Readability and Linearity

Yo Ehara
Tokyo Gakugei University
Faculty of Education, 4-1-1 Nukuikita-machi, Koganei-shi, Tokyo 184-8501 Japan
ehara@u-gakugei.ac.jp

Abstract
This paper investigates automatic assessments of text readability for second language learners. The goodness of text readability assessment methods has been evaluated using the correlation coefficients between their readability scores and labels annotated to the texts by language teachers. For measuring this correlation, previous papers have used Pearson’s correlation coefficient ($\rho$). In this paper, we show that the use of Pearson’s $\rho$ is inappropriate through experiments because it is easily affected by the linearity of the labels and scores, which are often not linear. In contrast, the linearity has no influence on rank correlations, which should be used for evaluating readability assessors. We also propose an unsupervised readability assessor that takes word difficulty for second language learners into account and achieves the best among all unsupervised assessors in our experiments in terms of Pearson’s $\rho$ and the rank correlations.

1 Introduction
Assessing readability plays an essential role in second language acquisition; it can be used for many educational applications such as intelligent reading support systems and placement tests for language classes. Readability assessment is a costly task for educational experts and language teachers. To perform it, they must read a text and assess its readability by guessing how difficult the text is for target learning readers. Hence, to reduce the cost of the labor required by educational experts, the task of automatically identifying the readability of texts for language learners, known as automatic readability assessment (ARA), has been extensively studied in the field of artificial intelligence (AI).

Unsupervised automatic readability assessment appeared early but has recently been reexamined as a research focus. Early studies such as the Dale-Chall formula (1948) (Dale and Chall(1948)), the Flesch Reading Ease formula (Flesch(1948)) (1948), and the Flesch-Kincaid readability formula (1975) (Kincaid et al.(1975)Kincaid, Fishburne Jr, Rogers, and Chissom) were unsupervised, as they did not use costly annotated readability labels. Given a text, these formulae calculate its readability score based on simple superficial textual features such as the average length of a word in the given text. However, most of these early formulae are designed to assess readability for children who are native speakers. Evaluation datasets with readability labels annotated by language teachers targeting second language learners appeared much later in the 2010s (Feng et al.(2010)Feng, Jansche, Huenerfauth, and Elhadad(Xia et al.(2016)Xia, Kochmar, and Briscoe) Vajjala and Lučić(2018)). In these works, automatic readability assessment tasks using these evaluation datasets were formalized as a supervised document classification problem, and substantial research efforts were invested into the construction of classifiers by feature engineering to find complicated textual features that correlate well with readability labels.

Recently, Martinc et al.(2021)Martinc, Pollak, and Robnik-Sikonja revisited the unsupervised approach. They proposed that the perplexity scores of neural language models can also be used to rep-
resent the readability of text for second language learners and proposed to use them for unsupervised automatic readability assessment. The upper part of Fig. 1 show their approach. Given a text, their method uses no valuable readability label for training but uses only the language model trained on other large corpora, and their method predicts the text’s readability score as an output.

While this idea is sound, however, their evaluation is validated using the correlation coefficient, or Pearson’s $\rho$, as illustrated on the right-hand side of Fig. 1. Pearson’s $\rho$ measures the degree of linear correlation between two random variables (Mukaka(2012)). As neither the readability levels of the evaluation corpora nor the readability scores output by unsupervised readability assessors are necessarily linear, the use of Pearson’s $\rho$ can lead to inaccurate evaluation.

This study investigates how the use of Pearson’s $\rho$ affects the evaluation of unsupervised readability assessors. We analyze how unsupervised assessors’ performance can be easily underestimated if the readability scores are not linear. For this purpose, we also build a lightweight unsupervised readability assessor, denoted by the lower part of Fig. 1. We show that, alternatively, rank-correlation coefficients are more robust to the linearity and appropriate for this evaluation.

The contributions of this paper are summarized as follows.

- We indicate that the previous evaluation of unsupervised readability assessors by using Pearson’s $\rho$ is problematic.
- We demonstrate the relationship between the degree of removing linearity from assessors’ scores and the degree to which Pearson’s $\rho$ underestimates the readability score without linearity on a publicly available reliable evaluation dataset.

- We show that, instead of Pearson’s $\rho$, the use of rank-correlation coefficients is appropriate.
- We propose a novel lightweight unsupervised readability assessor that achieves the best performance in terms of both Pearson’s $\rho$ and rank correlation coefficients.

2 Automatic Readability Assessment

This section formalizes the problem of automatic readability assessment. Let us suppose that we have $N$ texts to assess: we write the set of texts as $\{T_i|i \in \{1, \ldots, N\}\}$. Let $Y$ be the set of readability labels. Labels are typically ordered in the order of difficulty. For example, in the OneStopEnglish dataset (Vajjala and Lučić(2018)), we can set $Y = \{0, 1, 2\}$, where 0 is elementary, 1 is intermediate, and 2 is advanced. The number of levels depends on the evaluation corpus. Using $Y$, we write the label for $T_i$ as $y_i \in Y$.

2.1 Goal in Unsupervised Setting

Given each text $T_i$, an assessor outputs its readability score $s_i$. In a supervised setting, the assessor knows the number of levels in the evaluation corpus from training examples. Hence, $s_i$ ranges within $Y$: $s_i \in Y$. However, in an unsupervised setting, it is noteworthy that the assessor does not know $Y$, or how many levels the evaluation corpus has, because no label is given. Hence, even if only integers are allowed for $y_i$, $s_i$ can be a real value.

Throughout this paper, we write arrays using $[\cdot]$ and $\cdot$. Given $N$ texts $\{T_i|i \in \{1, \ldots, N\}\}$, our goal is to make an assessor output arrays of readability scores $\{s_i|i \in \{1, \ldots, N\}\}$ that correlate well with the array of labels $\{y_i|i \in \{1, \ldots, N\}\}$. Here, there are multiple types of correlation coefficients between the array of scores and the array of labels, which we explain in the later sections. Typically, we should use rank coefficients when $s_i$ is real-valued.

2.2 Evaluation and Correlation coefficients

In most evaluation datasets, educational experts are asked to assess text readability by choosing a label from the set of predefined readability labels, $Y$. In
In our setting, Kendall’s τ is defined as follows:

\[ \tau_{y,s} = \frac{\text{cov}(y, s)}{\sigma_y \sigma_s} \quad (1) \]

In Eq. 1, \( \text{cov}(y, s) \) denotes the covariance between \( y \) and \( s \), \( \sigma_y \) denotes the standard deviation of \( y \), and \( \sigma_s \) denotes the standard deviation of \( s \). Eq. 1 ranges \([-1, 1]\), where 1 is the perfect correlation.

However, the Pearson correlation coefficient measures the degree of linear correlation between two random variables (Mukaka(2012)). The readability levels of evaluation corpora are not necessarily linearly distributed. Readability scores that the assessors output are also not necessarily linear. In these cases, it is usually more appropriate to focus on the correlation between the rankings of the readability label \( y_s \) and scores \( s_i \). Rank correlation coefficients measure the correlation between two rankings with the range of \([-1, 1]\). Two types of them are notable: Spearman’s \( \rho \) and Kendall’s \( \tau \) (Alvo and Philip(2014)).

Spearman’s \( \rho \) is defined as the Pearson’s \( \rho \) between two rankings: \( \text{rg}_y \) and convert scores into rankings: \( \text{rg}_s \). Then, using Eq. 1, Spearman’s \( \rho \) is defined as \( \rho_{\text{rg}_y, \text{rg}_s} \). When converting labels into rankings, texts that have the same level are regarded as ties in a ranking. While there are many ways to handle ties, the mid-rank method is usually used in calculating Spearman’s \( \rho \) (Amerise and Tarsitano(2015)). This method simply uses the average of ranks for the rank of a tie. For example, let us consider an array of labels \([2, 1, 1, 0]\). The two 1s in this array are ties taking the 2nd and 3rd ranks. As the average of 2 and 3 is 2.5, the mid-rank ranking of this array is \([4, 2.5, 2.5, 1]\).

We first introduce the definition of Kendall’s \( \tau \) when there are no ties as follows:

\[ \tau = \frac{n_c - n_d}{\text{Num. of all pairs}} \quad (2) \]

Kendall’s \( \tau \) focuses on the pairs of the given arrays: in our setting, \((y_i, s_i)\) and \((y_i', s_i')\) where \( i < i' \). \( n_c \) denotes the number of concordant pairs, \( n_d \) denotes the number of discordant pairs. The pair is said to be concordant if either both \( y_i < y_i' \) and \( s_i < s_i' \) hold or \( y_i > y_i' \) and \( s_i > s_i' \); otherwise, the pair is said to be discordant. If \( y_i = y_i' \), we call \( y_i \) and \( y_i' \) ties. The same holds for \( s \). Num. of all pairs \( = \frac{1}{2} N(N-1) \) when there are no ties.

In reality, \( y \) has many ties, so Eq. 2 cannot be used for the evaluation. There are multiple correction methods to account for ties in Kendall’s \( \tau \); they are named \( \tau_a \), \( \tau_b \), and \( \tau_c \). In our setting, namely unsupervised readability assessment, tau-c should be used because \( y \) and \( s \) may have different scales.

\[ \tau_b = \frac{n_c - n_d}{\sqrt{(n_0 - n_1)(n_0 - n_2)}} \quad (3) \]

\( n_c \) denotes the number of concordant pairs, \( n_d \) denotes the number of discordant pairs. \( n_0 = N(N-1)/2, n_1 \) is the sum of all possible pairs within each tied group for the first quantity, \( n_2 \) is the sum of all possible pairs within each tied group for the second quantity.

\[ \tau_c = \frac{2(n_c - n_d)}{N^2 m^{-1}} \quad (4) \]

3 Proposed Method: Vocabulary Testing

This section describes our unsupervised readability assessor that employs a novel approach: instead of using valuable readability labels as the source of text difficulty for typical second language learners, our proposed method uses vocabulary tests as the source of word difficulty for typical second language learners and obtains readability scores based on accurately estimated word difficulty. To this end, this section explains how to analyze vocabulary test result data to obtain word difficulty.

---

https://en.wikipedia.org/wiki/Kendall\_rank_correlation_coefficient#Tau-a
https://en.wikipedia.org/wiki/Kendall\_rank_correlation_coefficient#Tau-b
https://en.wikipedia.org/wiki/Kendall\_rank_correlation_coefficient#Tau-c
responses to the questions. Statistical models such as item response theory (Baker(2004)) are used to estimate learners' abilities and test questions' difficulties from the learners' responses. To this end, we employed the idea of item response theory to estimate word difficulty values encoding learners' language knowledge. To this end, we used the vocabulary test results: Item Response Theory

We want to analyze vocabulary test results to obtain word difficulty values encoding learners’ language knowledge. To this end, we employed the idea of item response theory (Baker(2004)), a statistical model that can estimate learners’ abilities and test questions’ difficulties from the learners’ responses to the questions.

Let \( V \) be the set of vocabulary, and let \( L \) be the set of learners. Let \( z_{v,l} \in \{0, 1\} \) be the result of whether learner \( l \in L \) correctly answered the question for word \( v \in V \): \( z_{v,l} = 1 \) if \( l \) answered correctly for word \( v \); otherwise, \( z_{v,l} = 0 \). Correct answers usually imply that \( l \) knows word \( v \).

Then, by using \( \{z_{v,l}\} \) as the training data, we train the following model:

\[
p(z = 1|v, l) = \text{sigmoid}(a_l - d_v)
\]  

In Eq. 5, \( a_l \) is the ability parameter of learner \( l \), \( d_v \) is the difficulty of word \( w \), and sigmoid denotes the logistic sigmoid function, i.e., \( \text{sigmoid}(x) = \frac{1}{1 + \exp(-x)} \). The logistic sigmoid function is the binary version of the softmax function, which is frequently used in neural classifiers. It is a monotonously increasing function ranging within \((0, 1)\). As \( \text{sigmoid}(0) = \frac{1}{2} \), when a learner’s ability \( a_l \) is larger than the word difficulty \( d_v \), the probability that learner \( l \) knows word \( v \) can be written as follows: \( p(z = 1|v, l) > \frac{1}{2} \) in Eq. 5. Likewise, by using Eq. 5, we can compare a learner’s ability and word difficulty in the same dimension.

To estimate learner ability and word difficulty, \( z_{v,l} \) is given as \( z \) in Eq. 5 in the training phase. In this way, in item response theory, learner ability and word difficulty are comparable, and these parameters are to be estimated from the test result data.

3.2 Obtaining difficulty of words not in the vocabulary test

In Eq. 5, \( d_v \) denotes the word difficulty estimated from the vocabulary tests. Here, in addition to the word difficulty for the words within the vocabulary test, we also want to obtain word difficulty values for all words that may appear in the target language. To this end, we calculate \( d_v \) from the word frequency in large balanced corpora as follows:

\[
d_v = -\sum_{k=1}^{K} w_k \log(f_{k}(v) + 1)
\]  

In Eq. 6, \( K \) is the number of corpora to use, \( f_{k}(v) \) denotes the frequency of word \( v \) in the \( k \)-th corpus, and \( w_k \) is the weight parameter of the \( k \)-th corpus.

In summary, given the vocabulary test results \( \{z_{v,l}\} \) and corpus frequency features \( f_{k}(v) \), we can estimate the parameters: namely, the weight of the \( k \)-th corpus \( w_k \) and learner \( l \)'s ability \( a_l \). By putting Eq. 5 and Eq. 6 together, we can see that the inside formula of the sigmoid function is linear with respect to the parameters to be estimated because all terms consist of a product of a parameter and a constant calculated from features, and no term has a product of two or more parameters. As the sigmoid function of a linear combination of parameters can be reformulated as a logistic regression, we
can implement Eq. 5 and Eq. 6 by using typical logistic regression classifiers such as scikit-learn[^3] and LIBLINEAR[^4]. We will release our code upon the acceptance of the paper.

Note that we do not use the valuable readability label \{y_i\} in the training phase; hence, our method is categorized as an unsupervised method.

### 3.3 Proposed Automatic Readability Assessor

After estimating the parameters using the above-mentioned procedure, we use the following formula to obtain the readability of given \(T_i\). Here, \(l_{avg}\) denotes the test-taker whose estimated ability parameter is closest to the average of the estimated ability parameter values \(\{a_l\}\)’s. Intuitively, the following equation calculates the probability that the average learner knows all the words that appear in \(T_i\) and uses it as the readability score:

\[
s_i = score(T_i) = -\frac{1}{|T_i|} \log \left( \prod_{v \in T_i} p(z = 1 | v, l_{avg}) \right)
\]

(7)

### 4 Experimental Settings

#### 4.1 Choice of Dataset

We used the OneStopEnglish dataset[^5] for our evaluation because of the following reasons. First, it is one of the newest datasets. Second, it is publicly available and downloadable. Third, it is a reliable dataset in the sense that it has no known pitfalls when used as a corpus for evaluation.

While [Martinc et al.2021] uses other corpora such as the WeeBit corpus [Xia et al.2016] and the Newsela corpus [Xu et al.2015], both have known pitfalls when used for the evaluation of automatic readability assessment. The WeeBit corpus is not a parallel corpus, which is explained in the next subsection. This means that means each level consists of totally different articles covering different topics. As some topics such as politics tend to use more difficult phrases than other topics, it is difficult to see how the topic of content influences the resulting performance values. The Newsela corpus is a parallel corpus, which removes the influence caused by topics. However, according to [Martinc et al.2021], its readability labels can be easily identified from the average sentence length in a text: the average sentence length achieved 0.906 in the Pearson’s \(\rho\) correlation. Hence, even if a method works well on the Newsela corpus, it could be possible that the method merely inherently calculates and uses average sentence length.

#### 4.2 OneStopEnglish dataset

Regarding the source of the dataset, Vajjala and Lučič[^6] says that “onestopenglish.com is an English language learning resources website run by MacMillan Education, with over 700,000 users across 100 countries.” The dataset has three levels: elementary, intermediate, and advanced. According to Vajjala and Lučič[^6], the original articles were taken from the Guardian newspaper. The OneStopEnglish dataset is a parallel corpus, i.e., language teachers manually rewrote the original articles into the three aforementioned readability levels. Hence, one notable characteristic of this dataset is that all three levels have the same content with different readability levels. Hence, by using this dataset, we can avoid having classifiers learn differences in content or topic rather than readability levels.

All three levels have 189 texts each, 567 texts in total. We split these texts into a training set consisting of 339 texts, a validation set consisting of 114 texts, and a test set consisting of 114 texts. The training set and validation sets were used to train solely supervised methods for comparison. Unsupervised methods did not use the training and validation sets; they used only the test set.

#### 4.3 Baseline Methods

##### 4.3.1 Supervised methods

First, we introduce the supervised methods that we used for comparison because it involves the training data mentioned right above. As the BERT-based sequence classification has been reported to achieve excellent results [Devlin et al.2019], we use the following formula to obtain the readability of given \(T_i\). Here, \(l_{avg}\) denotes the test-taker whose estimated ability parameter is closest to the average of the estimated ability parameter values \(\{a_l\}\)’s. Intuitively, the following equation calculates the probability that the average learner knows all the words that appear in \(T_i\) and uses it as the readability score:

\[
s_i = score(T_i) = -\frac{1}{|T_i|} \log \left( \prod_{v \in T_i} p(z = 1 | v, l_{avg}) \right)
\]

(7)
Chang, Lee, and Toutanova), we applied the standard BERT-based sequence classification approach involving pretraining and fine-tuning. For the pretrained model, we used `bert-large-cased-whole-word-masking` in the Huggingface models.

Then, we fine-tuned the model using the aforementioned 339 training texts. For this fine-tuning, we used a GeForce RTX 3090 board that has 24 GiB of Graphical Processing Unit (GPU) memory. The fine-tuning and resulting model took up 16 GiB of GPU memory. This means that it is difficult to achieve similar performance without GPUs with large memory. We named this fine-tuned model `spvBERT`, in which “spv” denotes being supervised. In order to see how the size of training data has an influence on the performance, we also conducted experiments with 168 training texts, which amounted to almost half of the total 339 training texts. We named this model `spvBERT_half`.

All the fine-tuning procedures were conducted using the Adam optimizer (Kingma and Ba(2015)) with a setting of 10 epochs and a 0.00001 training rate.

### 4.3.2 Unsupervised methods

For the implementation of conventional readability formulae, we used the `readability` PyPI package. We used almost all readability formulae implemented in this package for our experiments: namely, Flesch-Kincaid (Flesch-Kincaid Grade Level, FKGL) (Kincaid et al.(1975)), Flesch Reading Ease (Flesch(1948)), the Gunning Fog Index (Gunning(1952)), the SMOG Index (Mc Laughlin(1969)), the RIX index (Anderson(1983)), and the Dale-Chall Index (Dale and Chall(1948)). Among these methods, Notably, some formulae such as the Dale-Chall Index depend on their own list of easy/difficult words. Others, such as the Flesch-Kincaid grade level (FKGL), do not require such a list of difficult words but use superficial features such as the total number of syllables in a text. For space limitation, we do not cite all equations, however, we only cite FKGL as being famous and cite Dale-Chall as showing good performance in our evaluation.

\[
FKGL = 0.39 \frac{\text{total words}}{\text{total sentences}} + 11.8 \frac{\text{total syllables}}{\text{total sentences}} - 15.59
\]  
\[
Dale-Chall = 15.79 \times \left( \frac{\text{difficult words}}{\text{words}} \right) + 0.0496 \left( \frac{\text{words}}{\text{sentences}} \right)
\]

More details of these formulae and their implementation are described on the project page. All of these readability formulae are unsupervised in the sense that they do not require any training data.

For the unsupervised neural language model, we also used the `bert-large-cased-whole-word-masking` pretraining model and used the `BertForMaskedLM` function to obtain the perplexity of each sentence of the text of interest. We chose this pretraining model because Martinc et al.(2021) reported that they used `bert-base-uncased` and reported not so good performance, so we chose a BERT-based model larger than the one that they used. Note that, unlike neural sequence classification, language models are designed to be unsupervised and thus do not require any training data to fine-tune. All we need to do for the neural language model is to input each sentence in the text of interest and calculate the perplexity score of the inputted sentence.

For splitting a text into sentences, we used the `sent_tokenize` function in the `nltk` Python package. After the split, we simply used the average of the perplexity scores of each sentence in a text as the readability score. As the perplexity score of a sentence encodes the fluency of the inputted sentence, this roughly measures the overall fluency of the inputted sentence. We call this method `BERTLMAvg`, where LM denotes a language model.

As `BERTLMAvg` does not use fine-tuning, it uses less GPU memory compared to `spvBERT`. However, `BERTLMAvg` uses 1,793 MiB of GPU memory to output perplexity scores, which is still impractical in a low-computational-resource environment.

---

5 [https://huggingface.co/models](https://huggingface.co/models)
6 [https://pypi.org/project/readability/](https://pypi.org/project/readability/)
7 [nltk.org](nltk.org)
According to Martinc et al. (2021), BERT language models do not perform good results. Hence, while not directly comparable because we could not obtain their test set, for a rough comparison, we cited their best model on the OneStopEnglish dataset, TCNRSSRS-simple. The model is temporal convolutional network (TCN) trained on the Simplified Wikipedia corpus. For space limitations, refer to Martinc et al. (2021) for the details of this method.

Proposed model was trained on a previously published and publicly available vocabulary dataset (Ehara (2018)). For the corpus word frequency, we used the frequencies taken from the British National Corpus (BNC Consortium (2007)) and the Corpus of Contemporary American English (COCA) (Davies (2008)). Both corpora are balanced general corpora used extensively in English education (Nation (2006)). Especially, the word frequencies of these corpora are important resources for determining word difficulty in English education. For counting text frequencies, we used nltk.stem.WordNetLemmatizer in the nltk package to lemmatize words appearing in running texts.

Our Proposed model uses the average of the negative log likelihood that an average learner knows each word in the text as presented in Eq. 7. As our Proposed model uses the BNC and COCA word frequencies, it could be possible that these word frequencies have an essential influence on the performance of the Proposed model. To check this, we also measured the correlation between the gold labels and the average negative log of the unigram probability values of the given text in each corpus. We name these feature-based methods as BNC and COCA.

4.4 Experimental Results: Pearson’s ρ and performance

This subsection describes the experimental results showing the problem of using Pearson’s ρ in evaluation.

Tab. I shows the experimental results. The columns of Tab. I show the rank correlation coefficients introduced in the previous sections. Namely, they are Spearman’s ρ, Kendall’s τ with tie correction type b (τ-b), and Kendall’s τ with tie correction type c (τ-c). Pearson’s ρ is shown in the rightmost column. As we explained in previous sections, Pearson’s ρ is affected by the linearity of scores. To see how Pearson’s ρ is affected by the linearity of scores, below each unsupervised method M, we show \( \exp(M) \) to indicate the resulting performance values when we replaced the scores of \( M \) with the exponentized scores of \( M \), i.e., \( \exp(\text{the scores of } M) \) to remove linearity. We also show \( \exp(cM) \) to denote the values of \( \exp(c \times \text{the scores of } M) \), where \( c \) is a constant. The distinction of “unsupervised” and “supervised” is clearly marked in the leftmost column.

In Tab. I we can easily see that, for all unsupervised methods except for BNC and COCA, the correlations of the exponentized scores measured by Pearson’s ρ are closer to 0 than their original scores. In contrast, the rank correlation coefficient values are kept unchanged because \( \exp \) is a monotonous function and hence the ranking is not altered by the use of \( \exp \). By choosing a small positive constant \( 0 < c < 1 \) in Pearson’s ρ of \( \exp(cM) \), we can weaken the effect of the drop of Pearson’s ρ: for example, in Tab. I the Pearson’s ρ of \( \exp(0.5Flesch-Kincaid) \) was 0.179, which is between 0.149 and 0.359. We can easily see this by considering the Taylor expansion of the \( \exp \) function: \( \exp(cx) = \sum_{n=0}^{\infty} \frac{c^n x^n}{n!} \). Hence, by choosing a small \( c \), the effect of \( c^n x^n \) approaches zero where \( n \geq 2 \) and only \( 1 + cx \) remains, so the effect caused by \( \exp(cx) \) is close to that of a linear function. The reason why the performance values of BNC and COCA seem slightly increased by \( \exp \) is presumably because of noise: they did not correlate with the readability labels statistically significantly. Neither did \( \exp(\text{BNC}) \) and \( \exp(\text{COCA}) \).

The drop in performance scores is enormous for some methods such as Proposed: its performance was originally 0.715 but plunges to 0.260 by using \( \exp \). This result indicates the vulnerability of using Pearson’s ρ in the evaluation: the evaluation by Pearson’s ρ is strongly affected by how linear the scores are, suggesting the use of rank correlation for better evaluation.

TCNRSSRS-simple is the best model using the same dataset in Martinc et al. (2021). As they show only the performance measured by the Pearson correlation,
Table 1: Experimental Results on the OneStopEnglish Dataset. For a method M, exp(M) denotes the correlations between the array of exp(M’s score) and the gold labels. (*) denotes that the value is cited from other papers.

<table>
<thead>
<tr>
<th>Supervision</th>
<th>Method</th>
<th>Spearman’s $\rho$</th>
<th>Kendall’s $\tau$-b</th>
<th>Kendall’s $\tau$-c</th>
<th>Pearson’s $\rho$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unsupervised</td>
<td>Flesch-Kincaid</td>
<td>0.324</td>
<td>0.253</td>
<td>0.308</td>
<td>0.359</td>
</tr>
<tr>
<td></td>
<td>exp(Flesch-Kincaid)</td>
<td>0.324</td>
<td>0.253</td>
<td>0.308</td>
<td>0.179</td>
</tr>
<tr>
<td></td>
<td>exp(0.5Flesch-Kincaid)</td>
<td>0.324</td>
<td>0.253</td>
<td>0.308</td>
<td>0.149</td>
</tr>
<tr>
<td></td>
<td>ARI</td>
<td>0.317</td>
<td>0.248</td>
<td>0.302</td>
<td>0.351</td>
</tr>
<tr>
<td></td>
<td>exp(ARI)</td>
<td>0.317</td>
<td>0.248</td>
<td>0.302</td>
<td>0.136</td>
</tr>
<tr>
<td></td>
<td>Coleman-Liau</td>
<td>0.373</td>
<td>0.295</td>
<td>0.359</td>
<td>0.185</td>
</tr>
<tr>
<td></td>
<td>exp(Coleman-Liau)</td>
<td>0.373</td>
<td>0.295</td>
<td>0.359</td>
<td>0.152</td>
</tr>
<tr>
<td></td>
<td>FleschReadingEase</td>
<td>-0.387</td>
<td>-0.301</td>
<td>-0.366</td>
<td>-0.426</td>
</tr>
<tr>
<td></td>
<td>exp(FleschReadingEase)</td>
<td>-0.387</td>
<td>-0.301</td>
<td>-0.366</td>
<td>-0.169</td>
</tr>
<tr>
<td></td>
<td>GunningFogIndex</td>
<td>0.331</td>
<td>0.257</td>
<td>0.313</td>
<td>0.362</td>
</tr>
<tr>
<td></td>
<td>exp(GunningFogIndex)</td>
<td>0.331</td>
<td>0.257</td>
<td>0.313</td>
<td>0.151</td>
</tr>
<tr>
<td></td>
<td>LIX</td>
<td>0.348</td>
<td>0.273</td>
<td>0.332</td>
<td>0.383</td>
</tr>
<tr>
<td></td>
<td>exp(LIX)</td>
<td>0.348</td>
<td>0.273</td>
<td>0.332</td>
<td>0.129</td>
</tr>
<tr>
<td></td>
<td>SMOGIndex</td>
<td>0.456</td>
<td>0.360</td>
<td>0.438</td>
<td>0.479</td>
</tr>
<tr>
<td></td>
<td>exp(SMOGIndex)</td>
<td>0.456</td>
<td>0.360</td>
<td>0.438</td>
<td>0.306</td>
</tr>
<tr>
<td></td>
<td>RIX</td>
<td>0.437</td>
<td>0.340</td>
<td>0.414</td>
<td>0.462</td>
</tr>
<tr>
<td></td>
<td>exp(RIX)</td>
<td>0.437</td>
<td>0.340</td>
<td>0.414</td>
<td>0.181</td>
</tr>
<tr>
<td></td>
<td>DaleChallIndex</td>
<td>0.495</td>
<td>0.387</td>
<td>0.472</td>
<td>0.506</td>
</tr>
<tr>
<td></td>
<td>exp(DaleChallIndex)</td>
<td>0.495</td>
<td>0.387</td>
<td>0.472</td>
<td>0.431</td>
</tr>
<tr>
<td></td>
<td>TCN RSRS-simple</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>0.615(*)</td>
</tr>
<tr>
<td></td>
<td>BERTLMavg</td>
<td>-0.220</td>
<td>-0.173</td>
<td>-0.210</td>
<td>-0.040</td>
</tr>
<tr>
<td></td>
<td>exp(BERTLMavg)</td>
<td>-0.220</td>
<td>-0.173</td>
<td>-0.210</td>
<td>-0.005</td>
</tr>
<tr>
<td></td>
<td>BNC</td>
<td>-0.012</td>
<td>-0.009</td>
<td>-0.010</td>
<td>-0.006</td>
</tr>
<tr>
<td></td>
<td>exp(BNC)</td>
<td>-0.012</td>
<td>-0.009</td>
<td>-0.010</td>
<td>-0.123</td>
</tr>
<tr>
<td></td>
<td>COCA</td>
<td>-0.018</td>
<td>-0.016</td>
<td>-0.020</td>
<td>-0.039</td>
</tr>
<tr>
<td></td>
<td>exp(COCA)</td>
<td>-0.018</td>
<td>-0.016</td>
<td>-0.020</td>
<td>-0.121</td>
</tr>
<tr>
<td></td>
<td>Proposed</td>
<td>0.730</td>
<td>0.592</td>
<td>0.709</td>
<td>0.715</td>
</tr>
<tr>
<td></td>
<td>exp(Proposed)</td>
<td>0.730</td>
<td>0.592</td>
<td>0.709</td>
<td>0.260</td>
</tr>
<tr>
<td>Supervised</td>
<td>spvBERT_half</td>
<td>0.751</td>
<td>0.729</td>
<td>0.725</td>
<td>0.747</td>
</tr>
<tr>
<td></td>
<td>spvBERT</td>
<td>0.866</td>
<td>0.856</td>
<td>0.854</td>
<td>0.864</td>
</tr>
</tbody>
</table>

we wrote — for other rank correlation coefficients. Additionally note that we cannot make a direct comparison as we could not obtain their test set used for their experiments. This is marked by the (*) after the value. While we can see that Proposed achieved better correlation than TCN RSRS-simple, we are not sure if this result indicates the linearity of the methods or the superiority of Proposed against TCN RSRS-simple. Likewise, the use of Pearson’s $\rho$ only makes follow-up papers’ efforts to compare their results difficult.

4.5 Performance Comparison

In all unsupervised methods, our Proposed method achieved the best results in all rank correlation coefficients and Pearson’s $\rho$, although we need to be careful with the interpretation of Pearson’s $\rho$ as we pointed out in the previous section. These results were statistically significant ($p < 0.01$): all correlation coefficients can also be used for statistical testing. In each of the statistical tests, the null hypothesis is that no association exists between the
scores and the gold labels. When measured using Spearman’s $\rho$, **Proposed** achieved a value of 0.730, which is close to 0.751, the performance achieved by supervised BERT using half of the training data.

**BERTLMavg** did not achieve good results in predicting readability labels. This result suggests that perplexity and readability are different measures and that, to measure readability, we should obtain and use information about what a typical language learner knows about the target second language.

Interestingly, **BNC** and **COCA** did not achieve good results in predicting readability labels. This result shows that the reason that **Proposed** method outperformed the others is not merely because the features that **Proposed** used are excellent. A good combination of the two balanced corpora results in significant results. The use of only one of the two does not achieve good results. Hence, we can see that **Proposed** works excellently for making the combination of the two corpus-based features.

For a comparison with supervised models, Tab. II shows the performance by the supervised models, namely **spvBERT** and **spvBERT_half**. Supervised models output labels rather than scores in their prediction phase: we directly used these labels to calculate rank correlation coefficients for a fair comparison with unsupervised models. Leveraging the supervision, they outperformed most of the unsupervised methods in all rank correlation coefficients. This means that using valuable supervision yields great improvement in the predictive performance of readability.

The performance differences among **spvBERT**, **BERTLMavg**, and **Proposed** can be interpreted as follows. **BERT** is a large model trying to use as much information as possible from a sentence, such as syntactic structure. Hence, it is difficult for the model to find useful information contributing to readability without supervision. **Proposed** is a bag-of-words model that is designed to be lightweight by sacrificing such complicated factors. Hence, the performance difference between **spvBERT** and **Proposed** can be regarded as the degree that information beyond word difficulty – such as syntactic information or sentence context – accounts for readability. While this is beyond the focus of this paper, a detailed error analysis between **spvBERT** and **Proposed** may lead to understanding what kind of syntactic information or contexts in a sentence contribute to readability.

### 4.6 Memory and Speed

We used a Core i7-10700K (3.80GHz) machine with a GeForce RTX 3090 board for all experiments. The **BERTLMavg**, which is an unsupervised BERT language model, uses 1,793 MiB GPU memory. In contrast, **Proposed** is merely a logistic regression, it does not need any GPU for practical use. The features of **Proposed** are also small compared to those of the BERT models. **Proposed** uses the BNC and COCA frequencies, which amount to 10 MiB of CPU memory, which is roughly $\frac{1}{100}$ of that used by the unsupervised BERT models. In terms of speed, to classify all texts in the test set, while **BERTLMavg** took 368 seconds, **Proposed** took only 5.37 seconds. This means that, compared to **BERTLMavg**, the **Proposed** is 68.5 times faster.

### 5 Conclusions

We investigated the correlation coefficients to evaluate unsupervised automatic readability assessors’ performance. In the experiments, we show that the readability performances measured by Pearson’s $\rho$ are strongly affected by the linearity of the outputted scores by taking their $exp(cx)$, whereas those measured by rank correlations are not affected. This indicates the appropriateness of using rank correlation coefficients to evaluate unsupervised automatic readability assessors. We also proposed an unsupervised assessor based on word difficulty for typical second language learners calculated from a vocabulary test result dataset. This achieved the best in the unsupervised assessors compared.

Future work includes a more detailed analysis to investigate which rank correlations, including those not introduced in this paper, should be more appropriate for the evaluation.

### Acknowledgment

This study was supported by JST ACT-X Grant Number JPMJAX2006 and JSPS KAKENHI Grant Number 18K18118. We used the ABCI infrastructure from AIST for the computational resources. We appreciate anonymous reviewers for their valuable comments.
References


