

NLP 2024 Term Project

Task: Learning Agency Lab – Automated Essay Scoring 2.0

Team ID: 39 Member: 陳軒宇、許祐瑜、楊采語、陳幸榆

一、任務描述

我們的任務是 Learning Agency Lab - Automated Essay Scoring 2.0，此任務延續 2012 年的 Automated Essay Scoring 競賽，目標是開發出能準確對學生文章進行評分的 AI 模型，提供一個及時且一致的回饋，幫助提升學生的寫作技能與學習經驗，並協助減輕老師批改文章的工作量，讓老師能更專心於教學的準備。

此次競賽提供的數據集包含約 24,000 篇學生撰寫的論證性文章，所有文章均根據一個整體性評分標準進行評分，分數範圍從 1 到 6 分不等。最高分 6 分代表學生在觀點掌握、文章結構及語言運用方面均表現優異。

數據集主要分為三個部分：訓練資料、測試資料和提交資料。其中，訓練資料包括完整的作文文本及其對應的評分，測試資料則包含僅有文本的文章以供模型進行評估，而提交資料則需要包含參賽者模型的預測結果。

比賽設置了兩種主要的評分方式，分別針對模型的準確性及效率進行評估。第一種方式根據模型在 Leaderboard 排名中的表現進行評比，其核心指標為 Quadratic Weighted Kappa，該指標專為衡量實際結果與預測結果之間的一致性而設計，能夠精準反映模型在處理具有順序性數據時的準確性。第二種方式則著眼於 Efficiency Leaderboard，該評估方式除了考慮模型的預測準確性，還將運算效率納入評估範疇，鼓勵參賽者在提高準確性的同時，設計出更高效的模型。Efficiency Score 的設計目的是在推動準確性與效率間達到平衡，旨在促使參賽者提出能適用於實際教育場景的解決方案，這對於人工智慧技術在教育領域的應用具有深遠意義。通過這次比賽，我們將面臨如何在保持高準確率的同時優化運算成本的挑戰，進一步推動自動化作文評分技術的發展。

二、模型比較及其困難點

這次的任務我們主要參考 Automated Essay Scoring (AES) 2.0 的 Notebook 上公開的 TF-IDF + LGBM Baseline 與 DeBERTa 模型，並進行比較與分析。

1. TF-IDF + LGBM Baseline

Model Architecture	TF-IDF	透過計算每個 text data 的詞頻 frequency-inverse document frequency，將文字資料轉換為數字特徵。
	LGBM	透過梯度提升 framework，使用基於樹狀結構的學習演算法進行迴歸（相關性評分 relatedness score）和分類（蘊涵判斷 entailment judgment）分析。
Training Time	<p>由於 TF-IDF 和 LGBM 模型的簡單性，與 DeBERTa 等深度學習模式相比，訓練速度通常更快，但是收斂的速度慢。</p> <div><div></div>5/? [06:37<00:00, 78.74s/it]</div> <div><div></div><div>Fold_1 Training ===== Training until validation scores don't improve for 75 rounds [25] train's QWK: 0.75155 valid's QWK: 0.746101 [50] train's QWK: 0.794645 valid's QWK: 0.77996 [75] train's QWK: 0.813624 valid's QWK: 0.792911 [100] train's QWK: 0.825631 valid's QWK: 0.799515 [125] train's QWK: 0.834944 valid's QWK: 0.800346 [150] train's QWK: 0.843876 valid's QWK: 0.802164 [175] train's QWK: 0.851607 valid's QWK: 0.806055 [200] train's QWK: 0.85921 valid's QWK: 0.80502 [225] train's QWK: 0.865647 valid's QWK: 0.806328 [250] train's QWK: 0.87011 valid's QWK: 0.807328 [275] train's QWK: 0.875305 valid's QWK: 0.806221 [300] train's QWK: 0.880016 valid's QWK: 0.807135 Early stopping, best iteration is: [243] train's QWK: 0.868853 valid's QWK: 0.808049 Evaluated only: QWK</div></div> <p>以程式實績執行結果來看： 程式要執行到 [243] iteration 以後才有 QWK: 0.80 的水準</p>	
Evaluation Metrics	<p>量測方式為 Relatedness Score 與 Entailment Judgment</p> <p>Relatedness Score: Mean Squared Error (MSE)</p> <p>Entailment Judgment: Accuracy</p>	

2. DeBERTa

Model Architecture	增強型 BERT 具有解譯文本相關性(Disentangled Attention)，提高了模型理解文本內上下文和關係的能力。
Training Time	由於 DeBERTa 模型的複雜以及需要對大型資料集進行微調，訓練速度通常較久，但是收斂的速度好。
Evaluation Metrics	Relatedness Score: Mean Squared Error (MSE) Entailment Judgment: Accuracy

3. 綜合比較

performance compare	
與 TF-IDF + LGBM Baseline 相比，DeBERTa 通常能夠實現更高的準確度和更低 MSE，因為它具有先進的架構和捕捉文本中複雜關係的能力。	
Training and Inference Time	
TF-IDF + LGBM	訓練和推理速度更快，適合計算資源有限或需要快速得出結果的場景。
DeBERTa	需要更多的運算資源和時間，但提供更好的效能，使其適合精確度至關重要的應用。
Scalability	
TF-IDF + LGBM	可以以最小的計算開銷輕鬆擴展到更大的資料集。
DeBERTa	需要大量運算資源來擴展，尤其是對於非常大的資料集。

4. TF-IDF + LGBM Baseline 困難點

由程式執行結果可知，程式要執行到 [243] iteration 以後才有 QWK: 0.80 的水準。

所以，如何善用 TF-IDF + LGBM Baseline 能快速訓練和推理的能力，以分析 Automated Essay Scoring (AES) 2.0 是很好的改善方向，而如何利用 DeBERTa 通常能夠實現更高的準確度的特性，又或者利用其他方式，能直接使用 TF-IDF + LGBM Baseline 以加速訓練和推理的結果，是我們的目標。

5. 程式碼範例附列

1. TF-IDF + LGBM

```
> AI > Kitt.2024 > NLP > NPL_FINAL_01 > TF-IDF+LGBM.py > ...
1 from sklearn.feature_extraction.text import TfidfVectorizer
2 from sklearn.model_selection import train_test_split
3 import lightgbm as lgb
4 from sklearn.metrics import mean_squared_error, accuracy_score
5 import pandas as pd
6
7 # Load and preprocess the dataset
8 dataset = load_dataset("learning_agency_lab/automated_essay_scoring_2.0")
9 df = pd.DataFrame(dataset['train'])
10 df['text'] = df['premise'] + " " + df['hypothesis']
11 train_data, test_data = train_test_split(df, test_size=0.2, random_state=42)
12 vectorizer = TfidfVectorizer(max_features=10000)
13 X_train = vectorizer.fit_transform(train_data['text'])
14 X_test = vectorizer.transform(test_data['text'])
15 y_train_relatedness = train_data['relatedness_score']
16 y_train_entailment = train_data['entailment_judgment']
17 y_test_relatedness = test_data['relatedness_score']
18 y_test_entailment = test_data['entailment_judgment']
19
20 # Train LightGBM models
21 lgb_train_relatedness = lgb.Dataset(X_train, label=y_train_relatedness)
22 lgb_eval_relatedness = lgb.Dataset(X_test, label=y_test_relatedness, reference=lgb_train_relatedness)
23 params_relatedness = {'objective': 'regression', 'metric': 'rmse', 'boosting_type': 'gbdt', 'num_leaves': 31,
24 | | | | | 'learning_rate': 0.05, 'feature_fraction': 0.9}
25 model_relatedness = lgb.train(params_relatedness, lgb_train_relatedness, num_boost_round=100,
26 | | | | | valid_sets=lgb_eval_relatedness, early_stopping_rounds=10)
27
28 lgb_train_entailment = lgb.Dataset(X_train, label=y_train_entailment)
29 lgb_eval_entailment = lgb.Dataset(X_test, label=y_test_entailment, reference=lgb_train_entailment)
30 params_entailment = {'objective': 'multiclass', 'metric': 'multi_logloss', 'boosting_type': 'gbdt',
31 | | | | | 'num_leaves': 31, 'learning_rate': 0.05, 'feature_fraction': 0.9, 'num_class': 3}
32 model_entailment = lgb.train(params_entailment, lgb_train_entailment, num_boost_round=100,
33 | | | | | valid_sets=lgb_eval_entailment, early_stopping_rounds=10)
34
35 # Evaluate models
36 y_pred_relatedness = model_relatedness.predict(X_test, num_iteration=model_relatedness.best_iteration)
37 mse_relatedness = mean_squared_error(y_test_relatedness, y_pred_relatedness)
38 print(f"Relatedness MSE: {mse_relatedness}")
39
40 y_pred_entailment = model_entailment.predict(X_test, num_iteration=model_entailment.best_iteration)
41 y_pred_entailment = [np.argmax(pred) for pred in y_pred_entailment]
42 accuracy_entailment = accuracy_score(y_test_entailment, y_pred_entailment)
43 print(f"Entailment Accuracy: {accuracy_entailment}")
44
```

2. DeBERTa

```
D: > AI > Kitt.2024 > NLP > NPL_FINAL_01 > DeBERTa.py > ...
1 #DeBERTa
2 from transformers import DebertaV2Tokenizer, DebertaV2ForSequenceClassification, Trainer, TrainingArguments
3 from datasets import load_dataset
4
5 # Load and preprocess the dataset
6 dataset = load_dataset("learning_agency_lab/automated_essay_scoring_2.0")
7 tokenizer = DebertaV2Tokenizer.from_pretrained("microsoft/deberta-v3-base")
8 def preprocess_function(examples):
9 | return tokenizer(examples['premise'], examples['hypothesis'], truncation=True, padding="max_length")
10 tokenized_dataset = dataset.map(preprocess_function, batched=True)
11 tokenized_dataset.set_format(type="torch", columns=["input_ids", "attention_mask", "relatedness_score",
12 | | | | | "entailment_judgment"])
13
14 # Define the model
15 model = DebertaV2ForSequenceClassification.from_pretrained("microsoft/deberta-v3-base", num_labels=3)
16
17 # Training arguments
18 training_args = TrainingArguments(output_dir="./results", evaluation_strategy="epoch",
19 | | | | | learning_rate=2e-5, per_device_train_batch_size=8, per_device_eval_batch_size=8,
20 | | | | | num_train_epochs=3, weight_decay=0.01)
21
22 # Define the Trainer
23 trainer = Trainer(model=model, args=training_args, train_dataset=tokenized_dataset["train"],
24 | | | | | eval_dataset=tokenized_dataset["validation"], tokenizer=tokenizer)
25
26 # Train and evaluate the model
27 trainer.train()
28 trainer.evaluate()
```

三、NLP 技術效能分析

在本次的實作方法中使用了以下技術：

1. 段落、句子、單字的統計分析

段落	計算段落長度、段落數量等統計量，這些特徵表示文本的結構與框架，評估文章的組織性以及連貫性。
句子	計算句子數量、平均句長等特徵，用於評估語言的表達能力。
單字	通過單字數量和平均單字長度，捕捉文本的複雜程度及詞彙豐富性。

2. TF-IDF: 分別計算 word 以及 character level 的權重。

3. DeBERTa tokenizer 所得到的 embedding: 透過 tokenizer 將輸入文本轉為能夠捕捉上下文語意的 embedding。





	Private Score	Public Score
Version 1	0.83230	0.81572
Version 3	0.83505	0.81887
Version 4	0.81345	0.79632
Version 5	0.54570	0.54467

Version 1：這是包含所有特徵的模型，作為基準版本，其分數表現良好，證明結合統計特徵、TF-IDF 特徵以及 DeBERTa 嵌入能有效捕捉數據的特徵信息。

Version 3：**去掉 TF-IDF** 特徵後，Private Score 與 Public Score 均略有提高，表明對於這個數據集，TF-IDF 特徵可能帶來了一些冗餘信息，進而干擾了模型的性能。

Version 4：**去除 DeBERTa embedding features** 後，分數顯著下降，表明 DeBERTa tokenizer 所得到的 features 對於模型性能具有重要影響。這是因為 DeBERTa 能夠捕捉深層次的語義信息和上下文特徵，去除後模型對文本的理解能力減弱。

Version 5：**只保留 DeBERTa 特徵**這個版本的分數顯著低於其他版本，顯示只依賴 DeBERTa 的 embedding vector 並不能充分捕捉該數據集的全部有效信息，仍需要 TF-IDF 以及統計分析特徵來補足文本結構等特性。

Submission and Description		Private Score	Public Score	Selected
	DeBERTa+Lgbm - Version 5 Succeeded (after deadline) · 1h ago · w/ DeBERTa features only	0.54570	0.54467	<input type="checkbox"/>
	DeBERTa+Lgbm - Version 4 Succeeded (after deadline) · 4h ago · w/o DeBERTa features	0.81345	0.79632	<input type="checkbox"/>
	DeBERTa+Lgbm - Version 3 Succeeded (after deadline) · 11h ago · w/o TDIDF features	0.83505	0.81887	<input type="checkbox"/>
	DeBERTa+Lgbm - Version 1 Succeeded (after deadline) · 12h ago	0.83230	0.81572	<input type="checkbox"/>

四、其他組別的做法討論與改進想法

1. 資料分析與處理

多數組別在數據分析中指出，本次訓練集的資料來源與 Persuade2.0 數據集高度重疊，尤其是其中 7 主題佔據了大部分數據分布，其中 3 個主題尤為集中，且分數偏低，導致可能存在的資料偏差問題。有組別進一步使用 TF-IDF 分析，驗證這種不平衡性對模型的影響，並提出透過調整 threshold 來緩解此問題。另一組則觀察到，不同來源的文章可能存在評分標準差異，並嘗試將數據來源作為輔助特徵加入模型中，以提升對標準差異的適應能力。

討論與改進想法：在資料偏差的情況下，可考慮採用數據增強技術，對低頻主題進行 oversampling 來改善主題分布的不均衡。對於不同來源的數據，可進一步採用分層建模的方法，例如為不同來源訓練子模型，然後通過集成方法進行融合，提升模型的適配性與泛化性。

2. 模型結構與優化策略

多數組使用了 DeBERTa 作為語言建模的基礎模型，並結合不同的特徵或技術進行優化。例如，有組別結合了 Contrastive Learning，以增強模型在相同主題內的分數區別能力；另一組則使用 FGM (Fast Gradient Method) 攻擊來加強模型的穩健性，但發現目前效果並不理想。此外，某些組別將回歸任務與分類任務進行對比，發現回歸模型在捕捉分數連續性上的表現優於分類模型。而有些組別與我們一樣使用 DeBERTa + LightGBM 模型，但他們還在此基礎上引入了基於文本結構的特徵，包括

起承轉合的關鍵詞和引用詞等，讓模型可以觀察到這些特徵與文章的分數具有一定的相關性。


討論與改進想法：使用 DeBERTa + LightGBM 可能需要進一步探索如何充分挖掘顯式特徵與隱式特徵的互補性，例如通過特徵重要性分析來調整不同特徵的權重。


3. 評分標準與 threshold 設定

部分組別在研究中指出，不同分數區間的文章可能受限於標註者的主觀偏差，導致模型難以準確學習分數間的距離差異。有組別採用了多數投票法來決定模型的 threshold，或者基於 LSTM 的語意分數與人工標註特徵進行拼接，並使用融合模型進行學習。

討論與改進想法：在 threshold 設定上，嘗試動態調整策略，例如基於數據分布進行自適應調整，而非固定的投票機制，以更好地適應不同數據集的特性。結合人工標註特徵的方式雖能提升模型效果，但需注意避免過於依賴人工特徵而削弱模型的自適應能力。或許可以嘗試通過特徵選擇算法自動篩選高相關性特徵。

五、最後 performance 數據

 Submission Details

**NLP G39 - Version 1**
Succeeded (after deadline) · 2h ago

Score: 0.80708
Private score: 0.82997

六、參考資料

1. TFIDF + LGBM: <https://www.kaggle.com/code/ye11725/tfidf-lgbm-baseline-cv-0-799-lb-0-799/input>
2. DeBERTa: <https://www.kaggle.com/code/cdeotte/deberta-v3-small-starter-cv-0-820-lb-0-800/notebook>