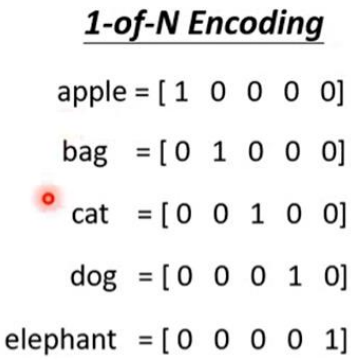
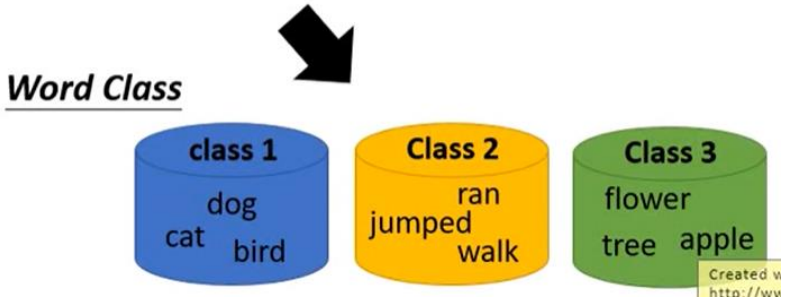
李弘毅 ELMO, BERT, GPT

1-of-N Encoding



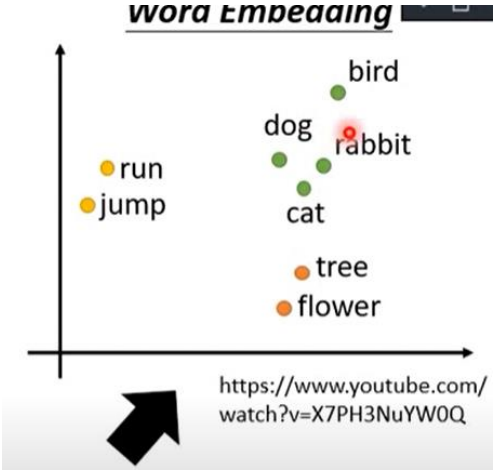
世界第一個讓機器讀文字的方式
缺點：難以讓電腦分辨出 cat and dog are animals

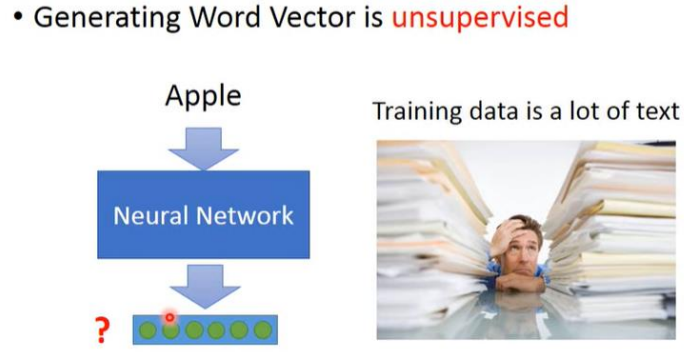
Word Class
我們應該要分類，讓某些詞彙處於同一類別



太粗糙，雖然都為動物，但哺乳類/鳥類之間的差別呢?

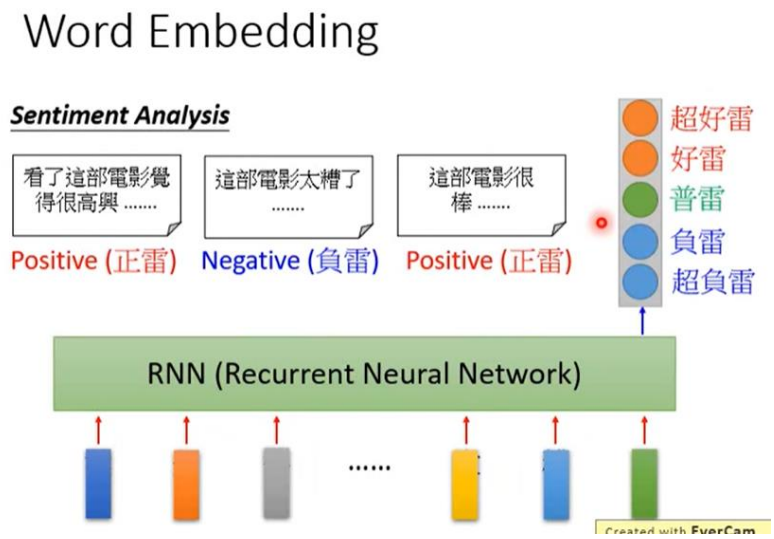
Word Embedding (50~100 dimension)
每一個維度都有其含義
unsupervised, machine read lots of context





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

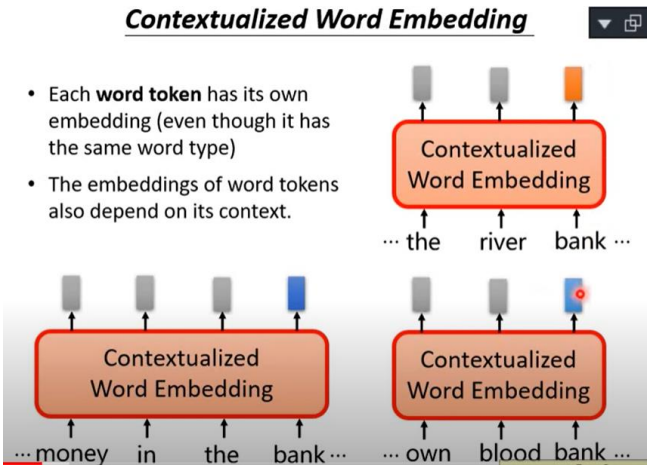
現在多為用 word embedding 表示一個詞彙



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

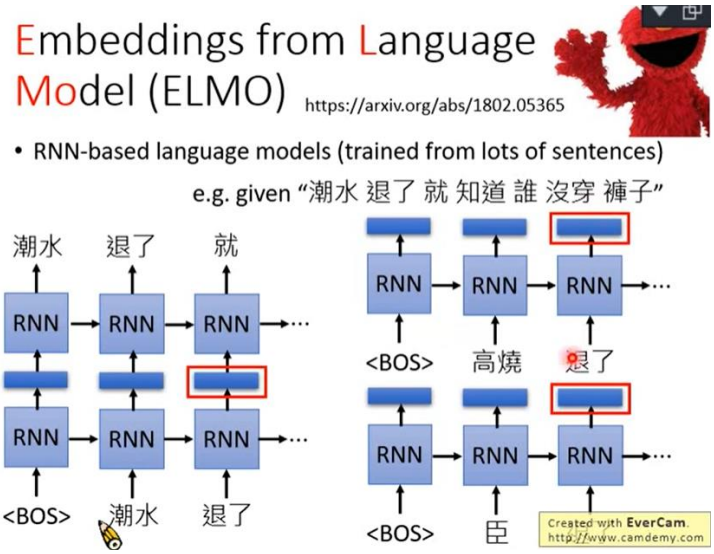
ex: 潮水(begin with) -> 退了= 潮水退了 -> 就…

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

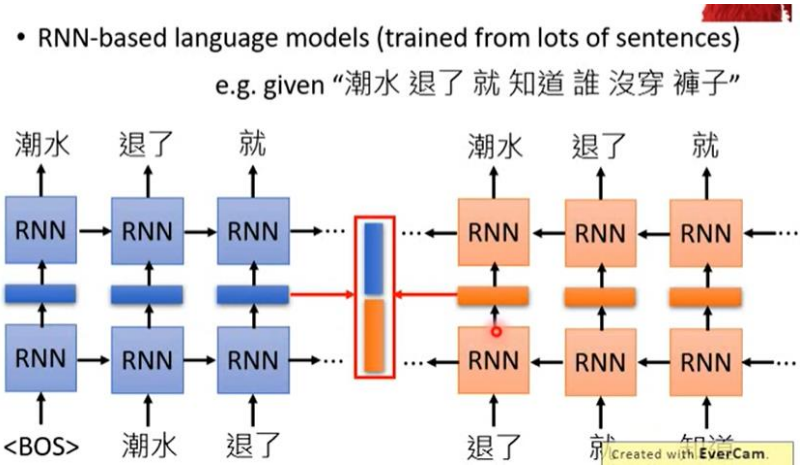


好想只有考慮到前文??

反向 RNN，也考慮下文

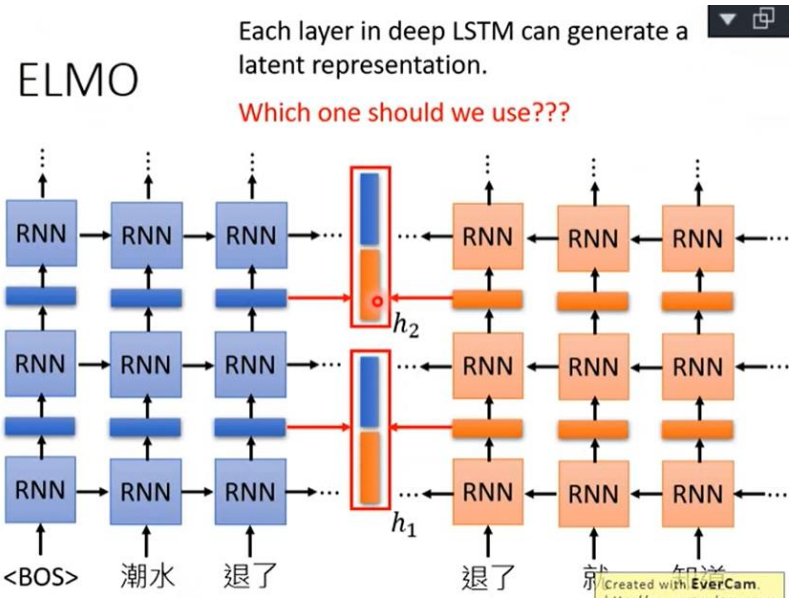
將正向反向的 embedding 接起來

同一個詞彙的上下文不同，會有不同的 embedding

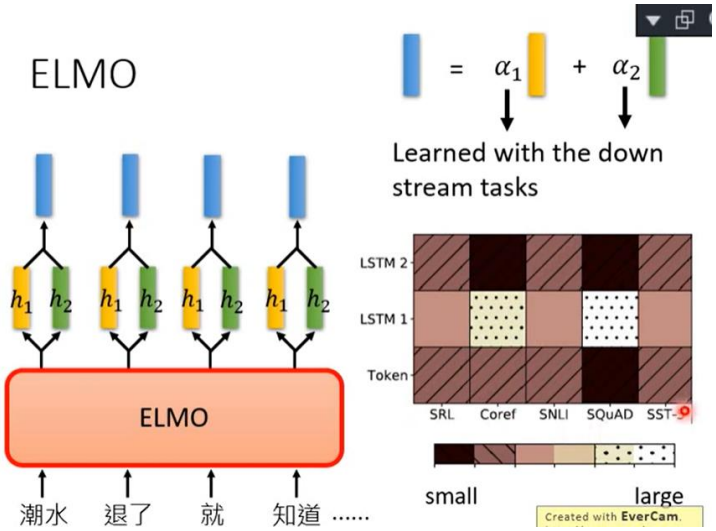


Each layer in deep LSTM can generate a latent representation.

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



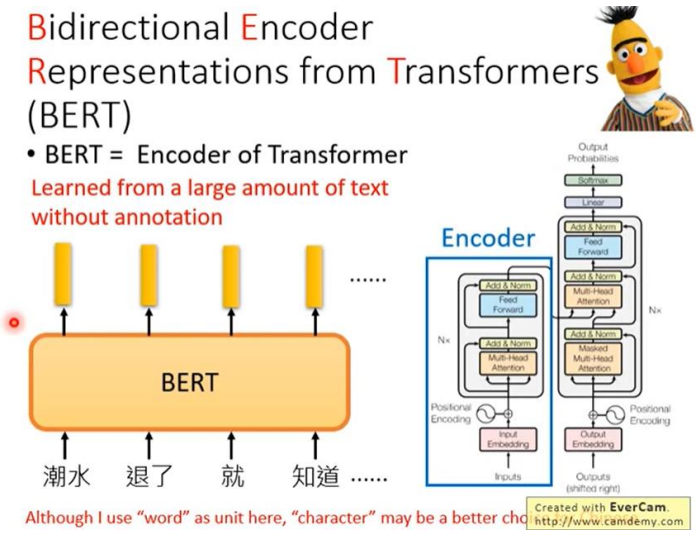
ex: Coref、SQuAD 特別需要第一層的 contextualized word embedding

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

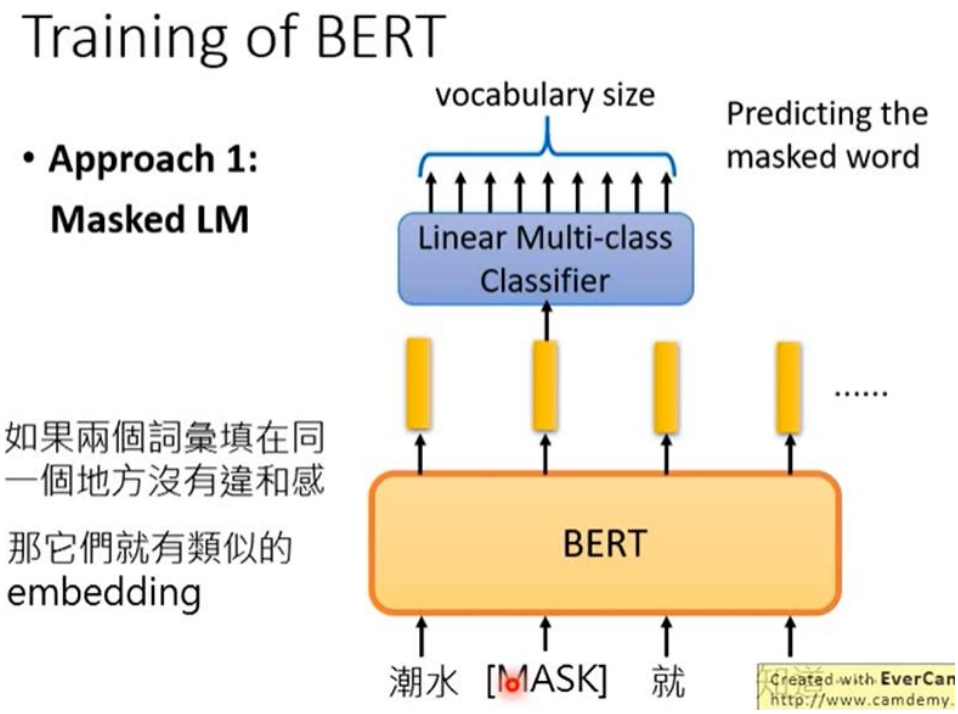
BERT 是一個什麼東西?
給一個句子進去，每一個句子都會吐一個 embedding 給你
input word sequence output 一串 embedding，某一個 embedding 對應到某一個 input word
注意: 訓練中文詞彙，最好用字來當作單位
因為中文的詞是無限的，input one hot vector 維度太大，而字的 character 是有限的，描述中文的 character 的 one hot encoder 的 vector 就不會太長



如何訓練 BERT?????

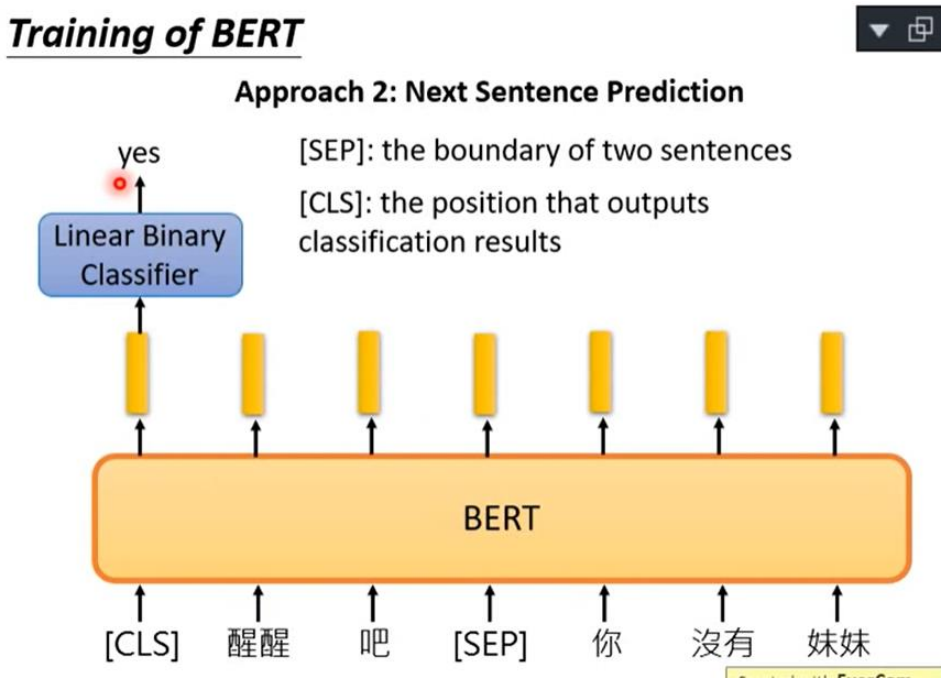
第一個訓練方法

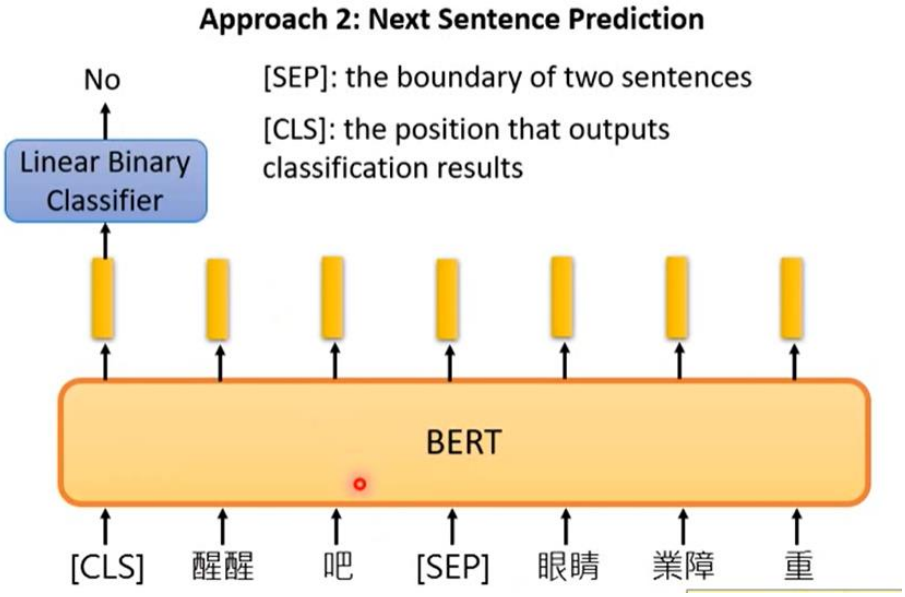
Masked LM: 把所有的句子，15%的詞彙被置換成[MASK]
去猜測這些有蓋住的地方，到底是什麼詞彙(克漏字的意思)
將這些 input token 丟進去 BERT -> 都會得到一個 embedding
接下來將第二個(挖空的那個)embedding，丟到一個 linear multi-class calssifier 裡面，預測那個被 MASK 的詞彙是哪一個詞彙，因為是 linear，預測能力非常弱，所以假如 BERT 有 24 或 48 層，那 BERT 這個 model 一定要剛好抽到一個非常好的 representation(才預測的出來這個被 MASK 掉的辭彙是哪個)，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



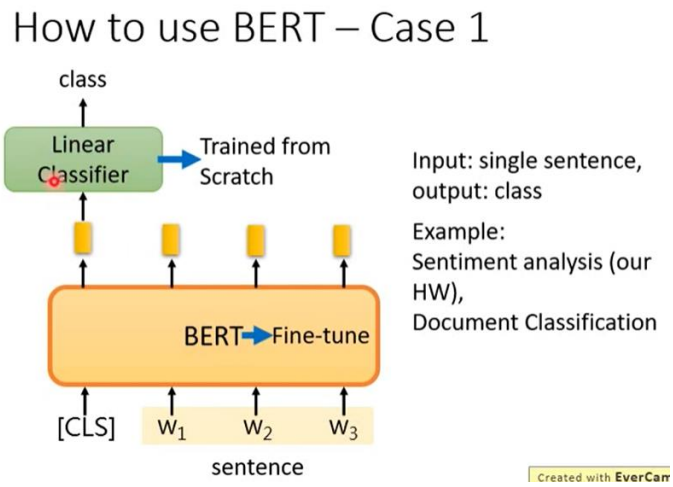


方法一、方法二 are used at the same time

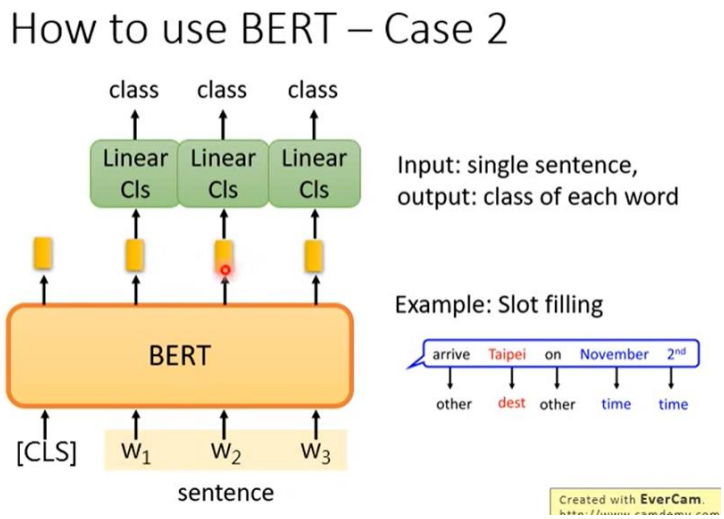
現在每一個 word(或 character)都有他的 contextualized word embedding 了，那要如何用它呢??

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)



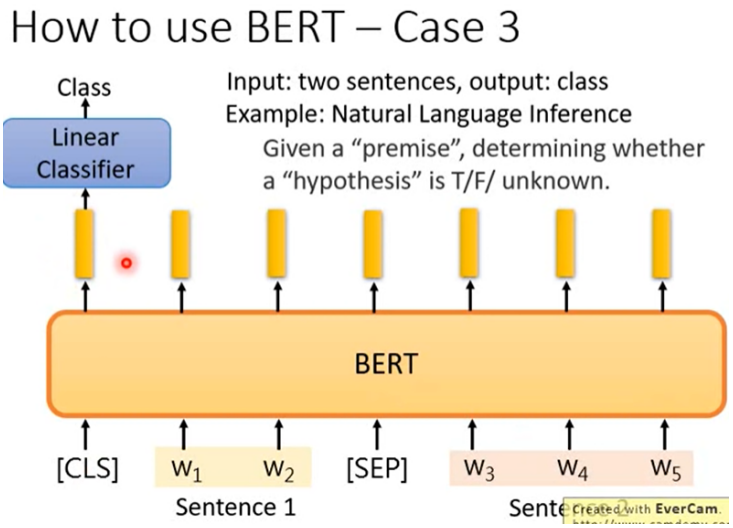
例子二
input single sentence output the class of each word
給定句子裡面的 word 要分類成甚麼 output，讓各個 linear classifier 從頭開始學
ex: arrive -> other；Taipei -> dest …….



例子三

input 2 sentences output class

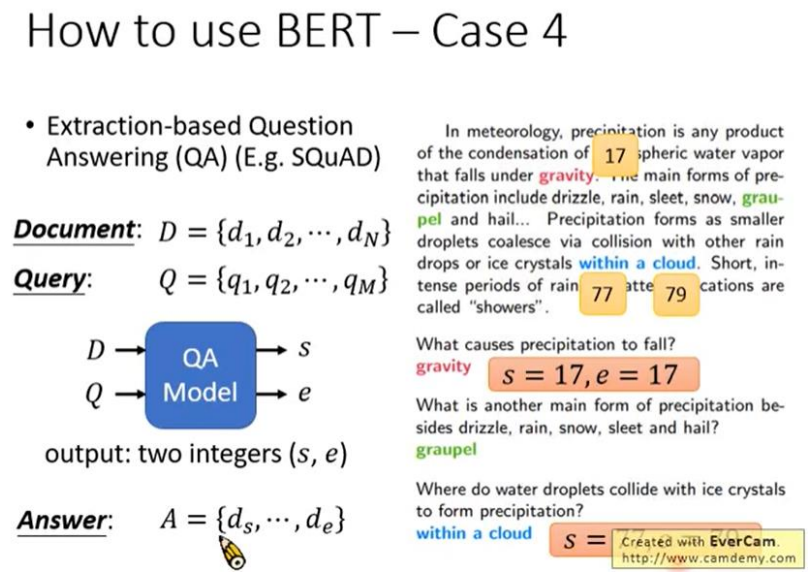
Natural Language Inference: 要機器學會透過這兩句話去推論。給機器一個前提(premise)，再給他一個假設(hypothesis)，問機器根據這個前提假設，到底是對/錯/不知道。



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

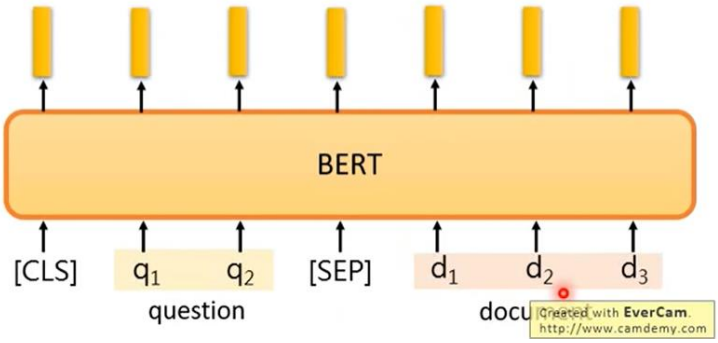
接下來讓 machine 去 learn 另外 2 個 vector(紅藍)

這 2 個 vector 的維度跟黃色的 vector 一樣

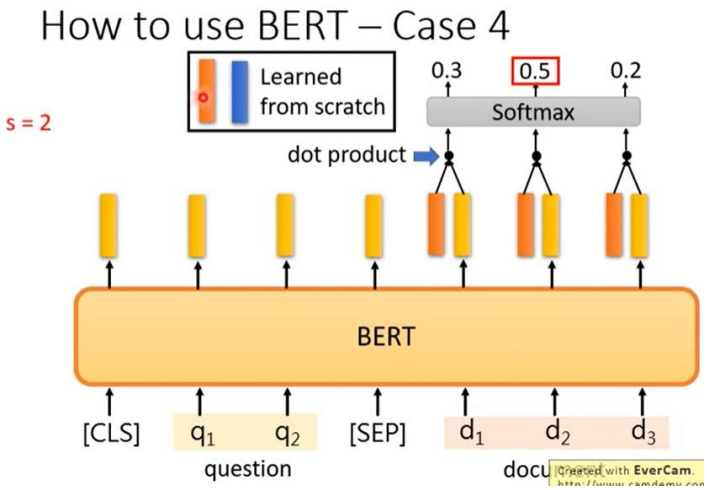
之後拿去跟每一個黃色的 vector 做 dot product

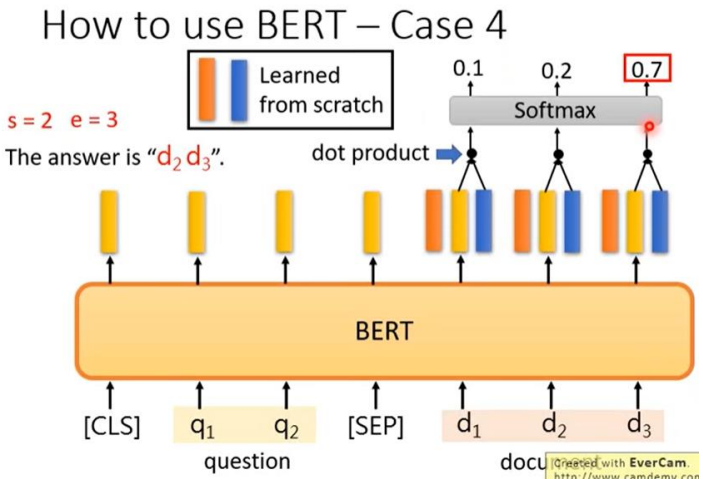
都會算出一個 scalar -> 算出 softmax(類似分數)

取最高分的



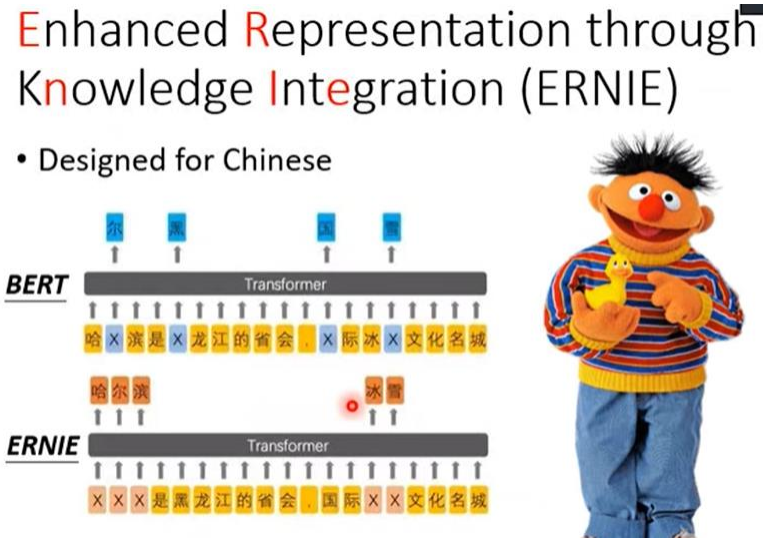
紅色的 vector 決定 s 等於多少
藍色的 vector 決定 e 等於多少



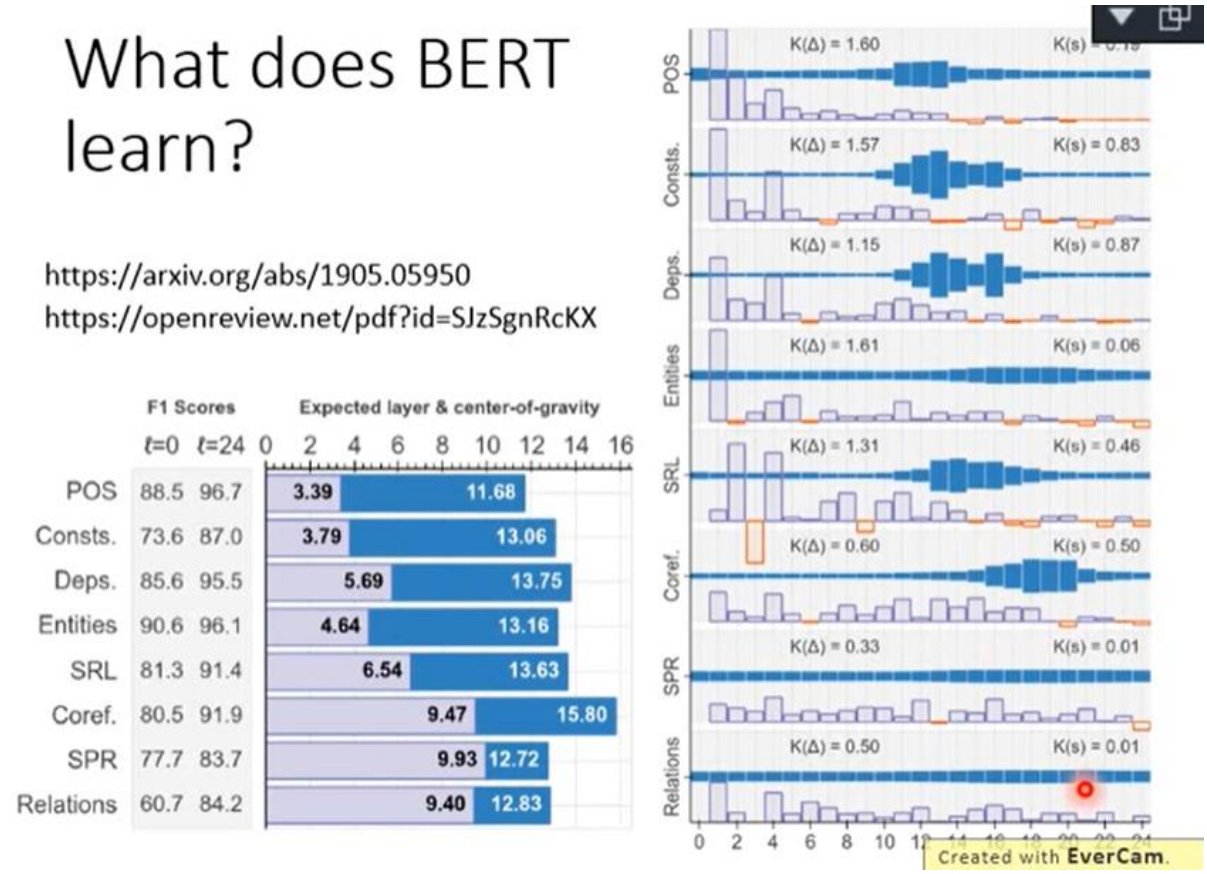


如果今天 output 的答案，e 落在 s 前面(ex: s=3, e=2) -> 沒有答案(此題無解也是個答案)
注意: A = {ds, ……., de}，s 應該落在 e 前面!!!

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!!!