A Neural Approach to Fine-Tuning for Pinyin-to-Chinese Conversion

Roy Shi
Simon Fraser University
rsa180@sfu.ca

Hanbo Ji
Simon Fraser University
hja60@sfu.ca

Sheng Liang
Simon Fraser University
sla466@sfu.ca

Abstract

This paper investigates the problem of converting Chinese Pinyin sequences into Chinese characters through fine-tuning a neural network Transformer model mT5. The project simulates practical scenarios encountered in Chinese Pinyin input method editors (IMEs), where due to tone ambiguity and the presence of many homophones in Chinese, effective Pinyin input requires context-aware disambiguation. Traditional approaches such as n-gram models and Hidden Markov Models (HMMs) have limitations in modeling long-range dependencies. Therefore, we propose a fine-tuning approach using pretrained variances of T5 Transformer models, mT5 and Mengzi-T5, to perform Pinyin-to-Chinese character translation. We conduct preprocessing to normalize tones, simulate user typos, and align Pinyin-character pairs. Pinyin typo is also simulated in our dataset and is used to fine-tune our model. Our results demonstrate that model size significantly affects performance, with mT5-large achieving the highest character-level accuracy. Model trained with typo-simulated dataset also obtains higher accuracy against model trained with clean dataset when evaluated on test set with Pinyin typo.

1 Introduction

Chinese is a logographic language with over 10,000 characters (Zhang et al., 2018). Typing this large character set efficiently on a standard Latin-style keyboard with only 26 keys is challenging. Pinyin, the standardized romanization system for Chinese characters, enables users to input Chinese using Latin alphabet letters based on the pronunciation of words. It is widely used in digital environments, particularly in Chinese Input Method Editors (IMEs), where users type the phonetic spelling of words, and the system converts them into the appropriate Chinese characters.

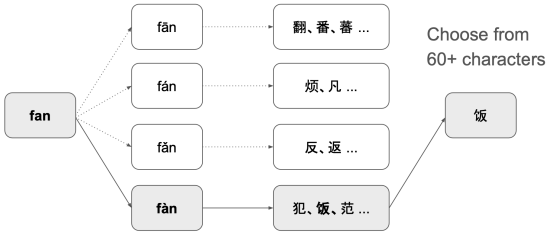

Figure 1: Pinyin IME processes user's Pinyin input to obtain Chinese character "饭"

However, this process is not straightforward due to the inherent tone ambiguity and the prevalence of homophones in Mandarin Chinese. Each syllable in Pinyin can have up to five tonal variations (four tones plus a neutral tone), and many distinct Chinese characters share the same Pinyin spelling —especially when tones are omitted for typing convenience. As a result, a single Pinyin input string may map to dozens of possible characters. An example demonstrates this scenario in Figure 1, where it is difficult for Pinyin IME to determine which Chinese character the user intends to input. This ambiguity presents a significant challenge in accurately converting Pinyin into Chinese text, requiring language models to disambiguate among candidates based on linguistic patterns and sentence context.

Luckily, context dependency plays a crucial role in disambiguating Pinyin-to-Chinese conversion. While a single Pinyin input may correspond to multiple valid Chinese characters, the presence of preceding words provides valuable linguistic cues that can help determine the most likely output. For instance, demonstrated in Figure 2, the Pinyin syllable "fan" could map to characters such as "翻"(flip), "饭"

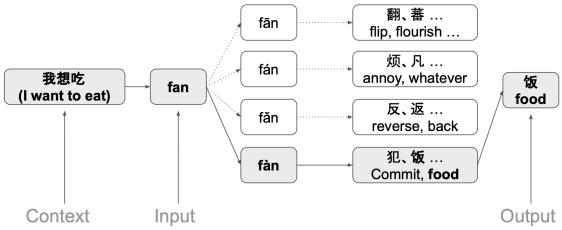

Figure 2: Pinyin IME processes user's Pinyin input but with contextual phrase

(food), or "犯" (commit), but if we know the context phrase is given as follows: "我想吃" (I want to eat), then the system should confidently select "饭" (food) based on semantic fit. This enables models to leverage sentence-level or even paragraph-level information to make more accurate predictions, thereby reducing the impact of tone ambiguity and homophonic nature of the Chinese language.

Traditionally, statistical models such as n-gram and Hidden Markov Models (HMMs) have been applied to Pinyin-to-Chinese conversion (Liu and Guthrie, 2009). While effective for capturing local patterns, they struggle with long-range dependencies and lack a deep understanding of semantic context. Therefore, neural sequence models may be more well-suited to address these limitations, particularly Transformer-based architectures, as this is a standard translation task.

In this work, we select mT5 (Xue et al., 2021) as our training target due to its strong multilingual performance, efficient training scalability, and compatibility with parameter-efficient tuning methods over the others like mBART (Liu et al., 2020). To further investigate the impact of language-specific pretraining, we also evaluate Mengzi-T5 (Zhang et al., 2021)—a Chinese-pretrained variant of T5—to determine whether a model trained specifically on Chinese corpora can outperform a general-purpose multilingual model. Additionally, we address progresses made to simulate actual user input typos while performing Pinyin to Chinese translation through Pinyin IME in our model fine-tuning, allowing better accuracy in practical user setting.

2 Related Work

Several approaches have been proposed for Pinyin-to-Chinese conversion.

2.1 Statistical Models

Early Chinese IMEs relied heavily on statistical models, such as n-gram and HMMs, to generate character sequences from Pinyin input (Chen and Lee, 2000; Zhou et al., 2007). These models use probabilities based on character co-occurrence to disambiguate input. While computationally efficient, they lack the capacity to understand long-term contextual dependencies.

2.2 Neural Network Models

Neural architectures like Seq2Seq with attention mechanisms (Bahdanau et al., 2016) and Transformer models (Vaswani et al., 2023) have achieved state-of-the-art results in sequence modeling and language generation tasks, making them natural candidates for IME modeling. These models capture global context and support flexible token-level generation. As an example, there has been proposal of a neural Pinyin IME with BiLSTM-based decoding and an online vocabulary adaptation mechanism to handle open vocabulary settings and real-time input corrections (Huang et al., 2018). While their approach focuses on user adaptation and vocabulary pruning, we emphasize context modeling through Transformer attention and addresses further challenges in Pinyin typo simulation.

2.3 Pretrained T5 Models

We adopt mT5 (Xue et al., 2021), a multilingual variant of the T5 model (Raffel et al., 2023), which follows a unified text-to-text framework and has been pretrained on the mC4 corpus covering over 100 languages, including Chinese. Its encoder-decoder architecture makes it particularly suitable for sequence generation tasks like Pinyin-to-Chinese conversion, where character sequences must be generated from phonetic input. Unlike encoder-only models such as BERT (Devlin et al., 2018), which are primarily designed for classification and token-level tasks, mT5 supports conditional text generation with full attention over both input and output sequences. This

makes it more aligned with the requirements of transliteration and translation-style problems.

In addition, we incorporate Mengzi-T5 (Zhang et al., 2021), a Chinese-pretrained variant of T5 that is lightweight and optimized for efficiency. Mengzi-T5 is trained exclusively on Chinese corpora, including web and news text. By comparing mT5 and Mengzi-T5, we aim to assess the trade-offs between multilingual generalization and domain-specific specialization.

3 Approach

3.1 Problem Formulation

We formulate the Pinyin-to-Chinese conversion task as a sequence-to-sequence (Seq2Seq) problem. Given an input sequence of Pinyin tokens, denoted as $X = (x_1, x_2, \ldots, x_n)$, the goal is to generate the corresponding Chinese character sequence $Y = (y_1, y_2, \ldots, y_m)$. We aim to learn an injective mapping function $f_\theta : X \to Y$, parameterized by θ, that maximizes the conditional probability $P(Y \mid X; \theta)$. This is achieved by modeling the output in an autoregressive manner, such that

$$P(Y \mid X) = \prod_{t=1}^{m} P(y_t \mid y_{<t}, X; \theta),$$

where the decoder generates each Chinese character based on both the encoded input and previously generated characters.

3.2 Model Architecture

Our approach is built on top of Transformer-based encoder-decoder models, which are well-suited for conditional sequence generation tasks like Pinyin-to-Chinese translation. We experiment with the mT5 family, including `mt5-small`, `mt5-base`, and `mt5-large` (Xue et al., 2021), as well as `Mengzi-T5-base` (Zhang et al., 2021), a lightweight variant pretrained on Chinese corpora. These models consist of a Transformer encoder that processes the Pinyin input and a Transformer decoder that generates the Chinese output. The architecture is fully attention-based, utilizing multi-head self-attention, cross-attention, and feed-forward layers, following the design of the original Transformer (Vaswani et al., 2023).

3.3 Prefix and Tokenization

To better guide the model, we format each training input as a natural language instruction, using the prefix: `Convert Pinyin to Chinese: <pinyin sequence>`. For example, the input "Convert Pinyin to Chinese: wo ai ni" corresponds to the target sequence "我爱你". We tokenize inputs and outputs using the `MT5Tokenizer`, which is based on SentencePiece. During tokenization, all input sequences are truncated and padded to the longest sequence, since our training data and outputs may have very distinct lengths. For loss calculation, padding tokens in the target sequence are replaced with -100 to ensure they are ignored by the loss function.

3.4 Hyperparameters

We fine-tune our models using Hugging Face's `Seq2SeqTrainer` with the AdamW optimizer and a learning rate of 3×10^{-6}. The models are trained for 20 epochs with a batch size of 16 and evaluated every 200 steps. Mixed-precision training is enabled using `bf16` to improve memory efficiency. The training objective is the standard cross-entropy loss (Mao et al., 2023), defined as

$$\mathcal{L}_{CE} = -\sum_{t=1}^{m} \log P(y_t \mid y_{<t}, X; \theta),$$

where y_t is the ground-truth Chinese character at time step t, and X is the Pinyin input. Evaluation metrics include character-level accuracy, BLEU, ChrF, and ROUGE.

3.5 Inference

During inference, we use beam search decoding to generate predictions. We set the beam size to 4 and apply early stopping once all beams have completed. The model uses the `generate()` function from the Hugging Face Transformers library, with a maximum output length of 64 tokens. Outputs are then decoded and post-processed to remove special tokens and normalize spacing.

4 Experiments

4.1 Data

We use the swaption2009/20k-en-zh-translation-pinyin-hsk dataset from the

Hugging Face Hub, which consists of approximately 20,000 aligned English, Chinese, and Pinyin sentence pairs. Each sample includes a Chinese sentence, its corresponding Pinyin transliteration (with tone), and an English translation. For our task, we use the Pinyin-Chinese pairs to fine-tune models for Pinyin-to-Chinese generation. This dataset is particularly well-suited for our task as it provides readily available Pinyin transliteration instead of having to re-generate Pinyin like many other datasets.

However, we still perform preprocessing to remove unwanted punctuation, normalize spacing, and standardize lowercases for Pinyin. Additional cleaning steps include converting full-width to half-width characters and handling special cases such as correcting errors in retroflex suffixes from e.g. "r" → "er" and romanize non-roman pinyin letter like "ü" → "v" . We also validated the alignment between the number of Pinyin syllables and Chinese characters to ensure training consistency. Overall, this dataset allows us to train and evaluate our models on a wide range of syntactically and semantically diverse sentence pairs.

To further improve robustness and simulate real-world Pinyin-to-Chinese input scenarios, we introduce another dataset that augments the training data in addition by incorporating synthetic typos into the Pinyin sequences. We inject one typo per three Pinyin syllables to apply one of three error types: substituting a character with a neighboring keyboard key, randomly deleting a character, or swapping adjacent characters. This augmentation mimics common user typing mistakes in Pinyin IME, such as hitting nearby keys or skipping a character. By exposing the model to noisy Pinyin inputs during training, we encourage better generalization and improved resilience to imperfect input at inference time.

4.2 Implementation

In our implementation, we utilize the Hugging Face `transformers` and `datasets` libraries, specifically employing the `Seq2SeqTrainer` API for training and evaluation. We incorporated pretrained models from the Hugging Face Hub, including `google/mt5-small`, `mt5-base`, `mt5-large`, and `Langboat/mengzi-t5-base`. Our models are ran on L4 GPU in Google Colab, with the exception of `google/mt5-large` model, which is fine-tuned on A100 GPU for higher RAM provision. For Pinyin conversion, we employ the `pypinyin` library.

While our team develop the fine-tuning scripts and preprocessing pipelines from scratch, we draw inspiration from Ethen Liu' s tutorial on German-to-English machine translation using mT5 (Liu, 2023). The tutorial provides useful implementation details and best practices for setting up the data collator for batch processing to speed up training process and evaluation metrics computations, which we adapt to our Pinyin-to-Chinese conversion task.

4.3 Evaluation

We evaluate model performance using a range of standard sequence generation metrics, each capturing different aspects of output quality. The primary metric we consider is a custom character-level accuracy, which measures the proportion of characters in the predicted output that exactly match the reference output at the same position. Given a predicted sequence $\hat{Y} = (\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_m)$ and a reference sequence $Y = (y_1, y_2, \ldots, y_m)$, character-level accuracy is computed as:

$$CharAcc = \frac{1}{m} \sum_{t=1}^{m} 1(\hat{y}_t = y_t),$$

where m is the length of both predicted and reference Chinese sentences with the assumption that both sentence lengths equate. This metric provides strict correctness, especially suitable for Chinese, where we want to assure each complete one-to-one mapping of each character. However, it is highly sensitive to small prediction errors such as insertions, deletions, or character swaps, which may result in huge score offset even when outputs are partially correct but ordering is wrong.

To complement this, we bring in ChrF (Popović, 2015), a character n-gram F-score that balances precision and recall over character-level n-grams. Unlike BLEU (Papineni et al., 2002), which is word-based, ChrF is better suited for logographic languages like Chinese, where character-level overlap is more meaningful. ChrF is computed as:

$$ChrF_\beta = \frac{(1+\beta^2) \cdot P_n \cdot R_n}{\beta^2 \cdot P_n + R_n},$$

where P_n and R_n are the n-gram precision and recall, respectively, and β is typically set to 2 to give higher weight to recall. ChrF is more forgiving than plain character accuracy, as it allows partial credit for overlapping character sequences even when alignment is imperfect.

We also reference BLEU and Rouge (Lin, 2004) as complementary evaluation metrics. These metrics are commonly used in text generation tasks and provide a balance between precision-oriented and recall-oriented evaluation. This allows us to assess the model's ability not only to produce correct characters but also to preserve the overall structure and meaning of the target sentence.

4.4 Methods Compared

We conduct two main sets of comparisons to evaluate the effectiveness of model architecture, scale, and data robustness strategies.

First, we compare the pretrained mT5-large model as a baseline (zero-shot) against its fine-tuned variants: mT5-small, mT5-base, mT5-large, and Mengzi-T5-base. All models are fine-tuned on the same standard Pinyin-to-Chinese dataset using identical preprocessing steps, training scripts, and hyperparameters. This comparison allows us to investigate how model size and the scope of pretraining (multilingual vs. Chinese-specific) influence translation quality and accuracy.

Second, we evaluate the impact of typo robustness by training an additional variant of mT5-large on a dataset augmented with synthetic Pinyin typos. We compare this model directly against the standard mT5-large fine-tuned on clean data. This setup assess how exposure to noisy input during training affects the model' s ability to generalize to real-world IME input, where user typos are common.

5 Results & Analysis

5.1 Clean Data Fine-tune

As shown in Table 1, performance improves notably with both fine-tuning and increased model size. mT5-small underperforms, likely due to limited capacity, while mT5-base and Mengzi-T5-base achieve similar results, suggesting that multilingual and Chinese-specific pretraining are equally effective at the base scale.

Model	Char Acc	chrF
mT5-large (pre-tuned)	0	0.19
mT5-small	4.61	2.75
mT5-base	41.11	24.34
mengzi-T5-base	40.79	-
mT5-large	**70.15**	**53.59**

Table 1: Custom character accuracy and ChrF performance comparison across models on Pinyin-to-Chinese conversion using clean dataset.

The baseline pre-tuned mT5-large model performs poorly. Once fine-tuned on clean data, mT5-large achieves the highest scores, with 70.15% character accuracy and a ChrF of 53.59%.

Some optimizations in training process can likely be applied to boost mT5-small's low performance and as well as the others. Attempts can have been made to test out higher learning rate, batch sizes, or number of training epochs, as smaller models usually benefit from careful hyperparameter tuning. Additionally, parameter-efficient fine-tuning (PEFT) techniques (Mangrulkar et al., 2022) such as prefix tuning or LoRA (Hu et al., 2021) can be employed to enhance learning without requiring full fine-tuning of the entire model.

When evaluating at BLEU and Rouge-L, which assesses on n-gram precision and common subsequence overlaps, we see that in Table 2, the strong BLEU and Rouge-L scores from mT5-large indicate it produces both fluent and semantically accurate outputs. In contrast, low BLEU from smaller or unfine-tuned models suggests poor token-level accuracy, even if some content is partially captured.

The evaluation loss curves in Figure 3 reveal clear differences in learning dynamics across models fine-tuned under the same setup: 20 epochs, AdamW optimizer, a learning rate of 3e-6, batch size 16, and `bf16` mixed precision. mT5-Large achieves the lowest and most stable loss. The similar convergence of mT5-Base and Mengzi-T5-Base suggests that at the base scale, the advantage of domain-specific pre-

Model	BLEU	Rouge-L
mT5-large (pre-tuned)	0.0	0.18
mT5-small	0.80	0.37
mT5-base	28.57	1.58
mengzi-T5-base	-	-
mT5-large	**59.78**	**1.62**

Table 2: BLEU and Rouge-L performance comparison across models on Pinyin-to-Chinese conversion using clean dataset.

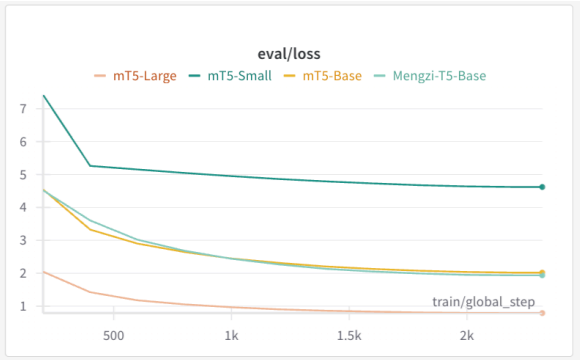

Figure 3: Training loss curves of our T5 variant models

training diminishes when fine-tuning is sufficiently strong. In contrast, mT5-Small starts with a much higher loss and quickly plateaus, indicating limited capacity and underfitting.

5.2 Typo-incorporated Fine-tune

Model	Char Acc	chrF
mT5-large (clean data)	38.73	28.62
mT5-large (typo data)	**44.20**	**35.88**

Table 3: Performance comparison across models trained on clean/typo-simulated data when evaluating on typo-simulated dataset.

We also examine the impact of typo robustness by training mT5-large on a dataset augmented with synthetic Pinyin input errors. In Table 3, mT5-large trained with typo data achieves higher accuracy than mT5-large trained with clean data, indicating improved generalization to noisy input conditions. These results confirm that large Transformer models like mT5-large are highly effective for disambiguating Pinyin input and that data augmentation with realistic noise improves robustness with minimal performance trade-off.

Although the accuracy improvements are not as significant, with minimum performance boost, it does indicate that some typo are well handled by our improved model. This could likely have been caused by us introducing typo in our dataset too frequently, leading to model unable to learn the pattern from overfitting. Attempts can be made in the future to include less typo in our dataset for potential higher performance.

5.3 Qualitative Analysis

Model	Chinese Prediction	English Translation
mT5-small	他把他带到他的房间里	He took him to his room
mT5-base	他喜欢拿他的奖学金开玩笑	He likes to joke about his scholarship
mengzi-t5-base	他喜欢那他的同学开课	He likes that his classmates start class
mT5-large	他喜欢拿他的同学开玩笑	He likes to joke about his classmates
Ground Truth	他喜欢拿他的同学开玩笑	He likes to joke about his classmates

Table 4: Qualitative example of fine-tuned model predictions with English translation reference.

Table 4 presents a representative example of model predictions for the same Pinyin input after fine-tuning on our aligned dataset. The mT5-small model generates a grammatically coherent sentence but fails to match the semantics of the reference, demonstrating limited capacity to resolve contextual relationships. Its outputs generated are mostly unrelated to the input's original meaning but remain similar in sentence structure. The mT5-large model performs closest to the ground truth, producing correct prediction to the reference in this example.

mT5-base and Mengzi-T5-base perform noticeably better, producing semantically closer predictions. However, both mT5-base and Mengzi-T5-base introduce incorrect phrases, possibly due to overfitting on domain-specific collocations. This is especially observable for Mengzi-T5-base, which misinterpreted the word "开玩笑" (make joke) after seeing "开" (make/start) and predicts "课" (class).

When evaluated on a typo-simulated input, as shown in Table 5, the mT5-large model fine-tuned on typo-augmented data significantly outperforms its counterpart trained on clean data. The Pinyin input for this example is *"ta ix huan na ta de tong xu kai an xiao"*, which contains the following simulated typos:

- **Character misordering**: `"xi"` → `"ix"`
- **Missing character (1)**: `"xue"` → `"xu"` (missing `"e"`)

- **Missing character (2)**: `"wan"` → `"an"` (missing `"w"`)

Model	Chinese Prediction	English Translation
mT5-large (clean data)	他拒绝拿他的同意并案小	He refused to take his consent to open a case
mT5-large (typo data)	他喜欢那他的同窗开玩笑	He likes to joke with his classmates
Ground Truth	他喜欢拿他的同学开玩笑	He likes to joke about his classmates

Table 5: Qualitative example of fine-tuned model predicting on typo datasets.

The mT5-large model trained on clean data fails to recover from these inconsistencies, instead producing a semantically irrelevant sentence that diverges early from the intended meaning. This suggests that the model is sensitive to malformed input and lacks robustness when encountering even minor deviations from the expected structure.

In contrast, the mT5-large model fine-tuned on typo-simulated data demonstrates improved resilience. Despite the noisy input, it successfully recovers two of the corrupted segments and generates a coherent sentence that closely aligns with the reference. This example highlights the value of incorporating realistic user input errors during training, as it enables the model to generalize better to the kinds of typos common in real-world IME scenarios.

6 Future Work

While our current model demonstrates strong performance on standard Pinyin-to-Chinese conversion, there are several promising directions for future improvements.

6.1 Data Augmentation

One major avenue is expanding the training data through data augmentation techniques. Although our dataset is clean and well-aligned, its size is relatively limited, which can lead to overfitting—especially for larger models. Introducing controlled noise, paraphrased sentence variants, or synthetic data generated from linguistic rules could improve model generalization and robustness.

6.2 Pinyin Initials Translation

Another valuable extension would be to support abbreviated Pinyin inputs, where users type only the initials of each syllable to speed up input. This is common in real-world IME usage (e.g., typing "w m" to represent "wo men" for "我们"). Handling such inputs would require rethinking the preprocessing pipeline and potentially training the model with both full and abbreviated forms. This task is inherently more ambiguous and context-dependent, and may benefit from joint modeling strategies or multi-task learning setups.

7 Conclusion

This paper explored the problem of Pinyin-to-Chinese character conversion using modern Transformer-based encoder-decoder models. Framing the task as a sequence-to-sequence generation problem, we fine-tuned several pretrained models, including mT5-small, mT5-base, mT5-large, and Mengzi-T5-base, on a curated dataset containing aligned Pinyin and Chinese sentence pairs. Our results demonstrates positive correlation that model capacity plays a dominant role in performance.

Interestingly, despite Mengzi-T5 being pretrained specifically on Chinese corpora, its performance closely matched mT5-base, a multilingual model. Model scale, rather than pretraining language scope alone, can be a more critical factor in improving performance on this task.

We also introduce typo augmentation to simulate real-world IME input scenarios. Our findings reveal that noisy input are significantly more robust to imperfect or malformed Pinyin strings, correctly recovering from character misorderings and deletions. This reinforces the value of training with realistic input variation, especially for deployment in practical applications where user input is often informal or error-prone.

Reflecting on our results, one challenge we encountered was the sharp performance drop in smaller models like mT5-small. Despite identical training procedures, it was unable to learn meaningful character mappings. We could have incorporated techniques to improve efficiencies across smaller models like PEFT and LoRA to perform fine-tuning with lower computational costs. More experiments could be conducted on refining hyperparameter to also improve result, such as increasing learning rate or learning rate scheduler.

In addition, we can further integrate more features as future work to better simulate real-

istic Chinese inputting with Pinyin IME, such as incorporating support on predicting Pinyin initials only.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate. *Preprint*, arXiv:1409.0473.

Zheng Chen and Kai-Fu Lee. 2000. A new statistical approach to Chinese Pinyin input. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 241–247, Hong Kong. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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, Zhuosheng Zhang, and Hai Zhao. 2018. Neural-based pinyin-to-character conversion with adaptive vocabulary. *CoRR*, abs/1811.04352.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Ethen Liu. 2023. Machine translation with mt5. Accessed: 2024-04-09.

Wei Liu and Louise Guthrie. 2009. Chinese pinyin-text conversion on segmented text. In *Text, Speech and Dialogue*, pages 116–123, Berlin, Heidelberg. Springer Berlin Heidelberg.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *CoRR*, abs/2001.08210.

Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. 2022. Peft: State-of-the-art parameter-efficient fine-tuning methods. `https://github.com/huggingface/peft`.

Anqi Mao, Mehryar Mohri, and Yutao Zhong. 2023. Cross-entropy loss functions: Theoretical analysis and applications. *Preprint*, arXiv:2304.07288.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA. Association for Computational Linguistics.

Maja Popović. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention is all you need. *Preprint*, arXiv:1706.03762.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online. Association for Computational Linguistics.

Zhuosheng Zhang, Yafang Huang, and Hai Zhao. 2018. Subword-augmented embedding for cloze reading comprehension. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1802–1814, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Zhuosheng Zhang, Hanqing Zhang, Keming Chen, Yuhang Guo, Jingyun Hua, Yulong Wang, and Ming Zhou. 2021. Mengzi: Towards lightweight yet ingenious pre-trained models for chinese. *CoRR*, abs/2110.06696.

Xiaohua Zhou, Xiaohua Hu, Xiaodan Zhang, and Xiajiong Shen. 2007. A segment-based hidden markov model for real-setting pinyin-to-chinese conversion. In *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, CIKM '07, page 1027–1030, New York, NY, USA. Association for Computing Machinery.