AttnInput: Advancing Context-Aware Pinyin Input with Efficient Language Model Integration

Anonymous ACL submission

Abstract

Pinyin input methods are essential for typing Chinese characters, yet existing approaches struggle to balance accuracy with computational efficiency when integrating large language models (LLMs). This paper introduces AttnInput, a novel framework that enhances pinyin input performance through lightweight language model adaptation. By integrating pinyin features directly into the model's inference process via a parameter-efficient side network, AttnInput eliminates the need for costly full-model fine-tuning while significantly improving prediction accuracy. The method employs constrained training and inference strategies to enforce phonetic alignment, reducing ambiguity in pinyin sequences. Experiments demonstrate state-of-the-art results across varying context and pinyin lengths, with a 20–34% accuracy improvement over existing methods on long sequences. AttnInput achieves these gains while maintaining linear computational complexity, enabling stable latency and low resource consumption even for extended contexts. The framework reduces training costs by over 50% compared to conventional fine-tuning approaches, showcasing practical advantages for edge-side deployment and scalable language model integration.

1 Introduction

Pinyin Input Method Engine (IME) allows users to input Chinese characters using a standard keyboard. Pinyin is the official romanization system for Chinese, which represents the pronunciation of Chinese characters using the Latin alphabet.

Pinyin input methods convert Romanized phonetic inputs into Chinese characters using pinyin. The primary approaches include **perfect pinyin**, which requires full syllable input (e.g., "zhongguo" for "中国"), and **abbreviated pinyin**, which uses initial letters (e.g., "zg" for the same phrase). While perfect pinyin reduces homophone ambiguity by specifying complete pronunciation, abbreviated pinyin prioritizes typing speed at the risk of increased character selection complexity.

The advent of GPT models has spurred research into applying large language models to input method engines. As illustrated in Figure 1(b), most of the previous research that achieve state-of-the-art performance like PinyinGPT-Concat (Tan et al., 2022) and GeneInput (Ding et al., 2023) simply concatenate the context and the pinyin sequence to form the prompt for the language model. However, inserting pinyin sequences disrupts the semantic flow between the prompt and target text, and poses challenges for effectively leveraging pretrained large language models, as their training objective primarily focuses on predicting the next token. Furthermore, these models are trained in a supervised fine-tuning(SFT) manner, indicating that only a small number of pinyin information in each training sample is learned, leading to a need for extensive training resources and difficulty in increasing context length. Our work confirms that concat-based method disrupts semantic consistency and leads to inefficient training. As illustrated in Figure 1(c), PinyinGPT-Embed (Tan et al., 2022) demonstrates superior training efficiency, however, its performance remains suboptimal due to its inability to fully utilize the pinyin information in the input during inference. Moreover, previous approaches have all relied on transformer models exhibiting quadratic time complexity, where both latency and GPU memory consumption escalate with increasing context length, resulting in limited context capacity. This inherent contradiction with the essential requirements of input methods - low latency, minimal resource usage, and extended context processing - fundamentally hinders further model scaling.

The Receptance Weighted Key Value (RWKV) model (Peng et al., 2023, 2024) represents a groundbreaking large language architecture employing

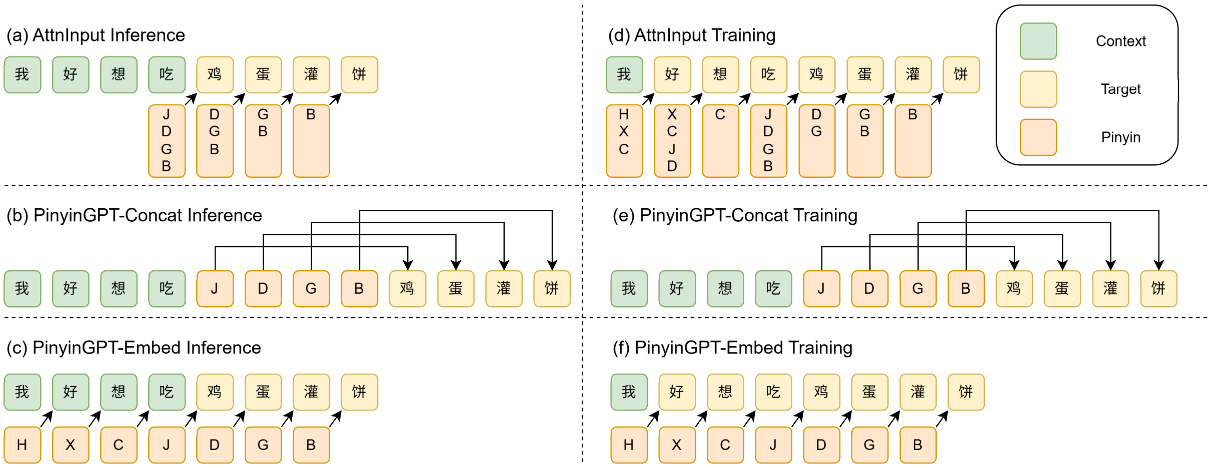

Figure 1: Illustration of the inference and training process of pinyin IMEs. The abbreviated pinyin of the Chinese characters "我好想吃鸡蛋灌饼"(I really want to eat an egg pancake) shown in the picture is "W H X C J D G B". See Appendix B for detailed information.

linear self-attention mechanisms, which maintains constant latency and fixed resource allocation while enabling "infinite"[1]-length context processing - achieving performance parity with conventional transformers, making it particularly well-suited for input method applications.

We explored the direct use of pinyin-constrained beam search outputs from RWKV6 model as candidate word lists at first, resulting in substantial performance improvement. Nevertheless, this method abandons pinyin information, which leads to a higher probability of prematurely pruning the correct answer during the initial stages of beam search, particularly when the target's prefix tokens are infrequent. This presents opportunities for further improvement.

Therefore, we propose a novel approach named AttnInput to leverage large language models for input method engine. It addresses the semantic discontinuity of previous methods by integrating pinyin information directly into the RWKV6's internal state through a lightweight side network. This side network uses ladder side-tuning (Sung et al., 2022), attaching to the main model without requiring backpropagation through it, thus saving computational resources. The model is pre-trained, unlike many previous approaches using supervised fine-tuning, leading to more efficient use of training data and lower computational cost. During inference, the model receives both the context and a sequence of pinyin, processing them together to predict the corresponding Chinese characters. The use of RWKV allows for efficient handling of long contexts and pinyin sequences. Pinyin-constrained training and beam search are employed to further improve accuracy by restricting predictions to characters matching the given pinyin. Notably, the principles of AttnInput are generalizable to other non-Latin scripts, presenting a promising solution for enhancing text input systems globally. AttnInput offers the following advantages:

- To the best of our knowledge, it achieves state-of-the-art performance on abbreviated pinyin.
- In the training stage, it requires significantly less computational resources and training data compared to previous work.
- The model exhibits linear time complexity, maintaining stable inference latency and consistent resource consumption that remain unaffected by increasing context lengths, thereby presenting an optimal architecture for edge-side deployment scenarios.
- The model demonstrates exceptional context length generalization capability, maintaining robust performance even when processing input sequences that far exceed its original training scope, showcasing remarkable extrapolation potential in long-context scenarios.

[1]The authors of RWKV6 claim that RWKV6 has "infinite" context length on https://rwkv.com/ due to the observed continuous decrease in loss as the context length extends beyond the context length used during training. However, this does not necessarily imply that RWKV6 outperforms Transformer-based models in long-text understanding or retrieval tasks.

2 Task

The input of pinyin input method includes a sequence of Chinese characters $W = \{w_1, ..., w_n\}$ representing the context and a sequence of pinyin $P = \{p_1, ..., p_m\}$. The pinyin might be abbreviated pinyin or perfect pinyin. The output is a sequence of Chinese characters $O = \{w_{n+1}, ..., w_{n+m}\}$. The output sequence follows the input sequence semantically, and the pronunciation corresponds to the pinyin.

3 Methods

In this section, we first introduce standard RWKV6 large language model. The vanilla RWKV6 model exhibits competitive performance compared to existing state-of-the-art models in IME tasks, even when ignoring pinyin information during inference. Afterward, we will introduce the new model named AttnInput, which can leverage enriched pinyin information during inference while maintaining efficient training and inference performance.

3.1 Enhancing HMM-Based Pinyin IME with LLMs

Traditional HMM(Hidden Markov Model)-based pinyin input methods (Chen and Lee, 2000) aim to predict the most probable Chinese character sequence O given a context W and pinyin sequence P. This is formulated using Bayes' rule:

$$\begin{aligned} O^* &= \arg\max_O \Pr(O \mid P, W) \\ &= \arg\max_O \Pr(P \mid O, W) \cdot \Pr(O \mid W) \end{aligned} \quad (1)$$

where $\Pr(O \mid W)$ is the language model probability, and $\Pr(P \mid O, W)$ is the typing model.

In this section, we propose replacing the traditional N-gram language model with a large language model (LLM) to estimate $\Pr(O \mid W)$, while simplifying $\Pr(P \mid O, W)$ to a binary indicator function that enforces pinyin constraints. Specifically:

Language Model The LLM computes the probability of the output sequence O conditioned on the context W:

$$\Pr(O \mid W) = \prod_{i=1}^{m} \Pr(w_{n+i} \mid W, w_{n+1}, ..., w_{n+i-1}) \quad (2)$$

Pinyin Constraint The typing model $\Pr(P \mid O, W)$ is simplified to enforce exact pinyin matching. The constraint is:

$$\Pr(P \mid O, W) = \prod_{i=1}^{m} \mathbb{I}\left(w_{n+i} \in V_{p_i}\right) \quad (3)$$

where $\mathbb{I}(\cdot)$ is an indicator function (1 if true, 0 otherwise) and V_{p_i} is the set of all possible Chinese characters matching the pinyin p_i.

During inference, a constrained beam search is employed to ensure all candidate characters at step i match p_i. The LLM generates probabilities for valid candidates, and the beam retains only sequences that satisfy the pinyin constraints. This approach leverages the LLM's contextual understanding while strictly adhering to phonetic input, mitigating ambiguity inherent in pinyin.

3.2 AttnInput

While the aforementioned framework demonstrates promising results, its over-simplified typing model poses critical limitations. By reducing $\Pr(P \mid O, W)$ to a binary indicator function, the model ignores real-world complexities such as typing errors, dialect variations, and ambiguous phonetic matches. This simplification fails to capture probabilistic relationships between pinyin sequences and candidate characters, leading to suboptimal robustness and accuracy in practical scenarios.

To address this, we propose AttnInput, a novel approach that replaces the rigid indicator function with a learnable side network to estimate $\Pr(P \mid O, W)$. The key innovation lies in integrating the LLM's contextual predictions with a dynamic pinyin-conditioned likelihood model.

3.2.1 Preliminaries

As illustrated in Figure 2(a), the RWKV6 backbone is composed of a stack of several residual-connected blocks, with each block containing a time-mixing and a channel-mixing sub-blocks. The time-mixing *aka* RWKV attention can be written

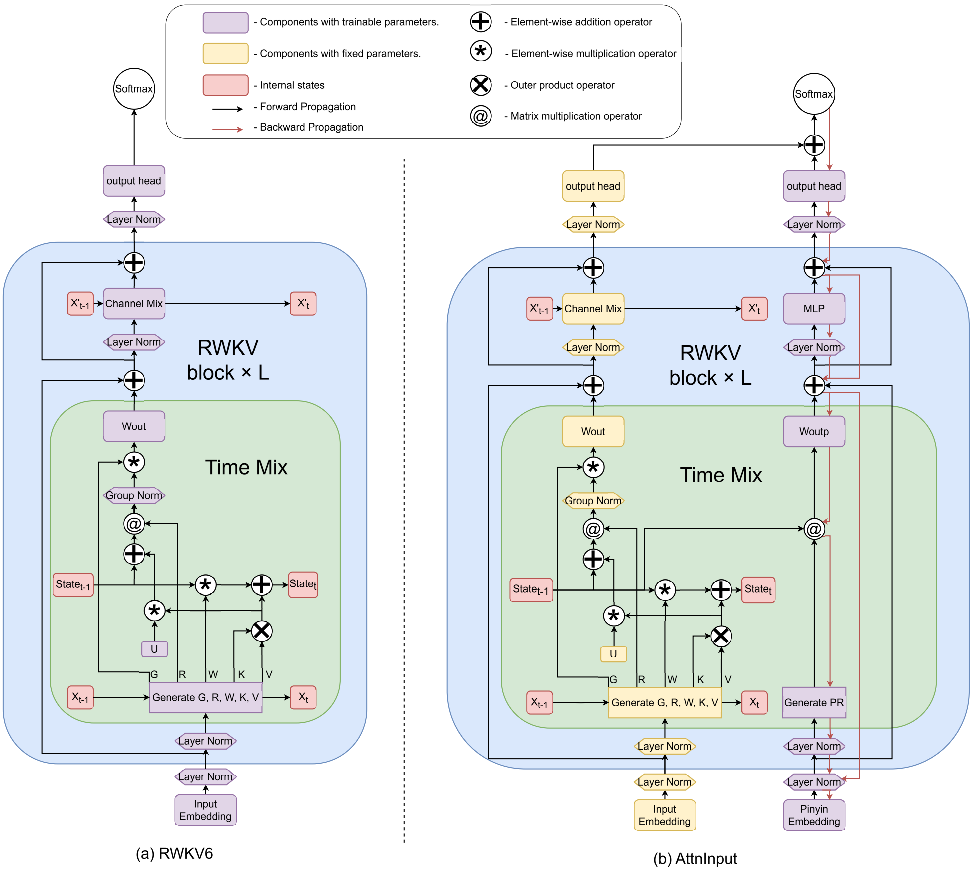

Figure 2: Architecture of the RWKV6 and proposed model, AttnInput.

in a recurrent manner:

$$
\begin{aligned}
\text{ddlerp}_{\square}(a,b) &= a + (b-a) \circ \\
&\quad \text{LoRA}_{\square}(a + (b-a) \circ \mu_{\square}) \\
\square_t &= \text{ddlerp}_{\square}(x_t, x_{t-1}) W_{\square}, \quad \square \in \{r,k,v,g\} \\
w_t &= \exp(-\exp(\text{LoRA}_d(\text{ddlerp}_d(x_t, x_{t-1})))) \\
s_t &= \text{diag}(w_t) \cdot s_{t-1} + k_t^\top v_t \\
o_t &= \text{concat}(\text{SiLU}(g_t) \circ \text{LayerNorm}(r_t(s_t + \\
&\quad \text{diag}(u) k_t^\top v_t))) W_o
\end{aligned}
\tag{4}
$$

where $\mu_{\square}$ are learnable vectors, $\text{LoRA}_{\square}(\cdot)$ applies low-rank adaptation to inject input-dependent adjustments, t is the time step, S represents the compact internal state (analogous to Transformer's KV cache), r retrieves information, k and v store information, w is forget gate, u is content-dependent bias.

3.2.2 Model Architecture

As illustrated in Figure 2(b), we use the RWKV6 model as the backbone model and attach a relatively small side network to the backbone model to extract the pinyin feature and integrate it with information from the context.

We integrate pinyin feature with context information by mapping the former to a fixed-size vector through a linear layer and multiplying it with the internal state of the RWKV6 model. The formula is as follows:

$$
\begin{aligned}
po_{l,t} &= \text{LayerNorm}(px_{l,t}) \cdot W_{pr,l} \cdot S_{l,t+1} \cdot \\
&\quad W_{po,l} + px_{l,t} \\
px_{l+1,r} &= \text{MLP}(\text{LayerNorm}(po_{l,t})) + po_{l,t}
\end{aligned}
\tag{5}
$$

where po is the pinyin-state mixed information, l is the layer index, t is the time step, $W_{pr,l}$ and $W_{po,l}$ are linear layers, and $S_{l,t+1}$ is the internal state of the RWKV6 model at time step $t+1$.

The LLM generates contextual logits $\textbf{logit}_{\text{LLM}}$, while the side network produces pinyin-specific logits. The final prediction logits are computed as:

$$
\textbf{logit}_{\text{final}} = \textbf{logit}_{\text{LLM}} + \textbf{logit}_{\text{pinyin}}. \tag{6}
$$

This additive fusion in log-space is mathematically equivalent to multiplying probabilities in linear space, thereby achieving the combined effect of $\Pr(O|W) \cdot \Pr(P \mid O, W)$.

3.2.3 Ladder Side-Tuning

As illustrated in Figure 2(b), we employ ladder side-tuning (Sung et al., 2022) to attach side networks for mixing pinyin and context information. This approach avoids backpropagating updated parameters through the backbone network.

Due to the significantly fewer parameters in the side network compared to the backbone network, it can save a large amount of computation and memory usage for storing activation values, gradients and optimizer states. See Appendix A for the detailed cost analysis.

3.2.4 Encoding Pinyin Sequence

As illustrated in Figure 1(a), for a certain position i in the pinyin sequence, we select this position and subsequent pinyin $P_i = \{p_i, ..., p_m\}$ as the pinyin information input to the model at this position. Therefore, there is no information interaction between the pinyin information at different positions in the input. The output O_i at each position i is only related to the text context $W_i = \{w_1, ..., w_{n+i-1}\}$ and the pinyin information P_i. This ensures the efficiency of training, as each character's pinyin information is trained, while also maintaining consistency in the data input during both training and inference.

To encode the pinyin sequence, we employ a concatenation operation to combine all pinyin embedding vectors into a unified representation. We pad pinyin sequences with zeros to a fixed length, which is 16 in our experiments. Sequences exceeding this length are truncated. We tokenize pinyin sequence by mapping each letter to its position in the alphabet.

3.2.5 Efficient Training

As illustrated in Figure 1(d), the AttnInput model is trained in a pre-training manner, which is similar to the one used in the large language models. The loss function is given as follows:

$$\mathcal{L}_{AttnInput} = -\frac{1}{n-1}\sum_{i=2}^{n} \log P(w_i | w_{<i}, P'_i), \tag{7}$$

where $w_{<i}$ is the context up to position i, $P'_i = \{p'_i, ..., p'_{m_i}\}$ is the pinyin sequence at position i, p'_i is the corresponding pinyin of w_i, m_i is the randomly selected length of the pinyin sequence at position i, and n is the total number of characters in the training text. The pinyin sequences at each position are independent, with no information interaction between them, to ensure consistency during training and inference. This method enables the model to learn pinyin information from the whole sequence.

However, for previous concat-based models like PinyinGPT-Concat and GeneInput, the design that connects pinyin to the context makes it necessary to train them using the SFT method, as shown in Figure 1(e). The loss function is given as follows:

$$\mathcal{L}_{concat} = -\sum_{i=1}^{m} \log P(w_{n+j} | w_{<n+j}, P), \tag{8}$$

where $w_{<n+j}$ is the context up to position $n+j$, P is the pinyin sequence, and m is the length of the pinyin sequence. The model processes $n+2m$ characters during training, but only m characters' pinyin information are learned, with $n \gg m$, which is inefficient for long contexts.

This suggests that AttnInput exhibits a $\frac{n}{m}$ times improvement in training data utilization compared to prior approaches.

3.2.6 Pinyin-Constrained Training and Inference

The model is trained using the Pinyin-Constrained Training (Tan et al., 2022) method. The probability distribution for the next Chinese characters is calculated solely over Chinese characters that perfectly match the pinyin. The formula is as follows:

$$P(w_i | w_{<i}, P_i) = \frac{\exp(g(w_i | w_{<i}, P_i))}{\sum_{w \in V_{P_i}} \exp(g(w_i | w_{<i}, P_i))} \tag{9}$$

where g is the output of the model, P_i is the pinyin sequence at position i, and $w_{<i}$ is the context up to position i.

Since pinyin can correspond to multiple Chinese characters, for those models mentioned in this paper including AttnInput, PinyinGPT-Concat, vanilla RWKV6, and RWKV6-concat-lora, we use beam search to generate possible character sequences. Each token is generated in a autoregressive manner, and only those Chinese characters that perfectly correspond to the pinyin are considered, in order to improve accuracy.

4 Experiment

4.1 Settings

SkyPile-150B Dataset We use SkyPile-150B (Wei et al., 2023) to generate training and evaluation dataset, which is a large-scale and comprehensive Chinese dataset including 150 billion tokens and 620 gigabytes of text data. SkyPile-150B is not included in the training datasets of the RWKV6 models. The corresponding abbreviated pinyin sequences are automatically generated using the public Python library, pypinyin[2]. The evaluation data is derived from SkyPile-150B, with pinyin lengths ranging from 1 to 16 and context lengths of 64, 512 and 1536. Each evaluation set contains 500 context-pinyin pairs, which are strictly separated from the training data. For each training and evaluation case, the input pinyin are all abbreviated pinyin.

PD Dataset PD dataset (Yang et al., 2012) is a widely used benchmark dataset for the evaluation of pinyin IMEs. We use 2000 segments of consecutive Chinese characters from PD dataset to evaluate the performance on perfect pinyin. For each case, the input pinyin are all perfect pinyin and the context is null.

Training We use RWKV6-1.6B, a pretrained RWKV6 model with 1.6B parameters, as the backbone model, which is fixed during training. AttnInput have a side network with 500M trainable parameters. The loss function is cross-entropy loss. The max learning rate is 3e-4. The learning rate is decayed by cosine annealing with a warmup period of 300 steps. The optimizer is AdamW with a weight decay of 0.01. The batch size is 8. The context length is 1024. The length of pinyin sequence at each position is randomly selected from $[0, 16]$. The model is trained for 40K steps on a single RTX 4090D GPU. To ensure a fair comparison with previous concat-based methods, we also trained a concat-based model with RWKV6-1.6B, labeled as RWKV6-concat-lora. This model was fine-tuned with LoRA (Hu et al., 2021) and includes 500M trainable parameters. All training data is derived from SkyPile-150B and the input pinyin are all abbreviated pinyin.

Evaluation Metric We use the precision at top-K as the evaluation metric, which measures if the ground-truth Chinese character sequence is among the top-K predicted sequences.

4.2 Results on Abbreviated Pinyin

We present the results of the proposed models for abbreviated pinyin on the SkyPile-150B dataset. We compare AttnInput with vanilla RWKV6, PinyinGPT-Concat and RWKV6-concat-lora. GeneInput is not included as its source code or datasets are not publicly released and it do not show better performance than PinyinGPT-Concat on abbreviated pinyin. All outputs are generated by Pinyin-Constrained beam search, with a beam size of 16. When testing PinyinGPT-Concat, we used a context window of size 128, as it was trained on text that does not exceed 128 tokens. The context lengths of 64, 512, and 1536 represent cases of short text, long text, and text exceeding the context window, respectively.

Table 1 demonstrates that the proposed AttnInput model consistently outperforms vanilla RWKV6, PinyinGPT-Concat and RWKV6-concat-lora across most pinyin and context lengths. Several key findings emerge from the results.

- We can see that when the length of the pinyin sequence increases, the performance advantage of AttnInput over vanilla RWKV6 becomes increasingly significant, as the proposed model can leverage more information from the pinyin sequence to generate more accurate Chinese characters.

- All models exhibit decreasing performance, as measured by P@K metrics, with increasing pinyin sequence length. This trend is clearly illustrated in Table 1. For example, focusing on the proposed AttnInput model with a context length of 512, the P@5 score drops consistently from 85.8 for pinyin lengths 1-4, to 76.5 for lengths 5-8, to 66.9 for lengths 9-12, and further to 61.2 for lengths 13-16. Similar downward trends are observed for P@10 and P@15 metrics across all models and context lengths. This performance degradation is attributable to the exponential growth in the number of possible character sequences that match a given abbreviated pinyin sequence as its length increases, which significantly magnifies the ambiguity the model must resolve.

- Leveraging longer contexts significantly benefits both AttnInput and the vanilla RWKV6, likely due to the richer information available in such contexts, including names and locations challenging to infer from pinyin alone. However, PinyinGPT-Concat, trained on contexts shorter than 128 tokens, struggles to exploit this additional information effectively.

[2] https://pypi.org/project/pypinyin

Context Length	Pinyin Length	Evaluation Metric	PinyinGPT-Concat	AttnInput (ours)	Vanilla RWKV6	RWKV6-concat-lora
64	1-4	P@5	71.1	83.7	**84.9**	76.5
		P@10	76.8	88.2	**88.5**	82.7
		P@15	79.6	**90.5**	89.8	85.6
	5-8	P@5	52.5	**71.8**	68.5	52.1
		P@10	57.8	**75.8**	71.8	56.9
		P@15	60.6	**77.5**	73.1	58.9
	9-12	P@5	41.8	**61.5**	55.4	38.0
		P@10	46.2	**65.7**	57.3	41.0
		P@15	48.2	**67.6**	58.0	42.2
	13-16	P@5	32.5	**54.0**	46.3	25.8
		P@10	36.2	**57.9**	47.6	27.8
		P@15	37.9	**59.0**	48.1	28.8
512	1-4	P@5	70.7	85.8	**86.7**	80.3
		P@10	76.4	**89.8**	89.6	85.4
		P@15	79.0	**91.8**	90.9	87.7
	5-8	P@5	48.8	**76.5**	75.6	60.0
		P@10	55.4	**80.8**	78.9	65.0
		P@15	58.2	**82.7**	80.5	67.1
	9-12	P@5	38.4	**66.9**	63.1	42.6
		P@10	42.8	**71.3**	65.6	46.6
		P@15	45.5	**73.0**	66.8	48.5
	13-16	P@5	27.9	**61.2**	55.8	32.7
		P@10	31.6	**64.7**	57.1	35.5
		P@15	33.6	**66.0**	58.0	36.3
1536	1-4	P@5	65.5	85.2	**86.4**	78.0
		P@10	72.9	**89.1**	88.4	83.4
		P@15	76.7	**91.1**	89.7	85.7
	5-8	P@5	43.7	**72.4**	72.3	55.7
		P@10	50.0	**77.4**	75.5	61.7
		P@15	52.9	**79.5**	76.3	64.2
	9-12	P@5	32.8	**62.4**	61.1	42.4
		P@10	37.2	**67.1**	63.8	44.2
		P@15	39.4	**69.5**	65.1	45.4
	13-16	P@5	25.2	**58.4**	52.9	30.8
		P@10	29.1	**62.5**	54.9	33.4
		P@15	30.8	**64.2**	55.4	34.3

Table 1: Evaluation results of the proposed model. To keep the table concise, only the average scores across consecutive sets of four lengths are shown.

- AttnInput exhibits strong length extrapolation capabilities, maintaining superior performance compared to other models even when the context length exceeds the context window.
- The observed inferior performance of RWKV6-concat-lora relative to vanilla RWKV6 provides compelling evidence in support of our proposition that concat-based method disrupts semantic consistency and leads to inefficient training.

We noticed that AttnInput performs slightly worse than vanilla RWKV6 in Top-5 accuracy in some cases. This phenomenon is also observed in previous works (Tan et al., 2022). Our hypothesis is that the training procedure led to a slight degradation in the original model's performance. We analyzed instances where the vanilla RWKV6 model provided the correct answer, while AttnInput failed to prioritize the target. Our investigation revealed that in these specific instances, the abbreviated pinyin corresponded to numerous contextually appropriate Chinese character sequences, causing AttnInput to encounter difficulties in accurately ranking them based on probability. This observation supports our initial hypothesis.

The performance gains observed in other metrics are hypothesized to be a consequence of AttnInput boosting the scores of the initial target tokens based on pinyin information. This mechanism effectively prevents the early elimination of potential target sequences during beam search, especially when the initial tokens are relatively rare.

4.3 Results on Perfect Pinyin

We present the results of the proposed models for perfect pinyin on the PD dataset in Table 2. We compare with Google IME, On-OMWA (Zhang et al., 2017), On-P2C (Zhang et al., 2019), PinyinGPT (Tan et al., 2022), and GeneInput (Ding et al., 2023). Since AttnInput is trained on abbreviated pinyin, we used the corresponding abbreviated pinyin instead of the perfect pinyin as the input of AttnInput during evaluation.

AttnInput outperforms all models except GeneInput, despite not being trained on perfect pinyin. GeneInput's superior performance on the PD dataset is likely due to two interconnected factors. Firstly, the PD dataset's texts (People's Daily, 1992-1998) are relatively old, increasing the likelihood of their inclusion in the extensive pre-training corpora of very large models, which can lead to data contamination. Secondly, GeneInput is substantially larger (15B parameters, 26M SFT instances) com-

Model	P@1	P@5	P@10
Google IME	70.9	78.3	82.3
On-OMWA	64.4	72.9	77.9
On-P2C	71.3	80.5	81.3
PinyinGPT	73.2	84.1	85.5
GeneInput	**88.4**	**96.2**	–
AttnInput (ours)	74.6	86.8	**88.8**
Vanilla RWKV6	75.8	85.9	87.4

Table 2: Comparison with different methods over PD dataset using perfect pinyin.

Model	P@5	P@10	P@15
AttnInput	72.6	76.6	78.3
-pinyin information	71.1	73.7	74.8
-pinyin-constrained training	55.3	57.8	59.0

Table 3: Ablation study for using pinyin information and pinyin-constrained training on SkyPile-150B dataset with context length 512.

pared to AttnInput (2.1B total parameters, 0.32M instances) and was trained with significantly more resources (8×A100-80G for one week vs. a single RTX4090D for 8 hours). This vast difference in scale not only implies greater modeling capacity but also makes GeneInput more susceptible to benefiting from such potential data contamination, thereby likely boosting its performance on this specific, older benchmark.

4.4 Ablation Study

This section describes an ablation study designed to confirm the importance of pinyin information and pinyin-constrained training. Results are shown in Table 3. The *-pinyin information* means that we use the same model configuration, training setup, and dataset as before, but replace the pinyin sequences with blank ones to ensure the model does not learn from pinyin information, aiming to confirm whether the model learns the inherent relationship between pinyin and text, as opposed to simply improving its general Chinese language modeling ability. The *-pinyin-constrained training* means that we remove pinyin-constraint during training but still keep it during inference.

As shown in Table 3, the model performance decreases significantly when pinyin information or pinyin-constrained training is removed, indicating that pinyin-constrained training is essential and AttnInput indeed learns and utilizes the information from the pinyin.

5 Related Works

Pinyin Input Methods Pinyin Input Method Engines (IMEs) have been extensively studied for decades, with a focus on improving accuracy and efficiency. Early methods include N-gram models (Chen and Lee, 2000; Chen, 2003), statistical machine translation (Hatori and Suzuki, 2011; Yang et al., 2012), noisy channel model (Mori et al., 2006), online discriminative training (Jiampojamarn et al., 2008), statistic model (Lin and Zhang, 2008), collocations and k-means clustering (Chen et al., 2012), and Conditional Random Fields (Xia and Cheung, 2016). These approaches heavily rely on a predefined fixed vocabulary and lack the ability to capture long-range dependencies in language. Recent years have witnessed the successful application of neural networks to pinyin IMEs. Neural Network Language Model (Chen et al., 2015), Long Short-Term Memory (LSTM) networks (Zhang et al., 2019; Huang and Zhao, 2018) and attention-based neural networks (Huang et al., 2018) have achieved promising results by modeling sequential data effectively. However, these models face limitations in capturing long-term dependencies and parallelization during training. The emergence of large language models has opened up new possibilities for pinyin IMEs. Recent work has explored the use of LLMs for generating candidate characters based on pinyin input (Tan et al., 2022; Ding et al., 2023). However, directly applying LLMs to pinyin IMEs presents challenges, including semantic discontinuity caused by inserting pinyin sequences and the need for large amounts of training data and computational resources. Our work differs from previous works in that we are the first one to fully leverage the power of large language models and train the models to learn pinyin-context relationships efficiently in a pre-training manner, achieving state-of-the-art performance with minimal training data and computational resources.

6 Conclusion

This paper introduces AttnInput, a novel approach for pinyin IME that effectively integrates pinyin information with a large language model, RWKV, for accurate and efficient Chinese character prediction. By addressing semantic discontinuity and reducing computational overhead, AttnInput achieves state-of-the-art performance on abbreviated pinyin input, paving the way for even more accurate and context-aware pinyin input methods.

7 Limitations

The model does not address common user errors like typos, incorrect tones, or ambiguous phonetic matches, which are critical for practical IMEs.

References

Long Chen, Xianchao Wu, and Jingzhou He. 2012. Using collocations and k-means clustering to improve the n-pos model for Japanese IME. In *Proceedings of the Second Workshop on Advances in Text Input Methods*, pages 45–56, Mumbai, India. The COLING 2012 Organizing Committee.

Shenyuan Chen, Hai Zhao, and Rui Wang. 2015. Neural network language model for Chinese Pinyin input method engine. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 455–461, Shanghai, China.

Stanley F. Chen. 2003. Conditional and joint models for grapheme-to-phoneme conversion. In *8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 2033–2036.

Zheng Chen and Kai-Fu Lee. 2000. A new statistical approach to chinese pinyin input. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, page 241–247, USA. Association for Computational Linguistics.

Keyu Ding, Yongcan Wang, Zihang Xu, Zhenzhen Jia, Shijin Wang, Cong Liu, and Enhong Chen. 2023. Generative input: Towards next-generation input methods paradigm. *Preprint*, arXiv:2311.01166.

Jun Hatori and Hisami Suzuki. 2011. Japanese pronunciation prediction as phrasal statistical machine translation. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 120–128, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *CoRR*, abs/2106.09685.

Yafang Huang, Zuchao Li, Zhuosheng Zhang, and Hai Zhao. 2018. Moon IME: Neural-based Chinese Pinyin aided input method with customizable association. In *Proceedings of ACL 2018, System Demonstrations*, pages 140–145, Melbourne, Australia. Association for Computational Linguistics.

Yafang Huang and Hai Zhao. 2018. Chinese Pinyin aided IME, input what you have not keystroked yet. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2923–2929, Brussels, Belgium. Association for Computational Linguistics.

Sittichai Jiampojamarn, Colin Cherry, and Grzegorz Kondrak. 2008. Joint processing and discriminative training for letter-to-phoneme conversion. In *Proceedings of ACL-08: HLT*, pages 905–913, Columbus, Ohio. Association for Computational Linguistics.

Bo Lin and Jun Zhang. 2008. A novel statistical chinese language model and its application in pinyin-to-character conversion. In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, CIKM '08, page 1433–1434, New York, NY, USA. Association for Computing Machinery.

Shinsuke Mori, Daisuke Takuma, and Gakuto Kurata. 2006. Phoneme-to-text transcription system with an infinite vocabulary. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, ACL-44, page 729–736, USA. Association for Computational Linguistics.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. 2023. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077, Singapore. Association for Computational Linguistics.

Bo Peng, Daniel Goldstein, Quentin Anthony, Alon Albalak, Eric Alcaide, Stella Biderman, Eugene Cheah, Xingjian Du, Teddy Ferdinan, Haowen Hou, Przemysław Kazienko, Kranthi Kiran GV, Jan Kocoń, Bartłomiej Koptyra, Satyapriya Krishna, Ronald McClelland Jr. au2, Niklas Muennighoff, Fares Obeid, Atsushi Saito, Guangyu Song, Haoqin Tu, Stanisław Woźniak, Ruichong Zhang, Bingchen Zhao, Qihang Zhao, Peng Zhou, Jian Zhu, and Rui-Jie Zhu. 2024. Eagle and finch: Rwkv with matrix-valued states and dynamic recurrence. *Preprint*, arXiv:2404.05892.

Yi-Lin Sung, Jaemin Cho, and Mohit Bansal. 2022. Lst: Ladder side-tuning for parameter and memory efficient transfer learning. In *Advances in Neural Information Processing Systems*, volume 35, pages 12991–13005. Curran Associates, Inc.

Minghuan Tan, Yong Dai, Duyu Tang, Zhangyin Feng, Guoping Huang, Jing Jiang, Jiwei Li, and Shuming Shi. 2022. Exploring and adapting Chinese GPT to Pinyin input method. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1899–1909, Dublin, Ireland. Association for Computational Linguistics.

Tianwen Wei, Liang Zhao, Lichang Zhang, Bo Zhu, Lijie Wang, Haihua Yang, Biye Li, Cheng Cheng, Weiwei Lü, Rui Hu, Chenxia Li, Liu Yang, Xilin Luo, Xuejie Wu, Lunan Liu, Wenjun Cheng, Peng Cheng, Jianhao Zhang, Xiaoyu Zhang, Lei Lin, Xiaokun Wang, Yutuan Ma, Chuanhai Dong, Yanqi Sun, Yifu Chen, Yongyi Peng, Xiaojuan Liang, Shuicheng Yan, Han Fang, and Yahui Zhou. 2023. Skywork: A more open bilingual foundation model. *Preprint*, arXiv:2310.19341.

Meng Xuan Xia and Jackie Chi Kit Cheung. 2016. Accurate Pinyin-English codeswitched language identification. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 71–79, Austin, Texas. Association for Computational Linguistics.

Shaohua Yang, Hai Zhao, and Bao-liang Lu. 2012. Towards a semantic annotation of English television news - building and evaluating a constraint grammar FrameNet. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*, pages 333–342, Bali, Indonesia. Faculty of Computer Science, Universitas Indonesia.

Xihu Zhang, Chu Wei, and Hai Zhao. 2017. Tracing a loose wordhood for chinese input method engine. *Preprint*, arXiv:1712.04158.

Zhuosheng Zhang, Yafang Huang, and Hai Zhao. 2019. Open vocabulary learning for neural Chinese Pinyin IME. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1584–1594, Florence, Italy. Association for Computational Linguistics.

A Computing Cost

Throught out this section, we denote by N the total number of parameters in the backbone RWKV6 model, M the total number of parameters in the side network, L the number of layers, h the number of heads and d the dimension of each head. All models are trained with $h = 32$, $L = 24$, $d = 64$, $N = 1.6B$ and $M = 500M$.

The inference FLOPs for each tokenis approximated as follows:

$$\#(\mathrm{InferFLOPs}) = 2(N + M) + 9d^2hL \quad (10)$$

since each matrix requires one multiplication and one addition operation and the RWKV attention requires $9d^2h$ operations(see Equation 4, 5).

The training FLOPs for each token is approximated as inference FLOPs plus four times the total number of trainable parameters plus the FLOPs for backpropagating in RWKV attention:

$$\#(\mathrm{TrainFLOPs})_{\mathrm{L}} = 2N + 6M + 14d^2hL \quad (11)$$

In Full fine-tuning, all parameters are updated, so the training FLOPs for each token is approximated as follows:

$$\#(\mathrm{TrainFLOPs})_{\mathrm{F}} = 6N + 6M + 21d^2hL \quad (12)$$

$$1 - \frac{\#(\mathrm{TrainFLOPs})_{\mathrm{L}}}{\#(\mathrm{TrainFLOPs})_{\mathrm{F}}} = 0.507 \quad (13)$$

That is, ladder side-tuning saves 50.7% FLOPs in training compared to full fine-tuning.

B A Brief Introduction to Hanyu Pinyin and Its Role in Chinese Text Input

Hanyu pinyin, or pinyin, is the standard romanization system for Standard Mandarin Chinese. It employs the Latin alphabet to represent the sounds of Mandarin, aiding in pronunciation and language learning. Importantly, pinyin is not a replacement for Chinese characters, which are the core written units conveying meaning in the language.

The relationship between pinyin and Chinese characters can be summarized as:

- Characters as Semantic Units: Chinese characters are primarily logographic, with each character representing a morpheme or word and carrying meaning.

- Pinyin as Phonetic Representation: Pinyin indicates the pronunciation of characters but does not convey meaning directly.

- Homophony and Context: A single pinyin spelling can correspond to multiple characters with different meanings due to homophones (same pronunciation, different meanings). Context is crucial for disambiguation. For example, the abbreviated pinyin "JDGB" in Figure 1 can match multiple Chinese phrases, such as "鸡蛋灌饼" (egg pancake) and "见到过吧" (have you seen it before).

- Tones: Pinyin uses diacritical marks to denote the four main tones in Mandarin, which are essential for distinguishing meaning.

The advent of computers and mobile devices has made pinyin indispensable for Chinese text input. Pinyin input methods allow users to type pinyin on a standard keyboard and then select the corresponding Chinese characters from a list of suggestions. This technology significantly bridges the gap between the phonetic representation of pinyin and the character-based writing system.

C Latency Analysis

To apply the proposed model to real-world scenarios, we need to analyze its latency. Since the context only expands at the end during the input process, we cache the internal state to avoid repeated prefill operations. Therefore, the latency is equal to the time it takes to generate one token multiplied by the length of the pinyin sequence. We tested the time it takes to generate a token under different beam size settings on a single RTX 4090D GPU, the results are summarized in Table 4.

Beam Size	**Time (ms)**
4	19.06
8	19.00
16	19.53
24	24.06
32	29.08

Table 4: The time it takes to generate one token under different beam size settings.

As we can see, with a beam size of 16, the latency is approximately 20 ms. Assuming the user inputs a pinyin sequence of length 4, the latency would be 80 ms, which is practical for real-world scenarios. The latency can be further optimized by using a smaller model or a faster GPU.

D Case Study

We list three cases in Table 5 to compare outputs produced by PinyinGPT-Concat, vanilla RWKV6, and AttnInput. In case 1 and 2, the vanilla RWKV6 fails to generate the correct answer due to the presence of uncommon characters at the beginning, whereas PinyinGPT-Concat and AttnInput succeed by utilizing pinyin information. In case 1 and 3, PinyinGPT-Concat fails as it lacks the necessary common-sense knowledge. Notably, in all cases, AttnInput consistently produces the correct output.

Case	Predictions: PinyinGPT-Concat	Predictions: vanilla RWKV6	Predictions: AttnInput
Context: 1998年 **Target:** 汉城举办的第二十四届奥运会 **Pinyin:** HCJBDD ESSJAYH **Translation:** The 24th Olympic Games held in Seoul in 1998	汉城举办的第二十三届奥运会 (The 23rd Olympic Games held in Seoul)	后重建并得到二十四届奥运会 (After reconstruction and getting the 24th Olympic Games)	汉城举办的第二十四届奥运会 (The 24th Olympic Games held in Seoul)
Context: 首先，问问目前A股市场的大多数投资者：你选择购买股票还是基金？阅读下面的新闻可能会有帮助。 **Target:** 开源证券最近发布了一份报告 **Pinyin:** KYZQZJFBLYFBG **Translation:** Firstly, ask most investors in the current A-share market: Do you choose to buy stocks or funds? Reading the following news may be helpful. KAIYUAN Securities recently released a report	开源证券最近发布了一份报告 (KAIYUAN Securities recently released a report)	可以在其中就发布了一份报告 (A report can be published within it)	开源证券最近发布了一份报告 (KAIYUAN Securities recently released a report)
Context: 磁性测厚法:适用导磁 **Target:** 材料上的非导磁层厚度 **Pinyin:** CLSDFDCCHD **Translation:** Magnetic thickness measurement method: applicable to the thickness of non-magnetic layers on magnetic materials	材料深度放大尺寸厚度 (Material depth, enlarged size, thickness)	材料上的非导磁层厚度 (Thickness of non-magnetic layer on material)	材料上的非导磁层厚度 (Thickness of non-magnetic layer on material)

Table 5: Case study on abbreviated pinyin.