Scientific Journal of Informatics
Vol. 12, No. 4, November 2025

p-ISSN 2407-7658 https://journal.unnes.ac.id/journals/sji/index e-ISSN 2460-0040

Optimized LSTM Model Using Simulated Annealing for Autoignition Temperature (AIT) Prediction as a Hazard Indicator

Nurul Izzah Abdussalam Zahra[1], Angel Metanosa Afinda[2], Isman Kurniawan[3]
[1,3]School of Computing, Telkom University, Indonesia
[2]School of Electrical Engineering, Telkom University, Indonesia

Abstract.
Purpose: Autoignition Temperature (AIT) is the lowest temperature at which a substance will spontaneously ignite in normal air without any external ignition source. AIT is an important safety parameter in industries that handles flammable materials. Measuring AIT with conventional method is unfortunately slow, costly, and dangerous. As an alternative, an AIT prediction model can be developed using *in silico* approaches, specifically based on machine learning.
Methods: One of the methods that can be used is Long Short-Term Memory (LSTM) since it is good at modeling the complex relationships that is involved, but unfortunately it is difficult to tune manually due to their numerous hyperparameters. Therefore, an automated strategy can be used to find the best hyperparameters for the architecture. This study aims to develop an AIT prediction model as a hazard indicator using an LSTM model optimized with Simulated Annealing (SA).
Result: The experiment showed that the SA-LSTM model which uses a cooling schedule of Delta T = 0.7 outperformed the unoptimized baseline model.
Novelty: The optimization raised the R^2 on test data from 0.5682 to 0.5939 while also lowering the RMSE from 74.35 K to 72.10 K and the MAPE from 9.29% to 8.87%. These results confirmed that optimizing LSTM with SA gave a more robust tool for hazard indicator.

Keywords: Autoignition temperature, Long short-term memory, Simulated annealing, Machine learning
Received May 2020 / **Revised** November 2020 / **Accepted** March 2021

INTRODUCTION

Autoignition temperature (AIT) is the lowest temperature at which a substance can ignite in air at normal atmospheric pressure without requiring an external energy source such as fire [1], [2], [3], [4], [5]. AIT is an important safety parameter in many common situations related to the fire potential of chemicals, such as production, processing, handling, shipping, and storage of materials [2]. Knowledge of AIT is crucial in industrial environments because this concept is used to minimize potential losses and incidents that could endanger safety. One example of its application is for electrical classification purposes, such as in NFPA 70 Section 500.8, which states that Class I equipment must not have open surfaces operating at temperatures exceeding the ignition temperature of certain gases or vapors [3].

Unfortunately, experimental studies measuring AIT are relatively rare. These measurements are known to be time-consuming and require expensive equipment. Testing usually takes 48 to 60 hours and operators are required to be present in some cases [3]. Conventional way of measuring AIT is also very risky when researchers handle hazardous chemicals [3]. Not to mention the average AIT measurement error is around 30°C according to some estimates [4]. Therefore, an alternative approach is needed in the form of a machine learning based *in silico* method for developing AIT prediction models as a hazard indicator to minimize testing time, costs, and risks.

Several studies have developed AIT prediction models using machine learning approaches. Pan et al. (2009) [5] utilized Support Vector Machine (SVM) to predict AIT with a dataset of 446 diverse organic compounds. The model achieved a Mean Absolute Error (MAE) of 28.88°C and a Root Mean Square Error (RMSE) of 36.86°C. Dashti et al. (2019) [6] explored various soft computing approaches, including Genetic

[1]*Corresponding author.
Email addresses: nrlzzahh@student.telkomuniversity.ac.id (Zahra), angelmetanosa@telkomuniversity.ac.id (Afinda), ismankrn@telkomuniversity.ac.id (Kurniawan)

DOI: 10.15294/sji.v12i4.38278

Programming (GP) and Adaptive Network-based Fuzzy Inference System (ANFIS), demonstrating the capability of non-linear modeling in capturing AIT characteristics.

Several studies that utilized deep learning methods have also contributed to this domain. Pan et al. (2008) [7] developed an Artificial Neural Network (ANN)-based Quantitative Structure-Property Relationship (QSPR) model for 118 hydrocarbon compounds. The model yielded an RMSE of 31.09°C. Similarly, Gharaghezi (2011) [8] employed an Artificial Neural Network-Group Contribution (ANN-GC) method on a larger dataset of 1025 pure compounds and the model achieved a high correlation coefficient of 0.984 and an RMSE of 15.44 K. Guo et al. (2024) [9] has also implemented advanced architectures. They specifically implemented Back Propagation Neural Network (BPNN) and One Dimensional Convolutional Neural Network (1DCNN) in their research. Their study highlighted the superiority of deep learning features, with the BPNN and 1DCNN models producing RMSE values of 3.613°C and 5.284°C respectively.

AIT prediction models' performance has room for improvement despite these research advancements. The Long Short-Term Memory (LSTM) network can be used as an alternative for QSPR modeling due to its potential in learning latent patterns from data that cannot be captured by ordinary feedforward models [10]. But it is noted that the success of an LSTM model depends entirely on how its hyperparameters are configured. The architecture involves a complex search space which includes embedding dimensions, number of units, dropout rates, and learning rates. This is why manual tuning in LSTM is inefficient and prone to sub-optimal convergence. It is important to employ an automated optimization strategy to fine-tune these parameters.

The use of metaheuristic optimization in AIT prediction has been validated in prior researches. Lazzus (2010) [11] developed a hybrid model which combines the Group Contribution Method (GCM) with ANN and Particle Swarm Optimization (PSO). The model predicted AIT values for external compounds with an Average Absolute Relative Deviation (AARD) of only 1.7%. Similarly, Dashti et al. [12] demonstrated that optimizing ANFIS with evolutionary algorithms such as Genetic Algorithms (GA), Differential Evolution (DE), and PSO significantly improved prediction accuracy. The model achieved AARD values as low as 6.29%.

To the best of our knowledge, there has been no prior research investigating the specific application of Simulated Annealing to optimize LSTM networks for the prediction of AIT. This study aims to develop an AIT value prediction model as a hazard indicator using the LSTM method optimized with SA. With the combination of LSTM and SA, it is expected that the resulting AIT prediction model will provide better performance than the unoptimized model. This study will compare the performance of the LSTM model optimized using SA with the LSTM model without optimization.

METHODS

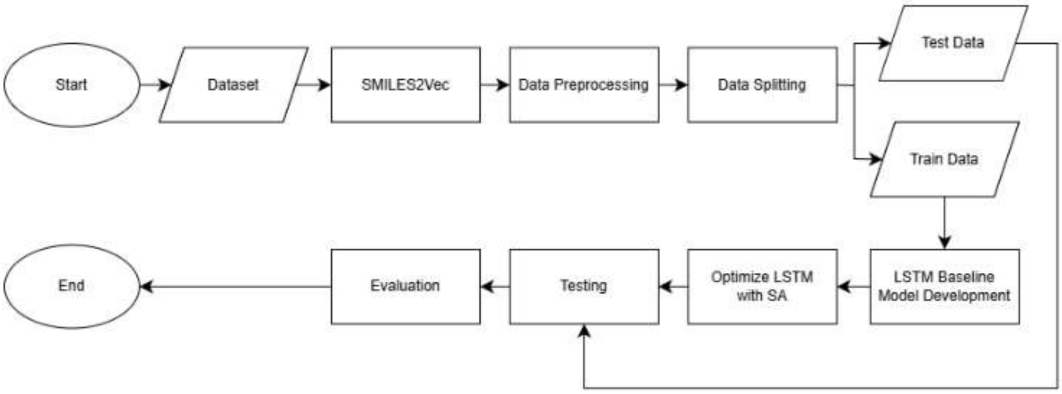

Figure 1. Research workflow

This study developed an AIT prediction model designed to function as a hazard indicator. The methodology follows a sequential workflow and it is illustrated in Figure 1. It begins with the SMILES2Vec technique to transform SMILES into vector representations. It is followed by data preprocessing to handle missing values and normalize the AIT target variable, which then the dataset is partitioned into an 80:20 train-test split. A baseline LSTM model is then developed. It is followed by the process of optimizing the baseline LSTM model using SA to optimize the hyperparameters. The process ends with the testing and evaluation of the final model.

The dataset used in this research was obtained from Taylor & Francis Online website [13]. The dataset is in CSV format. It contains information of 801 hydrocarbon compounds and their physical and chemical properties. This dataset contains 126 descriptors for each hydrocarbon compound. The descriptors cover a lot of properties such as molecular structure, thermodynamic properties, and other physical and chemical properties. The target variable in this dataset is the AIT value which serves as the label for prediction. The input feature is the SMILES column which is later vectorized with SMILES2Vec. The histogram of AIT values is shown in Figure 2. The AIT spans from approximately 300 K to over 900 K with a slightly positively skewed distribution centered around 650-700 K. The AIT values were then normalized to ensure stable gradient behavior.

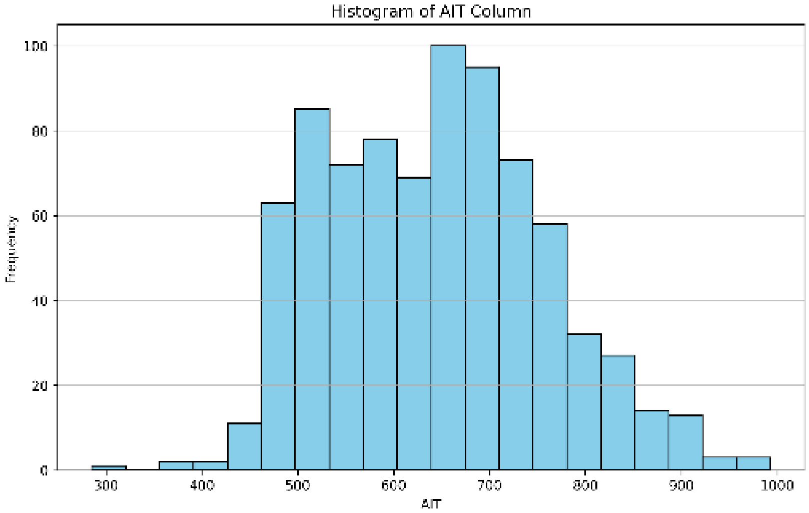

Figure 2. Histogram of AIT values

The SMILES2Vec method is used in this study to convert SMILES (Simplified Molecular-Input Line-Entry System) strings into numerical vector representations [14], [15]. It acts like a translator that takes a chemical sentence and turn it into a list of numbers that can be processed by deep learning architectures such as LSTM model which is proposed in this research. The SMILES2Vec flowchart is shown in Figure 3. It begins with the extraction of unique characters from the dataset to build a mapping dictionary. It is followed by identifying the maximum sequence length within the dataset. Padding is then applied to ensure that all input vectors possess the same dimension. It is followed by the process of converting each SMILES string into a matrix representation. The process is concluded with dimensionality reduction where specific functions are applied to compress the vector space. This step is used to reduce the computational complexity while preserving essential structural information.

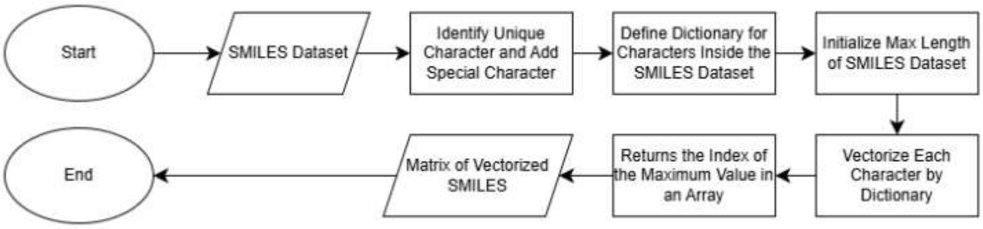
Start
SMILES Dataset
Identify Unique Character and Add Special Character
Define Dictionary for Characters Inside the SMILES Dataset
Initialize Max Length of SMILES Dataset
Vectorize Each Character by Dictionary
Returns the Index of the Maximum Value in an Array
Matrix of Vectorized SMILES
End

Figure 3. SMILES2Vec flowchart

The Long Short-Term Memory (LSTM) neural network is utilized in this study. LSTM is a variant of the Recurrent Neural Network (RNN) which is designed to overcome the exploding and vanishing gradient.

These problems usually arise when learning long term dependencies in a sequential data [16]. These issues are mitigated through the use of a Constant Error Carousel (CEC) which stabilizes the flow of error signals within each cell. It allows gradients to propagate over extended time periods without diminishing or growing uncontrollably.

The LSTM architecture mitigates the vanishing gradient problem through a gating mechanism that regulates the information flow [16]. The main core of LSTM architecture is the cell state. The cell state functions as the memory component that stores information across different time intervals. The process begins with the calculation of the block input, or candidate state, $z^{(t)}$ which is formulated in Equation 1 and it represents new information potentially relevant to the sequence.

The gates regulate the flow of information into, out of, and within the cell. The input gate determines which new information from the block input should be stored and it is calculated in Equation 2. The forget gate controls which information from the previous memory state that should be discarded. It is calculated in Equation 3. The current cell state $c^{(t)}$ is then updated in Equation 4 by combining the retained previous memory and the significant new input. The output gate regulates how much of the cell state contributes to the hidden state at each time step and it is calculated in Equation 5. And lastly the hidden state or network output $y^{(t)}$is calculated in Equation 6 by filtering the activated cell state through the output gate.

$$z^{(t)} = g(W_z x^{(t)} + R_z y^{(t-1)} + b_z) \tag{1}$$
$$i^{(t)} = \sigma(W_i x^{(t)} + R_i y^{(t-1)} + p_i \odot c^{(t-1)} + b_i) \tag{2}$$
$$f^{(t)} = \sigma(W_f x^{(t)} + R_f y^{(t-1)} + p_f \odot c^{(t-1)} + b_f) \tag{3}$$
$$c^{(t)} = z^{(t)} \odot i^{(t)} + c^{(t-1)} \odot f^{(t)}) \tag{4}$$
$$o^{(t)} = \sigma(W_o x^{(t)} + R_o y^{(t-1)} + p_o \odot c^{(t)} + b_o) \tag{5}$$
$$y^{(t)} = g(c^{(t)}) \odot o^{(t)} \tag{6}$$

A series of baseline LSTM models were trained to establish a performance benchmark for this experiment. To evaluate the impact of network depth on the model's performance, three different baseline schemes were constructed. The parameters of the three baseline LSTM models are shown in Table 1. These three baseline models utilized a fixed and standardized configuration. The Rectified Linear Unit (ReLU) activation function and Adam optimizer were employed to isolate the influence of structural depth prior to the application of the Simulated Annealing hyperparameter optimization.

Table 1. Parameter of LSTM model

Scheme	Hidden Layer	Unit	Activation	Optimization	Batch Size
Baseline 1	1	[64]	ReLU	Adam	32
Baseline 2	2	[64, 64]			
Baseline 3	3	[64, 64, 64]			

The Simulated Annealing (SA) algorithm is used in this study to optimize the hyperparameters of the LSTM model. SA is a meta-heuristic local search algorithm capable of escaping local optima through its hill-climbing mechanism [17], [18], [19]. SA is usually employed to solve discrete optimization problems. Its primary advantages over other local search methods include its flexibility and capacity to approach global optimality and also its robustness in handling highly non-linear models [18].

The procedure of the Simulated Annealing algorithm iterates through a sequence of neighbor generation and evaluation steps designed to locate the global minimum of the loss landscape [18]. At each iteration, a new candidate solution is generated by applying a random perturbation to the current set of LSTM hyperparameters. The algorithm then evaluates the quality of this candidate by calculating the change in the objective function (validation MSE), denoted as δf. If the candidate solution yields a lower error ($\delta f < 0$), it is automatically accepted as the new current state. However, to facilitate the escape from local optima, the algorithm employs a stochastic acceptance mechanism for non-improving solutions. A candidate that increases the error ($\delta f < 0$) is not immediately discarded. Instead, it is accepted with a specific probability determined by the Metropolis criterion. This probability p relies on the current temperature T and the magnitude of the performance degradation, as formulated in Equation 7.

$$p = \exp\left(-\frac{\delta f}{T}\right) \tag{7}$$

The bounds of SA algorithm used in this research are shown in Table 2. The algorithm starts with an initial temperature of 10.0 and proceeds for a maximum of 20 iterations to facilitate an extensive initial search of the hyperparameter space. A very important aspect of the configuration is the perturbation logic. It dictates how new candidate solutions are generated from the current state. Specific step sizes were assigned based on the parameter type. Discrete additive steps (±10 or ±20) were applied to structural integers (embedding and units). Finer continuous adjustments were utilized for the dropout rate (±0.05) and learning rate (±20%) to allow for precise tuning.

Table 2. Configuration parameters for the SA algorithm

SA Parameter	Value
Initial Temperature	10
Max Iterations	20
Perturbation (Emb and Unit)	±{10, 20}
Perturbation (Dropout)	±0.05
Perturbation (LR)	±20%

Three different cooling schemes were evaluated to analyze the impact of the cooling rate on the algorithm's behavior. The three cooling schemes are presented in Table 3. SA1 (ΔT = 0.7) represents a fast-cooling strategy used to maximize the computational efficiency. It sounds good because it prioritizes rapid convergence but it has a higher risk of premature entrapment in local minima. SA2 (ΔT = 0.8) serves as an intermediate control configuration to investigate the optimal balance between search thoroughness and convergence speed. SA3 (ΔT = 0.9) represents a slow cooling approach. It maintains higher temperatures for a prolonged duration to maximize the probability of escaping local optima through extensive exploration of the solution space.

Table 3. Experimental cooling schemes for SA

Scheme	Cooling Factor (ΔT)	Classification
1	0.7	Fast Cooling
2	0.8	Moderate Cooling
3	0.9	Slow Cooling

The embedding dimension, the number of LSTM units, the dropout rate, and the learning rate of the LSTM model were selected for modification and the defined search space is summarized in Table 4.

Table 4. Search space and bounds for LSTM hyperparameters

Hyperparameter	Data Type	Lower Bound	Upper Bound
Embedding Dimension	Integer	10	500
LSTM Units	Integer	20	500
Dropout Rate	Float	0.2	0.5
Learning Rate	Float	0.0001	0.01

Three statistical metrics were employed to assess the predictive capability and generalization performance of the proposed model. These metrics are the Coefficient of Determination (R^2), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). These three metrics provide a comprehensive evaluation of the model's accuracy in predicting the AIT of chemical compounds.

The R^2 score indicates the goodness of fit which represents the proportion of variance in the dependent variable (AIT) that is predictable from the feature [19]. An R^2 closer to 1 indicates a model that perfectly predicts the observed data, meanwhile a value of 0 indicates no correlation. R^2 is calculated in Equation 8. RMSE is particularly significant for hazard prediction as it penalizes larger errors more heavily than smaller ones [20]. Minimizing large deviations is very important to prevent underestimation of safety risks especially in the context of AIT. The formula is given in Equation 9. MAPE expresses the prediction accuracy as a percentage. It provides an intuitive measure of the error relative to the magnitude of the actual AIT values [21]. This allows for a clear interpretation of the model's reliability across different temperature ranges. MAPE is formulated in Equation 10.

In these equations, y_i represents the actual experimental AIT values and $\hat{y}_i$ represents the AIT value predicted by the model. The variable ȳ represents the mean of the observed experimental values and lastly the variable n represents the total number of the samples in the dataset.

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \quad (8)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (9)$$

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \quad (10)$$

RESULT AND DISCUSSION

Three different baseline LSTM configurations were constructed in the beginning of this study. These three models used standard hyperparameter configurations without optimization as discussed in the previous chapter. The objective was to investigate the influence of the layer depth and unit count on the model performance while also determining the baseline capability and identify the limitations when using manually selected parameter.

The performance metrics for the baseline models are shown in Table 5. While all three models demonstrated the ability to learn the underlying patterns in the training data, achieving training R^2 scores between 0.63 and 0.66, their performance significantly degraded when applied to the unseen testing dataset.

Table 5. Comparative of performance metrics of baseline LSTM models

Train Data			
Scheme	**R^2**	**RMSE**	**MAPE**
Baseline 1	0.6529	66.7547	8.44%
Baseline 2	0.6321	68.7244	8.59%
Baseline 3	0.6589	66.1812	8.20%
Test Data			
Scheme	**R^2**	**RMSE**	**MAPE**
Baseline 1	0.5682	74.3549	9.29%
Baseline 2	0.5421	76.5678	9.25%
Baseline 3	0.5586	75.1797	9.05%

As observed in Table 5, the average testing R^2 across the three baselines is approximately 0.56. This indicates that these unoptimized models can only explain roughly 56% of the variance in the AIT data. Furthermore, the testing RMSE values range from 74.35 K to 76.57 K. In the context of hazard indicators, a deviation of this magnitude suggests a high risk of misclassification. Lastly, the testing MAPE is in the range of 9.05% to 9.29% which is considerable.

The learning behavior of the baseline models is visually represented in Figure 4. This figure displays the convergence of Loss (MSE) over the training epochs. Baseline 1 shows a rapid decrease in training loss initially. However, a noticeable gap emerges between the training and validation curves around epoch 20. The validation loss plateaus and exhibits noise which suggests the model is struggling to generalize features from the training set to the validation set. Other than that, baseline 2 exhibits the clearest signs of instability. While the training loss continues to decrease, the validation loss flattens significantly earlier. Lastly, the convergence plot of baseline 3 reveals that the validation loss stops improving relative to the training loss which indicates overfitting.

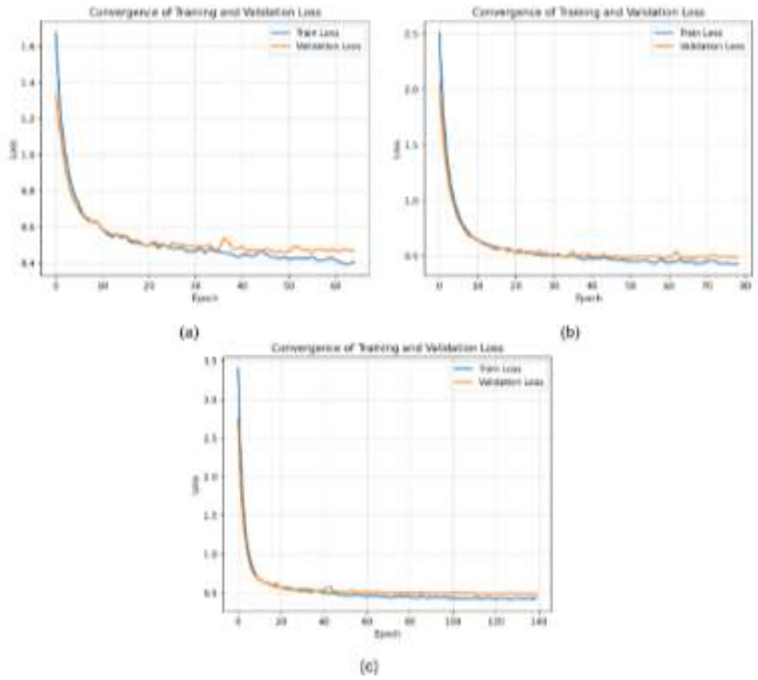

Figure 4. Convergence of training and validation loss plot of baseline 1 (a), baseline 2 (b), and baseline 3 (c)

To overcome the performance limitations observed in the baseline models, SA algorithm was employed to navigate the hyperparameter space in accordance with the cooling schemes previously outlined. The convergence of the MSE for each baseline is presented in Figure 5. For baseline 1, scheme 1 demonstrated the most efficient trajectory which rapidly converging to the global minimum MSE of 0.412 by the 4th iteration. For baseline 2, scheme 1 also yielded the lowest final error which shows a sharp descent after the 11th iteration. Lastly, for baseline 3, scheme 3 emerged as the superior strategy with the lowest final MSE of 0.421 at the 19th iteration.

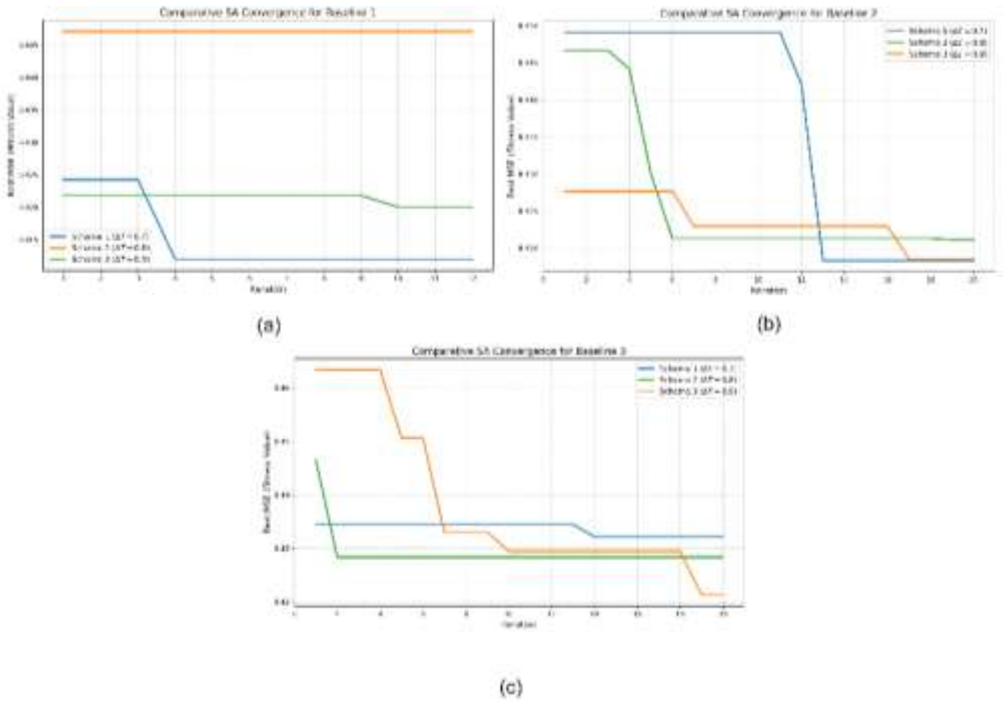

Figure 5. Comparative simulated annealing convergence of baseline 1 (a), baseline 2 (b), and baseline 3 (c)

The SA algorithm explored a search space consisting of embedding dimension, number of LSTM units, dropout rate, and learning rate. The optimal configurations identified for each baseline and scheme are summarized in Table 6. Among these configurations, the set yielding the lowest validation MSE was selected as the final SA-LSTM model for the detailed performance validation discussed in the following section.

Table 6. Optimal hyperparameters identified by SA

Baseline Scheme	SA Scheme	Embedding Dim	LSTM Unit	Dropout Rate	Learning Rate
Baseline 1	SA1	233	184	0.485	0.00182
	SA2	55	482	0.349	0.00040
	SA3	44	229	0.256	0.00270
Baseline 2	SA1	279	366	0.476	0.00613
	SA2	140	38	0.285	0.00325
	SA3	351	456	0.248	0.00081
Baseline 3	SA1	106	74	0.200	0.00863
	SA2	81	55	0.422	0.00263
	SA3	45	61	0.312	0.00772

Following the optimization phase, the best performing configurations for each baseline were evaluated. The models that were chosen are Baseline 1-SA1, Baseline 2-SA1, and Baseline 3-SA3 and the final training and testing metrics for these optimized models are summarized in Table 7. Baseline 1-SA1 emerged to be the superior model. It succeeded in achieving the highest testing R^2 (0.5939) and the lowest testing RMSE (72.10 K). This model demonstrated the best generalization capability, which can be seen in the minimal performance gap between training and testing scores. Baseline 2-SA1 showed signs of overfitting in contrast to Baseline 1-SA1. The training R^2 of Baseline 2-SA1 (0.6324) was competitive, but the testing R^2 dropped significantly to 0.4991. The RMSE and MAPE also spiked to 80.08 K and 9.64%. This showed that the Baseline 2-SA1 memorized noise in the training data rather than learning the true chemical characteristics of the data. Baseline 3-SA3 also performed moderately but failed to surpass Baseline 1-SA1. The performance metrics of Baseline 3-SA3 showed a testing MAPE of 9.30% which indicates a higher relative error margin compared to the 8.87% achieved by Baseline 1-SA1.

Table 7. Performance metrics of the optimized SA-LSTM models

Train Data			
Scheme	**R^2**	**RMSE**	**MAPE**
Baseline 1-SA1	**0.6491**	**67.12**	**8.44%**
Baseline 2-SA1	0.6324	68.70	8.28%
Baseline 3-SA3	0.6073	71.01	8.89%
Test Data			
Scheme	**R^2**	**RMSE**	**MAPE**
Baseline 1-SA1	**0.5939**	**72.10**	**8.87%**
Baseline 2-SA1	0.4991	80.08	9.64%
Baseline 3-SA3	0.5356	77.11	9.30%

Baseline 1-SA1 model was selected as the final proposed model based on the model's performances. The hyperparameters identified for Baseline 1-SA1 were an embedding dimension of 233, 184 LSTM units, a dropout rate of 0.485, and a learning rate of 0.00182. The robustness of this configuration can be attributed to the interaction between these parameters that were selected. The effectiveness of the SA optimization is evident when comparing the performance of the final best model against the unoptimized baseline model. The comparison between the unoptimized and optimized Baseline 1 model can be seen in Figure 6. The optimization with SA improved the testing R^2 from 0.5682 to 0.5939 and reduced the MAPE from 9.29% to 8.87%. An R^2 of approximately 0.59 indicates that the complexity of chemical structure-property relationships is still challenging to capture fully, but the reduction in RMSE from 74.35 to 72.10 K represents a real improvement in safety estimation. This reduction narrows the uncertainty window. It allows for a more reliable screening of chemical autoignition risks.

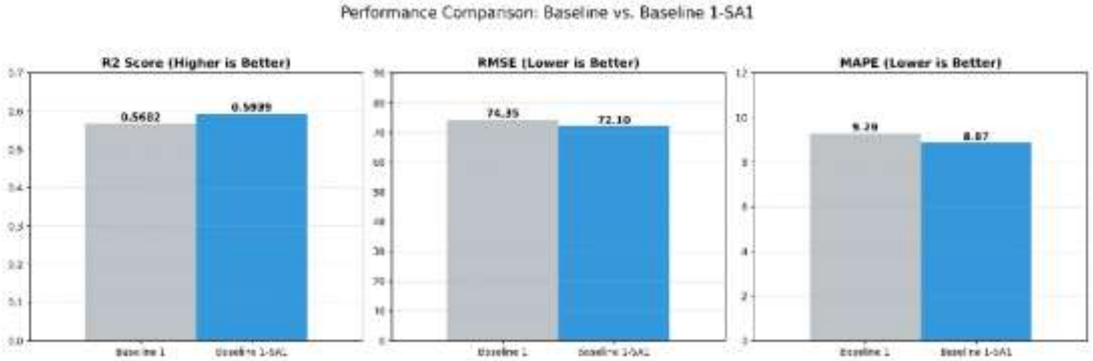

Figure 6. Performance comparison of baseline 1 and baseline 1-SA1

CONCLUSION

This study presented an optimized Long Short-Term Memory (LSTM) model optimized using Simulated Annealing (SA) to predict the Autoignition Temperature (AIT) of chemical compounds. The experimental results showed that the SA algorithm is effective in improving the performance of the unoptimized LSTM model. It can be seen from the performance of the best model selected, specifically Baseline 1-SA1. Baseline 1-SA1 model demonstrated the most efficient convergence that minimized the prediction error. The optimization using SA increased the R^2 from 0.5682 to 0.5939. It also succeeded in reducing the MAPE from 9.29% to 8.87% and the RMSE from 74.35 K to 72.10 K. Despite the improvements achieved, this study acknowledges certain limitations to the proposed model and methodology. The performance of deep learning architectures like LSTM is heavily dependent on the volume and diversity of the dataset used. The dataset utilized in this study may suffer from data scarcity regarding rare or highly specialized chemical families even though it is representative of common industrial chemicals. To advance the capabilities of the current framework, future research may explore more deep learning architectures like Graph Neural Network (GNNs) or Transformers. Future works might also explore more parameters of SA to further refine the search trajectory and enhance the algorithm's precision.

REFERENCES

[1] C.-C. Chen and Y.-C. Hsieh, “Effect of Experimental Conditions on Measuring Autoignition Temperatures of Liquid Chemicals,” Ind. Eng. Chem. Res., vol. 49, no. 12, pp. 5925–5932, Jun. 2010, doi: 10.1021/ie9020649.

[2] B. E. Mitchell and P. C. Jurs, “Prediction of Autoignition Temperatures of Organic Compounds from Molecular Structure,” J. Chem. Inf. Comput. Sci., vol. 37, no. 3, pp. 538–547, May 1997, doi: 10.1021/ci960175l.

[3] C. C. Chen, T. H. Han, S. X. Hong, and D. J. Hsu, “Auto-Ignition Temperature Data for Selected Ketones,” Adv. Mat. Res., vol. 560–561, pp. 145–151, Aug. 2012, doi: 10.4028/www.scientific.net/AMR.560-561.145.

[4] I. I. Baskin, S. Lozano, M. Durot, G. Marcou, D. Horvath, and A. Varnek, “Autoignition temperature: comprehensive data analysis and predictive models,” SAR QSAR Environ. Res., vol. 31, no. 8, pp. 597–613, Aug. 2020, doi: 10.1080/1062936X.2020.1785933.

[5] Y. Pan, J. Jiang, R. Wang, H. Cao, and Y. Cui, “Predicting the auto-ignition temperatures of organic compounds from molecular structure using support vector machine,” J. Hazard. Mater., vol. 164, no. 2–3, pp. 1242–1249, May 2009, doi: 10.1016/j.jhazmat.2008.09.031.

[6] A. Dashti, M. Jokar, F. Amirkhani, and A. H. Mohammadi, “Quantitative structure property relationship schemes for estimation of autoignition temperatures of organic compounds,” J. Mol. Liq., vol. 300, p. 111797, Feb. 2020, doi: 10.1016/j.molliq.2019.111797.

[7] Y. Pan, J. Jiang, R. Wang, and H. Cao, “Advantages of support vector machine in QSPR studies for predicting auto-ignition temperatures of organic compounds,” Chemometrics and Intelligent Laboratory Systems, vol. 92, no. 2, pp. 169–178, Jul. 2008, doi: 10.1016/j.chemolab.2008.03.002.

[8] F. Gharagheizi, “An accurate model for prediction of autoignition temperature of pure compounds,” J. Hazard. Mater., vol. 189, no. 1–2, pp. 211–221, May 2011, doi: 10.1016/j.jhazmat.2011.02.014.

[9] B. Guo, Z. Cheng, and S. Hu, “Neural network-based prediction of auto-ignition temperature of ternary mixed liquids,” Heliyon, vol. 10, no. 7, p. e28713, Apr. 2024, doi: 10.1016/j.heliyon.2024.e28713.

[10] M. Sundermeyer, H. Ney, and R. Schluter, “From Feedforward to Recurrent LSTM Neural Networks for Language Modeling,” IEEE/ACM Trans. Audio Speech Lang. Process., vol. 23, no. 3, pp. 517–529, Mar. 2015, doi: 10.1109/TASLP.2015.2400218.

[11] J. A. Lazzús, “Autoignition Temperature Prediction Using an Artificial Neural Network with Particle Swarm Optimization,” Int. J. Thermophys., vol. 32, no. 5, pp. 957–973, May 2011, doi: 10.1007/s10765-011-0956-4.

[12] A. Dashti, M. Jokar, F. Amirkhani, and A. H. Mohammadi, “Quantitative structure property relationship schemes for estimation of autoignition temperatures of organic compounds,” J. Mol. Liq., vol. 300, p. 111797, Feb. 2020, doi: 10.1016/j.molliq.2019.111797.

[13] J. Chen, L. Zhu, and J. Wang, “Quantitative structure-property relationship modelling on autoignition temperature: evaluation and comparative analysis,” SAR QSAR Environ. Res., vol. 35, no. 3, pp. 199–218, Mar. 2024, doi: 10.1080/1062936X.2024.2312527.

[14] A. M. Afinda, A. F. Karimah, and I. Kurniawan, “Gated Recurrent Unit with SMILES2Vec-based Descriptor for Predicting Drug Side Effects: Case Study of Hepatobiliary Disorders,” in 2023 International Conference on Data Science and Its Applications (ICoDSA), IEEE, Aug. 2023, pp. 426–431. doi: 10.1109/ICoDSA58501.2023.10276594.

[15] A. F. Karimah, A. M. Afinda, and I. Kurniawan, “Implementation of SMILES2Vec-based LSTM for Predicting Drug Side Effects: Case Study of Hepatobiliary Disorder,” in 2023 International Conference on Data Science and Its Applications (ICoDSA), IEEE, Aug. 2023, pp. 333–338. doi: 10.1109/ICoDSA58501.2023.10277245.

[16] G. Van Houdt, C. Mosquera, and G. Nápoles, “A review on the long short-term memory model,” Artif. Intell. Rev., vol. 53, no. 8, pp. 5929–5955, Dec. 2020, doi: 10.1007/s10462-020-09838-1.

[17] D. Bertsimas and J. Tsitsiklis, “Simulated Annealing,” Statistical Science, vol. 8, no. 1, Feb. 1993, doi: 10.1214/ss/1177011077.

[18] D. Henderson, S. H. Jacobson, and A. W. Johnson, “The Theory and Practice of Simulated Annealing,” in Handbook of Metaheuristics, Boston: Kluwer Academic Publishers, pp. 287–319. doi: 10.1007/0-306-48056-5_10.

[19] F. Busetti, “Simulated annealing overview.” [Online]. Available: http://www.geocities.com/

[20] M. S. Lewis-Beck and A. Skalaban, “The R -Squared: Some Straight Talk,” Political Analysis, vol. 2, pp. 153–171, Jan. 1990, doi: 10.1093/pan/2.1.153.

[21] T. O. Hodson, “Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not,” Geosci. Model Dev., vol. 15, no. 14, pp. 5481–5487, Jul. 2022, doi: 10.5194/gmd-15-5481-2022.