NLP Final Project

Task1 Report

Classification of Regulations for Unlawful Ads Based on Govt. Notice

政府公告的違規廣告之法條分類

Group 16

小組成員

R11922008　吳庭維

T11902210　張一凡

R09922046　陳奕均

實驗日期：2023 年 5 月

一、問題背景與介紹

1、任務描述

消費者每天都會接觸到各種形式的廣告，包括電視、網站、傳單、視頻等。廣告已經成為消費者瞭解新產品功能的最快管道。廣告是誇大還是欺騙的問題經常讓消費者頭疼。

本次工作任務一是政府公告的違規廣告之法條分類，即根據對應的法律條文將所給出的廣告條文進行分類，而且這是一個二分類問題，其擁有 3828 條訓練數據和 100 條測試數據，並且最終需要提交到 kaggle 平臺上進行評分檢測。

2、數據說明

廣告的數據來源於政府公開數據收集中的非法廣告，下方是所給出的示例：

臺北市政府衛生局98年7月份處理化粧品違規廣告處罰案件統計表

項次	產品名稱	來源	違規情節	處分商號名稱	罰則註記
1	超美白精華液	網路	廣告內容宣稱：「…最強 美白革命 零斑點，超美白…1.添加高濃度2%維他命C醣苷，具強大還原力，讓輕鬆妳由黑轉白 2.防禦、修護、亮白、保濕、淡斑，五大效能全面對抗黑色素，白的更迅速…超高濃度-維他命C醣苷2% 使用2週後明顯變白！！不斷產生黑色素的細胞 強大還原力 黑色素變白 經維他命C還原淡化…」等文詞。	研社股份有限公司	化粧品衛生管理條例第24條第2項

所給出的數據形式擁有四個特徵：（其中訓練集擁有所有的標籤，而測試集是需要將最後一個標籤作為預測值進行輸出最後評分）

標籤	說明
Source_id	id for each source_id
Text	False advertisement Segment(From: 違規情節)
Label	Name of the Regulation Violated(From: 罰則註記)
Label_for_kaggle	label for submit

二、解題思路

我們首先要根據問題的提示法律條文的時間來源 2009-2016（民國 98 年~105 年）確定具體的文字：

第二十四條第一項：化粧品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播猥褻、有傷風化或虛偽誇大之廣告。

第二十四條第二項：化粧品之廠商登載或宣播廣告時，應於事前將所有文字、畫面或言詞，申請中央或直轄市衛生主管機關核准，並向傳播機構繳驗核准之證

明文件。

第二十四條第三項：經中央或直轄市衛生主管機關依前項規定核准之化粧品廣告，自核發證明文件之日起算，其有效期間為一年，期滿仍需繼續廣告者，得申請原核准之衛生主管機關延長之，每次核准延長之期間不得逾一年；其在核准登載、宣播期間，發現內容或登載、宣播方式不當者，原核准機關得廢止或令其修正之。

由於本問題的分類任務只涉及 24 條第一項和第二項，因此我們在設計相關文字輸入的時候只需要考慮這兩條並進行分類。 其次根據模型的提示和數據量的大小（相對較少的訓練數據和測試數據對應著小樣本學習 few shot），我們應該使用大語言模型解決這個問題，因為大語言模型的好處在於推理能力强，採樣效率高，因為模型參數大，能儲存很多的知識，在相同的較少的訓練數據背景下，大模型可以捕捉到更多的特徵建立更大的參數量，使結果文字分類有更好的準確率。

在最近 chatgpt 語言模型的推出以及本次的模型建議下，我們小組也選擇了調用 chatgpt 的 api 解決這個問題，具體原因還在於 chatgpt 這個大語言模型本身就是通過極大量的數據訓練出來的 openai 系統，因此其數據儲備對於進行這種文字分類有著天然優勢。 因而，本問題的難點在於 gpt 的 api 調用和 prompt 的設計，對於 api 的程式調用設計我們結合了 openai 的相關程式碼介紹和網路資源提示先成功地完成了程式碼框架，我們只需要修改相關的 prompt 就可以得到答案。

```
import openai

if __name__ == '__main__':
    openai.api_key = "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"
    s1 = "請閱讀法規，判斷該廣告是否是不實廣告，並針對每一句違規語句說明違反哪一條法規。"
    s2 = "法規：第二十四條第一項 化粧品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播猥褻、有
    s3 = "廣告：又到了充滿聖誕歡樂氣氛的12月，今年不想在Party上成為乏人問津的派對壁花？就讓時尚達人benefit來助你一臂

    response = openai.ChatCompletion.create(
        model = "gpt-3.5-turbo",
        temperature = 0.2,
        max_tokens = 1000,
        messages = [
            {"role": "user", "content": s1+s2+s3}
        ]
    )

    raw_response = response['choices'][0]['message']['content']
    print(raw_response)
```

擁有了基礎的程式碼框架，我們進一步對 prompt 進行了修改，有幾個具體的想法：

（1） 只使用兩個法律條文進行提示，讓模型直接對測試集進行文字分類;

（2） 只使用兩個法律條文進行提示，但是將分類任務轉化成 0、1 的形式，判斷是否違反其中的一個法律；

（3） 只使用一個法律條文進行提示，將分類任務轉化成 0、1 的形式，判斷是否違反其中的該法律；

（4） 使用全部的訓練集，首先訓練模型，其次讓 chatgpt 模型根據自己的資料儲備和訓練的情況做出分類預測；

（5） 模型融合：將所有有成績分數的結果按照分數進行投票將結果準確率提升一個檔次。

以上就是我們最初設想的全部 prompting 設計，這也是本次實驗中的關鍵，其中有分數高的，也有具體實際操作起來有困難的，下麵我將在第三大點中結合程式碼解釋我們的做法並展示最終的準確率分數。

三、程式解釋與得分

具體程式的實現過程我將分為 7 個部分進行講解，分別是生成結果的基礎框架、上述五種想法的具體實現和最終選擇的管道。

1、基礎框架

本問題的基礎框架設計起來較為簡單，由於測試集、訓練集以及輸出的結果都是 csv 的檔案形式，回此我們讀取數據和綜合數據都採用 pandas 這個工具庫即可，通過 readcsv、to_csv 以及 DataFrame 的數據格式實現文字的讀入和數據的輸出。 具體操作的程式碼截圖見下：

```python
if __name__ == '__main__':
    train_FileName = "COS_train.csv"
    test_FileName = "COS_test.csv"
    train_text = pd.read_csv(train_FileName, encoding='utf-8')
    test_text = pd.read_csv(test_FileName, encoding='utf-8')
    text_data = test_text['sentence']
    result = []
    #train_gpt(train_text)
    for i in range(text_data.shape[0]):
        print(gpt_classification(text_data[i]))
        result.append(gpt_classification(text_data[i]))
    df = pd.DataFrame({"source_id": test_text['source_id'],
                       "label_for_kaggle": result
                       })
    df.to_csv("submission.csv", index=False, encoding='utf-8')
```

Chatgpt 實現分類部分，我們將不同的 prompt 進行組合輸入（s1+s2+s3+s4）到 gpt 的對話中，而且通過對其輸出格式的嚴格限制實現最後的 0 和 1 的數值獲取（類似於 s4 =“回答形式僅限於 0 和 1 這兩個數位中的一個，其中 0 代表違反第二十四條第一項，1 代表不違反第二十四條第一項”）。 同時由於 chatgpt 調用 api 的時候嚴格限制一分鐘內只能回答三次問題，回此我們加入了 sleep（20）實現程式碼的不間斷生成答案，這部分的設計結果如下：

```python
def gpt_classification(data):
    sleep(20)
    s1 = "請閱讀法規判斷該廣告違反哪一條法規。\n"
    #s1 = "請閱讀法規，根據之前訓練的分類模型判斷該廣告違反哪一條法規。\n"
    s2 = "法規：第二十四條第一項 化粧品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播
    #s2 = "法規：第二十四條第一項 化粧品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣
    s3 = "廣告：" + data + "\n"
    s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表違反第二十四條第一項，1代表不違反第二十四條第一項"

    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        temperature=0.2,
        max_tokens=1000,
        messages=[
            {"role": "user", "content": s1 + s2 + s3 + s4}
        ]
    )
    raw_response = response['choices'][0]['message']['content']
    return int(raw_response)
```

2、思路一

思路一的實現管道為只使用兩個法律條文進行提示，讓模型直接對測試集進行文字分類。 具體就是讓 s2 設定為 “法規：第二十四條第一項化妝品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播猥褻、有傷風化或虛偽誇大之廣告。第二十四條第二項化妝品之廠商登載或宣播廣告時，應於事前將所有文字、畫面或言詞，申請中央或直轄市衛生主管機關核准，並向傳播機构繳驗核准 之證明檔案。 \n“；將 s4 設定為”回答形式僅限於 0 和 1

這兩個數位中的一個，其中 0 代表違反第二十四條第一項， 1 代表違反第二十四條第二項“。這是最直接且最容易想到的方法，且我們直接將測試集進行預測，回此只調用了 gpt_classification 這一預測函數。最終的結果為 0.47619，通過了 strongline 但是沒有過 bossline，說明結果有待提升，這也給我們了其他想法的展示機會。

```
#s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表違反第二十四條第一項，1代表不違反第二十四條第一項"
s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表違反第二十四條第一項，1代表違反第二十四條第而項"
#s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表不違反第二十四條第二項，1代表違反第二十四條第二項"
```

submission.csv
Complete · eugene.chen · 17d ago · original method
0.47619

3、 思路二

思路二的實現管道為只使用兩個法律條文進行提示，但是將分類任務轉化成 0、1 的形式，判斷是否違反其中的一個法律。 具體就是讓 s2 設定為 “法規：第二十四條第一項化妝品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播猥褻、有傷風化或虛偽誇大之廣告。第二十四條第二項化妝品之廠商登載或宣播廣告時，應於事前將所有文字、畫面或言詞，申請中央或直轄市衛生主管機關核准，並向傳播機构繳驗核准 之證明檔案。 \n“；將 s4 設定為”回答形式僅限於 0 和 1 這兩個數位中的一個，其中 0 代表違反第二十四條第一項，1 代表不違反第二十四條第一項“。

這是在第一種思路下進行改進的方法，具體原因為文字分類任務其實輸入的文字越多越複雜則更加影響模型對最終結果的判斷，我們想通過將其改為二分類減少輸入的數據量，同時將分類任務改為 0、1 分類更是簡化了任務複雜度，最終結果會有很大的提升。 我們同樣直接將測試集進行預測，回此只調用了 gpt_classification 這一預測函數。 初步測試的最終結果為 0.49，同樣通過了 strongline 但是沒有過 bossline，說明結果有待提升，但是我們沒有放棄這個思路，進行了多組嘗試。

```
s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表違反第二十四條第一項，1代表不違反第二十四條第一項"
#s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表違反第二十四條第一項，1代表違反第二十四條第而項"
#s4 = "回答形式僅限於0和1這兩個數位中的一個，其中0代表不違反第二十四條第二項，1代表違反第二十四條第二項"
```

因為這個思路相比於上一個思路還是有很大提升的，回此我們在這個基礎上修改並測試了很多組 prompting，但都是比較小的修改，例如將上述的條文主角改為第二條，將 prompting 翻譯成英文或者其他語言再轉化為中文，我們講這些

結果都輸入到 kaggle 中進行了分數獲取， 但是不盡如人意甚至還出現了 0.3 多的準確率，這再給我們很大打擊的同時也讓我們意識到了其實 chatgpt 在沒有具體的特別大的訓練數據的背景下預測試相對來說不穩定的，其能够從 0.49 降低到 0.3 多，那麼是否有可能從 0.49 昇到 bossline 以上呢？ 我們抱著這樣的想法改回了原來的 prompt，不斷反復測試，終於在最後收穫了一個比較高的，超過 bossline 的分數：0.57983。

我們在這之後嘗試了其他的方法和模型融合，但是最終都沒有超過這個分數，回此我們最後提交的程式碼和結果也都是這次的結果。

4、思路三

思路三的實現管道為只使用一個法律條文進行提示，將分類任務轉化成 0、1 的形式，判斷是否違反其中的該法律。 其與思路二極其相似， 具體修改部分就是將 s2 改為 “法規：第二十四條第一項化妝品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播猥褻、有傷風化或虛偽誇大之廣告。第二十四條第二項化妝品之廠商登載或宣播廣告時，應於事前將所有文字、畫面或言詞，申請中央或直轄市衛生主管機關核准，並向傳播機构繳驗核准之證明檔案。 “只有一條條文的判斷，最後的 s4 設計也與上一個類似，採用 0、1 分佈，最終結果為 0.45454，在我們根據思路二預料得到的情理之中。

```
s2 = "法規: 第二十四條第一項 化粧品不得於報紙、刊物、傳
#s2 = "法規: 第二十四條第一項 化粧品不得於報紙、刊物、
s3 = "廣告: " + data + "\n"
```

5、思路四

思路四實際上是我們投入嘗試時間最多的一個方法，其具體想法為使用全部的訓練集，首先訓練模型，其次讓 chatgpt 模型根據自己的資料儲備和訓練的情況做出分類預測。 我們在網路上蒐集資料查詢到，chatgpt 在調用 api 的時候的每一句對話其實是獨立的，也就是如果每一次都調用 api 雖然其 token 很少，輸出較快且準確，但是沒有考慮到上下文，於是我們搜尋了一種方法，其將所有

的輸出記錄在一個大類中並形成 list 下 dict 的形式，分為 gpt 和 consumer 的角色進行每一次回答的記錄， 並不斷擴大這個字典，每一次調用 api 的時候將整個字典輸入進去，這就相當於讓 gpt 瞭解了之前其做的回答，擁有了上下文，這個類的設計如下：

```
class Chat:
    def __init__(self, conversation_list=[]) -> None:
        # 初始化对话列表，可以加入一个key为system的字典，有助于形成更加个性化的回答
        # self.conversation_list = [{'role':'system','content':'你是一个非常友善的助手'}]
        self.conversation_list = []


    def ask(self, prompt):
        self.conversation_list.append({"role": "user", "content": prompt})
        response = openai.ChatCompletion.create(model="gpt-3.5-turbo", messages=self.conversation_list)
        answer = response.choices[0].message['content']
        self.conversation_list.append({"role": "assistant", "content": answer})
        self.show_conversation(self.conversation_list)
```

conversation_list 就是用於記錄所有回答的清單，ask 函數在獲取相關回答後會將相關結果加入到 list 中。 擁有這樣的框架，我們就可以設計訓練集的應用函數，我們的想法是將訓練集按照測試集的形式先讓 gpt 進行預測，並根據 gpt 預測的結果和實際的結果進行比對，如果相同則給 gpt 進行獎勵，如果不同則把正確答案告訴 gpt，讓其修改模型，這種思想有點類似於深度學習中强化學習的思想。 具體的程式碼實現如下：

```
def train_gpt(train_text):
    i = 0
    for e in range(train_text.shape[0]):
        s = ""
        if i == 0:
            s += "現在需要你完成一個廣告法律違規的分類問題的模型，就是通過訓練數據將模型訓練好並應用在測試集上，得到最後的分類結果。 問
        i += 1
        s1 = "以下是第"+str(i)+"條訓練數據\n"
        s2 = "法規: 第二十四條第一項 化粧品不得於報紙、刊物、傳單、廣播、幻燈片、電影、電視及其他傳播工具登載或宣播猥褻、有傷風化或虛偽
        s3 = "廣告: " + train_text['sentence'][e] + "\n"
        s4 = "請先進行分類判斷,回答形式僅限於0和1這兩個數位中的一個，其中0代表違反第二十四條第一項，1代表違反第二十四條第二項"
        response = openai.ChatCompletion.create(
            model="gpt-3.5-turbo",
            temperature=0.2,
            max_tokens=1000,
            messages=[
                {"role": "user", "content": s + s1 + s2 + s3 + s4}
            ]
        )
```

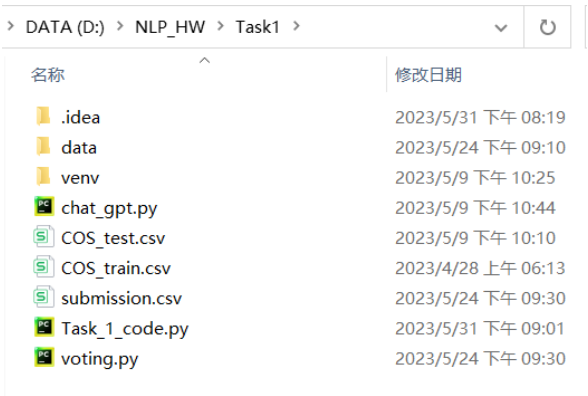
```
raw_response = response['choices'][0]['message']['content']
if int(raw_response) == int(train_text['label_for_kaggle'][e]):
    s6 = "針對本次訓練集數據的預測正確，請進一步保存模型"
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        temperature=0.2,
        max_tokens=1000,
        messages=[
            {"role": "user", "content": s6}
        ]
    )
else:
    s6 = "針對本次訓練集數據的預測錯誤，正確答案為"+str(int(train_text['label_for_kaggle
    response = openai.ChatCompletion.create(
        model="gpt-3.5-turbo",
        temperature=0.2,
        max_tokens=1000,
        messages=[
            {"role": "user", "content": s6}
        ]
    )
```

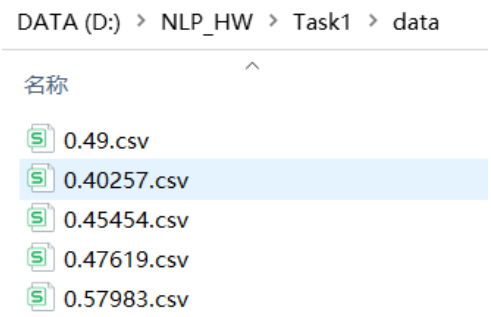
這個 train_gpt 函數就是完整的訓練過程，其 prompt 提示與直接預測測試集的大致相同，不過在其中加入了獎勵的話語。 雖然我們感覺這種方法可以實現並且效率肯定很高，但是我們沒有預料到 gpt 的輸入是有一定 token 的，這樣一次一次的輸入反覆運算會使得 list 中的文字量越來越大，遠遠超過了 gpt 的 api 輸入可能承受的範圍（即使不全部使用訓練集的所有數據，進行抽樣測試也會將最終的 list 輸入擴大到難以想像的地步）， 最終沒有獲得我們想要的結果和答案，但是我們認為這種管道如果在 token 允許的情形下是絕對可以收貨很好的結果的。 我們最後選擇了將編寫的程式碼保留在最後提交的檔案中（雖然主函數沒有調用），這樣能顯示出我們對任務一思考的過程。

6、思路五

思路五的想法就不僅僅是局限於使用大語言模型了，由於本問題是一種分類任務，且最後的結果只有 0 和 1 回此在收集多組數據後我們決定使用一種最簡單的模型融合管道：投票法。 具體思路為將所有有成績分數的結果按照分數進行投票將結果準確率提升一個檔次，當然這是我們的期望。

具體程式碼實現做法如下，我們單獨採用了另一個 code，名稱為 voting.py 並且最後也提交了這個模型融合所需要的程式碼。 我們將所有 kaggle 上得到分數的結果檔案都用分數命名並且放置到相對路徑下的 data 資料夾中，目錄結構見下方圖片。





在 voting.py 程式碼中我們通過 os 系統獲取了 data 下所有檔案的資訊和所得分數，包括 pandas 能够得到的所有資訊。

```python
def getfiles():
    filenames = os.listdir('D:\\NLP_HW\\Task1\\data')
    scorelist = []
    filelist = []
    for name in filenames:
        if '.csv' in name:
            filelist.append(name)
            scorelist.append(float(name.replace('.csv', '')))
    return filelist, scorelist
```

其次，通過對每一個 id 中的結果 0 或 1 按照所有分數加起來最高的進行投票，決定最後輸出的是哪個數值（0、1），最後再將整個結果按照順序輸出，生成 submission.csv 提交，最終結果為 0.47665。

```python
if __name__ == '__main__':
    files, scores = getfiles()
    print(files, scores)
    picks = []
    labels = []
    temp0 = './data/'+str(files[0])
    ids = pd.read_csv(temp0)['source_id']
    print(ids)
    for fname in files:
        temp = './data/' + str(fname)
        picks.append(pd.read_csv(temp)['label_for_kaggle'])
    for id in range(100):
        num = []
        for i in range(2):
            num.append(0)
        for i in range(len(picks)):  # 对于id号第i个模型的结果
            label = picks[i][id]
            num[label] += scores[i]
        labels.append(num.index(max(num)))

    with open('submission.csv', 'w') as f:
        f.write('source_id,label_for_kaggle\n')
        for i in range(100):
            f.write(str(ids[i]) + ',' + str(labels[i]) + '\n')
```

這個分數不理想，我們推測的可能原因為 chatgpt 模型不太穩定，回此 data 中收穫的原始數据集並不具有代表性，其做投票沒有什麼實質性的提升，且就算有幾個分數很高有說服力也會被下麵更多的低分拉下平均值，其次就是這種大語言模型不適合做和基礎模型的模型融合，容易導致分數不昇反降。 雖然這個模型融合的結果並不理想，但是我們也講思考的過程記錄下來，和相關程式碼一同提交。

7、最終方法

我們最後是按照最高分數對應的方法提交的結果檔案和程式碼檔案，但是程式碼也包括了我們其他思路的說明（在注釋部分），如果運行 Task_ 1_ code.py，按照常理會得到我們思路二期望的結果，總的來說我們的最終方法就是思路二：只使用兩個法律條文進行提示，但是將分類任務轉化成 0、1 的形式，判斷是否違反其中的一個法律。 最後收穫了 0.57983 的分數，超過了 bossline，在 public 數據測試上獲得了滿分。 （希望最後的 private 上也要收穫高分呀！）

四、總結

總的來說我們完成這次 NLP 課程大工作的任務一文字分類還是花費了一定的時間的，最後的結果感覺也對得起我們的付出，chatgpt 在 2022 年年末進行推廣並得到了廣泛的使用，我們這次的任務充分讓我們認識到了使用 chatgpt 方法的多元性，也認識到了其能力的高超， 雖然有一些遺憾在於我們抱有期望最高的方法由於上述介紹的原因沒有進一步實現，但是這也給我們之後的學習和對 nlp 的深入研究鼓足了動力，更進一步！