ML2018FALL Final Report

- 隊名: 老司機帶我飛

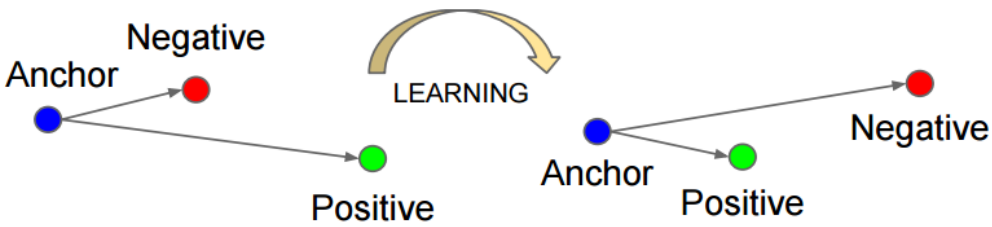
學號	姓名
b04902131	黃郁凱
b04902019	王士弘
b05902109	柯上優

Introduction & Motivation

- 在 VideoCaption 這個題目的引導中，助教建議我們實作 S2VT 。將影片變成 vector 並送入 RNN ，隨著時間序生出 sequence ，再和選項做連結，選出答案。然而 S2VT 作法，在轉換成sequence之後會失去許多information，因為影片可能有多種描述都符合這個影片，而model只能生出一句話來形容，顯然並非很好的作法。
- 我們決定參考論文[1]，藉著抽出影片和文字的 feature，直接在 feature level 上做比較，運用 triplet loss，將「影片的特徵向量」和「正確選項的特徵向量」拉近，和「錯誤選項的特徵向量」拉遠，直接以特徵向量做預測，如此不僅可以保留有完整的影片encode的資訊，更可以達到和選項做連結的目的。
- Triplet loss:

$$max(0, \alpha + S(f(v), f(w_n)) - S(f(v), f(w_p)))$$

其中 α 是 margin，p 是 positive example，n 是 negative example，$f(v)$是 video feature，也就是圖中的 anchor point ，$f(w)$是 sequence feature 。



- 我們認為在做選擇題的問題上，這種方法較為直觀且合理，甚者，對於 M 筆 Video 和各自 N 個 Caption，input data 可以產生組合 (v_p, w_p, w_n) 共 $N \times M \times (N-1) \times M$ 組資料做訓練，假設各個 Video 皆不相同，如次多樣的資料能讓我們的 model 更強大。

Data Preprocessing/Feature Engineering

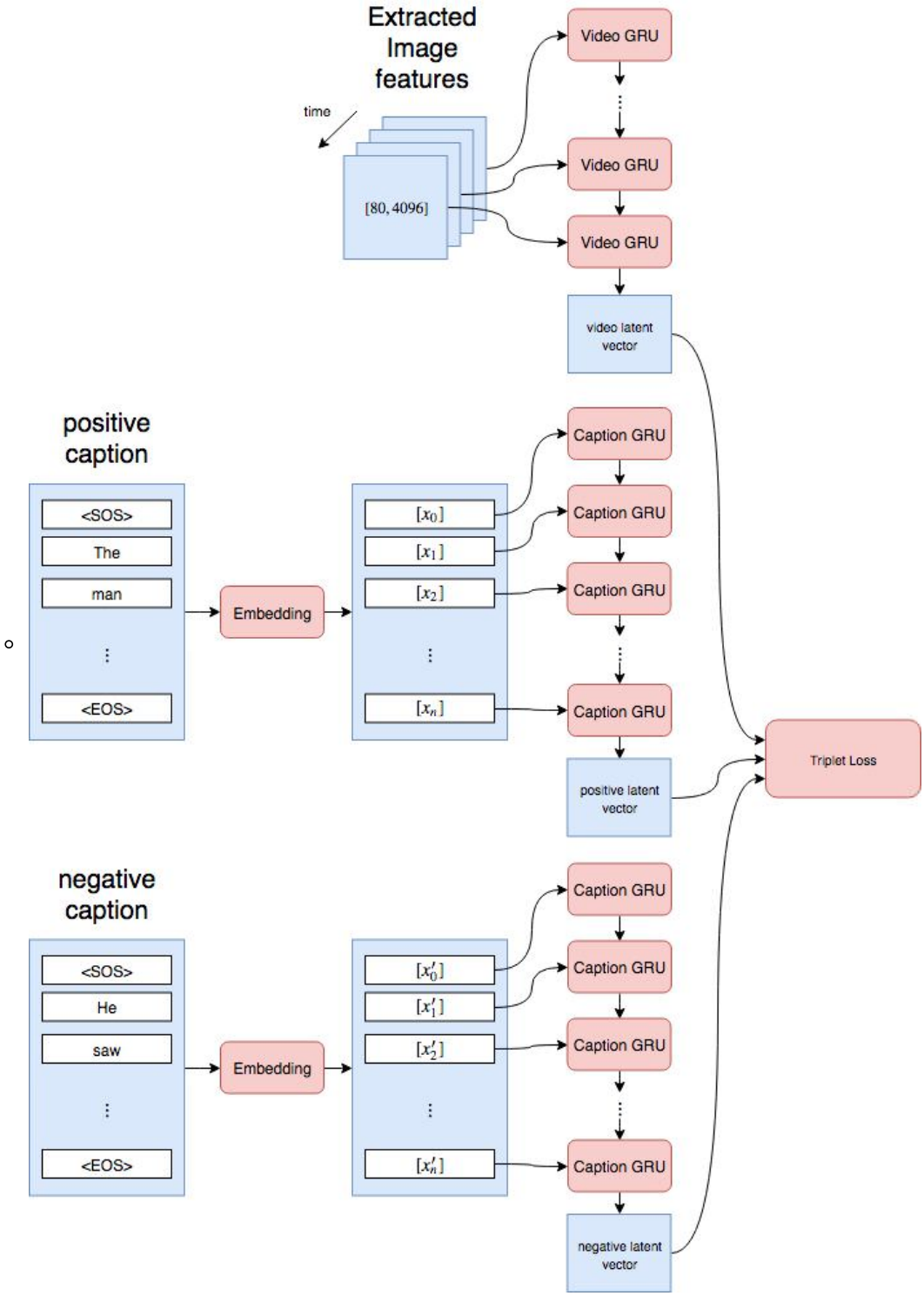
- Data Preprocessing
  - 英文讀檔的句型錯誤切割：由於 testing 的 csv 檔的儲存格式，和英文的逗點使用相同符號，若直接使用 python 套件讀取，會有錯誤的句型切割，我們花了些功夫處理。
  - 英文的等同意思合併：這個部分在英文的縮寫比較明顯，e.g. he's/he is、don't/do not 等等，其實是相同意義的，但為了減少不必要的誤會與計算，我們直接訂了統一標準，全數取代。
  - 特殊字元：選項裡有一些非基本單字的特殊字元，如冒號、驚嘆號等，我們把特殊字元都當一個個單字，獨立切割。

    - 句子的開始與結束：使用特殊符號做標註，這裡我們使用<SOS>(Start of Sentence)、<EOS>(End of Sentence)作為送入 RNN 的處理。

- Feature Engineering
    - WordEmbedding：我們沒有使用任何 pretrained model，而是使用 torch.nn.Embedding(V, D) 直接進行 end-to-end 的 forward 和 back propagation 。我們認為文字要和資料貼近，所以將他們一期進行訓練。大小是$|V| \cdot D$，V是vocabulary的大小，D是embedding dimension。
    - Padding：不同於有些 Padding 的教學中建議，把短的句子重複循環或用空字串填補，我們直接使用 torch.nn.utils.rnn.pack_padded_sequence()，能直接送入不同長度的句子。這樣能避免因RNN吃後面的 padding，而浪費訓練時間或改亂hidden state的值。
    - Hidden state：我們將 RNN 輸出的 hidden state 直接拿來使用，其中會有重要的隱藏特徵讓我們做訓練。

Model Description

- Model1 — Baseline model with triplet loss



- Video GRU
  - 負責萃取時間軸相關的影片資訊
  - 一開始 `hidden state` 給None（pytorch會initialize成全0）
  - 最後一個時間點GRU的 `hidden state` 當作濃縮影片的 feature vector，稱之 `video latent vector`
- Caption GRU
  - 萃取一個句子所含有的資訊
  - 一開始 `hidden state` 給None
  - 最後一個時間點的GRU的 `hidden state` 代表句子的 feature vector
  - 不管是 positive caption（與影片相關的句子）或者 negative caption（隨意抽取一個不相關的句子），都會使用同的GRU來 encode 資訊
- Score
  - 未調參數：41%
  - 調過參數：54%

        - 主要調整margin的距離，最佳參數使用0.01

- Model2 — Final Proposed model
    - 我們多了五項的作法，每項做法都幫助模型performance的提升，分別為 Embed avg, Hidden avg, Cosine, Common space, Ensemble
    - RNN 的架構和上面一樣，也沿用 triplet loss 等 loss function，以下一一介紹這五項改善的作法
    - Embedding vector average
        - 在參考論文[2]中，作者將 input sentence 做成 embedding vector 後，又在最尾端 concat 了所有 embedding vector 的 average，這使的他的 performance 進步。
        - 我們也使用相同的方法，舉例來說，假如有一句話「this is a pen.」，過完 word embedding 後會是

$$[V_{<sos>}, V_{this}, V_{is}, V_a, V_{pen}, V_., V_{<eos>}]$$

        - average後則變成

$$[(V_{<sos>} + V_{this} + V_{is} + V_a + V_{pen} + V_. + V_{<eos>})/7]$$

        - 直觀想法是，將每個字平均有 bag of word 的味道在，並且是 embedding 過後的，效果不輸純粹 bag of word 的做法。
    - Hidden state attention
        - Sentence vector 在經過 RNN 時，我們把每個時間點的 Hidden state 都取出來，並用 attention weight elementwise 相成合在一起。我們認為每個時間點的特徵都有其意義，並會隨每次遞迴而失去一些重要訊息，因此我們將他們都保留，一起作為輸出的特徵向量。
        - 由於 attention 訓練過程複雜，並且時間有限，我們在此使用 RNN 所有 hidden state 的平均做為代表，亦可以得到顯著的提升。
    - Cosine similarity
        - $similarity = cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$
        - 這也是出自於參考論文[2]的做法。相較於 L2 norm，論文的作者將 cosine similarity 作為 triplet loss 中計算向量距離的方法，我們也參考了這個構想並實作。
    - Common Space Learning
        - 由於 video 和 caption 的 feature 來源是兩個不同的 RNN encode 後的資訊，分別存在於兩個 feature space ，缺乏整合。因此 common space learning 就是將兩個不同 feature level 的資訊整合到同個 feature space 上。
    - Ensemble
        - 如同上課所教的，由於 task 複雜，巨大的神經網路訓練出的模型是 small bias, large variance 的，透過 diversity 很大的多個模型，做 ensemble 結果相當不錯。

- Inference
    - 將五句分別用caption GRU encode出五個feature vectors
    - 投影到 common space
    - 找與 video feature 距離最相近的當作輸出

Experiment and Discussion

Model	public accuracy
Baseline model	54.0
Embed avg + Cosine	56.4
Embed avg + Hidden avg + L2	57.4
Embed avg + Hidden avg + Cosine	58.0
Embed avg + Hidden avg + Cosine + common space	60.6
Ensemble	66.8

- 以上是對於 final model 增加各個方法時，正確率顯著提升的數據。其中 Embed avg 是 Embedding average ， Cosine 是用 cosine distance ， Hidden avg 是 RNN hidden state 的平均， common space 是運用 common space learning 技術。
- Triplet loss
    - Triplet Loss 很適合用於本次 final task，在選擇題的狀況下，一定會面臨「選擇較正確的選項」的情況，而 triplet loss 就能藉著調整句子間的距離，對的拉近，錯的拉遠，找出一種正確的平衡。
    - 這個方法十分有效，在完全沒有 tuning 參數的狀況下就在第一週成為排名第一名，此時的準確率就有 41%，直接超過當時的 simple badeline (21.6%) 和後來的 strong baseline (33.4%)。
- Cosine similarity
    - 與 L2 norm 做比較，使用後準確率都提高約 2~3 %，顯示了在這個 cosine similarity 在這個 task 上是較為適合的 distance function。
    - cosine similarity 是一種比起距離，更注重超平面上的角度的方式，這或許是選擇問題中，選較佳的解的好方法，畢竟比起探討物理距離的意義，相似度或許更重要。
    - 若我們沒有去閱覽其他參考論文，從一開始就只使用固定的的估算方法，可能就不會發現這種重要的差異性。當然，或許仍就有某個更適合的估算函數等著我們發現，更多的嘗試依然是重要的一環。
- average
    - 加上 embedding vector average 和 hidden state average 後，相較於沒加，都是約莫 2% 正確率的成長，可見得這些高維度向量的連接，取其平均值，在做預測與辨識時都是明顯有用的。
    - 由此可知，這些 feature 都是對於判斷選項有幫助，並無法忽視藏在當中的任何一點資訊。
- Common space
    - 加上之後，正確率又再飆升 2.5 % ，可見其重要性
    - 我們認為，要將 Video 的 feature 和 Caption 的 feature 放在一起做比較，或許該將他們都經過一個 activation function 和 linear function，在代數上才能視為在同一個空間。
- Ensemble
    - Voting
        - 每個 model 先各自選出自己的答案，並進行投票（票票等值），獲得最多票數的選項即為最終答案。若有多個選項平手，則選擇正確率最高的 model 所決定的選項。

- Sum of normalized distance
  - 我們每個模型，對於每個選項算出與影片的距離做標準化處理，然後再將不同模型預測的距離相加，取最小者，效果較前者為優。Ensemble 的 model 分別是用單獨模型排名的前幾名。
- 比較
  - 我們對前三名的 model 使用上述的兩種 ensemble 方法，所得結果如下：使用 Voting 所得的準確率為 61.8 %；使用 Sum of normalized distance 的準確率則為 64.0 %。
  - 由結果可以得知，使用 Sum of normalized distance 的效果比起 Voting 還要好。推測其可能原因為，Sum of normalized distance 較 Voting 更能根據選項在所給 model 中整體的好壞程度，來選擇最終的選項。
  - 舉例來說，對於某一題，若有一 model A 所算出的各個選項的距離值皆很接近，代表 model A 不擅長這題。而假設有另一 model B 較擅長這題，也就是他所算出的各個選項的距離值中，有一選項的值小於其他選項的值一定程度。以常理而言，model B 的答案應比 model A 的答案還要可靠，應賦予較高的權重。然而，Voting 會將每個 model 的答案視為同等重要，造成 model B 無法在這題上表現其優勢。反之，Sum of normalized distance 則會利用選項間距離值的差距來決定最終答案，使得 model B 對結果的影響比 model A 的還大，所以會有較高的準確率。

Conclusion

- 在多次實驗後，我們覺得以下幾點是十分重要的
  - triplet loss function
  - input feature: embedding and hidden state average
  - distence function
  - common space
- 他們都是機器學習裡，十分基本且不會被人注意到的學問與小技術，但是採用與否的影響，卻遠遠超過單純做參數的調整，大大改變了整個訓練的過程和準確度。
- 這些東西除了上課學習之外，課外的論文閱讀、期刊查閱、與他人的技術分享都是不可或缺的。當然，日日夜夜的嘗試，失敗與成功，都是必經之路。
- 在最後一堂課，聽完其他組的報告後，我們也發現了一些可以改進的地方，與未來可以嘗試的方向。
  - Train with Validation Data
    - 我們做訓練時，安全起見，都有切 $\frac{1}{10}$ 的資料做 validation testing。
    - 目前最佳模型是不使用完整的資料做訓練下的，有第二名的成績。若能將所有訓練資料全數投入，更高的準確率是必定的。
  - Data cleaning
    - 有一組提到，training data 並沒有我們想像的乾淨，他們有發現裡面混有不少雜訊與不完整的句子。
    - 一種方法是手動清理，我們也許能分工，花個三五個小時認真把 training data 瀏覽過一遍，清理意義不明的句子。
    - 另一種方法是去尋找其他能修正文法與拼音的套件，直接將錯字修正。畢竟錯字這方面我們幾乎沒有花時間去檢查，這種類型的雜訊也的確會嚴重影響訓練的準確率。
  - Pretrained embedding
    - 第一名的組別有提到，他們的 word embedding 是使用 pretrain model。
    - 我們是使用 unsupervised 的 end-to-end training，沒有使用任何套件。
    - 想當然爾，這種差異性所造成的差距絕對不小，也絕對能加速模型的收斂速度。

    - Extra data
        - 第一名的組別也說過，他們使用了更多的外部資料做訓練，其中一個便是來自 Google 的 Video Caption dataset [3]。
        - 更多的資料，一定能讓模型有更強大的能力。我們只靠著課堂給的資源，理所當然的會差對方一大截。
    - Distance function (combine two distance function with a combine rate)
        - 在我們的最終模型，大部分使用了 cosine similarity，少部分模型選擇 L2-norm。
        - 與其要在兩邊做選擇，或許也可以考慮結合著一起用？
          $Distance(v_1, v_2) = \alpha\|v_1 - v_2\|_2 + \beta \times cos(v_1, v_2)$
            - 用係數β, γ來調整之間的比例，當然這需要更多的嘗試，更多的時間與硬體資源
        - 又或者，完全沒有嘗試過，全新的距離公式？
            - Tanimoto: $T(v_1, v_2) = \frac{v_1 \cdot v_2}{\|v_1\|^2 + \|v_2\|^2 - v_1 \cdot v_2}$
- 還有很多技術與方向都等著被發掘，而這些都會未來的深入研究，可以走的方向。

Reference

- [1] Jianfeng Dong, Xirong Li, Chaoxi Xu, Shouling Ji, Xun Wang “Dual Dense Encoding for Zero-Example Video Retrieval”
- [2] Jianfeng Dong, Xirong Li, and Cees G. M. Snoek “Predicting Visual Features from Text for Image and Video Caption Retrieval”
- [3] Conceptual Captions: A New Dataset and Challenge for Image Captioning https://ai.googleblog.com/2018/09/conceptual-captions-new-dataset-and.html (https://ai.googleblog.com/2018/09/conceptual-captions-new-dataset-and.html)
- [4] Triplet Loss Introduction: https://blog.csdn.net/tangwei2014/article/details/46788025 (https://blog.csdn.net/tangwei2014/article/details/46788025)