DLM: A Decoupled Learning Model for Long-tailed Polyphone Disambiguation in Mandarin

Beibei Gao, Yangsen Zhang*, Ga Xiang and **Yushan Jiang**
School of Information Management, Beijing Information Science and Technology University, China
{gaobeibei, zhangyangsen, xiangga, jiangyushan}@bistu.edu.cn

Abstract

Grapheme-to-phoneme conversion (G2P) is a critical component of the text-to-speech system (TTS), where polyphone disambiguation is the most crucial task. However, polyphone disambiguation datasets often suffer from the long-tail problem, and context learning for polyphonic characters commonly stems from a single dimension. In this paper, we propose a novel model DLM: a **D**ecoupled **L**earning **M**odel for long-tailed polyphone disambiguation in Mandarin. Firstly, DLM decouples representation and classification learnings. It can apply different data samplers for each stage to obtain an optimal training data distribution. This can mitigate the long-tail problem. Secondly, two improved attention mechanisms and a gradual conversion strategy are integrated into the DLM, which achieve transition learning of context from local to global. Finally, to evaluate the effectiveness of DLM, we construct a balanced polyphone disambiguation corpus via in-context learning. Experiments on the benchmark CPP dataset demonstrate that DLM achieves a boosted accuracy of 99.07%. Moreover, DLM improves the disambiguation performance of long-tailed polyphonic characters. For many long-tailed characters, DLM even achieves an accuracy of 100%.

1 Introduction

Grapheme-to-phoneme conversion (G2P) plays an important role in the Chinese text-to-speech system (TTS), aiming to convert Chinese text to Pinyin. One of the biggest challenges in G2P is polyphone disambiguation, whose goal is to select a correct pronunciation for the polyphonic character from a set of candidate pronunciations.

There are a large number of polyphonic characters in Chinese. So various approaches have been proposed to address polyphone disambiguation, primarily focusing on *three problems*: **(i)** How to solve the long-tailed distribution issue in datasets. **(ii)** How to effectively leverage the context of polyphonic characters. **(iii)** How to simply solve the polyphone disambiguation data sparsity problem. Solutions for **(i)** are commonly realized by expanding the tail class corpus (Zhang et al., 2020; Qiang et al., 2022). Although such methods improve the model's classification ability, the failure to learn from the original distribution of the dataset causes a decrease in representation ability. Solutions for **(ii)** often involve extracting sequence features through neural networks (Shan et al., 2016; Zhang et al., 2020) or pre-trained models (Dai et al., 2019), supplemented by part-of-speech (POS) tags. However, most of these methods only concentrate on one dimension of context: global features from the entire sequence or local features from the neighboring characters. For **(iii)**, the reliance on self-built datasets persists, but existing methods using self-built datasets encounter challenges due to insufficient details or implementation difficulties.

Recently, decoupled learning in Computer Vision (CV) tasks divides the learning procedure into representation learning and classification learning (Kang et al., 2019), and demonstrates that the imbalanced distribution of long-tailed datasets is more conducive for learning high-quality representations and re-balanced data is helpful in classification. However, its experimental tests were only conducted on the balanced dataset. The method in (Zhou et al., 2020) proposed a unified Bilateral-Branch Network (BBN), which does not decouple the learning process but applies a cumulative learning strategy to shift the learning from representation to classification. Specifically, in classification learning, the reversed data is used to retrain the classifier, which improves the model's performance on tail classes.

Inspired by decoupled learning and BBN, this paper proposes a model called DLM (A **D**ecoupled **L**earning **M**odel for long-tailed polyphone disam-

*Corresponding author

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 5252–5262
June 16-21, 2024

biguation in Mandarin), which can address the above *three problems*. DLM adopts decoupled learning on the long-tailed dataset to enhance both representation and classification ability. Additionally, DLM integrates two improved attention mechanisms: convolution attention for local features extraction, and linear attention for global features extraction. Between them, employing a gradual conversion strategy akin to human reading, DLM realizes the transition learning of context from local to global. Finally, to evaluate the efficacy of decoupled learning in DLM, we leverage in-context learning (ICL) related to large language models (LLM) to generate a balanced polyphone disambiguation corpus easily and provide a detailed implementation. The contributions can be summarized as follows:

1. We propose a novel model DLM based on decoupled learning to simultaneously enhance representation ability and classification ability (evaluated in section 4.4), effectively alleviating the impact of the long-tail problem (evaluated in section 4.6).

2. We realize the transition learning of context from local to global based on the two improved attention mechanisms and a gradual conversion strategy, outperforming models that only learn context from a single dimension (evaluated in section 4.5).

3. We build a new balanced dataset using ICL to verify the validity of decoupled learning in DLM. Moreover, the incorporation of ICL presents a straightforward and practical approach to corpus generation.

2 Related Work

2.1 Polyphone disambiguation

The earliest approaches to polyphone disambiguation primarily rely on rules, which require linguistic experts to design robust dictionaries (Gou and Luo; Zhang et al., 2002; Dong et al., 2004). However, rule-based methods have been proven labor-intensive and challenging to cover all situations. As the amount of data increases, data-driven approaches have been widely applied to achieve reasonable performance, such as Decision trees (DT) (Liu and Zhou, 2011) and Maximum Entropy (ME) (Liu et al., 2007). Nonetheless, these methods often require extensive feature engineering, demanding a substantial background knowledge. Recently, methods based on deep learning have made great progress. Some studies (Zhang et al., 2020; Cai et al., 2019) adopted BiLSTM layers to obtain contextual features of polyphonic characters. The method in (Shan et al., 2016) encoded information from neighboring characters and the POS of neighboring words. The seq2seq model was also employed in a distantly supervised way (Zhang et al., 2020). With the advent of pre-trained language model (PLM), the method in (Dai et al., 2019) first used BERT[1] for polyphone disambiguation, achieving significant performance improvements. Subsequently, PLMs such as Electra (Clark et al., 2020) and RoBERTa (Liu et al., 2019) have also gained widespread adoption in polyphone disambiguation.

These deep learning approaches, especially PLMs, have demonstrated notable progress, showing their efficacy in handling polyphone disambiguation challenges.

2.2 Long-tail Problem

The long-tail problem refers to the fact that the samples from a few classes occupy a large portion of the dataset (*head class*) while the samples from the majority class only occupy a small portion of the dataset (*tail class*), which directly affects the performance of the model. Therefore, it has received increasing attention in recent years. The solution can be categorized into four types.

Re-sampling The method includes over-sampling and under-sampling. The former aims to balance the dataset by increasing the number of samples in the tail classes while the latter achieves balance by reducing samples in the head class. However, this approach may inadvertently lead to over-fitting on the tail classes, as the model fails to learn from the original data distribution.

Re-weighting The method can be achieved by modifying the loss function to allocate larger weights for the tail classes. However, it may not be capable of real-world long-tailed data and causes optimization difficulty (Mikolov et al., 2013).

Data augmentation The method involves manipulating existing data in a specific way to generate new data. Back-translation-style data augmentation (Qiang et al., 2022) extends the CPP dataset into a balanced one. Semi-supervised learning method (Shi et al., 2021) introduces three types of data augmentation that can be applied to texts with a target character.

[1] https://huggingface.co/bert-base-chinese

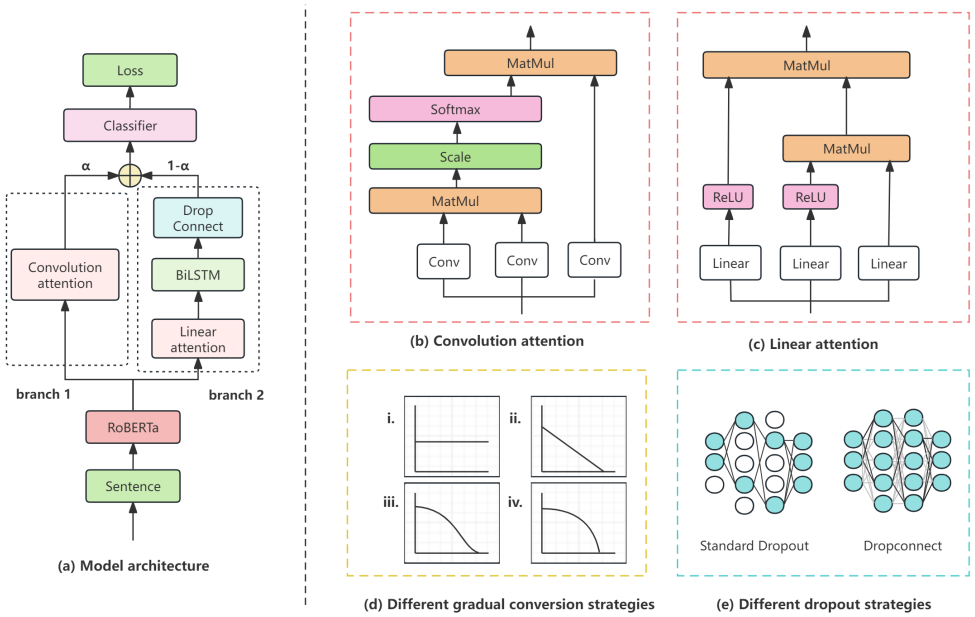

Figure 1: (a) The architecture of our proposed DLM; (b) Convolution attention mechanism; (c) Linear attention mechanism; (d) Different gradual conversion strategies: i. *equal weight*, ii. *linear decay*, iii. *cosine decay*, iv. *parabolic decay*; (e) Different dropout strategies.

Decoupled learning Decoupling representation and classification learning has shown a significant ability to handle long-tail problems (Kang et al., 2019). It divides the training process into two single stages. In the first stage, the model undergoes training on the original imbalanced data, followed by a second stage in which the classifier is retrained using re-balanced data.

2.3 In-context learning

Large language models (LLM), particularly GPT-3, have demonstrated remarkable in-context learning (ICL) capabilities (Brown et al., 2020). ICL operates with just a few input-output pairs dubbed demonstrations, enabling the model to predict outputs (*few-shot*). Additionally, ICL can even handle previously unseen tasks' predication without any demonstration context (*zero-shot*). In our study, we primarily utilize in-context few-shot learning.

The prompt for ICL mainly contains two components: one is a demonstration context, which comprises input-output pairs articulated in natural language templates, and the other is a new query. The prompt will be fed into the LLM to generate a prediction. Based on the demonstration context, LLMs can recognize and perform a new task without any parameter updates (Zhao et al., 2023), which proves particularly advantageous in the generation of polyphone disambiguation corpus.

3 Method

Our proposed DLM is illustrated in Figure 1(a). The input sequence is initially processed by RoBERTa to obtain the output vector h. Subsequently, h passes through two branches. The first branch employs convolution attention, emphasizing the sequence local features. The second branch is a combination of linear attention and BiLSTM, concentrating on sequence global features. And the Dropconnect is integrated into the second branch to prevent over-fitting. The outputs from these two branches are combined to derive the final output, where the weights for the two branches are obtained from α in the gradual conversion strategy. Moreover, DLM adopts decoupled learning. The model is trained to learn representation in the first stage and the classifier is retrained in the second stage.

3.1 Decoupled Learning

3.1.1 Data Sampler

In traditional deep learning, input data is typically sourced from a uniform sampler. In this case, each example in the training set is sampled only once, resulting in under-fitting on the tail classes when the dataset follows a long-tailed distribution. To ad-

dress this issue, our method incorporates three data samplers (Zhou et al., 2020) in decoupled learning.

Notation. We define the notations used in the paper. Let $K = \{k_1, k_2, \cdots, k_n\}$ denotes the classes in the training set, with k_i representing the ith class. The variable m_i signifies the number of examples in class k_i, and m_{max} represents the maximum of all m_i. Additionally, let w_i denote the weight assigned to class k_i. The total number of training examples is $\sum_{i=1}^{n} m_i$.

The data samplers employed are as follows: **Uniform sampler.** Sample data according to the original distribution. Each example in the training set is sampled with equal probability and each example is sampled once. **Balanced sampler.** Select a class k_i randomly from the set of classes K, and then pick one example randomly from it. **Reverse sampler.** The class is re-weighted by the number of examples within it. The smaller the number of examples in a class, the greater the probability assigned to that class. The sampled probability of an example belonging to class k_j is inversely proportional to m_j, as shown in Eq 1, where $w_i = \frac{m_{max}}{m_i}$

$$p_j = \frac{w_j}{\sum_{i=1}^{n} w_i} \tag{1}$$

3.1.2 Decoupled process

Typically, the model's representation ability and classification ability are trained through a unified process. However, since the original data distribution aids better representation and re-balanced data is more conducive for the classifier, DLM decouples representation learning and classification learning. In the first stage, the original distribution of the dataset is obtained using a uniform sampler. This allows the model to learn better representation. In the second stage, the reverse sampler is employed, enabling the model to relearn the tail classes. Specifically, the backbone parameters need to be frozen in the second stage and only the classifier's parameters need updating.

3.2 Improved attention

With the proposal of Transformer (Vaswani et al., 2017), self-attention has gained widespread adoption. However, its computational complexity is squarely related to the sequence's length, leading to a relatively high computational cost. Moreover, previous methods for polyphone disambiguation only learn context from one dimension, neglecting the local or global features of the sequence. This oversight precludes the attainment of transition learning of context from local to global. To solve these issues, we incorporate two improved attention mechanisms: convolution attention and linear attention, which reduce the computational cost and focus on local and global features separately.

3.2.1 Convolution attention

The sliding window in Convolutional Neural Networks excels in capturing local features. In contrast to recurrent networks, the convolutional approach stands out in discovering compositional structures in sequences (Zhang et al., 2020). As shown in Figure 1(b), the convolution layers enable the model to capture the local features of context. For the input sequence $X \in \mathbb{R}^{N \times d}$, the corresponding representations matrices Q, K, V are computed as:

$$Q = XW_{conv}^{Q}, K = XW_{conv}^{K}, V = XW_{conv}^{V} \tag{2}$$

where $W_{conv}^{Q}, W_{conv}^{K}, W_{conv}^{V} \in \mathbb{R}^{d \times d}$. The convolution attention can be calculated through

$$Catt(Q, K, V)_i = \sum_{j=1}^{N} softmax(\frac{Q_i K_j^T}{\sqrt{d}}) V_j \tag{3}$$

Furthermore, replacing the linear layers in the attention mechanism with the convolution layers can reduce the computational cost.

3.2.2 Linear attention

The traditional attention function maps a query and a set of key-value pairs to an output. It calculates relations between each token in the sequence. For an input sequence $X \in \mathbb{R}^{N \times d}$, the corresponding representations matrices Q, K, V are computed as:

$$Q = XW^{Q}, K = XW^{K}, V = XW^{V} \tag{4}$$

Then the self-attention is computed as:

$$Att(Q, K, V)_i = \frac{\sum_{j=1}^{N} sim(Q_i, K_j)}{\sum_{j=1}^{N} sim(Q_i, K_j)} V_j \tag{5}$$

When the similarity function is defined as $sim(q, k) = \exp(\frac{qk^T}{\sqrt{d}})$, the attention mechanism is commonly known as *softmax attention*. Its computational complexity is $O(N^2 d)$ due to the order of calculation being $(QK)V$, leading to a high computational cost.

In contrast, as depicted in Figure 2, linear attention is calculated in the order of $Q(KV)$. Utilized the similarity function $sim(q, k) = \phi(q)\phi(k^T)$, the self-attention can be rewritten as:

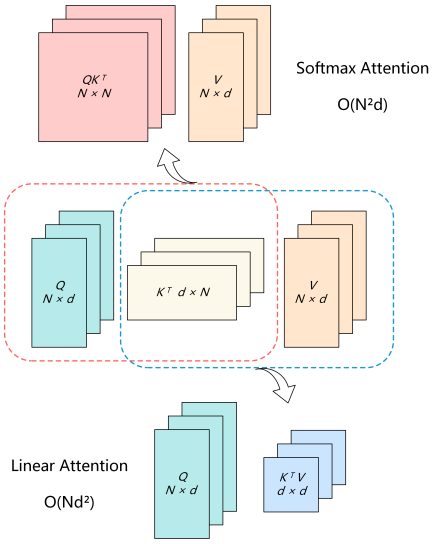

Figure 2: The different computational complexity of softmax attention and linear attention.

$$Att(Q,K,V)_i = \frac{\sum_{j=1}^{N}\phi(Q_i)\phi(K_j^T)}{\sum_{j=1}^{N}\phi(Q_i)\phi(K_j^T)}V_j \quad (6)$$

$$= \frac{\phi(Q_i)\sum_{j=1}^{N}\phi(K_j^T)}{\phi(Q_i)\sum_{j=1}^{N}\phi(K_j^T)}V_j \quad (7)$$

Let $\phi(x) = ReLU(x)$, the formulation of linear attention is

$$Latt(Q,K,V)_i = \frac{ReLU(Q_i)\sum_{j=1}^{N}ReLU(K_j^T)}{ReLU(Q_i)\sum_{j=1}^{N}ReLU(K_j^T)}V_j \quad (8)$$

Linear attention reduces the computational complexity to $O(Nd^2)$, making it more feasible for processing large-scale data. The architecture of linear attention is shown in Figure 1(c). In DLM, linear attention is combined with BiLSTM as a separate branch to learn context globally.

As shown in Figure 1(d), the α in different gradual conversion strategies will be evaluated in section 4.5. The combination function of the two branches is calculated by Eq 9, which can realize the transition learning of context from local to global.

$$CLSInput = \alpha v_{branch1} + (1-\alpha)v_{branch2} \quad (9)$$

3.3 In-context Learning to generate corpus

The CPP dataset (Park and Lee, 2020) is an imbalanced dataset. So we use ICL to build a balanced dataset named BCP (**B**alanced **C**hinese **P**olyphone dataset). As shown in Figure 3, ICL takes the demonstration containing some input-output pairs and a new query into LLM to get the generated corpus corresponding to the polyphonic character in that query. Specifically, the input consists of a word containing the polyphonic character, a number n of generated sentences, and the polyphonic character. The output is n sentences containing the polyphonic character, and the polyphonic character is connected to the end of each sentence.

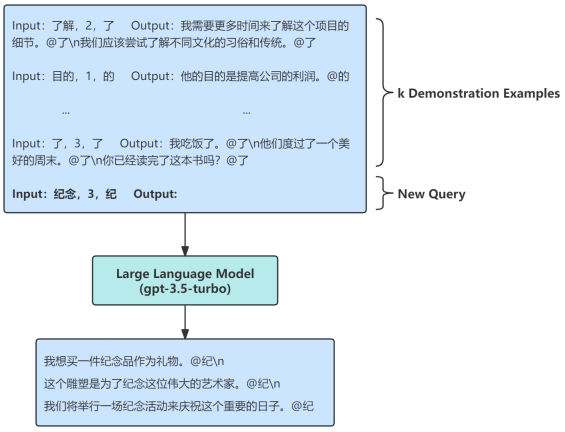

Figure 3: In-context learning to generate corpus.

First, pynlpir[2] is used to perform word segmentation on sentences within the CPP training set. The word segmentation unit (WSU) containing the polyphonic character may be single characters $char_i$ or words $word_i$. Every polyphonic character has multiple pronunciations, and every pronunciation p_i may map with multiple WSUs. We count the WSU candidate set C_{p_i} of each pronunciation. For instance, the polyphonic character "朝" has pronunciations "chao2" and "zhao1", where the different numbers represent different tones of pinyin. WSUs for "chao2" include "朝向"(toward) and "王朝"(dynasty), notably C_{chao2} ={"朝向", "王朝"}. Secondly, the element e in C_{p_i}, the number of sentences n and the polyphonic character in e are filled in the new query. Finally, the prompt with the new query is fed into LLM to generate n sentences of the current polyphonic character. The labels of them are the corresponding pronunciation p_i.

For most words that have a fixed pronunciation for polyphonic characters, labels for their generated sentences are directly assigned as p_i. For example, the pronunciation of "朝" in the word "朝向" (toward) is different from another word "朝气" (vigor), they are marked as "chao2" and "zhao1"

[2] https://github.com/tsroten/pynlpir

respectively. However, some polyphonic characters' WSUs are single characters or special words with more than one pronunciation. Labels for these sentences cannot be assigned directly, such as a single character "乐", pronounced "le4" or "yue4", and the word "朝阳", pronounced as "zhao1 yang" or "chao2 yang". Therefore, the labels of these generated sentences will be assigned manually.

4 Experiments

4.1 Dataset

To assess the effectiveness of the proposed method, we utilize an open benchmark named CPP Dataset (Chinese Polyphone with Pinyin) [3] for training and testing. As a supplement, we use our ICL-generate BCP dataset to verify the validity of decoupled learning.

CPP dataset The dataset comprises 99,264 sentences and each sentence contains one or more polyphonic characters but only one is annotated. It contains 623 different polyphonic characters. More details can be found in (Park and Lee, 2020).

BCP dataset The dataset consists of 9,820 sentences, each of which includes a polyphonic character with its correct pinyin. Specifically, BCP is a balanced dataset.

4.2 Thresholds of long-tailed characters

The long-tailed characters in the CPP dataset are selected along two dimensions. The following are some notes used to define the thresholds. In the CPP training set, N_c denotes the number of examples of the character c, and N_{max} represents the maximum among all N_c. The variable $p(c) = [p_1, p_2, \cdots, p_k]$ signifies the pronunciation candidates of the character c. Let $N_{p(c)} = [n_1, n_2, \cdots, n_k]$, where n_i denotes the number of examples of the pronunciation p_i. Obviously, $N_c = sum(N_{p(c)})$

Define long-tailed characters according to the following two rules: **(i)** At the character level, a character c is flagged as long-tail when $N_c \leq threshold_1$. **(ii)** At the pronunciation level, a character c is flagged as long-tail when $\frac{min(N_{p(c)})}{max(N_{p(c)})} \leq threshold_2$. Note that $min(N_{p(c)}) > 0$.

In this paper, we set $threshold_1 = \frac{N_{max}}{4}$ and $threshold_2 = 0.2$. In total, there are 299 long-tailed characters in the CPP dataset and we define this set as **LTCPP** (long-tailed CPP dataset).

[3] https://github.com/kakaobrain/g2pm/tree/master/data

4.3 Experimental setup and Factor Analysis

We adopt the pre-trained model RoBERTa, which consists of 12 Transformer layers, with 768 hidden dimensions, to extract semantic features from raw Chinese character sequences. The optimizer is Adam (Kingma and Ba, 2014). We set the learning rate to 5e-5. The batch size is 512. To prevent over-fitting, we utilize DropConnect (Wan et al., 2013) instead of Dropout (Hinton et al., 2012). Each stage of training involves 1000 iterations, and validation is conducted after every 10 iterations. The model achieving the best validation accuracy is selected for testing. The accuracy of polyphone disambiguation is used as the evaluation metric in all experiments.

Evaluation on dropout strategy. To prevent our model from over-fitting, we explore two different strategies to improve the model's generalization ability: Standard Dropout and DropConnect. As shown in Figure 1(e), while Standard Dropout sets a randomly selected subset of activations to zero within each layer, DropConnect sets a randomly selected subset of *weights* in the network to zero. The results in Table 1 demonstrate that DropConnect yields better results because it is effective in regularizing large neural network models.

Dropout	CPP Acc(%)	BCP Acc(%)
Standard	99.03	91.04
DropConnect	**99.07**	**91.14**

Table 1: Evaluation on different dropout strategies

Evaluation on the kernel size of convolution attention. To explore how different kernel sizes in convolution attention influence the scale of local features, we evaluate different kernel sizes. As demonstrated in Table 2, the optimal performance is achieved at $N = 5$. This suggests that larger or smaller scale sizes may not contribute to the effective extraction of local features.

Kernel Size	CPP Acc(%)	BCP Acc(%)
3	98.58	87.88
5	**99.07**	**91.14**
7	99.02	89.21

Table 2: Evaluation on different kernel sizes in convolution attention

4.4 Evaluation on Decoupled Learning

We decoupled the training process of DLM: learning representation in the first stage and classification in the second stage. So we explore the best sampler for each stage and demonstrate the effectiveness of decoupled learning for long-tailed data.

Best sampler for representation learning. To further evaluate the model's representation ability learned on the CPP dataset, we continue the second stage of training on the BCP dataset (Note that the BCP dataset is only used for the evaluation of best sampler but not for the training of DLM). The model obtained from the first stage under three different samplers serves as the initial model. Specifically, the uniform sampler in stage 2 could also reach a balanced distribution since the BCP dataset is balanced. As shown in Table 3, fixing the sampler for the second stage, the best representation ability is obtained when using the uniform sampler in the first stage. This indicates that the original distribution of the long-tailed dataset is more conducive to the model's representation learning.

stage 1	stage 2	Accuracy(%)
uniform	uniform	**87.88**
balanced	uniform	87.27
reverse	uniform	87.17

Table 3: Different samplers for representation learning

Best sampler for classification learning. After identifying the best sampler for representation learning, we fix the sampler of first stage to uniform sampler and explore the classification ability learned with different samplers in the second stage. The CPP dataset is used here because imbalanced data is the way to show the difference in sampling data under three different samplers. As shown in Table 4, the results indicate that the reverse sampler contributes to the model achieving the best classification ability. This is attributed to the reverse sampler, which allows the model's classifier to be retrained with the tail classes, enhancing the model's performance on them.

stage 1	stage 2	Accuracy(%)
uniform	uniform	99.02
uniform	balanced	99.04
uniform	**reverse**	**99.07**

Table 4: Different samplers for classification learning

4.5 Evaluation on the two improved attention mechanisms

To demonstrate the effectiveness of the two improved attention mechanisms, we conduct ablation experiments. The details are as follows.

- **w/o branch1:** Remove convolution attention from DLM, which leads the model to only focus on the global features of sequences.
- **w/o branch2:** Removed the branch that combines BiLSTM and linear attention from DLM, which leads the model to only focus on local features of sequences after RoBERTa.
- **DLM:** Our proposed model.

Model	CPP Acc(%)	LTCPP Acc(%)
w/o branch1	85.51	79.92
w/o branch2	98.82	98.56
DLM	**99.07**	**98.88**

Table 5: Ablation study

As shown in Table 5, *w/o branch1* performs the worst among the three. Because the entire model has no module to learn the local features of the sequence. In contrast, *w/o branch2* performs better, even though the branch for learning global features is removed, RoBERTa itself can focus on global features through vanilla attention to some extent. In summary, focusing only on local features or global features is not as comprehensive as DLM, which achieves transition learning of context from local to global.

Gradual Conversion	α	CPP Acc(%)	BCP Acc(%)
equal	0.5	98.86	**91.45**
linear decay	$1-\frac{T}{T_{max}}$	99.04	90.63
cosine decay	$cos(\frac{T}{T_{max}}\cdot\frac{\pi}{2})$	**99.07**	91.14
parabolic decay	$1-(\frac{T}{T_{max}})^2$	99.05	90.63

Table 6: Evaluation on different gradual conversion strategies, T is the current epoch and T_{max} is the total epoch

To further explore the gradual conversion strategies between the two branches, as shown in Figure 1(d), we evaluate four strategies: equal weight, linear decay, cosine decay, and parabolic decay (Zhou et al., 2020). As shown in Table 6, gradual conversion strategies (linear, cosine, and parabolic)

outperform on the imbalanced dataset while the constant strategy (equal weight) outperforms on the balanced dataset, which supports our proposal that DLM should gradually shift context learning from local to global when solving long-tail problems. Among these gradual conversion strategies, cosine decay exhibits the best performance.

4.6 Result and Analysis

4.6.1 Evaluation of different systems

DLM is trained on the CPP dataset, using the uniform sampler in the first stage and the reverse sampler in the second stage. We compare DLM and other systems which are also trained on the CPP dataset.

No.	Model	Accuracy(%)
1	g2pM(BiLSTM)	97.31
2	g2pM(BERT)	97.85
3	BERT with LSTM	98.04
4	Distant supervision	97.51
5	BERT-MFA	99.06
6	ELECTRA with MARC	98.81
7	g2pW	**99.08**
8	DLM	99.07

Table 7: The accuracy of different systems

As depicted in Table 7, DLM demonstrates superior performance when compared to models that overlook the local features of sequences, such as g2pM(BiLSTM), g2pM(BERT) (Park and Lee, 2020), and BERT with LSTM (Zhang, 2021). Additionally, several models excel at capturing local features. Distant supervision (Zhang et al., 2020) utilized a CNN-based approach, while BERT-MFA (Li et al., 2021) integrates window-based attention to improve neighboring information extraction. However, these models lack a robust approach (e.g. decoupled learning) to effectively address long-tail problems and they lack the transition learning of context from local to global. Although ELECTRA with MARC (Gao et al., 2022) employs decoupled learning, it also exhibits deficiencies in the learning of context. In summary, our proposed DLM achieves an accuracy of 99.07%, which is better than almost all other models.

4.6.2 Evaluation of long-tailed characters

We conduct testing on all 299 characters in the LTCPP dataset. As illustrated in Figure 4, 266 characters exhibit an accuracy exceeding 95%, while 292 characters demonstrate an accuracy surpassing 80%. Notably, 262 characters achieve a perfect accuracy of 100%, underscoring DLM's exceptional performance with long-tailed characters. Furthermore, the comparison of DLM and g2pW (Chen et al., 2022) on long-tailed characters also reveals DLM's superior performance when tackling the long-tail problem.

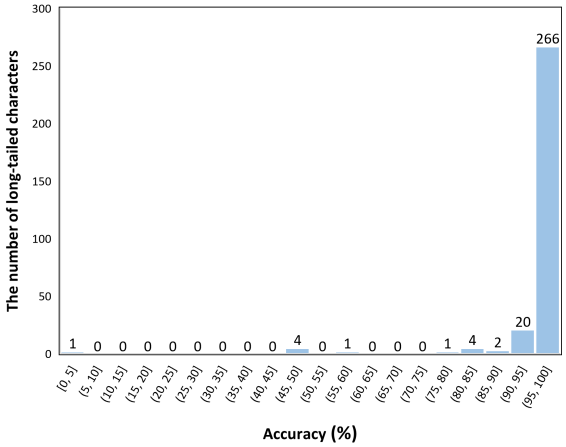

Figure 4: The accuracy distribution of characters in the LTCPP dataset.

We classify these characters into three categories according to accuracy: well-performing polyphones ($acc \in (95\%, 100\%]$), average-performing polyphones ($acc \in (60\%, 95\%]$), and poorly-performing polyphones ($acc \in [0, 60\%]$). The following is the analysis of the different performances.

Char	polyphone	g2pW	DLM
捋	lv3:7 luo1:5	50.00	**100.00**
偻	lou2:2 lv:6	100.00	**100.00**
臭	chou4:150 xiu4:11	85.71	**95.24**
咽	yan1:124 yan4:31 ye4:5	65.00	**100.00**
漂	piao4:37 piao1:110 piao3:13	90.48	**95.24**
着	zhe5:151 zhao2:2 zhuo2:3	90.00	**100.00**

Table 8: Well-performing long-tailed characters

Well-performing polyphones. Table 8 exhibits some examples of well-performing polyphones. At the character level, DLM achieves an accuracy of 100% on some very sparse data in the training set, such as "捋" and "偻". At the pronunciation level, DLM also exhibits an accuracy exceeding 95% on characters suffering from long-tailed pronunciation

distribution, such as "臭" and "漂". These may be attributed to decoupled learning. In the second stage of training, DLM is retrained by the reversed CPP dataset, which increases the chances of DLM learning about sparse data. Besides, the transition learning of context from local to global makes DLM learn sparse data sufficiently.

Char	polyphone	g2pW	DLM
酢	cu4:38 zuo4:2	**100.00**	83.33
镐	hao4:153 gao3:4	75.00	**90.00**
泡	pao4:131 pao1:12	83.33	**94.40**
蠡	li3:135 li2:8	66.67	**94.40**
薄	bo2:81 bao2:42 bo4:5	58.82	**94.12**

Table 9: Average-performing long-tailed characters

Average-performing polyphones. As shown in Table 9, here are some characters that haven't been adequately learned by DLM. Upon analysis, it's evident that DLM has grasped all pronunciations of the characters. Despite some cases of wrong choice causing a decrease in accuracy, DLM has taken into account even the rarer category of pronunciations and the accuracy of every character still achieves at least 75%.

Char	polyphone	g2pW	DLM
铤	ting3:15	50.00	50.00
拧	ning3:28 ning2:5	60.00	60.00
唉	ai4:8 ai1:1	50.00	50.00
莳	shi2:3 shi4:6	0.00	0.00

Table 10: Poorly-performing long-tailed characters

Poorly-performing polyphones. Despite DLM achieving superior performance on most long-tailed characters, there are instances where the accuracy is less than 60%. Examples of such cases are illustrated in Table 10. Through analysis, three main reasons are identified. **(i)** In the CPP dataset, certain characters' pronunciations appear in the test set but not in the training set, rendering DLM incapable of selecting the correct pronunciation based on the parametric knowledge acquired during training. For example, the character "铤" is only pronounced as "ting3" in the training set but additionally as "ding4" in the test set. **(ii)** The performance of DLM is constrained for characters that suffer from long-tail problems at both the character and pronunciation levels. During the first stage, DLM learns based on the dataset's original distribution, resulting in insufficient learning of sparse characters. Although these characters are re-learned during the second stage via the reverse sampler, there remains a possibility that a few characters are not learned adequately by relying primarily on the second stage, such as "拧" and "唉". **(iii)** The reversed data sampler utilized in this paper is based on the class of pronunciations. Despite the limited samples of pronunciations such as "shi2" and "shi4" for the character "莳", other characters might share the same pronunciations, resulting in a lot of samples of "shi2" and "shi4" across the dataset. Therefore, the reverse sampler might fail to correctly pick up samples of the character "莳", which leads to insufficient learning in the second stage as well as the first stage.

5 Conclusions

In this paper, we propose a novel model DLM. In experiments, we validate the effectiveness of decoupled learning to solve the long-tail problem and prove that the original distribution of the long-tailed dataset enhances the model's representation ability, while the re-balanced dataset contributes to classification ability. Besides, we further confirm the efficacy of two improved attention mechanisms, particularly the convolution attention proposed in this paper. We find the significance of employing a gradual conversion strategy between the two improved attention to realize the transition learning of context from local to global when tackling the imbalanced dataset. Finally, a simple approach for generating the polyphone disambiguation corpus via in-context learning is proposed and we build a balanced corpus to verify the validity of decoupled learning. Overall, DLM demonstrates outstanding performance on the CPP dataset, especially achieving superior performance in handling long-tailed polyphonic characters.

Limitations

Our work encounters some limitations. Firstly, a few cases need human involvement during corpus generation. These cases include scenarios where polyphonic characters are segmented as a single character or words that possess multiple pronunciations during word segmentation. Secondly, to address the specific issue of long-tailed polyphone

disambiguation, RoBERTa is utilized as a component of DLM. However, as a part of the front end of the TTS system, the inference speed of DLM still needs to be improved since the parameter of RoBERTa is relatively large. Thirdly, the label for the polyphone disambiguation task should not just be pronunciations, but character-pronunciation pairs. In future work, we will optimize the corpus-creation approach to accommodate all cases, aiming to minimize the requirement for human intervention. Moreover, we will explore reducing the number of the model's parameters and improving the inference speed. Finally, we will build character-pronunciation pairs as labels for polyphone disambiguation tasks.

Acknowledgements

This work was supported by Beijing Natural Science Foundation (L233008). Besides, the authors thank all reviewers for their valuable comments.

References

Jiawen Zhang, Yuanyuan Zhao, Jiaqi Zhu, and Jinba Xiao. 2020. Distant supervision for polyphone disambiguation in mandarin chinese. In *INTERSPEECH*, pages 1753–1757.

Chunyu Qiang, Peng Yang, Hao Che, Jinba Xiao, Xiaorui Wang, and Zhongyuan Wang. 2022. Back-translation-style data augmentation for mandarin chinese polyphone disambiguation. In *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 1915–1919. IEEE.

Changhao Shan, Lei Xie, and Kaisheng Yao. 2016. A bi-directional lstm approach for polyphone disambiguation in mandarin chinese. In *2016 10th International Symposium on Chinese Spoken Language Processing (ISCSLP)*, pages 1–5. IEEE.

Haiteng Zhang, Huashan Pan, and Xiulin Li. 2020. A mask-based model for mandarin chinese polyphone disambiguation. In *INTERSPEECH*, pages 1728–1732.

Dongyang Dai, Zhiyong Wu, Shiyin Kang, Xixin Wu, Jia Jia, Dan Su, Dong Yu, and Helen Meng. 2019. Disambiguation of chinese polyphones in an end-to-end framework with semantic features extracted by pre-trained bert. In *Interspeech*, pages 2090–2094.

Bingyi Kang, Saining Xie, Marcus Rohrbach, Zhicheng Yan, Albert Gordo, Jiashi Feng, and Yannis Kalantidis. 2019. Decoupling representation and classifier for long-tailed recognition. *arXiv preprint arXiv:1910.09217*.

Boyan Zhou, Quan Cui, Xiu-Shen Wei, and Zhao-Min Chen. 2020. Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9719–9728.

Daju Gou and Wanbo Luo. Processing of polyphone character in chinese tts system. *Chinese Information*, 1:33–36.

Zi-Rong Zhang, Min Chu, and Eric Chang. 2002. An efficient way to learn rules for grapheme-to-phoneme conversion in chinese. In *International symposium on Chinese spoken language processing*.

Honghui Dong, Jianhua Tao, and Bo Xu. 2004. Grapheme-to-phoneme conversion in chinese tts system. In *2004 International Symposium on Chinese Spoken Language Processing*, pages 165–168. IEEE.

Fangzhou Liu and You Zhou. 2011. Polyphone disambiguation based on tree-guided tbl. *Jisuanji Gongcheng yu Yingyong(Computer Engineering and Applications)*, 47(12):137–140.

Fang-zhou Liu, Qin Shi, and Jianhua Tao. 2007. Maximum entropy based homograph disambiguation. *NCMMSC2007*.

Zexin Cai, Yaogen Yang, Chuxiong Zhang, Xiaoyi Qin, and Ming Li. 2019. Polyphone disambiguation for mandarin chinese using conditional neural network with multi-level embedding features. *arXiv preprint arXiv:1907.01749*.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. 2020. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv preprint arXiv:2003.10555*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

Yi Shi, Congyi Wang, Yu Chen, and Bin Wang. 2021. Polyphone disambiguition in mandarin chinese with semi-supervised learning. *arXiv preprint arXiv:2102.00621*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Kyubyong Park and Seanie Lee. 2020. g2pm: A neural grapheme-to-phoneme conversion package for mandarin chinese based on a new open benchmark dataset. *arXiv preprint arXiv:2004.03136*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Li Wan, Matthew Zeiler, Sixin Zhang, Yann Le Cun, and Rob Fergus. 2013. Regularization of neural networks using dropconnect. In *International conference on machine learning*, pages 1058–1066. PMLR.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*.

Haiteng Zhang. 2021. Polyphone disambiguation in chinese by using flat. In *Interspeech*, pages 4099–4103.

Junjie Li, Zhiyu Zhang, Minchuan Chen, Jun Ma, Shaojun Wang, and Jing Xiao. 2021. Improving polyphone disambiguation for mandarin chinese by combining mix-pooling strategy and window-based attention. In *Interspeech*, pages 4104–4108.

Yu Gao, Yijin Xiong, and Jiancheng. Ye. 2022. Double-weighted disambiguation algorithm for long-tail polyphone problem. *Journal of Chinese Information Processing*, 36(11):169–176.

Yi-Chang Chen, Yu-Chuan Chang, Yen-Cheng Chang, and Yi-Ren Yeh. 2022. g2pw: A conditional weighted softmax bert for polyphone disambiguation in mandarin. *Proc. Interspeech 2022*.