

腎臟病檢測報告書

Author: Ethan Lai, Sam Wang

前言	P1
問題描述	P1
資料預處理	P2
名詞解釋	P4
研究配置	P6
研究結果與分析	P7
研究結論	P11
研究展望	P11
參考資料	P13

[Figure1] 如上圖所示，一張本應滿是資料的表格卻因嚴重的資料缺失問題看起來滿布空白；如何把這些空白填補起來，是我們即將著手進行處理的問題。

資料預處理：

為了解決資料缺失的問題，我們在訓練類神經網路前，特別針對資料進行了預處理 – 選用了3種不同的方法進行空值填補，具體作法如下：

(1) Normal Value + Gaussian Noise

我們首先計算同一數值所有人的平均(Mean)，藉由平均計算得到標準差(Variance)，創造一個平均為0，標準差為V的高斯雜訊。

再來我們取得醫生在臨床上用於判斷該數值是否為正常值的標準，以該正常值加上經由上述計算而得的高斯雜訊做為我們所要填補空白的值。

下圖為本預處理方法的虛擬碼(pseudo code)。

Algorithm 1:

Data: An array A , which includes 6864 patients' info and up to 600000 data
column: Person ID, B_CRE, B_K, etc.
row: each data.

Result: A complete array A without NULL value

```
1  $l \leftarrow$  length of array  $A$ ;  
2  $w \leftarrow$  width of array  $A$ ;  
3 Let  $Mean[1 \dots w + 1]$  and  $Variance[1 \dots w + 1]$  be new arrays;  
4 for  $i \leftarrow 1$  to  $l$  do  
5   |  $Mean \leftarrow Mean + A[i]/l$ ;  
6 end  
7 for  $i \leftarrow 1$  to  $l$  do  
8   |  $Variance \leftarrow Variance + power(A[i] - Mean, 2)/l$ ;  
9 end  
10 for  $i \leftarrow 1$  to  $l$  do  
11   | for  $j \leftarrow 1$  to  $w$  do  
12     |   | if  $A[i][j] == NULL$  then  
13       |   |   |  $A[i][j] \leftarrow Normal\_Value + Gaussian\_noise(mean = 0, variance = Variance[j])$ ;  
14     |   | end  
15   | end  
16 end
```

(2) Personal-wise Mean

我們於方法一計算平均與標準差時納入了所有人的數值做為考量，然而我們後來發現每個人都有屬於自己的獨特資料分布，不應與其他人混在一起做資料處理；因此我們嘗試了方法二，也就是Personal-wise Mean，在計算平均時我們便只選擇單一病人的資料。

若是該病人的資料空缺十分嚴重，導致有局部的資料沒有值可以取平均，我們便再依循方法一的規則填補正常值。

下圖為本預處理方法的虛擬碼(pseudo code)。

Algorithm 3:

Data: An array A , which includes 6864 patients' info and up to 600000 data
 column: Person ID, B_CRE, B_K, etc.
 row: each data.

Result: An complete array A without *NULL* value

```

1  $l \leftarrow$  length of array A;
2  $w \leftarrow$  width of array A;
3 for  $i \leftarrow 1$  to  $l$  do
4   for  $j \leftarrow 1$  to  $w$  do
5     while true do
6       if find the next values  $A[i+k][j] \neq NULL$  then
7         | break;
8       end
9     end
10    calculate slope;
11     $A[i][j] \leftarrow$  interpolation using slope;
12  end
13 end
```

(3) Interpolation

我們於上述兩種方法所使用的皆是取平均填值，然而卻忽略了那一筆筆的資料其實都有著時間的先後順序，也就是說我們必須將數值隨時間的趨勢變化用更好的方式表達出來；於是我們嘗試了第三種方法，也就是內插法，以期能夠更好的呈現數據的漲跌趨勢。

下圖為本預處理方法的虛擬碼(pseudo code)²。

Algorithm 2:

Data: An array A , which includes 6864 patients' info and up to 600000 data
 column: Person ID, B_CRE, B_K, etc.
 row: each data.

Result: An complete array A without *NULL* value

```

1  $l \leftarrow$  length of array A;
2  $w \leftarrow$  width of array A;
3 Let  $Mean[1\dots l+1]$  be a dictionary which points to array  $M[1\dots w+1]$  ;
4 for  $i \leftarrow 1$  to  $l$  do
5   if person ID of  $A[i] ==$  person ID of  $A[i-1]$  then
6     |  $Mean[personID] \leftarrow Mean[personID] + A[i] /$  (data number of that person);
7   end
8 end
9 for  $i \leftarrow 1$  to  $l$  do
10  for  $j \leftarrow 1$  to  $w$  do
11    if  $A[i][j] == NULL$  and  $Mean[personID][j] \neq NULL$  then
12      |  $A[i][j] \leftarrow Mean[personID][j];$ 
13    else
14      |  $A[i][j] \leftarrow Normal\_Value;$ 
15    end
16  end
17 end
```

(4) Moving Average Filter

在我們使用上述三種的資料預處理方式填補資料缺失之後，亦嘗試對資料進行移動平均(moving average)的處理；移動平均通常和具有時序性的資料一起使用，以消除短

² 內差法的實踐細節如下：當我們在同一個病人ID中遇到兩筆資料，則位在該兩個時間點中間的資料即用內差法補值，外面的則是用外差法；而當同一個病人ID中只有一筆資料，則我們就用該值作填補。

期的波動以及突出的長期趨勢或週期，達到穩定資料集的目的。具體來說，假設病人A在時間點3資料為A[3]，則我們便將其和先前第一、二個時間的資料A[1]、A[2]，和之後第四、五個時間的資料A[4]、A[5]做平均化處理，即

$$A[3] = (A[1] + A[2] + A[3] + A[4] + A[5])/5$$

我們將於實驗中比較以上三種資料處理的方式，以及分別加上移動平均觀察其結果的表現，以期能夠比較出一個表現最好的資料預處理方式。

名詞解釋：

1. 梯度下降法：

在訓練神經網路時，我們的目標訓練出一組參數，使得損失函數的值最小化。欲最小化一函數的值，直覺上的方法是求其微分。然而，實作上求損失函數的微分並不簡單，因此我們需採取梯度下降法(Gradient Descent)。梯度下降法是指每次欲更新模型參數時，沿著損失函數的梯度相反方向更新(一函數的梯度指向遞增最快的方向)。然而，這樣的方法只保證我們能找到局部最小值(local minimum)，因此我們需要更多優化的措施以幫助我們找到損失函數的最小值。

2. Learning Rate:

雖然我們知道要往梯度反方向更新，然而一次更新的值之大小還需要靠learning rate決定。對於神經網路的一個參數，更新公式為：

$$W_{t+1}^i = W_t^i - \eta \frac{\partial L}{\partial W_t^i}$$

其中， η 為 learning rate，L 為損失函數。

3. Adam / AdamW [6, 7]:

我們也可以利用優化器，在訓練時適時改變 Learning rate 以增進訓練效果。我們這次實驗使用 Adam [6] 和 AdamW [7] 這兩個優化器。Adam和 AdamW的好處在於可幫助我們增加訓練效率，他會計算更新的動量(momentum)，使過去用來更新的梯度也對未來的更新有所影響，避免單一梯度影響過大，進而達到抑制震盪、優化更新路徑的效果。AdamW則是 Adam加上正則化的版本，使得神經網路較不易過擬合。

Algorithm 1: Adam, our proposed algorithm for stochastic optimization. See section 2 for details, and for a slightly more efficient (but less clear) order of computation. g_t^2 indicates the elementwise square $g_t \odot g_t$. Good default settings for the tested machine learning problems are $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. All operations on vectors are element-wise. With β_1^t and β_2^t we denote β_1 and β_2 to the power t .

```

Require:  $\alpha$ : Stepsize
Require:  $\beta_1, \beta_2 \in [0, 1]$ : Exponential decay rates for the moment estimates
Require:  $f(\theta)$ : Stochastic objective function with parameters  $\theta$ 
Require:  $\theta_0$ : Initial parameter vector
 $m_0 \leftarrow 0$  (Initialize 1st moment vector)
 $v_0 \leftarrow 0$  (Initialize 2nd moment vector)
 $t \leftarrow 0$  (Initialize timestep)
while  $\theta_t$  not converged do
     $t \leftarrow t + 1$ 
     $g_t \leftarrow \nabla_{\theta} f_t(\theta_{t-1})$  (Get gradients w.r.t. stochastic objective at timestep  $t$ )
     $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$  (Update biased first moment estimate)
     $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$  (Update biased second raw moment estimate)
     $\hat{m}_t \leftarrow m_t / (1 - \beta_1^t)$  (Compute bias-corrected first moment estimate)
     $\hat{v}_t \leftarrow v_t / (1 - \beta_2^t)$  (Compute bias-corrected second raw moment estimate)
     $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$  (Update parameters)
end while
return  $\theta_t$  (Resulting parameters)

```

4. Learning Rate Scheduling:

在訓練後期，神經網路會趨向收斂。這時，我們需要適時降低 learning rate，避免過大的 learning rate 使神經網路持續震盪，無法利用梯度下降至最低點。

5. Batch Normalization:

Batch Normalization 是將一個batch的數據在每進入一個全連接層時便將資料標準化成平均為0，標準差為1；如此一來可以將分散的數據統一，有助於減緩梯度消失以及解決Internal Covariate Shift² 問題，並加速收斂。

Batch normalization 之所以可以減緩梯度消失，是因為他將數值分布移至0附近，避免經過sigmoid function時因為數值過大而達到飽和區，導致梯度消失。

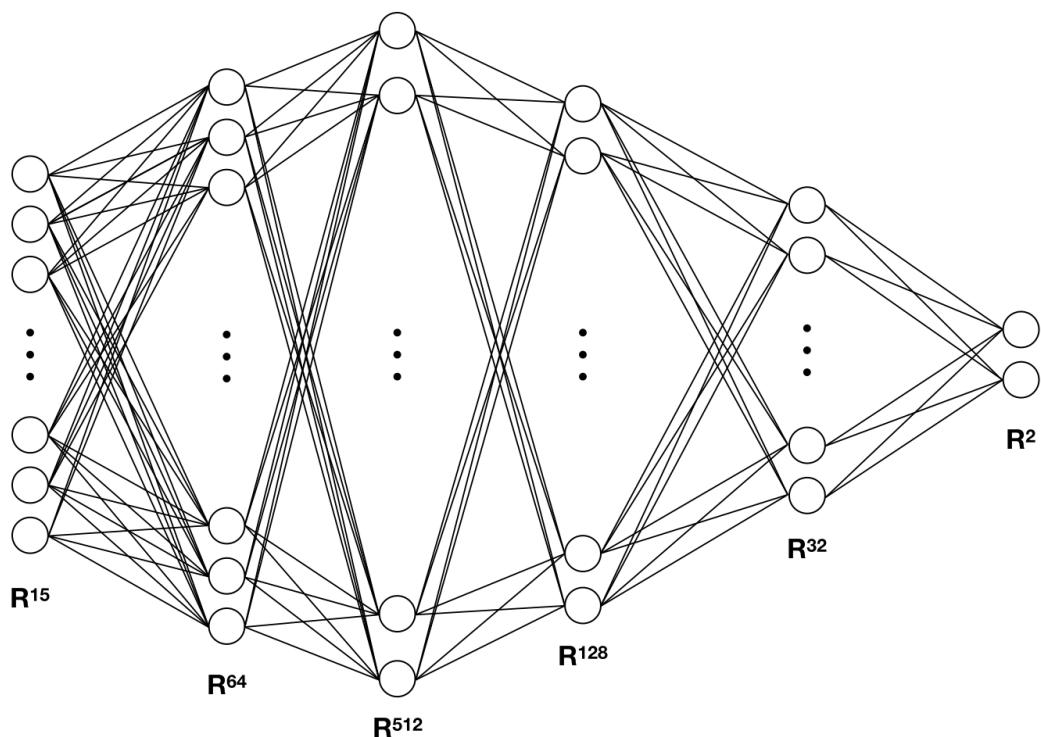
²所謂Internal covariate shift指的是神經網路各層數據分布不一致的問題，導致網路要不斷學習新分布，最後卻無法調整出一個統一的分布。藉由batch normalization，統一各層的特徵維度，讓數值更穩定，如此一來便可以使用較高的更新率優化，藉此提高模型的學習速度。

研究配置：

1. 神經網路模型

這次資料的特性為一個病患搭配一個時間點的生理訊號。在進行分類時，適合使用多層感知器(Multilayer Perceptron)的架構訓練。多層感知器是一種前向傳遞類神經網路，為層狀的結構。最前和最後分別為輸入及輸出層，中間的數層統一稱作隱藏層。從輸入層輸入生理訊號向量，多層感知器經由隱藏層，最終會將此生理訊號向量映射至輸出層，即為我們分類的結果。而讓多層感知器學習的方法是，其每讀過一批資料，我們會對其預測的成果與實際成果計算一損失函數，並利用反向傳播演算法(Backpropagation)，使我們的多層感知器得以學習(即更新參數)。

下圖[Figure2] 為本次實驗所使用的模型示意圖，我們的全連階層一共5層 – 從輸入層的15維，逐漸擴張到第二層的64維，以及第三層的512維，再降維至128和32維，最後輸出我們的結果。



[Figure2]

2. 研究方法

(1) 實驗一、在相同模型架構下，我們比較三種資料預處理方式對於模型分辨腎臟病能力的影響

(2) 實驗二、我們選出實驗一裡表現最好的資料預處理方式，並在相同模型架構下，觀察添加moving average filter之後模型對於腎臟病分辨的能力變化

(3) 實驗三、在我們經由實驗一和實驗二得到最好的資料預處理方式之後，我們比較不同的模型架構對於腎臟病的分析準確度，並挑選出表現最好的一個。

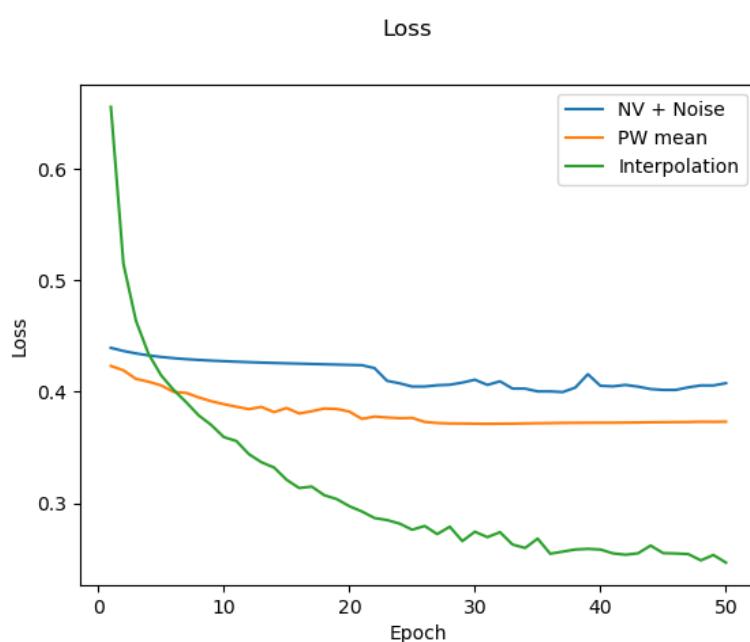
3. 訓練配置

損失函數 (loss function)	優化器 (optimizer)	初始學習率 (initial learning rate)	期數 (epoch number)	學習率規劃 (learning rate scheduling)
Cross Entropy Loss	Adam	0.001	50	30th epoch: x 0.01 70th epoch: x 0.01

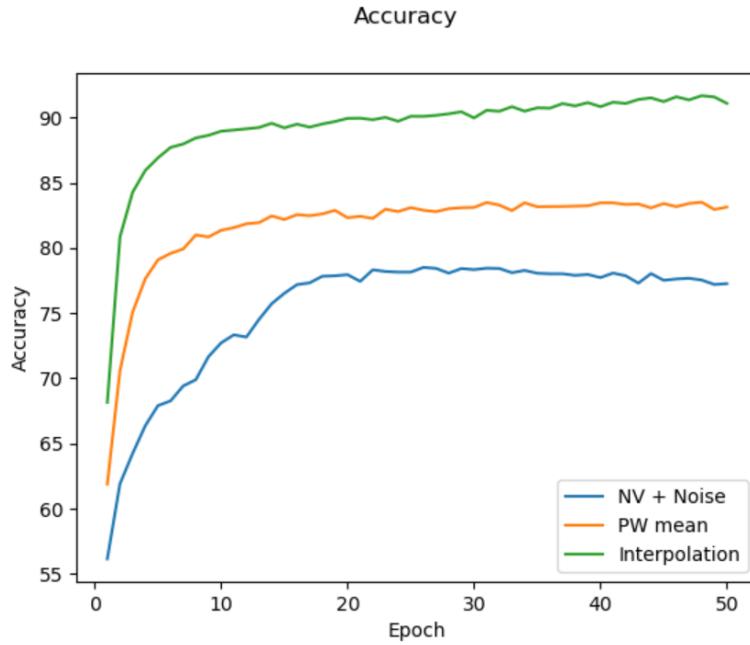
結果呈現與分析：

1. 實驗一

實驗一我們研究在相同模型架構下，三種不同的資料預處理方式對於模型分辦腎臟病能力的影響；我們將數據分別生成出來之後，用模型訓練三十期(epoch)，紀錄他們的損失函數值(loss)和預測準確度(validation accuracy)，實驗結果如下圖所示Figure3, 4]：



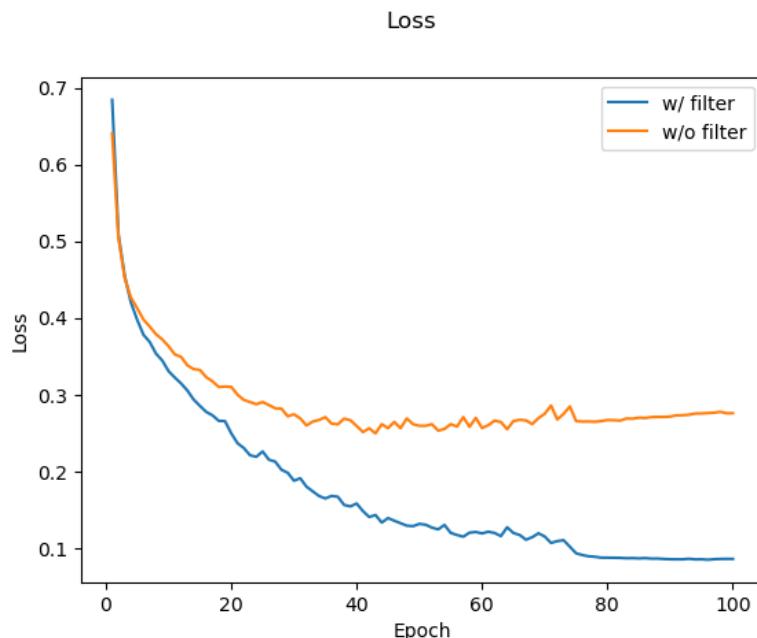
[Figure3] 上圖顯示模型在三種預處理過後的資料上的Loss-Epoch趨勢圖，其中 NV+Noise(Normal Value + Gaussian Noise)相較PW mean(Personal-Wise mean)、Interpolation表現亮眼許多，有著較低的損失函數值(Loss)，意味著該模型預測較為準確。



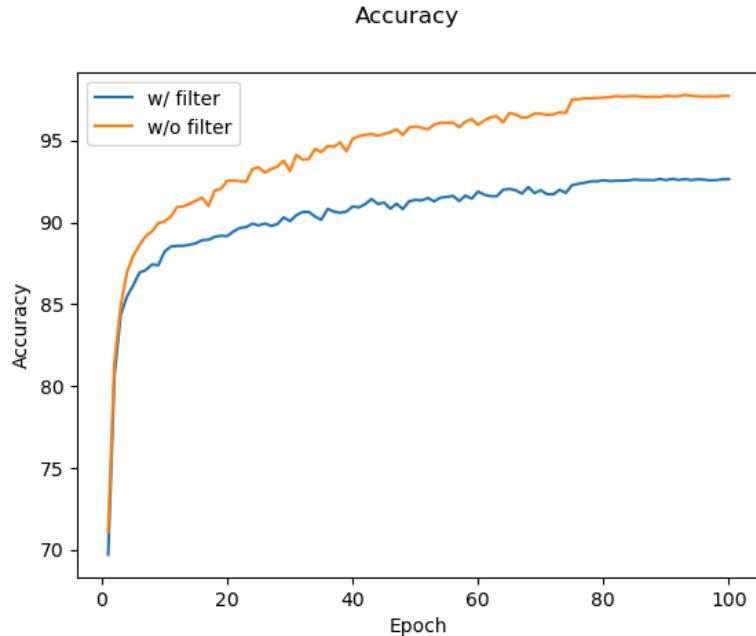
[Figure4] 上圖顯示模型在三種預處理過後的資料上的Accuracy-Epoch趨勢圖，可以發現隨著Epoch的遞增，內插法(Interpolation)在三者中的預測準確度亦為最高，且相較其餘兩種方法可以提升高達10~15%的準確度；因此在往後的實驗二及實驗三中，我們將會內差法作為我們的資料處理方式。

2. 實驗二

實驗二我們選出實驗一裡表現最好的資料預處理方式(也就是內插法)，並在相同模型架構下，觀察添加moving average filter之後模型對於腎臟病分辨的能力變化。實驗結果如下圖所示 [Figure5, 6] :



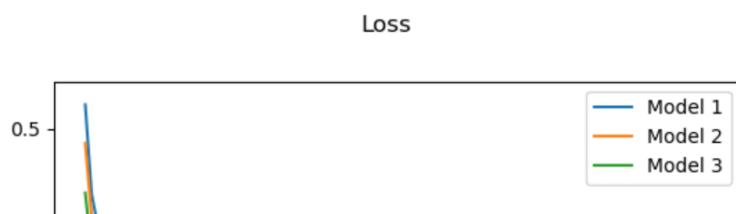
[Figure5] 我們從圖中可以得知，當我們使用內插法+moving average filter時(w/ filter)相較沒有使用時(w/o filter)有著比較快的模型收斂速度，且當兩者收斂達到飽和時，前者有著明顯較低的損失函數值，由此可知添加了移動平均的計算之後可以提升模型的訓練準確度。



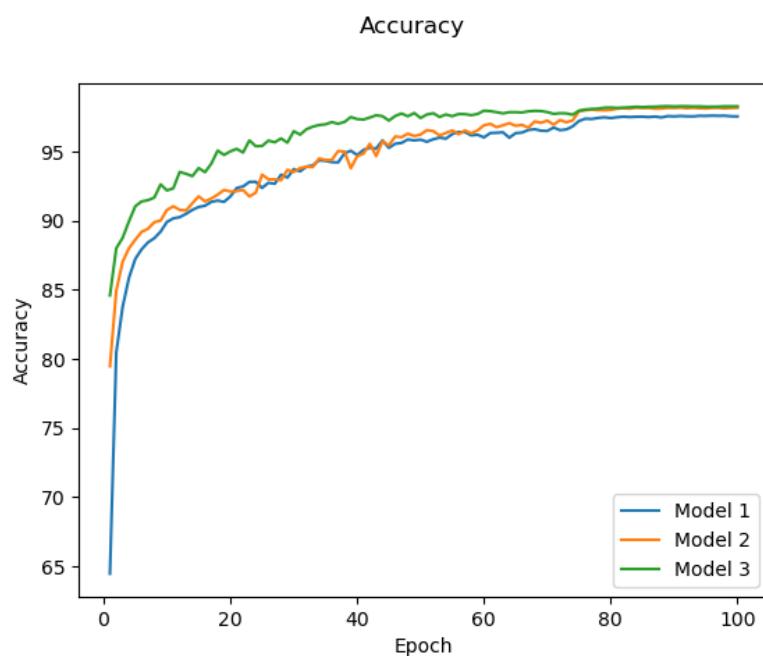
[Figure6] 我們從圖中可以得知，當我們使用內插法+moving average filter時(w/ filter)相較沒有使用時(w/o filter)有著比較高的預測準確度，且在第100期(Epoch)時添加了moving average filter可以將準確度從92%提升至98%，效果顯著。

3. 實驗三

在我們經由實驗一和實驗二得到最好的資料預處理方式之後，我們比較不同的模型架構對於腎臟病的分析準確度，並挑選出表現最好的一個；如下圖[Figure7, 8]所示，Model1的模型大小為三者之中最小的，Model3最大；根據實驗結果，我們發現大模型雖然對於提升預測準確度來說效果不大，卻可以加速模型收斂，達到快速訓練的效果。



[Figure 7] 從上圖我們可以觀察到Model3相較於Model1,2有著較快的loss下降速率，顯示了較大模型的快速收斂能力。



[Figure 8] 從上圖我們可以觀察到Model3在大約第五期(epoch)時便達到了90%的預測準確度，相較於Model2的8 epoch和Model3的10 epoch, Model3收斂得更快。

研究結論：

本次實驗在給定病患的 15 個生理特徵的前提下，進行病患是否在兩年內需要洗腎的預測。我們在面臨資料缺漏比例較大的情況，使用 3 種不同的資料處理方案將訓練資料進行預處理，並使用深度學習的多層感知器模型進行預測，最終根據實驗一的結果，在內插法的預處理上得到超過 98% 的準確度。實驗二、三中可發現，使用 Moving Average Filter 將資料做平滑化的預處理，搭配較深、參數較多的多層感知器，可以得到最好的結果。

除此之外，我們也發現多層感知器在預測時間精度較精細的預測（例如：預測確切病患洗腎之年份），準確度會明顯下降。我們推測這是因為多層感知器無法觀察病人隨時間的變化，因而漏掉許多和隨時間變化的特徵。為因應這個問題，我們之後的目標是訓練循環神經網路，因為他擅長處理時間序列的特性可以幫助我們獲得更多病患的特徵，並預期可以在得到更精準預測的同時維持一定的正確率。

研究展望：

1. LSTM is All You Need

本次實驗，我們將病患每次到院檢查當作一筆資料，並希望能預測病患未來洗腎的時間。然而，有時較難從一次檢驗的生理訊號數據觀察出特徵。若能觀察一段時間該生理訊號的變化，模型就有機會依據各生理訊號的變化之趨勢掌握更多該病患的生理狀況。不過，若要使模型學習一段時間內生理訊號的變化，在我們參考諸多論文著作後 [1, 2, 3, 4, 5]，使用循環神經網路(RNN, Recurrent Neural Network)會較多層感知器適合。

舉例來說，傳統的神經網路(像是本次實驗所使用到的MLP)常常假設模型輸入是彼此獨立的，然而這對於某些應用來說並不可行。循環神經網路的特色在於可以接受輸入有時間序列結構。這是因為循環神經網路會將前一時間的狀態記錄起來，在下一個輸入進來時，會根據記錄的狀態來進行預測，因此循環神經網路適合用來模擬病人生理訊號這類會隨時間變化的輸入。

然而，和典型的神經網路類似，RNN也會有梯度消失與梯度爆炸的問題，且該問題在越長的時間序列輸入上特別明顯。於是我們衍伸出了一種RNN的變形，對RNN的隱藏層做了改進，解決了其長期依賴的問題，此即為LSTM (Long Short-Term Memory)。

與RNN不同的是，LSTM利用閘門保存與捨棄來自上一個時間區間的訊息，如下列式子所示：

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

其中 h_{t-1} 為上一個時間的隱藏層， x_t 為該時間點的輸入，兩者經過線性變換之後經過activate function後輸出 f_t 。

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \quad (4)$$

我們再利用上述三條式子決定在該時間點有多少資訊要被保留。(4)決定了我們最終保留的資訊為前一個時間點 C_{t-1} 和本時間點 C_t 的線性組合。

$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

最後，我們決定有多少資料要被輸出。將我們所保留的資訊經過激勵函數的調控之後，即為該時間點t的單位輸出，以及傳送到下一個時間點t+1的資訊。

2. 實驗配置說明

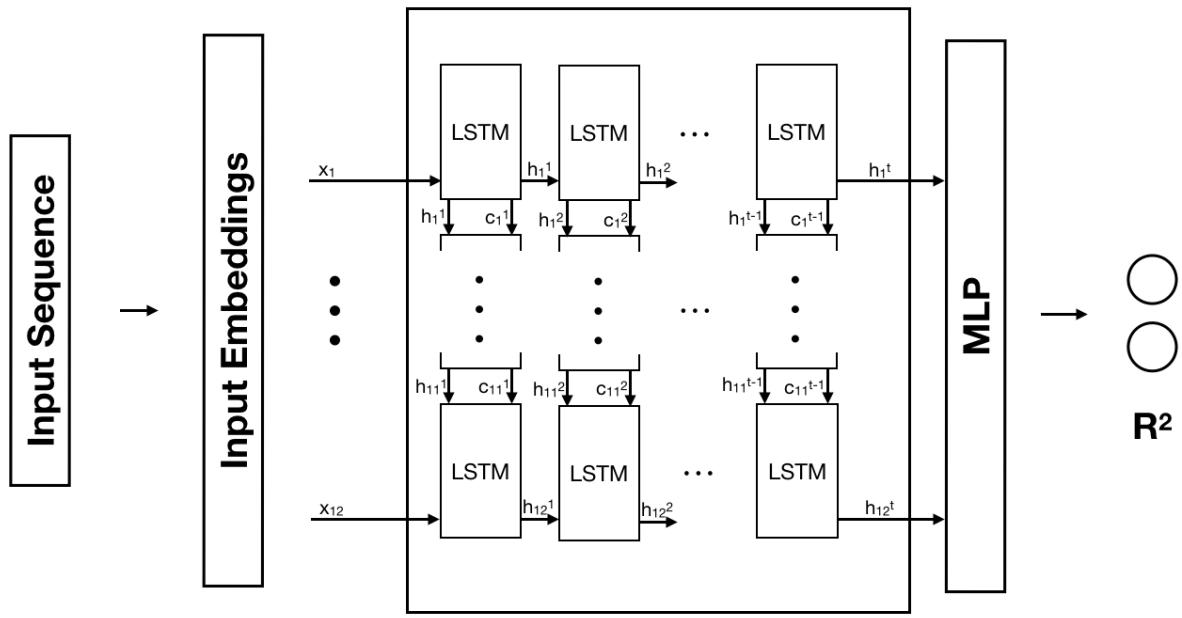
如上所述，RNN網路適合處理序列問題。就本次實驗而言，只要將輸入改為一病人一段時間的生理訊號序列(例如：為期一年，蒐集一病人每月檢查的數據，我們即可得到長度為12的輸入序列)，便可利用RNN的特性進行分類。

首先，我們預計輸入為一病人一年內每個月的檢查數據，即是長度為12的序列(以下均稱呼其為輸入序列)。此序列每個元素都是該病人當次檢查的15項生理指標。在將序列輸入LSTM之前，我們會對其進行線性轉換以轉換為對模型來說辨識度較好的表示，緊接著便將其輸入LSTM網路中，最後再將預測的序列接上東層感知器，即可預測機率分佈。上面過程可以算式表示：

$$e_i^t = \phi(x_i^t, W_e)$$

$$h_i^t = LSTM(h_i^{t-1}, e_i^t, W_l)$$

其中， x_i^t 為病人i在時間t的生理訊號之向量、 W_e 為線性轉換之參數、 ϕ 為神經網路的激發函數、 h_i^{t-1} 為LSTM前一隱藏層之輸出、 W_l 為LSTM的參數
整體神經網路模型可見下圖 [figure 8]:



[figure 8]

Input Sequence指的是病人的輸入序列、Input Embeddings指的是線性轉換後的輸入序列、MLP是多層感知器、上標為時間、下標為序列的位置

參考資料：

- [1] Tan, Qingxiong et al. “DATA-GRU: Dual-Attention Time-Aware Gated Recurrent Unit for Irregular Multivariate Time Series”. *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 01, Apr. 2020, pp. 930-7, doi:10.1609/aaai.v34i01.5440.
- [2] Nguyen, Minh et al. “Predicting Alzheimer's disease progression using deep recurrent neural networks”. *NeuroImage*, vol. 222, Nov. 2020.
- [3] Chen, David et al. “Early Detection of Post-Surgical Complications using Time-series Electronic Health Records”. *AMIA Jt Summits Transl Sci Proc*, May 2021, pp. 152-160.
- [4] Choi, Edward et al. “Using recurrent neural network models for early detection of heart failure onset.” *Journal of the American Medical Informatics Association : JAMIA*, vol. 24(2), Mar. 2017, pp. 361-370, doi:10.1093/jamia/ocw112.
- [5] Tangri, Navdeep et al. “A predictive model for progression of chronic kidney disease to kidney failure”. *JAMA*, vlo. 305(15). Apr. 2011, pp. 1553-9, doi: 10.1001/jama.2011.451.
- [6] Kingma, Diederik P. et al. “Adam: A Method for Stochastic Optimization”. *arXiv preprint arXiv:1412.6980*, 2014.
- [7] Loshchilov, Ilya et al. “Decoupled Weight Decay Regularization”. *arXiv preprint arXiv:1711.05101*, 2017.
- [8] Ronneberger, Olaf et al. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. *arXiv preprint arXiv:1505.04597*, 2017.