Information Engineering Express
International Institute of Applied Informatics
2023, Vol. 9, No. 2, IEE744

Revisiting the Algorithm RBSC-SubGen: Exploring its Limits and Improving its Convergence

Parisa Supitayakul *, Kohei Furuya *, Zeynep Yücel *,
Akito Monden *, Pattara Leelaprute †

Abstract

This study focuses on the algorithm `RBSC-SubGen`, which is originally offered for generation of vocabulary decks but can potentially be applied for a variety of problems. We first study the resilience of `RBSC-SubGen` against various hyper-parameters by testing it under various constraints. Our results indicate that `RBSC-SubGen` is sensitive to subset size S, followed by desired RBSC coefficient ρ^*, permissible disparity ε and, finally, parent set size L. We then offer a simple modification for reducing the computational load of `RBSC-SubGen` and avoiding saturation. In particular, we offer to carry out an intermediate-level verification of the desired conditions. Additional tests show that the rate of saturation and the number of iterations decrease, whereas the disparity in the RBSC coefficient increases within acceptable limits.

Keywords: Simple difference formula, subset selection problem, rank-biserial correlation.

1 Introduction and Motivation

Real-world processes involve a certain degree of randomness. In modeling and estimation, the observations obtained from such processes are often treated as random variables and represented in terms of stochastic models.

The problem of *Ranking-and-selection* (R&S) from the computational statistics research field, focuses on processes involving random factors [1]. In particular, the objective is to choose the best of two (or sometimes more) processes (or items), which bear a certain degree of randomness, according to a given metric or measure. For reliably solving the R&S problem, it is necessary to take as many measurements as possible. However, this also implies that numerous physical experiments need to be carried out, which in certain cases may not be possible (e.g. vehicular traffic safety analysis). In addition, in certain other cases (e.g. pharmacological studies), they may cost a long time and/or financial resources.

In that matter, a convenient way of exploring process performance is to make *simulations* of stochastic models on computer platforms. Simulations enable the extension of

* Okayama University, Okayama, Japan
† Department of Computer Engineering, Faculty of Engineering, Kasetsart University, Bangkok, Thailand

Specifically, according to Kerby [19], the non-parametric correlation equals the simple difference between the proportion of "favorable" and "unfavorable" evidence, where favorable stands for the pairs supporting the hypothesis and unfavorable for the ones disagreeing with the hypothesis. Thus, RBSC coefficient ρ can be computed in explicit terms as,

$$\rho = \frac{N_S - N_C}{N_S + N_C}. \tag{3}$$

It is clear from Equation 3 that ρ is bounded in the range $[-1, 1]$. If the data are all favorable, then ρ will be exactly 1. On the contrary, if the data are all unfavorable, then ρ will be -1, whereas a correlation of 0 indicates an equal amount of favorable and unfavorable evidence. In that respect, if $\rho > 0$, it can be said that the hypothesis holds, and the level of validity can be measured by taking into account how close ρ is to 1.

3.1 Outline of the original RBSC-SubGen algorithm

Generally speaking, the purpose of `RBSC-SubGen` is to build a desired number (2 or possibly more) subsets (out of a large parent set). The parent set X involves certain items, each of which is associated with a *score*. The score can be an objective value (e.g. the height of a person as in the aforementioned example) or a subjective value (e.g. evaluation of a product in an online retail store). In the case of [4], the parent set is a lexicon, the subsets are vocabulary decks and the scores are the number of occurrences of the words in daily language.

Since the purpose of our work is not to adapt `RBSC-SubGen` to a certain data set or to try it with a certain set of constraints, in what follows, we avoid specifying the parent set (i.e. its items or how they are scored) and work on randomly generated sets from the standard normal distribution.

The essential condition of `RBSC-SubGen` is to build the subsets so as to attain the desired ranking relation, which is quantified in terms of rank-biserial correlation. In that respect, the inputs of `RBSC-SubGen` are composed of the parent set, the size of the output subsets, the desired RBSC coefficient ρ^* and a tolerable disparity on ρ^*, ε. Namely, any pair of subsets with an RSCB coefficient within the range $[\rho^* - \varepsilon, \rho^* + \varepsilon]$ is considered to be an output of `RBSC-SubGen`[1].

Specifically, [4] proposes building two initial (i.e. potential) subsets in an arbitrary manner and then updating them in an iterative fashion (by inserting and removing elements) such that the RBSC coefficient of the subsets converge to the desired value at each update. The algorithm ceases the update as soon as an RSCB coefficient within the acceptable range is attained and returns the subsets.

The most crucial step of `RBSC-SubGen` is the iterative update (see Lines 4-8 of Alg. 1). In realizing the update, firstly the current state of the subsets is assessed by calculating ρ (see Line 3 of Alg. 1). Then, based on the disparity of ρ in relation to ρ^* (i.e. whether it is too high or too low), the sort of necessary update (i.e. insertion/removal) is determined.

Suppose that between the subsets A and B, the one with lower scores (on average) is A. Further suppose that ρ_{AB} is lower than the desired value ρ^* than a maximum permissible disparity ε, i.e. $\rho^* - \rho_{AB} > \varepsilon$. In this case, A has to be updated such that an item with a relatively low score needs to be inserted into it and an item with a relatively high score needs to be removed (i.e. returned from A to X', where X' is the remaining elements of the

[1] In that respect, the solution is not unique.

Algorithm 3: `UpdateSubset`

Input: A, U_A, X'
Output: A, X'
if $U_A = -1$ **then** `// Decrease scores in` A
  `/* Pick a high-score item` $a \in A$ `and a low-score item` $x \in X'$ `*/`
  sample$(a \in A,\ x \in X' \mid s_a > m_A,\ s_x < m_A)$
else if $U_A = 1$ **then** `// Increase scores in` A
  `/* Pick a low-score item` $a \in A$ `and a high-score item` $x \in X'$ `*/`
  sample$(a \in A,\ x \in X' \mid s_a < m_A,\ s_x > m_A)$
$A = A \bigcup \{x\} \setminus \{a\}$ `/* Move` a `from` A `to` X' `and` x `from` X' `to` A `*/`
$X' = X' \bigcup \{a\} \setminus \{x\}$
return A, X'

4.1 Hyper-parameters

The hyper-parameters of `RBSC-SubGen` are presented in Table 1. In what follows, we discuss the impact of each hyper-parameter on performance.

As the desired value of the RBSC coefficient ρ^* increases, there will be a larger margin between the scores of the items in the subsets. In addition, since the parent set is assumed to come from a normal distribution, the bulk of the items have medium scores. Retrieving an item from one of the two extremes advances ρ much more quickly in the desired direction, but the chance of sampling such a value is lower due to the normal distribution assumption. Thus, higher values of ρ^* indicate more challenging problems, which are likely to terminate after a larger number of iterations. If ρ^* is significantly high (or low), the algorithm is likely to get saturated, i.e. reach the maximum number of iterations $N_{\max}$ without a successful completion.

Table 1: Hyper-parameters of the algorithm.

Variable	Description	Min	Max	Step size	Default
ρ^*	Desired value of RBSC coefficient	0.3	0.7	0.04	0.5
ε	Maximum permissible disparity on ρ^*	0.05	0.15	0.01	0.1
L	Size of the parent set ($\lvert X \rvert$)	100	900	50	500
S	Size of subsets ($\lvert A \rvert$, $\lvert B \rvert$)	100	500	20	300

In most R&S problems, the parent set X is collected through physical or simulated experiments and thus has a limited size. This implies that it may not be possible to achieve the exact value of ρ^*. In that case, it is necessary to define a maximum permissible value ε for the disparity on ρ^*, $\Delta\rho = |\rho^* - \rho|$. Provided that $\Delta\rho \leq \varepsilon$, ρ is considered to be sufficiently close to ρ^*. Clearly, if ε is low, a more accurate solution is desired, which may require a larger number of iterations and may even result in a failure in achieving convergence.

The challenge due to the limited size of the parent set deserves a devoted hyper-parameter, which we denote with L. If L is large, the algorithm is expected to have sufficient freedom in choosing the scores and is likely to converge (provided that the other hyper-parameters are not too strict). Nevertheless, if L is too small, even if the other hyper-parameters are not

to the number of all attempts (i.e. I executions). In that respect, β_0 quantifies how often `RBSC-SubGen` fails to build the subsets.

If two subsets are built in less than $N_{\max}$ iterations successfully, the algorithm is said to *converge*. In addition, the number of iterations until convergence denoted with M_0) is considered to be another performance indicator. Specifically, M_0) expresses how quickly `RBSC-SubGen` builds the subsets[3].

As mentioned in Section 3.1, ε defines the maximum permissible disparity on ρ^*. If two subsets are built successfully, ρ is certainly closer to ρ^* than ε. This actual value of disparity $\Delta\rho$ is considered as another performance indicator. The observed disparity $\Delta\rho$ ideally depends strictly on ε. In that respect, the reason for computing $\Delta\rho$ is for making sure that the maximum number of iterations $N_{\max}$ is sufficiently high. In other words, if it can be observed that $\Delta\rho$ depends solely on ε, it can be claimed that the values β_0 and M_0) are likely to grasp the general characteristics, and additional test runs are not necessary.

4.4 Results concerning rate of saturation

If the algorithm ceases without yielding an outcome (i.e. it is saturated), there is no point in studying most performance indicators. In that respect, the rate of saturation β_0 is considered to be the most important marker of performance.

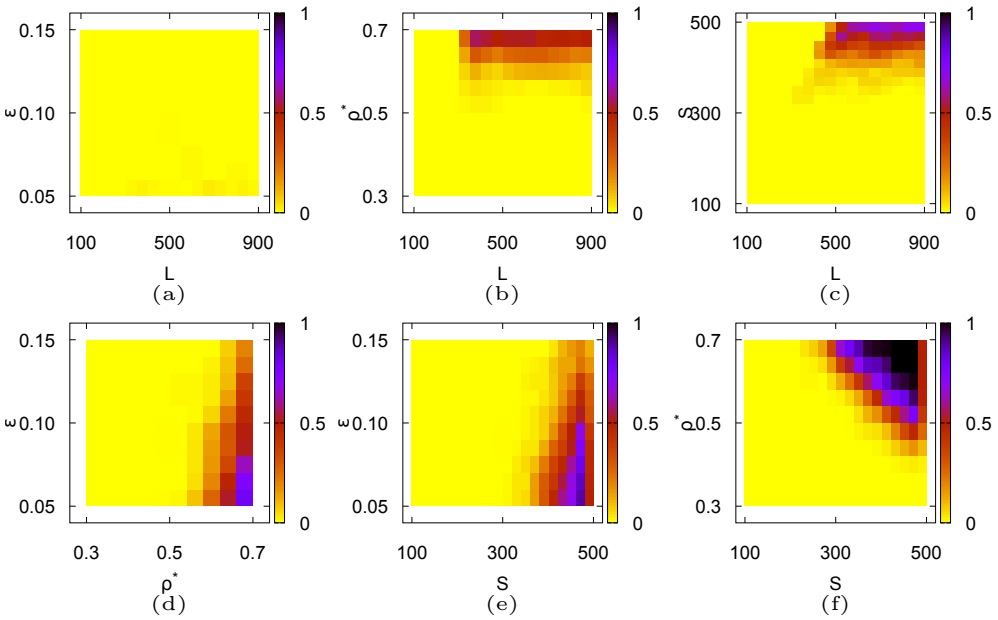

Figure 1: The effect of each possible hyper-parameter pair on β_0.

Figure 1 illustrates the rate of saturation β_0 for each possible hyper-parameter pair. In particular, the hyper-parameter pair (ρ^*, S) often poses a challenge. Namely, there is a

[3]If the algorithm gets saturated, it does not converge and thus the number of iterations is not considered in computing M_0). Moreover, in the case that the algorithm gets constantly saturated for a certain set of hyper-parameters, M_0) is not available.

higher risk that the algorithm gets saturated when both the desired value of RBSC coefficient ρ^* and the size of target subsets S are large.

One may see in Figures 1-(d), (e) that large values of ρ^* pose a risk of saturation, also when they are coupled with small values of ε. In particular, the algorithm is observed to be more sensitive to ε than L (see also Figures 1-(b) and (d)).

In addition, a similar observation is valid also for S. Namely, when large values of S are coupled with small values of ε, a risk of saturation emerges. In addition, such a risk is more serious for ε than L (see Figures 1-(c) and (e)). Moreover, concerning the hyper-parameter pair of (ε, L), the risk of saturation is not serious (see Figure 1-(a)).

4.5 Results concerning number of iterations

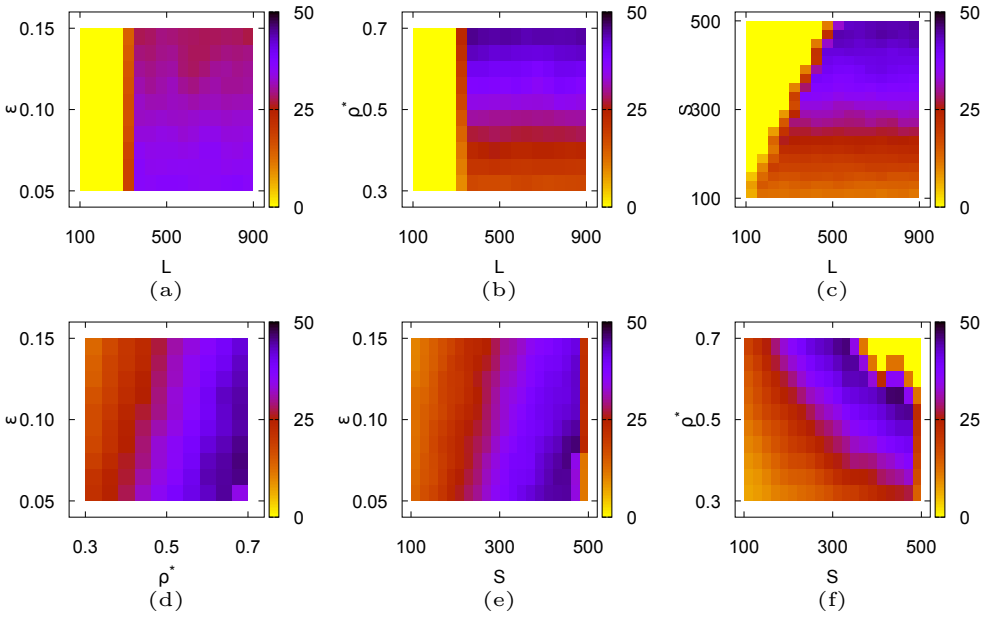

Figure 2: The effect of each possible hyper-parameter pair on M_0).

Figure 2 illustrates our results concerning the number of iterations M_0) that it takes `RBSC-SubGen` to terminate successfully. In this figure, the yellow regions indicate either (i) the cases where the problem is not feasible (i.e. $L < 2 \cdot S$) or (ii) that all I runs are found to be saturated. If (i) is the case, there exists no solution due to an ill definition of the problem statement. However, if (ii) is the case, then one may try to improve the algorithm to overcome saturation. In Figures 2-(a), (b), (c), the yellow regions arise due to infeasibility. However, in Figures 2-(e), (f), they arise due to constant saturation.

Omitting the aforementioned cases (i) and (ii), we focus on successful executions. It is interesting that although the hyper-parameter pair of (ρ^*, S) has the largest number of saturations (see Figure 1-(f)), a higher number of iterations is required to solve most problems relating to the hyper-parameter pair of (ε, S). This observation is somewhat valid also for the hyper-parameter pair of (ε, ρ^*). As a consequence, the number of iterations is higher

for most (ε, S) and (ε, ρ^*) pairs, although saturation is not as common as in the case of (ρ^*, S).

4.6 Results concerning disparity on desired RBSC coefficient

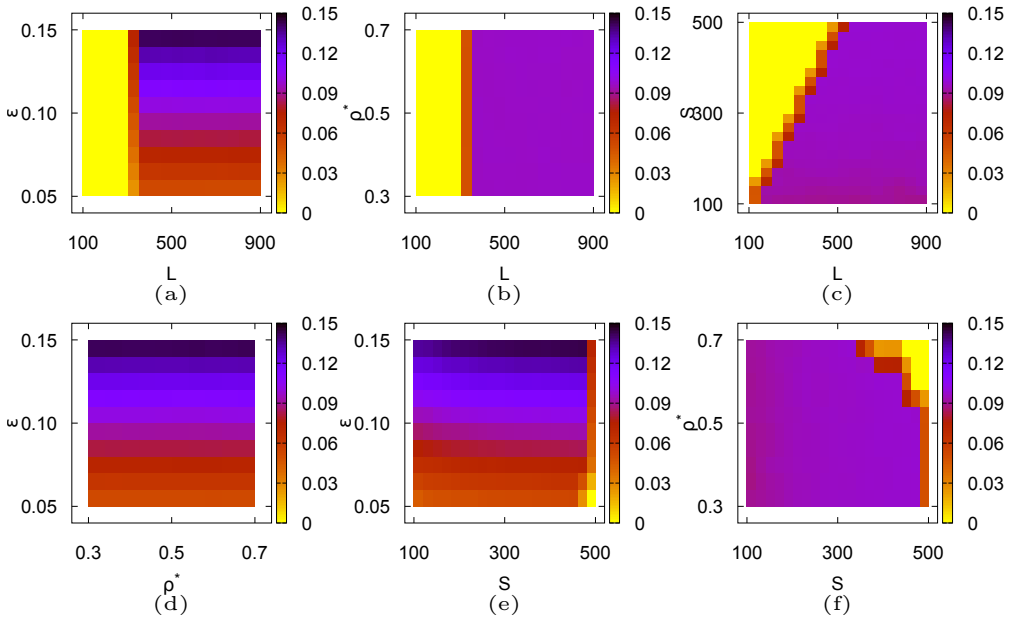

Figure 3: The effect of each possible hyper-parameter pair on $\Delta\rho$.

Finally, the evolution of disparity $\Delta\rho$ on the desired RBSC coefficient ρ^* is investigated relating to the six hyper-parameter pairs. It is observed in Figures 3-(a), (d), (e) that as long as ε is one of the hyper-parameters under investigation, $\Delta\rho$ depends solely on it and the other hyper-parameter under investigation does not introduce a prominent effect on $\Delta\rho$. This is not surprising since the termination of iterations is decided based on ε.

5 Improving Convergence Characteristics of RBSC-SubGen

In Alg. 1, the subsets are updated *in turn* at each iteration (see Lines 6, 7 in Alg. 1). Specifically, an insertion/removal is performed first on the subset with the lower ranks A, and then on the subset with higher ranks B. In that respect, the RBSC coefficient is modified already once after updating A. Therefore, it is worth checking whether it is within the acceptable range after this first update.

In that respect, we propose repeating the computation of ρ (see Line 8 of Alg. 1) between the updates of A and B (i.e. between Lines 6 and 7 of Alg. 1). This modification is expected to reduce β_0 and M_0, whereas $\Delta\rho$ is expected not to be affected. We test the modified algorithm with the same rules as defined in Sections 4.1 and 4.2 and evaluate its performance with the same indicators defined in Section 4.3. We present the results concerning this modification in Sections 5.1∼5.3.

5.1 Results concerning rate of saturation

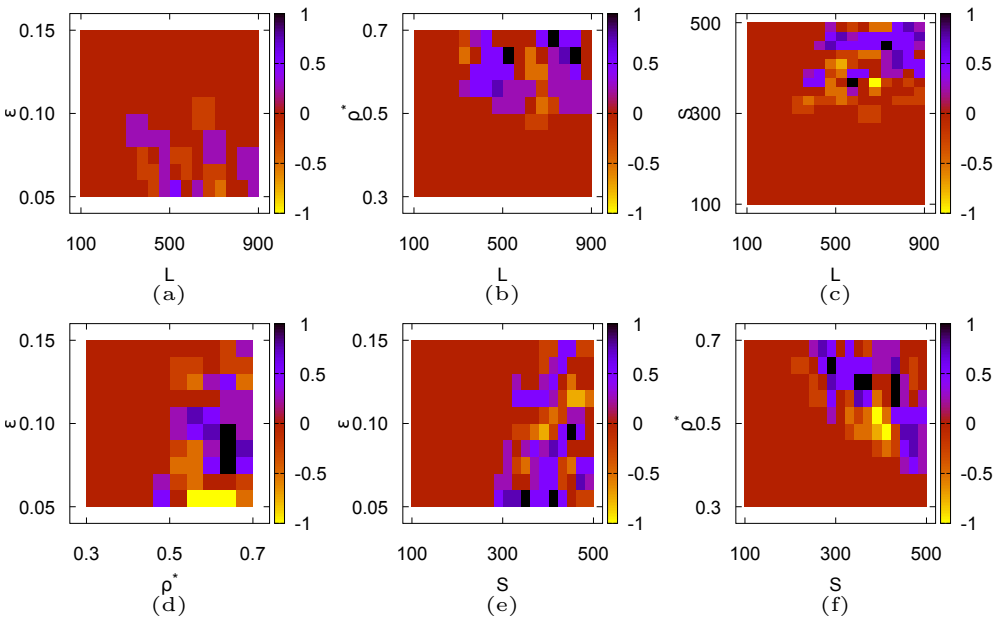

Figure 4: The change induced by the modified algorithm on β_0 for each possible hyper-parameter pair.

Figure 4 shows the change induced by the modified algorithm on β_0 for each possible hyper-parameter pair. Specifically, we run the modified algorithm I times and compare the number of saturated cases to the average number of saturated cases of the original algorithm. The improvement or deterioration is decided in a binary manner. Namely, if at a particular execution, the modified algorithm has less saturations than the original algorithm, the result is registered as 1, and otherwise, as a -1. This process is repeated I times and the averages corresponding to each hyper-parameter pair are illustrated in Figure 4. Excluding the cases where there is no difference, we can see in Figure 4 that positive values are in majority. In particular, we have improvement rates between 0.51 and 0.69.

5.2 Results concerning number of iterations

Figure 5 depicts the change induced by the modified algorithm on M_0 for each possible hyper-parameter pair. We carry out additional tests as explained in Section 5.1. We assess the change in performance in a binary manner. Namely, comparing the number of iterations of the modified algorithm concerning a particular hyper-parameter pair at a particular execution, to the average number of iterations of the original algorithm for the same pair, we register the result as a 1, if the former is larger, and otherwise, as a 0. We then take the average of th registered value for the I executions. Excluding the cases where there is no difference, we can see in Figure 5 that most M_0 values are positive, and only in a small number of cases the number of iterations increases. In particular, we have improvement rates between 0.57 and 0.69.

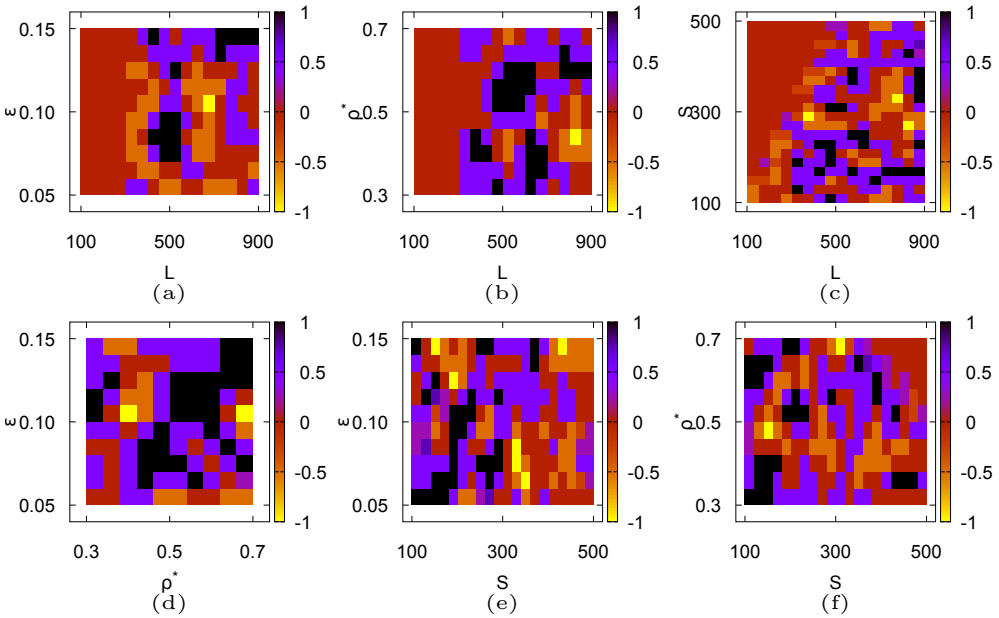

Figure 5: The change induced by the modified algorithm on M_0 for each possible hyper-parameter pair.

5.3 Results concerning disparity on desired RBSC coefficient

Figure 6 depicts the change induced by the modified algorithm on $\Delta\rho$. The details of experimentation and the method of assessment of improvement or deterioration are similar to Sections 5.1 and 5.2. Excluding the cases where the problem is not feasible and where there is no difference, we can see that there is a fair degradation. Note that since ε is the determining factor on $\Delta\rho$ as shown in Section 4.6, it is not surprising that a smaller number of iterations lead to subsets with slightly larger disparity on $\rho*$. It is important to note that the disparity is still smaller than the maximum permissible value ε.

6 Conclusions

This study focuses on `RBSC-SubGen`, which is originally designed for building vocabulary decks (out of a large corpus) with the desired level of word frequency relation. Exploiting the fact that this objective shares many common aspects with the generic subset selection problem, we tried `RBSC-SubGen` in generating subsets out of different hypothetical parent sets. We also imposed varying constraints on subset size, desired ranking relation, and permissible disparity. Our results indicate that `RBSC-SubGen` can be used in subset selection, provided that it is formulated as a ranking relation. In addition, `RBSC-SubGen` is found to be sensitive to subset size S, followed by desired RBSC coefficient ρ^*, permissible disparity ε and, finally, parent set size L. We then proposed a simple modification to the original algorithm such that the value of the RBSC coefficient is checked once after each update. Our results indicate to a decrease in the rate of saturation as well as the number of iterations,

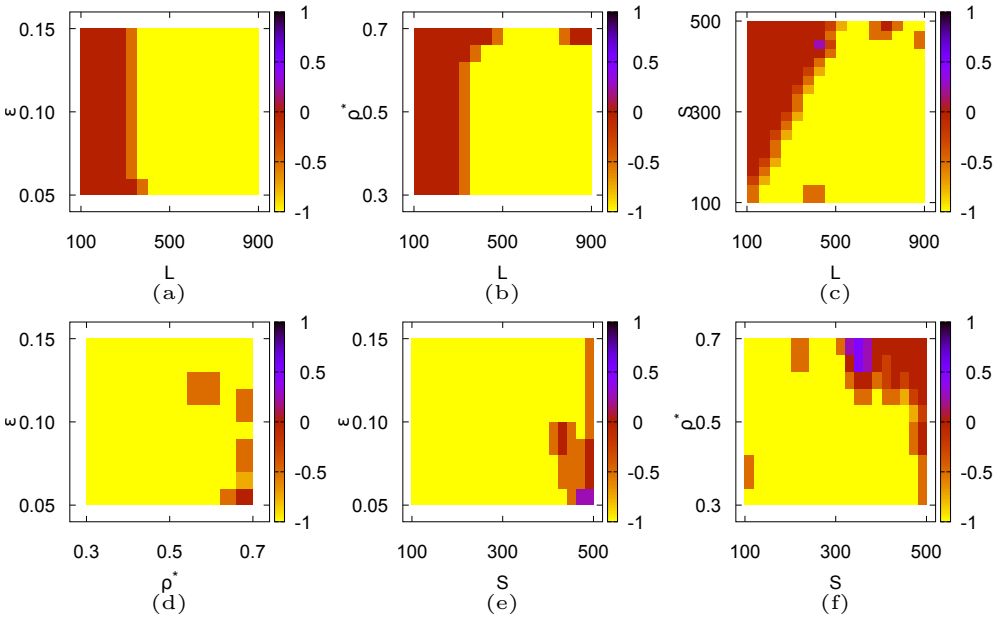

Figure 6: The change induced by the modified algorithm on $\Delta\rho$ for each possible hyper-parameter pair.

whereas the disparity in the RBSC coefficient degrades slightly.

Acknowledgements

The codes developed for this study can be downloaded freely from [20].

References

[1] L. J. Hong, W. Fan, and J. Luo, "Review on ranking and selection: A new perspective," *arXiv preprint arXiv:2008.00249*, 2020.

[2] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A simulation study of the model evaluation criterion MMRE," *IEEE Tran. Software Engineering*, vol. 29, no. 11, pp. 985–995, 2003.

[3] D. J. Eckman, M. Plumlee, and B. L. Nelson, "Revisiting subset selection," in *Proc. Winter Simulation Conf.*, pp. 2972–2983, IEEE, 2020.

[4] Z. Yücel, P. Supitayakul, A. Monden, and P. Leelaprute, "An algorithm for automatic collation of vocabulary decks based on word frequency," *IEICE Tran. Information and Systems*, vol. 103, no. 8, pp. 1865–1874, 2020.

[5] S. S. Gupta, "On some multiple decision (selection and ranking) rules," *Technometrics*, vol. 7, no. 2, pp. 225–245, 1965.

[6] L. Huang, J. Wei, and E. Celis, "Towards just, fair and interpretable methods for judicial subset selection," in *Proc. AAAI/ACM Conf. AI, Ethics, and Society*, pp. 293–299, 2020.

[7] G. C. McDonald, "Applications of subset selection procedures and Bayesian ranking methods in analysis of traffic fatality data," *Computational Statistics*, vol. 8, no. 6, pp. 222–237, 2016.

[8] C.-H. Yeh, B. A. Barsky, and M. Ouhyoung, "Personalized photograph ranking and selection system considering positive and negative user feedback," *ACM Tran. Multimedia Computing, Communications, and Applications*, vol. 10, no. 4, pp. 1–20, 2014.

[9] C.-H. Chen, S. E. Chick, L. H. Lee, and N. A. Pujowidianto, "Ranking and selection: Efficient simulation budget allocation," *Handbook of Simulation Optimization*, pp. 45–80, 2015.

[10] S. H. Choi and T. G. Kim, "A heuristic approach for selecting best-subset including ranking within the subset," *IEEE Tran. Systems, Man, and Cybernetics-A*, vol. 50, no. 10, pp. 3852–3862, 2018.

[11] M. H. Alrefaei and M. Almomani, "Subset selection of best simulated systems," *Journal of the Franklin Institute*, vol. 344, no. 5, pp. 495–506, 2007.

[12] Y. Wang, L. Luangkesorn, and L. J. Shuman, "Best-subset selection procedure," in *Proc. Winter Simulation Conf.*, pp. 4310–4318, IEEE, 2011.

[13] S. Gao, H. Xiao, E. Zhou, and W. Chen, "Robust ranking and selection with optimal computing budget allocation," *Automatica*, vol. 81, pp. 30–36, 2017.

[14] M. Mitchell, D. Baker, N. Moorosi, E. Denton, B. Hutchinson, A. Hanna, T. Gebru, and J. Morgenstern, "Diversity and inclusion metrics in subset selection," in *Proc. AAAI/ACM Conf. AI, Ethics, and Society*, pp. 117–123, 2020.

[15] S. Gao and W. Chen, "A note on the subset selection for simulation optimization," in *Proc. Winter Simulation Conf.*, pp. 3768–3776, IEEE, 2015.

[16] C.-H. Chen, D. He, M. Fu, and L. H. Lee, "Efficient simulation budget allocation for selecting an optimal subset," *INFORMS Journal on Computing*, vol. 20, no. 4, pp. 579–595, 2008.

[17] M. J. Groves, *Efficient Pairwise Information Collection for Subset Selection*. PhD thesis, University of Warwick, 2020.

[18] L. J. Hong, J. Luo, and Y. Zhong, "Speeding up pairwise comparisons for large scale ranking and selection," in *Proc. Winter Simulation Conf.*, pp. 749–757, IEEE, 2016.

[19] D. S. Kerby, "The simple difference formula: An approach to teaching nonparametric correlation," *Comprehensive Psychology*, vol. 3, pp. 11–IT, 2014.

[20] Z. Yücel, "Solving the subset generation problem with RBSC-SubGen." `https://github.com/yucelzeynep/Subset-generation-with-RBSC-SubGen`, 2021. [Accessed 2022-03-18].