PLOS ONE

RESEARCH ARTICLE

A syllable-character collaborative model for enhanced Pinyin and Chinese recognition

Zeyuan Chen[1], **Cheng Zhong**[1,2], **Danyang Chen**[1,2]*

1 School of Computer,Electronics and Information, Guangxi University, Nanning, Guangxi, China, **2** Key Laboratory of Parallel, Distributed and Intelligent Computing in Guangxi Universities and Colleges, Guangxi University, Nanning, Guangxi, China

* chendanyang@gxu.edu.cn

Citation: Chen Z, Zhong C, Chen D (2025) A syllable-character collaborative model for enhanced Pinyin and Chinese recognition. PLoS One 20(7): e0325045. https://doi.org/10.1371/journal.pone.0325045

Editor: Zeheng Wang, Commonwealth Scientific and Industrial Research Organisation, AUSTRALIA

Received: January 23, 2025

Accepted: May 6, 2025

Published: July 7, 2025

Data availability statement: The dataset used in our study is the AISHELL-1 corpus, which is publicly available at: https://www.aishelltech.com/kysjcp. The source code used to implement and evaluate the proposed model has been made publicly available on GitHub at: https://github.com/chen-ze-yuan/SCCM.

Abstract

In Chinese speech recognition, end-to-end speech recognition models usually use Chinese characters as direct output and perform poorly compared with other language models. The main reason for this phenomenon is that the relationship between Chinese text and pronunciation is more complex. Inspired by the learning process of Chinese beginners, who first master initials, finals, and pinyin before learning characters, we propose the Syllable-Character Collaborative Model (SCCM), which incorporates these phonetic elements into the training process. Additionally, we design a Pinyin-Ensemble module that employs an ensemble learning approach to reduce pinyin recognition errors, which in turn leads to a reduction in text recognition errors. Experiments on AISHELL-1 show that our approach not only reduces pinyin and character error rates compared to a prior end-to-end method using pinyin as auxiliary information, but also achieves a 45.7% relative reduction in Character Error Rate (CER) over the AISHELL-1 baseline.

Introduction

Automatic Speech Recognition (ASR) [1] is the process of converting human speech into corresponding text sequences. Most languages, such as those using Latin or Greek alphabets, are phonetic scripts, meaning their written characters relatively correspond to their pronunciation [2,3]. However, Chinese is more closely tied to the meaning of the characters rather than their pronunciation [4]. This fundamental difference makes the mapping between speech and characters more complex, posing greater challenges for speech recognition in Chinese compared to other languages [5,6].

Chinese uses pinyin as a phonetic notation system, where each character is transcribed into a syllable composed of an initial and a final. For example, the character "中" (zhong) has the initial 'zh' and the final 'ong'. Chinese language learners typically begin by mastering initials and finals, which they then combine into pinyin syllables to understand pronunciation rules and establish connections between pronunciation and Chinese characters.

In traditional speech recognition [7–9] systems, researchers use acoustic models to convert audio into corresponding phonemes through a series of independent modules. These

Funding: This study was funded by the Natural Science Foundation of Guangxi (2025GXNSFAA069540 to D.C.) and the Research Capacity Improvement Project of Young Researchers (2024KY0017 to D.C.).

Competing interests: The authors have declared that no competing interests exist.

phonemes are then combined with a pronunciation dictionary and language model to produce the most likely text sequence. Traditional ASR models have excellent capabilities in modeling language features and perform well across various languages. However, the independent optimization of disparate modules leads to complexity in system development, as it requires meticulous alignment of individual component objectives with the overall system goals, which is challenging and does not fully exploit the complementary strengths of the components.

In recent years, end-to-end speech recognition [10–12] has gained widespread attention and achieved great success across many languages. This approach simplifies the architecture of traditional systems by treating the target text as the direct output and learning the mapping from speech to text directly from paired data. However, compared to recognition tasks in other languages, the complex mapping relationship between Chinese speech and text, compounded by the challenges of homophones and polysemy recognition, often results in suboptimal performance for end-to-end systems.

To solve the above problems, we propose Syllable-Character Collaborative Model (SCCM). Our model includes a shared encoder and three decoders, which decode pinyin, initials and finals, and characters respectively. To mitigate the complexity of directly mapping speech to characters, we fuse the Pinyin embedding vector with the speech feature vector as the input of the character decoder. Pinyin embedding vectors share acoustic representations through the multi-task learning framework, which has been proven to reduce the impact of noise in speech [13,14] and narrowing the search space. While speech features represent local acoustic states and are more focused on physical properties like pitch and duration, they are less effective at capturing contextual information. In contrast, pinyin embeddings provide a higher-level semantic analysis of speech, offering richer contextual information to the decoder. To ensure effective coordination among these modules, we designed a Pinyin-Ensemble (PE) module, which synthesizes the outputs of the three decoders to generate more accurate pinyin. The generated pinyin is further processed through the Pinyin Feature (PF) module to produce pinyin embedding vectors, which are used as inputs to the next round of text decoding. To achieve unified optimization across modules, we developed a joint loss function. Finally, to reduce errors caused by homophones, we incorporate Pycorrector before text output.

In this paper, we use a multi-task learning model that incorporates pinyin, initials and finals to improve the accuracy of speech recognition. Our main contributions are summarized as follows:

- We propose Syllable-Character Collaborative Model, which includes a shared encoder and multiple decoders. We adopt different decoding methods for different modeling units, and use the same encoder and joint loss for training to achieve coordination between modules, thus improving the final recognition accuracy.
- We design Pinyin-Ensemble module using the idea of ensemble learning, which integrates pinyin, initials and finals, and Chinese characters to produce more accurate predicted pinyin. Subsequently, we utilize Pinyin Feature module to transform the pinyin into embedding vectors, which are concatenated with the shared feature vectors and input into the next round of text decoder, significantly reducing the error rate of the text.
- Experimental results on the AISHELL-1 dataset demonstrate that our model achieves significant improvements over baseline models.

The remainder of this paper is organized as follows: In the Related Work section, we review previous studies. The framework architecture and methodology are detailed in the Methodology section. In the Experiment section, we present the experimental setup and analyze the results. Finally, the findings are concluded in the Conclusion section.

Related work

End-to-end speech recognition models

Speech recognition technology has evolved from the era of Gaussian Mixture Models and Hidden Markov Models [15] (GMM-HMM) to the Deep Neural Networks and Hidden Markov Models [16,17] (DNN-HMM) era, eventually advancing to the current state-of-the-art end-to-end models. Currently, there are three main types of end-to-end technologies in the field of speech recognition:

Connectionist Temporal Classification [18–20] (CTC): CTC is commonly employed to handle sequences where the lengths of input and output are not aligned. A major drawback of CTC is its assumption that inputs at each time step are independent, which does not hold in speech recognition where there is semantic information across time steps. Recent work [21] mitigates this via BERT/GPT-2 [22,23] knowledge transfer, but its effectiveness is limited due to the inherent mismatch between pre-trained text representations and acoustic features [24].

Attention-based Encoder-Decoder architectures [25–27] (AED): Most research [28–30] on ASR primarily utilizes AED based on transformers [31], because it can capture long-range semantic relationship. However, they suffer from high latency and error propagation in noisy conditions. The mixed method [32] alleviates this problem by jointly decoding pinyin and characters, but it requires additional fuzzy pinyin datasets.

Recurrent Neural Network Transducer [33–35] (RNN-T): While RNN-T is suitable for streaming ASR, it is inherently limited when applied to non-streaming, full-audio ASR tasks due to its left-to-right decoding nature and lack of access to future context [36]. And it is ill-suited for our multi-task framework [37] due to RNN-T lacks the flexibility to integrate external intermediate predictions into its decoding process. Prior studies [38] show that RNN-T's shared joiner causes gradient conflicts when handling divergent tasks.

To address the alignment and long-range dependency issues in speech recognition, we use CTC and encoder-decoder architecture at the same time in this paper. CTC solves the problem of monotonic alignment between audio frames and sub phoneme units (such as pinyin and finals) by providing hard alignment, while the encoder decoder architecture can capture long-range dependencies and contextual information [39,40]. Compared with RNN-T, CTC has better alignment ability in full audio tasks, can handle long sequences and avoid the limitations of streaming decoding, making it more suitable for our multitasking framework.

Chinese modeling units

In traditional speech recognition, modeling units [41,42] are the fundamental units used to represent the mapping between speech signals and text. In traditional Chinese ASR, modeling units often include phonemes, initials and finals, and Pinyin. However, in strict end-to-end ASR, Chinese characters or words are used directly as speech modeling units.

The choice of modeling units significantly impacts recognition performance. A comparative study [43] of acoustic modeling units of deep neural networks for large-vocabulary Chinese speech recognition has proved that combining multiple modeling units performs better than each individual modeling unit. Further research [44] has explored end-to-end Chinese speech recognition. This study examined modeling units at three scales: context-dependent

phonemes, syllable tones, and Chinese characters. Phonemes lack direct correspondence to Chinese orthography, increasing grapheme-phoneme alignment complexity. Using word-based models in Chinese presents sparsity issues due to the large vocabulary.

Motivated by these findings, our approach integrates multiple modeling units into a unified framework. CTC decoding is applied to Pinyin and initials-finals because of its alignment-free training and efficiency in modeling sub-word units. The attention-based decoder is employed for Chinese characters, leveraging its strong capability in capturing long-range dependencies and contextual information.

Methodology

The structure of our SCCM is shown in Fig 1. The training of SCCM consists of three phases. In phase 1, the pinyin decoder and initials-finals decoder are trained independently using CTC loss, focusing on acoustic-phonetic alignment. These decoders predict the corresponding pinyin and initials-finals. In phase 2, the character decoder is introduced, and the system undergoes joint training with a multi-task loss function. During this phase, the outputs of all three decoders are integrated into the Pinyin-Ensemble (PE) module, which refines the pinyin predictions. Finally, in phase 3, the model undergoes full end-to-end training, with each round refining pinyin and text decoding through the iterative feedback of predictions. We use the predicted pinyin obtained from the previous round to generate the pinyin embedding vector through our pinyin feature module and fuse it with the shared feature as the input for the text decoder.

Shared encoder module

In this paper, we employ the conformer [45] as the encoder module. It contains 6 conformer blocks, which are composed of four stacked components: the FeedForward Network module (FFN), the Multi-Head Self-Attention module (MHSA), the Convolutional module (Conv), and a second FeedForward Network module. By combining convolutional operations with

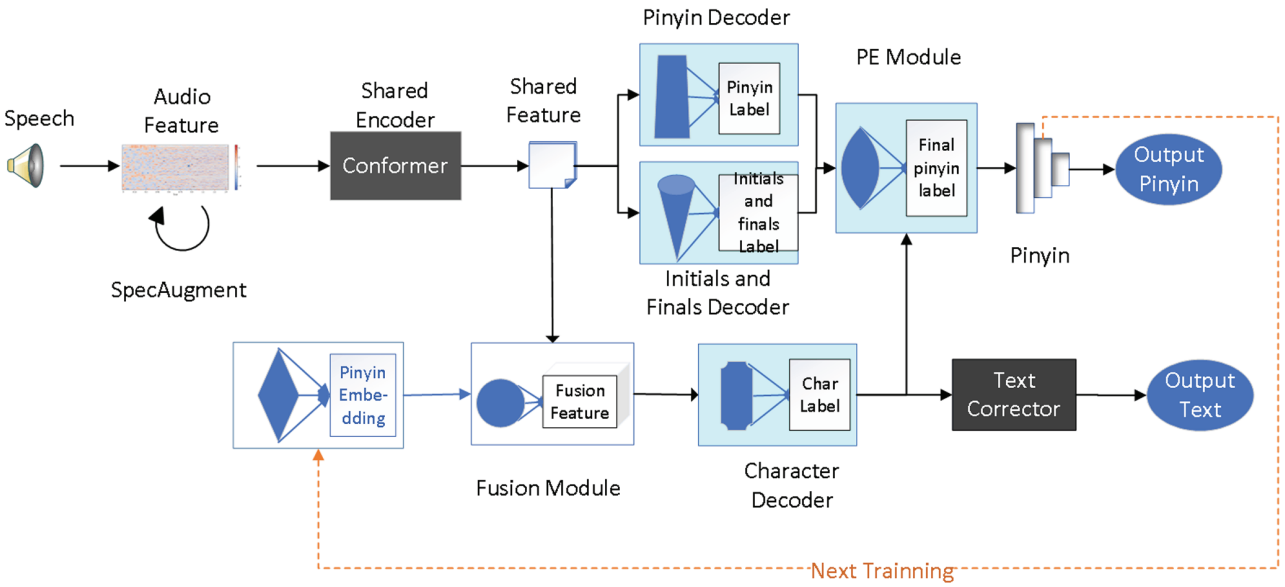

Fig 1. The structure of SCCM.

https://doi.org/10.1371/journal.pone.0325045.g001

self-attention mechanisms, conformer effectively captures both short-term and long-term features in speech signals. For an input $x = (x_1, x_2, x_3, ...x_T)$ to the Conformer module, where T is the number of time steps, the shared feature $h = (h_1, h_2, ...h_T)$ can be expressed as:

$$h = Conformer(x) \tag{1}$$

Pinyin decoder module

In the pinyin decoding module, we use the CTC decoding module to obtain the probability distribution of the pinyin sequence, CTC technology allows us to bypass the need for alignment between audio and pinyin outputs and can handle input-output sequences of varying lengths. It has been demonstrated that using pinyin as modeling units achieves the highest accuracy when decoded with CTC [46]:

$$P_{pinyin}(y|h) = \prod_{t=1}^{T} p(y_i|h_t), y_i \in V_{pinyin} \tag{2}$$

where $P_{pinyin}(y|h)$ is the probability distribution of the pinyin sequence, $y = (y_1, y_2, y_3, ..., y_u)$ is the predicted target sequence, u is the length of the prediction sequence y, $u \leq T$, i is the index of each target pinyin in prediction sequence y, V_{pinyin} is the pinyin dictionary we create. And the n-best method [47] is used to select the best-performing pinyin sequences, which can enable us to obtain the global optimal solution:

$$y^{pin} = \underset{y \in B(V_{pinyin})}{argmax} \; P_{pinyin}(y|h) \tag{3}$$

where y^{pin} is the best-performing pinyin sequences, $B(V_{pinyin})$ is the set of legal pinyin paths formed in the pinyin dictionary after removing spaces and merging consecutive repeated characters.

Initials and finals decoder module

Similarly, we use the CTC decoding module to obtain the probability distribution of the sequence of initials and finals, and we use the n-best method to select initials and finals. The calculation formula is as follows:

$$P_{if}(y|h) = \prod_{t=1}^{T} p(y_i|h_t), y_i \in V_{if} \tag{4}$$

$$y^{if} = \underset{y \in B(V_{if})}{argmax} P_{if}(y|h) \tag{5}$$

where $P_{if}(y|h)$ is the probability distribution of the initials and finals sequence, V_{if} is the initials and finals dictionary we create, i is the index of each target initial or final in prediction sequence y, $B(V_{if})$is the set of legal initials and finals paths formed in the initials and finals dictionary after removing spaces and merging consecutive repeated characters. The repetition prediction mechanism of CTC effectively handles multi-syllable words and connected speech phenomena in Chinese, thereby improving the robustness of initial and final recognition.

Pinyin feature module and feature fusion module

As a bridge between speech and text, Pinyin provides phonetic information that can aid character decoding by offering finer-grained pronunciation details [48,49]. Inspired by ChineseBERT [50], which integrates pinyin and glyph information to enhance the representation of Chinese text, we also incorporate pinyin embeddings into our model. However, unlike ChineseBERT, which is designed for text-based tasks, our focus is on improving speech recognition. In our approach, we fuse the shared features with the pinyin features to help the character decoder better understand the context information of the input. A pinyin representation rarely exceeds 6 characters (for example, "zhuang" for "壮"). However, to ensure consistency in the input sequence length across the model, we set the length of each pinyin vector to eight, with the remaining positions padded with the symbol ' - '. To obtain a compact and informative Pinyin representation, we first apply a Convolutional Neural Network (CNN) to extract hierarchical phonetic features from the padded Pinyin sequence. This CNN-based Pinyin embedding captures sub-syllabic dependencies, allowing the model to better understand pronunciation variations. In order to adapt to the input of the text decoder, we concatenate the shared features with the pinyin embedding vector into a new vector and convert this new vector into the input vector of the character decoder through a fully connected layer. This fusion reinforces phonetic constraints during decoding, thereby mitigating homophone errors and improving text generation quality [51,52]. The two modules are shown in Figs 2 and 3. The calculation formulas are as follows:

$$y_i^{pad} = Pad(y_{i-1}^{pp}) \tag{6}$$

$$y_i^{emb} = Conv(y_i^{pad}) \tag{7}$$

$$h_i^{conc} = Concatenate(h, y_i^{emb}) \tag{8}$$

$$h_i^{char} = FC(h_i^{conc}) \tag{9}$$

where y_{i-1}^{pp} is the output of the ensemble learning module in the previous round of training, y_i^{pad} is padded pinyin, y_i^{emb} is our pinyin embedding feature vector, h_i^{conc} is concatenated vector, h_i^{char} is the output feature of feature fusion module, i refers to the index of our training rounds. FC is the fully connected calculation function.

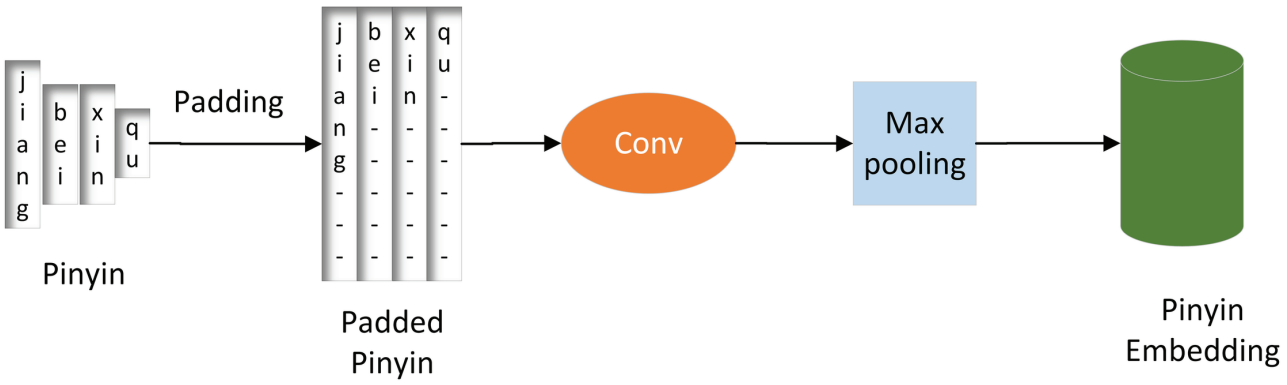

Fig 2. The pinyin feature module.

https://doi.org/10.1371/journal.pone.0325045.g002

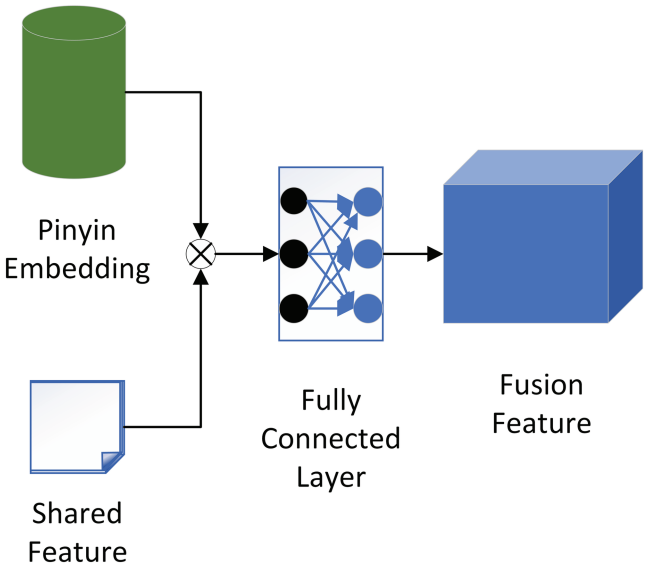

Fig 3. The feature fusion module. ⊗ denotes vector concatenation.

https://doi.org/10.1371/journal.pone.0325045.g003

Character decoder module and Pycorrector

We use the Transformer as the character decoder. During the initial training round, the decoder input consists solely of the shared features generated by the encoder. The formula for calculation is as follows:

$$y_0^{char} = Transformer(h) \tag{10}$$

In subsequent rounds, the fusion features are used as inputs, with characters as outputs. The calculate detail is as follows:

$$y_i^{char} = Transformer(h^{char}), \quad i = 1, 2, 3... \tag{11}$$

where y^{char} is the output of decoder, i refers to the index of our training rounds. Additionally, at the end of the process, we incorporate a Chinese text correction tool called Pycorrector to refine the decoder's output.

$$y^{fc} = Pycorrector(y^{char}) \tag{12}$$

where y^{fc} is our final Chinese text. This tool analyzes text using a combination of contextual and linguistic rules, along with statistical and deep learning models, to identify potential errors and suggest appropriate corrections. Pycorrector can effectively detect common spelling, grammatical, and phonetic errors in Chinese text. Its integration into our workflow enhances the accuracy of text post-processing, thereby improving the overall performance of the speech recognition model.

Pinyin-ensemble module

Since the accuracy of pinyin affects the accuracy of the final text, improving pinyin correctness is a critical issue. We pre-trained Pinyin-Ensemble (PE) module, which is constructed with the following ten base classifiers: Logistic Regression [53], Support Vector Classifier (SVC) [54], K-Nearest Neighbors (KNN) [55], Decision Tree [56], Random Forest [57], Gradient Boosting [58], Naive Bayes [59], Multi-layer Perceptron (MLP) [60], AdaBoost [61], and Bagging Classifier [62]. Each base classifier is trained individually on the extracted character-level features from pinyin, syllable, and Chinese character data. A Random Forest classifier acts as the meta-learner, receiving predictions from each base classifier as input features. Data set is split with an 80-20 ratio for training and testing, using a 5-fold cross-validation approach. Hyperparameters were optimized for each base classifier based on grid search results. The calculation formula is as follows:

$$y^{pp} = Jc(y^{pin}, y^{if}, y^{char}) \tag{13}$$

where the inputs include pinyin, initials and finals, and Chinese pinyin, and the output y^{pp} is the predicted pinyin of this round of training. Jc is the calculation function of our Pinyin-Ensemble module. The main purpose of this module is to integrate the pinyin information obtained from these three decoding methods, resulting in more accurate pinyin and reducing the impact of subsequent pinyin errors on text decoding.

Training loss

In order to achieve unified optimization between modules, we optimize weight sum of the loss functions of different modules uniformly. The calculation formula is as follows:

$$L_{combine} = 0.25 * L_{CTC}^{pinyin} + 0.25 * L_{CTC}^{if} + 0.5 * L_{CE}^{char} \tag{14}$$

where $L_{combine}$ is the combined loss, L_{CTC}^{pinyin} and L_{CTC}^{if} is the CTC loss of the pinyin and initials and finals, and L_{CE}^{char} is the cross-entropy loss of the character.

Experiment

Data set and metrics

In this paper, we use the AISHELL-1 dataset [63] in 16 kHz WAV format to verify the performance of all the models. AISHELL-1 contains high-quality Mandarin speech recorded from multiple channels including iOS, Android, and microphones, thus covering diverse acoustic conditions. Each speech is represented as 80-dimensional filterbank coefficients computed every 10ms with a 25ms window length. All feature sequences are normalized using the mean and variance of each audio sample. SpecAugment [64] is employed to augment audio data for model training.

For both pinyin and text, we adopt Character Error Rate (CER) [65,66] as the final evaluation metric, which is the most recognized metric in speech recognition. The calculation formula of CER is as follows:

$$CER = \frac{S + D + I}{N} \tag{15}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions and N is the number of characters.

Implementation details

The proposed model consists of six encoder layers, and six character decoder layers. The dimensions used in the model are as follows: the model dimension is 512, the embedding dimension is 512, and both the key and value dimensions of conformer are 64. We use the Adam optimizer with a learning rate of 0.001, following the same optimizer settings as the transformer model. To further prevent overfitting, label smoothing and dropout are applied, both with a probability of 0.1. All models are trained for 60 epochs with a batch size of 32.

Experimental result and analysis

In Table 1, we compare the results of our model SCCM on AISHELL-1 dataset with commonly used ASR models, including: AISHELL-1 Baseline [63], SpeechTransformer [67], Conformer, CNN with CTC, Wav2vec2.0 [68], Dual-Decoder [32]. All models are evaluated on both Pinyin and Character levels, enabling a comprehensive assessment across phonetic and textual representations. Our proposed model consistently outperforms all baselines, achieving the lowest CER (6.4%) and pinyin CER (3.1%). In contrast, the Dual-Decoder model attains competitive results but relies on an additional fuzzy Pinyin dataset, introducing extra data dependency. Our method, trained solely on standard AISHELL-1 data, demonstrates superior performance without such reliance. The simultaneous reduction in both character and pinyin error rates indicates a systematic improvement rather than a result of random variation. We use CTC for initials and finals and pinyin, which can prove that CTC is more suitable for these than transformer.

In order to explore the contribution of phonetic information and tone information of our performance, we designed an ablation experiment whose results are shown in Table 2. Dc refers to using a text decoder, Dp means using a pinyin decoder, Dif refers to using an initials and finals decoder, PE/PE (Dp & Dif) means combining the outputs of decoders to generate new pinyin. Fusion means the pinyin embedding vectors will be fed into the text decoder. '✓' indicates that the module is included.

Table 1. CER of our model and commonly used ASR models on AISHELL-1 test sets.

Models	**CER**	
	Pinyin	**Text**
Baseline	6.15	11.97
CNNwithCTC	4.96	10.21
SpeechTransformer	5.03	10.47
Conformer	3.20	7.30
wav2vec2.0	2.80	7.20
DUAL–DECODER	2.65	6.60
SCCM	**2.41**	**6.50**

https://doi.org/10.1371/journal.pone.0325045.t001

Table 2. Ablation study of our model on AISHELL-1 test sets.

Modules	*Dc*	*Dp*	*Dif*	*Fusion*	*PE*(*Dp*&*Dif*)	*PE*	**CER**	
							Pinyin	**Text**
	✓						––	8.20
	✓	✓		✓			3.54	7.83
SCCM	✓		✓	✓			3.69	7.92
	✓	✓	✓	✓	✓		2.55	6.73
	✓	✓	✓	✓		✓	**2.41**	**6.50**

https://doi.org/10.1371/journal.pone.0325045.t002

We observe that using only the character decoder does not yield satisfactory results. Incorporating the pinyin decoder greatly reduces the character error rate, which indicates that pinyin directly corresponds to the pronunciation of speech and can provide more phonetic information. By comparing the second and third rows, we find that pinyin is more effective than initials and finals in assisting character decoding and results in a lower CER. Furthermore, by integrating both pinyin and initials/finals information through our Pinyin-Ensemble (PE) module, we achieve a 28.0% reduction in pinyin error rate (from 3.54% to 2.55%) and a 14.0% reduction in text CER (from 7.83% to 6.73%). This improvement confirms that combining multiple syllable unit information through ensemble learning can significantly reduce error rates.

To explore how the Pycorrector contributes to the performance of our model, we conduct an ablation experiment, whose result is shown in Table 3. Although the improvement might seem not significant, it indicates that Pycorrector effectively reduces errors in the predicted text. To further demonstrate the effectiveness of Pycorrector, we show some examples in Table 4. Case 0 and case 1 show the state that SCCM with and without Pycorrector are both correct. Pycorrector can help us solve some personal names problems as in case 2. In case 3, we find that it may make mistakes when matching some uncommon words. We can see the final output text of SCCM without Pycorrector may still contain a small number of errors due to misdecoding of pinyin or syllables as in case 4, Pycorrector further optimizes the character-level output and improves accuracy by correcting the generated text.

Conclusion

In this paper, we propose the Syllable-Character Collaborative Model. The model fully utilizes the units of Chinese speech and simultaneously performs decoding tasks for initials and finals, pinyin and Chinese characters. We also design a Pinyin-Ensemble module to integrate the outputs of these three tasks for learning, improving the recognition accuracy of pinyin and text. At the same time, we add Pycorrector to reduce the homonym errors. The results on the test set of AISHELL-1 dataset show that the proposed model outperforms commonly used mainstream ASR models on AISHELL-1 dataset.

Table 3. Ablation study of pycorrector on AISHELL-1 test sets.

Models	CER	
	Pinyin	**Text**
SCCM Without Pycorrector	2.45	6.53
SCCM	**2.41**	**6.50**

https://doi.org/10.1371/journal.pone.0325045.t003

Table 4. Example output of SCCM with and without Pycorrector.

Case	SCCM without Pycorrector	SCCM	Ground Truth
0	苹果此举是为了节约用电量	苹果此举是为了节约用电量	苹果此举是为了节约用电量
1	博士	博士	博士
2	陈延希穿着粉色上衣	陈妍希穿着粉色上衣	陈妍希穿着粉色上衣
3	黄蓉穿着它不仅刀剑不入	黄蓉穿着它不仅刀枪不入	黄蓉穿着它不仅刀剑不入
4	完善主地承包经营全流市市场	完善土地承包经营权流转市场	完善土地承包经营权流转市场

https://doi.org/10.1371/journal.pone.0325045.t004

Author contributions

Conceptualization: Zeyuan Chen.

Data curation: Zeyuan Chen.

Formal analysis: Zeyuan Chen.

Investigation: Zeyuan Chen.

Methodology: Zeyuan Chen.

Project administration: Zeyuan Chen.

Resources: Cheng Zhong.

Software: Zeyuan Chen.

Supervision: Danyang Chen.

Validation: Zeyuan Chen.

Visualization: Zeyuan Chen.

Writing –original draft: Zeyuan Chen.

Writing –review & editing: Danyang Chen.

References

1. Alharbi S, Alrazgan M, Alrashed A, Alnomasi T, Almojel R, Alharbi R, et al. Automatic speech recognition: systematic literature review. IEEE Access. 2021;9:131858–76. https://doi.org/10.1109/access.2021.3112535
2. Zhang X, Zhang R. Evolution of ancient alphabet to modern Greek, Latin and cyrillic alphabets and transcription between them. In: Proceedings of the 2018 4th International Conference on Economics, Social Science, Arts, Education and Management Engineering (ESSAEME 2018). 2018. https://doi.org/10.2991/essaeme-18.2018.30
3. Taft M, Hambly G. The influence of orthography on phonological representations in the lexicon. J Memory Lang. 1985;24(3):320–35. https://doi.org/10.1016/0749-596x(85)90031-2
4. Duyen TMT. Exploring phonetic differences and cross-linguistic influences: a comparative study of english and mandarin chinese pronunciation patterns. OJAppS. 2024;14(07):1807–22. https://doi.org/10.4236/ojapps.2024.147118
5. Li J, Zheng TF, Byrne W, Jurafsky D. A dialectal chinese speech recognition framework. J Comput Sci Technol. 2006;21(1):106–15. https://doi.org/10.1007/s11390-006-0106-9
6. Li L, Long Y, Xu D, Li Y. Boosting character-based Mandarin ASR via Chinese Pinyin representation. Int J Speech Technol. 2023;26(4):895–902. https://doi.org/10.1007/s10772-023-10050-z
7. Zhang X, Peng Y, Xu X. An overview of speech recognition technology. In: 2019 4th International Conference on Control, Robotics and Cybernetics (CRC); 2019. p. 81–5.
8. Rabiner LR. A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE. 1989;77(2):257–86. https://doi.org/10.1109/5.18626
9. Mohri M, Pereira FCN, Riley M. Speech recognition with weighted finite-state transducers. Springer handbook of speech processing. 2008.
10. Li J. Recent advances in end-to-end automatic speech recognition. arXiv preprint 2021. https://doi.org/abs/2111.01690
11. Wang A, Zhang L, Song W, MEeng J. Review of end-to-end streaming speech recognition. Comput Eng Appl. 2023;59(2):22–33.
12. Bijwadia S, Chang S y, Li B, Sainath T, Zhang C, He Y. Unified end-to-end speech recognition and endpointing for fast and efficient speech systems. In: 2022 IEEE Spoken Language Technology Workshop (SLT). 2023. p. 310–6.
13. Lavechin M, Métais M, Titeux H, Boissonnet A, Copet J, Rivière M, et al. Brouhaha: multi-task training for voice activity detection, speech-to-noise ratio, and C50 room acoustics estimation. In: 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU); 2023. p. 1–7.

14. Thanda A, Venkatesan SM. Multi-task learning of deep neural networks for audio visual automatic speech recognition. arXiv preprint 2017. https://arxiv.org/abs/1701.02477
15. Levinson SE, Rabiner LR, Sondhi MM. An introduction to the application of the theory of probabilistic functions of a markov process to automatic speech recognition. Bell System Technical Journal. 1983;62(4):1035–74. https://doi.org/10.1002/j.1538-7305.1983.tb03114.x
16. Dahl GE, Dong Yu, Li Deng, Acero A. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. IEEE Trans Audio Speech Lang Process. 2012;20(1):30–42. https://doi.org/10.1109/tasl.2011.2134090
17. Wong T, Li C, Lam S, Chiu B, Lu Q, Li M, et al. Syllable based DNN-HMM Cantonese speech to text system. arXiv preprint 2024. https://arxiv.org/abs/2402.08788
18. Graves A, Fernández S, Gomez F, Schmidhuber J. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine Learning. ICML ' 06. New York, NY, USA: Association for Computing Machinery; 2006. p. 369–76. https://doi.org/10.1145/1143844.1143891
19. Zhou J, Zhao S, Liu Y, Zeng W, Chen Y, Qin Y. KNN-CTC: enhancing ASR via retrieval of CTC pseudo labels. In: ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2024. p. 11006–10.
20. Chen C, Gong X, Qian Y. Efficient text-only domain adaptation for CTC-based ASR. In: 2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU); 2023. p. 1–7.
21. Deng K, Cao S, Zhang Y, Ma L, Cheng G, Xu J, et al. Improving CTC-based speech recognition via knowledge transferring from pre-trained language models. In: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2022. p. 8517–21.
22. Devlin J, Chang MW, Lee K, Toutanova K. BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein J, Doran C, Solorio T, editors. Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Minneapolis, Minnesota: Association for Computational Linguistics; 2019. p. 4171–86. https://aclanthology.org/N19-1423/
23. Radford A, Wu J, Child R, Luan D, Amodei D, Sutskever I. Language models are unsupervised multitask learners. OpenAI Technical Report. 2019.
24. Joshi R, Singh A. A Simple baseline for domain adaptation in end to end ASR systems using synthetic data. In: Proceedings of The Fifth Workshop on e-Commerce and NLP (ECNLP 5). 2022. p. 244–9. https://doi.org/10.18653/v1/2022.ecnlp-1.28
25. Chan W, Jaitly N, Le Q, Vinyals O. Listen, attend and spell: a neural network for large vocabulary conversational speech recognition. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2016. p. 4960–4.
26. Egorova E, Vydana HK, Burget L, Cernocky JH. Spelling-aware word-based end-to-end ASR. IEEE Signal Process Lett. 2022;29:1729–33. https://doi.org/10.1109/lsp.2022.3192199
27. Yang GP, Tang H. Supervised attention in sequence-to-sequence models for speech recognition. In: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2022. p. 7222–6.
28. Yusuf B, Gandhe A, Sokolov A. Usted: improving ASR with a unified speech and text encoder-decoder. In: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2022. p. 8297–301.
29. Tang J, Kim K, Shon S, Wu F, Sridhar P. Improving ASR contextual biasing with guided attention. In: ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2024. p. 12096–100.
30. He J, Shi X, Li X, Toda T. MF-AED-AEC: speech emotion recognition by leveraging multimodal fusion, ASR error detection, and ASR error correction. In: ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2024. p. 11066–70.
31. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. In: Guyon I, Luxburg UV, Bengio S, Wallach H, Fergus R, Vishwanathan S, et al., editors. Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc.; 2017. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
32. Yang Z, Ng D, Fu X, Han L, Xi W, Wang R. On the effectiveness of pinyin-character dual-decoding for end-to-end Mandarin Chinese ASR. arXiv preprint 2022. https://arxiv.org/abs/2201.10792
33. Tian Z, Yi J, Tao J, Bai Y, Wen Z. Self-attention transducers for end-to-end speech recognition. In: Interspeech 2019. 2019. https://doi.org/10.21437/interspeech.2019-2203
34. Joshi V, Zhao R, Mehta RR, Kumar K, Li J. Transfer learning approaches for streaming end-to-end speech recognition system. arXiv preprint 2020. https://arxiv.org/abs/2008.05086

35. Radfar M, Barnwal R, Swaminathan RV, Chang FJ, Strimel GP, Susanj N. ConvRNN-T: Convolutional augmented recurrent neural network transducers for streaming speech recognition. arXiv preprint 2022. https://arxiv.org/abs/2209.14868
36. Zhao W, Li Z, Yu C, Ou Z. Cuside-t: Chunking, simulating future and decoding for transducer based streaming ASR. arXiv preprint 2024. https://arxiv.org/abs/2407.10255
37. Graves A, Mohamed AR, Hinton G. Speech recognition with deep recurrent neural networks. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. 2013. p. 6645–9.
38. Sak H, Senior A, Rao K, Beaufays F. Fast and accurate recurrent neural network acoustic models for speech recognition. arXiv preprint 2015. https://arxiv.org/abs/1507.06947
39. Kim S, Hori T, Watanabe S. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2017. p. 4835–9.
40. Park H, Kim C, Son H, Seo S, Kim J-H. Hybrid CTC-attention network-based end-to-end speech recognition system for Korean language. JWE. 2022. https://doi.org/10.13052/jwe1540-9589.2126
41. Gu Y, Jin Y, Ma Y, Jiang F, Yu J. Multimodal emotion recognition based on acoustic and lexical features. J Data Acquisit Process. 2022;37(6):1353. https://doi.org/10.16337/j.1004-9037.2022.06.016
42. Rousso R, Cohen E, Keshet J, Chodroff E. Tradition or innovation: a comparison of modern ASR methods for forced alignment. arXiv preprint 2024. https://arxiv.org/abs/2406.19363
43. Li X, Yang Y, Pang Z, Wu X. A comparative study on selecting acoustic modeling units in deep neural networks based large vocabulary Chinese speech recognition. Neurocomputing. 2015;170:251–6. https://doi.org/10.1016/j.neucom.2014.07.087
44. Zou W, Jiang D, Zhao S, Yang G, Li X. Comparable study of modeling units for end-to-end mandarin speech recognition. In: 2018 11th International Symposium on Chinese Spoken Language Processing (ISCSLP). 2018. p. 369–73.
45. Gulati A, Qin J, Chiu CC, Parmar N, Zhang Y, Yu J, et al. Conformer: convolution-augmented transformer for speech recognition. arXiv preprint 2020. https://arxiv.org/abs/2005.08100
46. Vishnoi A, Aggarwal A, Prasad A, Prateek M. An encryption method involving homomorphic transform. In: 2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON). 2021. p. 359–63.
47. Schwartz R, Chow YL. The N-best algorithms: an efficient and exact procedure for finding the N most likely sentence hypotheses. In: International Conference on Acoustics, Speech, and Signal Processing, vol.1; 1990. p. 81–4.
48. Chen L, Perfetti CA, Fang X, Chang L-Y, Fraundorf S. Reading Pinyin activates sublexcial character orthography for skilled Chinese readers. Lang Cogn Neurosci. 2019;34(6):736–46. https://doi.org/10.1080/23273798.2019.1578891 PMID: 33015216
49. Liang Z, Quan X, Wang Q. Disentangled phonetic representation for Chinese spelling correction. arXiv preprint 2023. https://arxiv.org/abs/2305.14783
50. Sun Z, Li X, Sun X, Meng Y, Ao X, He Q, et al. ChineseBERT: Chinese pretraining enhanced by glyph and pinyin information. arXiv preprint 2021. https://arxiv.org/abs/2106.16038
51. Li Y, Qiao X, Zhao X, Zhao H, Tang W, Zhang M, et al. Large language model should understand Pinyin for Chinese ASR error correction. In: ICASSP 2025 - 2025 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP); 2025. p. 1–5.
52. Zhang R, Pang C, Zhang C, Wang S, He Z, Sun Y, et al. Correcting chinese spelling errors with phonetic pre-training. In: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021. 2021. p. 2250–61. https://aclanthology.org/2021.findings-acl.198/
53. Cox DR. The regression analysis of binary sequences. J Roy Statist Soc Ser B: Statist Methodol. 1958;20(2):215–32. https://doi.org/10.1111/j.2517-6161.1958.tb00292.x
54. Cortes C, Vapnik V. Support-vector networks. Mach Learn. 1995;20(3):273–97. https://doi.org/10.1007/bf00994018
55. Cover T, Hart P. Nearest neighbor pattern classification. IEEE Trans Inform Theory. 1967;13(1):21–7. https://doi.org/10.1109/tit.1967.1053964
56. Loh W. Classification and regression trees. WIREs Data Min Knowl. 2011;1(1):14–23. https://doi.org/10.1002/widm.8
57. Breiman L. Random Forests. Mach Learn. 2001;45(1):5–32. https://doi.org/10.1023/a:1010933404324
58. Friedman JH. Greedy function approximation: a gradient boosting machine. Annals Statist. 2001;29:1189–232.
59. Yang FJ. An implementation of naive bayes classifier. In: 2018 International Conference on Computational Science and Computational Intelligence (CSCI). 2018. 301–6.

60. Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature. 1986;323(6088):533–6. https://doi.org/10.1038/323533a0
61. Freund Y, Schapire RE. A decision-theoretic generalization of on-line learning and an application to boosting. J Comput Syst Sci. 1997;55(1):119–39. https://doi.org/10.1006/jcss.1997.1504
62. Breiman L. Bagging predictors. Mach Learn. 1996;24(2):123–40. https://doi.org/10.1007/bf00058655
63. Bu H, Du J, Na X, Wu B, Zheng H. AISHELL-1: An open-source Mandarin speech corpus and a speech recognition baseline. In: 2017 20th Conference of the Oriental Chapter of the International Coordinating Committee on Speech Databases and Speech I/O Systems and Assessment (O-COCOSDA); 2017. p. 1–5.
64. Park DS, Chan W, Zhang Y, Chiu C-C, Zoph B, Cubuk ED, et al. SpecAugment: a simple data augmentation method for automatic speech recognition. In: Interspeech 2019. 2019. https://doi.org/10.21437/interspeech.2019-2680
65. Park C, Kang H, Hain T. Character error rate estimation for automatic speech recognition of short utterances. In: 2024 32nd European Signal Processing Conference (EUSIPCO); 2024. p. 131–5.
66. K TD, James J, Gopinath DP, K MA. Advocating character error rate for multilingual ASR evaluation. arXiv preprint 2024. https://arxiv.org/abs/2410.07400
67. Dong L, Xu S, Xu B. Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition. In: 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). 2018. p. 5884–8.
68. Yuan J, Cai X, Gao D, Zheng R, Huang L, Church K. Decoupling recognition and transcription in Mandarin ASR. In: 2021 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU). 2021. p. 1019–25.