A Speech Recognition Model for Mandarin Chinese Pronunciation Evaluation

1st Tadeusz Brzeski
Faculty of Electronics, Telecommunications and Informatics
Gdańsk University of Technology
Gdańsk, Poland
s191343@student.pg.edu.pl

2nd Kamil Fischbach
Faculty of Electronics, Telecommunications and Informatics
Gdańsk University of Technology
Gdańsk, Poland
fischbach.kamil@gmail.com

***Abstract*—This paper investigates convolutional neural networks for automated pronunciation assessment of Mandarin Chinese on a constrained vocabulary task. We evaluate three architectures: a baseline (using only word identity without audio features), ContextFusionCNN (a multi-modal architecture integrating MFCCs with temporal features), and ContextCNN (a simplified variant using only MFCCs). The dataset consists of 5,855 single-syllable Mandarin numeral pronunciations by Polish native speakers. Results demonstrate that both CNN-based models outperform the baseline, with ContextCNN achieving the best test accuracy of 71.7%. Trained models and source code are released under the MIT license on GitHub[1].**

***Index Terms*—Pronunciation assessment, Mandarin Chinese, convolutional neural networks, MFCC, computer-assisted language learning, speech recognition, multi-modal learning**

I. Introduction

Automated pronunciation assessment plays an important role in computer-assisted language learning (CALL) systems, enabling learners to receive feedback without requiring constant human supervision. Mandarin Chinese pronunciation assessment presents challenges due to its tonal system and phonetic features that may be unfamiliar to speakers of other languages.

This paper investigates convolutional neural network architectures for Mandarin pronunciation assessment on a constrained vocabulary task. We evaluate three models: a baseline using only word embeddings, and two CNN-based architectures that incorporate acoustic features (i.e. MFCCs) with varying complexity. The dataset consists of 5,855 single-syllable numeral Mandarin pronunciations from Polish native speakers.

The contributions of this work are: (1) empirical comparison of CNN-based models against a baseline for pronunciation assessment, (2) release of trained models and source code under the MIT license. The codebase provides a framework with composable abstractions for declarative dataset construction, designed to reduce coupling and prevent errors. Additional experiments are documented in the project repository issues[2].

The remainder of this paper is organized as follows: Section II reviews related work in Mandarin pronunciation assessment. Section IV describes the experimental methodology and dataset. Section V details the experimental setup. Section VI presents results. Section VII discusses findings and limitations. Section VIII outlines future research directions.

[1]https://github.com/tad1/Mandarin_Pronunciation_Recognition_Project

[2]https://github.com/tad1/Mandarin_Pronunciation_Recognition_Project/issues

II. Related Work

Automatic pronunciation assessment for Mandarin Chinese has evolved from traditional HMM/GMM-based approaches to deep learning methods. Chen et al. [1] developed a pronunciation assessment system using HMM-based speech processing for phoneme recognition combined with GMM-based tone recognition on sentence-level audio. Li et al. [2] proposed a DNN-based tone extended recognition network (ERN) for tone mispronunciation detection, reducing equal error rate by 10.98% relative compared to conventional GOP systems. More recently, Wang et al. [3] introduced a stateless RNN-T model utilizing HuBERT features with pitch embedding, achieving a 3% improvement in Phone Error Rate trained solely on native speaker data.

In contrast to these sequence-based approaches on continuous speech, this study investigates convolutional neural networks on a constrained vocabulary of single-syllable words, evaluating pronunciation correctness without tone assessment.

III. Related Work

(Automatic Pronunciation Assessment for Mandarin Chinese) (sequence model; audio of sentences; score the pronuncation quality; uses HMM for speech processing, recognition; and GMM for tone recognition)

IV. Experiment

This section describes the experimental details and methodology employed to evaluate model architectures for Mandarin pronunciation assessment. While multiple experiments were conducted during development and documented in the project repository, this paper consolidates several of them into a single experiment.

A. Problem formulation

This study investigates the feasibility of using convolutional neural networks for automated pronunciation assessment on a constrained vocabulary. The experiment focuses on 12 single-syllable Mandarin numerals from the dataset, evaluating only pronunciation correctness while excluding tone assessment. The problem is formulated as binary classification, where models predict whether a given pronunciation is correct or incorrect based on audio features and word identity.

B. Methodology

The experimental approach employed a comparative evaluation of multiple model architectures. Given the absence of comparable prior work for this specific task configuration, a zero model was established as a baseline reference. This baseline model operates solely on word embeddings without incorporating any audio features, providing a reference point for performance without acoustic information. Comparing the audio-based models against this baseline demonstrates whether incorporating audio features improves prediction accuracy.

Three architectures of increasing complexity were trained and evaluated on identical train-validation-test datasets to ensure fair comparison. Performance was measured across training, validation, and test sets to assess both learning capacity and generalization capability.

C. Data

1) Dataset Description: The dataset used in this study originates from Yen Ying Ng's doctoral research. This proprietary dataset contains over 12,000 audio samples of Mandarin pronunciation of 30 Chinese words from 548 students across different universities. Each sample is manually assessed for both pronunciation and tone correctness. The speakers are predominantly of Polish native language background, with a minority representing different native languages.

2) Data preparation: This experiment uses a subset of 12 numerals (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, and 100) from the dataset, where each word pronunciation consists of a single pinyin syllable. This experimental subset contains a total of 5,855 samples from 547 speakers. The data was split into training, validation, and test sets using a stratified approach based on word distribution (60/20/20 ratio). A fixed random seed was used to ensure consistent evaluation across experiments.

To ensure consistency and compatibility with the models used, all audio samples were resampled to 16 kHz. The audio samples contained mouse clicks and unvoiced segments, which were addressed using Silero VAD [4] to extract only voiced segments from each audio sample. Audio parameters including MFCC, zero-crossing rate, and RMS energy were extracted using the torchaudio [6] and librosa [5] libraries. The extracted parameters were padded or truncated to a fixed length without significant loss of information.

To speed up the training process, a custom PyTorch DataLoader was implemented that loads the entire processed dataset into GPU memory, eliminating the bottleneck of CPU-GPU data transfer.

3) Feature Extraction: Two types of audio features were extracted from the samples:

- **2D Spectral Features:** Mel Frequency Cepstral Coefficients (MFCCs) were computed using torchaudio [6] with 40 coefficients, 128 mel bands, FFT size of 1024, and hop length of 512 samples.
- **1D Temporal Features:** Zero-crossing rate and RMS energy were extracted using librosa [5] with a frame length of 1024 and hop length of 512 samples.

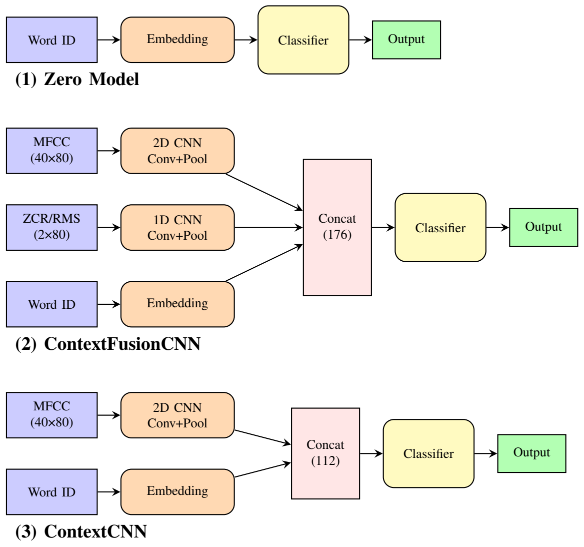

Fig. 1. High-level diagrams of the evaluated model architectures.

All feature sequences were padded or truncated to a fixed length of 80 frames to ensure uniform input dimensions across the dataset.

D. Model Architecture

While multiple architectures were evaluated during development, this paper focuses on the models that demonstrated the best performance after extensive experimentation and refinement. Detailed implementation specifics are available in the project repository, while high-level architectural diagrams are presented in Figure 1. Three architectures were evaluated:

1) Zero Model: serves as a baseline for comparison. It utilizes only word embeddings as input features, omitting any audio-derived information. This model provides a reference point to assess added value of incorporating audio features in subsequent architectures. It consists of an embedding model followed by classification model.

2) ContextFusionCNN: integrates MFCC parameters (2D), temporal and energy features (1D), and word id (scalar) as input. These inputs are passed through 2D CNN, 1D CNN, and embedding model, respectively. The resulting feature vectors are concatenated and passed to a classification head, enabling the model to leverage both spectral and temporal information alongside word context for pronunciation assessment.

3) ContextCNN: is a simplified variant of ContextFusionCNN that processes only 2D audio features (MFCC) and word embeddings, excluding temporal and energy parameters. This architecture was designed to evaluate the contribution of 1D features to overall model performance.

V. Experimental Setup

The experiment described in this paper corresponds to repository commit `paper_v1.0`. Reproduction details, including hardware specifications, software dependencies, dataset verification, and execution instructions, are provided in Appendix A.

All three models were trained on identical data splits using the same hyperparameters. Training employed the Adam optimizer (learning rate 10^{-4}, weight decay 10^{-4}) with binary cross-entropy loss. A ReduceLROnPlateau scheduler reduced the learning rate by 0.5 after 5 epochs without validation loss improvement. Models were trained for 140 epochs with batch size 16. CNN-based architectures incorporated batch normalization after each convolutional layer, ReLU activation, max pooling, adaptive average pooling for global feature aggregation, and progressively increasing dropout rates (0.1, 0.2, 0.3) across convolutional blocks. The classifier head used dropout rate 0.5.

VI. Results

The results of the experiments are summarized in Table I. Zero Model, which serves as a baseline, achieved test accuracy of 64.4%. Both audio-based models outperform the baseline: ContextFusionCNN achieved 70.1% test accuracy, while ContextCNN demonstrated the highest performance with 71.7% test accuracy. The small gap between training and test accuracy across all models indicates good generalization without significant overfitting.

Table II presents the confusion matrices comparing model predictions against human expert assessments on the test set.

TABLE I
Model Performance Across Training, Validation, and Test Sets

Model	**Accuracy**			**Test Set Metrics**		
	Train	**Val**	**Test**	**Precision**	**Recall**	**F1**
Zero Model	0.631	0.643	0.644	0.692	0.701	0.696
ContextFusionCNN	0.743	0.703	0.701	0.712	**0.816**	0.761
ContextCNN	**0.757**	**0.730**	**0.717**	**0.744**	0.783	**0.763**

TABLE II
Confusion Matrices: Model Predictions vs. Human Assessment on Test Set

Zero Model		
Model / **Expert**	**Correct**	**Incorrect**
Correct	480	205
Incorrect	214	277

ContextFusionCNN		
Model / **Expert**	**Correct**	**Incorrect**
Correct	559	126
Incorrect	226	265

ContextCNN		
Model / **Expert**	**Correct**	**Incorrect**
Correct	536	149
Incorrect	184	307

VII. Discussion

ContextFusionCNN achieved lower accuracy than ContextCNN despite incorporating additional features (i.e. zero-crossing rate and RMS energy). This result suggests several possible explanations: (1) the temporal features may be redundant with information already encoded in MFCCs, (2) these features may be irrelevant for pronunciation assessment in this constrained vocabulary task, or (3) the 1D CNN branch may process these features suboptimally. Future work should evaluate whether ZCR and RMS energy alone (without MFCCs) contribute to classification accuracy, which would help distinguish between feature redundancy and architectural limitations.

The Zero Model baseline achieves 64.4% test accuracy by learning word-level error rates without acoustic features. The stratified split preserves these error distributions across train and test sets. If the dataset reflects real-world learner patterns, this statistical approach could provide meaningful information.

The constrained vocabulary of 12 single-syllable numerals limits the generalizability of these findings to broader pronunciation assessment tasks. Additionally, the dataset consists predominantly of Polish native speakers, which may not represent pronunciation patterns from learners of other language backgrounds. Finally, this study evaluates pronunciation correctness without assessing tonal accuracy, which is essential in Mandarin.

VIII. Future Work

Future research could systematically evaluate alternative audio features to identify optimal representations for pronunciation assessment. Investigating model interpretability would help understand what acoustic patterns the CNN learns. Incorporating tone assessment into the framework would address a critical limitation of the current approach. Thorough investigation of pretrained models such as Wav2Vec2 would determine whether acoustic representations from ASR-trained models are effective for pronunciation assessment. Developing public benchmark datasets would enable standardized evaluation and comparison of different methods.

Acknowledgment

Special thanks to dr Adam Przybyłek and Yen Ying Ng for mentorship and for providing the dataset.

References

[1] J.-C. Chen, J.-S. Jang, J.-Y. Li, and M.-C. Wu, "Automatic pronunciation assessment for Mandarin Chinese," in *IEEE International Conference on Multimedia and Expo (ICME)*, vol. 3, 2004, pp. 1979–1982.

[2] W. Li, S. M. Siniscalchi, N. F. Chen, and C.-H. Lee, "Using tone-based extended recognition network to detect non-native Mandarin tone mispronunciations," in *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2016, pp. 1–4.

[3] X. Wang, M. Shi, and Y. Wang, “Pitch-aware RNN-T for Mandarin Chinese mispronunciation detection and diagnosis,” *arXiv preprint arXiv:2406.04595*, 2024.

[4] Silero Team, “Silero VAD: pre-trained enterprise-grade Voice Activity Detector (VAD), Number Detector and Language Classifier,” GitHub repository, 2024. [Online]. Available: https://github.com/snakers4/silero-vad

[5] B. McFee *et al.*, “librosa/librosa: 0.11.0,” Zenodo, version 0.11.0, Mar. 2025, doi: 10.5281/zenodo.15006942. [Online]. Available: https://doi.org/10.5281/zenodo.15006942

[6] J. Hwang *et al.*, “TorchAudio 2.1: Advancing speech recognition, self-supervised learning, and audio processing components for PyTorch,” *arXiv preprint arXiv:2310.17864*, 2023.

APPENDIX

TABLE III
HARDWARE SPECIFICATIONS

Component	**Specification**
OS	Manjaro Linux x86_64
Kernel	6.16.8-1-MANJARO
CPU	Intel i3-9100F (4) @ 4.200GHz
GPU	NVIDIA GeForce GTX 1060 6GB
Memory	12GB DDR4 2133MHz

TABLE IV
SOFTWARE DEPENDENCIES

Package	**Version**
Deep Learning Framework	
PyTorch	2.9.0
torchaudio	2.9.0
torchvision	0.24.0
Audio Processing	
librosa	0.11.0
soundfile	0.13.1
silero-vad	6.0.0
Machine Learning Utilities	
accelerate	1.11.0
Data Processing	
polars	1.35.1
numpy	2.3.1
Visualization & Monitoring	
matplotlib	3.10.7
wandb	0.22.3

A. Reproduction

To reproduce the results, follow these steps:

1) Repository: Clone the repository at tag paper_v1.0:

```
git clone https://github.com/tad1/
    Mandarin_Pronunciation_Recognition_Project
git checkout paper_v1.0
```

2) Dataset: The dataset files needs to be placed in `./data/source/pg_dataset/`. Verify the correct file versions using these MD5 hashes:

```
d9bd78c33481236e86daef08b8862f08  assesment.csv
933cffd064ca324669996a969a294856  assesment_round2.csv
f53bcea6845883c79d6f6a945d9561bc  experiment.csv
7c71fbc6c1d44f0d19eaef6356a81dde  tones_with_label.xls
```

3) Environment: Install dependencies using uv (recommended) or pip:

```
uv sync
```

The implementation uses CUDA acceleration on GPU (tested on NVIDIA GTX 1060 6GB). Other devices supported by PyTorch can be used by changing target device in the notebook. Development and testing are primarily conducted on Linux. While the code should work on Windows, compatibility is not continuously verified.

4) Execution: Open and run the experiment notebook:

```
src/hypothesis.ipynb
```

Execute all cells sequentially using a notebook environment of your choice. The notebook handles data loading, preprocessing, model training, and evaluation.