Effective time-aware knowledge compression For recommender systems

Hadis Ahmadian Yazdi
Islamic Azad University, Neyshabur
Seyyed Javad Seyyed Mahdavi (mahdavi@mshdiau.ac.ir)
Islamic Azad University, Mashhad
Maryam Kheirabadi
Islamic Azad University, Neyshabur

Research Article

Keywords: Distillation, Recommender Systems, Feature selection, Correlation Matrix, Dataset Compression

Posted Date: June 22nd, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1745995/v1

Abstract

Due to the big data and wide availability of various types of information, data compression plays a significant role in the current age. This task can be done in two aspects. First, it can be done by reducing the redundant features of each entity. Second, it can be employed on each record of the corresponded dataset. These techniques should maintain the crucial and useful information which presents a pivotal role in further process. This work presents an effective knowledge compression for recommender systems based on the attention mechanism. In this method, the data compression is performed in two feature and record levels. The technique is based on time windows and the activity of users. The result of this technique can be efficiently utilized for deep networks which the amount of data is one of the serious problems. The experimental results show that with the help of this technique not only does it reduce the amount of data and process time, but also it can reach acceptable and considerable accuracy in the training and testing phases of networks.

1. Introduction

Over the last two decades, data has increased dramatically in many scopes such as in business, scientific, and social media. Therefore, the storage and transmission of data are growing at an enormous rate [15]. In other words, with the rapid development of the Internet and communication channels, the information explosion has become a serious problem [12]. A large amount of data and the dimensionality of them has become a severe task for machine learning and data-mining systems [2, 5, 20]. As known, the dimension of data has direct effects on the performance of these algorithms, especially clustering, classification, regression, and time-series prediction. Moreover, a huge amount of data increases the cost of computation and storage pressure. Besides, there is valuable hidden knowledge in the Dataset that plays a crucial role in analyzing tasks.

Totally, the original data contains redundant/irrelevant features [21]; In detail, the outside features provide misleading information, which leads to a fall in the learning accuracy. For example, in K-Nearest Neighbor, the irrelevant features increase the distances between samples from the same class, which makes it more challenging to truly classify data [4,13].

To prevent losing useful information and improve the performance of learning systems, efficient mechanisms should be considered as prepossessing to extract and maintain the information [3, 8, 12]. Hence, to cope with the mentioned problem, an effective knowledge compression algorithm for compressing and fetching useful data is proposed in this work. Generally, there are two types of compression, including lossy and lossless. In lossy mode, which is employed in multimedia data, the initial data are different compared to those recovered from the decompressing phase. On the other hand, the original data are the same as the retrieved in lossless compression. It is hugely employed in documents and execution files [8]. Also, compression can be classified into two categories in terms of dimensionality reduction. Firstly, the data can be compressed in row levels; In the following, the feature selection or extraction can be applied to compress the dataset [6]. During the last decades, the researchers only concentrate on compression at the features level.

In the proposed method, first, a subset of data is selected based on rows. Afterward, relevant features based on the target concept are selected to compress data more. The feature selection techniques can decrease the dimension of data and also speed up the analysis process. In other words, it is used to simplify the training phase and improve the quality of feature sets. Overall, it can significantly shorten the running time and improve the accuracy [3, 10, 20].

1.1 Literature review

As mentioned before, the compression techniques can be classified into two categories as the feature (column) or record (row) [9, 11, 19] reduction. Unfortunately, in the last decades, most research focused on feature selection for reducing the volume of data and improving the performance of learning and mining algorithms; Generally, feature selection contains two main goals, including the number of features and the classification accuracy [21]. On the other hand, record reduction plays a significant role in decreasing the space and time order of the mentioned algorithm. In this section, a brief review of several feature-based schemes proposed in recent years is discussed. Moreover, the advantages and weaknesses of techniques are presented.

In [8], a novel scheme was proposed for database compression based on the data mining technique. In this work, the redundant data which exist in the transaction database were eliminated. In this way, the redundant data were replaced with the mean of compression rules. Totally, it illustrated how association rules could be fetched using mining; Also, the compression rules can be employed to generate a compressed database. The experimental results proved the efficiency of the proposed method compared to the Apriori-based techniques in terms of both compression ratio and running time.

In [16], a threshold-based scheme proposed for feature selection in high-dimensional data. In detail, 11 feature selection techniques for ranking attributes were presented by considering the strength of the relationship between attributes and classes. For this aim, each attribute is normalized and paired individually with the class. The method was compared by Naive Bayes and Support Vector Machine algorithms to learn from the training datasets. The results demonstrated the superiority of the technique compared to traditional standard filter-based feature selection.

In [18], a new scheme based on K-Nearest-Neighbor(KNN) was proposed to decrease the time complexity of the feature selection phase. In detail, a novel strategy is presented to evaluate the quality of candidate features with the help of KNN. The primary purpose of this work is to accelerate wrapper-based feature subset selection. The results demonstrated that the scheme could significantly increase the evaluation process and decrease the running time cost without degrading the accuracy.

In [17], the authors proposed a feature selection method based on Ant Colony Optimization (ACO) and Genetic Algorithm(GA). The method includes two main models visibility density and pheromone density models. In this approach, each feature is modeled as a binary bit, and each bit has two orientations for selecting and deselecting. In the experimental result, the scheme was compared with other evolutionary algorithms. The results prove the performance of the optimization algorithm for solving the feature selection problems.

In another work, [5], a novel feature selection scheme was proposed based on genetic programming. In detail, a permutation strategy was employed to select features for high-dimensional

symbolic regression. The regression result proved the method's efficiency compared to other traditional schemes by considering truly relevant features.

In [21], a novel unsupervised feature selection method was proposed based on Particle Swarm Optimization(PSO). In this way, two filter-based strategies were presented to speed up the convergence of the algorithm. The first filter was based on average mutual information, and the second one is based on feature redundancy. The first is employed to remove irrelevant and weakly relevant features; The second is applied to improve the exploitation capability of the swarm. The experimental results illustrated the effectiveness of classification accuracy by reducing the number of features.

In [6], for the first time, the ensemble feature selection is modeled as a Multi-Criteria Decision-Making (MCDM) process. used the VIKOR method to rank the features based on the evaluation of several feature selection methods as different decision-making criteria. Their proposed method first obtains a decision matrix using the ranks of every feature according to various rankers. The VIKOR approach is then used to assign a score to each feature based on the decision matrix. Finally, a rank vector for the features generates an output in which the user can select a desired number of features.

1.2 Key contributions

As discussed, one of the main solutions to compress the dataset is record reduction. In this work, one (or more) window is considered as background; In this way, we face the activity of the user which appear in the different time window. In detail, a wight is chosen for each time window. the smaller weights are considered for distant time windows. On the contrary, near time windows, the bigger weights are selected for them. Afterward, the attention mechanism can manage the whole proposed by setting them in universal windows. The main idea of the attention mechanism is learning to set accurate weights to the set of features; These updates are done to prove the corresponded feature for the desired task has crucial information. Finally, the technique calculates the matter of each click for each user and yields the results with the system to present the priority and interest of users.

Generally, the significant properties of the proposed method are listed below:

- In contrary to clustering techniques that ignore the least repetitive, the whole record that carries the user's activity during the lifetime is maintained.
- To improve the performance of recommender systems, the dynamic values are considered for the user's behavior based on the happened time.
- The performance of the neural network is boosted by decreasing the number of windows.
- the compression operation is employed in both record and feature levels.
- amount of data compression in the row level depends on the selected time window length

Road map

The rest of this paper is organized as follows: The details of the proposed method are described in Section 3. The experimental results, analysis, and performance comparison are given in Section 4. Finally, the conclusions and future scope are shown in Section 5.

2. Proposed Method

In this section, the details of the knowledge compression method are described. As known, record compression is one of the best solutions for decreasing the volume of data. Unfortunately, most schemes presented in recent years were focused on inefficient compression in the level of features. Moreover, the distant activity with low repetition is meaningless in these works. To overcome these weaknesses, in this work, none of the records that indicate the activity of users during their lifetime are deleted; Also, the compression operation is employed in both record and feature levels. The activity of the user can be reading a web page, downloading PDFs, watching educational movies, etc. Besides, for improving the

performance of recommender schemes in terms of recommendation, the value of users' behavior becomes valuable over time. In other words, current activities are more substantial in comparison to the previous.

2.1. Symbols

Assuming U and V represent a collection of users and items, respectively. In this scheme, the main goal is extracting the interests and priorities of users by looking at User-Item interactive events. For example, clicking on an educational source is considered as an action for users. Also, for each user, $u \in U$ is sequential time windows as $Wu = \{w_1u, w_2u, ..., w_tu\}$ which t represents the total number of time windows; Also, w_tu shows a collection of smaller time units as $w_tu = \{d_1, d_2, ..., d_x\}$ where x indicates the length of time windows. Moreover, the related items of the user (u) in time windows (t) express by w_tu. There are a number of events in each time window as $\{et, iu \in R_m \mid i = 1, 2, ..., |w_tu|\}$ that iu and et describe the event i in time units of windows time (d_x). Something else which should be mentioned is that user u interacts with the item $|v_i \in V|$ in each event.

2.2. Compression in Record Level

First, one window (or limited number) is considered as a background. In this way, the user activities appear in a different time window as items are observed. As mentioned, the different weights based on the average of constituent days are assigned for each window.% based on occurrence time

With the help of these techniques, each user's activity with attention to the category of the time window is marked by proportional weight. Hence, the smaller weights can be considered for distant time windows. On the contrary, due to the crucial role of near time windows, the bigger weights are selected for them. In the following, the weight of the feature is multiplied by the summation of the corresponded activity; Then, the outcome is aggregated with the results of the rest windows. Finally, each feature that is considered as a background is repeated only once in the section; and in the new field, it maintains the frequency of the merged features.

2.3. Compression in Feature Level

After compression at the record level, the feature selection with the correlation matrix technique is applied to further compress data at the feature level. This technique contains a table that demonstrates correlation coefficients between the collection of features. With the help of this strategy, the feature pairs with the highest correlation value are selected. For more info about correlation matrix refer to [14, 16].

The network's performance is significantly improved by decreasing the number of time windows and features. Notice that the Attention strategy is employed in most engineering problems such as information retrieval, machine vision, recommender systems, etc.

The details of the algorithm are summarized in the following steps:

1. First, the required preprocessing, including data cleaning, removal of missing data, etc., is done (Algorithm 1).
2. Determine the length of time windows (Ex. one month).
3. Do the following steps for each user.
4. Do the following steps for each time window.
5. Compute the average length of the time window.
6. Do the following steps for each item.
7. Compute whole activities (clicking) of the special item for determining time windows.
8. Now, the results are divided by the average of step 5. Thanks to this step, the near and distant time windows are categorized based on considering different weights.
9. In this step, the new candidate resource which appears in current time windows is appended to the collection of previous time windows. In the same candidate case, the computed value of the current windows is added to the previous value of resources.
10. After compression at the record level, the feature selection with the correlation matrix technique is applied to further compress data at the feature level. This technique contains a table that demonstrates correlation coefficients between the collection of features. With the help of this strategy, the feature pairs with the highest correlation value are selected. For more info about correlation matrix refer to [14, 16].
11. Create a new field to maintain the frequently integrated features in each window.
12. Apply the Correlation Matrix Algorithm technique on the database; To do so, the compression step must be done in the columns to compress data in column diminutions based on the importance of the available features.
13. Finally, the results are considered as input, and the output of the system is illustrated as the recommended list. In other words, the order of recommended resources is sorted by the network.

Generally, unlike presented schemes in the last decades that concentrate on feature compression (feature selection), this work presents a compressing strategy in terms of record (row). With the help of these mechanisms, the time complexity and the training time are significantly improved. Meanwhile, the accuracy of the system is optimized compared to previous works. These claims are proved with the experimental results reported in the next section.

By reducing the number of records, the network speed is increased; Also, the necessary hardware resources such as Ram and GPU to implement the deep learning network are reduced. The process of training a network with multimillion-record datasets on Kulb's with high-level hardware takes

several days; Moreover, We are facing the challenge of memory shortage due to working on datasets with a tabular structure requiring all table values to be fetched in memory to perform the network training process. By using techniques such as data generators to manage the memory challenge, the time spent on training will be much more than before. Totally, it may not be possible to use techniques such as data generators in some types of networks, such as DBN, which encounter the enlargement of the weight matrix during the training process, and researchers will have to select several records.

Algorithm 1. The pseudo-code of preprocessing Data.
Input: The CSV files of Student Info, Student VLE, Student Assessment and Assessments.
Output: Cleaned Data.
1:procedure Preprocessing Data
2: Read input files
3: Merge files
4: Remove missing values
5: Convert string values to number
6: Update date from ”far to now” to ”now to far”
7: Create compressed Dataset with following attributes
8: [id student, age band, gender, highest education, number of previous attempts, final result, code module, date, date count, mean of sum click]
9: return Optimized (Cleaned) Tables

3. Experimental Results

3.1. Dataset

In the presented work, the Open University Learning Analytics Dataset (OULAD) is used for analyzing algorithms [1, 7]. This dataset contains demographical data, including students' information, their attended courses, and the final results of each course. In detail, it contains the students' interactions with Virtual Learning Environment (VLE) for seven selected courses. The dataset includes 22 modules with the information of over 30,000 students. This information is fetched by the daily summaries of student clicks on several resources. In OULAD, the tables are connected using unique identifiers; It should be noted, that the tables are stored in the CSV format. The utilized files are briefly described below:

- Assessments: It contains information about assessments in module presentations.
- StudentInfo: This file holds demographic information about the students together with their results.
- StudentVle: The file includes information about each student’s interactions with the elements in the VLE.
- StudentAssessment: This file contains the results of students’ assessments.

For more info about the employed Dataset refer to [1, 7]

3.2. Results

Attention mechanism has been used in many tasks such as information retrieval, machine translation, computer vision, and recommender. The main idea of these techniques is to learn accurate (normalize) weights to assign for a set of attributes; In this way, higher weights indicate that the corresponding attributes contain more significant information for the corresponding task. This technique calculates the importance of each item for a specific user and gives this information to the system in order to represent the user's interests and priorities. Therefore, with the help of this strategy, important parts of the input that can be effective as a result of the output are automatically identified and gain more weight; Ultimately, it helps to get a better result.

As you can see in Table 2 and Fig. 4, better results have been achieved in structures that use the Attention layer in network architecture; So, it is extremely recommended to employ the Attention layer in network architecture. In this research, we have trained our networks in two platforms including Personal Computer with Nvidia GeForce GTX 1060 6GB and Colab.

The number of records which is used in this work after the initial stages of pre-processing is 10403715 and each record consists of 12 fields. We have evaluated the data compression effect of our algorithm in several different time window sizes. It can be seen in the WIN Count column of Table 1, that the number of windows matched on data records with different window lengths. As expected, the larger window length leads to a smaller number of windows; Due to the Correlation Matrix algorithm and the dataset fields, the compression output has a constant value equal to 11 in the

column dimension in the whole evaluation process. While the amount of data compression in the row level depends on the selected time window length;

Table 1 The effect of compression on the number of rows and columns

Original dataset					
10,543,682*12					
win count	Time Data Label(sec)	Label Count	Time Data dimensions has reduced(sec)	Record Test	Record Train
-	125.221211	562	0	3479416	7064266
Dataset Compress Win7					
5167599*11					
win count	Time Data Label(sec)	Label Count	Time Data dimensions has reduced(sec)	Record Test	Record Train
116	123.32142686843872	252	48.17566108703613	1705308	3462291
Dataset Compress Win 14					
4239764*11					
win count	Time Data Label(sec)	Label Count	Time Data dimensions has reduced(sec)	Record Test	Record Train
58	122.14785075187683	453	23.967104196548462	1399123	2840641
Dataset Compress Win 30					
3396653*11					
win count	Time Data Label(sec)	Label Count	Time Data dimensions has reduced(sec)	Record Test	Record Train
27	132.04798412322998	1123	12.850017070770264	1120896	2275757
Dataset Compress Win 60					
2816927*11					
win count	Time Data Label(sec)	Label Count	Time Data dimensions has reduced(sec)	Record Test	Record Train
14	128.292387008667	3754	8.632514953613281	929586	1887341

Table 2 Investigating the effect of selected window length in accuracy and error of training and testing phases.

		Epoch 50		Original dataset				Epoch 100		Compress Win 7			
Structure of Network		Time (seconds)		loss		accuracy		Time (seconds)		loss		accuracy	
		PC	Colab	Val	train	Val	train	Colab	PC	Val	train	Val	train
Lstm	train	115395	low Memory or Time	0.3231	0.3185	0.8652	0.8645	15853	65003	0.2731	0.2474	0.8767	0.8872
	test	1933		-				686	983	-		0.88	
Gru	train	154456		0.2742	0.2434	0.8809	0.893	42546	82781	0.2022	0.194	0.9048	0.9087
	test	2486		-				836	1558	-		0.91	
Lstm + Attentiom	train	172034		0.1998	0.172	0.9122	0.9229	15108	90539	0.2472	0.2302	0.8855	0.8934
	test	1970		-				720	1056	-		0.89	
Gru + Attentiom	train	205520		0.479	0.268	0.809	0.815	48011	93042	0.2207	0.2086	0.8977	0.9034
	test	2518		-				1005	1756	-		0.90	
Bilstm	train	188240		0.4678	0. 3575	0.8299	0.8609	99400	160885	0.5672	0. 4872	0.8104	0.8256
	test	2020		-				875	1152	-		0.82	
		Epoch 100		Compress Win 14				Epoch 100		Compress Win 30			
Structure of Network		Time (seconds)		loss		accuracy		Time (seconds)		loss		accuracy	
		PC	Colab	Val	train	Val	train	Colab	PC	Val	train	Val	train
Lstm	train	53769	12529	0.647	0.5591	0.7506	0.7735	10418	43670	1.2694	1.1116	0.5592	0.5986
	test	762	587	-		0.77		296	556	-		0.60	
Gru	train	68944	30388	0.3681	0.3352	0.837	0.8508	25549	55505	0.6365	0.7173	0.7062	0.7351
	test	1347	583	-		0.85		311	1153	-		0.72	
Lstm + Attentiom	train	86345	11982	0.5188	0.4741	0.7864	0.8029	9816	59918	0.8063	0.7019	0.6811	0.7219
	test	839	654	-		0.80		412	702	-		0.72	
Gru + Attentiom	train	79475	42066	0.515	0.4932	0.7911	0.8001	35313	60301	0.6546	0.5869	0.7283	0.753
	test	1520	875	-		0.80		401	1293	-		0.75	
Bilstm	train	132128	59400	1.155	1.0703	0.6723	0.686	50733	106813	1.1911	0.9882	0.5721	0.6218
	test	912	756	-		0.69		590	807	-		0.62	

By selecting the larger time window, the more compression and the fewer records have resulted from the compression operation. Certainly, some of the possible details are ignored and it will affect the accuracy and loss obtained during network training. On the other hand, as can be seen in the "Time Data dimensions has reduced (sec)" column of Table 1, since the proposed is separately applied to each window, the larger window length leads the smaller number of windows in less compression time

Table 3 Summarizing the impact of compression in terms of accuracy, saving memory and time. (wi means the length of window).

Dataset	Percentage reduction of data records after compression	Average execution time in Colab for trained models (seconds)	Average execution time in PC for trained architectures (seconds)	Average accuracy in trained models
Original	0	NAN	84657.2	0.88
Compressing (WI = 7)	50.98867%	22504	49875.5	0.88
Compressing (WI = 14)	59.78858%	15982	42604.1	0.78
Compressing (WI = 304)	67.78494%	13383.9	33071.8	0.682

As you can see in the "Time (seconds)" column of Table 2, the training speed would have been much higher by running on the colab platform; Due to the limitations of this type of free platform, network training with uncompressed data was not provided and we faced a shortage of memory. As we have already mentioned, researchers sometimes miss the possibility of network training due to the high volume of data and the impossibility of accessing the appropriate hardware. With our proposed method, this is provided by choosing the appropriate window length.

We have trained and evaluated five architectures such as LSTM, GRU, LSTM + Attention, GRU + Attention, and Bi-LSTM in 3 different window lengths. As shown in Table 2, in the first step for the different architectures, we have trained and tested the original and uncompressed data in 50 epochs. In the next steps, during 7, 14, and 30-day windows, we have trained and tested the same architectures with 100 epochs. In this way, plus using fewer hardware resources such as RAM, you will see that although the number of epochs has doubled, it has saved much time.

As you can see in Table 3, by comparing the results of the main data and the accuracy and loss after applying data compression in the training and testing phase of the implemented models (Table 2), it can see that the network with the length of the window equal to 7 and 14 learns with high speed and acceptable accuracy. In addition, the data volume is almost halved during window length equal to 7; despite halving the data and increasing the execution speed, The average accuracy of training and testing of the implemented models is maintained. Also, for a window with a length of 14, when our data volume has almost reached 40%, the accuracy has dropped by 0.10%.

The different parts of Fig. 2, Fig. 3, Fig. 4, Fig. 5 and Fig. 6 show the training and loss diagrams of the mentioned architectures. By looking in detail, it can be gained that over-fitting has not occurred. Also, the downward trend in the slope of the Loss charts at the early epoch indicates the appropriateness of the selected learning rate in the training process.

[1]https://analyse.kmi.open.ac.uk/open_dataset

4. Conclusion And Future Works

Due to the huge amount of data and low space for maintaining and processing them, the compression schemes attract more attention in compression in the last decades. On the other hand, the neural network including shallow and deep models plays a significant role in various problems such as data mining, image processing, machine vision, or even in medical or economic scopes. Hence, efficient mechanisms should be designed to provide a trafe of between the requirement space, process time, and the final accuracy in such networks. In this work, a new scheme to handle the mentioned problem was presented for a dataset in the type of table. In contrast to previous works which only compress data at the feature level, in this work, the compressing process was employed in both row and column dimensions. One of the main contributions of this scheme is the compressed data can be utilized without any uncompression or extraction step for the training phase. The experimental results illustrated the efficient performance of the proposed method in terms of accuracy, time, and memory space.

In future work, we will perform feature augment and assemble techniques to further improve the compressing process.

Declarations

Disclosures

The authors declare no conflict of interest.

Funding

Not Applicable

Authors Contribution

Authors’ contributions equaly in this paper

Ethical Approval and Consent to participate

Not applicable

Consent for publication

Not applicable

Acknowledgements

The authors would like to thank the anonymous reviewers for their useful comments and suggestions on drafts of this paper.

Data availability statement

Data sharing not applicable to this article as no datasets were generated during The current study

Human and Animal Ethics

Not applicable

References

1. Oulad: Open university learning analytics dataset. https://www.kaggle.com/vjcalling/oulad-open-university-learning-analytics-dataset.
2. H. Ahmadian Yazdi, S. J. Seyyed Mahdavi Chabok, and M. Kheirabadi. Dynamic educational recommender system based on improved recurrent neural networks using attention technique. Applied Artificial Intelligence, pages 1–24, 2021
3. L. Brezo˘cnik, I. Fister, and V. Podgorelec. Swarm intelligence algorithms for feature selection: a review.Applied Sciences, 8(9):1521, 2018.
4. S. Chahboun and M. Maaroufi. Performance comparison of k-nearest neighbor, random forest, and multiple linear regression to predict photovoltaic panels' power output. In Advances on Smart and Soft Computing, pages 301–311. Springer, 2022
5. Q. Chen, M. Zhang, and B. Xue. Feature selection to improve generalization of genetic programming forhigh-dimensional symbolic regression.IEEE Transactions on Evolutionary Computation, 21(5):792–806,2017.
6. A. Hashemi, M. B. Dowlatshahi, and H. Nezamabadi-pour. Ensemble of feature selection algorithms: A multi-criteria decision-making approach. International Journal of Machine Learning and Cybernetics, 13(1):49–69, 2022
7. J. Kuzilek, M. Hlosta, and Z. Zdrahal. Open university learning analytics dataset.Scientific data,4:170171, 2017.
8. C.-F. Lee, S. W. Changchien, W.-T. Wang, and J.-J. Shen. A data mining approach to databasecompression.Information Systems Frontiers, 8(3):147–161, 2006.
9. A. Lieder and R. Stolletz. Scheduling aircraft take-offs and landings on interdependent and heterogeneousrunways.Transportation Research Part E: Logistics and Transportation Review, 88:167–188, 2016.
10. B. H. Nguyen, B. Xue, and M. Zhang. A survey on swarm intelligence approaches to feature selectionin data mining.Swarm and Evolutionary Computation, 54:100663, 2020.
11. X. Ni, H. Wang, C. Che, J. Hong, and Z. Sun. Civil aviation safety evaluation based on deep beliefnetwork and principal component analysis.Safety Science, 112:90–95, 2019.
12. Y. Pan, F. He, and H. Yu. A novel enhanced collaborative autoencoder with knowledge distillation fortop-n recommender systems.Neurocomputing, 332:137–148, 2019.
13. D. Prasetyawan and R. Gatra. Algoritma k-nearest neighbor untuk memprediksi prestasi mahasiswa berdasarkan latar belakang pendidikan dan ekonomi. JISKA (Jurnal Informatika Sunan Kalijaga), 7(1):56–67, 2022.
14. D. Roobaert, G. Karakoulas, and N. V. Chawla. Information gain, correlation and support vectormachines. InFeature extraction, pages 463–470. Springer, 2006.
15. J. Uthayakumar, T. Vengattaraman, and P. Dhavachelvan. A survey on data compression techniques:From the perspective of data quality, coding schemes, data type and applications.Journal of King SaudUniversity-Computer and Information Sciences, 2018.
16. J. Van Hulse, T. M. Khoshgoftaar, A. Napolitano, and R. Wald. Threshold-based feature selection techniques for high-dimensional bioinformatics data.Network modeling analysis in health informaticsand bioinformatics, 1(1-2):47–61, 2012.
17. Y. Wan, M. Wang, Z. Ye, and X. Lai. A feature selection method based on modified binary coded antcolony optimization algorithm.Applied Soft Computing, 49:248–258, 2016.
18. A. Wang, N. An, G. Chen, L. Li, and G. Alterovitz. Accelerating wrapper-based feature selection withk-nearest-neighbor.Knowledge-Based Systems, 83:81–91, 2015.
19. Z. Wang, M. Liang, and D. Delahaye. A hybrid machine learning model for short-term estimated time ofarrival prediction in terminal manoeuvring area.Transportation Research Part C: Emerging Technologies,95:280–294, 2018.
20. B. Xue, M. Zhang, W. N. Browne, and X. Yao. A survey on evolutionary computation approaches tofeature selection.IEEE Transactions on Evolutionary Computation, 20(4):606–626, 2015.
21. Y. Zhang, H.-G. Li, Q. Wang, and C. Peng. A filter-based bare-bone particle swarm optimization algorithm for unsupervised feature selection.Applied Intelligence, 49(8):2889–2898, 2019.

Figures

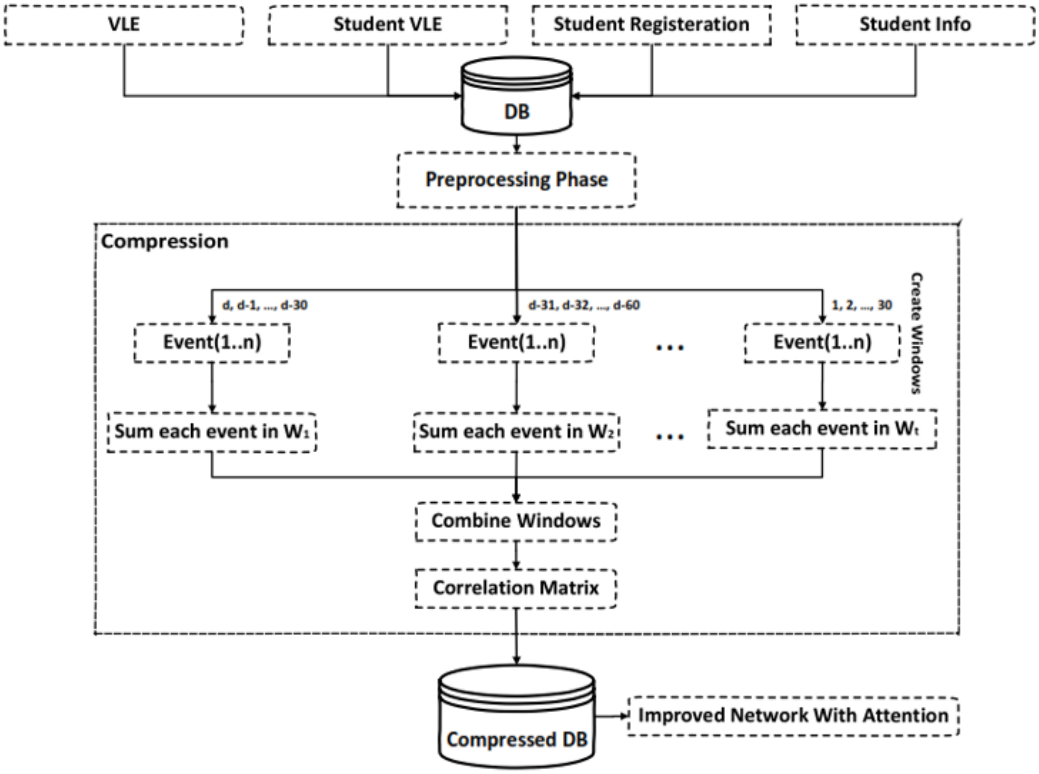

Figure 1

The block diagram of proposed method.

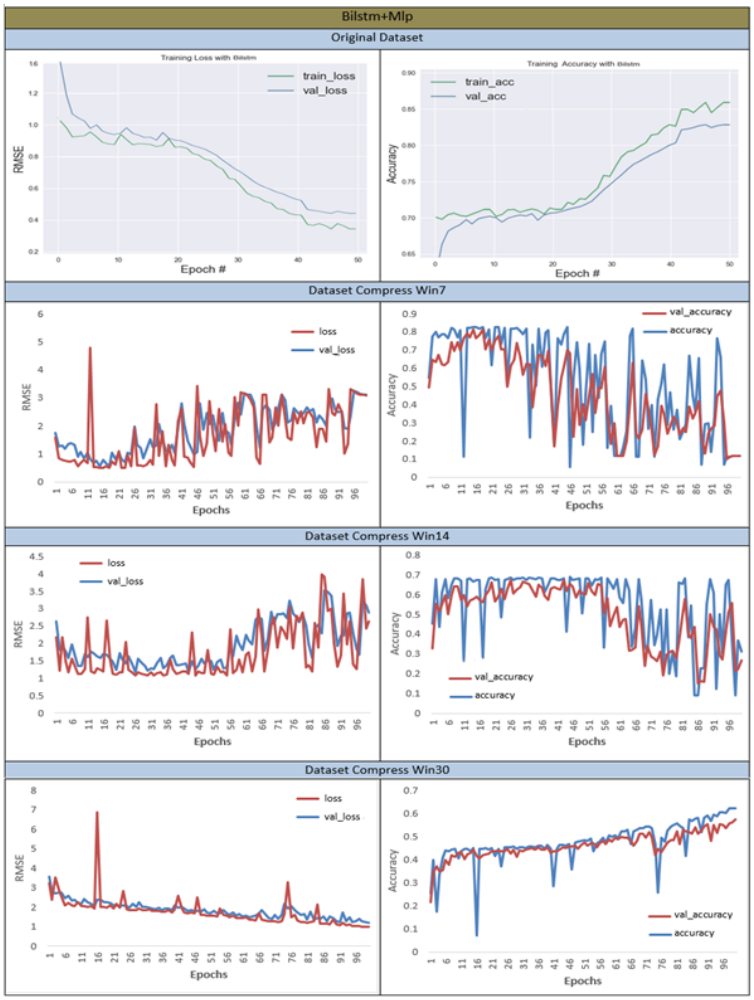

Figure 2

bilstm

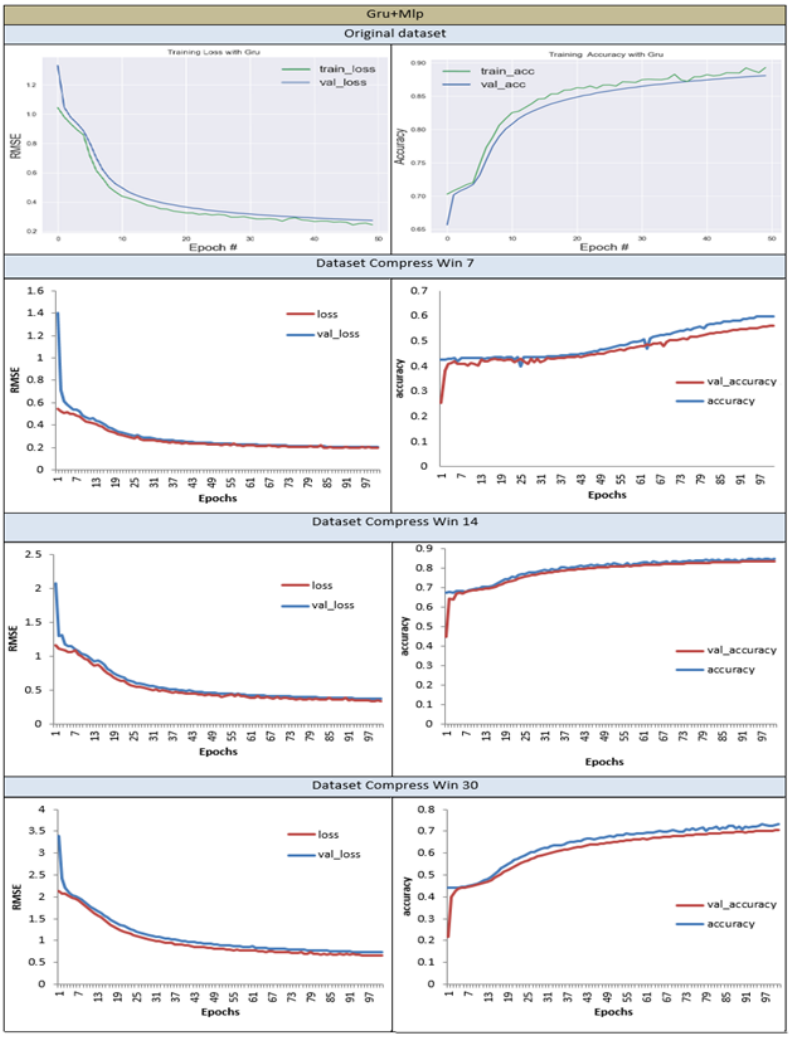
Gru+Mlp
Original dataset
Training Loss with Gru
train_loss
val_loss
RMSE
Epoch #
Training Accuracy with Gru
train_acc
val_acc
Accuracy
Epoch #
Dataset Compress Win 7
loss
val_loss
RMSE
Epochs
val_accuracy
accuracy
accuracy
Epochs
Dataset Compress Win 14
loss
val_loss
RMSE
Epochs
val_accuracy
accuracy
accuracy
Epochs
Dataset Compress Win 30
loss
val_loss
RMSE
Epochs
val_accuracy
accuracy
accuracy
Epochs

Figure 3

gru

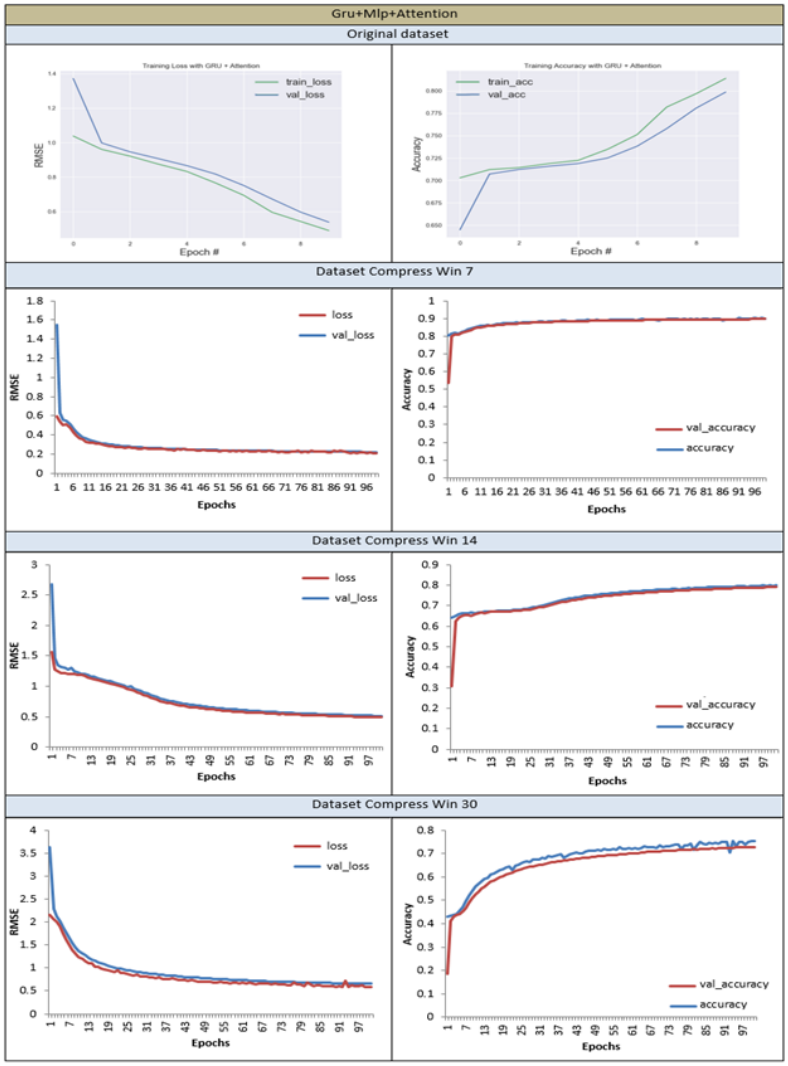

Figure 4

gru+attention

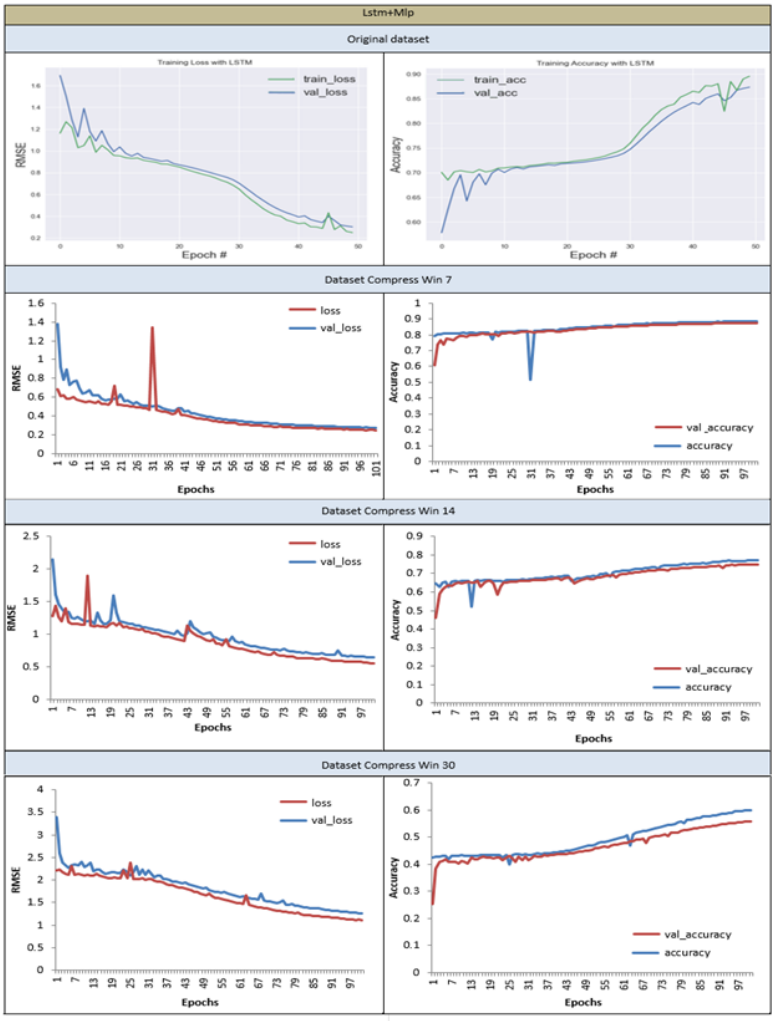

Figure 5

lstm

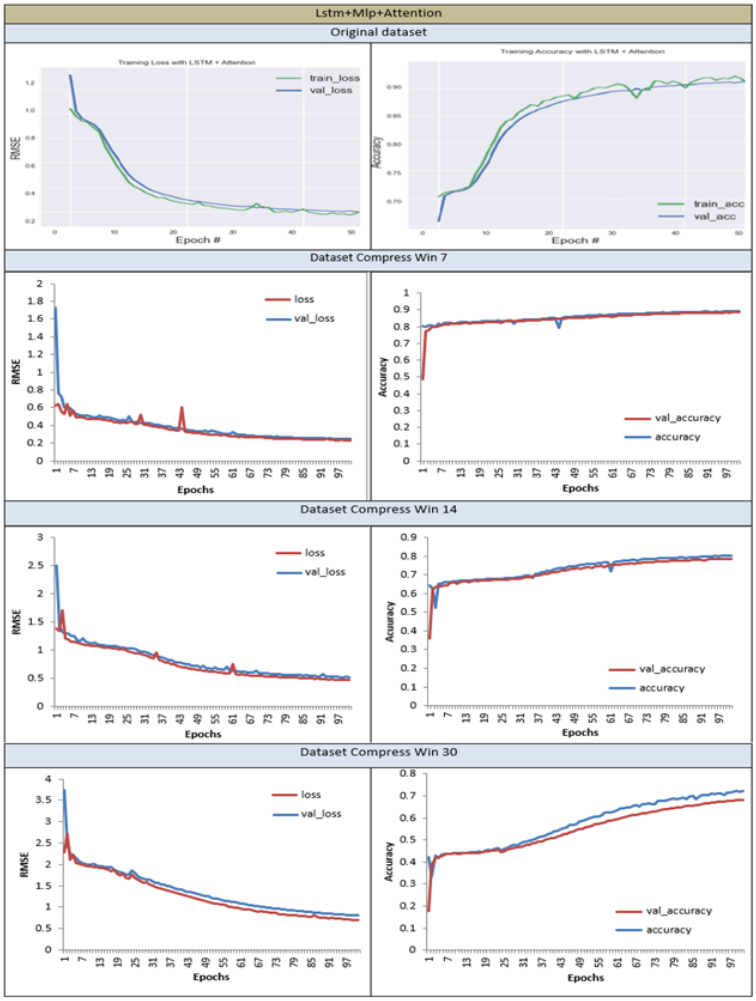

Figure 6

lstm+attention