Robust Neural Machine Translation for Abugidas by Glyph Perturbation

Hour Kaing[1] Chenchen Ding[1] Haiyue Song[1] Jiannan Mao[21] Hideki Tanaka[1] Masao Utiyama[1]
[1]ASTREC, UCRI, NICT, Japan [2]Gifu University, Gifu, Japan
[1]{hour_kaing, chenchen.ding, haiyue.song, hideki.tanaka, mutiyama}@nict.go.jp
[2]mao@mat.info.gifu-u.ac.jp

Abstract

Neural machine translation (NMT) systems are vulnerable when trained on limited data. This is a common scenario in low-resource tasks in the real world. To increase robustness, researchers intently added realistic noise in the training phase. Noise simulation using text perturbation has been proven to be efficient in writing systems that use Latin letters. In this study, we further explore perturbation techniques on more complex abugida writing systems, for which the visual similarity of complex glyphs is considered to capture the essential nature of these writing systems. Besides the generated noise, we investigated three training approaches such as subword regularization, adversarial training, and consistency training. Finally, we propose to combine them to maximize the translation performance. We conducted experiments on six languages: Bengali, Hindi, Myanmar, Khmer, Lao, and Thai. Our training approach obtained the best performance for five languages.

1 Introduction

Neural machine translation (NMT) systems have been shown to be vulnerable in noisy settings, where slightly modified inputs cause serious translation failures [1, 4]. Boucher et al. [2] showed that techniques using pre-trained language models cannot prevent this. This drawback is more disastrous in low-resource scenarios, where the model's robustness becomes a crucial issue.

Several text perturbation techniques have been developed to improve robustness by introducing synthesized textual noise [12, 6]. Most techniques mostly focus on languages that use alphabetic systems, such as Latin letters. As a more complex writing system, Chinese characters were investigated by Zhang et al. [20]. In the present study,

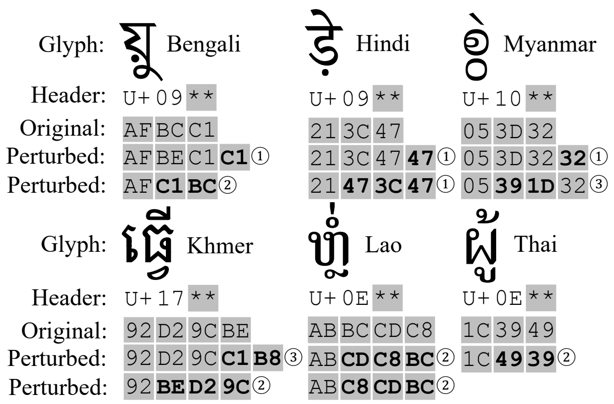

Figure 1 Homoglyph perturbation examples for various abugida systems. The Unicode of each character is listed below the glyph. Perturbed characters are emphasized in bold font. Various patterns cause homoglyphs: 1) repetition, 2) permutation, and 3) decomposition (e.g., BE → C1 B8 in Khmer).

we further fill the gap in text perturbation techniques for understudied abugida writing systems, which vary and are used widely in South-East Asia.

A reasonable perturbation technique should produce meaningful and readable text that is indistinguishable for humans, but disastrous for a system's prediction [11]. Visually similar glyphs or homoglyphs[1)] were investigated in Eger et al. [6], Boucher et al. [2], and Le et al. [11] obtained realistic samples from large corpora. As a primary contribution, we further develop these previous studies for abugida writing systems. Some exemplary homoglyphs in various abugida systems are illustrated in Figure 1.

To address noise, we propose a training strategy that leverages adversarial training, subword regularization, and consistency training. We selected six languages that use abugida systems, Bengali, Hindi, Myanmar, Khmer, Lao, and Thai, and experimented on them for low-resource tasks. Overcoming noisy perturbations improved the robustness, with non-degenerate performance.

1) I.e., glyphs with identical visuals, but different encodings.

2 Background: Abugidas

An abugida is a writing system that combines features of both syllabic and segmental systems. Text is written as a sequence of syllables, which resemble Japanese hiragana, but can be broken down into separate consonants and vowels, as in a segmental system. A typical abugida syllable consists of a base consonant accompanied by a default vowel or additional vowels represented by diacritics. In computer systems, these syllables are rendered into glyphs, which are visual symbols in the rendering process. A glyph represents a letter or a composition of multiple letters. For example, in Latin, the letter *a* is a glyph, and combined with a grave accent (diacritic), it becomes another glyph *à*; similarly, in abugidas, as shown in Figure 2, a consonant is represented by a glyph, as in (a), and combined with multiple diacritics to become another glyph, as in (b). As in Figure 1, similar glyphs or homoglyphs commonly occur in the composition of complex diacritics, which have numerous patterns and are difficult to engineer. We explore such diacritic composition from human-generated corpora. Hereafter, we use the term *glyph* to refer to a visual symbol and *glyph token* to refer to its Unicode characters.

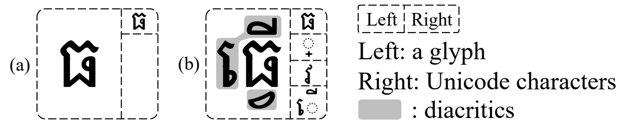

Figure 2 Examples of Khmer glyphs. (a) is a glyph without diacritics and (b) with diacritics.

3 Proposed Method

3.1 Perturbation for Abugidas

3.1.1 Overall Processing

Given a sentence $\mathbf{x} = (x_1, \ldots, x_n)$, each token x_i has a chance of being replaced with an adversarial candidate $x' \in V$ chosen based on its similarity score w.r.t x_i [6], where V is vocabulary that contains all possible tokens, including clean and noisy tokens[2]. The perturbation probability for each targeted token x_i is formulated as

$$g(x'|x_i) = \begin{cases} \alpha \cdot \dfrac{score(x', x_i, \beta)}{Z(x_i)}, & \text{if } x' \neq x_i \\ 1 - \alpha, & \text{otherwise} \end{cases} \quad (1)$$

$$Z(x_i) = \sum_{x'' \in V \setminus \{x_i\}} score(x'', x_i, \beta) \quad (2)$$

$$score(a, b, \beta) = I(s(a, b) \geq \beta) \cdot s(a, b), \quad (3)$$

where $I(\cdot)$ is an indicator function; α and β control the chance of x_i being perturbed and the similarity threshold, respectively; and $s(a, b)$ is a similarity function between the continuous vectors of two tokens a and b, for example, the cosine similarity $s(a, b) = \cos(v(a), v(b))$, and where $v(\cdot)$ is a vector. The overall perturbation process is illustrated in Figure 3. Next, we present the process for obtaining V from corpora that contain diverse adversarial candidates in Section 3.1.2, and describe how vector $v(\cdot)$ is represented by an image in Section 3.1.3 and by counting diacritics in Section 3.1.4.

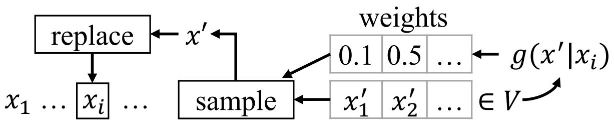

Figure 3 Overall perturbation processing.

3.1.2 Vocabulary Construction

This step is similar to a typical vocabulary preparation process that consists of tokenization and unique token extraction to obtain V. Specifically, we categorize each character as a consonant or diacritic based on Unicode Standard data. After that, we base tokenization on the consonant position such that each token starts with a consonant followed by many or zero diacritics, and then obtain a list of unique tokens as V. Additionally, because the similarity is mostly around the diacritics, we want to perturb only the diacritic parts of each targeted token. To achieve this, our trick is to replace the consonant counterpart of each token in V with that of the targeted token x_i, which varies every time step i. This trick is based on the assumption that the visual form of the consonant never changes when it is combined with diacritics. However, we discovered one case in Bengali and Hindi in which the base consonant changed its visual form. Hence, we simply skipped the perturbation for such case.

3.1.3 Image-based Glyph Embeddings (IGE)

We convert each glyph image[3] into a linear vector of $m \cdot n$ dimensions by arranging rows in the $m \times n$ matrix,

2) As V is fixed in practice, we skip the process if $x_i \notin V$.

3) We used Pillow9.4.0 and Google Noto Serif fonts with 100px for all languages.

where each entry corresponds to a pixel in the grayscale image. The pixel values range from 0 (representing the empty area) to 255 (representing the visible part of a glyph). Because the image size varies greatly across glyphs, we predetermine the maximum size $m \times n$ based on all glyphs and then render them into the $m \times n$ size. They must align to the left on the horizontal axis and to the middle on the vertical axis. Additionally, we empty the pixels that correspond to the consonant to ensure that the similarity value is not affected by the common pixels of the base consonant. Finally, we use the cosine similarity function for IGE, which is defined as $s(a, b) = \cos(v(a), v(b))$.

3.1.4 Diacritic-Count Embeddings (DGE)

A simpler approach involves counting the diacritics that exist in a glyph token and how many times they occur. Specifically, a glyph token is represented by a frequency vector, where each entry corresponds to a diacritic in the language and the value of each entry is the count of the corresponding diacritic in the glyph token. Additionally, we smooth each frequency value using an exponent γ. For instance, if a language l has a set of diacritics $\{a, b, c\}$ and a glyph token consists of diacritics acc, DGE represents it using a frequency vector $[1, 0, 2]^\gamma$ because a occurs once and c occurs twice. Using DGE, we can identify two glyphs that have similar sets of diacritics, regardless of the order of the diacritics. We set $\gamma = 0.3$ in all experiments and use the inverse Euclidean distance as the similarity function, which is defined as $s(a, b) = (Euclidean(v(a), v(b)) + 1)^{-1}$.

3.2 Robust NMT Training

To generalize a model in the presence of noisy inputs, we explore variable noises and train the model on them. Previously, Eger et al. [6] proposed adversarial training (AT) that perturbs the training data in the same manner as the test data, while Kudo [10] introduced subword regularization (SR), which samples variable subwords for training. Both techniques have been proven effective against noisy inputs. In this work, we combine both techniques by first perturbing each training sample and then sampling variable subwords for each perturbed sample. For generalization, we extract V to perturb the test data from external corpora, whereas we extract V to perturb the training data only from the training data itself. We hypothesize that visually similar text exists in the training data and can be used in our robust training. In this study, we use unigram subword tokenizer [10], and sample subwords with the n-best size $l = \infty$ and distribution smoothness $\mu = 0.2$.

We further adopt consistency training [17] to enforce the model's prediction being invariant between noisy and clean inputs. Let $\mathbf{x}$ is an input sequence and $\mathbf{x}'$ is its noisy variant. Therefore, given a training set $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, the objective function can be expressed as

$$\mathcal{L}(\theta) = \sum[-\frac{1}{2}\log p_\theta(\mathbf{y}_i|\mathbf{x}_i) - \frac{1}{2}\log p_\theta(\mathbf{y}_i|\mathbf{x}'_i) + \lambda D(p_\theta(\mathbf{y}_i|\mathbf{x}_i)||p_\theta(\mathbf{y}_i|\mathbf{x}'_i))], \quad (4)$$

where θ is a set of model parameters and $D(\cdot||\cdot)$ is a non-negative distance metric between two distributions that are controlled by the hyperparameter λ. Following Wang et al. [17], we use Kullback–Leibler divergence for $D(\cdot||\cdot)$ and set $\lambda = 0.2$.

4 Experiments

4.1 Settings

We experimented on six abugida languages: Bengali (bg), Hindi (hi), Myanmar (my), Khmer (km), Lao (lo), and Thai (th). We constructed V from the cleaned CommonCrawls [18, 3] and evaluated translation performance on the Asian Language Treebank dataset [16] from abugida languages to English. We tokenized the training/validation/test sets using SentencePiece, with a joint vocabulary of 4k. We perturbed the test data using IGE and the training data using either IGE or DGE.

We used the transformer architecture for all the models and implemented them using Fairseq [13] in our experiments. We trained all the models on the eight GPUs (Tesla V100 SXM2 with 32 GB memory) and the number of parameters was approximately 54 million. We mostly based the configuration on Guzmán et al. [7], which was specifically designed for the Indic low-resource NMT setting. However, we further fine-tuned the number of epochs and found that increasing the number of epochs to 1k achieved improvements across all models.

4.2 Results and Discussions

We evaluated the performance of the vanilla model (Base) with respect to our perturbation technique using $\beta = 1$. The results are presented in Figure 4. We ob-

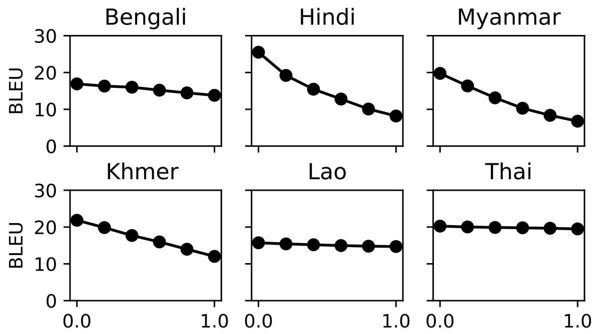

Figure 4 BLEU scores of NMT with α from 0.0 to 1.0, with a step of 0.2 on the x-axis, and β set to 1.0.

served performance degradation across all languages, with a more significant influence observed for Hindi, Myanmar, and Khmer. This could be attributed to the higher number of perturbed glyphs in these languages compared with Bengali, Lao, and Thai. The details regarding the number of perturbed glyphs per line are presented in Figure 5.

Table 1 demonstrates the effectiveness of the robust approaches against perturbation, even with SR, and show the improved performance of SR with consistency training, SR$_{ct}$. Our robust training using IGE and DGE further improved robustness performance, in most cases, while maintaining comparable performance on clean input, as discussed in the Appendix A. It is noteworthy that our approach only exploited the visual similarity within the training set and the results supported our hypothesis.

In this study, we only examined the impact of perturbation using homoglyphs with $\beta = 1$. Reducing β, for example, to 0.95, would be likely to result in a decrease in NMT performance. However, this may not be a realistic scenario because it would also reduce the readability of the text for humans. We provide examples of perturbed samples with smaller β in Table 3 in the Appendix. Interestingly, a native Khmer speaker was able to read samples with a β value of 0.9, which indicates that the perturbed text was still readable at lower β values. However, more extensive assessments with native speakers are required in our future study to better understand the impact of β on text readability.

Table 1 BLEU results on perturbed inputs. SR$_{ct}$ denotes SR with consistency training. * and † indicate the significance values $p < 0.01$ and $p < 0.05$ compared with SR$_{ct}$, respectively.

	bg	hi	my	km	lo	th
Base	13.8	8.2	6.8	12.0	14.7	19.5
SR	16.6	11.0	15.3	17.9	16.8	20.6
SR$_{ct}$	18.8	14.0	17.2	19.5	18.7	22.5
IGE	18.1	14.8*	18.3*	21.1*	19.1†	22.5
DGE	19.0*	14.4†	18.3*	21.2*	18.7	22.5

5 Related Work

5.1 Text Perturbation

Text perturbation has been extensively studied in the literature, with two scenarios: white-box and black-box. In the white-box scenario, the model's gradients are leveraged [12, 5], whereas in the black-box scenario, only the model's input and output are known [12, 5]. Various perturbation operations have been proposed, such as randomly inserting/deleting/replacing/swapping characters [9], character shuffling, perturbation based on the keyboard layout and natural typos [1], extraction of visually similar glyphs of characters [6], and similar embedding subwords [14]. Our study is similar to that of Eger et al. [6], where visually similar glyphs were explored.

5.2 Consistency Training

In various studies, researchers have used consistency training in various ways to enhance the performance of natural language processing (NLP) models. Previously, Wang et al. [17] used consistency training to improve subword tokenization in multilingual models. Xie et al. [19] and Kambhatla et al. [8] improved data augmentation techniques for NMT using consistency training. Furthermore, Park et al. [15] used consistency training on virtual noise to improve the performance of text classification and natural language inference tasks. In this study, we adopted consistency training to regularize our training on the joint sampling of adversarial text and subwords to enhance the robustness of the NMT model against perturbations.

6 Conclusion

In this study, we presented a perturbation approach that leverages visual similarity and introduced a training strategy to maintain the performance of the NMT model. We exposed the vulnerability of the vanilla NMT model through experiments that perturbed test data using monoglyphs, and demonstrated the importance of robust training against text perturbation. The findings of this study can aid future research effort in evaluating the generalization capabilities of NMT models, particularly for low-resource settings and understudied languages.

References

[1]Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. In **Proc. of ICLR**, 2018.

[2]Nicholas Boucher, Ilia Shumailov, Ross Anderson, and Nicolas Papernot. Bad characters: Imperceptible nlp attacks. In **Proc. of SP)**, pages 1987–2004. IEEE, 2022.

[3]Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In **Proc. of ACL**, pages 8440–8451, 2020.

[4]Javid Ebrahimi, Daniel Lowd, and Dejing Dou. On adversarial examples for character-level neural machine translation. In **Proc. of COLING**, 2018.

[5]Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In **Proc. of ACL**, 2018.

[6]Steffen Eger, Gözde Gül Şahin, Andreas Rücklé, Ji-Ung Lee, Claudia Schulz, Mohsen Mesgar, Krishnkant Swarnkar, Edwin Simpson, and Iryna Gurevych. Text processing like humans do: Visually attacking and shielding nlp systems. In **Proc. of NAACL**, pages 1634–1647, 2019.

[7]Francisco Guzmán, Peng-Jen Chen, Myle Ott, Juan Pino, Guillaume Lample, Philipp Koehn, Vishrav Chaudhary, and Marc'Aurelio Ranzato. The flores evaluation datasets for low-resource machine translation: Nepali-english and sinhala-english. In **Proc. of EMNLP**, page 6098â 6111, 2019.

[8]Nishant Kambhatla, Logan Born, and Anoop Sarkar. Cipherdaug: Ciphertext based data augmentation for neural machine translation. In **Proc. of NAACL**, pages 201–â 218, 2022.

[9]Vladimir Karpukhin, Omer Levy, Jacob Eisenstein, and Marjan Ghazvininejad. Training on synthetic noise improves robustness to natural noise in machine translation. In **Proc. of W-NUT**, 2019.

[10]Taku Kudo. Subword regularization: Improving neural network translation models with multiple subword candidates. In **Proc. of ACL**, pages 66–75, 2018.

[11]Thai Le, Jooyoung Lee, Kevin Yen, Yifan Hu, and Dongwon Lee. Perturbations in the wild: Leveraging human-written text perturbations for realistic adversarial attack and defense. In **Proc. of ACL**, pages 2953–2965, 2022.

[12]Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. Textbugger: Generating adversarial text against real-world applications. In **Proc. of NDSS**, 2018.

[13]Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In **Proc. of NAACL-HLT**, 2019.

[14]Jungsoo Park, Mujeen Sung, Jinhyuk Lee, and Jaewoo Kang. Adversarial subword regularization for robust neural machine translation. In **Proc. of EMNLP**, 2020.

[15]Jungsoo Park, Gyuwan Kim, and Jaewoo Kang. Consistency training with virtual adversarial discrete perturbation. In **Proc. of NAACL**, pages 5646–5656, 2022.

[16]Hammam Riza, Michael Purwoadi, Gunarso, Teduh Uliniansyah, Aw Ai Ti, Sharifah Mahani Aljunied, Luong Chi Mai, Vu Tat Thang, Nguyen Phuong Thai, Rapid Sun, Vichet Chea, Sethserey Sam, Sopheap Seng, Khin Mar Soe, Khin Thandar Nwet, Masao Utiyama, and Chenchen Ding. Introduction of the Asian language treebank. In **Proc. of O-COCOSDA**, pages 1–6, 2016.

[17]Xinyi Wang, Sebastian Ruder, and Graham Neubig. Multi-view subword regularization. In **Proc. of NAACL**, pages 473–482, 2021.

[18]Guillaume Wenzek, Marie-Anne Lachaux, Alexis Conneau, Vishrav Chaudhary, Francisco Guzmán, Armand Joulin, and Edouard Grave. Ccnet: Extracting high quality monolingual datasets from web crawl data. In **Proc. of LREC**, pages 4003–4012, 2020.

[19]Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. Unsupervised data augmentation for consistency training. In **Proc. of NeurIPS**, volume 33, pages 6256–6268, 2020.

[20]Zihan Zhang, Mingxuan Liu, Chao Zhang, Yiming Zhang, Zhou Li, Qi Li, Haixin Duan, and Donghong Sun. Argot: generating adversarial readable chinese texts. In **Proc. of IJCAI**, pages 2533–2539, 2021.

A Additional Results and Analysis

The impact of our perturbation technique with $\beta = 1$ on the dataset is measured by the amount of noise introduced, as shown in Figure 5. The graph indicates that Hindi, Myanmar, and Khmer had a relatively high amount of noise per sentence compared with Bengali, Lao, and Thai. As a result, the performance of the NMT models on Hindi, Myanmar, and Khmer was significantly impacted compared with Bengali, Lao, and Thai. The amount of noise could be increased using highly similar glyphs with $\beta < 1$, as demonstrated in Table 3. However, it is important to ensure that these perturbations are still readable by humans so that robustness studies are realistic.

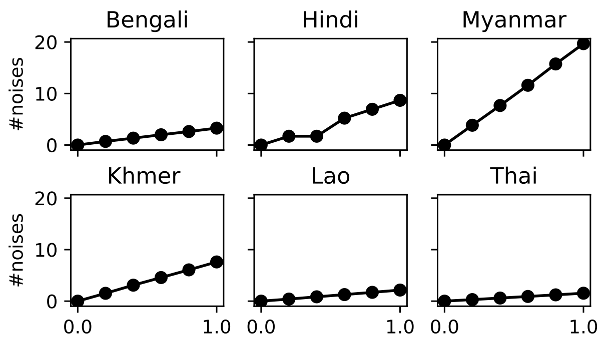

Figure 5 Average amount of noise per sentence on the test set with various α.

Table 2 BLEU results on clean inputs.

	bg	hi	my	km	lo	th
Base	16.9	25.5	19.8	21.8	15.7	20.3
SR	17.1	26.7	19.9	22.3	16.9	20.8
SR_{ct}	19.5	28.9	22.0	24.2	18.8	22.6
IGE	18.7	28.2	21.5	24.0	19.1	22.5
DGE	19.0	28.0	21.4	24.4	18.7	22.4

Table 2 summarizes the performance of all NMT models on a clean test set. Our results show that SR outperformed the baseline for all languages and consistency training further boosted performance. Our training strategy with both IGE and DGE was comparable with SR_{ct}.

Table 3 Perturbed samples. The first one is clean and the followers are perturbed with β of 1, 0.95 and 0.90 in top-down order.

Bengali:

এই ফ্লুটি খুবই সংক্রামক কিন্তু এটি মানুষের দেহে ছড়াতে পারে না।
এই ফ্লুটি খুবই সংক্রামক কিন্তু এটি মানুষের দেহে ছড়াতে পারে না।
এই ফ্লুটি খুবই সংক্রামক কিন্তু এটি মাঁনুষের দেহে ছড়াঁতে পঁারে নঁা।
এই ফ্লুটি খুঁবই সঁংক্রামক কিন্তু এটি মঁানূষের দেহে ছড়ঁাতে পাঁরে নঁা।

Hindi:

फ्लू बेहद संक्रामक है लेकिन इसे मनुष्यों में नहीं हो सकता है।
फ्लू बेहद संक्रामक है लेकिन इसे मनुष्यों में नहीं हो सकता है।
फ्लू बेहद सेंक्रामक है लेकिंन इसे मनुष्यों में नहीं हों सकतो है।
फ्लू बेहद सेंक्रामक है लेकिंन इसे मऩुष्यों में नहाँ हौं सकतीं है।

Myanmar:

တုတ်ကွေး သည် အလွန် အလွယ်တကူ ကူးစက် သော်လည်း လူသားများ သို့ မထုတ်လွှင့်နိုင်ပါ ။
တုတ်ကွေး သည် အလွန် အလွယ်တကူ ကူးစက် သော်လည်း လူသားများ သို့ မထုတ်လွှင့်နိုင်ပါ ။
တုံတ်ကွေ့း သည် အလွှန် အလှွယ်တကူ့ ကူ့စက် သော်လည်း လူ့သားများ သို့ မထုံတ်လွှင့်နှိုင်ပိ ။
တွုတ်ကွေ့း သည် အလွှဲန် အလွှယ်တကွဲ ကူ့စက် သော်လည်ိး လူ့သံးမျ်ား သို့ မထွုတ်လွှင့်နှိုင်ပိ ။

Khmer:

មេរោគនេះងាយឆ្លងតែមិនឆ្លងដល់មនុស្សឡើយ។
មេរោគនេះងាយឆ្លងតែមិនឆ្លងដល់មនុស្សឡើយ។
ម៉េរោគនេះង៉ាយឆ្លុងតែ៉មិនឆ្លងដល់មន៉ុស្ស៍ឡើយ។
ម៉េរោ៉គនេ៉ះង៉ាយឆ្លុងតែិមីនឆ្លុងដល់មនិស្សឡើយ។

Lao:

ໄຂ້ຫວັດໃຫຍ່ແມ່ນ ພະຍາດຕິດຕໍ່ຂະໜາດສູງ ແຕ່ບໍ່ສາມາດຕິດຕໍ່ຫາຄົນໄດ້.
ໄຂ້ຫວັດໃຫຍ່ແມ່ນ ພະຍາດຕິດຕໍ່ຂະໜາດສູງ ແຕ່ບໍ່ສາມາດຕິດຕໍ່ຫາຄົນໄດ້.
ໄຂ້ຫວັ່ດໃຫຍ່ແມ່ນ ພ່ະຍ໋າດຕິດຕໍ່ຂະໜ່າດສູ່ງ ແຕ່ບໍ່ສຳມ່າດຕິດຕໍ່ຫ໋າຄົນໄດ້.
ໄຂ້ຫວັ໋ດໃຫຍ່ແມ່ນ ພ່ະຍ່າດຕຶດຕໍ່ຂືະໜ້າດສູງ ແຕ່ບໍ່ສາມືາດຕຶດຕໍ່ຫົາຄົນໄດ້.

Thai:

สนามแข่งม้าแร็นด์วิคถูกปิดและคาดว่าจะยังคงปิดอยู่ต่อไปอีกถึง 2 เดือน
สนามแข่งม้าแร็นด์วิคถูกปิดและคาดว่าจะยังคงปิดอยู่ต่อไปอีกถึง 2 เดือน
สนำมแข่งม้ำแร็นด์วิคถูกปิดและคำดวาจ่ะยังคงปิดอยู่ต่อไปอีกถึง 2 เดือน
สนาํมแข่งม่าแร็นด์วีคถู๋กปีดและค่าดว์า่จ์ะยั่งคงปีดอยู่ต่อไปอิกถึง 2 เดือน