109-1 EE5193 雲端計算與資安

Term Project: Quick Draw, but only Digits

(github url: https://github.com/kai860115/Quick-Draw-but-only-Digits.git)

(project website: https://cccs-deploy-jpabvzhbuq-df.a.run.app/)(網站將於1/17失效)

組員: r09921057 林士平、r09921054凌于凱、r09921062 游子慶

1. abstract

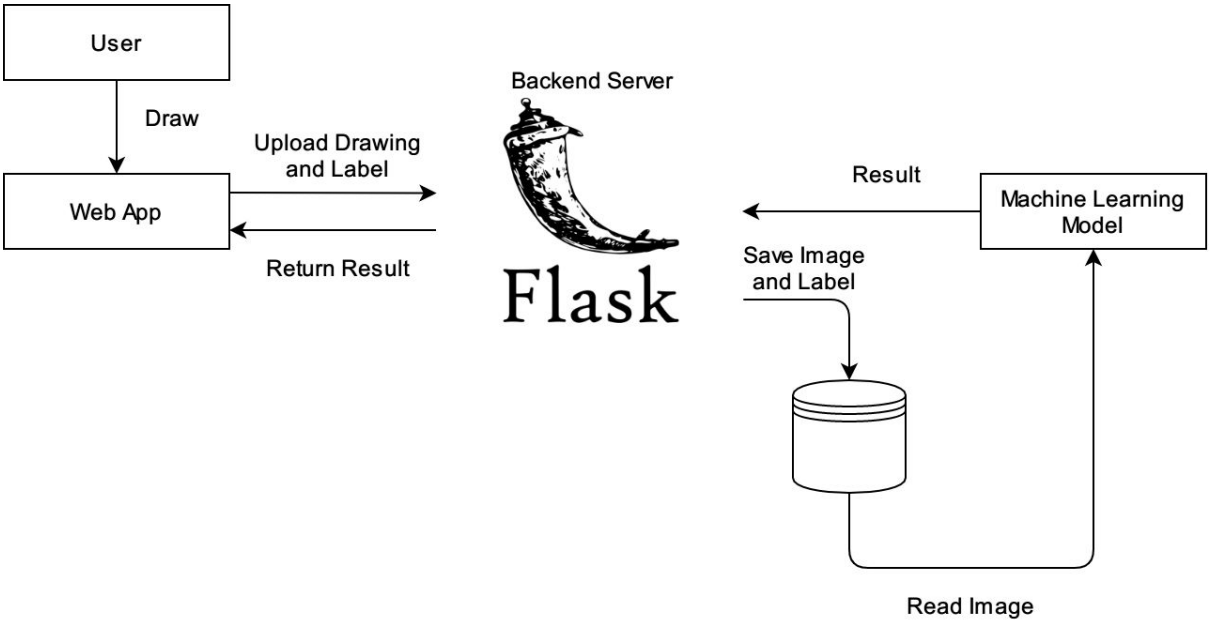
機器學習演算法需要使用大量資料訓練，但標記資料要耗費相當多的人力資源以及時間。現在很多大公司像是 Google 會利用一些有趣的方法蒐集資料，例如：Quick Draw 這個網站就是一個例子，它讓使用者畫出給定的物品，並且讓使用者在畫的過程可以知道預測的結果。這樣不僅可以提供使用者趣味，Google也可以藉此得到更多標註後的訓練資料，更能進一步做像semi supervised learning等訓練分析。

在這個term project中，我們創造了一個有趣的機器學習互動網站。在此我們將焦點放在手寫數字資料的預測與收集上，用戶可以手寫數字，按下送出便能得知機器學習預測結果，並且使用者能順便告訴開發者預測正確與否然後回傳正確的數字為何，這樣標註後的手寫數字圖片就能傳到資料庫，供開發者做後續的資料分析。

這會是一個對使用者和開發者都有利的網站，也能支援手機版，如此能更切合「手寫數字」這個命題。此外我們亦有佈署這個網站，透過以下網址(將於1/17失效)：https://cccs-deploy-jpabvzhbuq-df.a.run.app/，讓使用者能隨時隨地進入網站。

2. approach

系統的操作為：I. 使用者對伺服器送出要求，伺服器回傳前端介面，II. 使用者在繪圖區寫上數字，發送給後端伺服器，後端會回傳機器學習模型做出預測，III. 預測會顯示在介面上，使用者也可以再傳入正確的答案，後端則會將這筆資料收集到資料庫中，整體系統的架構如下圖所示。



I. Frontend

前端由HTML、CSS、JavaScript撰寫，搭配Bootstrap做畫面配置以及裝飾，並且因為Bootstrap的元素例如card, jumbotron等有使用media query做響應式網站設計(responsive web design)，讓我們的網站能在手機上面使用而不會跑版，除此之外我們有使用font awesome圖示外掛來讓前端畫面更豐富一些。連接後端的部分，我們利用JavaScript原生的Fetch函式對後端伺服器傳送HTTP要求。

II. Backend

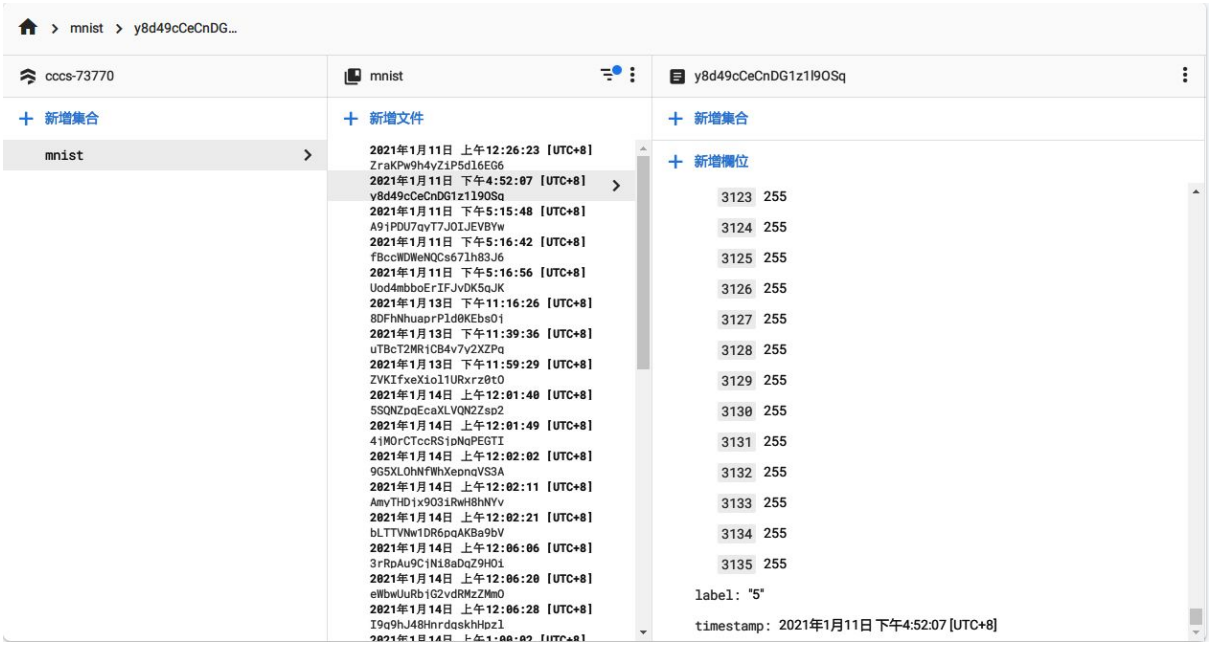
A. Server

後端的伺服器是利用Flask架設，傳送前端頁面給使用者，當使用者傳送圖片時，呼叫機器學習模型做出預測並回傳結果，當使用者傳送正確答案時，伺服器會將該筆資料加上時間戳記上傳至資料庫。

B. Database

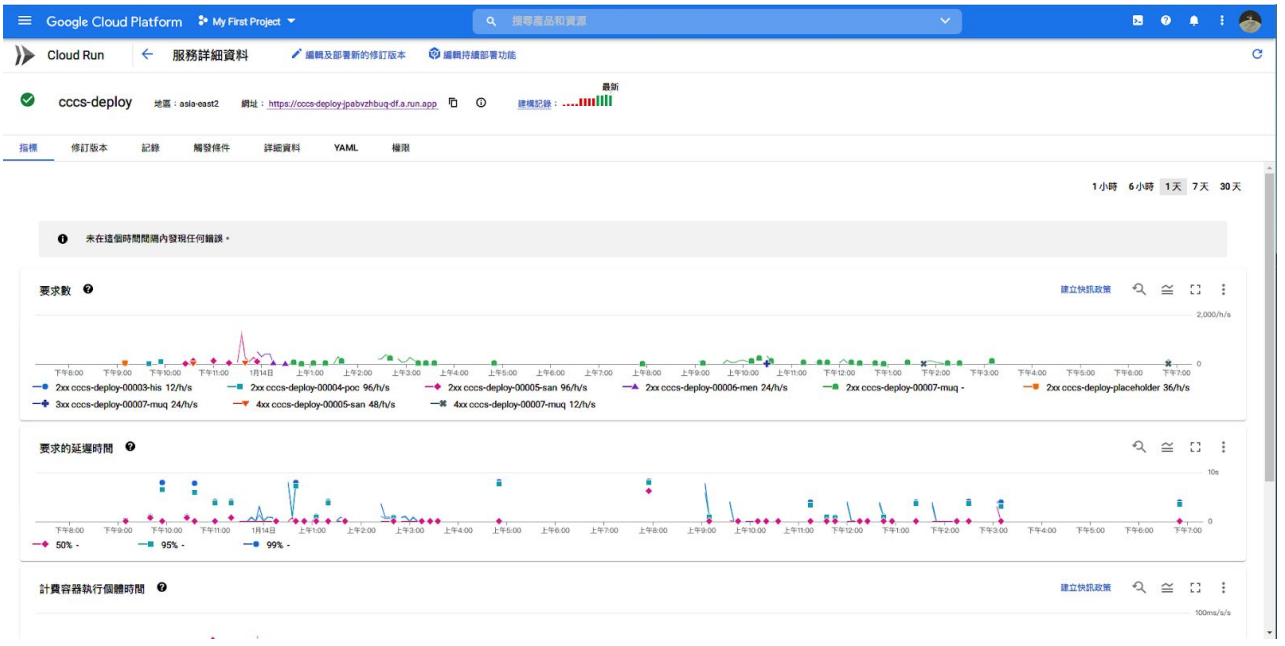
資料庫是利用Google Firebase中的Cloud Firestore服務，而因為Firestore屬於NoSQL的資料庫，所以也就不會有SQL Injection的問題，要

存取資料庫中的資料都需要利用私鑰認證，下圖為存放在Firestore的資料集。



C. Deployment

整個系統的部屬是利用Cloud Run，如此我們不需要太專注於細部管理，Google Cloud Platform會分配一個域名給這個應用程式，也會根據使用情況自動管理伺服器與調整資源分配，而且當GitHub的repo更新時會觸發Cloud Run重新建置Docker image，接著將最新的版本跑起來，下圖為Cloud Run的管理數據。



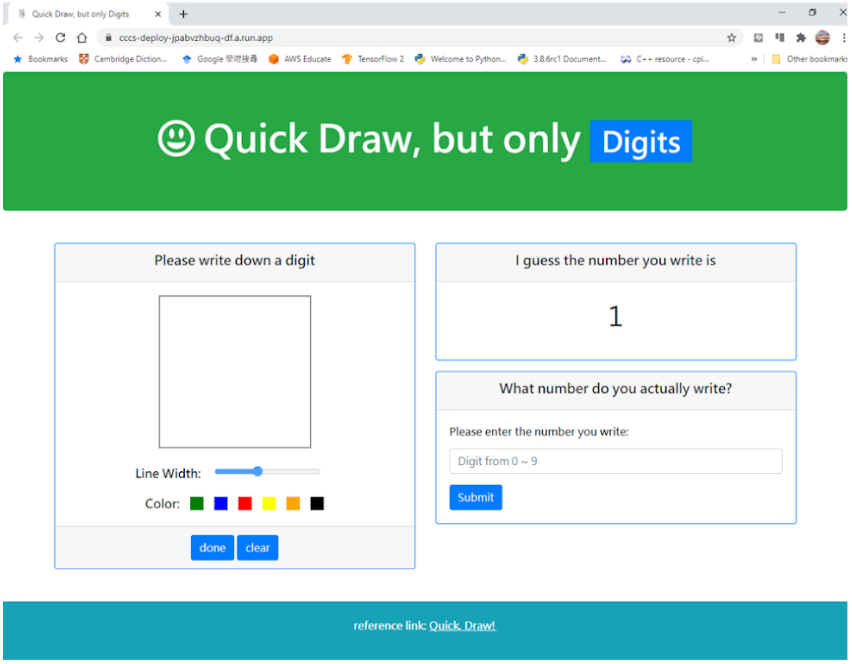
III. Machine Learning

使用的框架為PyTorch，對Mnist手寫數字圖片資料來做訓練，建立 Convolutional Neural Network 模型，透用 Cross Entropy Loss 去做 gradient descent；因為Mnist為黑背景白字，造成直接用沒處理過資料去做訓練對於我們問題的目標不太好，因此有先將圖片轉成白背景黑字；此外，使用者可能會在邊邊位置寫字，或是寫字角度不同，因此我們使用 random rotate 以及 random resize crop 來新增資料，對於我們的結果很有幫助。

3. results

I. 初始畫面

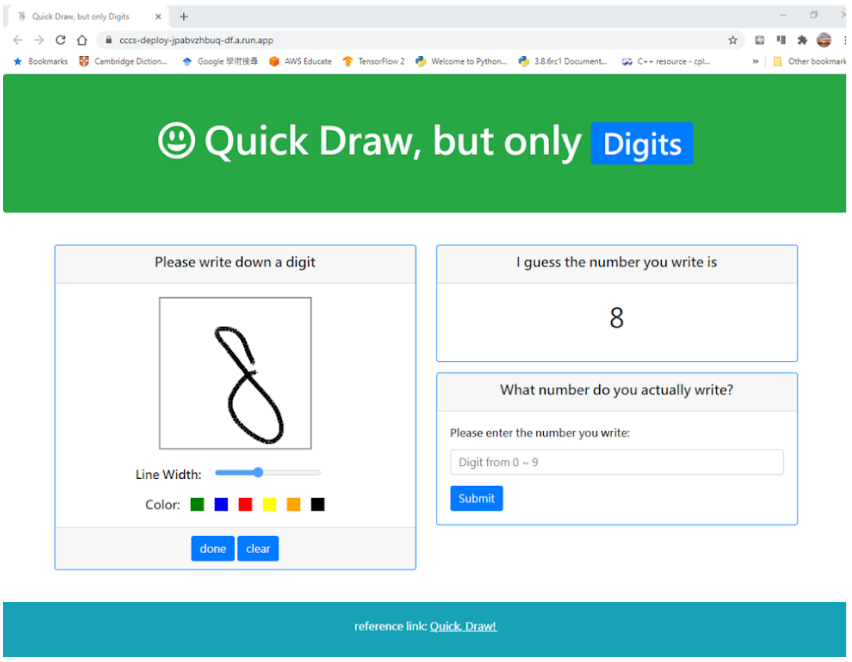
電腦版



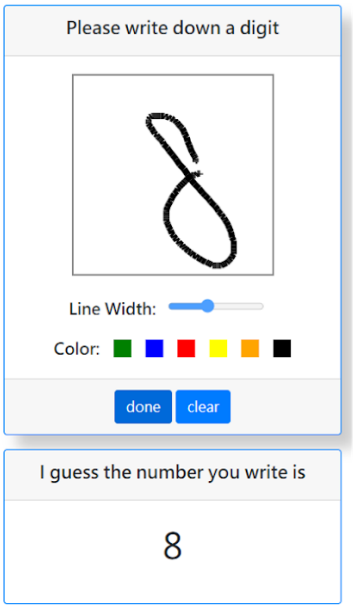
手機版

II. 使用者手寫數字並按下done後

電腦版

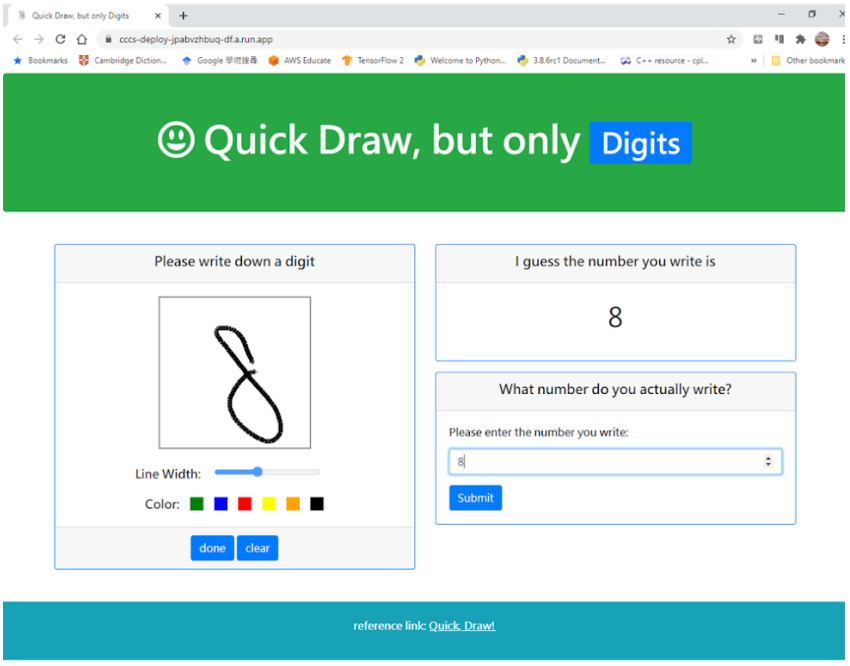


手機版

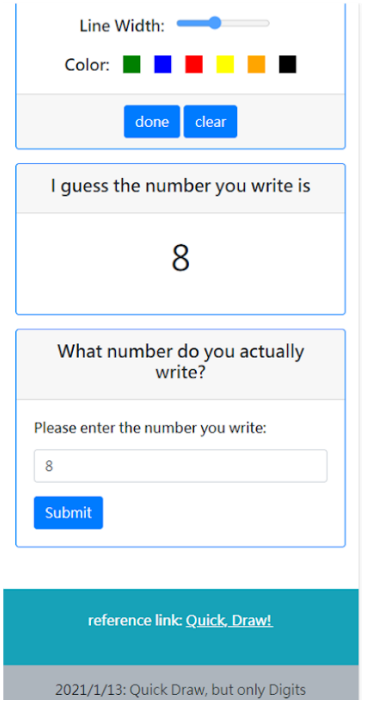


III. 使用者輸入他實際寫下的數字在右下角"What number do you actually write?"框中

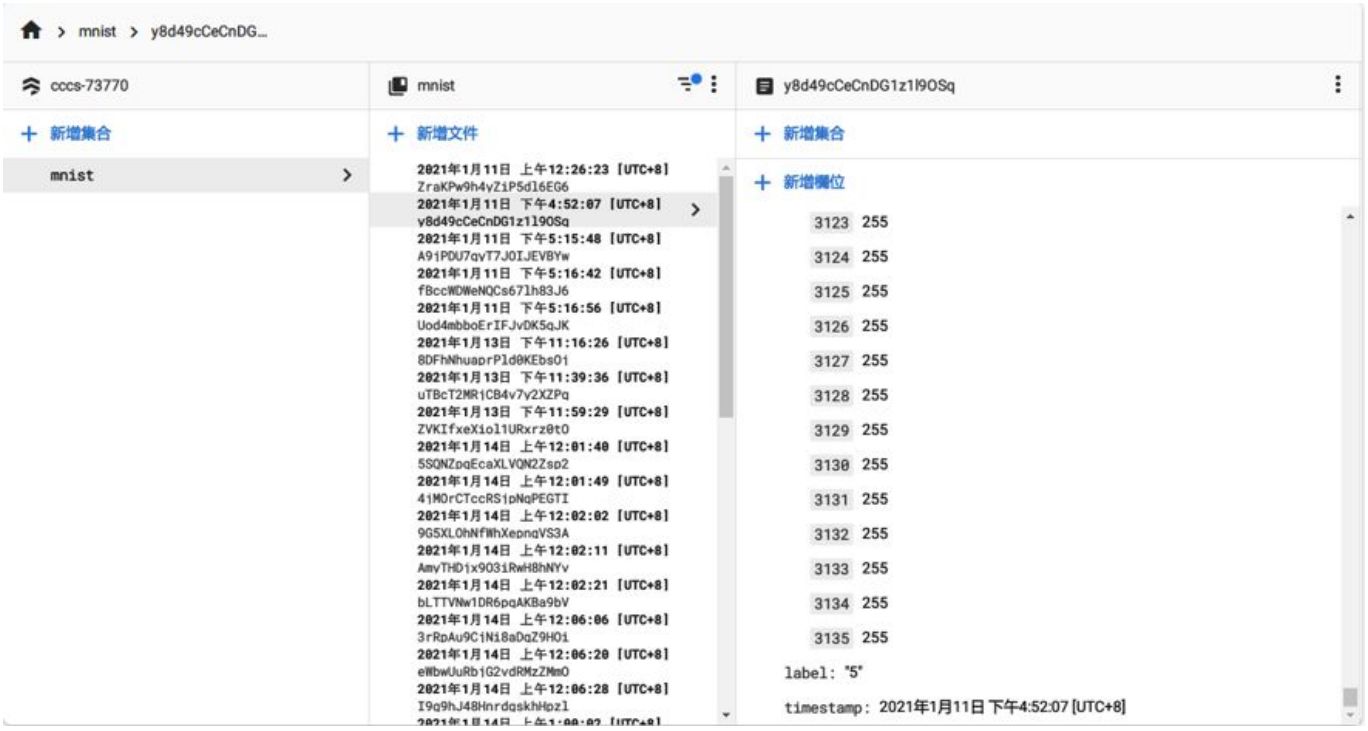
電腦版



手機版

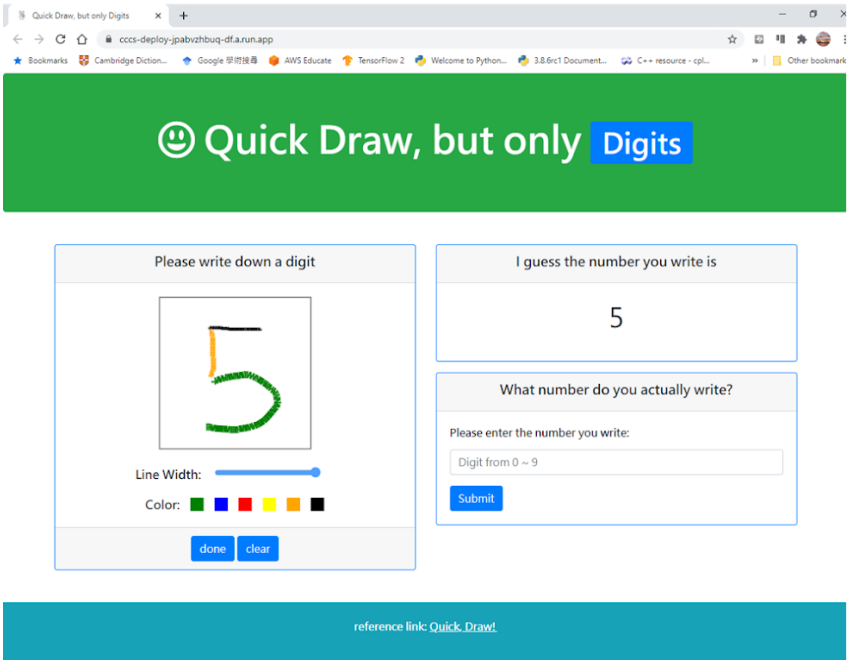


IV. 按下submit後，手寫數字的影像和實際的數字值傳入資料庫中

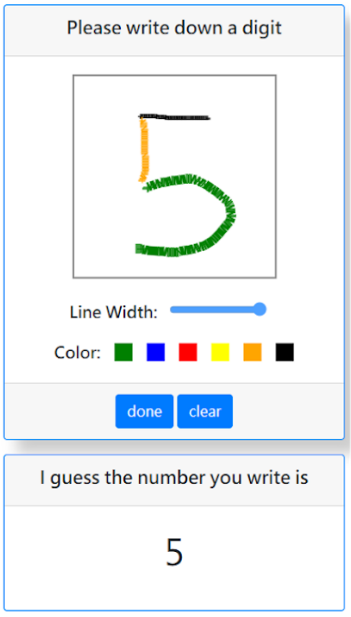


V. 使用者亦可改變手寫數字的顏色和粗細

電腦版



手機版



4. discussion

I. Inaccurate Prediction

因為我們的機器學習模型是用MNIST資料集訓練，MNIST中的資料數字大多都在整個圖片的中央，但實際上使用者可能是寫在任何地方，所以會讓我們的模型誤判這些寫在角落的數字，這個問題可能可以藉由先從傳入的整個圖片中找到數字，然後裁切留下數字的部分，再將這個部分當作模型的輸入；或是直接透過使用者寫的筆跡的順序進行預測，利用Sequential mnist資料來訓練；此外因為利用滑鼠寫的還是跟實際手寫的數字有差別，所以可能要再利用收集到的資料對原本的模型做Fine-tune。

II. Limits of Cloud Run

部屬方面，雖然利用Cloud Run可以讓整個部屬變得方便許多，但由於Cloud Run預設限制整個部屬流程最長10分鐘，所以需要注意應用程式dependencies的數量，以及選定的基底Docker image，否則可能會遇到部屬Timeout而失敗的問題。

III. Canvas on the smartphone

使用HTML的Canvas標籤製作手繪版在電腦版跟手機版上需要綁定的事件不同。在電腦版上綁定的是和滑鼠相關的事件，包括：“mousemove”、“mousedown”、“mouseup”和“mouseout”，然而這個到手機版上會起不到作用，必須要改成綁定“touchmove”、“touchstart”和“touchend”跟觸控相關的事件，並且在程式一開始判定使用者的裝置為手機或電腦做相對應的綁定。除此之外chrome檢查工具亦有提供手機版測試功能，非常方便。

5. references

I. quickdraw website
https://quickdraw.withgoogle.com/

II. How To Save Canvas As an Image With canvas.toDataURL()?
https://stackoverflow.com/questions/10673122/how-to-save-canvas-as-an-image-with-canvas-todataurl

III. Bootstrap v4.5 Document
https://getbootstrap.com/docs/4.5/getting-started/introduction/

IV. Font Awesome
https://fontawesome.com/

V. Detect Mobile Browsers | Open source mobile phone detection
http://detectmobilebrowsers.com/

VI. Cloud Run Document
https://cloud.google.com/run/docs

VII. Firestore Document
https://firebase.google.com/docs/firestore

VIII. Flask
https://flask.palletsprojects.com/en/1.1.x/