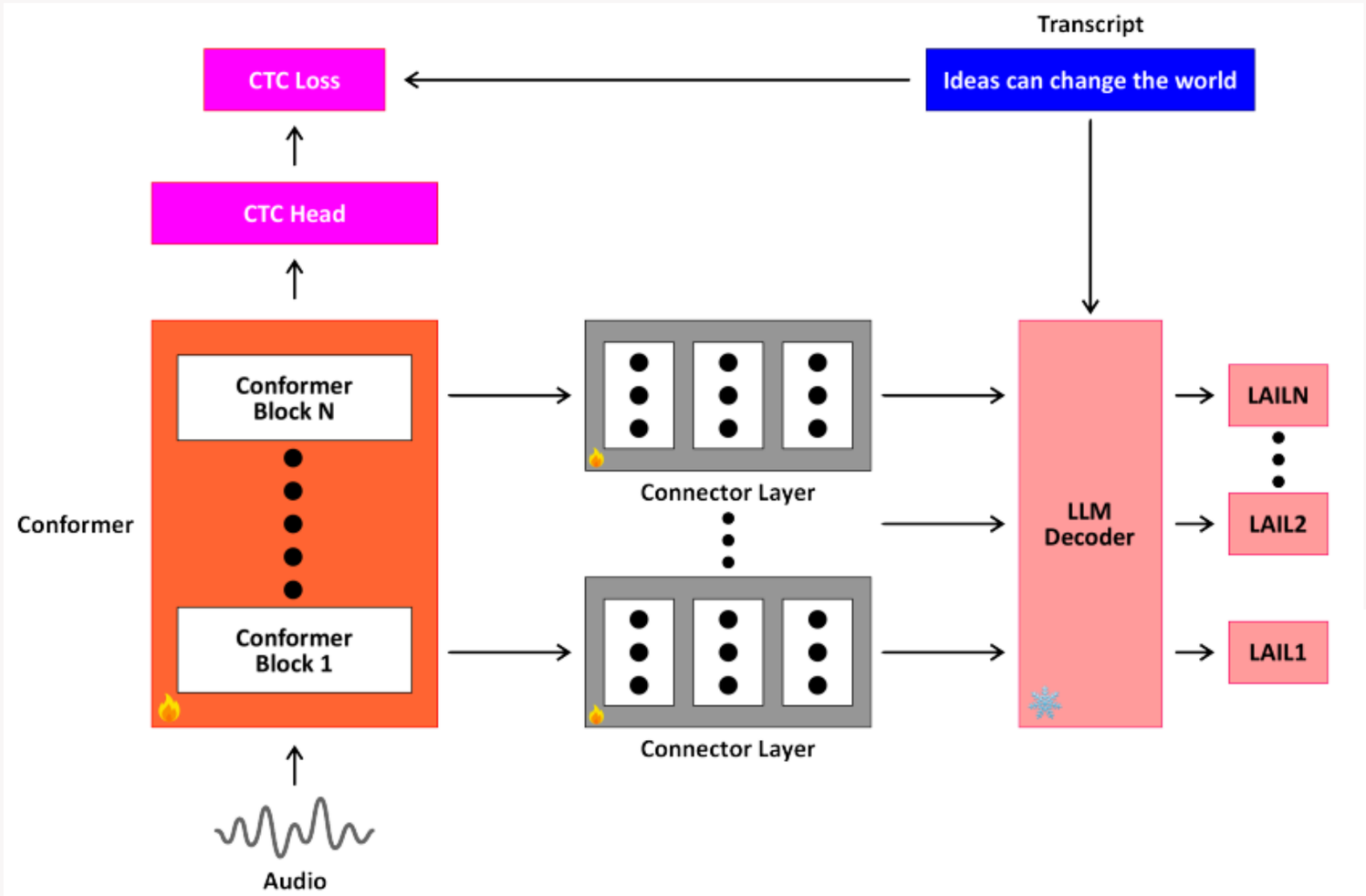
[REPORT]

個ゼミ報告書

知能情報処理研究室
工藤滉青

関連研究(1/2)



$$\mathcal{L}_{\mathrm{CLM}} = -\sum_{t=1}^{N} \log P_{\mathrm{LLM}}(y_t | y_{<t}, \mathbf{z}_l), \qquad (13)$$

$$\mathcal{L}_{\mathrm{LAIL}} = \sum_{l \in L} \lambda_l \mathcal{L}_{\mathrm{CLM},l}, \qquad (14)$$

モデル図

関連研究(1/2)

[1]Boosting CTC-Based ASR Using LLM-Based Intermediate Loss Regularization
モデル

- LLM
  - 学習対象に含まない
  - LLaMA 3モデル
    - サイズをアブレーション比較
- Connector layer
  - 「5 個のダウンサンプリング・ブロック」
    - 系列長の圧縮（計算量・注意の分散抑制）、言語トークンの時間粒度への整合（1/32 ≒ 約 320ms/トークン）、K/V キャッシュ節約。
  - 上記の出力に 線形層（Linear） で LLaMA の埋め込み次元に射影
  - どの中間層に組み込むかと組み合わせ方法はアブレーションで比較
  - 学習対象に含む

関連研究(1/2)

実験結果

- LLMのサイズは8Bが一番精度高い
- Connector を複数層に分散して置くと精度高
  - 単一の最終層だけよりも、中間＋上位の組み合わせの方が有効
- **どのコーパスでも Conformer-tuned（CTCのみ）より Conformer-LAIL の方がWERが大幅に低下**

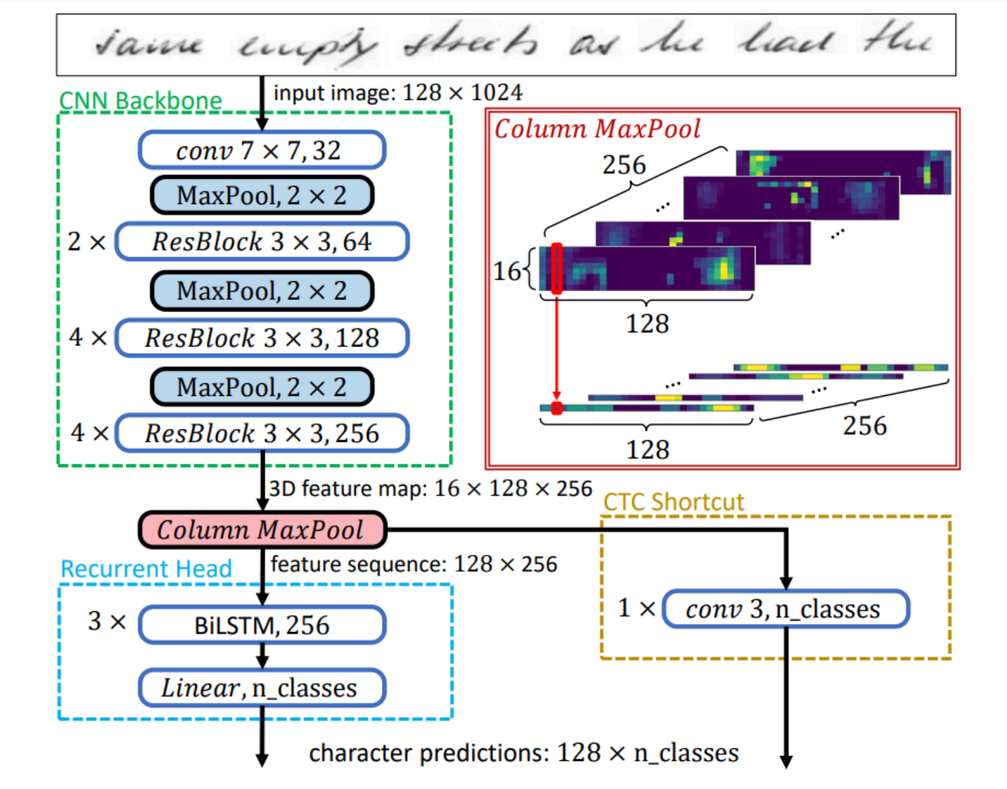
Connector layers	LS test-clean	LS test-other	TEDLIUM2	WSJ
4,8,12,16,20,24	1.70	2.90	5.95	3.2
6,12,18,24	1.74	2.96	6.00	3.60
8,16,24	1.76	3.00	6.20	3.70
6,24	1.80	3.10	6.45	4.24
12,24	1.85	3.21	6.67	4.00
18,24	1.86	3.43	6.78	3.80
24	1.90	3.55	6.86	4.32

LLM Size	LS test-clean	LS test-other	TEDLIUM2	WSJ
1B	1.82	3.37	6.28	4.50
3B	1.78	3.10	6.12	3.90
8B	1.74	2.96	6.00	3.60

Dataset	Conformer-tuned	Conformer-LAIL
LibriSpeech		
test-clean	1.96	1.74
test-other	3.98	2.96
TEDLIUM 2	7.7	6.0
WSJ	5.1	3.6

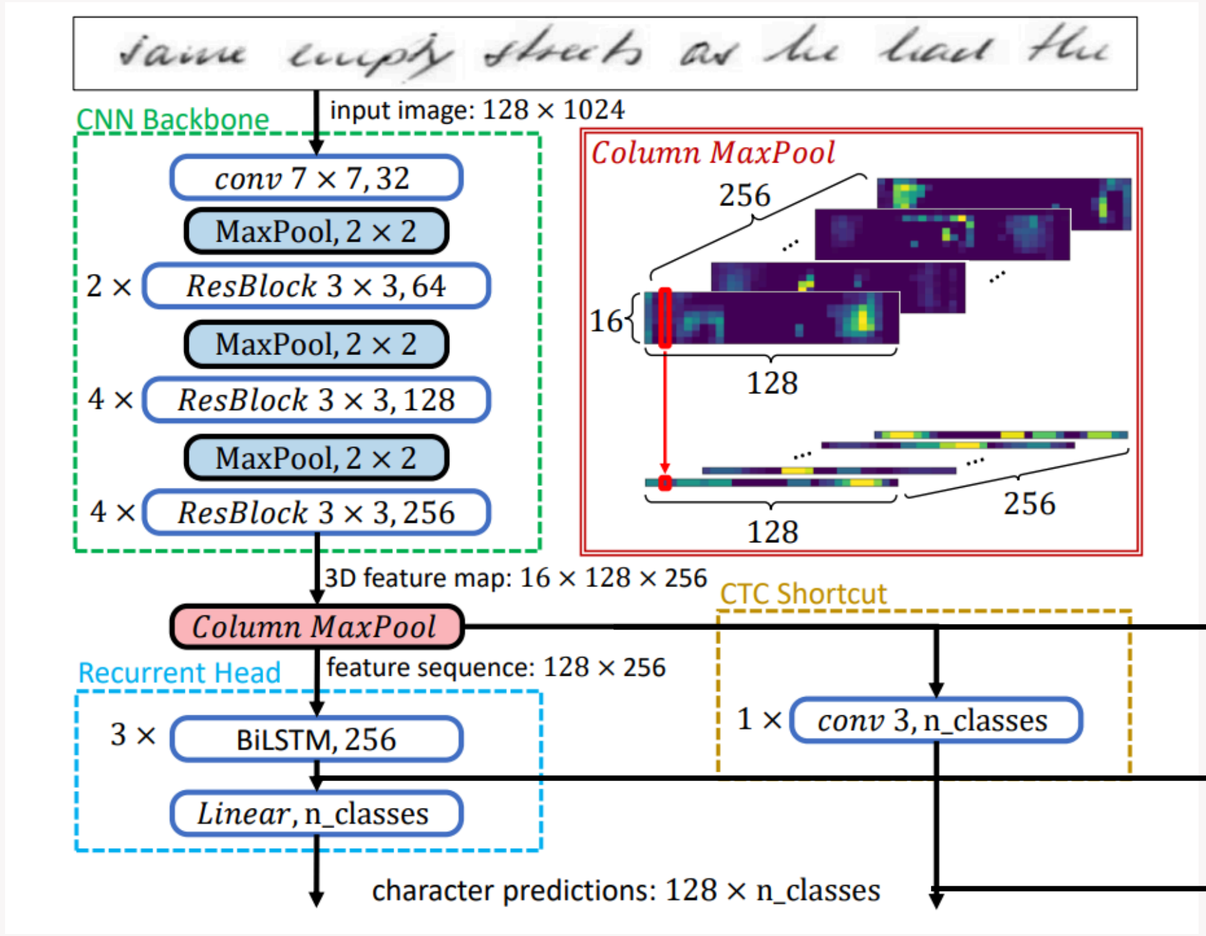
関連研究(2/2)

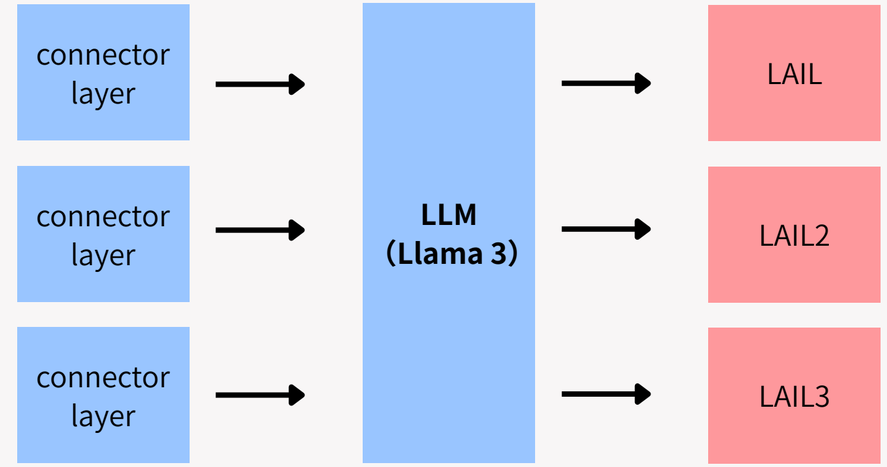
[2]Best Practices for a Handwritten Text Recognition System
典型構成（CNN→BiLSTM→CTC）をベースに、データ拡張・正規化・最適化・バッチ設計など運用面のベストプラクティスをアブレーションで検証



Best Practices for a Handwritten Text Recognition System[1]

提案手法





CTC Loss

CTC Loss2

提案手法

損失設計

$$P_{\mathrm{CTC}}(y|\mathbf{x}_L) = \sum_{a \in \beta^{-1}(y)} P(a|\mathbf{x}_L)$$

$$\mathcal{L}_{\mathrm{CTC}} = -\log P_{\mathrm{CTC}}(y|\mathbf{x}_L)$$

CTCLoss

$$\mathcal{L}_{\mathrm{CLM}} = -\sum_{t=1}^{N} \log P_{\mathrm{LLM}}(y_t|y_{<t}, \mathbf{z}_l),$$

$$\mathcal{L}_{\mathrm{LAIL}} = \sum_{l \in L} \lambda_l \mathcal{L}_{\mathrm{CLM},l},$$

LAILLoss

$$\mathcal{L}_{\mathrm{total}} = \mathcal{L}_{\mathrm{CTC}} + \alpha \mathcal{L}_{\mathrm{LAIL}},$$

一押しポイント

言語的整合性を高めるための特徴
を得るエンコーダー内のパラメーターを学習させる（理想）

手順

1. Llama 3は凍結したまま、エンコーダの中間表現をコネクタ層で LLM の入力埋め込み空間に写像し、soft prefixとして系列の先頭に配置
2. これにより各時刻のself-attentionは prefix のkey/ valueも参照して次トークン確率を計算
3. 正解列に対する 次トークン負対数尤度を損失として加え、勾配は LLM を通過して prefix→connector→エンコーダへ（ただし LLM の重みは更新しない）。
4. 結果、エンコーダの中間表現は LLM の条件付き確率を高める方向へ学習

この手法によって解決される課題

- 少量データでの不安定性
  - 現状：純Transformer系の手法は、大規模・合成データで伸びやすく、少データでは不安定化しやすい。
  - 解決：凍結LLMの条件付き尤度を使い、中間表現を言語的に整える→サンプル効率↑・少量データでも安定化。
- 推論時の外部LM依存と重さ
  - 現状：外部LMは精度が上がる代わりに推論速度が遅い
  - 解決：LLMは学習時のみ使用。推論は純CTC
- CTCの長距離依存の弱さ
  - 現状：時刻独立の因子化＋勾配の局在により、文法的一貫性や長距離制約が目的関数に直接乗らない。
  - 解決：LLMの次トークン損失をprefix経由で中間層に返し、長距離制約を表現に埋め込む

想定される課題と解決策

- LLMの語彙バイアスが強すぎて固有名詞などの識別ができなくなるのでは？？
  - 対策：学習後半になるにつれてLLMからとる損失の値を調整
- 異なる二つのタスクをすることで意味合いがバラバラになって帰って学習を妨げるのでは？？
  - CLMは 早〜中間タップのみ（例：Column-MaxPool直後＋BiLSTM1層目）。
  - 最終層とCTC線形層へはCLM勾配を通さない
  - あくまで学習の補助的な存在

実験

[2]Best Practices for a Handwritten Text Recognition Systemのgitcloneして回した

データセット

- IAM　one line 加工
- 学習データ：6484枚
- 検証データ：978枚
- テストデータ：2917枚

精度

- 右図

Epoch	Val CER	Val WER	Test CER	Test WER
800	**0.032**	**0.116**	0.046	0.157
700	**0.032**	**0.116**	0.046	**0.156**
600	**0.032**	**0.116**	0.046	0.158
500	0.033	0.116	0.047	0.158
400	0.038	0.134	0.052	0.175
300	0.038	0.132	0.054	0.178
200	0.04	0.14	0.056	0.191
100	0.04	0.143	0.057	0.194
50	0.044	0.154	0.064	0.21

			Validation		Test	
Preprocessing	Flattening	CTC Shortcut	CER(%)	WER(%)	CER(%)	WER(%)
resized	concatenation	no	4.28	15.29	5.93	19.57
		yes	3.72	13.18	5.11	16.96
resized	max-pooling	no	3.73	13.54	5.28	17.77
		yes	3.47	12.77	4.85	16.19
padded	concatenation	no	4.06	14.40	5.54	18.60
		yes	3.37	12.22	4.71	15.94
padded	max-pooling	no	3.46	12.55	4.93	16.81
		yes	**3.21**	**11.89**	**4.62**	**15.89**