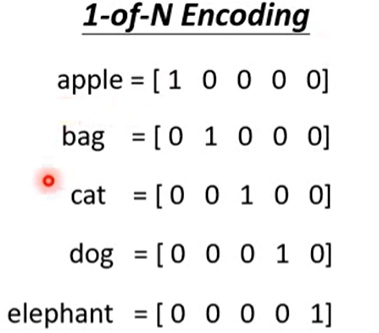
李弘毅 ELMO, BERT, GPT

1-of-N Encoding

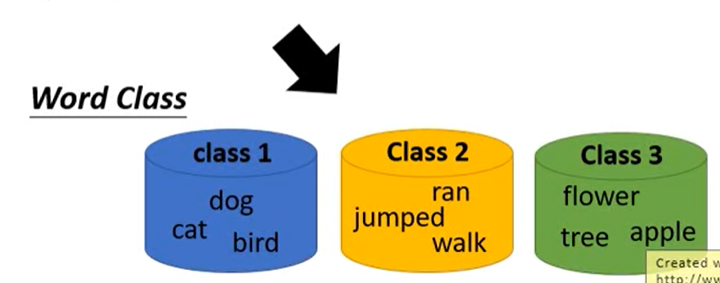


世界第一個讓機器讀文字的方式

缺點：難以讓電腦分辨出cat and dog are animals

Word Class

我們應該要分類，讓某些詞彙處於同一類別

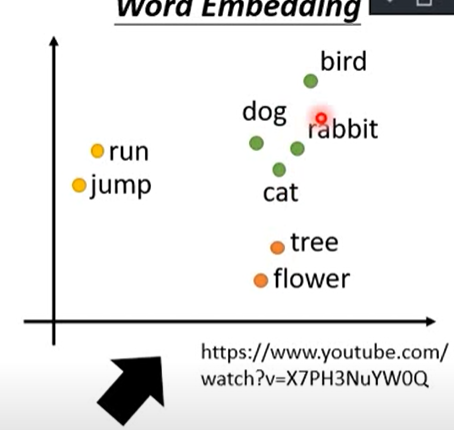


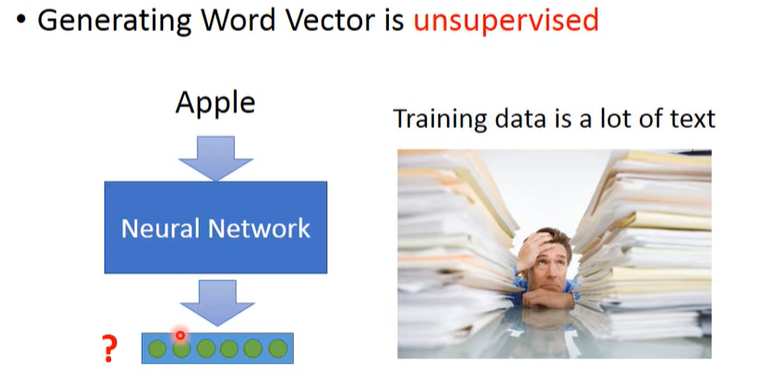
太粗糙，雖然都為動物，但哺乳類/鳥類之間的差別呢?

Word Embedding (50~100 dimension)

每一個維度都有其含義

unsupervised, machine read lots of context





將字彙用向量來表示，字彙相近的，向量在坐標軸上也比較相近

ex: dog, cat and rabbit -> 哺乳類；bird -> 鳥類

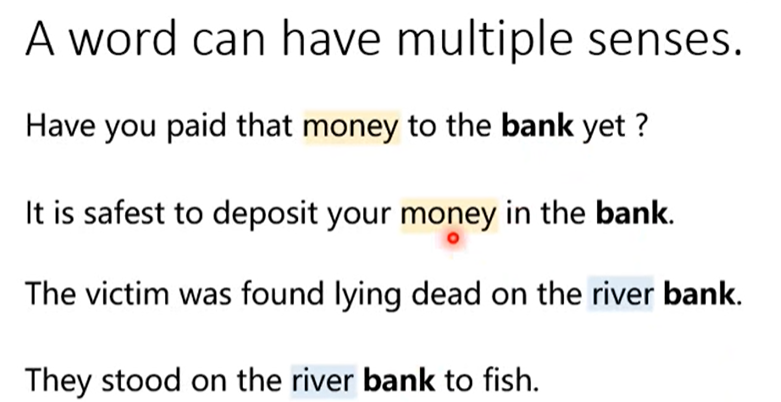
現在多為用word embedding表示一個詞彙



同一個詞彙可能有不同意思

ex: 4個bank是不同token，不同type，不同token有可能有不同的word embedding

1. 前2個句子的bank前面都有money這個詞彙 -> bank代表銀行
2. 後兩個前面有river -> bank表示河堤

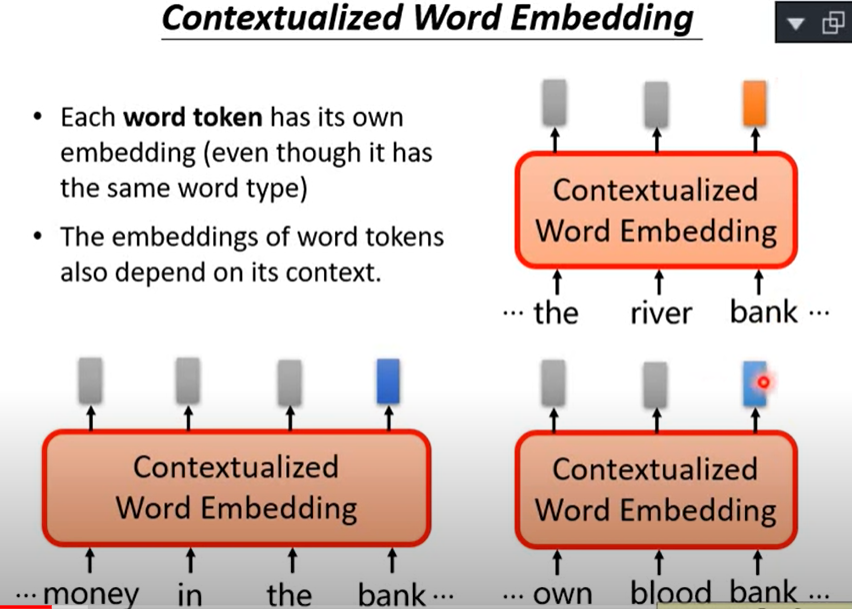




過去每一個type都有一個embedding，現在我們將不同token給不同embedding

過去是去查字典，bank這個type有2種意思，給2種embedding

-> 不夠，因為bank常有無限意思，語意太微妙



之前是一個type有一個or固定多個embedding

現在是每一個token都要有一個embedding

怎麼給呢? 看上下文，上下文愈接近的token

，有愈接近的embedding

each word token has its own embedding(even though it has

the same word type)

the embeddings of word tokens also depends on its context.

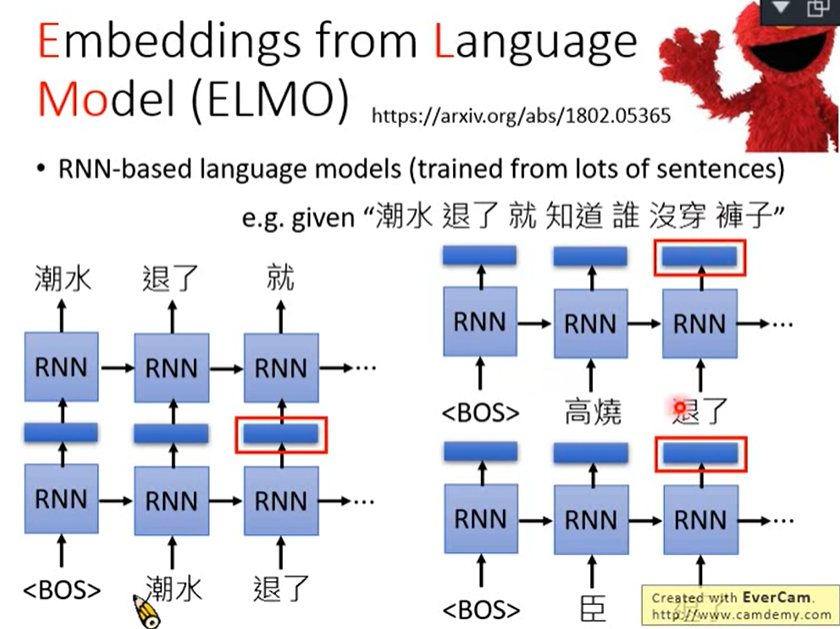
-> contextualized word embedding

下面2個bank可能有較接近的embedding，而上面那一個

bank的vector 可能離下面2個比較遠。

how to do it??? -> ELMO

RNN-based language model -> 給一堆句子，讓機器去學習如何預測下一個token會是什麼

ex: 潮水(begin with) -> 退了= 潮水退了 -> 就…

會得到embedding

ex: 讀了潮水退了 -> 得倒退了的embedding

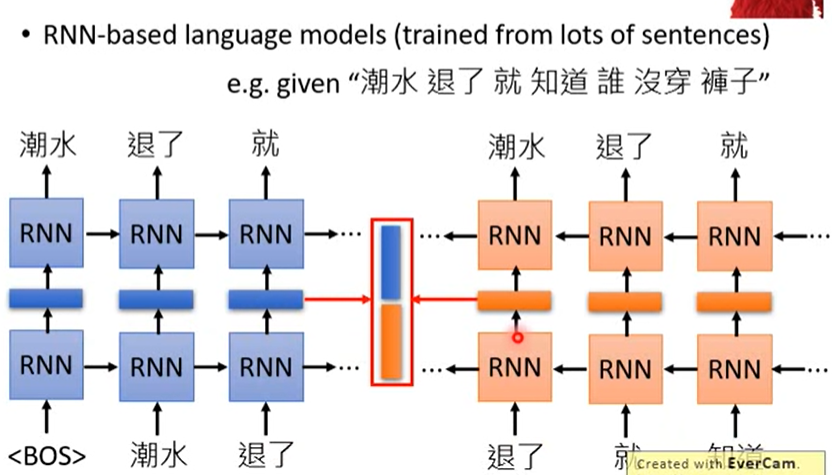
讀了 高燒退了-> 得到另一個token的embedding

讀了 臣退了 -> 又得到另一個token的embedding

好想只有考慮到前文??

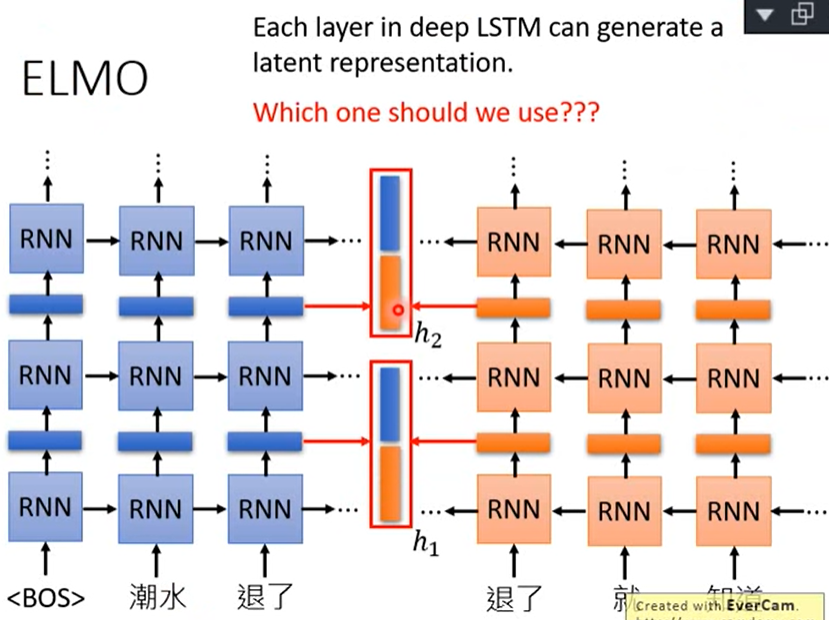
反向RNN，也考慮下文

將正向反向的embedding接起來

同一個詞彙的上下文不同，會有不同的embedding

Each layer in deep LSTM can generate a latent representation.

train deep，有很多層，同一個詞彙會有很多embedding(h1, h2, ……)，到底該用哪一層? -> 我全都要

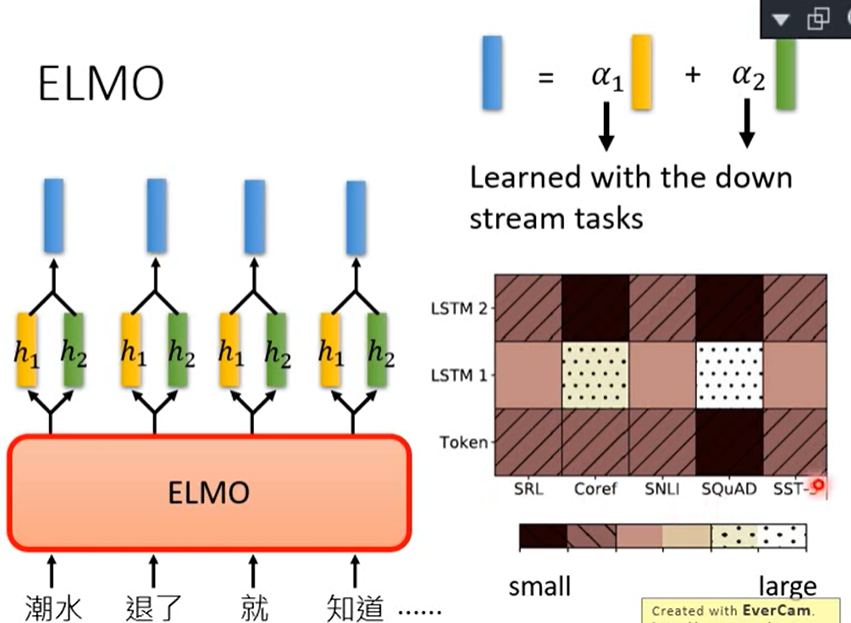


ex: RNN有2層 -> 吐出兩個embedding h1, h2 -> alpha1\*h1 + alpha2\*h2

用在接下來的down stream task的embedding有三個來源

token(原始沒有contextualized的embedding)、LSTM1(通過ELMO第一層的embedding)、LSTM2(第二層的embedding) -> 去做weighted sum 來得到藍色的那個embedding

以下圖例：不同的任務學出不同的weight



ex: Coref、SQuAD特別需要第一層的contextualized word embedding

BERT

先去看懂sequence2sequence, transformer等等

<https://leemeng.tw/neural-machine-translation-with-transformer-and-tensorflow2.html>

再看

<https://leemeng.tw/attack_on_bert_transfer_learning_in_nlp.html>

transformer的encoder其實就是BERT的network架構

原來要train一個transformer的時候，需要有一些task(ex: summarization, translation)，給transformer input並告訴他正確的output是甚麼

BERT只要train transformer裡面的encoder就好，且不需要有label資料，只要收集一大堆句子

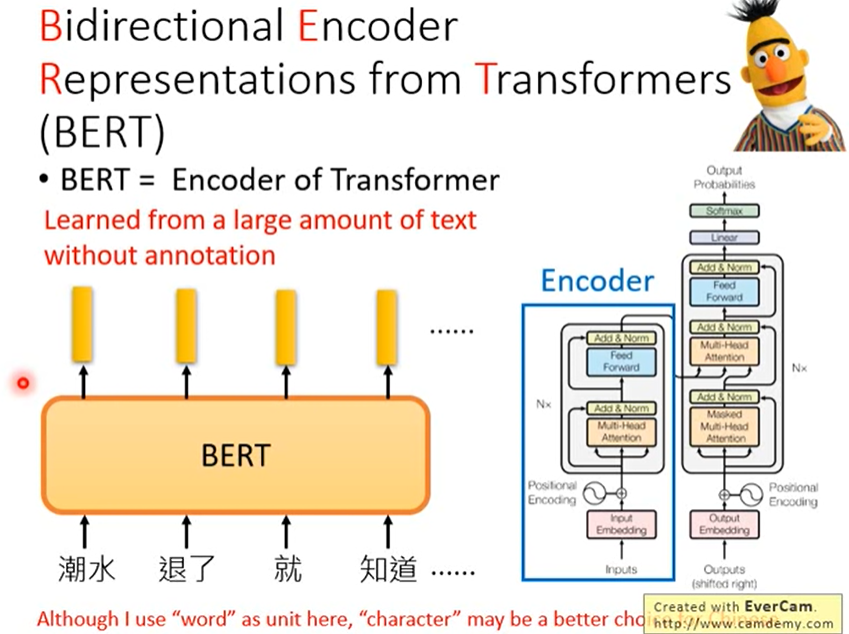
BERT是一個什麼東西?

給一個句子進去，每一個句子都會吐一個embedding給你

input word sequence output 一串embedding，某一個embedding對應到某一個input word

注意: 訓練中文詞彙，最好用字來當作單位

因為中文的詞是無限的，input one hot vector維度太大，而字的character是有限的，描述中文的character的one hot encoder的vector就不會太長



如何訓練BERT?????

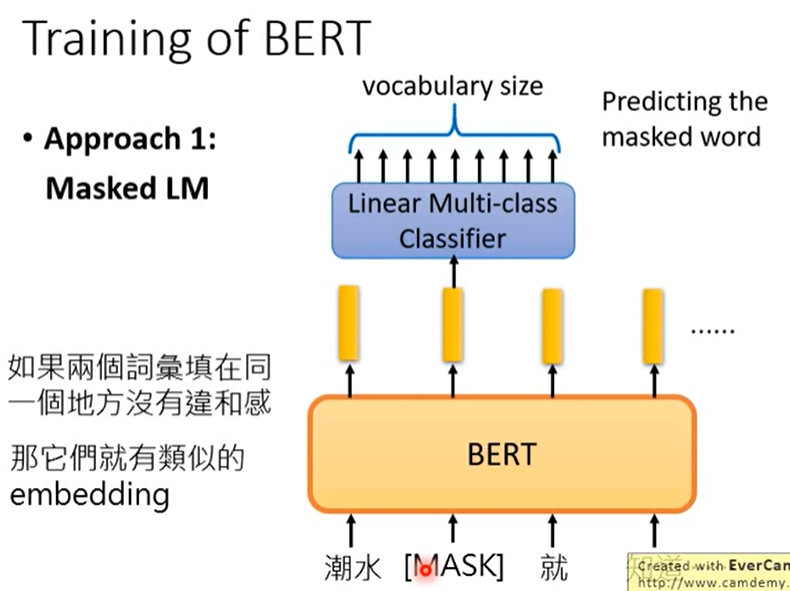
第一個訓練方法

Masked LM: 把所有的句子，15%的詞彙被置換成[MASK]

去猜測這些有蓋住的地方，到底是什麼詞彙(克漏字的意思)

將這些input token丟進去BERT -> 都會得到一個embedding

接下來將第二個(挖空的那個)embedding，丟到一個linear multi-class calssifier裡面，預測那個被MASK的詞彙是哪一個詞彙，因為是linear，預測能力非常弱，所以假如BERT有24或48層，那BERT這個model一定要剛好抽到一個非常好的representation(才預測的出來這個被MASK掉的辭彙是哪個)，可想而知，BERT到時候抽出來的representation(embedding)會是一個兩個詞彙填再同一個地方，沒有違和感的embedding(ex: 退了、落了)



第二個訓練方法

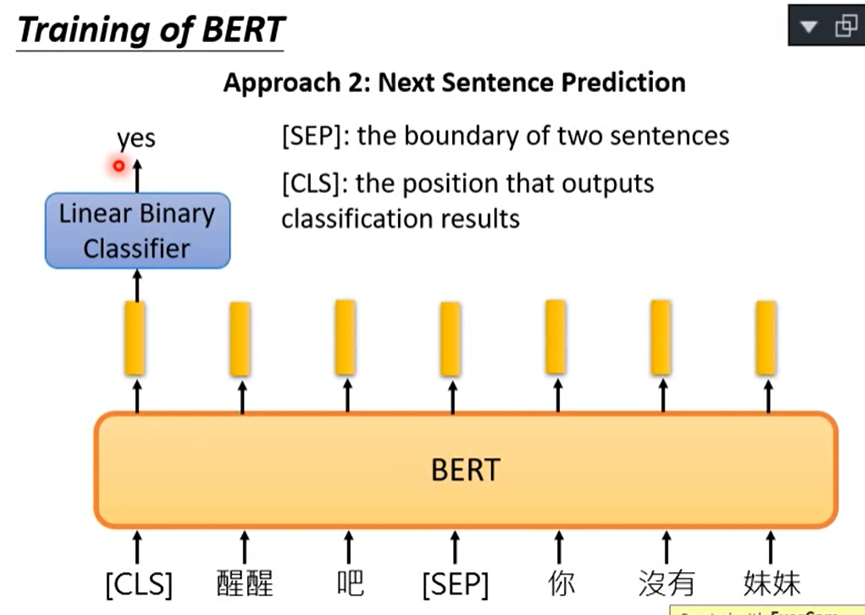
Next Sentence Prediction: 給他兩個句子，讓BERT去判斷這兩個句子，是接在一起的，還是不是接在一起的。

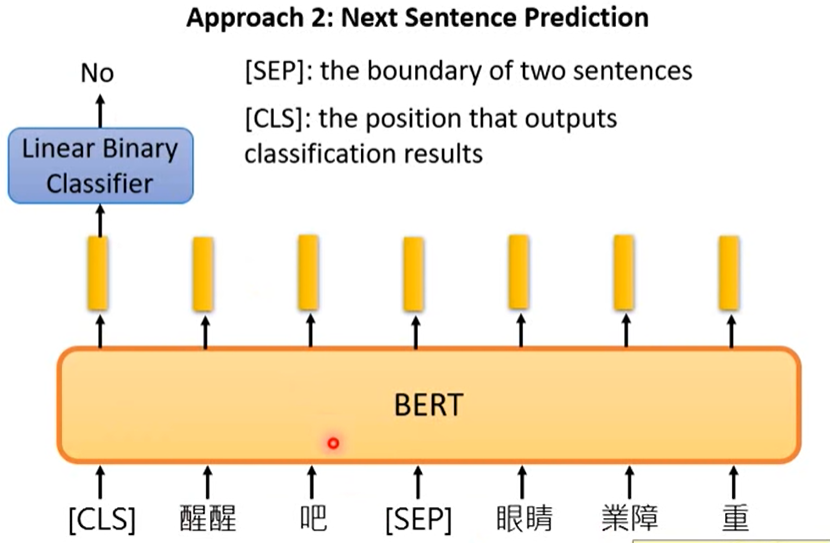
ex: 給BERT兩個句子：醒醒吧、你沒有妹妹，讓其判斷是否是接在一起的兩個句子。

[SEP] -> 告訴BERT兩個句子中間的交界在哪裡

句子的開頭放一個特殊的token: [CLS] -> 這個token輸出來的embedding丟到一個linear binary (yes/no) classification(是否接在一起)

為何在開頭? -> BERT內部是transformer，放在句首/尾沒差





方法一、方法二are used at the same time

現在每一個word(或character)都有他的contextualized word embedding了，那要如何用它呢??

例子一

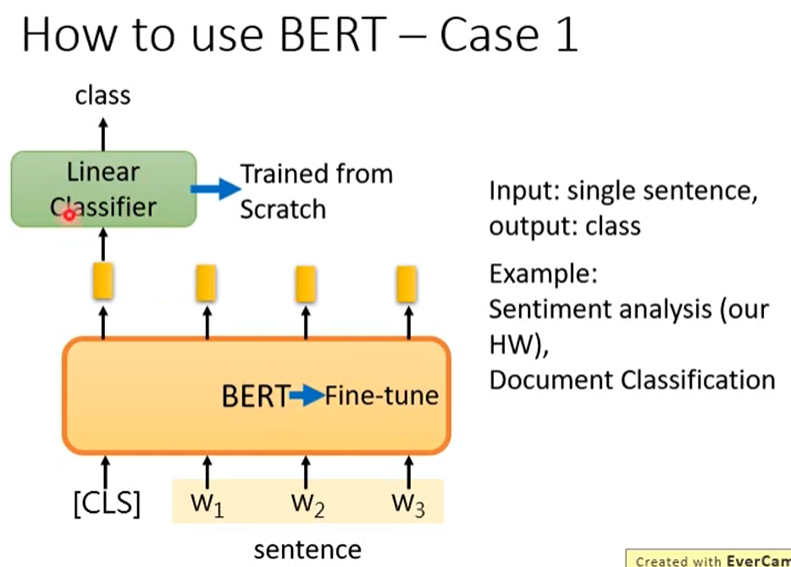
input sentence output classification

ex: 給一段句子，判斷他是正面還是反面

開頭給一個分類的token [CLS]、w1 w2 w3…..等sentence

CLS output的embedding丟到linear classifier，預測句子是正面還是反面

(需要從頭學的參數只有linear classifier，其他都已經train好了)

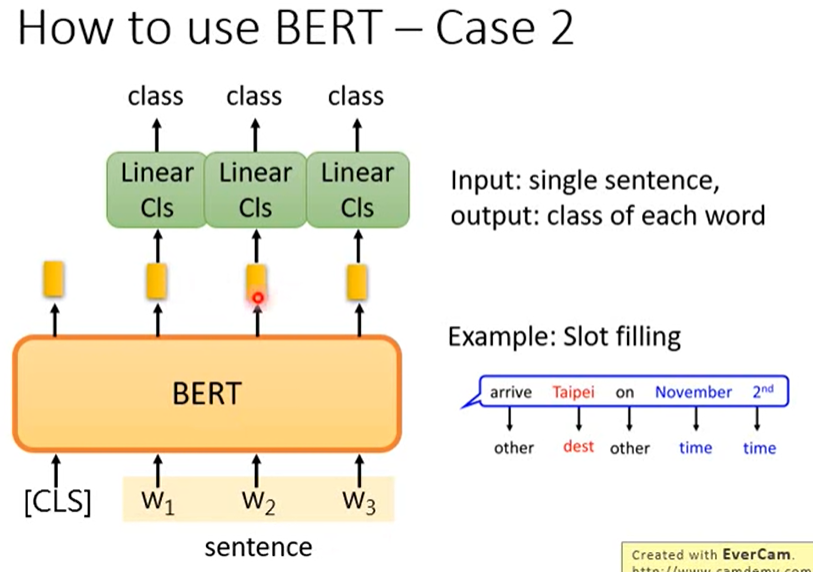


例子二

input single sentence output the class of each word

給定句子裡面的word要分類成甚麼output，讓各個linear classifier從頭開始學

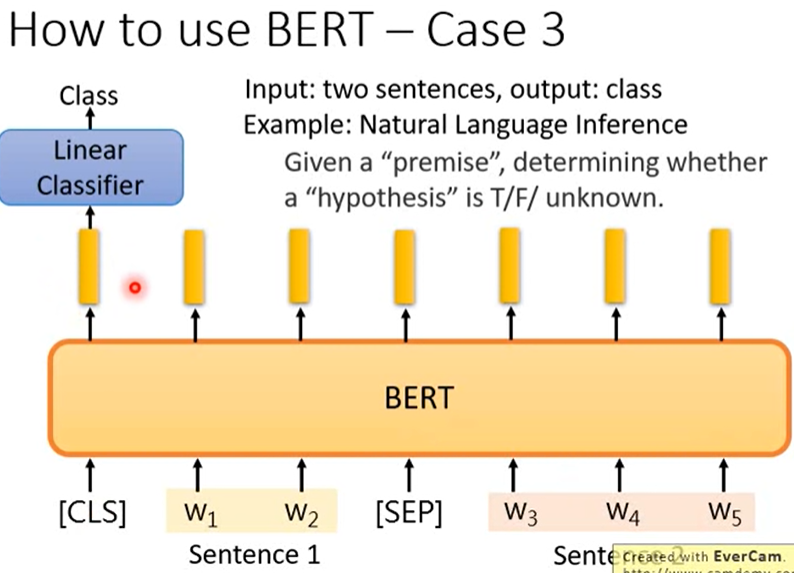
ex: arrive -> other；Taipei -> dest …….



例子三

input 2 sentences output class

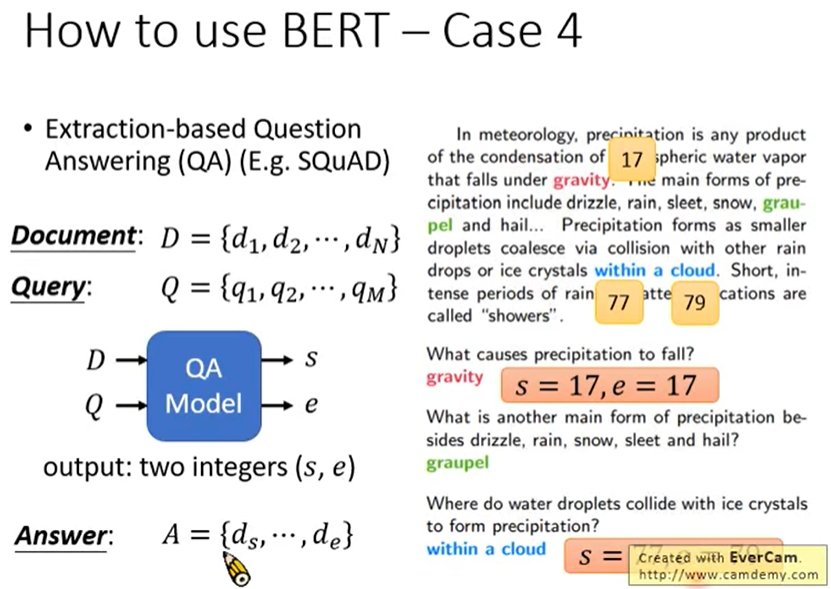
Natural Language Inference: 要機器學會透過這兩句話去推論。給機器一個前提(premise)，再給他一個假設(hypothesis)，問機器根據這個前提假設，到底是對/錯/不知道。



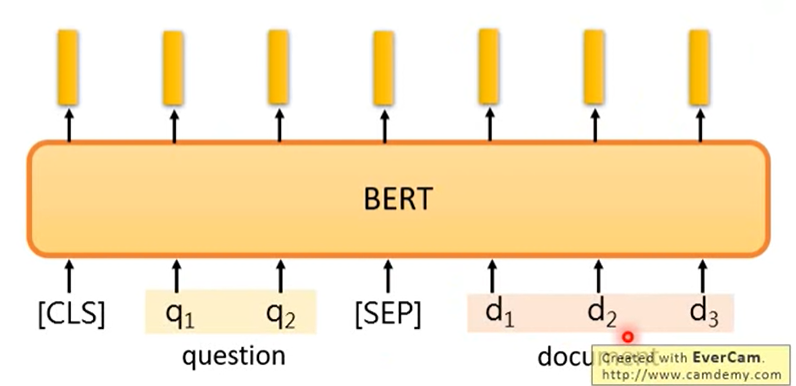
例子四

Extraction-based Question Answering

給BERT一篇文章，要BERT回答一段問題的答案，而這個答案一定出現在文章中。



question輸進去，給分隔符號[SEP]，再把文章輸進去

文章裡面每一個詞彙都會產生一個embedding

接下來讓machine去learn另外2個vector(紅藍)

這2個vector的維度跟黃色的vector一樣

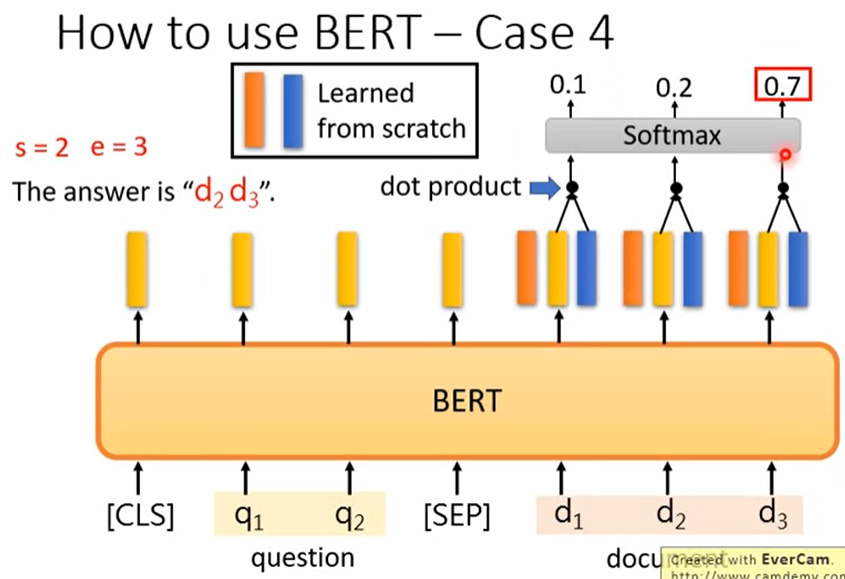
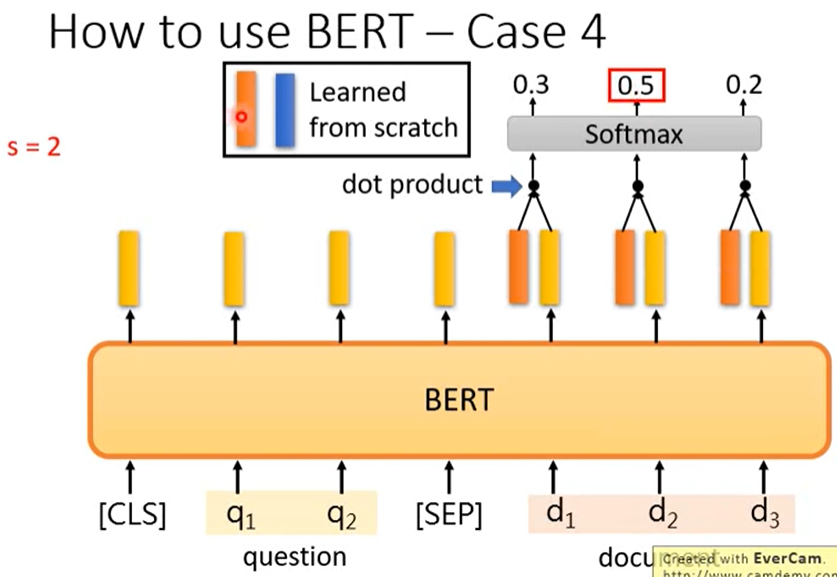
之後拿去跟每一個黃色的vector做dot product

都會算出一個scalar -> 算出softmax(類似分數)

取最高分的

紅色的vector決定s等於多少

藍色的vector決定e等於多少



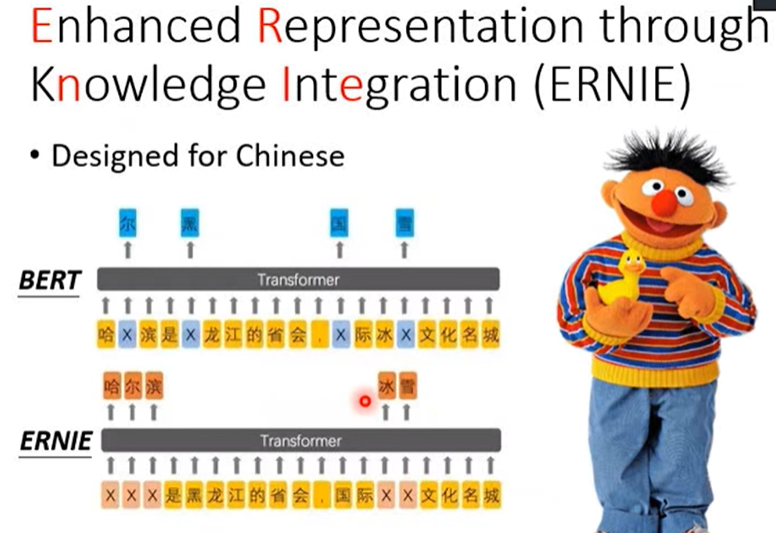
如果今天output的答案，e落在s前面(ex: s=3, e=2) -> 沒有答案(此題無解也是個答案)

注意: A = {ds, ……., de}，s應該落在e前面!!!

ERNIE

特別為了中文而設計的模型(BERT的好朋友)

BERT在做Masked LM時，如果是中文句子，只蓋掉一個字非常容易猜出來，因此改為蓋掉多個字。



BERT每一個層在做什麼?

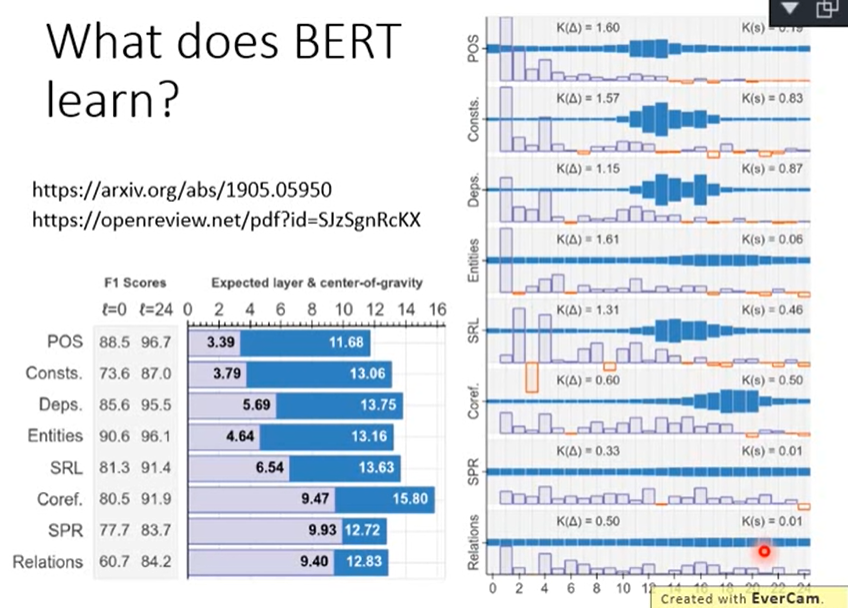
ex: 比較接近input的層，可能做一些簡單的文法的東西，接近output則比較複雜

把BERT裡面的24層的vector(24個)抽出來，去做weighed sum(like ELMO)

看每一個NLP的任務任的weight怎麼樣，就可以看出來這個任務特別需要BERT的那些層

任務舉例: POS packing 詞性標記，去標記每一個詞彙是甚麼詞性

最需要BERT第11~13層



Multilingual BERT

trained on 104 language

讀過104種語言，自動學到了不同語言之間的對應關係

ex: 給BERT英文的文章分類，BERT也自動學會去做中文文章的分類!!!

