Improving Accuracy of Low-resource ASR using Rule-Based Character Constituency Loss (RBCCL)

Rupak Raj Ghimire, Prakash Poudyal, Bal Krishna Bal
rughimire@gmail.com, { prakash, bal }@ku.edu.np
Information and Language Processing Research Lab (ILPRL)
Kathmandu University, Nepal

Abstract

Modern general-purpose speech recognition systems are more robust in languages with high resources. However, achieving state-of-the-art accuracy for low-resource languages is still challenging. To deal with this challenge, one of the popular practice is fine-tuning the pre-trained model on low-resource settings. Nevertheless, a pre-trained or fine-tuned model fails to capture the complex character and word constituency in the Devanagari script transcription. We proposed a complementary loss function designed to force the model to learn the character constituency of Devanagari script.

Our complementary loss function, called Rule-Based Character Constituency Loss (RBCCL), penalizes incorrect transcriptions and updates the overall loss during the model training phase. This loss function can also be combined with Connectionist Temporal Classification (CTC) loss or cross-entropy loss which are widely used in ASR training. Our experiment shows that combining the existing cross-entropy loss with a new complementary loss (RBCCL) improves the Word Error Rate (WER), reducing it from 47. 1% to 23. 41% which is a very promising result.

1 Introduction

Automatic Speech Recognition (ASR) is a subset of speech technology that uses machine learning and neural networks to transcribe audio data into its corresponding written text. Machine learning-based ASR systems can be trained using different methods such as supervised, semi-supervised, or unsupervised techniques. In supervised approach the spoken audio and its text transcription must match exactly for the system to learn efficiently. This requires a large amount of carefully selected data, with the precise alignment done manually. Ensuring that each spoken words matches accurately with the written text. This necessitates a considerable expenditure of time and effort in human alignment.

The initial idea for implementing unsupervised ASR was introduced by Liu et al. (2018). Since then, unsupervised methods have become quite popular. A recent study by Baevski et al. (2022) showed that unsupervised models now perform competitive to supervised models. This progress is mainly due to advances in deep learning and better access to computing resources, which have made large pre-trained speech models more widely available. An example of this progress is the recently released $Wav2Vec2-BERT2.0$ (Chung et al., 2023) is trained on $4.5M$ hours of audio data covering more than 143 languages. In line with this, the *whisper-large* models (Radford et al., 2022) are trained on $680,000$ hours of labeled audio data and comprise $1550M$ parameters. These models capture complicated audio and linguistic patterns properly, allowing them to generalize across languages, accents, and sounds.

Training these models we need a extensive amount of memory, storage, and computing resources. Because of these high resource demands, full parameter fine-tuning can be time-consuming and resource-intensive. Parameter-Efficient Fine-Tuning (PEFT) (Mangrulkar et al., 2022) is ideal for resource-constrained contexts and still yields comparable performance. PEFT-based approach such as Low-Rank Adaptation (LoRA) (Hu et al., 2022) technique significantly reduces the number of trainable parameters, making it computationally efficient and reducing the risk of overfitting, particularly in low-resource settings. For example, in the $GPT-3175B$ model, LoRA reduced the trainable parameters by

Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL 2025), pages 61–70
January 19, 2025.

10,000 times and reduce the GPU requirements by 3 times (Hu et al., 2022).

Connectionist Temporal Classification (CTC) and cross-entropy loss are popular choices for training ASR models. CTC aligns input and target sequences without predefined alignment, but struggles with the complexities of the Devanagari script, where ligatures and half-letters require specific handling. Cross-entropy loss, used for frame-wise classification, ensures alignment but may miss linguistic nuances, especially in scripts like Devanagari where characters merge visually. Also, in low-resource settings, fine-tuning can often lead to overfitting, which may cause the model to overlook language-specific patterns.

To address this issue, we propose a technique that incorporates linguistic rules defined (details in Section 3) into the training process. This is achieved by implementing a unique loss function that utilizes the linguistic rules of a particular language. Specifically, our proposed loss function penalizes loss (cross-entropy loss in our case) based on the word construction rules of the Devanagari script (Nepali). This enforces the model to learn the linguistic rules, mitigates overfitting, and thereby improves the prediction accuracy. We conducted experiments using both full-parameter and PEFT-based fine-tuning approaches.

The organization of the remainder of the paper is as follows: In Section 2, the related works are explained, followed by the methodology in Section 3. Section 4 presents the experiments conducted. The discussion and interpretation of the results are presented in Section 5. Finally, the paper concludes with Section 6, where a summary of the findings, future plans, and potential extensions to the work is explained.

2 Related Works

Pre-trained large speech models have revolutionized speech-related downstream tasks, such as ASR. We can use various types of pre-trained models for the fine-tuning task. We can effectively fine-tune multilingual, supervised, semi-supervised, and unsupervised models. Wav2Vec2-Conformer (Wang et al., 2020), Whisper (Radford et al., 2022), MMS-1B (Pratap et al., 2023), HuBERT (Hsu et al., 2021), Wav2Vec2-BERT 2.0 (Chung et al., 2023), Wav2Vec2-Phoneme (Xu et al., 2022), Wav2Vec2.0 (Baevski et al., 2020a; Baevski et al., 2020b), Wav2Vec (Schneider et al., 2019) are some examples of pre-trained speech models trained on massive amounts of multilingual speech datasets. *Whisper*, for example, is a powerful encoder-decoder model that can be used for multilingual ASR. Wav2Vec and its successors (Chung et al., 2023; Xu et al., 2022; Wang et al., 2020) use contrastive learning to learn robust speech representations. These pre-trained models have significantly improved the accuracy and robustness of ASR systems, making them more accessible and useful in a variety of applications. However, this is only true for resourceful languages.

Various published research (Arunkumar et al., 2022; Khare et al., 2021; Luo et al., 2021; Singh et al., 2023; Zheng et al., 2023; Ghimire et al., 2023a) show that the accuracy of the ASR in low-resourced languages, including Nepali, can be improved by fine-tuning pre-trained models. This has been proven in work proposed by Ghimire et al. (2023a) by decreasing the Character Error Rate significantly. As per these researches, the fine-tuning approach requires less computing and also reduces the model training time significantly compared to full model training. However, due to the higher number of parameters involved in the network, the full parameter fine-tuning is still challenging.

The use of the Parameter-Efficient Fine-Tuning (PEFT) approach, such as Low-Rank Adaptation (LoRA) and its variants, is becoming very common in fine-tuning of speech models. The effectiveness of the LoRA on *Whisper* model is reported by various scholars (Liu et al., 2024; Song et al., 2024).

The Nepali ASR is still in its early stages of research and development. However, there are some promising results as reported in various works (Ghimire et al., 2023a; Shrestha et al., 2021; Regmi and Bal, 2021; Ghimire et al., 2023b). Among them, the work reported by Ghimire et al. (2023a) is the only work related to fine-tuning for building the Nepali ASR system. The author proposes semi-supervised fine-tuning of a pre-trained model using an active learning approach. This research uses

the SLR54 (Kjartansson et al., 2018) dataset. They obtained Character Error Rate (CER) of 6.77% by fine-tuning the Massively Multilingual Speech (MMS-1B) model (Pratap et al., 2023).

Dutta et al. (2018) has implemented three complementary loss functions for the optical character recognition task of Indic script while training the model, but this is not explored in the training of ASR model. The language-specific rule-based ASR error correction mechanism is presented by Yang et al. (2022). This work reported the use of rules in the decoding phase. However, the use of the language-specific loss function in ASR model training and fine-tuning, which forces the model to learn language-specific patterns, is not yet studied for the Devanagari script.

Developing a customized loss function to complement cross-entropy loss is essential when dealing with sophisticated scripts like Devanagari, employed in languages like Nepali and Hindi. Typically, cross-entropy loss penalizes the inaccurate categorization of each character separately, which may not adequately address the complexities of scripts that include several character combinations and contextual relationships.

An optimized loss function can incorporate the linguistic feature of the Devanagari script, including more efficient processing of conjuncts and modifiers and enhanced management of the sequence and context sensitivity necessary for precise speech recognition. Adapting or enhancing the cross-entropy loss by considering these aspects, the model can enhance its resilience, reducing error rates while improving its capacity to generalize from training data to real-world scenarios. This overall purpose serves as the primary motivation for our work.

3 Methodology

3.1 Fine-Tuning and Parameter-Efficient Adaptation of Pre-Trained Models

Fine-tuning a large, pre-trained model is critical for adapting it to the specific characteristics of a new language dataset. Initially, we perform full-parameter fine-tuning to reintroduce language-specific patterns into the model. Let W represent the model weight matrix, with updates ΔW derived as $\Delta W = \alpha \times (-\nabla L_W)$, where α is the learning rate and L_W the loss function. The updated weights become $W' = W + \Delta W$. This stage is performed on a representative subset, $D_{introduce}$, of the entire data set D.

To achieve efficient adaptation with fewer computational resources, we further apply parameter-efficient fine-tuning, leveraging the low intrinsic dimensionality of the model for new tasks. Rather than updating the full weight matrix, we approximate the weight update ΔW by decomposing it into two smaller matrices: $\Delta W = W_A W_B$, where $W_A \in R^{A \times r}$ and $W_B \in R^{r \times B}$, and r is a reduced dimension. This approach, implemented through low-rank adaptation (LoRA), keeps the original weights (W) frozen, updating only the smaller matrices W_A and W_B, thus forming a lightweight adapter for the specific task. Figure 1 illustrates this LoRA fine-tuning architecture.

3.2 Error Analysis

Identification of the transcription errors of the existing model is very important while conducting the fine tuning of larger models. Both before and after we fine-tuned the parameters using the default loss function (Cross Entropy Loss in the case of *Whisper*), we observed a similar pattern of errors. A few samples of transcription along with an error description are presented in Table 1.

Based on our inspection, we identified that the model was unable to predict the proper order of the vowel markers (ाक vs. क ा). Likewise, the model sometimes fails to identify the similar sounding consonants (श vs स vs ष OR व vs ब). Another issue arises when dealing with complex characters. In Devanagari script, a complex character typically consists of multiple consonants or vowels, along with markers or special characters. For example, क्ष is a combination of क + ् + ष. Since all three characters are valid tokens in the Whisper model, the way cross-entropy loss cannot well represent the scenarios when the model predicts only two tokens क and ् instead of three tokens क , ् and ष.

This analysis leads us to the conclusion that handling the character complexity, positional awareness of the markers and special symbols, and properly choosing the similar sounding

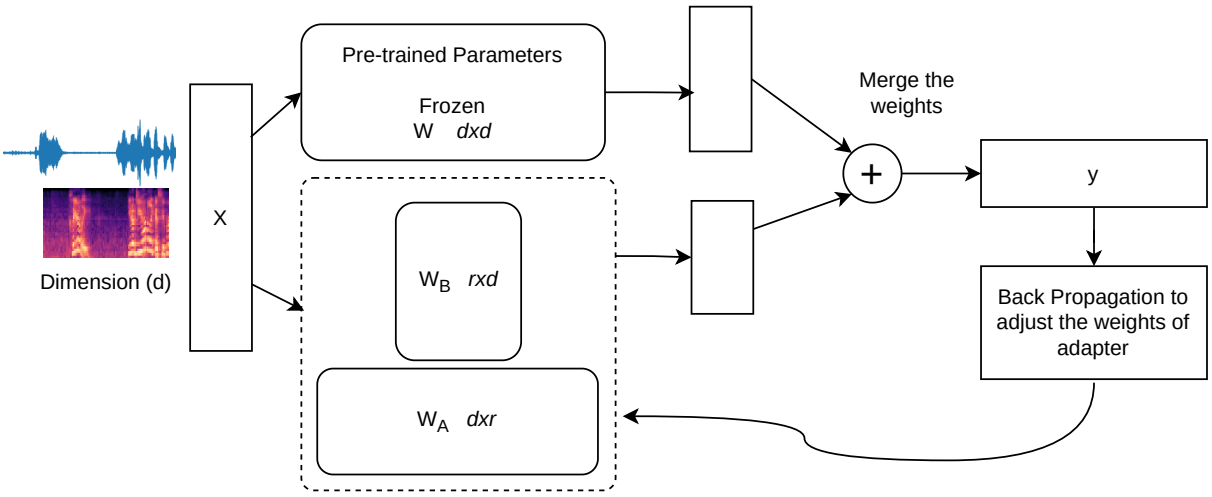

Figure 1: LoRA fine tuning of base model

letter causes the higher Character Error Rate (CER) and Word Error Rate (WER) in the Whisper models. We can resolve this by forcing the model to learn those patterns during fine-tuning. We designed a custom loss function based on the rules that detect the transcription error and penalize the cross-entropy loss.

3.3 RBCCL: Rule-Based Character Constituency Loss

Character constituency in the Devanagari script refers to the organizational arrangement of individual vowels, consonants, and other characters to create coherent units such as syllables, words, or phrases. The Devanagari language employs letters that represent a consonant, a vowel, or a mixture of both. Letters also combine with diacritical signs (matras) to denote complete syllables. The constituency is essential for recognizing the linguistic arrangement of words, since the configuration of letters determines both the sound and the meaning. These types of complexities are not captured by widely used loss functions such as CTC and/or cross-entropy loss. This motivates us to explore the complementary loss function, which forces the model to learn patterns guided by the rules and proves useful for the Devanagari script. We named our method **RBCCL** which statnds **R**ule-**B**ased **C**haracter **C**onstituency **L**oss.

3.3.1 Character Constituency Rules

It is very important to document the generic and script-specific rules. For Devanagari text, we considered following character constituency rules as listed in the list below.

- Rule 1: The vowel markers should appear only after consonants
- Rule 2: The vowel markers should not be added to vowel characters
- Rule 3: Similar sounding characters should be correct

The instances of those rules present in the word will be used while computing RBCCL. Each of these rules (Rules 1, 2, and 3) forms the basis for calculating the counts C_m, C_n, C_e, and C_a. We assess the labels and predicted text by evaluating its adherence to these rules, allowing us to measure character correctness according to the following:

C_m: Represents the count of instances in the ground truth (label) where Rules 1, 2, and 3 are correctly applied. This is obtained by verifying compliance with each rule in the true labels and accumulating the instances.

C_n: Counts the instances in the predicted text where all rules should apply. This indicates the expected adherence to each rule based on the prediction output.

C_e: Measures errors in the predicted text by calculating the instances where rules were not followed, even though they should have been based on the ground truth. This difference identifies specific rule violations.

C_a: Counts additional instances where rules were applied in the predicted text, despite no

True Label	Transcription	Observation
टाउकोमा राख्ने संगठनको	टाउँकोाम राख्ने सङ्गठनक	- उँ ⇒ उ, ऊ, उँ, उं are similar sounding vowel and its variations combined with matras. - कोाम ⇒ ◌ा , but ◌ा should be associated with म forming मा as कोा - संग and सङ्ग sounds similar
आकर्षणलाई अझै	आकर्सन्दलाई अजै	- स and ष sound similar - जै and झै are different sound but some speaker with different mother tongue generate similar sounding speech
समय बिताएका छन्	समय विटाएका छन	- ब and व sound similar. In most case they are unable to recognized by the model - There are some community who unable to produce ता as they do not have त syllable in their mother tongue. So they sound like टा
यस क्षेत्रलाई आफ्नो	यस क्◌ेतरलाई आफनो	- क्ष is complex character made up of 3 characters (क + ◌् + ष). While recognizing only two characters क + ◌् are capture which leads to half letter क् - त्र is also a complex character made from त + ◌् + र but त र is captured

Table 1: Analysis of transcription of full parameter fine tuned $Whisper - Large - V2$ model with Cross Entropy Loss ($\mathcal{L}_{CE}$).

corresponding rule requirement in the ground truth.

3.3.2 Error Rate ($\mathcal{L}_{ER}$)

The error rate provides the proportion of error out of all predictions.

$$ErrorRate = \frac{C_e}{C_n} \tag{1}$$

The error rate we calculated in Equation (1) can be used for the loss function. We should perform some mathematical operations to smooth the value, prevent extreme gradients, and avoid negative values and logs of zero. Following are the formulae for computing $\mathcal{L}_{ER}$:

$$\mathcal{L}_{ER} = log(\frac{C_e}{C_n} + 1) \tag{2}$$

Equation (2) can be further expanded for batch processing and reduced to mean loss of batch as shown in Equation (3).

$$\mathcal{L}_{ER} = \frac{\sum_i^{|B|} log(\frac{C_e(B_i)}{C_n(B_i)} + 1)}{|B|} \tag{3}$$

Where, $|B|$ is number of batch and B_i represents the individual label and predicted labels used to compute necessary counts.

3.3.3 Coverage Penalty ($\mathcal{L}_{CP}$)

Now we have to penalize the loss function for any imbalance between the number of ground truth instances and the number of predictions. We can accomplish this by applying the coverage penalty:

$$CoveragePenalty = \frac{|C_m - C_n|}{C_m} \tag{4}$$

This penalizes the difference between predictions and ground truth, normalized by the number of ground truth instances. Using the same convention as Equation 3, the loss value based on the coverage penalty will be computed, as in Equation 5.

$$\mathcal{L}_{CP} = \frac{\sum_i^{|B|} log(\frac{|C_m(B_i) - C_n(B_i)|}{C_m(B_i)} + 1)}{|B|} \tag{5}$$

3.3.4 Penalizing for Additional Rule ($\mathcal{L}_{AR}$)

We can penalize any additional or missing rules explicitly by defining an additional loss term based on the absolute number of excess or missing rules. This loss can be calculated, as in

Equation (7).

$$AdditionalRulePenalty = \frac{C_a}{C_m} \quad (6)$$

$$\mathcal{L}_{AR} = \frac{\sum_i^{|B|} log(\frac{C_a(B_i)}{C_m(B_i)} + 1)}{|B|} \quad (7)$$

3.3.5 Combining all loss

All loss values are combined as a weighted sum to obtain the total RBCCL.

$$\mathcal{L}_{RBCCL} = \beta_1 \times \mathcal{L}_{ER} + \beta_2 \times \mathcal{L}_{CP} + \beta_3 \times \mathcal{L}_{AR}$$

We can adjust the individual value of β_1, β_2, β_3 depending on whether we want to prioritize error rate, coverage penalty, or additional rule.

3.4 Combining Cross Entropy Loss with RBCCL

We can combine the cross-entropy loss with newly computed loss values. For computational efficiency, we can combine these two losses using a weighted sum.

$$\mathcal{L} = \alpha \times \mathcal{L}_{CE} + \beta \times \mathcal{L}_{RBCCL}$$

Where,

$\mathcal{L}$ is a total loss value

$\mathcal{L}_{CE}$ is a Cross Entropy Loss

$\mathcal{L}_{RBCCL}$ is a Rule-BasedCharacter Constituency Loss computed as of Section 3.3

α and β are the weight which control the emphasis on each part of the combined loss computed.

For the sake of simplicity, and for easier parameter setting, we generalized the weight (parameters) as follows.

$$\beta = (1 - \alpha)$$

$$\beta_1 = \beta_2 = \beta_3 = \beta$$

Choosing the right values for α and β is important. A higher value of α puts more emphasis on obtaining the correct classification with the right probability, whereas a higher value of the β directly penalizes the proportion of incorrect predictions and the number of prediction differences.

4 Experimental Setup

4.1 Speech Corpus

We, the authors, are native Nepali speakers, a language that uses the Devanagari script for writing. So, we decided to use the Nepali Speech Corpus (SLR54) (Kjartansson et al., 2018) for our experiment, which is available under the Open Speech Language Resources[1]. This is the only publicly available speech corpus and is suitable for ASR tasks. A small subset of the SLR54 dataset is used for the experiment. This closely resembles the very low-resource setting and also allows for conducting experiments in a limited computing environment.

4.2 Selection of pre-trained model

The *Whisper* (Radford et al., 2022) model family is used as the pre-trained speech model. They have *tiny*, *base*, *small*, *medium*, and *large* models ranging from 39M parameters to 1550M parameters. We used the multilingual *Large V2* model, which also includes the Nepali language. The claimed WER of selected model is 47.1% (Radford et al., 2022). However, a thorough examination reveals that the transcribed texts align rather with Hindi. This is due to the fact that both languages, being written in Devanagari script, utilize the same token. To solve this issue, we decided to reintroduce the language by full-parameter fine-tuning using 30 minutes of the labeled dataset. In this experiment, we used the default loss function.

4.3 Choosing parameters and Experimental setup

For this experiment, we used the Hugging Face Transformer library[2] for training, fine-tuning, data processing, etc.

We run each training for a total of 5 epochs. All training parameters are summarized in Table 2. These parameters are obtained from hyperparameter tuning.

5 Result and Discussion

We performed various combinations of the experiments that involve the full parameter fine-tuning and LoRA fine-tuning. All experiments

[1] [SLR54] - https://www.openslr.org/54/

[2] Hugging Face: https://huggingface.co/docs/transformer

Details	Parameters
Full parameter fine tuning	- Precision:float16(*fp16*) - 8-bit Adam optimizer (*adamw_bnb_8bit*) - learning rate: $1e-5$ - batch size: 4
PEFT (LoRA) Parameters	- r : 32 - *alpha* : 64, - *dropout* : 10%
Loss Weight Parameters	- α: 0.7 - β: 0.3

Table 2: Model training parameters for both full-parameter and LoRA fine-tuning

we performed and used for comparison purposes, with their descriptions and results in terms of WER and CER, are listed in Table 3.

The $Whisper-Large-V2$ (Radford et al., 2022) is a base pre-trained model. Its WER for Nepali is reported as 47.1%. When inspecting the output, we found that the transcription of the model more closely resembles the Hindi text. To solve this issue, we reintroduced the Nepali speech by full-parameter fine-tuning the model using a 30-minute labeled dataset. This fine-tuning itself significantly improved the model, resulting in a WER of 36.2%. All other subsequent experiments are now based on this newly fine-tuned model, which we call the fine-tuned base model (FT_{base}).

The performance of the fine-tuned model after incorporating the Cross-Entropy loss($\mathcal{L}_{CE}$) and RBCCL($\mathcal{L}_{RBCCL}$) indivisually did not show substantial improvement compared to FT_{base}. Specifically, $\mathcal{L}_{\mathcal{RBCCL}}$, although designed to enhance the robustness of the model, achieved a WER of 34.2%, which represents only a marginal improvement over baseline FT_{base} with a WER of 36.2%. This limited improvement highlights the challenges of effectively using RBCCL in this context. In contrast, using cross-entropy loss alone during fine-tuning yielded a more notable improvement, reducing the WER to 31.20%.

Among the various approaches explored, the full parameter fine-tuning with the combined loss function($\mathcal{L}$) achieved the best performance, with a WER of **23.41%** and a CER of **5.37%**. This represents a significant improvement of 23.69% relative to the pre-trained *Whisper−Large−V2* model and 12.79% relative to FT_{base}, showcasing the effectiveness of the loss function ($\mathcal{L}$) in leveraging complementary loss functions for improved transcription accuracy.

Full-parameter fine-tuning requires substantial computational resources and time investment. To streamline the process and achieve a high performance model efficiently, we explore directly fine-tuning the pre-trained *Whisper−Large−V2* model. This approach yielded a WER of **25.15%** and a CER of **6.51%**. Although this performance is slightly lower than the best results achieved through further fine-tuning (23.41% WER and 5.37% CER), it represents a significant improvement over the baseline *Whisper−Large−V2* model with a WER of 47.1%. This outcome demonstrates that skipping the initial fine-tuning step with FT_{base} is a viable alternative to obtain a model with competitive performance. Direct fine-tuning of *Whisper−Large−V2* offers a balance between accuracy and efficiency, reducing the effort required to achieve substantial improvements in both WER and CER. These results are particularly encouraging for scenarios where computational resources or time are limited, highlighting the flexibility and adaptability of the proposed fine-tuning strategies.

For LoRA-based fine-tuning, the combined loss function ($\mathcal{L}$) led to a WER of 31.50% and a CER of 7.47%. Although this approach did not outperform full parameter fine-tuning with $\mathcal{L}$, it demonstrated a clear advantage over cross-entropy-based training alone. The results suggest that incorporating $\mathcal{L}$ into the LoRA fine-tuning framework effectively balances model complexity and performance, achieving competitive results with reduced parameter updates.

The findings highlight the efficacy of carefully designed loss functions, especially when combining complementing objectives, to significantly improve model performance in low-resource ASR tasks. The exceptional results obtained using $\mathcal{L}$-based training, particularly in terms of full-parameter fine-tuning, highlight its importance as an essential element for improving ASR models in Nepali.

Experiment Description	WER%	CER%
$Whisper - Large - V2$ by (Radford et al., 2022)	47.1	×
Nepali ASR module full-parameter fine-tuned on mms-1b by (Ghimire et al., 2023a)	×	6.77%
FT_{base}: full parameter fine-tuning to (re)introduce Nepali to $Whisper - Large - V2$ with $\mathcal{L}_{CE}$	36.2	15.4
$FT_{\mathcal{L}_{CE}}$: full parameter fine-tuning of FT_{base} model with $\mathcal{L}_{CE}$	31.2	7.47
$FT_{\mathcal{L}_{RBCCL}}$: full parameter fine-tuning of FT_{base} model with $\mathcal{L}_{RBCCL}$	34.2	14.2
$FT_{\mathcal{L}}$: full parameter fine-tuning of FT_{base} model with $\mathcal{L}$	**23.41**	**5.37**
$FT_{\mathcal{L}}$: full parameter fine-tuning of $Whisper - Large - V2$ model with $\mathcal{L}$	25.15	6.51
$FT\_LoRA_{\mathcal{L}_{CE}}$: LoRA fine-tuning of FT_{base} model with $\mathcal{L}_{CE}$	32.60	8.01
$FT\_LoRA_{\mathcal{L}}$: LoRA fine-tuning of FT_{base} model with $\mathcal{L}$	**31.50**	**7.47**

Table 3: WER % of models produced during experiment

6 Conclusion

Low-resource fine-tuning of large language models is a prevalent and growing practice, particularly in the context of speech-related tasks. Since Nepali is a low-resource language, the fine-tuning task has received relatively less attention. Our study focused on introducing the language-specific loss function to regularize and force the model to learn the language-specific patterns. We proposed a loss function based on the set of rules built on a basic mathematical foundation. We named it Rule-Based Character Constituency Loss (**RBCCL**).

Our strategy involves the initial (re)introduction of the language into the larger model, achieved through full-parameter tuning with default training parameters. After forming the base model, we apply our loss function to complement the cross-entropy loss. We experimented with both full-parameter fine-tuning and adapter-based fine-tuning using LoRA. The complemented loss function in both cases compelled the model to learn features that the default loss function failed to capture effectively.

Although we observed significant improvements in the implementation of the suggested strategy, there is still plenty of room to enhance the precision of the mode. Our study focused on the Whisper model. We could expand the study to include larger models and compare the corresponding results in other languages that use the Devanagari script.

7 Limitations

Throughout the experiments, we only investigated the *Whisper* model. *Whisper* uses cross-entropy loss. We demonstrate through a set of experiments that our loss function nicely complements cross-entropy loss. However, we did not extensively explore the impact of this new function on other loss functions, such as the CTC loss. We have a plan to extend this to tests on other loss functions as well.

Another limitation of this work is computing resources. Due to a lack of the high computing resources demanded by the larger speech model, we were unable to use the full available dataset. We believe that using a full dataset further enhances accuracy. We focused on the Nepali language, but there are many other languages that use the Devanagari script. Therefore, we can expand the work to include other languages as well.

Note: All the datasets (test, train, and validation) and the final models can be accessed through Information and Language Processing Research Lab's website (https://il-prl.ku.edu.np).

References

A Arunkumar, Vrunda Nileshkumar Sukhadia, and Srinivasan Umesh. 2022. Investigation of Ensemble Features of Self-Supervised Pretrained

Models for Automatic Speech Recognition. In *INTERSPEECH 2022*, pages 5145–5149. ISCA.

Alexei Baevski, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. 2022. Unsupervised speech recognition. In *Advances in Neural Information Processing Systems 34*, arXiv:2105.11084, pages 27826–27839.

Alexei Baevski, Steffen Schneider, and Michael Auli. 2020a. Vq-Wav2vec: Self-Supervised Learning of Discrete Speech Representations. ArXiv preprint arXiv:1910.05453 (2020).

Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020b. Wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations. In *Advances in neural information processing systems 33*, pages 12449–12460.

Yu-An Chung, Yu Zhang, Wei Han, Chung-Cheng Chiu, James Qin, Ruoming Pang, and Yonghui Wu. 2023. W2v-BERT: Combining contrastive learning and masked language modeling for self-supervised speech pre-training. *Preprint*, arxiv:2108.06209 [cs, eess].

Kartik Dutta, Praveen Krishnan, Minesh Mathew, and C.V. Jawahar. 2018. Towards spotting and recognition of handwritten words in indic scripts. In *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 32–37.

Rupak Raj Ghimire, Bal Krishna Bal, and Prakash Poudyal. 2023a. Active learning approach for fine-tuning pre-trained ASR model for a low-resourced language: A case study of nepali. In *Proceedings of the 20th International Conference on Natural Language Processing (ICON)*, pages 82–89. NLP Association of India (NLPAI).

Rupak Raj Ghimire, Bal Krishna Bal, Balaram Prasain, and Prakash Poudyal. 2023b. Pronunciation-aware syllable tokenizer for nepali automatic speech recognition system. In *Proceedings of the 20th International Conference on Natural Language Processing (ICON)*, pages 36–43. NLP Association of India (NLPAI).

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. HuBERT: Self-Supervised Speech Representation Learning by Masked Prediction of Hidden Units. *IEEE/ACM Transactions on Audio, Speech, and Language Processing 29*, pages 3451–3460.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Shreya Khare, Ashish Mittal, Anuj Diwan, Sunita Sarawagi, Preethi Jyothi, and Samarth Bharadwaj. 2021. Low Resource ASR: The Surprising Effectiveness of High Resource Transliteration. In *INTERSPEECH 2021*, pages 1529–1533. ISCA.

Oddur Kjartansson, Supheakmungkol Sarin, Knot Pipatsrisawat, Martin Jansche, and Linne Ha. 2018. Crowd-Sourced Speech Corpora for Javanese, Sundanese, Sinhala, Nepali, and Bangladeshi Bengali. In *6th Workshop on Spoken Language Technologies for Under-Resourced Languages (SLTU 2018)*, pages 52–55. ISCA.

Da-Rong Liu, Kuan-Yu Chen, Hung-Yi Lee, and Lin-shan Lee. 2018. Completely Unsupervised Phoneme Recognition by Adversarially Learning Mapping Relationships from Audio Embeddings. In *INTERSPEECH 2018*, pages 3748–3752. ISCA.

Wei Liu, Ying Qin, Zhiyuan Peng, and Tan Lee. 2024. Sparsely shared lora on whisper for child speech recognition. In *ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 11751–11755.

Jian Luo, Jianzong Wang, Ning Cheng, and Jing Xiao. 2021. Loss Prediction: End-to-End Active Learning Approach For Speech Recognition. In *2021 International Joint Conference on Neural Networks (IJCNN). IEEE.*

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`.

Vineel Pratap, Andros Tjandra, Bowen Shi, Paden Tomasello Arun Babu, Sayani Kundu, Ali Elkahky, Zhaoheng Ni, Apoorv Vyas, Maryam Fazel-Zarandi Alexei Baevski, Yossi Adi, Xiaohui Zhang, Wei-Ning Hsu, Alexis Conneau, and Michael Auli. 2023. Scaling speech technology to 1,000+ languages. arXiv preprint arXiv:2305.13516 (2023).

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2022. Robust speech recognition via large-scale weak supervision. *Preprint*, arxiv:2212.04356 [cs, eess].

Sunil Regmi and Bal Krishna Bal. 2021. An end-to-end speech recognition for the Nepali language. In *Proceedings of the 18th International Conference on Natural Language Processing (ICON)*, pages 180–185, National Institute of Technology Silchar, Silchar, India. NLP Association of India (NLPAI).

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. Wav2vec: Unsupervised Pre-training for Speech Recognition. In *INTERSPEECH 2019*. ISCA.

Rupesh Shrestha, Basanta Joshi, and Suman Sharma. 2021. Nepali Speech Recognition using LSTM-CTC. In *10th IOE Graduate Conference*, pages 170–174.

Satwinder Singh, Feng Hou, and Ruili Wang. 2023. A Novel Self-training Approach for Low-resource Speech Recognition. In *NTERSPEECH 2023*, pages 1588–1592. ISCA.

Zheshu Song, Jianheng Zhuo, Yifan Yang, Ziyang Ma, Shixiong Zhang, and Xie Chen. 2024. Lora-whisper: Parameter-efficient and extensible multilingual asr. *ArXiv*, abs/2406.06619.

Changhan Wang, Yun Tang, Xutai Ma, Anne Wu, Dmytro Okhonko, and Juan Pino. 2020. Fairseq S2T: Fast speech-to-text modeling with fairseq. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: System Demonstrations*, pages 33–39, Suzhou, China. Association for Computational Linguistics.

Qiantong Xu, Alexei Baevski, and Michael Auli. 2022. Simple and effective zero-shot cross-lingual phoneme recognition. In *INTERSPEECH 2022*, pages 2113–2117. arXiv.

Jingyuan Yang, Rongjun Li, and Wei Peng. 2022. ASR error correction with constrained decoding on operation prediction. *Preprint*, arxiv:2208.04641 [cs, eess].

Zhisheng Zheng, Ziyang Ma, Yu Wang, and Xie Chen. 2023. Unsupervised Active Learning: Optimizing Labeling Cost-Effectiveness for Automatic Speech Recognition. In *INTERSPEECH 2023*, pages 3307–1532.