Locality Sensitive Hashing on ALBERT
使用位置敏感雜湊的ALBERT語言模型

組別：A47　組員：106061146 陳兆廷 106000147 沈永聖　指導教授：孫民

一、前言

在自然語言模型越趨龐大的時代，如何降低運算空間或時間是極其重要的。因此有語言模型ALBERT，A lite BERT 的出現，大幅減少了運算時間及空間的需求，而另一個語言模型，Reformer，所提出的注意力機制：Locality Sensitive Hashing（位置敏感雜湊），雖然在空間複雜度上進步許多，但並沒有在更進階的自然語言工作中實作。因此我們將實驗若是將 **ALBERT 的注意力機制替換至 Locality Sensitive Hashing**，是否也能達到相當不錯的效果。

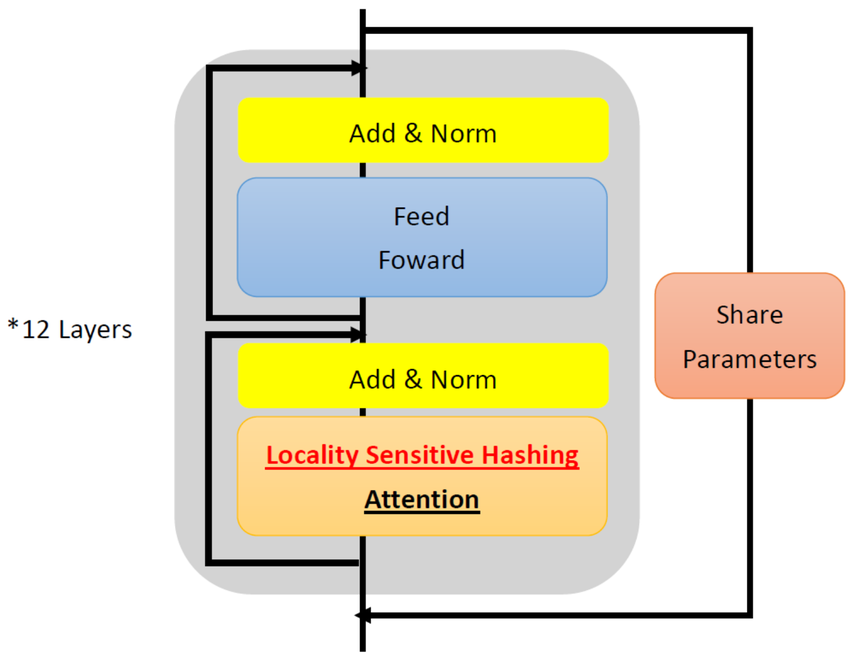
二、實驗目的

了解運用 Locality Sensitive Hashing 注意力機制的 ALBERT 是否能達成更困難之自然語言處理工作。

三、模型介紹

架構

我們將原先 ALBERT 的注意力層替換為 Locality Sensitive Hashing 注意力層（圖一），並保留其餘 ALBERT 之參數及特徵，替換為 LSH 後能將空間複雜度從 $O(N^2)$ 降低至 $O(N \lg N)$。

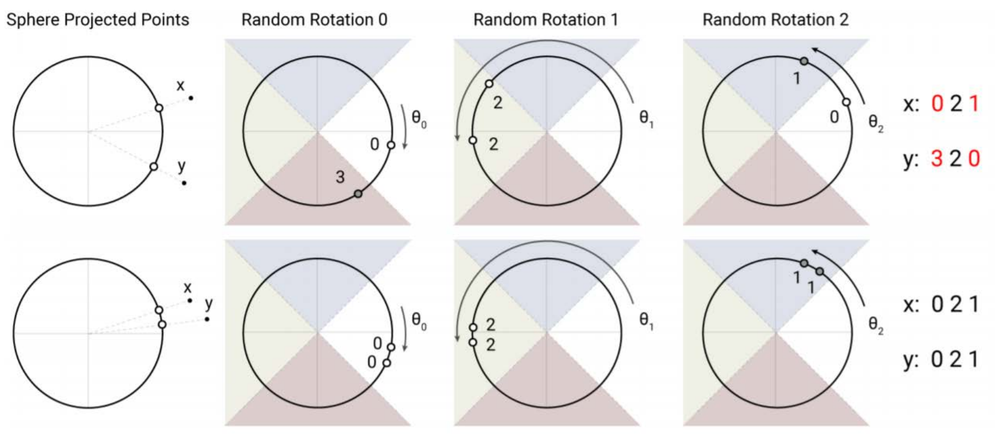


圖一
模型架構（每層）

我們使用 Multiprobe scheme for the cross-polytope LSH 作為雜湊之方法，透過一個隨機高斯旋轉矩陣與 $k_j = \frac{q_j}{\|q_j\|}$ 相乘，相當於將k_j在一個二維平面上做隨機旋轉（圖二），以式子表示如下式：

$$Pr_{A^{(1)},\ldots,A^{(i)}}\left[h_i(p) = r_{v_i}^{(i)} \, for\ all\ i \in [k] \middle|\ \mathrm{A}^{(i)}q = x^{(i)}\right]$$

其中A代表隨機高斯旋轉矩陣、 x為q旋轉後得到之雜湊值、 r_v代表x在各個分區依絕對值大到小排序得到的第v個值。



圖二
文字編碼雜湊方法

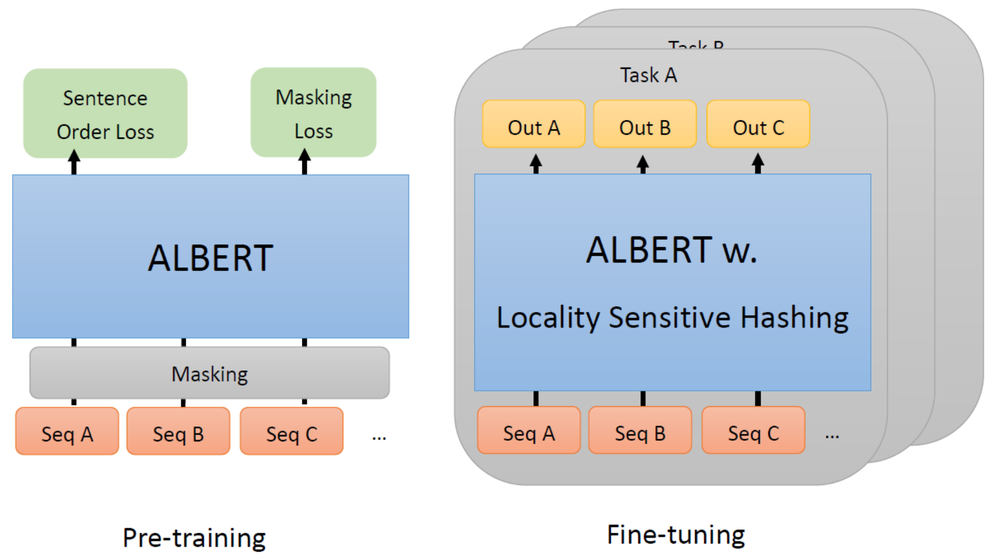
接著藉由多次旋轉可以降低將相似的q 分配到不同的分區之中。根據各個分區將各個 q 做排序並定義，只針對同一個分區內的q做注意力計算。

$$o_i = \sum_{j \in \widetilde{P_i}} \exp\left(q_i * k_j - m(j, P_i) - z(i, P_i)\right) v_j\,,$$

$$where\ m(j, P_i) = \begin{cases} \infty\ if\ \ j \notin P_i \\ 0\ otherwise \end{cases}$$

模型架設

我們使用Google在網路上公布之標準ALBERT預訓練模型。由於硬體的限制，在微調的步驟無法進行如預訓練般大量之訓練步驟，我們使用同樣12層之編碼器，隱藏層大小為768，嵌入大小128，序列長度為128至64，依照不同的工作及GPU能夠負荷之大小，變更 batch size。訓練流程如（圖三）。

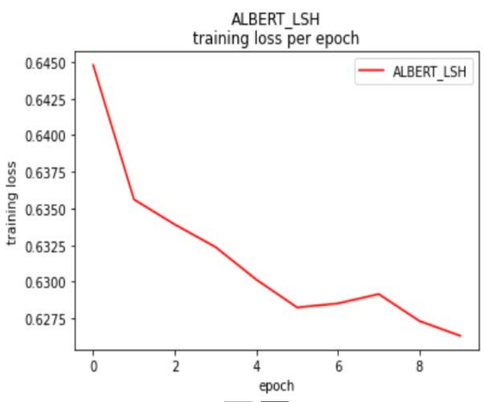


圖三
訓練流程

四、實驗結果

模型確認

在 GLUE 的 MRPC 數據集上進行多 epoch 的訓練，發現 Loss 有持續降低的趨勢，如圖四。因此可證明： Locality Sensitive Hashing 是可以訓練語言模型的。



圖四
Loss 隨 epoch 收斂

模型準確度

	MRPC	MNLI	SST-2	SQuAD1.1	SQuAD2.0
ALBERT_LSH	0.78	0.35	0.51	13.93	50.07
ALBERT	0.88	0.84	0.93	78.45	72.63

自注意力層因為使用LSH而造成的訊息丟失或許都是來自於關於語言的「細節」部分，而高度相似的文意訊息則會被保留，所以ALBERT_LSH在做有關於需要對語言的細節部分有高度可能會表現較差。

五、結論

Locality Sensitive Hashing在自然語言處裡工作上是可行的，並且可以減少模型運算所需空間，使訓練架構更龐大，是以訓練時間換取運算空間的方法。我們的ALBERT_LSH也就是減少了硬體使用的空間，但雜湊的注意力機制會丟失或簡化文字特徵使模型的準確率下降，但或許其需要更長的訓練時間，又抑或是此雜湊方法的極限就是如此，有待往後再進行研究。