh dataset
  - “bert-base-chinese” Model on Github
- Contextualization Issue - (2)
  - Anisotropy
  - Self-similarity
  - Intra-sentence similarity
  - Maximum explainable variance
- Chinese Word Segmentation - (3)
  - Ctb cws Dataset
  - “bert-base-chinese” Model on Huggingface

BERT - NLI
Text Entailment

Homework Task

Ground Truth: Entailment



4-1-1 (5%) Natural Language Inference - (1)

1. Finetune “bert-base-chinese” Model
2. on XNLI-zh dataset achieve 73 - 76 % performance
3. Save Model.

Step 1 Train xnli and achieve accuracy performance approximately 73-76%

Hint: Utilize `run_xnli.py`

- Train a model on multinli.train.zh.tsv and test on xnli.dev.tsv
- Plot the training loss and testing accuracy
- 3 epochs may be sufficient
- Remember to save the best model for later analysis

Contextualized v.s static

A brief history of word representations:

- pre-2018: static (skipgram, GloVe, etc.)
- post-2018: contextualized (ELMo, BERT, etc.)

On virtually every NLP task,

$$\textit{contextualized} \gg \textit{static}$$

圖片來源:
EMNLP2019

Example - (1)

Consider sentences from SemEval STS data:

- *A panda dog is running on the road.*
- *A dog is trying to get bacon off its back.*

$$\vec{dog} = \vec{dog} \implies \text{no contextualization}$$

圖片來源:
EMNLP2019

Example - (2)

Consider sentences from SemEval STS data:

- *A panda dog is running on the road.*
- *A dog is trying to get bacon off its back.*

$$\vec{dog} = \vec{dog} \implies \text{no contextualization}$$

$$\vec{dog} \neq \vec{dog} \implies \textit{some} \text{ contextualization}$$

圖片來源:
[EMNLP2019]

How to measure the Contextuality ?

- Self-similarity
- Intra-sentence similarity
- Maximum explainable variance
- Anisotropy

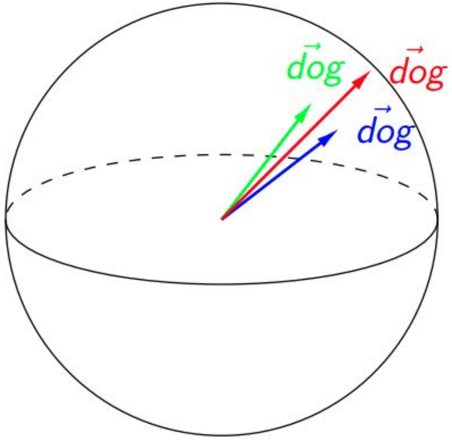
4-1-2 Definition(s)

- Self-similarity
- Intra-sentence similarity
- Maximum explainable variance
- Anisotropy

Self-similarity

- Average cosine similarity of a word with itself across all contexts, where representations are drawn from the same layer of a given model.

e.g., high self-similarity for 'dog' across contexts

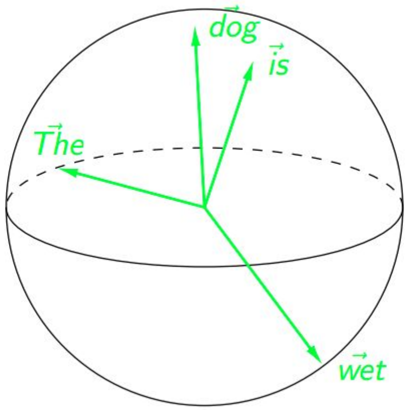


圖片來源: EMNLP2019

Intra-sentence similarity

- Average cosine similarity between a word and its context, where the context is represented as the average of its word representations.

e.g., low intra-sentence similarity for ‘The dog is wet.’

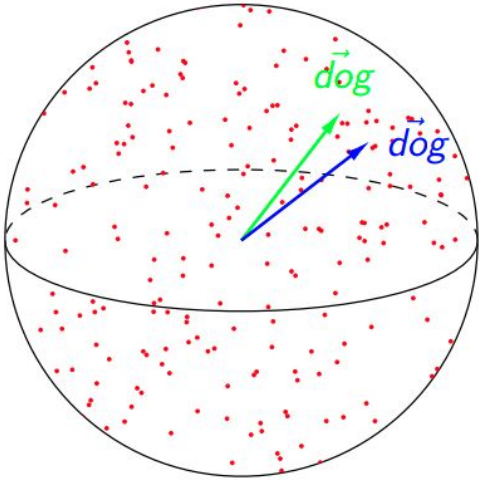


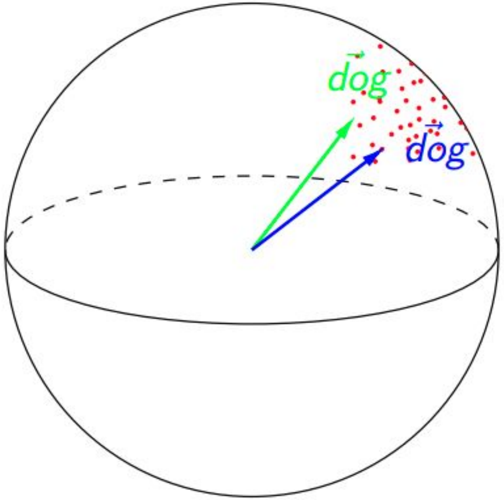
圖片來源:
[EMNLP2019]

If more Contextualized on vector space, Expect:

- Lower self-similarity
- Larger intra-sentence similarity => More Context-specific
- Lower maximum explainable variance

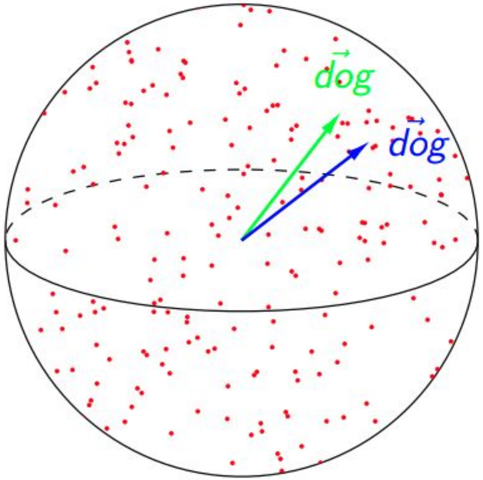
Issue ?

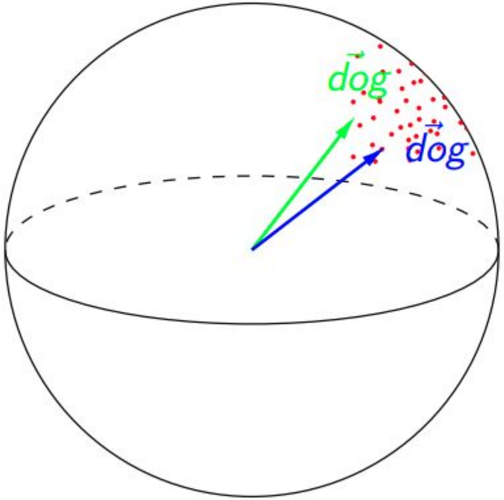




圖片來源:
EMNLP2019

Isotropy vs Anisotropy





圖片來源:
EMNLP2019

Isotropy vs Anisotropy

Self-Similarity(w) = 0.95 is relatively high if all embeddings are isotropic.

Self-Similarity(w) = 0.95 is relatively high if all embeddings are isotropic but relatively low if they are anisotropic.

Prepare Data

- Pre-trained Embedding vector
  - Use “bert-base-chinese” model

  extract embedding vector of each word on each layer.

- Fine-tuned Embedding vector
  - Use “saved model” model

  extract embedding vector of each word on each layer.

Warning

在接下來，我們要畫的圖裡面，會畫出 BERT 內 從 第 0 層 到 第 12 層的 intra-sentence similarity 以及 anisotropy 還有 self-similarity 這三個 數值，要注意的一點是 所謂的 第0層 指的就是 前面 BERT 圖示的 token embedding 的 vector，並沒有加上 positional embedding 以及 segment embedding (token type embedding)，如果你是使用助教這份 github的話，我在 這份 github的 transformers/model_bert.py 內已有做修改，在第0層的時候是會吐出embedding matrix那一層的embedding，line 375，但你如果是自己做的話，記得要去修正這一點。

4-1-2 Extract each layer Embedding(s) - (2)

Step 2 Generate pretrained data and finetune data from xnli-sample data

Example Code: `generate-similarity-data.py` (Using analysis data)(You also can write by yourself)

Write a code named `generate-similarity-data.py`

store each xnli data with its

{"input_ids": ..., "layer_0": the embedding of each data in numpy array in layer_0, "layer_1": the embedding of each data in numpy array in layer 1, "layer_2": the embedding of each data in numpy array in layer 2, ... "layer_12":the embedding of each data in numpy array in layer 12}

(generated data is a list of dict)

Hint: you need to save the data generate from pretrained model and fintuned-model (the model you save)

Anisotropy

1. 隨機sample出1000個隨機 word pair，各自算出它們的similarity 並取平均 (每一層都要) 作為那一層的 anisotropy.

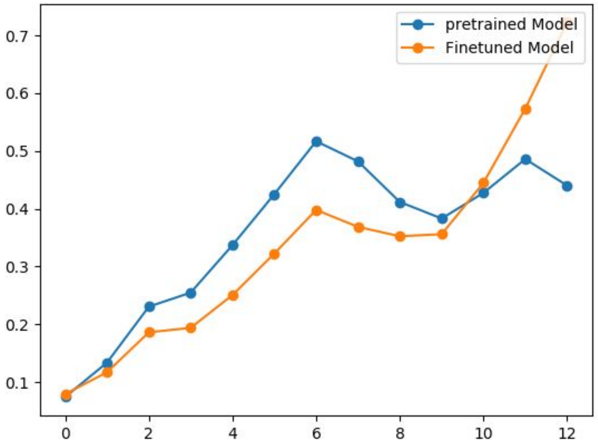
Implementation

Hint: similarity_student.py - finish todo block!

(5%) Implementation -Anisotropy

Plot the Anisotropy:

Self-similarity, Intra-sentence similarity(Anisotropy)

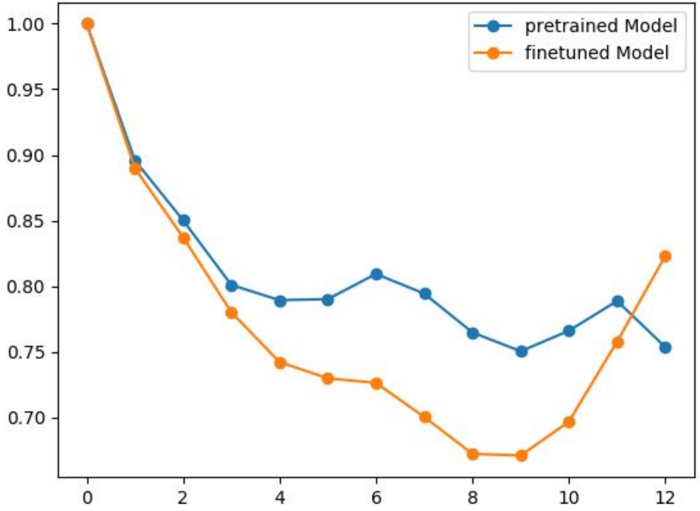


Self-similarity

每一層 對同一個 word 不同 context 兩兩做 cosine-similarity，並取平均

即為 該層的 self-similarity

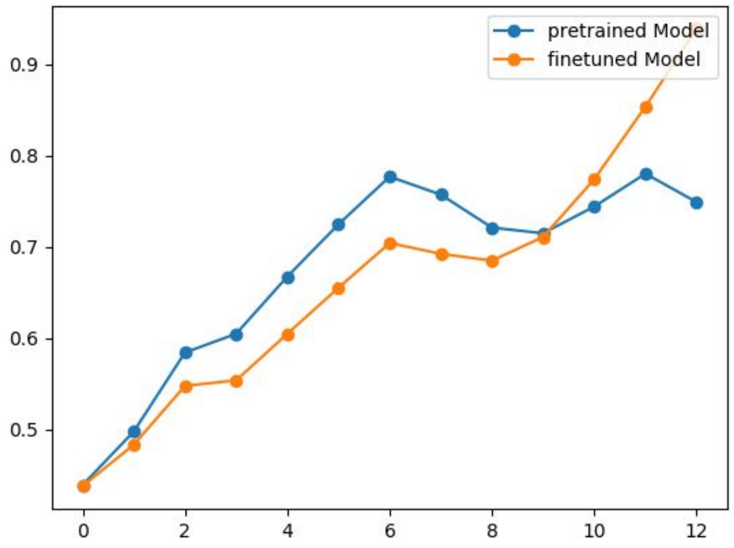
(5%)
Implementation - Self Similarity



Intra-sentence similarity

每一層，同一個句子，把句子內的字的embedding 取平均，即為該句子的 sentence embedding，並把句子內的每一個字對這句話的sentence embedding算cosine similarity，再取平均即為一筆 intra-sentence similarity，算出所有句子的 intra-sentence similarity 再取平均，即為該層的 intra-sentence similarity.

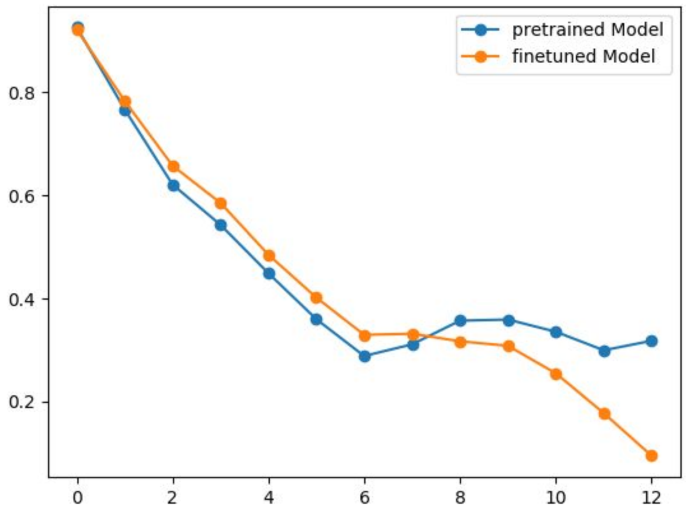
(5%) Implementation - Intra sentence similarity



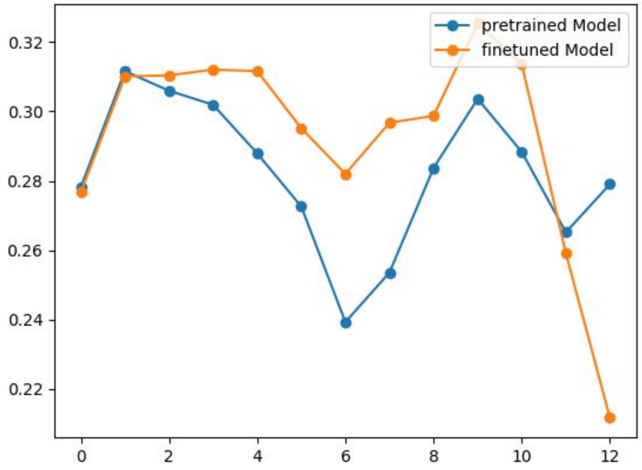
(10%) Adjusting for Anisotropy

Adjust Version:

Self-similarity



Intra-sentence similarity



Bonus: Maximum explainable variance (MEV)

- The variance explained by the first principal component of a word's representations across different contexts.

PCA (Principal Component Analysis)

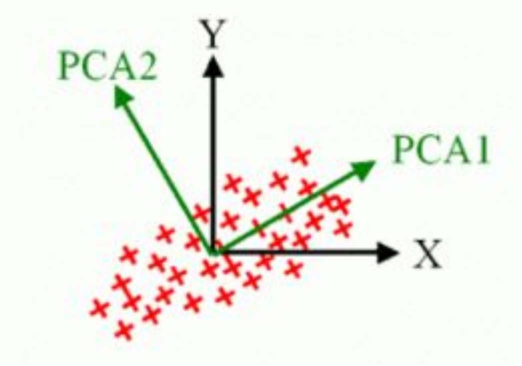
Ref: PCA , SVD

做SVD分解所對應到的最大 singular value

(singular value)^2 / 所有singular value平方和

即為 MEV 的值

參考: paper equation (3).



Chinese Characters & Words

Characters: 漢字，中文的基本組成單位。e.g. 揶、揄...

Words: 詞彙，中文裡表達意思的最小單位。e.g. 家、國、鍋、揶揄...

Chinese Word Segmentation (CWS)

我一直是在親自指揮，親自部署。

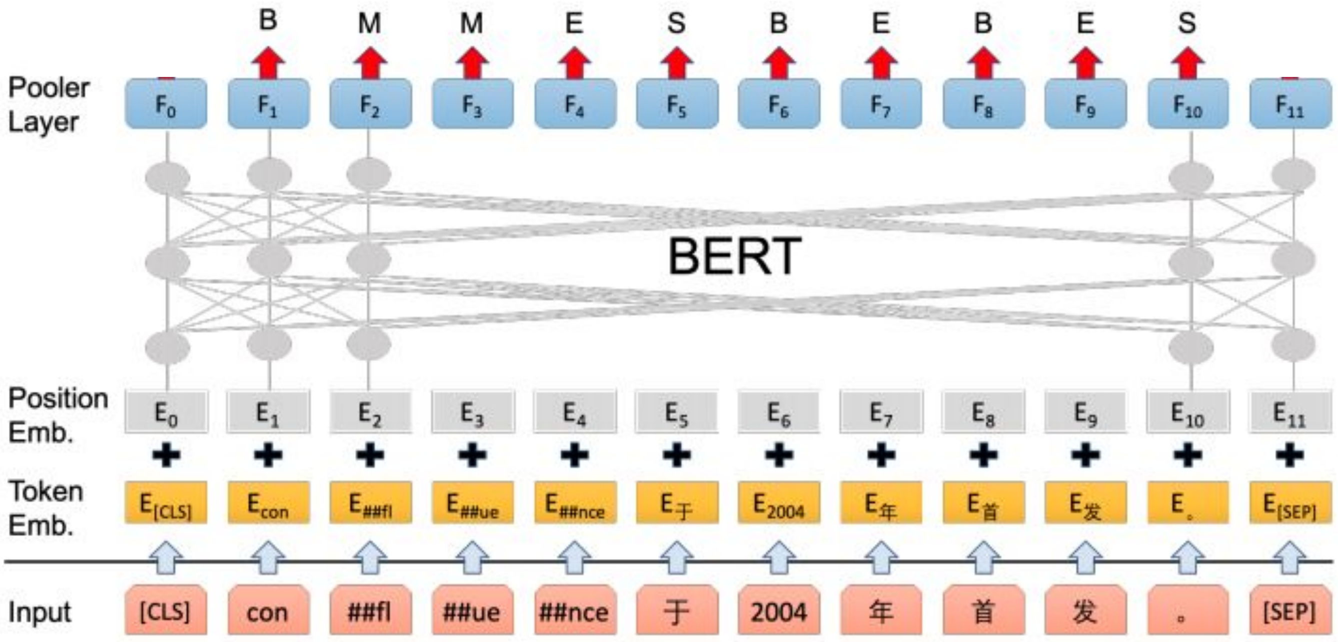
↓

我 | 一直 | 是 | 在 | 親自 | 指揮 | ， | 親自 | 部署 | 。

Task

- Transformer models always take Chinese characters but words as inputs
- Does a pretrained Chinese language model know what a Chinese word is?
- Can we make the claim that a pretrained Chinese language can do CWS if it has learnt Chinese words during pretraining?
- In this part, we formulate CWS as a Sequence Labeling task

As a sequence labeling task

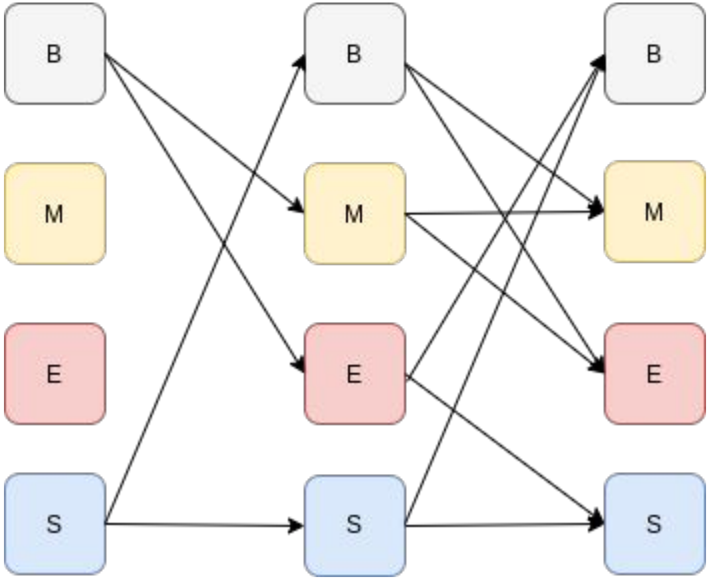


Bert Meets Chinese Word Segmentation

Prepare data

- Training & Dev data: https://github.com/Sologa/CWS_dlhlp
- Modify modeling_bert.py and utils_ner.py on Huggingface
- The labels are [‘B’, ‘M’, ‘E’, ‘S’]
- Note that the labels of a sequence are required to appear in some certain order. You may need to decode your result with Viterbi algorithm if it is not a valid sequence label
- For Viterbi algorithm or CRF, refer to Sequence Labeling tutorial

Transition Diagram



B, M, E, S,

B
M
E
S

```
[-float("Inf"), 0, 0, -float("Inf")],
[-float("Inf"), 0, 0, -float("Inf")],
[0, -float("Inf"), -float("Inf"), 0],
[0, -float("Inf"), -float("Inf"), 0]
```

Github submission

- Restricted to Python version 3.6.8 and you **cannot** import additional packages in modeling_bert.py and utils_ner.py
- We will test your code by [example.py]
- Submit the following files to YOUR_REPO/hw4/cws
  - modeling_bert.py
  - utils_ner.py
  - config.json (the config file in the same directory as your finetuned model)
  - download_model.sh for downloading your finetuned model which must be named **my_cws_bert.pt (originally named pytorch_model.bin).** It will be executed by `bash download_model.sh`
  - segmented.csv for results of the segmented sentences (1. of part2 in report) with words separated by commas

```
我,一直,親自,指揮,、,親自,部署,，,我,相信,只要,我們,堅定,信心,、,同舟共濟,、,科學,防治,、,精準,施策,，,我們,一定,會,戰勝,這,一,次,疫情,。
這,個,聲明,讓,我,再次,想起,了,安徒生,的,童話,《,皇帝,的,新,裝,》,。
希望,他們,能夠,聽,一,聽,這,個,忠告,，,不,要,再,信口雌黃,地,抹黑,，,居心叵測,地,挑撥,，,煞有介事,地,恫嚇,。
有關,部門,當然,就,是,有關,的,部門,了,。,無關,的,就,不,能,稱為,有關,部門,。,所以,我,建議,你,還是,要,向,他們,詢問,。
不,要,搞,奇奇怪怪,的,建築,。
現在,提請,表決,。,同意,的,代表,請,舉手,。,請,放下,；,不,同意,的,請,舉手,。,沒有,；,棄權,的,請,舉手,。,沒有,。,通過,!
人均,國內,生產,總值,接近,八千萬,美元,。
我,青年,時代,就,對,法國,文化,抱有,濃厚,興趣,，,法國,的,歷史,、,哲學,、,文學,、,藝術,深深,吸引,著,我,。,讀,法國,近現代史,特別是,法國,大,革命史,
輕關,易道,，,通商,寬衣,。
因為,我,那,時候,，,扛,兩百,斤,麥子,，,十,里,山路,不,換,肩,的,。
```

Report

report 連結 請一併放到 YOUR_REPO/hw4 命名為 report.pdf 一併繳交

Deadline

2020.5.27 9:00