Augmented words to improve a deep learning-based Indonesian syllabification

Suyanto Suyanto *, Ade Romadhony, Febryanti Sthevanie, Rezza Nafi Ismail

School of Computing, Telkom University, Bandung, Indonesia

Article Info

Keywords:
Creating acronyms
Indonesian orthographic syllabification
Flipping onsets
Transposing nuclei
Swapping consonant-graphemes

Abstract

Recent deep learning-based syllabification models generally give low error rates for high-resource languages with big datasets but sometimes produce high error rates for the low-resource ones. In this paper, two procedures: massive data augmentation and validation, are proposed to improve a deep learning-based syllabification, using a combination of bidirectional long short-term memory (BiLSTM), convolutional neural networks (CNN), and conditional random fields (CRF) for a low-resource Indonesian language. The massive data augmentation comprises four methods: transposing nuclei, swapping consonant-graphemes, flipping onsets, and creating acronyms. Meanwhile, the validation is implemented using a phonotactic-based scheme. A preliminary investigation on 50k Indonesian words informs that those augmentation methods significantly enlarge the dataset size by 12.8M valid words based on the phonotactic rules. An examination is then performed using 5-fold cross-validation. It reports that the augmentation methods significantly improve the BiLSTM-CNN-CRF model for 50k formal words and 100k named-entities datasets. A detailed investigation informs that augmenting the training set can reduce the word error rate (WER) coming from the long formal words and named entities.

1. Introduction

Syllabification is a process of defining syllable boundaries within a word to extract its syllables. It is used as a basis in a wide range of studies and various linguistics-based tasks. It’s commonly utilized in synthesizing [1] and recognizing speech [2, 3], converting text-to-phoneme [4], estimating a speaking rate [5], scoring the speaking proficiency [6], classifying the speaker’s emotion [7], assigning lexical stress [8], and analyzing metric [9], among other things.

As explained in [10], a grapheme is the writing system unit. Meanwhile, a phoneme is the smallest unit of speech that distinguish one word from another. For example, in Indonesian language, the grapheme (b) is always pronounced as the phoneme /b/ in a word “baca” (read) distinguishes that word from “kaca” (glass) that consists of a phoneme /k/. In Indonesian language, as one of the simplest language in the world, most graphemes correspond to only one phoneme. In [11], Suyanto et al. (2016, p. 430) show stated an observation on 50K Indonesian words shows that 15 out of 26 graphemes (57%) are pronounced as certain phonemes (no ambiguity) and the rest are phonemicized as several possible phonemes. For example, a grapheme (b) is always pronounced as the phoneme /b/ wherever its position in any word, such as “baca” (read), “abah” (father), and “adab” (courtesy). In contrast, grapheme (e) is probably phonemicized as phonemes /e/, /o/, /ei/, /e + i/, or /o + i/, such as in the word “kakek” (grand father), “emas” (gold), “survei” (survey), “beol” (defecate), and “seikat” (bunch). Grapheme (n) may be pronounced as phonemes /n/, /ni/, or /uj/, such as in the word “mana” (where), “abang” (older brother), and “puncak” (peak).

The syllabification can be performed to either graphemes or phonemes. Graphemic syllabification is done directly to the sequence of graphemes in a word, for example “langit” (sky) is syllabified to “la.ngit”. Meanwhile, phonemic syllabification is applied to the pronunciation of the word as a sequence of phonemes, for example /tajit/ is syllabified to /ta.jit/. In [12], Suyanto (2019, p. 1031) stated that compared to phonemic syllabification, graphemic syllabification generally gives lower performance but has a simpler implementation and...
gives more versatility. Hence, graphemic syllabification is preferable, especially when handling unseen terms and named-entities with many exceptions and uncertainties.

Syllabification is generally developed either with data-driven approach or rule-based approach. Data-driven approach typically gives better results and easier to implement, therefore it is more favoured than rule-based approach [13]. For instance, syllabification on Romanian using a simple Naïve Bayes results in a low syllable error rate (SER) of 12.90% [14]. Other examples of data-driven approach models use conditional random fields [15], decision tree, random forest, support vector machine [14], unsupervised model [16], hidden Markov model [17], dropped-and-matched model [18], syllabification by analogy [13], and context-free grammars [19].

In recent years, neural language models generally give great results, such as a BiLSTM-CNN-CRF-based model that performs well on a variety of languages [20], especially high-resource languages with large datasets. However, this model may obtain a low performance for some low-resource languages due to the small dataset. Therefore, rule-based and conventional machine learning techniques are commonly developed for low-resource languages. For instance, a rule-based model called dropped-and-matched syllabification obtains a small word error rate (WER) of 2.61% for the Malay language [18]. Another example is an n-gram model that offers a lower computational cost with more straightforward implementation while still give competitive results [21]. Syllabification applied to phonemic sequences in German gives a low WER of 0.15% [22]. It can be adapted to another language since no particular knowledge of the language is required. The main problem with n-gram models is that it produces many out-of-vocabulary (OOV) n-grams in a small dataset. In some simple languages, such as Indonesian, several data augmentation methods can be applied to solve this problem. For instance, a model named combination of flipping-onsets with standard-trigram and augmented-bigram syllabification (CFTABS) incorporate three augmentation techniques of flipping onsets, transposing nuclei, and swapping consonant-graphemes is developed in [23]. CFTABS produces a much lower SER than the original n-gram model with no augmentation. Nevertheless, its performance is worse than the BiLSTM-based model.

In this paper, another augmentation technique is introduced by creating acronyms between pairs of words. It is combined with the previous three augmentation methods to develop the BiLSTM-CNN-CRF-based Indonesian syllabification model. In addition, the augmented dataset is validated based on Indonesian phonotactic rules.

2. Research method

The proposed model is illustrated in Fig. 1. It contains four processes. First, the given dataset of formal words and their syllabifications is split to generate five new datasets (for training and testing) based on the 5-fold cross-validation scheme. Next, each training set is augmented with four techniques: swapping consonant-graphemes, flipping onsets, transposing nuclei, and creating acronyms. The augmented data is then validated based on phonotactic rules. Finally, each dataset is used to develop three models of BiLSTM-CNN-CRF using the original, augmented, and original + augmented training set, respectively.

2.1. Massive data augmentation

There are onset consonant-graphemes that can be substituted depending on its phoneme classification based on [10] as most Indonesian graphemes have only one corresponding phoneme. Each of the graphemes and their substitute is mapped into those phoneme classifications. The mapping is made possible by the close relationship between each corresponding phoneme pair [10]. Swapping consonant-graphemes is done by substituting graphemes in a word to generate several new words.

Fig. 1. Proposed data augmentation and the training of BiLSTM-CNN-CRF-based model.

The grapheme (b) and (p) are classified into plosive-bilabial. Swapping (b) with (p) and vice versa within a word generally results in several other formal words. For example, “ba.dan” (body) is swapped to be “pa.dan” (equivalent) and “pa.yung” (umbrella) is swapped to be “ba.yung” (machete).

Both (d) and (t) fall into plosive-dental class. Swapping (d) with (t) and vice versa in a word mostly produces several other formal words. For instance, “de.kan” (dean) is swapped to be “te.kan” (press) and “tang.kal” (block) is swapped to be “dang.kal” (shallow).

The graphemes (g) and (k) are categorized into the same group of plosive-velar. Therefore, swapping (g) with (k) and vice versa typically results in several other formal words. For example, “ga.wat” (precarious) is swapped as “ka.wat” (wire) and “ke.rah” (collar) is swapped as “ge.rah” (sultry).

Both graphemes (c) and (j) are classified as affricative-palatal. In general, swapping (c) with (j) and vice versa generates more formal words. For instance, “can.da” (joke) is swapped as “jan.da” (widow) and “je.bol” (broken-down) is swapped as “ce.bol” (midget).

The grapheme (v) is always articulated as /l/ since there is no phoneme /v/ in Indonesian as explained in [10]. Hence, the grapheme (v) is possible to be swapped with the grapheme (l), such as the word “fi.si” (fission) is swapped as “vi.si” (vision) and “vo.li” (volley) is swapped as “fo.li” (thin metal).

Both graphemes (s) and (z) are categorized as fricative-dental. Substitution between (s) and (z) results in some new formal words. For instance, “sa.lim” (healthy) is swapped as “sa.lim” (crue) and “sa.man” (era) is swapped to be “sa.man” (indictment).

Lastly, the grapheme (l) and (r) both fall into thrill/lateral-dental class. Substitution between (l) and (r) produces some new formal words. For example, “lu.wan” (opponent) is swapped as “ru.wan” (prone) and “ru.sa” (deer) is swapped to be “lu.sa” (the day after tomorrow).

Substituting the onset graphemes sometimes can create informal words. For instance, swapping (d) with (t) in “de.pan” (front) results in “te.pan” (OOV), swapping (k) with (g) turns the word “ka.pal” (ship) into “ga.pal” (OOV), and so on.

Flipping onsets is done by swapping the onset grapheme of the first syllable with that of the second one in a word. This method requires

Original dataset of formal words and their syllabifications

Generating five datasets based on the 5-FCV scheme

Data augmentation using four procedures

Validation of augmented data using phonotactic rules

Training of BiLSTM-CNN-CRF-based syllabification models

Trained syllabification model

S. Suyanto, A. Romadhony, F. Sthevanie et al. Heliyon 7 (2021) e08115
the original word to have at least two syllables and each syllable having an onset. For example, “ru.mah” (house) can be flipped into “mu.rah” (cheap) but “sa.ung” (hut) cannot be flipped because the second syllable does not have an onset. This method can produce new formal words like the former example, but in some cases, it may produce informal words such as “kan.dang” (cage) that is flipped to “dan.kung” (OOV).

Transposing nuclei works the same way as flipping onsets but instead it swaps the vowel of the first syllable with the second syllable. For instance, transposing the word “pu.lang” (go home) results in a new formal word “pa.tung” (trench). It may also generate informal words like when transposing “per.gl” (leave) into “pir.ge” (OOV).

Creating acronyms takes two words and generates a new word by combining two syllables, one from each word. The number of new words generated depends on the number of syllables from each word. For example, given two words, “mo.bi.le.rah” (red car), new syllables are created beforehand from each syllable of both words in two ways. First, for an open syllable (syllable ended with a vowel), onset grapheme from the next syllable (if any) is added to the end of the syllable. Second, for a closed syllable (syllable that ends with a consonant), all ending graphemes (codas) are removed to create an open syllable. From the two words, four new syllables are generated: (mo) (bi) (e) (rah). Next, each original and augmented syllable from the first word is paired with each original and augmented syllable from the second word. For example, (mo) and (rah) are combined into “mo.rah”, (bi) and (me) are combined into “bi.me”. and so on. In total, 16 new words are generated by combining the two words.

The four augmentation methods are combined to create new words. The augmentation methods are applied in the following order: 1) swapping consonant-graphemes, 2) flipping onsets, 3) transposing nuclei, and 4) creating acronyms. For example, the word “bi.ru” (blue) is first swapped to generate “bi.lu” (OOV), “pi.ru” (OOV), and “pi.lu” (sad). Next, the original and swapped words are flipped which creates “riibu” (thousand), “li.bi” (OOV), “ri.pu” (OOV), and “li.pu” (gloomy). Then, the original, swapped, and flipped words are transposed that results in “pu.ri” (castle), “ru.bi” (ruby), “lu.pi” (board), and five other OOV words: “bu.ri”, “bu.li”, “pu.li”, “lu.bi”, and “ru.pi”. From the first three augmentation methods, 15 new words are generated, where nine of them are OOV words. Note that some OOV words may be a sub-word from other formal words, such as the earlier OOV “li.bi” is a sub-word from the word “li.bu.ran” (holiday). The last method, creating acronyms, is only applied to the original word because the amount of words that can be generated from just a single pair of words. The original word is paired with every other word in the original dataset to form their acronyms. The augmented words from creating acronyms are then combined with the swapped, flipped, and transposed words.

The four augmentation techniques generally produce numerous illegal syllables and words for English and many other languages. However, Indonesian is one of the simpler languages where most of its graphemes correspond to only one phoneme. Hence, the augmentation produces more valid syllables to generate new words than the invalid ones. Swapping consonant-graphemes generates new words by swapping consonants from the original word with others that falls in the same category in terms of articulations. For example, swapping the original word “ge.las” (glass) results in three new words: “ge.ras” (OOV), “ke.las” (class), and “ke.ras” (hard) without moving any syllable boundaries as the grapheme (g) and (k) are articulated as the plosive-velar; whereas (l) and (r) are articulated as the trill/lateral-dental. Next, flipping those original and three swapped words produces four new words: “le.gas” (OOV), “re.gas” (cut), “le.kas” (quick), and “re.kas” (OOV). Finally, transposing the original, swapped, and flipped words generates eight new words: “ga.les” (OOV), “ga.res” (OOV), “ka.les” (OOV), “ka.res” (OOV), “la.ges” (OOV), “ra.ges” (OOV), “la.kes” (OOV), and “ra.kes” (OOV). In total, 15 new valid words are generated from the augmentation of the original word “ge.las” (glass) using the first three augmentation techniques.

The last augmentation technique, creating acronyms, is separated from the previous three techniques. It creates new words by combining syllables from two separate words. For example, the earlier word, “ge.las” (glass) is paired with another word “ba.ru” (new). Creating acronyms from these two words results in 12 acronyms: “ge.ba”, “ge.ru”, “ge.bar”, “las.ba”, “las.ru”, “las.bar”, “gel.ba”, “gel.ru”, “gel.bar”, “la.ba”, “la.ru”, and “la.bar”. Each word in the dataset is paired with every other word in the dataset beside itself. Suppose the dataset contains $n$ words, there are $n \times (n - 1)$ word combinations that can be created. Every word combination can produces many acronyms, depending on the number of syllables in each word. The dataset used in this research, which contains 50k formal words, can generate millions of new words from creating acronyms alone, hence why this technique is separated from the previous three. The resulting augmented words from the previous techniques are combined with the generated acronyms which then are validated with phonotactic rules. Additionally, Indonesian has 18 prefixes [10]. Interestingly, swapping several consonant-graphemes from the prefixes in derivative words may happen in other valid prefixes. Although, it can produce many invalid prefixes as well. Based on initial observation, swapping consonant-graphemes, flipping onsets, and transposing-nuclei applied on 50k formal Indonesian words produces up to 1,925,690 new words. Meanwhile, creating acronyms generates up to 11,247,809 new words. Combining both of them results in 13,060,941 unique words. The validation procedure reduces the augmentation words relatively by 1.52% to 12,862,207 valid augmented words.

Moreover, the four data augmentation methods can also be applied to named-entities. For example, flipping onsets (ky) and (t) on the city name “Kyo.to” produces a new city name “To.kyo”. Swapping a consonant-grapheme (d) on the person name “Do.ny” with a similar grapheme (t) creates a new person name “To.ny”. Transposing nuclei (i) and (o) on the person name “Di.no” creates a new person name “Do.nl”. Creating an acronym from two person names “A.ri” and “Tamam” comes to a new person name “Ar.tam”. Combining four methods on the dataset of named-entities also significantly enlarges its size like in the formal words.

2.2. Validation of the augmented words

Syllables of the augmented words generated from the four augmentation methods may contain illegal grapheme sequences. Each augmented word is checked with a list of illegal grapheme sequences to validate the augmentation results. If a word contains one or more illegal grapheme sequences, it is discarded. The illegal grapheme sequences list is made according to Indonesian phonotactic rules described in [10].

2.3. BiLSTM-CNN-CRF model

The BiLSTM-CNN-CRF model in [20] generally gives leading-edge performances on numerous datasets with varying languages. The model is considered language-agnostic; therefore, it is adapted to tackle Indonesian syllabification in this research.

The model receives a sequence of graphemes as input. The length of all grapheme sequences is made uniform by padding them based on the length of the most extended grapheme sequence, denoted as $n$. Thus, each input has a fixed length of $n$ and is defined as $g = (g_{0}, g_{1}, \ldots, g_{n-2}, g_{n-1})$. (1)

The grapheme sequence $g$ is converted into a $d \times n$ embedding vector $x$ defined as $x = (x_{0}, x_{1}, \ldots, x_{n-2}, x_{n-1})$ (2)

Then, the input vector $x$ is processed by both the BiLSTM and CNN. The BiLSTM consists of a forward and backward LSTM. The output of the backward LSTM is concatenated with the forward LSTM that produces the output $h$, which has $2l \times n$ dimension where $l$ is an optimized
Table 1. Illegal grapheme sequences and their frequencies appear in the augmented words.

| No. | Sequence | Freq. | Percent | No. | Sequence | Freq. | Percent |
|-----|----------|-------|---------|-----|----------|-------|---------|
| 1   | nk       | 48,092| 0.37    | 21  | pz       | 2,404| 0.02    |
| 2   | nd       | 14,186| 0.11    | 22  | zl       | 1,074| 0.01    |
| 3   | nd       | 14,186| 0.11    | 23  | ld       | 723  | 0.01    |
| 4   | ch       | 13,475| 0.10    | 24  | lb       | 272  | 0.00    |
| 5   | mn       | 13,462| 0.10    | 25  | bd       | 168  | 0.00    |
| 6   | gn       | 13,456| 0.10    | 26  | bt       | 168  | 0.00    |
| 7   | kz       | 8,655 | 0.07    | 27  | pd       | 168  | 0.00    |
| 8   | mz       | 8,402 | 0.06    | 28  | zf       | 132  | 0.00    |
| 9   | pn       | 6,835 | 0.05    | 29  | lz       | 128  | 0.00    |
| 10  | hn       | 6,811 | 0.05    | 30  | rz       | 128  | 0.00    |
| 11  | nh       | 6,788 | 0.05    | 31  | zh       | 124  | 0.00    |
| 12  | ft       | 6,781 | 0.05    | 32  | bs       | 58   | 0.00    |
| 13  | rh       | 6,780 | 0.05    | 33  | ds       | 52   | 0.00    |
| 14  | sb       | 4,629 | 0.04    | 34  | zn       | 34   | 0.00    |
| 15  | zh       | 4,629 | 0.04    | 35  | zm       | 34   | 0.00    |
| 16  | zp       | 4,629 | 0.04    | 36  | mp       | 29   | 0.00    |
| 17  | zy       | 3,209 | 0.02    | 37  | jh       | 20   | 0.00    |
| 18  | zk       | 3,159 | 0.02    | 38  | lg       | 17   | 0.00    |
| 19  | bz       | 2,404 | 0.02    | 39  | dm       | 16   | 0.00    |
| 20  | bz       | 2,404 | 0.02    | 40  | Others   | 13   | 0.00    |
|     | Total    | 198,734| 1.52    |

hyperparameter. The CNN has a 1-dimensional convolutional filter with width \( w \) that processes the input. The output \( c \) of the CNN has a dimension of \( f \times n \) where \( f \) denotes how many filters are used. Both \( h \) and \( c \) are concatenated that results in a \((2l + f) \times n \) combined vector \( o \) formulated as

\[
o = [h + c].
\]

The output label classifies each grapheme as either a syllable boundary or not. Therefore, the output vector \( o \) is reduced to be a \( 2 \times n \) vector by applying a time-distributed fully connected layer, which results in

\[
y = (y_0, y_1, ..., y_{n-2}, y_{n-1}).
\]

A linear-chain CRF is used as the classifier to model the output probability. Taking into account the previous and the subsequent output tags, the conditional probability \( p(y|o) \) is modeled as

\[
p(y|o) \approx \frac{1}{Z(o)} \prod_{n=1}^{L} \exp \left\{ \sum_{k=1}^{K} \theta_k f_k(y_i, y_{i-1}, o_i) \right\},
\]

where \( \theta \) is a trained parameter in proportion to the value of \( f \) and \( Z(o) \) is the normalizing function formulated as

\[
Z(o) = \sum_y \prod_{n=1}^{L} \exp \left\{ \sum_{k=1}^{K} \theta_k f_k(y_i, y_{i-1}, o_i) \right\}
\]

3. Results and discussion

First, the investigation is performed on the massive augmentation method using the dataset of formal words. The four augmentation methods enlarge the dataset size of 50k into 13M (13,060,941) unique words. Next, the validation scheme produces 198,734 words containing illegal grapheme sequence, as illustrated in Table 1, reducing the augmentation words relatively by 1.52% to 12,862,207. As shown in Table 1, 49 most common illegal grapheme sequences appear in the augmentation results. The most frequent illegal grapheme sequence is \( nk \) that appear up to 48,092 (0.37%), which is generated based on swapping consonant-grapheme \( g \), which is in words containing a legal grapheme sequence \( ng \), into a similar consonant-grapheme \( k \).

However, the augmentation methods massively enlarge the dataset by up to 257 times, where some examples of the valid augmented words are illustrated in Table 2. Next, those 12.8M valid augmented words are used to train a BiLSTM-CNN-CRF syllabification model. The parameters for the model used in this research are as follows: the size of the recurrent neural network is 300, the number of CNN layers is 2, the amount of filters \( f \) is 200, the size of the filter is 3, the max pool size is 2, and the embedding size \( d \) is 300.

The evaluation of the BiLSTM-CNN-CRF models is based on the 5-fold cross-validation scheme on the 50k Indonesian formal words from Kamus Besar Bahasa Indonesia (The Great Dictionary of the Indonesian Language) and 100k named-entities. The results based on the models trained using the valid augmented dataset are compared to the model with the original dataset. Finally, the combination of the original and valid augmented dataset is evaluated.

3.1. Investigation on formal words

For the dataset of formal words, the results in Figs. 2 and 3 shows that, on average, the BiLSTM-CNN-CRF model trained with augmented dataset gives SER and WER of 0.42% and 2.86% respectively, which are lower than the model trained using the original dataset (0.44% and 3.74%). Although, the augmented model is more unstable as there is an error rate spike on the fourth fold. All these results indicate that the augmentation and validation methods create many new valid words with similar patterns (or even the same) as the formal words so that the augmented dataset can adapt to the original one very well.

Meanwhile, combining the original and the valid augmentation datasets gives more stability and even much lower SER and WER of 0.36% and 2.50%, respectively. This achievement shows that the proposed data augmentation and phonotactic rules-based validation can boost the performance of the BiLSTM-CNN-CRF-based syllabification model for formal words. Incorporating valid augmented words reduces the SER and WER relatively by up to 18.92% and 33.08%, respectively.

A more detailed investigation shows that almost all WERs produced by the original dataset come from the words with one-syllable errors (around 47%) and two-syllable errors (around 50%). Very few WER comes from words with three or more syllable errors (around 3%). The combination of original and valid augmentation datasets gives significantly lower WER for words with one syllable error (around 6%), with most of WER comes from the words with two-syllable errors (around 91%), as illustrated in Fig. 4. These results aligned with the fact that the SER relative reduction (18%) is smaller than the WER relative reduction (33%).

Moreover, further investigation is performed to see the error contribution of the length of the words. First, the words in the datasets are classified into three categories: short (less than six characters), medium
| Number | Original word | Valid augmented words |
|--------|---------------|-----------------------|
| 1      | ba.ca (read)  | ca.ba (haphazard), ba.ja (steel), pa.ja (children), ... |
| 2      | ba.pa (father)| pa.pa (poor or father), ba.ba (call for men), ... |
| 3      | ba.ru (new)   | ra.ru (wednesday), pa.ru (lung), pa.ru (hammer), ... |
| 4      | ba.ta (stone) | ta.ta (taboo), pa.da (solid), bu.ta (blind), ... |
| 5      | be.da (different) | pe.da (a kind of fish living in the sea), be.ta (I am), ... |
| 6      | ba.tu (help)  | pan.du (guide), bu.da (mother), ... |
| 7      | can.du (opium)| jan.du (frog mouth owl), jun.ta (government council), ... |
| 8      | da.du (dice)  | pa.du (solid), du.da (widower), du.ta (ambassador), ... |
| 9      | da.sar (basic)| sa.dar (aware), pa.sar (market), ... |
| 10     | ka.mi (we)   | ma.ki (curse), mi.ka (mica), ... |
| 11     | lan.tai (floor)| ran.tai (chain), lan.dai (sloping), ... |
| 12     | pa.ku (nail)  | ba.ku (formal), pa.gu (ceiling), ba.gu (net), ... |
| 13     | pan.dai (smart)| pan.tai (beach), bu.nai (slaughter), ... |
| 14     | pang.ku (lap) | bang.ku (seat), pung.gu (part), ... |
| 15     | ra.mah (friendly)| ma.ra (mad), ra.tam (ra.mah ra.mah or courteous)... |
| 16     | ru.mah (house)| mu.ra (cheap), ru.sun (ru.mah su.sun or flat)... |
| 17     | sa.kit (sick)| si.kat (brush), si.gi (si.kat gi.gi or tooth brush), ... |

Fig. 2. SERs produced by three BiLSTM-CNN-CRF models trained using original, valid augmentation, and original + valid augmentation using the training sets of 40k formal words.

Fig. 3. WERs produced by three BiLSTM-CNN-CRF models trained using original, valid augmentation, and original + valid augmentation using the training sets of 40k formal words.
Fig. 4. Contribution to WER based on the number of syllable errors for the dataset of formal words.

Fig. 5. Contribution to WER based on the length of words for the dataset of formal words.

(six to ten characters), and long (more than ten characters). The resulting WER from the original dataset contains 5% from short words, 65% from medium words, and 30% from long words. Meanwhile, the WER from the combined dataset consists of 10% from short words, 77% from medium words, and 13% from long words, as illustrated in Fig. 5. Interestingly, the combined dataset has a much lower ratio of long words in its WER than the original dataset, with a relative reduction of 57%.

A careful observation informs that the combined dataset can correctly syllabify many long derivative words with the prefix “ber”, “meng”, “peng”, and “ter” where the original dataset fails to do so. Table 3 illustrates several examples of the syllable sequences resulted by the BiLSTM-CNN-CRF-based model trained using the original dataset and the combined original and valid augmented dataset. This result shows that the four augmentation methods are able to enrich the training set to more accurately syllabify longer words. For instance, swapping the consonants in the word “la.ja” (galangal) produces a new word “ra.ja” (king). The derivative word “beraja-raja” (kings) is syllabified correctly by the combined dataset into “be.ra.ja-ra.ja”, but the original dataset fails to do so. Flipping onsets transform the word “me.rah” (mad) into “ra.mah” (friendly). The derivative word “beramah-ramahan” (get along) is failed to correctly syllabify by the original dataset but the combined dataset is able to correctly syllabify it into “be.ra.mah-ra.ma.han”. Swapping the consonants in the word “lan.tai” (floor) results in “ran.tai” (chain). The derivative word “berantai-rantai” (connected) is syllabified correctly by the combined dataset into “be.ran.tai-ran.tai” but the original dataset fails to do so.

3.2. Investigation on named-entities

Next, an investigation is carried out using the dataset of named-entities. The experimental results in Figs. 6 and 7 informs that the BiLSTM-CNN-CRF model trained with augmented dataset gives mean SER and WER of 0.67% and 5.59%, respectively, which is lower than using the original dataset that produces 0.70% and 6.63%. This result also implies that the augmentation and validation methods create many new valid words with similar patterns (or even the same) as the named-entities. Hence, the augmented dataset can adapt to the original one. Although, it is not significantly as in the case of formal words.

Moreover, like in the formal words, the combination of original and valid augmented named-entities also obtains lower SER and WER: 0.62% and 5.01%, respectively. This result indicates that the proposed data augmentation and phonotactic rules-based validation can also improve the performance of the BiLSTM-CNN-CRF-based syllabification model for the dataset of names-entities. The valid augmented words reduce the mean SER and WER by up to 12.40% and 24.42%, respectively. These relative reductions are slightly lower than that given by the formal words. Therefore, an advanced validation scheme to filter the augmented names-entities can be created.

A similar pattern can be found here, like in the formal words, when investigating WER contributions in the syllabification results of both the original and combined datasets. Most WER from the original dataset comes from the words with one syllable error (around 39%) and two-syllable errors (around 55%). Only a small WER comes from words with three or more syllable errors (around 6%). The combined dataset gives a lower WER for words with one syllable error (around 22%) and a slight increase in WER for words with three or more syllable errors (around 8%), as shown in Fig. 8. Again, these results explain the lower SER relative reduction (11%) than the WER (24%).

The WERs produced from the original dataset consist of 24% from short words, 66% from medium words, and 10% from long words. For the combined dataset, the produced WERs consist of 28% from short words, 65% from medium words, and 7% from long words, as depicted in Fig. 9. Like in the formal words, the combined dataset has a lower ratio of long words in its WER than the original dataset, albeit
Table 3. Examples of Indonesian formal words that are wrongly syllabified by the BiLSTM-CNN-CRF-based model trained using the original dataset but correctly by the one trained using the combined original and valid augmented dataset.

| No. | Word         | Syllable sequence resulted using the formal words dataset | Original | Original + Valid Augmented |
|-----|--------------|---------------------------------------------------------|----------|----------------------------|
| 1   | beraja-raja  | be.ra.ja-ra.ja                                         |          |                            |
| 2   | beramah-ramahan | be.ra.mah-ra.ma.han                                   |          |                            |
| 3   | berandai-randai | be.an.dai-run.dai                                     |          |                            |
| 4   | berangsanang  | be.ang.sa.ngan                                         |          |                            |
| 5   | berantai-rantai | be.an.tai-run.tai                                     |          |                            |
| 6   | berawa-rawa   | be.a.wa-ra.wa                                          |          |                            |
| 7   | berempat-empat | be.rempat-empat                                        |          |                            |
| 8   | beresah-resah | be.re.sah-re.sah                                       |          |                            |
| 9   | berinkarnasi  | be.in.kar.na.st                                        |          |                            |
| 10  | berkeberatan  | be.ke.be ra.tan                                        |          |                            |
| 11  | berlonggoklonggok | be.long.go.klong.gok                               |          |                            |
| 12  | berperistiwa  | be.pe.ris.ti.wa                                        |          |                            |
| 13  | berstruktur   | be.struk.tur                                            |          |                            |
| 14  | beruntun-runun | be.run.tun-run.tun                                     |          |                            |
| 15  | gastroenterologi | gas.tro.en.te.ro.lo.gi                        |          |                            |
| 16  | isokeraunik   | i.so.ke.ru.ni.k                                        |          |                            |
| 17  | jailangkung   | jai.lang.kung                                           |          |                            |
| 18  | karborundum   | kar.bo.run.dum                                         |          |                            |
| 19  | kasregister   | ka.re.gis.ter                                           |          |                            |
| 20  | keberengsekan | ke.be.reng.se.kan                                       |          |                            |

Fig. 6. SERs produced by three BiLSTM-CNN-CRF models trained using original, valid augmentation, and original + valid augmentation using the training sets of 100k named-entities.

Fig. 7. WERs produced by three BiLSTM-CNN-CRF models trained using original, valid augmentation, and original + valid augmentation using the training sets of 100k named-entities.
not by much. Since the named-entities dataset does not contain derivative words, the type of long words correctly syllabified by the combined dataset is more diverse and does not follow a specific pattern.

However, a further investigation shows that the four augmentation methods also enrich the training set to recognize longer words. For example, two named-entities “asriningtyas” and “tan.jungarum” is accurately syllabified as “as.ri ning.ty.as” and “tan.jung.a.run” by the combined dataset where the original dataset unable to do so. Swapping grapheme in the named-entity from the training set “as.li” and “dy.as” produces “as.ri” and “ty.as”, which are sub-words of “as.ri ning.ty.as”. Transposing nuclei in the word from the training set “tan.jang” produces “tan.jung”, which is a sub-word of “tan.jung.a.run”. Table 4 illustrates some examples of syllable sequences resulted by the BiLSTM-CNN-CRF-based model trained using the original dataset and the combined original and valid augmented dataset of named-entities.

4. Conclusion

The proposed massive data augmentation and phonotactic-based validation significantly enlarge the dataset up to 257 times (from 50k to 12.8M). Leveraging both procedures can enhance the BiLSTM-CNN-CRF-based syllabification, which is shown by reducing the averaged SER and WER relatively by up to 18.92% and 33.08% for formal words; and 12.40% and 24.42% for named-entities. As future work, an advanced validation procedure to filter the augmented words and named-entities will be designed to increase the performance.

Declarations

Author contribution statement

Suyanto Suyanto: Conceived and designed the experiments; Analyzed and interpreted the data; Wrote the paper.

Ade Romadhony: Contributed reagents, materials, analysis tools or data.

Febryanti Sthevanie: Performed the experiments; Analyzed and interpreted the data; Contributed reagents, materials, analysis tools or data.

Rezza Nafi Ismail: Performed the experiments; Wrote the paper.

Funding statement

This work was supported by Ministry of Education, Culture, Research, and Technology, Indonesia (DRPM 310/E4.1/AK.04/PT/2021, Matching Fund Kedaireka 3541/E3/SK.09/KL/2021).

Data availability statement

The data that has been used is confidential.

Declaration of interests statement

The authors declare no conflict of interest.

Additional information

No additional information is available for this paper.

References

[1] D. Magdum, M. Suman, System for identifying and correcting invalid words in the devanagari script for text to speech engine, Int. J. Innov. Technol. Exploit. Eng. 8 (6 Special Issue 4) (2019) 1001–1006.

[2] S. Zhou, L. Dong, S. Xu, B. Xu, Syllable-based sequence-to-sequence speech recognition with the transformer in Mandarin Chinese, in: Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH, Vol. 2018, International Speech Communication Association, 2018, pp. 791–795.
Table 4. Examples of named-entities that are wrongly syllabified by the BiLSTM-CNN-CRF-based model trained using the original dataset but correctly by the one trained using the combined original and valid augmented dataset.

| No. | Word              | Syllable sequence resulted using the named-entities dataset | Original | Original + Valid Augmented |
|-----|-------------------|-------------------------------------------------------------|----------|-----------------------------|
| 1   | asriningsuyasa    | as.rin.in.suyasa                                          | as.rin.ning.tyas | as.rin.ning.tyas            |
| 2   | baetubror          | ba.e tu.bro.r                                              | ba.e tu.ab.ro.r | ba.e tu.ab.ro.r            |
| 3   | bakalanpyak        | ba.kalan.pyak                                              | ba.kalan.kra.pyak | ba.kalan.kra.pyak          |
| 4   | bojongsoang        | bo.jong.soang                                              | bo.jong.so.ang | bo.jong.so.ang          |
| 5   | chakrartauma       | chakra.u.ta.ma                                             | cha.kra.u.ta.ma | cha.kra.u.ta.ma          |
| 6   | colkoamimotom      | co.kroa.mi.mo.to                                           | co.kroa.n.mi.mo.to | co.kroa.n.mi.mo.to      |
| 7   | daaruassalam       | da.a rus.sal.lam                                           | da.rus.rus.lam | da.rus.rus.lam         |
| 8   | darussadah         | da.rus.saa.dah                                             | da.ru.s.s.a.dah | da.ru.s.s.a.dah        |
| 9   | gempolitumloko     | gem polo.tuk.lm.ko                                         | gem.pol.tuk.mlo.ko | gem.pol.tuk.mlo.ko  |
| 10  | intiswadaya        | in.tic.wa.da.ya                                            | in.ti.rwa.da.ya | in.ti.rwa.da.ya        |
| 11  | karyasupragha      | kar.ya.nu.gra.ha                                           | kar.ya.nu.gra.ha | kar.ya.nu.gra.ha     |
| 12  | mangunharjo        | ma.ngu.n.har.jo                                            | ma.ngu.har.jo | ma.ngu.har.jo        |
| 13  | marthathlaar       | mar.tha.ti.la.ar                                           | mar.tha.ti.la.ar | mar.tha.ti.la.ar      |
| 14  | musadwipantra      | nu.sad.wi.pan.tra                                         | nu.sa.dwi.pan.tra | nu.sa.dwi.pan.tra |
| 15  | padjiastuti        | pus.dji.as.tu.ti                                           | pus.ji.as.tu.ti | pus.ji.as.tu.ti      |
| 16  | randublatung       | ran.dub.la.tung                                            | ran.du.bla.tung | ran.du.bla.tung     |
| 17  | sasraatmaja        | sas.trast.ma.ja                                            | sas.tr.at.ma.ja | sas.tr.at.ma.ja     |
| 18  | soeryaningsih      | soe.ya.n.ning.sih                                         | soe.ya.ning.sih | soe.ya.ning.sih    |
| 19  | suksesemilang      | suk.se.semilang                                           | suk.se.sce.mi.lang | suk.se.sce.mi.lang |
| 20  | tanjungrum         | tan.ju.nga.rum                                             | tan.jung.a.rum | tan.jung.a.rum      |