好的，我將開始分析您提供的專案文件，以逐步解釋該研究所執行的所有實驗步驟。根據您的指示，我會特別詳細地闡述 GOP 計算流程背後的原理，並嚴格遵守理論上的計算順序。

觀察與分析：

我正在檢視 Evaluating Logit-Based GOP Scores for Mispronunciation Detection_v2.pdf 論文、myImplementation/Src2/python/logit_based_gop_score_analyzer.py 核心腳本、official/SO/speechocean_quantification.py、official/MPC/mpc_ctc_segment_quantification.py 等數據處理腳本以及相關的說明文件。

根據這些文件，我歸納出整個實驗的核心流程。這是一個多階段的過程，從數據準備開始，通過聲學模型進行特徵提取，然後進行強制對齊，最後計算並評估多種 GOP 分數。

以下是我根據分析整理出的實驗詳細步驟與流程：

實驗步驟分析

第一階段：環境設定與模型載入

1. 載入預訓練模型與處理器：
   實驗的第一步是載入一個預訓練好的語音辨識模型。研究中使用的是 facebook/wav2vec2-xlsr-53-espeak-cv-ft 模型。這個模型是一個強大的聲學模型，能夠將音訊波形轉換成一系列高維度的聲學特徵，稱為 logits。同時，也會載入對應的 Wav2Vec2Processor，它包含了對音訊進行預處理（例如重採樣到 16kHz）和文本進行 token 化（將音素轉換為模型認識的數字 ID）所需的所有工具。

第二階段：數據準備與特徵提取

2. 音訊與文本輸入：
   實驗的輸入包含兩個部分：一個是音訊檔案（例如 .wav），另一個是與該音訊內容對應的音素轉錄文本（Transcript）。這個文本是由音素符號（例如 ARPAbet）組成的，並用空格隔開。
3. 音訊預處理：
   讀取音訊檔案後，使用 Wav2Vec2Processor 將其波形轉換為模型可

接受的格式。這通常包括將音訊的採樣率統一轉換為 16kHz，因為這是模型訓練時所使用的採樣率。

4. 模型推論以獲取 Logits：
   將預處理過的音訊數據輸入到 Wav2Vec2 模型中。模型會進行推論（Inference），輸出一系列的 logits。logits 是一個二維矩陣，其維度為 (序列長度, 詞彙量大小)。每一行代表一個音訊幀（通常約 20 毫秒），每一列則對應模型詞彙表中一個 token（在這裡是音素）的未正規化分數。這個 logits 矩陣是後續所有 GOP 計算的基礎。

第三階段：強制對齊 **(Forced Alignment)**

5. 使用 CTC Segmentation 進行音素對齊：
   這是實驗流程中至關重要的一步。單純的 logits 矩陣只告訴我們每一幀「可能」是什麼音素，但我們需要知道文本中每一個音素確切對應到音訊中的哪一段時間。這一步利用 ctc-segmentation 這個工具庫來實現「強制對齊」。它會拿著 logits 矩陣和完整的音素轉錄文本，計算出文本中每一個音素最有可能的開始和結束時間點（以幀為單位）。
   - 比喻：這就像你有一段影片和一份劇本，強制對齊就像是為影片配上精確時間戳的字幕，告訴你每一句台詞是在影片的第幾秒到第幾秒之間說出的。沒有這個步驟，後續的評分就會失去焦點。

第四階段：**GOP** 分數計算

對於每一個透過強制對齊所定位出的音素片段（例如，音素 /p/ 對應第 10 幀到第 15 幀），實驗會提取這一段的 logits（稱為 logits_slice）以及目標音素的 ID，然後依序計算以下五種 GOP 分數。

6. 計算傳統 **GOP** 分數 **(GOP_DNN)**：
   - 流程：首先，對 logits_slice 的每一幀進行 softmax 運算，將其轉換為機率分佈。然後，提取出每一幀對應「目標音素」的機率值，並計算這些機率值的平均數。最後，取該平均機率的負對數（-log）作為最終分數。
   - 為何這麼做？：這是最傳統的 GOP 計算方式。其核心思想是：如果發音是標準的，那麼在該音素的時間段內，模型辨識出這個音素的平均機率應該會很高。取負對數是為了讓分數的解讀更直觀：一個很高的機率（接近 1）取 -log 後會得到一個很低的分數（接

近 0)，代表「好的發音」；反之，一個很低的機率取 -log 後會得到一個很高的分數，代表「差的發音」。
    - 比喻：這就像一位評審在音素持續的時間內，每一瞬間都在打分數（0 到 1 的機率），GOP_DNN 就是這位評審給出的「平均分」的負對數。分數越低，代表平均評價越高。
7. 計算最大 **Logit** 分數 **(GOP_MaxLogit)**：
    - 流程：在 logits_slice 中，只看屬於「目標音素」的那一欄的 logit 值，然後找出其中的最大值作為分數。
    - 為何這麼做？：這個方法認為，即使整個音素發音過程中有些波動，但只要模型在某個瞬間對這個音素表現出極高的信心（即 logit 值非常高），就足以證明發音是好的。它能捕捉到發音的「峰值信心度」，並且可以避免 softmax 函數可能因過於自信而壓縮分數差異的問題。與 GOP_DNN 相反，這裡的分數是越高越好。
    - 比喻：這就像跳高比賽，我們只關心選手跳出的「最高高度」（最大 logit），而不是他每次試跳的平均高度。跳得越高，成績越好。
8. 計算 **Logit** 邊距分數 **(GOP_Margin)**：
    - 流程：對於 logits_slice 中的每一幀，找出「目標音素」的 logit 值，再找出所有「其他競爭音素」中最高的 logit 值。將兩者相減，得到一個「邊距（Margin）」。最後，計算整個音素片段內所有幀的平均邊距作為分數。
    - 為何這麼做？：這個方法不僅關心模型對目標音素的信心有多高，更關心它與最容易混淆的音素之間的「區分度」。如果目標音素的 logit 遠高於所有其他競爭者，那麼這個邊距就很大，代表模型能清晰地辨認出這個音素，發音品質可能很高。這個分數也是越高越好。
    - 比喻：這就像一場選舉，GOP_Margin 不只看冠軍候選人（目標音素）拿了多少票，更關心他「領先第二名多少票」。領先的票數越多，贏得越是毫無懸念。
9. 計算 **Logit** 變異數分數 **(GOP_VarLogit)**：
    - 流程：在 logits_slice 中，同樣只看屬於「目標音素」的那一欄的 logit 值，然後計算這些值的「變異數（Variance）」。
    - 為何這麼做？：這個指標衡量的是模型在識別一個音素的過程中，其信心的「穩定性」。如果發音清晰穩定，模型在音素持續的每一幀給出的 logit 值應該都比較一致，因此變異數會很小。如果發

音含糊或有雜音，模型的判斷可能會來回搖擺，導致 logit 值起伏很大，變異數也隨之增大。高變異數通常對應較差的發音。
    - 比喻：這就像測量一位外科醫生的手在手術過程中的「穩定度」。變異數小，代表手非常穩，手術品質高；變異數大，則代表手在顫抖，存在風險。
10. 計算組合分數 **(GOP_Combined)**：
    - 流程：這是一個混合分數，它將 GOP_Margin 和 GOP_DNN 透過一個權重參數 alpha 結合起來。公式為：alpha * GOP_Margin - (1 - alpha) * GOP_DNN。
    - 為何這麼做？：研究者認為單一指標可能各有優劣，GOP_Margin 關注區分度，而 GOP_DNN 關注平均機率。將兩者結合，希望能綜合兩者的優點，創造一個更魯棒的評分。
    - 比喻：這就像學生的總成績，不僅看他的考試平均分 (GOP_DNN)，還要參考他在難題上的解題能力（與其他人的差距，GOP_Margin），最後加權計算出一個綜合評價。

第五階段：評估與分析

11. 性能評估：
    實驗最後階段，是使用上述計算出的 GOP 分數來進行發音錯誤檢測的性能評估。腳本 MPC_eval/mpc_evaluate.ipynb 和 SO_eval/so_evaluation.ipynb 展示了這個過程。
    - 分類性能：透過設定一個閾值（通常是最大化 MCC 分數的點），將 GOP 分數分為「正確」和「錯誤」兩類，然後計算準確率、精確率、召回率、F1-score 和 MCC 等指標。
    - 與人類評分員的相關性：對於有真人專家評分的數據集（如 SpeechOcean762），會計算 GOP 分數與人類評分之間的皮爾森相關係數（PCC）和均方誤差（MSE），以評估分數是否與人類的感知一致。
    - 視覺化分析：使用小提琴圖（Violin Plot）來視覺化比較不同 GOP 分數在正確與錯誤發音樣本上的分佈差異，判斷其區分能力。

以上便是我根據您提供的文件，對整個實驗流程的詳細分析。我已遵循您的所有指示，特別是對 GOP 的計算進行了詳細的闡釋。目前仍在 **RESEARCH** 模式，等待您的下一步指示。