

基於 K8s 叢集的深度學習模型部署系統

DL model deployment on K8s

實習生: 劉昱劭
Mentor: 陳昭宏
[下載檔案](#)

I. 簡介(研究動機)

I-1. 雲端服務 (as a Service)

現今許多軟體架構趨向微服務化，軟體內的某些功能會使用雲端服務而非本地服務。

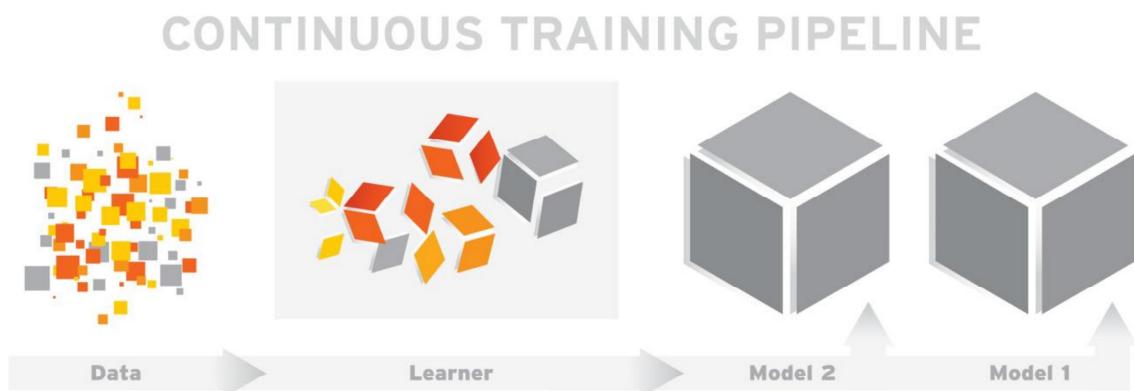
以 Uber 手機 App 為例，在 Uber 的 App 中的地圖資訊是直接使用 Google Map，而不是 Uber 自己重寫一個地圖系統，能省下重造輪子的時間。



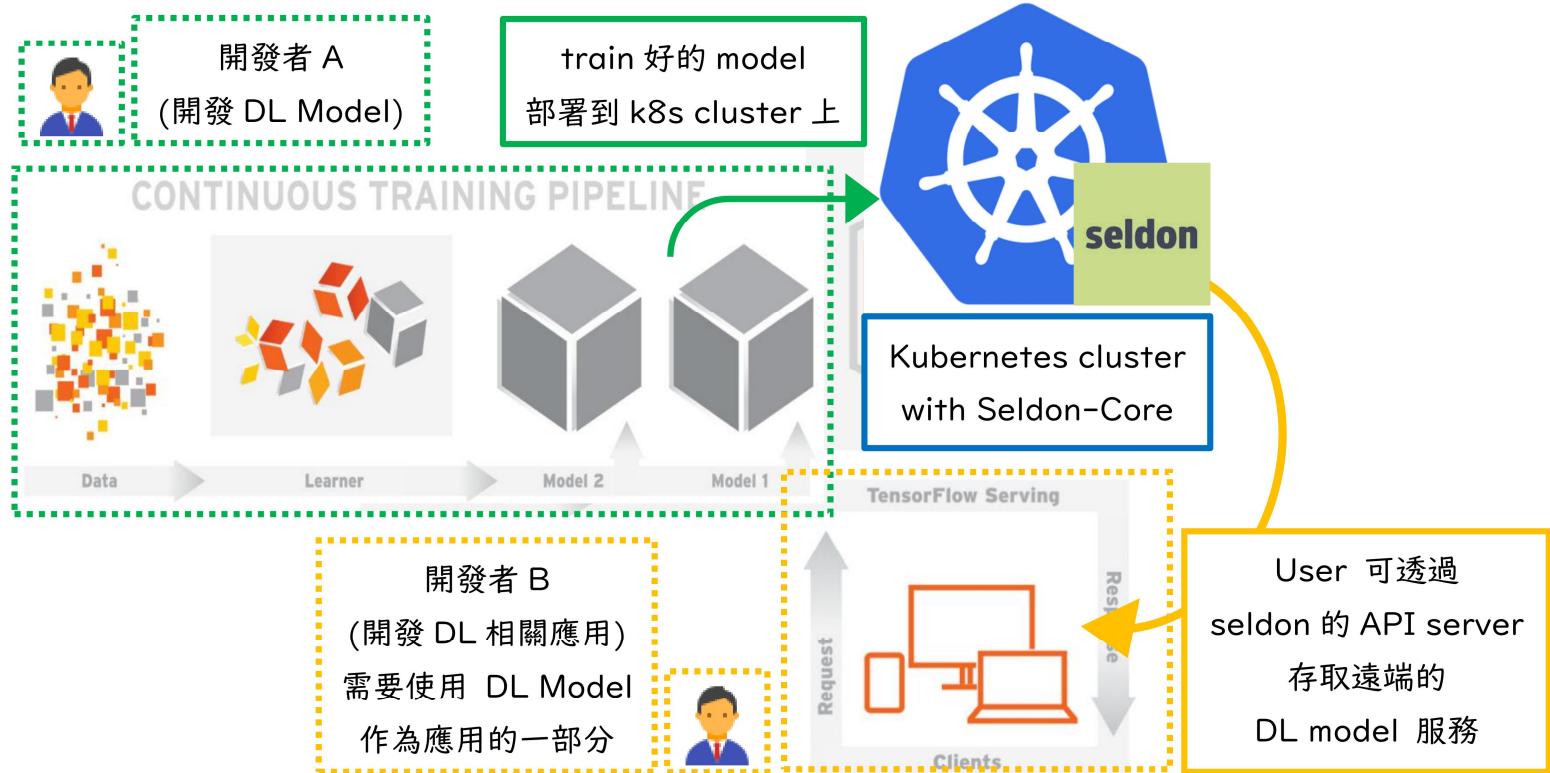
I-2. 目標：DLaaS (DL model 即服務)

開發 DL 應用的過程中，往往訓練模型是最花時間的。

DL 應用開發完畢後，如果能把已訓練完的模型公開出來作為服務供他人使用，需要同樣模型的開發者就不需要再花同樣的訓練時間。



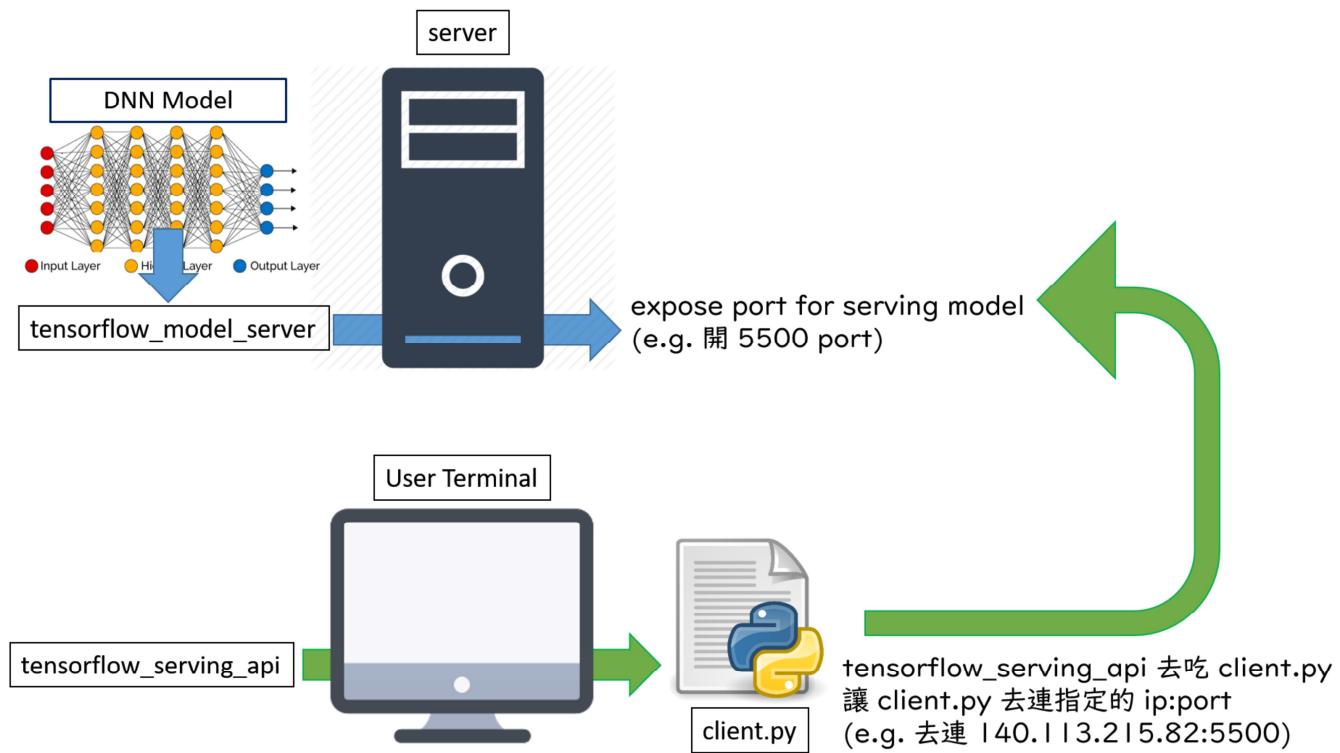
I-3. 情境



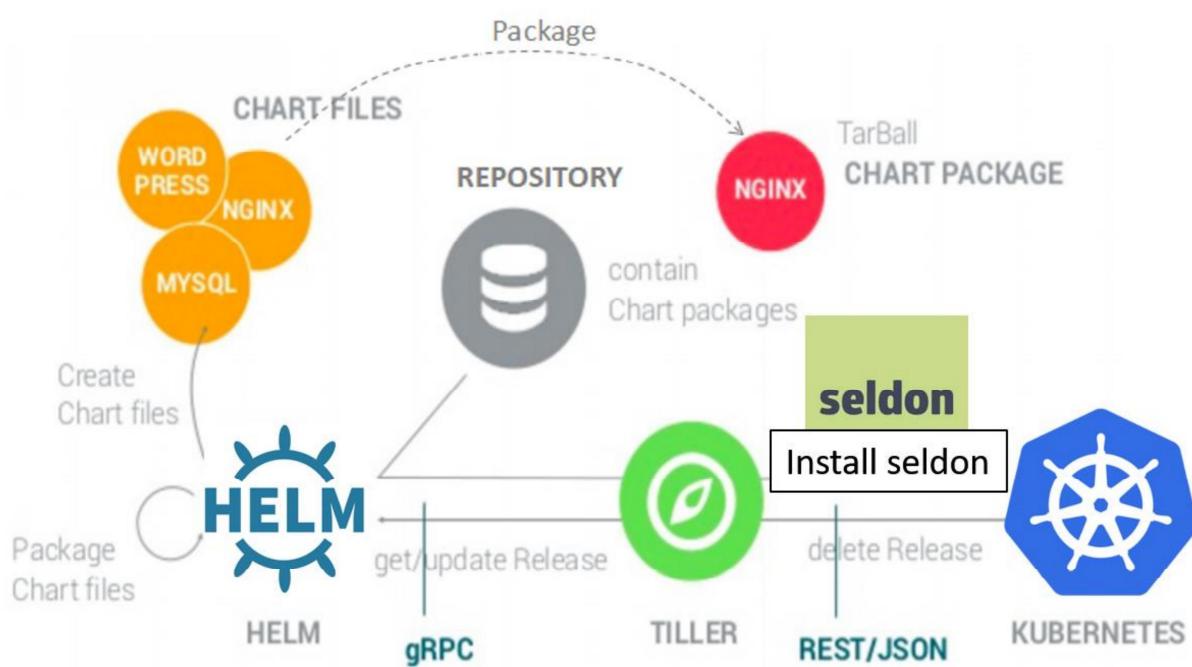
- 有一些人開發 DL model, 另一些人開發 DL 相關應用, 對於 B 這種開發團隊, 除了 DL model 外可能更重要的是 web 或 app 那種接觸使用者的應用。
- 但要開發一個完整的 DL 應用, 常常最花時間的是 train model。如果已經有人 train 好 B 要的 model , 並且願意公開出來給別人用, 那就能大幅減少整個開發的時間。
- Seldon core 做的就是中間的接口,
讓 DL 開發者 A 的模型可以輕易部署到該平台上
讓其他開發者 B 可以透過 RESTful / gRPC 等 API 連到平台使用 A 提供的模型。

2. 文獻探討(方法比較)

2-1. Tensorflow serving 架構



2-2. Seldon Core 架構

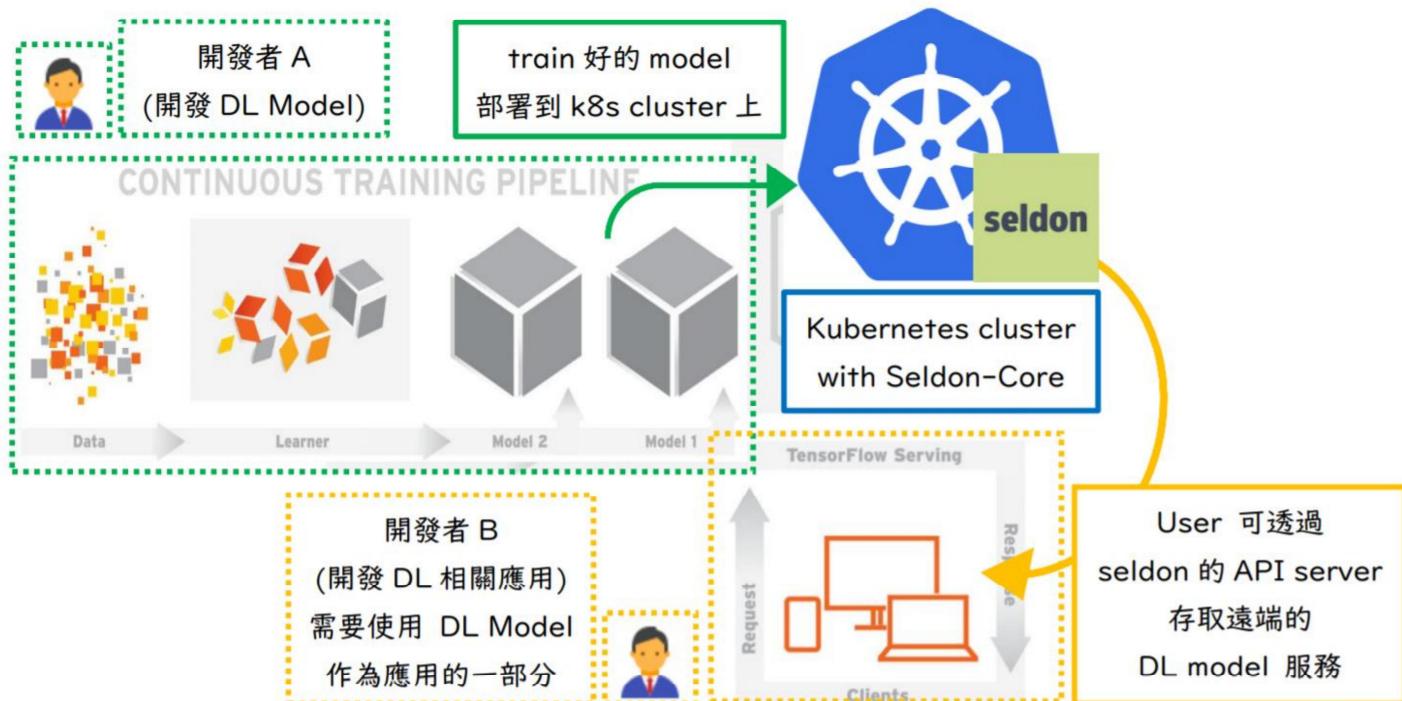


2-3. 比較

	Tensorflow serving	Seldon Core
安裝方式	安裝 tensorflow-model-server 1. 用 apt 直接安裝 package 2. 用 bazel 編譯 3. 官方推薦用 docker image 如果要用 GPU 跑 serving model, 官方也推薦使用他們包好的 GPU 版本 image	安裝 seldon-core 需依賴 helm/ksonnet 架在 k8s 叢集上
適合環境	單機 無 k8s 環境	1. 使用公有雲 (如 AWS, GCP 原生支援 k8s 管理) 2. 使用私有雲或自架 k8s 叢集
特性 (優缺點)	1. 對 python 支援度最高。 2. 支援 tensorflow, keras model。 3. 預設只支援 gRPC API。 4. 每 serve 一個 model 即占用一個 external port。 若在 container 環境下使用, 可能可以只使用 internal port。 5. 要使用 serving model, 需要用 pip 另載 API, 並寫一個 client.py 去存取給 serving model 暴露的 port。 <pre>pip install tensorflow-serving-api</pre>	1. 支援 python, R, Java, PMML 2. tensorflow, keras, sklearn... 應該說不管哪個 model 都支援 反正最後都是包成 docker image。 3. 支援 RESTful / gRPC API, 但兩種 API 會各自占用一個 port。 4. 因 seldon-core 在 k8s 叢集內, 可視需求只開 Internal port 只允許叢集內使用, 就不用佔用 external port。 5. 使用 serving model 只需要知道服務暴露的 ip:port, 即可直接發 REST/gRPC request。 [註]: k8s 在 v1.8 版本後移除不安全的 REST 連線機制, 因此 REST API 必須先和 k8s 叢集交換 token 和 secret。 [註]: 目前 gRPC 還是可以用 insecure channel 連到 k8s service。 6. 能輕鬆整合 kubeflow, FFDL, Istio 等 k8s 上的管理平台。 有其他常見的 k8s 管理測試工具已包成 helm chart 可一併建置。

	<p>兩者皆視為 service, 會占用伺服器 port 灌給 model 的 input 都透過網路傳輸，若 data 太大(如影片)可能造成效能瓶頸。</p>	
先備技能	<p>1. 只要會基本 Linux 命令列操作即可用 apt 下載並架起服務。 若要使用 Docker image 方式架設需要會基本的 Docker 使用。</p> <p>2. 只要會寫 client.py 即可使用 model。 結論: python, docker(optional)</p>	<p>1. 需要有基礎的 Docker 知識, 並會 k8s 的基本操作就能架起服務。</p> <p>2. 只要會發 REST/gRPC request 即可使用 model 結論: docker, k8s, REST(web/python)</p>

3. 研究方法 (系統架構)



4. 實驗結果與討論

4-1. 建置流程

- [hackmd](#)
- word

4-2. Demo 流程

1. 列出 K8s 的服務，顯示已 serve 的 mnist model 和 seldon-apiserver
(REST 開在 31794 port)

2. 預先準備了兩個 shell script 去抓 seldon-apiserver 的 REST 暴露在
140.113.215.82:31794

- **getMasterIP.sh**

抓 k8s cluster 的 master IP (適用於本地端 K8s 單 master 的 cluster)

```
kubectl get no --selector=node-role.kubernetes.io/master=master -o jsonpath='{.items[*].status.addresses[?(@.type=="InternalIP")].address}'
```

- **getSeldon.sh**

抓 seldon-apiserver 的第一個 external port

預設 REST API 會開在 gRPC 前面，所以是抓 REST 的 port

```
kubectl get svc --all-namespaces -l app=seldon-apiserver-container-app -o jsonpath='{.items[0].spec.ports[0].nodePort}'
```

3. 把 web 跑起來，並開在 8880 port，把 seldon-apiserver 的 port 作為參數傳入 web

4. 左邊已有開好的一個網頁畫板，是畫板原作者架好的，
右邊輸入 140.113.215.82:8880 連進剛架起來的網頁畫板。

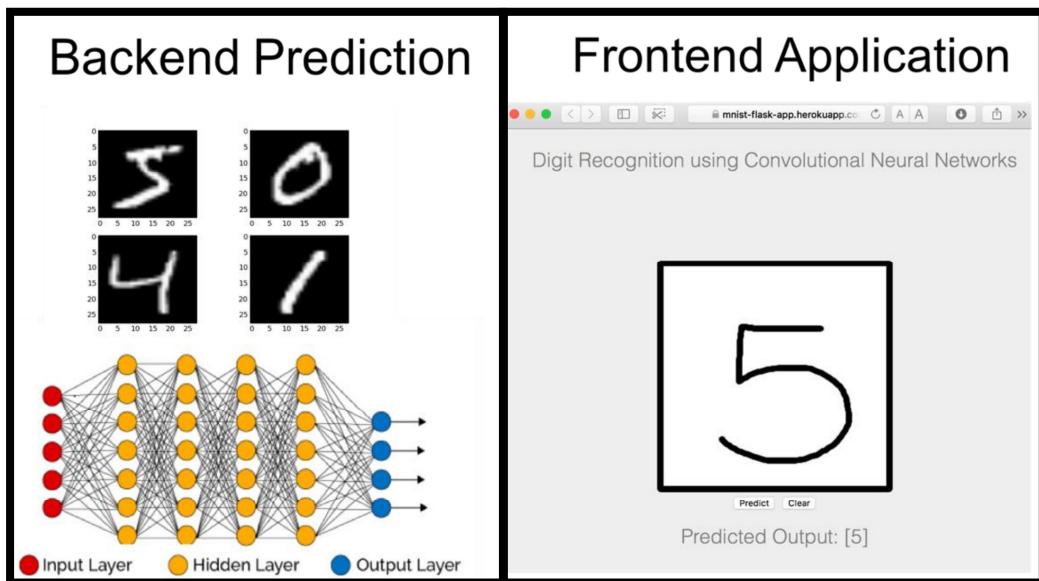
5. 左邊的網頁會將畫板的圖作 input，傳給作者機器 local 寫好的 mnist model 做預測。
右邊的網頁會將畫板的圖作 input，傳給我在 seldon-core 部署的 mnist model 做預測

4-3. Demo 影片 mp4

是不是感覺不太出來差異？

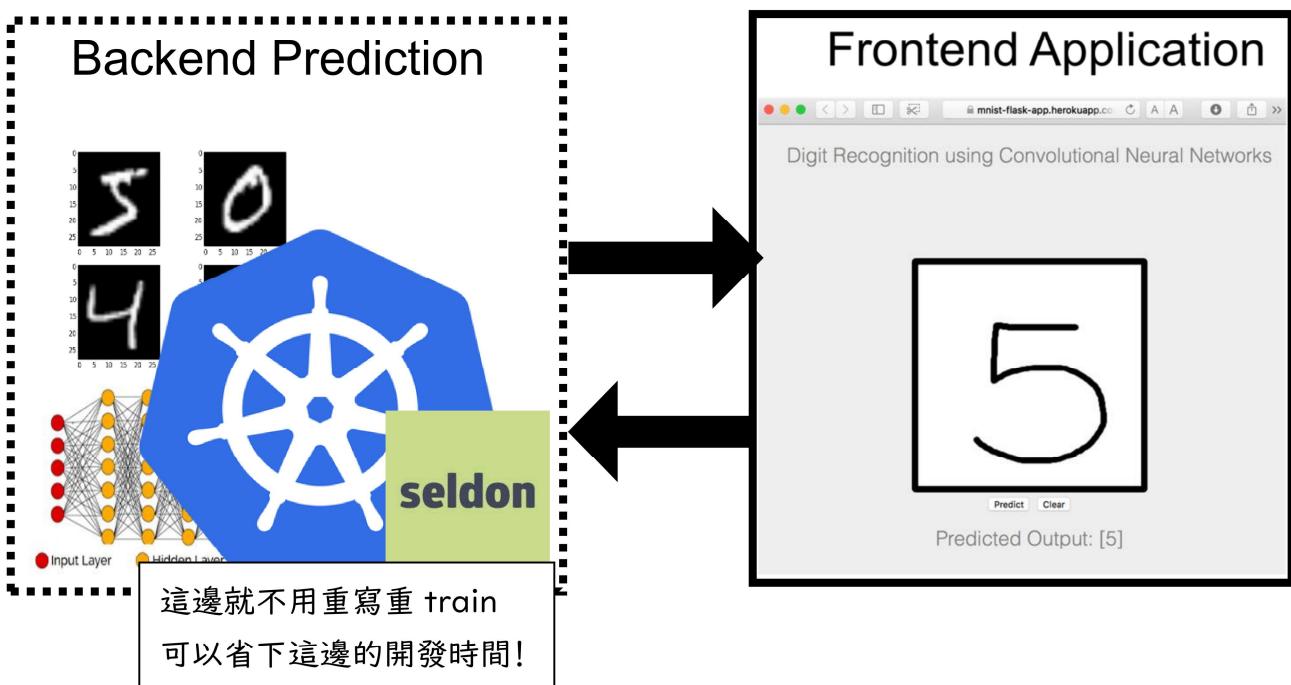
沒錯，如果目的是寫一個 MNIST 手寫辨識的 web app，就沒有必要再去重寫一個後端的預測模型，用現成的即可。

- 畫板原作者架構
 - 需開發前端 UI
 - 需重 train 後端 model



- 改過後端的架構

- 只需開發前端 UI



5. 待解決問題

seldon-core 的缺點

- seldon-core 是 seldon 放棄維護 seldon-server 後的新專案, 現在版本還在 v1 alpha2, 因此穩定度跟方便性不高。
- 透過 seldon-core 部署的模型將會暴露在 seldon-apiserver, 但是一次只能 serve 一個 model, 如果在已部署 model 的狀態下再部署新 model 的話, 將會直接覆蓋舊的那個, 即 apiserver 永遠只 serve 最後部上來的那個 model。
 - 部署了新 model, 舊 model 的 pod 依然會存在不會自動消失, 要手動砍掉。(類似 zombie 問題)
- 然而照這份流程來啟動 seldon 的話, seldon 相關的 addon 都是由 helm 啟動的, helm 無法同時安裝多個相同的 addon。
- 結論是: 一個 k8s cluster 目前只能用 seldon 部署一個 model。
(應該是可解決的問題, 目前猜測 k8s 建 replica set, 或是看 helm 如何啟動複數相同 chart)

6. 結論與建議

- 如果有建置叢集的經驗, 且手邊機器夠多, 可以使用 seldon-core 來減少開發時間成本
- 但如果有多個 model 要部署建議先使用 TF serving
- 要是會使用該模型的使用者不夠多, 乾脆整包 model 放在 NFS 之類的共享硬碟比較快。
(雖然這樣有可能遇到開發環境的相依套件互衝的問題)

7. 參考資料

7-1. [Kubeadm 快速建立 k8s 叢集](#)

7-2. [TF serving](#)

7-3. [Seldon Core](#)

[FfDL with seldon](#)

7-4. [前端畫版](#)

[畫版接上 local 端 mnist 模型](#)

7-5. IBM 其餘 k8s ML/DL platform 相關:

- [Fashion mnist using FfDL](#)
 - IBM 架在 FfDL 上的 Fashion Mnist 網頁應用
- [FfDL repo](#)
 - [註] FfDL 要吃的資源還蠻多的
- [IBM 的 object detection web api](#)
- [object detection web-app](#)