數位語音處理概論 Reading Report

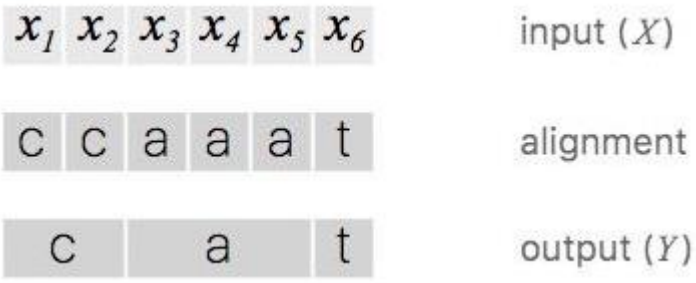
資工碩二 R06922134 葉沛陽

因為老師上課對於最近很紅的 deep learning 應用於語音的部分，只是簡單的帶過，並沒有詳細介紹，因此想要找一些最近一兩年的 paper 來閱讀，首先找到了 Google 的 State-of-the-art speech recognition with sequence-to-sequence models 以及 CMU、Facebook、Microsoft 合力發表的 Improved training for online end-to-end speech recognition systems 這兩篇 papers，但因為其中有些部分沒辦法看懂，而兩篇中都有提到 CTC(Connectionist Temporal Classification)這個技術，所以就找了 Hannun Awni 的 Sequence Modeling with CTC（Distill, 2017）這一篇 paper 來閱讀。以下是 Hannun Awni 的 Sequence Modeling with CTC（Distill, 2017）這一篇 paper 的內容。

首先，作者在 Introduction 提到。如果要將一段音頻轉成文字，但因為音頻跟文字的長度往往是不一樣的，所以有人提出第一個方法是" one character corresponds to ten inputs"，但是每個人的講話速度不一樣，這個方法其實是有很大的問題。第二個方法是"由人手動對齊音頻跟文字"，雖然這個方法能夠大大提高準確率，但是需要大量人力來完成，因此也不合乎實際。因此，作者指出 CTC（Connectionist Temporal Classification）這個正好能解決這個問題的方法。

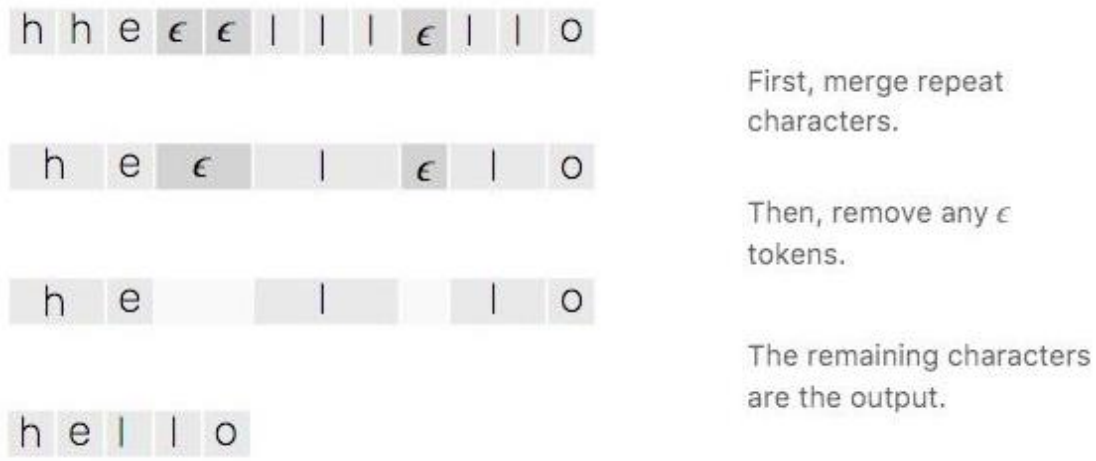
CTC 是一種改進的 RNN 模型，能夠讓機器學會對兩個序列自動對齊的神經網路，因此也能用在音頻與文字間的對齊。在音頻與文字轉換中，假設輸入(音頻)是 X=[x1, x2, …, xT]，輸出(文字)是 Y=[y1, y2, …, yU]，我們要找到一個準確率高的 mapping 來讓 X->Y。但現在有三個挑戰，第一個是 X,Y 的長度都是不固定的，第二個是 X 跟 Y 的長度比例也是不固定的，第三是我們沒有 X 跟 Y 準確地對齊答案。所以這篇 paper 也提到 CTC 如何來克服這些挑戰。

CTC 可以根據輸入 X mapping 到一些可能的 Y，而計算這些個別 Y 的機率的關鍵在於如何看待輸入和輸出之間的一致性。CTC 並沒有嚴格要求輸入與輸出完全的對齊。如果是一般的方法，例如下圖，X 長度有 6，如果分別預測出 c c a a a t 則將排在一起且重複的字母刪除而會得到 output Y 是 cat。



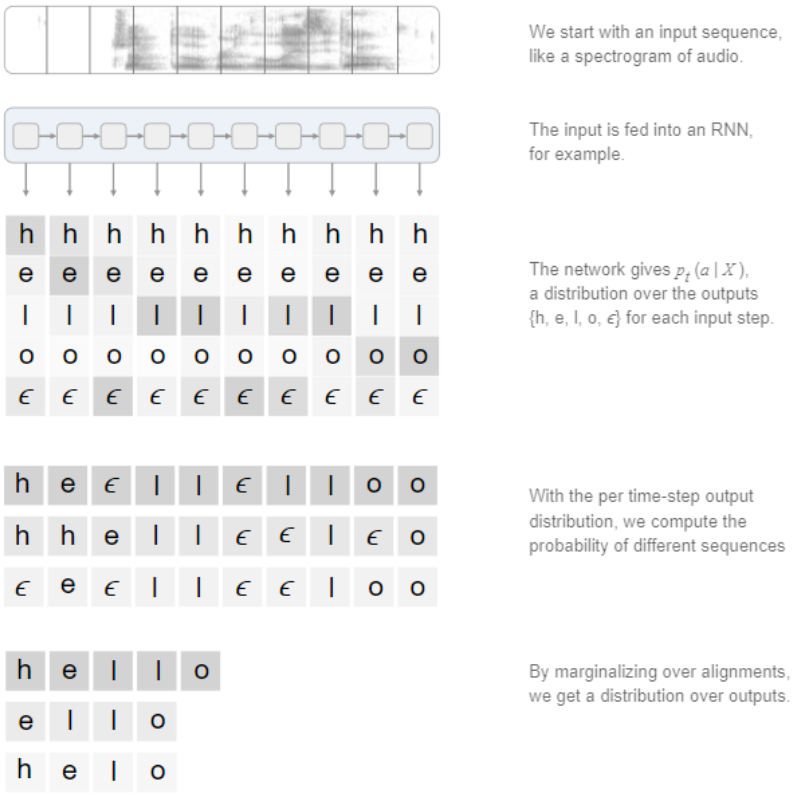
但這樣其實有很多問題，例如一段音頻中其實有一部份是 silent，但這個方法會 output 不出 silent 而硬給一個不對的字母；另一方面，如果 output 應該

是’hello’，因為會將排在一起且重複的字母刪除而會得到錯誤的 output ‘helo’。要解決這個問題，CTC 多加入了 blank 空白符號，空白符號不對應任何輸入且會在最後輸出時被刪除。如下圖:



每個 blank 之間會刪除排在一起且重複的字母，但因為 l 與 l 之間有 blank 所以他們不會被刪到只剩一個 l，因此最後結果就能成功 output 出’hello’。CTC 所 output 出的每一格稱為路徑，通常會有多個路徑並有相對應的機率，而這些路徑有三個性質，第一是這個路徑是單向的，如果到了下一格，前面的結果是不會再被改變的，第二是通常會有多個 X 對應到一個 Y，第三是輸出 Y 的長度不能大於輸入 X。

這些有機率的路徑，CTC 是利用 RNN 依照時間序列，來預測每一格的可能結果，因為 RNN 有考慮前面的音頻，所以效果很不錯，將每一格都 output 出多個含有機率的結果，最後將每一格都合併出多條可能的路徑且每條路徑都有相對應高低的機率。



$$p(Y \mid X) = \sum_{A \in \mathcal{A}_{X,Y}} \prod_{t=1}^{T} p_t(a_t \mid X)$$

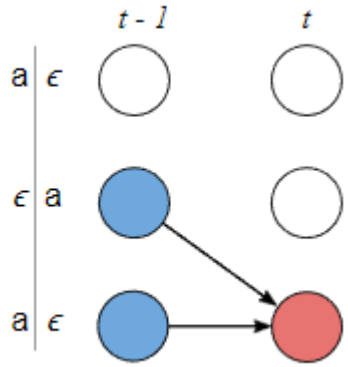
The CTC conditional **probability** | **marginalizes** over the set of valid alignments | computing the **probability** for a single alignment step-by-step.

根據上圖，機率為所有有效對齊的路徑的機率加總，而個別的路徑機率會是根據 X 每一格的 output 機率相乘。但因為通常路徑數量非常龐大，會造成很大的運算輛，所以為了減輕運算量，使用 Dynamic programming 來合併路徑，如果有兩種路徑可是使用相同格數產生出相同結果，就將它們合併。以下講述 Dynamic programming 如何來合併路徑。

假設 ϵ 代表 blank，Z = [ϵ, y1, ϵ, y2, ϵ, …, ϵ, yU, ϵ]，Z 為在 Y 的開頭結尾以及每個字母之間都加上 blank。$\alpha_{s,t}$ 代表 output 出 $Z_{1:s}$ 的分數 在 t 格的時候。

下列為兩種不同的可能 case:

Case1.

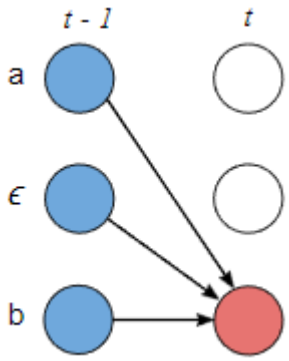


在這個 case 中，不能跳過 z_{s-1}，因為他很有可能是 Y 裡面的元素，如果 z_{s-1} 是 Y 的元素，代表 z_s 一定是 ϵ。另一可能是因為兩個重複的元素中間一定要有 ϵ，所以我們就知道 $z_s = z_{s-2}$ 。

$$\alpha_{s,t} = \underbrace{(\alpha_{s-1,t-1} + \alpha_{s,t-1})}_{\text{The CTC probability of the two valid subsequences after } t-1 \text{ input steps.}} \cdot \underbrace{p_t(z_s \mid X)}_{\text{The probability of the current character at input step } t.}$$

所以 $\alpha_{s,t}$ 由上面這個式子求出。

Case2.



假如 z_{s-1} 是介於兩個不同字母間的 ϵ，在這個情況下，可以跳過之前的 z_{s-1}，所以 $\alpha_{s,t}$ 由下面這個式子求出。

$$\alpha_{s,t} = \underbrace{(\alpha_{s-2,t-1} + \alpha_{s-1,t-1} + \alpha_{s,t-1})}_{\text{The CTC probability of the three valid subsequences after } t-1 \text{ input steps.}} \cdot \underbrace{p_t(z_s \mid X)}_{\text{The probability of the current character at input step } t.}$$

現在知道了 P(Y|X)，然後訓練時，為了能夠使用梯度下降法最小化 error，因此 loss function 使用下列的 negative log-likelihood。

$$\sum_{(X,Y)\in\mathcal{D}} -\log\ p(Y \mid X)$$

Inference 方面，訓練好模型後，我們就需要根據給定輸入計算可能的輸出，因此需要計算下面的式子。

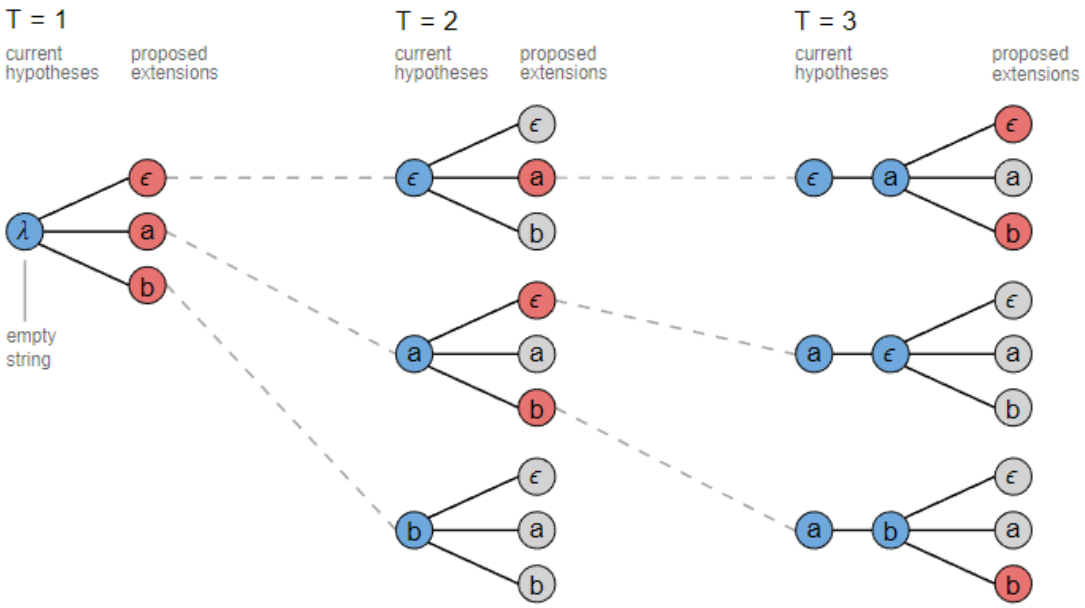
$$Y^* = \underset{Y}{\operatorname{argmax}}\ p(Y \mid X)$$

另一種 heuristic 方式為計算每一格中的最高機率的那個 output，式子如下

$$A^* = \underset{A}{\operatorname{argmax}} \prod_{t=1}^{T} p_t(a_t \mid X)$$

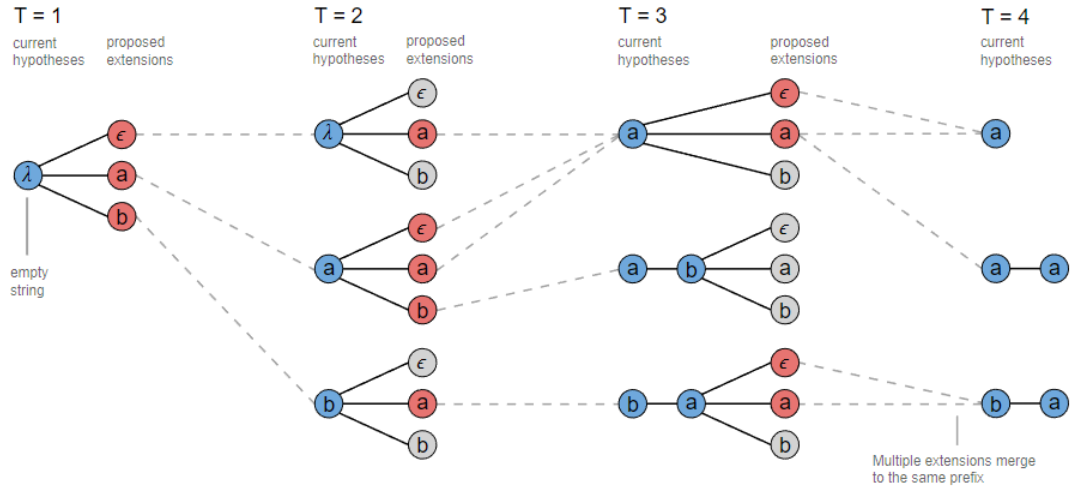
雖然這個方法在大部分情況都表現很好，但這個方法在特定情況可能會造成預測錯誤，例如: 假設輸出[a，a， ε]和[a，a，a]的概率比[b，b，b]低，但它們的概率之和高於後者。在這種情況下，heuristic 方法會給出 Y = [b]這樣的錯誤結論，因為它忽視了[a，a， ε]和[a，a，a]可以算是同一個輸出，而真正的答案應該是 Y = [a]。對於這個問題，這篇 paper 提出了一個改良過的 beam search 來為解決辦法。

Beam search 為一開始設定一個數為 beam size，然後每一個時間點會選出 beam size 個機率最高的可能，在依照這些可能預測它們到下一個時間點的可能再選出 beam size 個機率最高的可能，這樣一直下去。雖然 beam search 依然無法保證得到最佳解，但是可以依照調整 beam size 在計算量與一定程度的最佳解之間取得平衡。下圖為 beam size = 3 的 beam search 範例，紅色點為 top 3 的選擇。



A standard beam search algorithm with an alphabet of $\{\epsilon, a, b\}$ and a beam size of three.

而因為結果可能含有 blank ϵ，但是 blank 其實只是用來區隔字母而已，所以這篇 paper 將原本的 beam search 改良，刪除 blank ϵ，如下圖，可以看到藍色的點已經將 ϵ 去除了且如果是 a ϵ a 藍色點會是 a a 而 a a ϵ 藍色點會是 a。



The CTC beam search algorithm with an output alphabet $\{\epsilon, a, b\}$ and a beam size of three.

因此使用這個改良過的 beam search，在上述問題中就能知道[a，a， ε]和[a，a，a]一樣是[a]了，而不會認為 Y=[b]。

接下來 Paper 開始談到 CTC 的性質。雖然前面就有提到一些 CTC 的性質，不過在這個部分作者更加深入的探討 CTC 的性質。CTC 最廣為人知的缺點就是它的條件獨立假設(conditionally independent)，即每個輸出都跟其他輸出條件獨立，基本上在一般問題上，沒有問題，但如果今天有個 input 是"triple A"，一個可能的 output 應該要是"AAA"，所以"A"的機率應該要遠遠大於"r"，但是因為 CTC 的條件獨立假設，"r"機率可能會大於"A"。所以，

利用 CTC 建立的語音系統，不能根據前面的輸出來改善語言模型，也不能學習變成條件相依(conditionally dependent)的模型，它只能利用額外訓練的語言模型來提高準確率，這是 CTC 的很大的缺點。優點是，因為語言模型是使用額外訓練好的模型來使用，因此當場景轉換或是談話主題改變時，CTC 相較於使用自己的語言模型的系統，只要更改外接的語言模型，便能輕鬆轉換主題，提高準確率。

CTC 還有另外一個性質就是只允許單向對齊(monotonic alignments)，這在語音識別當中，大部分情況雖然可行，不過我覺得有些時候我們人聽，也是要聽到後面的話才能了解前面的意思，或是聽到後面才發覺自己誤會了前面的同音字到底正確的字是甚麼。因此，在這種需要考量後面的語句才能做出好的判斷的情況，CTC 是相對弱勢的。

最後，CTC 還有一個性質，就是輸出 Y 一定要比輸入的 X 長度短，這再大部分情況也許可行，不過，現實中還是存在著預測的 Y 就是比 X 長的例子。這篇 Paper 舉例，如果是字符"th"為輸出，但是它的正確的 input 卻是單一個音，這種情況就會造成，CTC 永遠無法輸出"th"，這也是 ctc 的一大缺點。

再來，這篇 paper 討論 CTC 與 HMM 之間的關係。乍看之下，會覺得 CTC 與 HMM 差異很大，不過其實本質上是很相似的。作者提到了解這兩者之間的關係是有助於我們把握 CTC 的優勢，並且能夠針對各種情況對 CTC 做調整。

下圖為 HMM 求 P(X)的算式。

$$p(X) = \sum_{A \in \mathcal{A}} \prod_{t=1}^{T} p(x_t \mid a_t) \cdot p(a_t \mid a_{t-1})$$

The probability of the input	Marginalizes over alignments	The emission probability	The transition probability

可以知道兩者呈正比關係

$$p(X) \propto \sum_{A \in \mathcal{A}} \prod_{t=1}^{T} p(x_t \mid a_t).$$

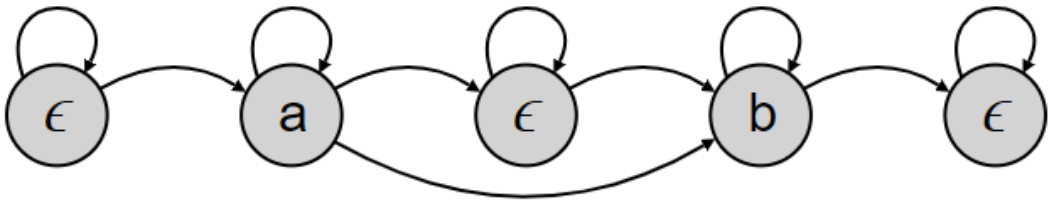
然後依照貝式定理得到下列正比關係。

$$p(X) \propto \sum_{A \in \mathcal{A}} \prod_{t=1}^{T} \frac{p(a_t \mid x_t)\, p(x_t)}{p(a_t)}$$

$$\propto \sum_{A \in \mathcal{A}} \prod_{t=1}^{T} \frac{p(a_t \mid x_t)}{p(a_t)}.$$

$$p(X) \propto \sum_{A \in \mathcal{A}} \prod_{t=1}^{T} p(a_t \mid X).$$

然後會發現最後一個式子其實就是 CTC 的 loss function。
因此，這篇作者提到將 blank ϵ 也加進 HMM 的 state 中，如下圖。



CTC HMM: The first two nodes are the starting states and the last two nodes are the final states.

在 CTC 算法中，我們有空白標記 ε ;而在 HMM 裡，我們有允許交換子集的左右模型.CTC 和 HMM 都有兩個起始狀態和兩個接受狀態。容易造成混淆的一個原因可能是在 HMM 中，模型會因不同的輸出 Y 而出現差異，事實上，這在語音識別領域是準確的，模型狀態圖的確會因輸出 Y 出現一定變化。轉錄過程中的觀察預測和機率函數是共享的。

了解完 CTC，我開始回來看 CMU、Facebook、Microsoft 合力發表的 Improved training for online end-to-end speech recognition systems 這篇 paper。這篇 paper 發表於 2018 年 8 月 30，也是一篇使用類神經網路應用於語音的 paper，而且才剛發表，相信可以看到在語音辨識界最新的技術是如何。

要提高端對端(end-to-end)語音辨識系統的準確度，需要在訓練之前慎選參數的初始值，否則模型很難達到 local minimum，特別是在 online network，例如像是單向的 LSTM。而目前最好的辦法是採用從 tied-triphone 系統 bootstrap 出訓練集，但是這樣很耗時且沒有很好的發音辭庫是做不到的。因此在這篇 paper，作者提出了一個初始參數的方法，主要使用 teacher-student learning 去 transfer 在大且訓練得很好的 offline end-to-end 語音辨識模型，這樣就不需要任何的辭庫或是語言資源，然後也在這基礎上加入了 curriculum learning 跟 label smoothing，這樣的方法實驗在 microsoft 的 Cortana 上，結果比起隨機初始的 baseline 系統，在 word error rate 上，改善了 19%。

接下來介紹這篇 paper 有提到的主要技術。

1. End-to-end speech recognition with character-based CTC

首先，CTC 技術允許神經網路可以學習如何將不定長度的 sequence 做標記，即使 input 跟 output 是還不知道怎麼對齊的。如同前面幾頁所講的，CTC 是會列出所有可能的對齊路徑下，選出最有可能的路徑當成 output。CTC 在計算路徑機率時，是假設每個 label 都是 independent，所以是採用各個 label 的機率相乘。以下是 CTC 的 loss function。

$$\mathcal{L}_{\text{CTC}} \triangleq -\ln P(\boldsymbol{y}|\boldsymbol{x}) \approx -\ln \sum_{\boldsymbol{\pi}\in\Phi} \prod_{t=1}^{T} P(k=\pi_t|\boldsymbol{x})$$

其中 y 是 output sequence，x 是 input sequence，π 代表是在所有可能的 label sequences Φ 中的其中一個 sequence，k 代表 K 個 graphemes 與一個 blank ϵ 共 K+1 個 symbols 其中一個。

CTC 使用 softmax 去定義 P(k|x)，而每個 RNN 的 hidden layer 會計算出一個 sequence $\boldsymbol{h}^l = (h_1^l, \cdots, h_T^l)$，因此 P(k|x)就是用下列式子求出:

$$P_t(k|\boldsymbol{x}) = \frac{exp(h_t^L(k))}{\sum_{i=1}^{K+1} exp(h_t^L(i))}$$

在 offline 方面，也是一樣使用這方法應用於雙向(bidirectional)的 RNN 中。

2. Teacher-student learning from offline to online models

Teacher-student learning 主要觀念是將資訊從大且深的已經訓練好很棒的模型(teacher)轉移給一個小且較淺的模型(student)。這個 student 的網路會被訓練到使自己的機率分布跟 teacher 的網路差異最小化。Teacher-student learning 已經成功地在聲音模型(acoustic model)、語音增強(speech enhancement)以及領域適應(domain adaptation)這些方面有好的結果。這個方法可以有效地預先訓練好一個需要很大運算資源、時間 offline 的 model，然後應用在一個真的要用的只要少量運算資源的 online model，就能以較少資源獲取更大的準確率。

在這篇 paper 裡，作者先建立一個 offline end-to-end 的 model 當成 teacher model，因為沒有 latency 的限制，所以使用 deep bidirectional 的 LSTM 去預測 output y 根據 input x。下列為雙向的 LSTM 簡易算式，雙向的意思即為一個從 t=0->T 開始往後 training，另一個從 t=T->0 往前 training :

$$\begin{aligned} h_t^l &= W_{fw}\overrightarrow{h}_t^l + W_{bw}\overleftarrow{h}_t^l + b \\ \overrightarrow{h}_t^{l+1} &= \text{LSTM}(h_t^l, \overrightarrow{h}_{t-1}^{l+1}) \\ \overleftarrow{h}_t^{l+1} &= \text{LSTM}(h_t^l, \overleftarrow{h}_{t+1}^{l+1}) \end{aligned}$$

訓練好強大的 teacher model 之後，作者就可以開始訓練一個 online 的單向(unidirectional)的 LSTM 來當作 student，因為 teacher model 已經含有雙向的資訊了，所以只要將 teacher model 轉移到 student model，student model 就能在 online(有限計算資源下)只使用單向(unidirectional)的 model 就好。關於 student 的網路要使自己的機率分布跟 teacher 的網路差異最小化，使用的是 Kullback-Leibler(KL) divergence 來評估 teacher model 跟 student model 的分布接不接近。假設 θBLSTM 代表 teacher model(BLSTM)中已經是最佳化的參數，而 $P_t(k|\boldsymbol{x};\theta_{\text{BLSTM}})$ 是 teacher model(BLSTM)的 output 在 step t 的機率分布。假設 $Q_t(k|\boldsymbol{x};\theta_{\text{LSTM}})$ 是 student model(LSTM) 的 output 在 step t 的機率分布。我們的目的就是要找出最好的 θLSTM 讓 KL divergence 最小，如下列算式:

$$D_{KL}(P_t \mid\mid Q_t) = \sum_{k=1}^{K+1} P_t(k|\boldsymbol{x};\theta_{\text{BLSTM}}) \ln \frac{P_t(k|\boldsymbol{x};\theta_{\text{BLSTM}})}{Q_t(k|\boldsymbol{x};\theta_{\text{LSTM}})} \tag{6}$$

$$= H(P_t, Q_t) - H(P_t) \tag{7}$$

可以忽略 H(Pt) 因為以 θLSTM 它的梯度為 0，所以最小化 P 與 Q 的 KL divergence 相當於最小化 P 與 Q 的 cross entropy H(Pt , Qt)。這樣就能求出最佳的 student model(LSTM)參數 θLSTM。

3. Curriculum learning

Curriculum learning，中文為課程學習，是一個用來改善訓練穩定度的方法。它如同課程一樣，一開始先給予簡單的工作，在逐漸地增加難度，直到原本的問題的難度。這個方法源自於觀察人類學習的方式，應用於語言模型(language modeling)、工作記憶(task memorization)以及語音辨識(speech recognition)等等領域都有更好的表現。

在這篇 paper，作者做了兩種不同的課程策略。第一個課程為，先訓練相較原本更短的 input 聲音(utterance)，因為比較短，所以有比較少可能的路徑，比較簡單去學，等到 model 學會這個比較簡單比較短的 utterance，在給予它原始長度的 utterance。第二個課程為，減少要預測的類別的數量，只有四個類別分別是元音(vowel)、輔音(consonant)、空白音(space)以及另一個空白音(blank)。等到這個四個類別的訓練完成，在給予原始的類別來訓練。

4. Label smoothing

Label smoothing 是一種藉由增加 noise labels 來改善泛化(generalization)的方法，以避免 overfitting。作者觀察到 blank symbol 占了 label 的分布很大一部份，所以增加了 regularization term 在 CTC 的 objective function，這個

regularizaton term 為模型預測的分布 P 與 labels 的均勻分布(uniform distribution)U 的 KL divergence，如下列算式:

$$\mathcal{L}(\theta_{\text{online}}) \triangleq (1-\alpha)\mathcal{L}_{\text{CTC}} + \alpha \sum_{t=1}^{T} D_{KL}(P_t||U)$$

在這篇 paper，α 設為 0.05。

接下來開始描述這篇 paper 的實驗部分。訓練集(training set)為大約 3400 小時的聲音(utterances)，驗證集(validation set)為大約 10 小時的聲音(utterances)，測試集(testing set)也為大約 10 小時的聲音(utterances)。 acoustic 輸入 features 方面，從每 10ms 從 25ms frames 的音頻(audio)中抽取 80 維的 log MFCC，接著使用 frame-skipping 技術將三張連續 frames 合併成 240(80*3)維的 log MFCC。模型方面 online LSTM-CTC 採用 5 層的 LSTM，每一層有 1024 cells，每一層會線性投影(linear projection)到 512 維的向量。Offline BLSTM-CTC 採用 5 層的雙向 LSTM，每一層有 1024 cells，每一層會線性投影(linear projection)到 512 維的向量。所有的 weight 隨機初始於均勻分布(uniform distribution)在[-0.05, 0.05]這個範圍之間，使用 SGD(stochastic gradient descent) 加上動量(momentum)。Learning rate 方面，初始設為 0.0001 如果 performance 沒有改善，就乘以 0.7。沒有使用任何辭庫(lexicon)或是語言模型(language model)。

結果方面，根據下方這個 Table (TS 代表 Teacher-student approach，CL 代表 Curriculum learning，LS 代表 Label smoothing regularization) :

Training Strategy	WER(%) w/o LM
LSTM + random initialization	41.0
LSTM + tied-triphone pre-training	30.8
LSTM + CL	37.8
LSTM + LS	35.4
LSTM + TS	36.0
LSTM + TS + LS	33.9
LSTM + TS + CL + LS	33.2
BLSTM	27.8
BLSTM + LS	25.5

可以看到 LSTM 加上隨機初始的 word error rate(WER)高達 41.0%，而 LSTM 加上 tied-triphone pre-training 的 WER 是 30.8%，相較之下改善了幾乎 25%，但是有使用語言資源(linguistic resources)。而這篇 paper 就是要不使用語言資源(linguistic resources)的情況下，盡可能的改善這樣的差距。

在不使用語言資源(linguistic resources)的情況下，可以看到 LSTM 加上 CL (Curriculum learning) WER 來到了 37.8%，相較於隨機初始(random initialization)，改善了 7.8%。LSTM 加上 LS (Label smoothing regularization) WER 來到了 35.4%，相較於隨機初始(random initialization)，改善了 13.7%。LSTM 加上 TS (Teacher-student approach) WER 來到了 36.0%，相較於隨機初始(random initialization)，改善了 12.2%。即使將 TS、CL 以及 LS 都加進去，WER 降低到了 33.2%，相較於隨機初始(random initialization)，改善了 19%，但還是輸給了 LSTM 加上 tied-triphone pre-training 的 WER 30.8%。而如果使用了 offline bidirectional LSTM 並且有無使用 LS，WER 分別大幅降低到了 27.8%及 25.5%，可以看到比 LSTM 加上 tied-triphone pre-training 的 WER 30.8% 好了。

這篇 paper 主要是提出了一個可以透過 teacher-student learning 方法使得線上(online) 端對端 (end-to-end) 語音辨識系統大幅提高準確率，作者將一個大、深且訓練得很好的的 offline BLSTM 轉移(transfer)它的知識(knowledge)到一個 online LSTM 模型，藉由最小化它們 labels 分布的 KL divergence。也因為這個方法不需要 boostrapping 從 tied-triphone 系統，它簡化了訓練的過程並且有利於某些語言資源不易取得的非主流語言的訓練。這個方法也很容易可以加上 CL(Curriculum learning)以及 LS(Label smoothing regularization)並且不輸給原本使用 tied-triphone pre-training 的效果。

看完這兩篇 paper，讓我學到了許多非 HMM 但是語音相關的技術，包括 CTC、end-to-end、Curriculum learning、Teacher-student approach、Label smoothing regularization、LSTM…等等。因為老師上課對於類神經網路方面只有概括提過，因此這兩篇 paper 讓我更深入了解如何將類神經網路應用於語音世界中，而因為修過數位語音處理概論這門課，讓我出乎意料的沒有很辛苦的就看懂了這些 papers，例如: HMM、log MFCC、language model、WER、lexicon 、KL divergence…等等許多語音相關的名詞。雖然如此，還是深刻感覺到語音處理這個領域還有很多未知的東西值得去探索。