Journal of Mathematical Extension
Vol. 18, No. 8, (2024) (6)1-19
URL: https://doi.org/10.30495/JME.2024.2870
ISSN: 1735-8299
Original Research Paper

Cross-Efficiency Evaluation Considering Big Data Set; A Numerical Approach

F. Moradi
Sanandaj Branch, Islamic Azad University

A. Ghomashi
Kermanshah Branch, Islamic Azad University

S. Shahghobadi*
Kermanshah Branch, Islamic Azad University

Abstract. Big Data Envelopment Analysis (Big DEA) and Cross-Efficiency (CE) evaluation are two important areas of study in Data Envelopment Analysis (DEA). However, when dealing with large sets of data, the computational demands of CE evaluation can be quite high. The evaluation of cross-efficiency for big data sets has been overlooked by researchers despite the considerable attention given to Big DEA. This research primarily focuses on calculating cross-efficiency in Big DEA using a numerical approach that eliminates the need for complex optimization models. This method is not only efficient in terms of runtime, but also applicable to a wide range of data sets. We propose a complementary method to identify efficient DMUs, a crucial aspect of most Big DEA algorithms. The algorithms were rigorously tested on multiple simulated datasets under different scenarios, successfully achieving both cross-evaluation and identification of the set of efficient units. These findings have significant implications for the field of Big DEA.

AMS Subject Classification: 90Bxx; 90C60; 91B08
Keywords and Phrases: Data Envelopment Analysis, Cross-Efficiency Analysis, Big DEA

Received: September 2023; Accepted: August 2024
*Corresponding Author

1 Introduction

Data envelopment analysis (DEA) is a method for performance evaluation of Decision-Making units (DMUs), such as businesses, organizations, or departments which proposed by Charnes et al. [**?**] for the first time. The aim of the DEA is to assess the relative efficiency of DMUs by considering multiple inputs and outputs. DEA has a wide range of applications, including performance evaluation in production and service industries, healthcare, education, and public administration, among others [**?**, **?**, **?**, **?**].

Big DEA and cross-efficiency (CE) evaluation are two important topics in DEA.It emerged with the growth of big data. The field has acquired attention because of the rise in large and complex data sets. Recent advancements in Big DEA involve the integration of machine learning techniques, network DEA, humanitarian supply chain, and a common set of weights (CSW). Big DEA research can be divided into two categories. The first group has developed an efficient numerical algorithm assuming a data set with much DMUs such as [**?**, **?**, **?**, **?**] . The second group includes those researches that have used existing algorithms to solve real problems with big data. Ding et al.[**?**] proposed a new algorithm called the Parallel DEA-DW algorithm. They integrated DEA and the Dantzig-Wolfe (DW) decomposition algorithm and propose a parallel DEA-DW algorithm to facilitate the computing of efficiency scores. Zhou et al.[**?**] proposed a new algorithm by modifying the two fastest approaches (build hull approach and pre-score approach) to make them capable of dealing with undesirable output in DEA. Khezrimotlagh [**?**] looked at how parallel processing affects the application of a DEA model to a large data set. It compared existing methods based on their cardinality, dimension, and density, and found that a new method combining two existing methods was faster than all the others, regardless of the cardinality, dimension, and densities. Ma et al.[**?**] introduced a novel parallel framework algorithm to solve large-scale DEA models. The proposed algorithm used a hybrid parallel framework that combined the hierarchical decomposition (HD) and parallel processing (PP) methods. The HD method was used to divide large-scale problems into smaller subproblems that could be solved in parallel. The PP method was used

to solve the subproblems in parallel using multiple processors or cores. The algorithm was shown with a dataset of 1000 DMUs.

The cross-efficiency (CE) evaluation was introduced by Sexton (1986) and later developed by Doyle and Green [**?**] and Green et al.[**?**]. The process involves calculating the maximum relative efficiency for each DMU and determining an optimal set of weights. These weights are then used to compute the efficiency scores (CE scores) for the other units. The final CE score for each DMU is obtained by averaging its CE scores. In terms of discrimination power, the CE scores outperform the DEA efficiency scores. However, the CE evaluation can be computationally intensive, particularly when dealing with large data sets. Additionally, the choice of a weighting scheme can significantly impact the results.

This paper introduces a method for evaluating the CE for big data sets. The method uses a numerical approach, eliminating the need for complex optimization models. It is efficient in terms of runtime and can apply to a wide range of data sets. Moreover, it possesses the ability to identify a significant portion of efficient DMUs, which is vital for Big DEA. Tests were conducted using simulated data sets, showing the effectiveness of both the primary and supplementary methods in identifying the percentage of effective units in different scenarios. Experiments were also conducted to assess the proposed method's efficiency. The proposed method calculated the CE in a very economical time. Also, it could identify the set of efficient units for data sets with dimension less than or equal to 5 of any size. To enhance this feature, a complementary method is also introduced to fully identify efficient DMUs.

The rest of this paper is organized in the following manner. Section 2 covers fundamental notations, starting with DEA and then CE evaluation. In Section 3, we delve into our approach, examining its accuracy and validation. Section 4 presents a method to identify efficient DMUs not found using the main approach. The paper concludes in Section 5.

2 Fundamental Notations

2.1 DEA

Let there are a set of n DMUs that each one use m inputs to produce s outputs, where x_{ij} and y_{rj} denote the ith input and rth output of DMU$_j$, respectively, for $j \in \mathcal{J} = \{1, 2, ..., n\}$, $i \in \mathcal{I} = \{1, 2, .., m\}$ and $o \in \mathcal{O} = \{1, 2, ..., s\}$. It is assumed that x_{ij} and y_{rj} are positive or zero, and neither vector has all elements zero. The set of Production Possibility Set (PPS) is defined by

$$T = \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{m+s} | x_i \geq \sum_{j \in \mathcal{J}} \lambda_j x_{ij},\ y_r \leq \sum_{j \in \mathcal{J}} \lambda_j y_{rj}\}$$

If $(\mathbf{u}, \mathbf{v}) \in \mathbb{R}_+^{s+m}$, the efficiency and relative efficiency of DMUj, with respect to this profile of weights, is defined by $E_j(\mathbf{u}, \mathbf{v}) = \frac{\sum_{r \in \mathcal{O}} u_r y_{rj}}{\sum_{i \in \mathcal{I}} v_i x_{ij}}$. and $RE_j(\mathbf{u}, \mathbf{v}) = \frac{E_j(\mathbf{u}, \mathbf{v})}{Max_{k \in J} E_k(\mathbf{u}, \mathbf{v})}$, respectively. The maximum value of relative efficiency of DMU$_j$, called DEA efficiency, is obtained using the multiplier form of CCR model as follows, [**?**]:

$$\begin{aligned} \theta_k^* \quad = \quad & Max \sum_{j \in \mathcal{J}} u_r y_{rk} \\ s.t \quad & \sum_{i \in \mathcal{I}} v_i x_{ik} = 1 \\ & \sum_{r \in O} u_r y_{rj} - \sum_{i \in I} v_i x_{ij} \leq 0, \ \text{for}\, j \in \mathcal{J} \\ & u_r \geq \epsilon, \ v_i \geq \epsilon \qquad \forall\, i \in \mathcal{I}\, \text{and}\, r \in \mathcal{O} \end{aligned} \tag{1}$$

where ϵ is a very small positive number called a *non-archimedean* number. [**?**] has suggested a way to set a value for it. The dual problem of model (1), which is called the envelopment form of CCR model, is as

follows:

$$\begin{aligned} \theta_k^* \quad = \quad & Min\,\theta - \epsilon(\sum_{i\in\mathcal{I}} s_i^- + \sum_{r\in\mathcal{I}} s_r^+) \\ s.t \quad & \sum_{j\in\mathcal{J}} \lambda_j x_{ij} + s_i^- = \theta x_{io} \quad i \in \mathcal{I} \\ & \sum_{j\in\mathcal{J}} \lambda_j y_{rj} - s_r^+ = y_{ro} \quad r \in \mathcal{O} \\ & \lambda_j \geq 0, s_i^-, s_r^+ \geq 0 \,\forall\, (j \in \mathcal{J},\, i \in \mathcal{I},\, r \in \mathcal{O}) \end{aligned} \tag{2}$$

Definition 2.1. (CCR-efficiency, Technical efficiency)
Let $(\theta^*, \lambda^*, \mathbf{s}^{-*}, \mathbf{s}^{+*})$ and $(\mathbf{u}^*, \mathbf{v}^*)$ are optimal solutions of models (2) and (1) for DMU_k, respectively :

1. If θ^*=1, $\mathbf{s}^{-*} = 0$ and $\mathbf{s}^{+*} = 0$ or $\mathbf{u}^* > 0$ & $\mathbf{v}^* > 0$ with $\beta^* = 1$, then DMU_k is CCR-efficient
2. if θ^* =1, then DMU_k is technical efficient.
3. If $\theta^* < 1$, DMU_k is technical inefficient.

Note that, we have from Complementary slackness theorem, see Vanderbei et al. [**?**], that $\lambda_j^* = 0$ for inefficient DMUs. Therefore, you can drop the index of inefficient DMUs from set $\mathcal{J}$ in (2) , without affecting the optimal solution. This feature is used in big DEA algorithms.

2.2 CE Evaluation

Let $(\mathbf{u}_k^*, \mathbf{v}_k^*)$ is a set of favorite weights for DMU_k, $k \in \mathcal{J}$ obtained from (1) . The steps of the CE evaluation are as below:

Input data:$\{(\mathbf{x}_j, \mathbf{y}_j))\}_{j=1}^n$ and $N \in \mathbb{N}$ Output result: $\{\bar{e}_k\}_{k=1}^N$

0 *Step 0:* Set $S := [s_{k,j}]_{N\times n}$

Step 1: Run the model (1) to obtain $(\mathbf{u}_k^*, \mathbf{v}_k^*)$ for each DMU_k, $k \in \mathcal{J}$

Step 2: Construct CE matrix: $CE = \left[e_{kj}(\mathbf{u}_k^*, \mathbf{v}_k^*) = \dfrac{\sum_{r\in\mathcal{O}} u_{kr}^* y_{rj}}{\sum_{i\in\mathcal{I}} v_{ki}^* x_{ij}}\right]_{n\times n}$

Step 3: Calculate the finall CE scores: $\bar{e}_j = \frac{1}{n}\sum_{k\in\mathcal{J}} e_{kj}(\mathbf{u}_k^*, \mathbf{v}_k^*)$ for $j \in \mathcal{J}$.

The non-uniqueness solution of the Model (1) is the main disadvantage of this method, making the output unstable and reducing its appeal considerably. To address this issue,Doyle and Green [**?**] utilized secondary goal models *aggressive* and *benevolent* formulations, respectively:

$$
\begin{aligned}
& Max \sum_{r\in\mathcal{O}} u_r\Big(\sum_{j\in\mathcal{J},j\neq k} y_{rk}\Big) \\
s.t \quad & \sum_{i\in\mathcal{I}} v_i\Big(\sum_{j\in\mathcal{J},j\neq k} x_{ik}\Big) = 1 \\
& \sum_{r\in O} u_r y_{rk} - \theta_k^* \sum_{i\in I} v_i x_{ij} = 0 \\
& \sum_{r\in O} u_r y_{rj} - \sum_{i\in I} v_i x_{ij} \leq 0, \ \text{for}\, j \in \mathcal{J} \\
& u_r \geq 0, \ v_i \geq 0 \qquad \text{for all}\, i \in \mathcal{I}\, \text{and}\, r \in \mathcal{O}
\end{aligned}
$$

and

$$
\begin{aligned}
& Min \sum_{r\in\mathcal{O}} u_r\Big(\sum_{j\in\mathcal{J},j\neq k} y_{rk}\Big) \\
s.t \quad & \sum_{i\in\mathcal{I}} v_i\Big(\sum_{j\in\mathcal{J},j\neq k} x_{ik}\Big) = 1 \\
& \sum_{r\in O} u_r y_{rk} - \theta_k^* \sum_{i\in I} v_i x_{ij} = 0 \\
& \sum_{r\in O} u_r y_{rj} - \sum_{i\in I} v_i x_{ij} \leq 0, \ \text{for}\, j \in \mathcal{J} \\
& u_r \geq 0, \ v_i \geq 0 \qquad \text{for all}\, i \in \mathcal{I}\, \text{and}\, r \in \mathcal{O}
\end{aligned}
$$

Various researchers followed and developed this line of research [**?**, **?**, **?**, **?**, **?**]. The final CE scores are affected by the strategy employed in secondary models, which limit the set of optimal solutions for Model (1). Therefore, the combination of CE approach with a secondary model cannot rank the units in an unbiased and reliable way.

3 The Proposed Approach

The optimal solutions of the multiplicative model play an essential role in the CE evaluation. It can be easily seen that each optimal solution of (1) is a normal vector of a supporting hyperplane of PPS, in short a normal vector, containing a technically efficient DMU. In this method, a sample, $\mathcal{H}$, containing n observations of these normal vectors, is always used.

3.1 Driving normal vectors

With the same goal in mind, we generate a set of normal vectors without the necessity of solving optimization models. The production process does not cause any primary assumption or secondary objective. Initially, we will show converting a set of non-negative weights into a normal vector.

Theorem 3.1. *For each* $(\boldsymbol{u}, \boldsymbol{v})$ *in* $\mathbb{R}_+^{s+m}$, $(\boldsymbol{u}, A\boldsymbol{v})$ *is an optimal solution of Model (1) for at least one* DMU_j *in* $\{DMU_j\}_{j\in\mathcal{J}}$, *where* $A = Max_{j\in\mathcal{J}}\{E_j(\boldsymbol{u}, \boldsymbol{v})\}$

Proof. Let $(\mathbf{u}, \mathbf{v}) \in \mathbb{R}_+^{s+m}$ is a set of inputs-outputs weights, and $A = Max_{j\in J}\{E_j(\mathbf{u}, \mathbf{v})\}$. For each $j \in J$,

$$\begin{aligned} E_j(\mathbf{u}, A\mathbf{v}) &= \frac{1}{A}E_j(\mathbf{u}, \mathbf{v}) \\ &\leq 1 \end{aligned}$$

For $k \in \text{Argmax}_{j\in J}\{E_j(\mathbf{u}, \mathbf{v})\}$, we have $(\mathbf{u}, A\mathbf{v})$ is an feasible solution of model (1) for DMU_k. That $E_j(\mathbf{u}, A\mathbf{v}) = 1$ signifies it as an optimal solution and completes the proof. □ By randomly generating a set of positive weights, it is possible to create a normal vector. This process can be repeated to obtain a sample of normal vectors with a desired size of N. In terms of calculation cost and bias in sample formation, this method is much better than the CE method.

Input data:$\{(\mathbf{x}_j, \mathbf{y}_j))\}_{j=1}^n$ and $N \in \mathbb{N}$ Output result: $\mathcal{H}$

0 *Step 0:* Set N as the number of iterations. For $k = 1$ to N:

Step 1: Generate$(\boldsymbol{u}_k, \boldsymbol{v}_k) \in \mathbb{R}_+^{s+m}$ *randomly, and calculate* $E_j(\boldsymbol{u}_k, \boldsymbol{v}_k)$ *for each* $j \in \mathcal{J}$

Step 2: Calculate $A_k = Max_{j\in\mathcal{J}}\{E_j(\boldsymbol{u}_k, \boldsymbol{v}_k)\}$

Step 3: Calculate $CE[k,:] := [E_1(\boldsymbol{u}_k, \frac{1}{A_k}\boldsymbol{v}_k)), ..., E_n(\boldsymbol{u}_k, \frac{1}{A_k}\boldsymbol{v}_k))]$ *Step 4: Calculate* $\bar{e}_j := \frac{1}{N}\sum_{k=1}^{N} CE[k,j]$ *for* $j \in \mathcal{J}$

3.2 Efficient DMUs identification

The maximum of CE scores got for each DMU can approximate the CCR efficiency score of that unit. Obviously, the higher the number of these scores, N, the better the accuracy of this approximation.

Definition 3.2. Let $CE[:,j]$ is the j_{th} column of CE matrix of the proposed approach. For each $j \in \mathcal{J}$ define:

$$\hat{\theta}_j := Max_{k=1\,\texttt{to}\,N} CE[k,j]$$

.

We claim that $\hat{\theta}_j$ can estimate θ_j. Multiple error measures exist. The maximum absolute error (MAXE), that measures the largest difference between predicted and true values.. It shows the biggest gap between predicted and actual values. Mean Squared Error (MSE), which is a common metric used to measure the average squared difference between the predicted values and the true values in a dataset or model. It quantifies how accurately the model's predictions match the actual values. In order to determine the accuracy of this approximation, we employ MSE and MAXE measures:

$$MSE(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{n}\sum_{j=1}^{n}(\hat{y}_j - y_j)^2$$

$$MAXE(\hat{\mathbf{y}}, \mathbf{y}) = \max_{j=1}^{n}|\hat{y}_j - y_j)|$$

If $\hat{\theta}_j = 1$, then $\theta_j = 1$. So, $\bar{E} := \{DMU_j | \hat{\theta}_j = 1\}$ is a subset of CCR

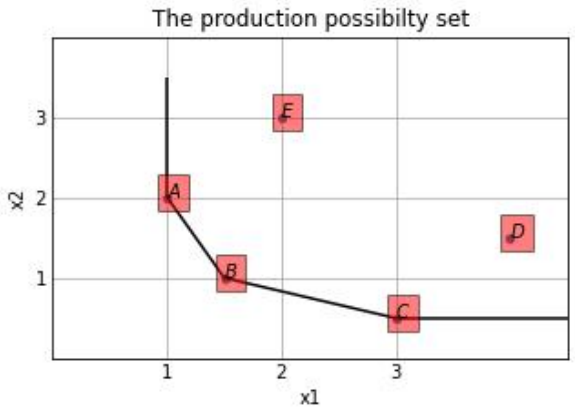

Figure 1: Example 1 data set's PPS

efficient DMUs. As a result, Algorithm 2 produces $\bar{E}$, which is crucial in Big DEA. It uses model (2), which has fewer constraints. Moreover, the reference set, $\mathcal{J}$, is confined to efficient DMUs to decrease variables. Complex numerical methods have been proposed to obtain the set of efficient units. See [**?**, **?**, **?**, **?**]. Therefore, having the set of efficient DMUs is significant in Big DEA.

To see more details about the proposed method, consider the following intuitive example.

Example 3.3. Consider a set of five DMUs with input and output matrices: $X = \begin{bmatrix} 1 & 1.5 & 3 & 2 & 4 \\ 2 & 1 & 0.5 & 3 & 1.5 \end{bmatrix}$ and $Y = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}$

The PPS is depicted in Figure 1. We set $N = 1000$ and ran the Algorithm (3.1). The weights were generated by squaring a normal distribution that had a mean of 0 and a standard deviation of 0.33. Definition (3.2) was used to determine the $\hat{\theta}_j$ value for each unit. Because there were so many CE scores, the results were illustrated using a violin plot in Figure 2. A violin plot is a data visualization technique that combines aspects of a box plot and a kernel density plot. It is used to display the distribution of a continuous variable or numerical data across different

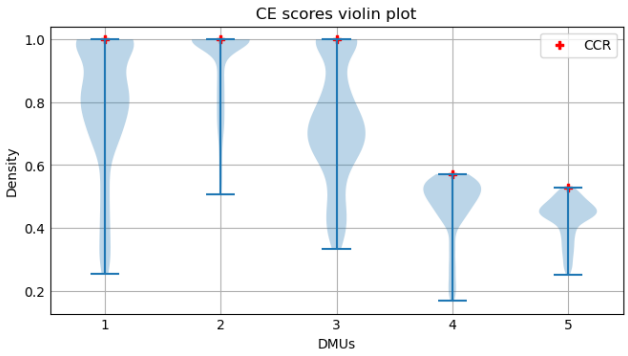

Figure 2: Violin plot of CE scores resulting from Algorithm 2

categories or groups [**?**]. In Figure (2), it can be observed that the maximum CE scores align with the CCR-efficiency scores of each unit. Figure 3, shows the final CE scores achieved using the classical method with aggressive and benevolent strategies, as well as the proposed method, along with the CCR-efficiency scores.

Different simulated data sets $(m, s) \in \{(1,1),(2,1),(2,2),(3,2),(3,3),(4,4)\}$ were employed to test the accuracy and capacity to identify efficient units. For each scenario, there were one hundred datasets with n values ranging from 25 to 100, x_{ij} values between 100 and 1000 and y_{rj} between 1000 and 10000 uniformly. The ratio $P = \dfrac{card(\{DMU_j | \hat{\theta}_j = 1\})}{card(\{DMU_j | \theta_j^* = 1\})}$ was employed to assess the method's capability in identifying efficient units. To calculating P for each generated data set, both the proposed method and Model (1) were implemented. The aim was to precisely determine the efficient units and compare them to the suggested approach's outcomes.

Descriptive statistics of indicators SME, MAXE, and P are shown in Table (1) according to different scenarios. Based on the findings for $m = 2$ and $n = 2$, $MSE = 0.000028$, $MAXE = 0.020644$, $P =$ is 100. It can be said that the approximation's accuracy was satisfactory. Additionally, it can identify approximately 97 % of the efficient DMUs on average. Figure 4 illustrates the inverse relationship between dataset dimension and

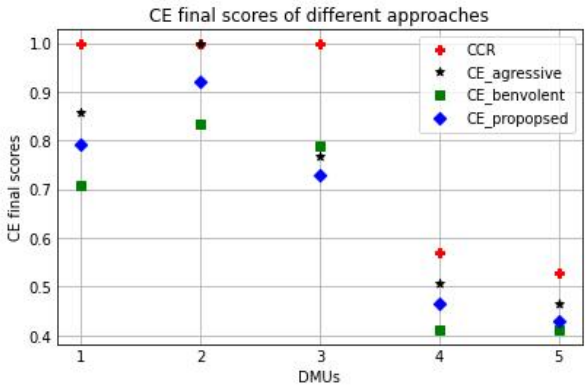

Figure 3: Comparison of CE scores between Algorithm 2 and Algorithm 1 with aggressive and benevolent strategies

the accuracy and capacity to identify efficient units. Another technique is proposed in the subsequent section to improve the ability to identify efficient DMUs.

4 An Approach to Fully Identify the Efficient DMUs

In DEA, as in almost all data-driven research strands, inference based on large data sets is also becoming increasingly important [**?**]. In large data sets containing a huge number of DMUs, we need to solve many large scale linear models, which will make the method computationally difficult. If the number of DMUs is greater than 100, then it is worth to apply big data methods [**?**]. As mentioned before, finding the set of efficient DMUs is the main key to most Big DEA methods. The main idea behind most proposed frameworks for partitioning DMUs into smaller groups is aimed at finding the set of efficient DMUs first, and after that calculating the performance scores of the inefficient DMUs [**?**].

Table 1: The accuracy test results for the proposed method

		m=1	s=1			m=2	s=1	
	n	SME	MAXE	P	n	SME	MAXE	P
mean	65	2.5271E-18	5.24E-09	100	64.5	1.8106E-08	0.000549	100
std	29.5	4.3536E-18	4.37E-09	0	26.6	4.4476E-08	6.09E-04	0
min	25	4.1598E-22	7.17E-11	100	25	7.8107E-13	4.00E-06	100
25%	25	5.9195E-20	9.05E-10	100	50	9.7159E-10	1.39E-04	100
50%	75	6.2320E-19	4.22E-09	100	75	4.8359E-09	3.89E-04	100
75%	100	2.7827E-18	8.51E-09	100	81.25	1.4315E-08	7.56E-04	100
max	100	2.5913E-17	1.52E-08	100	100	2.6884E-07	3.84E-03	100

		m=2	s=2			m=3	s=2	
	n	SME	MAXE	P	n	SME	MAXE	P
mean	60.75	0.000028	0.020644	97.82	60.5	7.0400E-04	1.06E-01	90.54
std	26.1	3.3000E-05	1.28E-02	6.3	29.5	6.9300E-04	5.80E-02	9.3
min	25	3.0000E-06	3.30E-03	67	25	2.3000E-05	1.22E-02	67
25%	50	9.0000E-06	1.11E-02	100	25	2.9500E-04	6.80E-02	83
50%	50	1.8000E-05	1.78E-02	100	50	5.7400E-04	1.02E-01	91
75%	75	3.2000E-05	2.59E-02	100	100	8.1600E-04	1.31E-01	100
max	100	2.2000E-04	6.27E-02	100	100	4.6570E-03	3.75E-01	100

		m=3	s=3			m=4	s=4	
	n	SME	MAXE	P	n	SME	MAXE	P
mean	64.25	2.1980E-03	1.67E-01	79.81	63	1.0075E-02	2.91E-01	62.28
std	26.8	1.2820E-03	5.61E-02	14.06	27.6	4.9210E-03	7.80E-02	12.3
min	25	4.7600E-04	5.22E-02	22	25	2.3720E-03	1.32E-01	26
25%	50	1.2970E-03	1.26E-01	71	50	6.2270E-03	2.29E-01	54
50%	75	1.9020E-03	1.65E-01	80	75	9.0520E-03	2.85E-01	63
75%	75	2.5690E-03	1.97E-01	91.25	75	1.2153E-02	3.51E-01	71
max	100	7.1350E-03	3.67E-01	100	100	2.8357E-02	4.56E-01	94

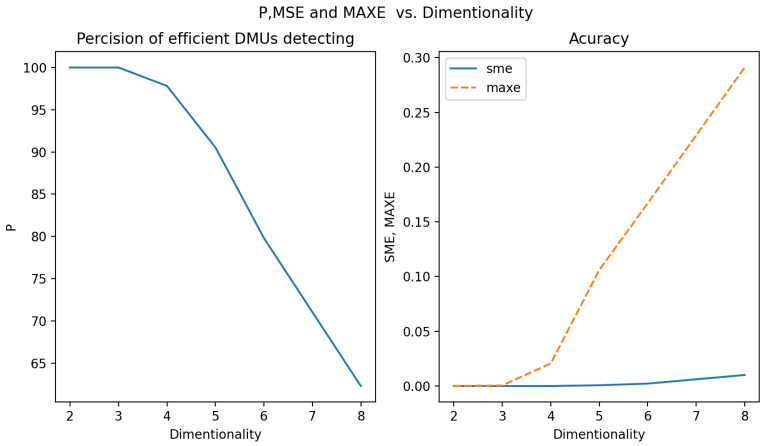

Figure 4: Variation of MAXE and MSE regarding the dimension of the dataset

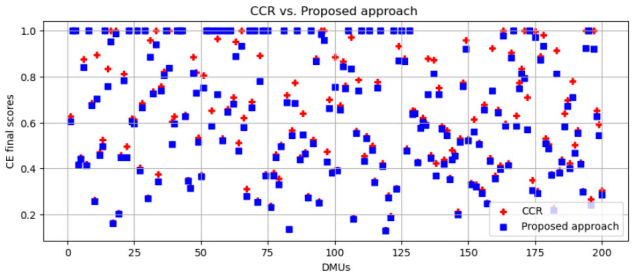

Figure 5: θ_j^* versus $\hat{\theta}_j^*$ for different DMUs

In the previous section, we saw that the proposed method can detect a significant part of efficient units depending on the of the dimension of data. By performing a few simple steps, this set can be extended to include all efficient units. For this reason, we focus on the set of DMUs that have a higher chance of being efficient, as $\mathcal{J}_1 := \{j|\hat{\theta}_j \geq \gamma\}$, where $\gamma \in [0.8, 1)$. Even though you can extend the range for gamma, this would cause more computations and increased scrutiny of candidates with a very slim likelihood of being efficient. Next, for each DMU_j that $j \in \mathcal{J}$, run the following modified CCR model for $j \in \mathcal{J}_1 \setminus \mathcal{J}_0$, where in which the main reference set, $\mathcal{J}$, is replaced by $\mathcal{J}_0$:

$$\begin{aligned} \alpha^*_{j_0} \quad = \quad & Min\, \theta - \epsilon(\sum_{i\in\mathcal{I}} s_i^- + \sum_{r\in\mathcal{I}} s_r^+) \\ & \sum_{j\in\mathcal{J}_0} \lambda_j x_{ij} + s_i^- = \theta x_{io} \quad i \in \mathcal{I} \\ & \sum_{j\in\mathcal{J}_0} \lambda_j y_{rj} - s_r^+ = y_{ro} \quad r \in \mathcal{O} \\ & \lambda_j \geq 0, s_i^-, s_r^+ \geq 0 \,\forall\, (j \in \mathcal{J}_0,\, i \in \mathcal{I},\, r \in \mathcal{O}) \end{aligned} \tag{3}$$

[**?**] showed that the modified CCR model is infeasible if and only if certain pattern of zero inputs/outputs is involved. However, one should note that this type of data is seldom to occur in a real-world situation.

If $\alpha^*_{j_0} < 1$ is established, it means that DMU_j has been dominated by a positive combination of DMUs in the current reference set and is CCR-inefficient. So, we will omit j from $\mathcal{J}_1$ and update it. It is easy to see that $\mathcal{J}_1^c$ contains the index of all inefficient DMUs. Hence, we limit the reference set to $\mathcal{J}_1$. Although this set may be larger than $\mathcal{J}_E := \{j|\theta^*_j = 1\}$, it is much smaller than $\mathcal{J}$. In addition, to reduce the set size of $\mathcal{J}_1$, you can use a bigger γ. Finally, the new collection of efficient DMUs candidates, denoted as $\mathcal{J}_1$, will be further refined. The above process is repeated by using the Model (3) to assess the performance of each individual and then adjust $\mathcal{J}_1$. A summary of these steps is provided:

Input data:$\mathcal{J}_0 \leftarrow \{j|\hat{\theta}_j = 1\}$ and $\mathcal{J}_1 \leftarrow \{j|\hat{\theta}_j \geq 1\}$ Output result: $\mathcal{J}_0$

0 *Step 0:* For $j \in \mathcal{J}_1 \setminus \mathcal{J}_0$ run Model 3 with $\mathcal{J}_0$ as deference set. If $\alpha_j^* < 1 \longmapsto$ remove j from $\mathcal{J}_1$

Step 1: For all $j \in \mathcal{J}_1 \setminus \mathcal{J}_0$ run again Model (3) with $\mathcal{J}_1$ as deference set. If $\alpha_j^* < 1 \longmapsto$ remove j from $\mathcal{J}_1$

In Big data DEA, $\mathcal{J}_E$, is often much smaller than the reference set. By considering $card(\mathcal{J}_0) \leq card(\mathcal{J}_E)$ and $card(\mathcal{J}_1 \setminus \mathcal{J}_0) << card(\mathcal{J})$, this algorithm can not be consuming-time. The following example is given for further explanation.

Example 4.1. A data set with $n = 200$, $m = 5$ and $s = 5$. Data have drawn from uniform distribution within $[100, 1000]$ for inputs and $[1000, 10000]$ for output were generated.

At first, we determine $\mathcal{J}_E$ and $\mathcal{J}_0$ using Model (1) and Algorithm (3.1), respectively. The summary of the results is displayed in Table 2. The index set of CCR-efficient DMUs which had not been detected by the Proposed algorithm was:

$$\mathcal{J}_E \setminus \mathcal{J}_0$$

:= {10, 12, 28, 37, 48, 60, 65, 68, 88, 98, 131, 190, 209, 212, 235, 244, 277} Then, we set $\gamma = 0.85$ and got the index set of units that have a chance to be efficient:

$$\mathcal{J}_1 \setminus \mathcal{J}_0 := \{10, 12, 22, 23, 28, 36, 37, 44, 48, 57, 60, 65, 68, 88, 96, 98, 116, 131, 139, 172, 180, 190, 194, 204, 209, 212, 235, 237, 244, 250, 269, 270, 277, 286\}$$

Table 2: The main result of Algorithms 2 and Model 1

SME	MRE	card($\mathcal{J}_E$)	card($\mathcal{J}_0$)	Percision
0.001409	0.136001	94.0	77.0	82.0

Next, according to the Step 1, Model (3) was implemented for $\mathcal{J}_1 \setminus \mathcal{J}_0$, and $\mathcal{J}_1$ was updated:

$$\mathcal{J}_1 \setminus \mathcal{J}_0$$

={10, 12, 22, 28, 36, 37, 48, 60, 65, 68, 88, 98, 131, 190, 209, 212, 235, 244, 277 }

Because $\mathcal{J}_0$ is a subset of efficient DMUs set, there may still be indexes in the above list that corresponding DMUs are inefficient. Therefore, in the second step, their efficiency was checked once again by Model (3) with the larger reference set $\mathcal{J}_1$. The obtained results were $[1.0, 1.0, 0.993, 1.0, 0.997, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]$

It was revealed that both DMU_{22} and DMU_{36} were inefficient. By removing the indices of these two units from $\mathcal{J}_1$, we will have $\mathcal{J}_1 = \mathcal{J}_E$.

4.1 Assessing the execution time of algorithm (4)

4.1.1 Computer and software

We used a computer with Processor: Intel(R) Core(TM) i3-4150 CPU @ 3.50GHz 3.50 GHz, RAM: 8.00 GB and a 64 bit Windows 10 Pro. This was also the main limitation of this research. Python 3.10.0 with Scipy and Numpy modules were used to write the codes.

4.1.2 Data

Random data with a uniform distribution were generated across 24 different scenarios :

$(m, s) \in \mathfrak{D} = \{(m, s) | \, (m, s) = (1, 1), (1, 2), (2, 2), (4, 2), (3, 3), (5, 5), (7, 8)\}$

and

$$n \in \mathfrak{M} = \{2000, 10000, 15000, 30000\}$$

For each $(m, s, n) \in \mathfrak{D} \times \mathfrak{M}$, Algorithm (4) was employed and the results are shown in Table 3. Due to the mentioned limitation, each scenario was only run once. It was run with $N = 100000$ for $n < 30000$, and with $N = 10000$ for $n = 30000$.

The runtime of algorithm (3.1) and algorithm (3.1) were separately calculated and displayed in columns T_1 and T_2 per *second*. Also, the number of CCR-efficient units identified using the algorithm (3.1), $k_1 =$

Table 3: Final results

	$m=1,s=1$					$m=1,s=2$				
$DMUs$	k_1	$T1$	k_2	$T2$	P	k_1	$T1$	k_2	$T2$	P
n=2000	1	0.72	0	0.62	100	3	0.86	0	2.33	100
n=10000	1	3.94	0	0.69	100	6	3.59	0	1.74	100
n=15000	1	6.23	0	0.77	100	3	4.81	0	9.03	100
n=30000	1	14.18	0	17.21	100	3	13.26	0	4.70	100
	$m=2,s=2$					$m=4,s=3$				
n=2000	7	7.67	0	9.68	100	83	0.75	28	44.20	66
n=10000	20	4.42	0	8.15	100	131	3.71	59	128.23	55
n=15000	18	4.78	0	17.52	100	127	5.91	65	170.72	49
n=30000	20	10.29	0	28.60	100	132	14.81	75	261.00	43
	$m=3,s=2$					$m=5,s=5$				
n=2000	26	5.55	0	8.63	100	166	9.68	63	72.28	62
n=10000	63	32.01	0	27.15	100	324	27.15	253	432.61	22
n=15000	59	21.51	0	35.71	100	400	250.08	307	344.84	23
n=30000	68	66.36	1	89.60	99	360	44.90	493	375.91	-37

$card(\mathcal{J}_0)$, and the algorithm (4) , $k_2 = card(\mathcal{J}_1 \setminus \mathcal{J}_0)$, have been provided. The last column is derived by subtracting the ratio of $K2$ to $K1$ from 1, and then multiplying by 100. This column shows the power of Algorithm 1 in identifying CCR-efficient DMUs based on the data set's dimensions.

4.1.3 Discussion

The results show that the Proposed algorithm has been 100% successful in identifying CCR-efficient units for data sets with low dimensions and with any number of DMUs. Also, the execution time as a key feature in the literature of big data DEA has been significantly low. It is expected

that, in case of using a computer with better technical specifications and considering larger values for N in Algorithm 1, more favorable results will be got. As seen from the results in Table 2, the proposed method for data sets with a dimension of 5 or less could accurately identify the set of efficient units. For larger dimensions, this ability decreased sharply, whereas the number of CCR-efficient units detected increased sharply compared to the complementary method. The execution time of these two algorithms has been proportional to the number of data set units and is acceptable. It should be noted that these two proposed methods together identify the set of CCR-efficient units that are used in big DEA to calculate the relative efficiency of DMUs. However, if the goal is to evaluate the CE, the proposed algorithm can do this in a reasonable amount of time alone.

5 Conclusion

This study introduced a novel approach for calculating cross-efficiency in Big DEA. The method had low computational cost and offered major benefits compared to the current approaches. One notable benefit was its ability to apply to datasets of any size. Unlike existing approaches, it generated the normal vectors of supporting hyperplanes based on the production possibility set without solving optimization models. Moreover, the normal vectors were obtained without bias and could be seized by the user. This led to greater stability and reliability in the final outcomes compared to conventional cross-validation methods.

Another advantage of this approach was its capability to identify a substantial portion of efficient units, particularly for datasets with dimensions less than 5. However, to fully identify the efficient units, a supplementary algorithm was proposed. The algorithms were rigorously tested on multiple simulated datasets under various scenarios, successfully achieving both cross-evaluation and identification of the set of efficient units. These findings hold a successful for the field of Big DEA.

Farhad Moradi
Assistant Professor of Mathematics
Department of Mathematics
Sanandaj Branch, Islamic Azad University
Sanandaj, Iran.
E-mail: fmoradi54@hotmail.com

Abbas Ghomashi
Assistant Professor of Mathematics
Department of Mathematics
Kermanshah Branch, Islamic Azad University
Kermanshah, Iran.
E-mail: abbasghomashi@yahoo.com

Saeid Shahghobadi
Assistant Professor of Mathematics
Department of Mathematics
Kermanshah Branch, Islamic Azad University
Kermanshah, Iran.
E-mail: s.shahghobadi@gmail.com