Proceedings of the LREC 2020 – 4th Workshop on Computational Approaches to Code Switching, pages 65–70
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Towards an Efficient Code-Mixed Grapheme-to-Phoneme Conversion in an Agglutinative Language: A Case Study on To-Korean Transliteration

Won Ik Cho, Seok Min Kim, Nam Soo Kim
Department of Electrical and Computer Engineering and INMC, Seoul National University,
wicho@hi.snu.ac.kr, smkim@hi.snu.ac.kr, nkim@snu.ac.kr

Abstract

Code-mixed grapheme-to-phoneme (G2P) conversion is a crucial issue for modern speech recognition and synthesis task, but has been seldom investigated in sentence-level in literature. In this study, we construct a system that performs precise and efficient multi-stage code-mixed G2P conversion, for a less studied agglutinative language, Korean. The proposed system undertakes a sentence-level transliteration that is effective in the accurate processing of Korean text. We formulate the underlying philosophy that supports our approach and demonstrate how it fits with the contemporary document.

Keywords: code-mixed G2P, sentence-level transliteration, agglutinative language, open-source software

1. Introduction

Grapheme-to-phoneme (G2P) conversion is an essential process for speech recognition and synthesis. It converts textual information called *grapheme* into phonetic information called *phoneme*. The graphemes, represented by symbols that let the language users pronounce, are not real audio data, nor do not have a necessary correspondence with the genuine sound. For example, 'apple' sounds more like *æpl*, while 'America' sounds like *əmérikə*. This process implies that the character 'a' does not have a direct correspondence with the sound '*æ*' or '*ə*'; instead, the appropriate symbol to transcribe each pronunciation might have been 'a'. This is influenced by that the English alphabet is a segmental script, but other writing systems do not necessarily guarantee greater correspondence. For example, in the case of logograms such as Chinese characters, there is little relationship between the composition of the character (*bushu*) and the pronunciation of the symbol (Figure 1, top).

In a little different viewpoint notwithstanding, Hangul representation of Korean is a featural writing system (Daniels and Bright, 1996) in which each sub-characters of morpho-syllabic blocks corresponds to a phonetic property (Figure 1, bottom) (Kim-Renaud, 1997). For instance, in a syllable *khak* placed at the right end of the bottom of Figure 1, the three clock-wisely arranged characters *kh*, *a*, and *k*, which sound *khiukh* (among 19 candidates), *ah* (among 21 candidates), and *kiyek* (among 27 candidates), refers to the *first*, the *second* and the *third* sound of the given character, respectively (Cho et al., 2019). This is a unique feature of the Korean writing system, which distinguishes Hangul from Chinese characters that do not have a direct relationship with syllable pronunciations. Also, Hangul is more delicately decomposed compared to mora-level Japanese Kana.

Due to the above characteristics, the process of transforming grapheme in Korean to phoneme is widely performed by using the Korean alphabet itself (Jeon et al., 1998; Kim et al., 2002), that is, the Hangul sub-character *Jamo*, unlike cases such as Chinese *pinyin* that borrows the English alphabet (Figure 1, top). For this reason, even though the widely used Korean G2P sometimes uses English expressions (Cho, 2017), the full phoneme sequence is primarily written in Hangul Jamo, to reflect the Korean pronunciation system. This property, the grapheme and phoneme set sharing the same symbols, allows Korean G2P a phonological approach within the language itself.

Figure 1: Comparing the Chinese language written with Hanzi (along with *pinyin*, top) and the Korean language written with Hangul, the featural writing system (along with Yale romanization, bottom).

Currently, Korean G2P systems in use (Cho, 2017; Park, 2019) follow the pronunciation rules of the National Institute of Korean Language in principle, and we can confirm that the conventional modules perform well on a rule-based basis. However, in this study, we implement a preprocessing module for challenging code-mixed G2P, which regards co-existing Korean and non-Korean expressions (Shim, 1994), considering the case where the basis cannot be found in the monolingual rule. In specific, we deal with the English alphabet and Chinese characters, and mainly on the former[1], concerning that environment in which English is mixed with text often exist in modern scripts such as technical reports or scripts (Shim, 1994; Sitaram et al., 2019).

[1]Depending on the configuration and arrangement of Chinese characters, the duration of the syllable may change or a particular consonant may be inserted, but this is a task to handle in G2P after converting to a Hangul once and not a target here. Also, Japanese Kana is seldom used among Korean text.

Due to the human language being arbitrary, there are limitations in obtaining phoneme sequences using only the rules in some cases. Firstly, because code-mixing is not restricted only to two languages (ko-en), that English letters co-existing with Chinese characters and numbers are also observable. Second, various acronyms with non-deterministic pronunciation exist and are frequently utilized (e.g., word2vec, G2P), usually not spoken in a code-switched way. Finally, due to the agglutinative property of the Korean language, it is often vague to decide which phrase to transform within a sentence. Accordingly, we decided to fully utilize the information given by existing libraries and dictionaries to implement a sentence-level transliteration for Korean/English code-mixed G2P, taking into account the syntactic property of decomposed tokens. The contributions of this study and demonstration are as follows:

- Easily adjustable multi-stage system for a sentence-level code-mixed Korean G2P; detecting foreign expressions and replacing them with Hangul terms
- Suggesting morphological and phonological tricks that can handle the pronunciation of cumbersome non-Korean expressions

The system and code is to be publicly available[2].

2. Background

Sentence-level transliteration may seem simple, but it is involved in all phonetics, phonology, and morphology. In other words, at least the background in the Korean writing system, morphological analysis, Korean-English code-mixed writing is essential for implementing en-ko code-switching G2P (Kim et al., 2002). Phonetically, the Korean language is a language pronounced as a sequence of syllables, and the related phonemes locally correspond to graphemes represented by morpho-syllabic blocks (Kim-Renaud, 1997). The grapheme consists of a block as a single character, and is decomposed to sub-characters of first to third sound; CV(C). They are spoken straightforwardly in singleton cases, but when two or more characters are contiguous, the pronunciation differs from that of the single one (Jeon et al., 1998).
A code-mixed sentence, in this paper, is a Korean utterance (mainly written in text), where the syntax follows the Korean grammar, but some content phrases (non-functional expressions) are replaced by non-Korean terms, including English, Chinese and some numbers (Figure 2) (Shim, 1994). These expressions are often not promising in pronunciation for users of the same language, and acronyms are often confusing to resolve even when the source language is known (e.g., LREC as el-rec, or AAAI as triple-A-I). Therefore, for G2P, the biggest problem that code-mixed sentences bring is the difficulty of applying a rule for generating a phoneme sequence for speech processing, especially speech synthesis (Chandu et al., 2017). This, in turn, is directly related to the difficulty of transliteration (Sitaram et al., 2019).

[2]https://github.com/warnikchow/translit2k

G2P의 관점에서, code−mixed 文章이 가져오는 가장 큰 문제점은, 音聲 처리, 특히 音聲 합성에 文章을 活用하는 데에 있어, 기존의 음소 sequence를 생성하는 rule을 적용하기 어렵다는 것이다.

Figure 2: Given that the modern Korean writing system does not utilize Chinese characters, the sentence above is a code-mixed Korean sentence with Chinese characters (green), English words (blue), and numbers (yellow). The translation is: *"From the point of view of G2P, the biggest problem with code-mixed text is that it is difficult to apply the rules for generating existing phonetic sequences in the use of text for speech processing, especially for speech synthesis."*

There has been a lot of work on word-level transliteration process and its evaluation (Kang and Kim, 2000; Oh and Choi, 2002; Oh and Choi, 2005; Oh et al., 2006), but little on the sentence-level processings. Unlike in English, where each word consists of either source or target language that arbitrary word can be transliterated into the source language, In Korean code-mixed sentences, it is usual that the foreign expressions are augmented with the functional particles, in a truly code-mixed format in morphological level (Figure 2). It looks like a pidgin language, but is fully comprehensible by native readers since the symbols are distinguished. It is assumed that many industrial units are applying various heuristics to handle them, but we could not find an established academic approach for this issue. Built on the preceding discussions on code-mixed sentences and transliteration, we provide a detailed description of our resolution afterward.

3. Proposed Method

As the main contribution of this paper, we will implement a sentence-level code-mixed G2P that operates efficiently. For this, we took two methods into account.

(1) On training a transformation module that maps Korean/non-Korean code-mixed raw text directly to Korean phoneme sequence

(2) Multi-stage method of primarily changing non-Korean vocabulary to Korean pronunciation in code-mixed text and applying separate G2P module

The method of (1) is very suitable for utilizing neural network-based training and the implementation of end-to-end speech recognition/synthesis system, but usually, the number of Korean lexicons is significantly higher than the English vocabulary size. In other words, as long as Korean sentences are used for speech recognition or synthesis, a large amount of artificially made code-mixed sentences are required for reliable learning, of which the effectiveness is not guaranteed. In addition, it does not seem data-efficient in that Korean text dominant in the dataset may deter the enhancement of transliterating arbitrary foreign expressions. These issues can result in the degradation of G2P precision and the performance of recognition/synthesis.

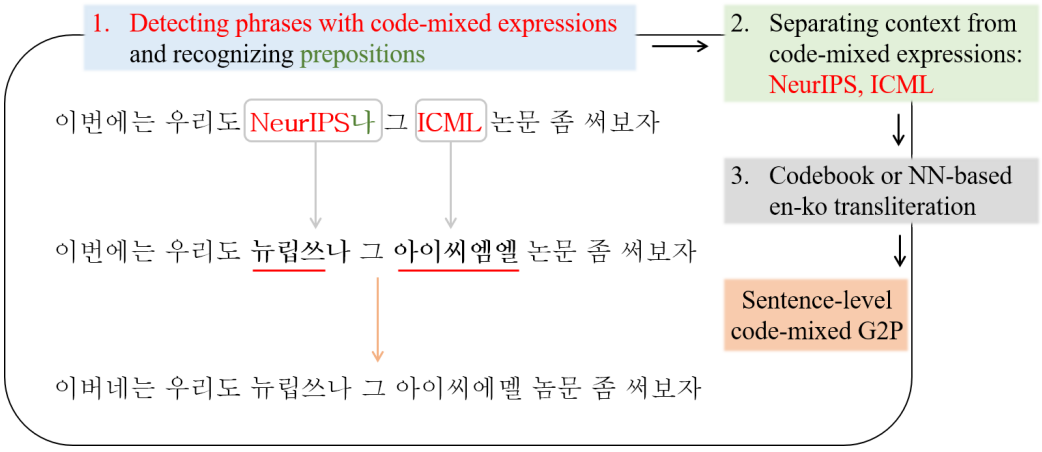

Figure 3: A brief diagram of the proposed code-mixed transliteration system. The translation is: *"Why don't we write a NeurIPS or ICML paper this year?"*, and the non-Korean terms *NeurIPS* and *ICML* are identified and transformed.

On the other hand, in (2), non-Korean expressions are transliterated into Korean primarily, and then rule-based precise Korean G2P is performed. For the latter part of the process, a well-used module already exists (Cho, 2017; Park, 2019), so we can concentrate on performing the former task, the transliteration to Korean. In this study, we adopt (2), mainly enhancing the transliteration process by detecting English and other non-Korean expressions (including Chinese characters and numbers) in code-mixed sentences and transforming them into Korean pronunciation. The specific procedure using the method of (2) is as Following (Figure 3).

1. **Detecting phrases with code-mixed expressions:** First of all, in the result of merely splitting a sentence into white space, detect an eojeol (Korean term for a whitespace-split word) containing an English or non-Korean (Chinese characters, numbers) expressions. In this process, Unicode information is exploited. The tokenization is done basically by a morphological analyzer, and each eojeol is considered as a chunk of morphemes.

2. **Separating context:** Subsequently, use the eojeols of interest as the target of transformation, except for the functional particles (if present). In this process, the outcome of the morphological analyzer above is adopted.

3. **Hybrid transliteration:** Finally, transliterate the detected English/non-Korean expressions into Korean pronunciation. It is viable to use a dictionary or train a neural network-based model, but we want to mix the two approaches. In more detail, one can collect a variety of English loanwords, and list them with the commonly used (lexicographical) Korean pronunciations, using it as a dictionary. After the primary rule-based transliteration, a trained transliteration system can be used for words that do not fall into the pre-defined categories. In this process, Chinese characters and numbers are all taken into account, along with the context that is present in the rest of eojeol. The tricks used here are:

- Trick 1. **On Chinese characters:** All Chinese characters are replaced with corresponding Hangul symbols, since such cases are Sino-Koreans which already have an established pronunciation. Here, a subsequent chunk of Chinese characters is tied and transformed together to reflect the possible change of pronunciation regarding word-initial rules. If the Chinese character and numbers/English alphabet come together, the Chinese characters are transformed first, followed by the transliteration of other parts.

- Trick 2. **On numbers:** For lone case, the pronunciation may follow the corresponding Chinese character as default, and if not alone, the tokens nearby are taken into account. If a number is placed between English words, consider using the result of transliteration of English words into Korean (e.g., 2 = two > *thu*, 4 = four > *pho*). Even when a number is between the English alphabet and Chinese/Korean at the same time, the pronunciation may follow English, as in the case of '*number 3 kka-ci* (till number 3)'. Otherwise, between Chinese characters, the number is read as in Trick 1. If the number between is followed by Korean Hangul, the cardinality, ordinality, or being Sino-Korean of the number is determined upon a convention, which might change the pronunciation. This follows the conventions of the Korean language, and can be modified based on the dictionary.

- Trick 3. **On acronyms:** Acronyms are easy to detect if written in capital letters, but people do not necessarily follow such the standard. Therefore, we added some tricks for the ones that are not in the dictionary. If they are all composed of consonants or have separate symbols between characters, each consonant is subsequently pronounced in Korean. However, if there is a corresponding English word, the dictionary output, or the result that is yielded by the trained system is used.

In the above process, methods such as a recurrent neural network (RNN) or Transformer (Vaswani et al., 2017) may be used for machine learning approach through training. The method using training means to make seq2seq (Sutskever et al., 2014) model with alphabet input and Hangul output using a parallel corpus of English and transliterated English. However, it is not necessary to take

a training-based approach to English words that are already in the dictionary. Thus, words in the codebook[3] produce a precise output in the form of look-up tables, and words not in the codebook are predicted by seq2seq models learned through parallel corpus (here it is the same as the codebook). This allows the model to learn pronunciations for words that are not in the dictionary, and possibly for acronyms, as many previous machine learning-based transliteration modules did (Karimi et al., 2011; Finch et al., 2016). Translating English into Korean first in this way and then applying rule-based G2P allows the modeling of the entire G2P to be more robust to Korean pronunciation rules.

We note here that though the codebook we adopt already incorporates a precise transformation of many words (about 37K), we need to train a system that can pronounce words that are not on the list. That is, we need to observe beyond the rules of how the arrangement of English consonants and vowels has determined Korean pronunciation. Once the Hangul characters are padded sub-character-level, or jamo-level, and compared with the English alphabet, the correspondence between the two is not consistent. Beyond the limitation of symbol representation, what makes this more challenging are 1) the different sound produced by the Korean consonants that come to the first and third sound, and 2) the sound change that takes place when the third sound meets the first sound of the next syllable. Moreover, 3) in English, one needs to observe the vowels where the consonant is located around, the vowel placement within the word, and what unique phonetic properties the various bigram/trigram characters have.

Therefore, in the implementation of a non-rule-based transliteration system, the seq2seq approach is carried out to character level in English and sub-character-level in Korean. Moreover, in characterizing Korean, the first sound and the third sound, that are similarly the consonant, can be represented distinctly. Using this, with about 37K pairs of English word-Korean pronunciation pairs, we trained the (attention-based) RNN encoder-decoder (Cho et al., 2014; Luong et al., 2015), under the consideration that the Transformer would be too large-scale for just a word-level transformation. The implementation detail is to be released along with the model and system.

4. Experiment

The concept of sentence-level code-mixed Korean G2P has been proposed in the previous section, and we aim to implement a fast and accurate code-mixed G2P that can be used for practical speech recognition/synthesis, that integrates other models in use. However, since standard transliteration studies have sought for character/word-level accuracy, mainly in word-level transformations, referring them might not be suitable for direct comparison with this work. Therefore, in this section, we will demonstrate the flexibility and utility of our approach with a concrete example.

[3] In this paper, we interchangeably utilize *codebook* and *dictionary*.

4.1. Implementation

For an efficient construction that divides and conquers the sub-modules, we leveraged various open-source libraries in our implementation. The sub-modules and corresponding libraries are as follows:

- **mixed_g2p:** Transforms a code-mixed sentence to the phoneme sequence. Consists of *sentranslit* and *KoG2P*/*g2pK*.
  - *KoG2P*[4]*:* An easily employable Python-based Korean G2P library that transforms the Korean text to alphabetical symbols that represent the Korean phonemes, based on a rulebook.
  - *g2pK*[5]*:* An up-to-date Python-based open-source G2P library for Korean, that transforms a Korean grapheme sequence to a Hangul syllable sequence that is more familiar with human reading.
- **sentranslit:** Performs sentence-level transliteration of code-mixed sentences. Consists of *align_particles*, *trans_eojeol* (eojeol-level transliteration), *trans_number*, *trans_hanja*, and *trans_latin*. Undertakes transliteration only if the string contains non-Hangul expressions.
  - *hgtk*[6]*:* A software that recognizes, decomposes, and reconstructs Hangul/Jamo sequence. Also detects if the string contains Chinese characters or the Latin alphabet.
- **trans_eojeol:** Controls the operation of *trans_number*, *trans_hanja*, and *trans_latin*, given the result of morphological analysis.
  - *MeCab*[7]*:* A statistic model-based Korean morphological analyzer that performs fast and accurate, which was first developed for the analysis of the Japanese language. Here, we utilize *python-mecab*[8] for convenience, which is an easily accessible wrapper.
- **trans_number:** Reads the numbers in Chinese style (Korean pronunciation), in English (en-ko transliteration), or in Korean (ordinal, cardinal, or Sino-Korean).
  - Bases on the characteristics of the context tokens, incorporating various exceptional cases.
- **trans_hanja:** Reads Chinese characters in Korean pronunciation, considering the word-initial rules.
  - *hanja*[9]*:* A library that translates Chinese characters into Korean syllables, also obeying word-initial rules. This module is utilized in two parts of the system; at the very first of the sentence analysis and again in eojeol-level, to complement the possible fail of Chinese character recognition.
- **trans_latin:** Performs en-ko transliteration, with rule and learning hybrid approach.

[4] https://github.com/scarletcho/KoG2P
[5] https://github.com/Kyubyong/g2pK
[6] https://github.com/bluedisk/hangul-toolkit
[7] https://bitbucket.org/eunjeon/mecab-ko-dic/src/master/
[8] https://github.com/jeongukjae/python-mecab
[9] https://github.com/suminb/hanja

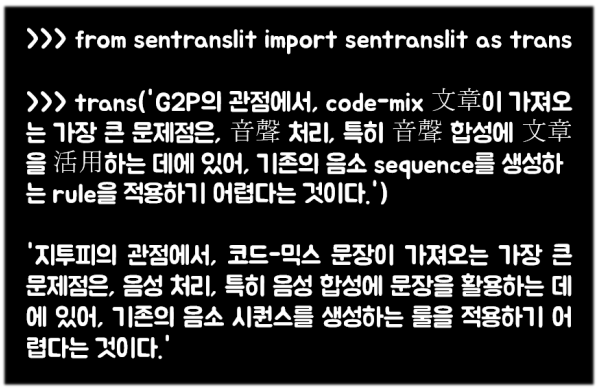

Figure 4: Demonstration for the sample sentence in Figure 2. Note that the code-mixed expressions in each eojeol are transliterated based on the scheme and tricks in Section 3.

– *transliteration*[10]*:* Our utilized dictionary comes from the pre-built dataset[11] of this library, where the results of learning-based en-ko transliteration was previously published. We train a new system based on this, and this module can be replaced with whatever transliteration module that shows sufficient performance.

4.2. Demonstration

Our demonstration with the sample sentence in Figure 2 is suggested in Figure 4. Since the G2P conversion is straightforwardly performed, we discuss here the sentence-level transliteration process.

There are mainly three points that show how our system works. First, regarding Chinese code-mixed expressions, some in sole words and others mixed with Korean functional particles, our module (*trans_eojeol*) detects the terms and translate them into Korean pronunciation via *trans_hanja*, with the help of *hanja* library. Next, similar is done for English expressions such as *code*, *mix*, *sequence* and *rule*, possibly utilizing *trans_latin*, where the dictionary and training are engaged in. Lastly, for a challenging term *G2P*, which may not be in the dictionary (and is at the first place decomposed by the morphological analyzer), the sub-modules above succeed to split them into *G*, 2, and *P*, transliterating each of them to Korean pronunciation *ci*, *thu*, and *phi*, given that *g* and *p* should be read as a single alphabet (due to being sole consonant) and also 2 is read in English concerning its surroundings. In this way, our module divides and conquers the challenging task and finally yields the desired output.

4.3. Discussion

Though our work is mainly on a code-mixed G2P, the recently released library *g2pK* partially shares some features with ours; various functions are inserted regarding the pronunciation of English terms and numbers in Korean sentences. We concentrate more on reading numbers and acronyms in a code-mixed context, trying to make a rule-learning hybrid approach for en-ko transliteration. On the other hand, in *g2pK*, such functions are implemented as a utility, while G2P rules are main and quite thoroughly investigated. We claim that both systems are not mutually exclusive, and rather might be complementary to each other.

Again, to be specific on the architecture, each of our sub-modules can be replaced with whatever the user wants as customization, without losing the additional flexibility of the user-generated dictionary. For instance, as suggested in *transliteration* library, one can define a new word list and accumulate the wanted result to it. Making up a look-up table can sometimes and inevitably be more efficient and accurate. Also, since MeCab was basically proposed for the analysis of the Japanese language, whose syntax a lot resembles Korean, one who wants to implement a similar module for Japanese code-mixed writings may benefit our system. The above factors support the scalability and generalizability of our approach.

5. Application

The code-mixed G2P implemented in this paper can be used for both research and industry. First of all, as emphasized, the application onto speech synthesis is very intuitive. Korean corpus has many sentences that consist only of Hangul characters, of course, but there may also be enough code-mixed expressions in modern text, and especially in chat dialogues, which is close to being synthesized. Therefore, if one can take advantage of this system well, it might be possible to promote plausible code-mixed pronunciation without regulating the generation of the script to only one kind of language. Without a doubt, this does not have a conflict with the option of not doing code-switching. That is, merely preserving the pronunciation of the source language is also recommended, if technically available.

The multi-stage approach we present can, of course, generate bottlenecks. However, it is expected to have significant advantages over end-to-end learning, in other words, using code-mixed text for training speech synthesis systems. For instance, the English language, once used with Korean notation, hardly reflects the phonetic traits shared with other Korean alphabets. This is primarily because the structure of CV(C) is not clear in the English writing system as in Hangul. Also, since agglutinative language usually displays functional particles after nouns or verbs, a corpus configuration with insufficient English words does not guarantee the performance of end-to-end architecture. It is also challenging to ensure that doing so yields transliterated pronunciations that we pronounce in real life, nor better than the transliteration modules that concentrate on word-level seq2seq. Therefore, we believe that it is practically advantageous to detect non-Korean expressions first and use hybrid transformation with some tricks.

The implementation of G2P for speech synthesis is a typical application, but besides, this algorithm can be exploited in sentence correction, corpus refinement, script construction/pronunciation guidelines, and translation service quality improvement. Also, this methodology is expected to apply not only to Chinese characters, English words, and numbers in Korean sentences, but also to sentences and code-mixed expressions in various agglutinative languages, especially the ones that require morphological analysis.

[10] https://github.com/muik/transliteration

[11] https://github.com/muik/transliteration/tree/master/data/source

6. Conclusion

In this paper, we constructed a stable and efficient g2p by presenting a hybrid transliteration method with rule and training for code-mixed Korean sentences. To this end, we detected words containing non-Korean expressions, separated a grammatical part of the word from the rest content via morphological analysis, and replaced the code-mixed expressions with transliterated ones.

In this process, by using a statistical model-based morphological analyzer with fairly high performance, we performed non-Korean expression detection that is suitable for colloquial context, with a less computational burden. Also, by separating the grammatical part from the content part in this process, the actual part that needs to be converted in the code-mixed sentence is detected so that the expressions that contain English/Chinese/number can be smoothly converted into Korean pronunciation.

Our subsequent studies aim to improve the accuracy of handling proper nouns in code-mixed text pre-processing by collecting more commercial expressions. Also, we plan to verify common pronunciation patterns through media/real-life examples, exploiting the neural network structure with external memory. As a result, research will be carried out to enable controllable to-Korean transliteration by allowing more user-specified stop-words to be reflected in the training of the systems and conversion process itself.

7. Acknowledgements

This research was supported by Projects for Research and Development of Police science and Technology under Center for Research and Development of Police science and Technology and Korean National Police Agency funded by the Ministry of Science, ICT and Future Planning (PAJ000001-2017-101). Also, this work was supported by the Technology Innovation Program (10076583, Development of free-running speech recognition technologies for embedded robot system) funded By the Ministry of Trade, Industry & Energy (MOTIE, Korea). Besides, we thank to the three anonymous reviewers for their helpful comments. After all, the authors appreciate all the contributors of the open source libraries which were essential for our project.

8. Bibliographical References

Chandu, K. R., Rallabandi, S. K., Sitaram, S., and Black, A. W. (2017). Speech synthesis for mixed-language navigation instructions. In *INTERSPEECH*, pages 57–61.

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Cho, W. I., Kim, S. M., and Kim, N. S. (2019). Investigating an effective character-level embedding in Korean sentence classification. *arXiv preprint arXiv:1905.13656*.

Cho, Y. (2017). Korean grapheme-to-phoneme analyzer (kog2p). `https://github.com/scarletcho/KoG2P`.

Daniels, P. T. and Bright, W. (1996). *The world's writing systems*. Oxford University Press on Demand.

Finch, A., Liu, L., Wang, X., and Sumita, E. (2016). Target-bidirectional neural models for machine transliteration. In *Proceedings of the sixth named entity workshop*, pages 78–82.

Jeon, J., Cha, S., Chung, M., Park, J., and Hwang, K. (1998). Automatic generation of Korean pronunciation variants by multistage applications of phonological rules. In *Fifth International Conference on Spoken Language Processing*.

Kang, I.-H. and Kim, G. (2000). English-to-Korean transliteration using multiple unbounded overlapping phoneme chunks. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*, pages 418–424. Association for Computational Linguistics.

Karimi, S., Scholer, F., and Turpin, A. (2011). Machine transliteration survey. *ACM Computing Surveys (CSUR)*, 43(3):1–46.

Kim, B., Lee, G. G., and Lee, J.-H. (2002). Morpheme-based grapheme to phoneme conversion using phonetic patterns and morphophonemic connectivity information. *ACM Transactions on Asian Language Information Processing (TALIP)*, 1(1):65–82.

Kim-Renaud, Y.-K. (1997). The phonological analysis reflected in the Korean writing system. *The Korean alphabet: its history and structure*, pages 161–192.

Luong, M.-T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.

Oh, J.-H. and Choi, K.-S. (2002). An English-Korean transliteration model using pronunciation and contextual rules. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pages 1–7. Association for Computational Linguistics.

Oh, J.-H. and Choi, K.-S. (2005). An ensemble of grapheme and phoneme for machine transliteration. In *International Conference on Natural Language Processing*, pages 450–461. Springer.

Oh, J., Choi, K., and Isahara, H. (2006). A comparison of different machine transliteration models. *Journal of Artificial Intelligence Research*, 27:119–151.

Park, K. (2019). g2pk. `https://github.com/Kyubyong/g2pk`.

Shim, R. J. (1994). Englishized Korean: Structure, status, and attitudes. *World Englishes*, 13(2):225–244.

Sitaram, S., Chandu, K. R., Rallabandi, S. K., and Black, A. W. (2019). A survey of code-switched speech and language processing. *arXiv preprint arXiv:1904.00784*.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.