PriMera Scientific Engineering
Volume 4 Issue 4 April 2024
DOI: 10.56831/PSEN-04-120
ISSN: 2834-2550

PriMera Scientific Publications

# Thermodynamic Equilibrium State Prediction by Deep Learning Method

**Type:** Research Article
**Received:** March 03, 2024
**Published:** March 20, 2024

**Citation:**
Yu Okano., et al. "Thermodynamic Equilibrium State Prediction by Deep Learning Method". PriMera Scientific Engineering 4.4 (2024): 23-30.



**Yu Okano*, Takeshi Kaneshita*, Katsuki Okuno, Shimpei Takemoto and Yoshishige Okuno**

*Resonac Corporation, 8, Ebisu-cho, Kanagawa-ku Yokohama 2210024 Japan*

***Corresponding Author:** Yu Okano, Takeshi Kaneshita, Resonac Corporation, 8, Ebisu-cho, Kanagawa-ku Yokohama 2210024 Japan.

## Abstract

The method of calculation of phase diagrams (CALPHAD) is a calculation method that searches for a state of providing minimal Gibbs energy as an equilibrium state. To perform a thermodynamic equilibrium calculation for a single material composition and to predict a phase diagram, we can complete the CALPHAD method calculation within a realistic time. However, screening many material compositions associated with predicting the corresponding phase diagrams takes much time. For alloy materials, for example, it would take 161 hours to calculate phase diagrams of all alloy compositions to screen 10,000 sets of explanatory variables, i.e., compositions and manufacturing conditions, since it takes 58 seconds to calculate each set. The present study aims to provide a calculation device, method, and program for quickly predicting the thermodynamic equilibrium state. We developed a deep learning model based on the Transformer architecture to achieve this objective, primarily used for various natural language processing tasks, such as machine translation, text summarization, question answering, and sentiment analysis. The encoder part of our developed model extracts the necessary features for phase diagram prediction from the inputted alloying elements. In contrast, the decoder part predicts a phase diagram for each temperature based on the results from the encoder. We calculated 800,000 species using the CALPHAD method and employed these data to train our developed model. Our trained model can calculate thermodynamic equilibrium states more than 100 times faster than the CALPHAD method and correctly reproduce the phase diagrams of ground truths. Based on the present result, we could invent a calculation device, a calculation method, and a calculation program for predicting the thermodynamic equilibrium state in a short time.

***Keywords:*** Thermodynamic equilibrium state; Transformer; Neural network; Deep learning; CALPHAD

## Introduction

Prediction of mechanical properties of aluminum alloys is essential to apply the aluminum alloy materials to various purposes. For example, some automobile parts need high-strength aluminum alloy materials at high temperatures. The process determines the structure, and the structure determines the property [Fig1]. In this research, we focus on predicting the thermodynamic equilibrium structure from the process because the thermodynamic equilibrium structure provides an idea of the properties of aluminum alloy so well.

Process condition consists of additive elements, amount of Si, Fe, other elements, heat treatment, working temperature, time, etc. Structure information includes chemical compounds, types, phase fractions, grain size, and shape. The properties have mechanical properties, ultimate tensile strength, tensile yield stress, elongation, etc. If we focus on the multiple-step prediction, we need to calculate thermodynamic equilibrium structures for various numbers of candidates and then forecast the corresponding properties. We can use the calculation-of-phase-diagrams (CALPHAD) method [1] to calculate thermodynamic equilibrium structures, but the CALPHAD method is time consuming when calculating for various numbers of candidates.

To accelerate the thermodynamic equilibrium calculation speed, we have tried to use the deep learning method by regarding the CALPHAD calculation results as train data. In particular, we have attempted to use transformer architecture, a deep learning method. Transformer [2] is a neural network architecture demonstrating high accuracy in Natural Language Processing. The Transformer consists of an encoder and a decoder. The Encoder calculates self-attention, representing the correlation strength among subjects such as words. The Decoder predicts the collection of the subjects based on the masked target-target self-attention, the source-target attention, and the output obtained from the Encoder. Transformer is a basic architecture of a large language model, e.g., ChatGPT, but we can apply it to other purposes.

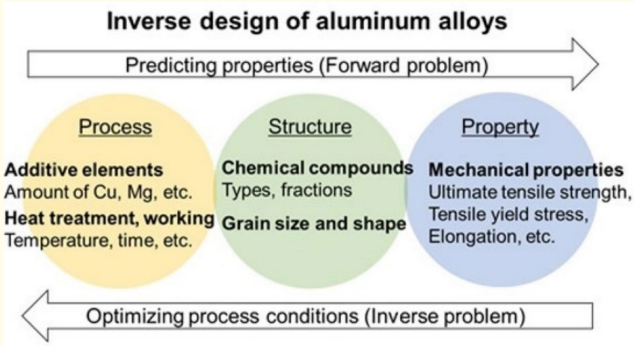


***Figure 1:*** Relationship between process, structure and property.

## Transformer Architecture

The Transformer is a deep learning model for various tasks, including natural language processing, image recognition, and time series prediction. The phase fractions of aluminum alloys vary with temperature, but predicting phase fractions from low to high temperatures is similar to forecasting a time series problem. Thus, we utilized the architecture of the Transformer used for forecasting a time series problem to predict phase fractions from the alloy compositions of aluminum alloys. The only modification made to the standard Transformer model is processing the Input Embedding for the Encoder and Decoder.

As shown in Table 1, the alloy composition consists of 13 values. We directly fed this 13-value set into the Input Embedding of the Encoder. In the Encoder's Input Embedding, we used a fully connected layer combined with an activation function to transform each of the 13 alloy composition values into a 1536-dimensional embedding vector. The parameters of this fully connected layer are subject to optimization during training. We design the Decoder to output phase fractions; thus, it generates 11-dimensional vectors for each of 70 temperature points, ranging from 100°C to 790°C in 10°C increments. Each value in the 11-dimensional vector represents the phase fraction of one of the 11 predefined phases used in the phase fraction calculations, and the sum of phase fractions at each temperature point is constrained to be 1. In the Transformer architecture, it is necessary to maintain a consistent dimensionality for the embedding vectors between the Encoder and Decoder to perform Attention calculations. Therefore, in the Decoder's Input Embedding, we employed a combination of a fully connected layer and an activation function to transform the 11-dimensional vector into a 1536dimensional embedding vector. The parameters of this fully connected layer are subject to learning, similar to those in the Encoder.

The Transformer architecture involves multiple hyperparameters. In the model used for this study, the hyperparameters are as follows: number of Heads in Multi-Head Attention: 24, number of Encoder Stacks: 2, and number of Decoder Stacks: 3.

## Training Data

Thermo-Calc was used to calculate the phase fractions of 6000 series aluminum alloys based on the CALPHAD method. When calculating phase fractions, we defined 11 phases consisting of Liquid(Al), Al, $Al_9Fe_2Si_2$, $Al_8Fe_2Si_2$, $Al_{15}(Mn, Fe)_3Si_2$, $Mg_2(Si, Sn)$, $(Al, Si)_3Ti$(Lt), $(Al, Si)_{11}Cr_4$, $DO_{23}$-$Al_3$(Ti, Zr), Liquid(Solder), and Pb. Based on these phases, we generated 841,689 training data points. In the Japanese Industrial Standards (JIS), which set the typical alloy standards in Japan, specific 6000 series of aluminum alloys include the 6010, 6013, 6060, 6061, 6063, 6066, 6070, 6101, 6181, and 6351 series. For each of these series, we specified the upper and lower limits for the values of the alloying elements, as shown in Table 1. The alloying elements include 12 types: Si, Fe, Cu, Mn, Mg, Cr, Zn, Ti, Zr, B, Bi, and Pb. We randomly selected alloys that meet the standard specifications to generate the training data. For each selected alloy, we randomly determined the composition of each alloying element within the specified upper and lower limits. We can determine the composition of Al to make the sum of all alloying elements become 1. Using this set of 13 values to define the alloy composition, we calculated the phase fractions using Thermo-Calc. Using this established set of 13 values to determine the alloy composition, we calculated the phase fractions using Thermo-Calc. For 70 temperature points ranging from 100°C to 790°C in 10°C increments, we used the phase fraction values from the 11 pre-defined phases as the training data. The reason for obtaining values at 10 °C intervals is to optimize memory usage and reduce computation time during inference with the trained model.

We prepared 811,023 training data samples and 30,666 validation data samples.

| ***Element*** | ***Si*** | ***Fe*** | ***Cu*** | ***Mn*** | ***Mg*** | ***Cr*** | ***Zn*** | ***Ti*** | ***Zr*** | ***B*** | ***Bi*** | ***Pb*** | ***Al*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Min | 0.60 | 0.00 | 0.60 | 0.20 | 0.80 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | 0.00 | Bal. |
| Max | 1.00 | 0.50 | 1.10 | 0.80 | 1.20 | 0.15 | 0.25 | 0.10 | 0.05 | 0.05 | 0.05 | 0.05 | Bal. |

***Table 1:*** Upper and Lower Composition Limits of JIS Standard 6013 Series Aluminum Alloy.

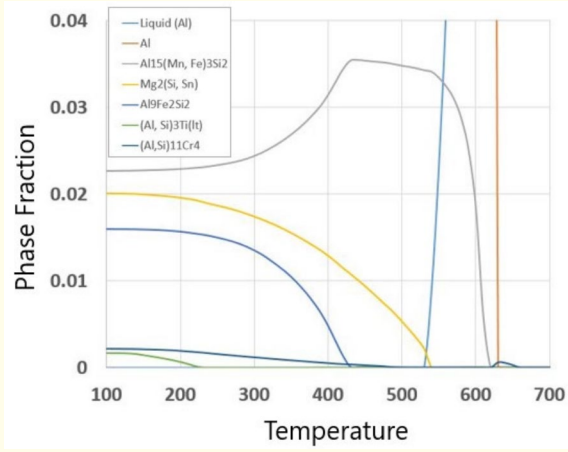


***Figure 2:*** Example of Phase Fraction Calculation.

Note that when visualizing the phase fractions, as shown in Fig. 3, we display values at 1 °C intervals, which have been interpolated linearly.

## Model Training

When treating the prediction of phase fractions in aluminum alloys as a time series forecasting problem, we should note the following points:

a) In aluminum alloys, 11 phase fractions are defined for each temperature. Therefore, it is necessary to simultaneously forecast 11 data points as a time series rather than just one.
b) The range required for phase fraction prediction spans a broad order of magnitude, ranging from $10^{-3}$ to 1.
c) As a target for time series forecasting, it is crucial to consider that the shape of phase fractions can exhibit discontinuous changes, disappearances, and generation.

In the context of time series prediction problems, it is common to use loss functions such as Mean Absolute Error (MAE), Mean Squared Error (MSE), and Mean Squared Log Error (MSLE) for model training. However, for the above reasons, using such simple loss functions did not yield satisfactory results for phase fraction prediction. Therefore, we found that a combination of multiple loss functions is effective. We will explain Further details in Section 5.

## Results and Discussion

In this chapter, we begin by examining the efficacy of various loss functions in training for phase fraction predictions. We then explore the applicability of architectures other than the Transformer to these predictions. Lastly, we verify the time required for model inference.

### *Loss Functions*

When using MAE as the loss function as shown in Equation (1), we should calculate the error as the absolute difference between the actual and predicted values.

$$Loss_{MAE}(y^{true}, y^{pred}) = \frac{1}{11} * \frac{1}{70} \sum_{i=1}^{11} \sum_{t=1}^{70} |y_{i,t}^{true} - y_{i,t}^{pred}| \quad (1)$$

In the equations mentioned above, the subscripts $i$ in $y_{i,t}$ represents the 11 phase fractions to be predicted, and the subscript t represents the temperature of 70 points for each phase fraction. When learning multiple phase fractions with different scales simultaneously, such as in the prediction of aluminum alloy phase fractions, there is a risk that learning may not proceed accurately for the following reasons. The phase fraction values of solid and liquid aluminum phases are close to 1, which is significant, while the phase fraction values of other phases are small, around 10-3. Therefore, the error derived from the solid and liquid aluminum phases becomes dominant, and there is a risk that the learning may not proceed as the error for phases with small phase fraction values is underestimated. MSE calculates the error as the square of the difference between the true and predicted values, making MAE's shortcomings more pronounced.

On the other hand, when using MSLE as the loss function as shown in Equation (2), the error is calculated as the square of the difference between the logarithmic values of the actual and predicted values.

$$Loss_{MSLE}(y^{true}, y^{pred}) = \frac{1}{11} * \frac{1}{70} \sum_{i=1}^{11} \sum_{t=1}^{70} (\log(y_{i,t}^{true} + 1) - log(y_{i,t}^{pred} + 1))^2 \quad (2)$$

Therefore, it is suitable when the possible scale of the variables to be learned is broad, and it is beneficial for learning data like the phase fractions of aluminum alloys, where the potential scale of phase fractions is wide.

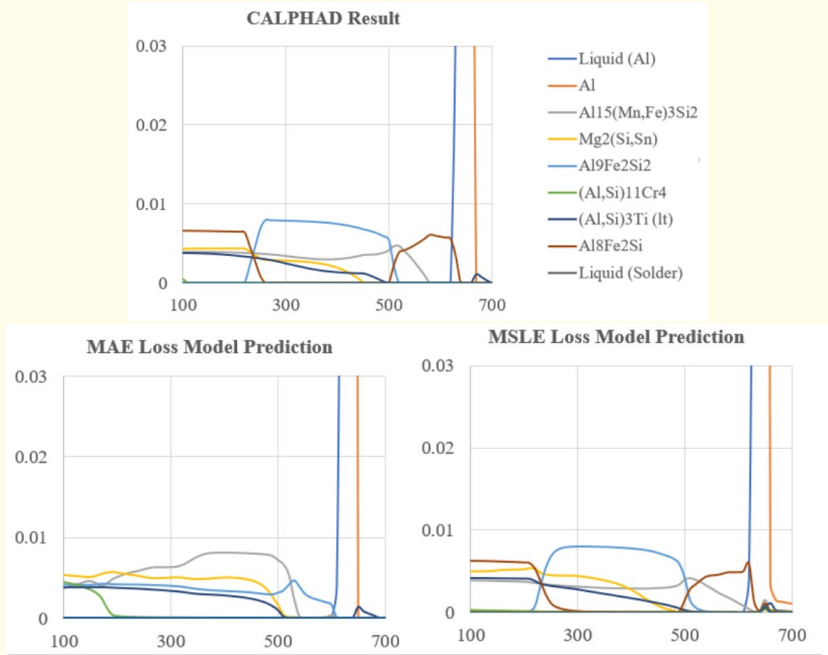


***Figure 3:*** The predictive outcomes of models trained with MAE and MSLE Loss.

Fig. 3 shows examples of the results learned with only MAE and MSLE as the loss functions. As the loss function, the figure indicates that neither MAE nor MSLE has accurately predicted the phase fractions. The model trained with MAE as the loss function produced predictions where the phase fraction morphology was overall inappropriate. The model trained with MSLE did not replicate the discontinuous changes in the phase fraction, and the phase fraction of the orange Al did not become zero around 700°C.

The vertical axis represents the phase fraction, while the horizontal axis represents temperature.

Therefore, in addition to MAE and MSLE, we added the following to the loss function during learning.

(a) ***Errors related to Cross-entropy***: By definition, each phase fraction value of all phases is 0 or more, and the sum of them should be 1.0. Therefore, it is possible to interpret the problem of predicting multiple-phase fractions as a problem of predicting a probability distribution. In this case, it is possible to employ the cross-entropy function used in multi-class classification learning as the error function. Equation (3) shows the cross-entropy function.

$$Loss_{CrossEntropy}(y^{true}, y^{pred}) = -\sum_{i=1}^{11}\sum_{t=1}^{70} y_{i,t}^{true} \log y_{i,t}^{pred} \quad (3)$$

(b) ***Errors related to temperature dependence***: The physical and mechanical properties of alloys are significantly influenced by the types and distributions of phases present within them. Consequently, in alloy manufacturing, it is essential to accurately predict the temperature-dependent phase stability. At specific temperatures, particularly those related to the formation and disappearance of phases that delineate the phase existence range, it is preferable to predict the phase fraction with higher precision than at other temperatures. Therefore, we calculated the error between the actual values and the model-predicted values using MSLE at specific temperatures $T^*$ where the phase fraction value becomes zero. In the loss function denoted as Loss MSLE in equation (2), the subscript $t$ uniformly calculates the loss across the entire temperature range from 100°C to 700°C. In contrast, the temperature-dependent loss function Loss Temp in equation (4) computes the error specifically at the particular temperature $T^*$ where the phase fraction is zero, rather than across the entire temperature range.

$$Loss_{Temp}(y^{true}, y^{pred}) = \frac{1}{11} * \frac{1}{K}\sum_{i=1}^{11}\sum_{T^*=1}^{K} \left(\log\left(y_{i,T^*}^{true} + 1\right)\right) - \log\left(y_{i,T^*}^{pred} + 1\right))^2 \quad (4)$$

In equation (4), the temperature subscript $T^*$ refers to the specific temperatures where the phase fraction value becomes zero, and K represents the total number of such temperatures considered.

(c) ***Error related to the difference***: The phase fraction graph tends to have discontinuous changes and rapid increases and decreases. To accurately predict such changes in the phase fraction, it is considered adequate to use the difference in phase fractions as an error. Therefore, one can calculate the increase or decrease from the adjacent temperature as a difference for the actual values of the phase fractions at each temperature. We can use the same method to calculate the model's prediction results difference. By using MSLE as the error function for these two differences, one can calculate the error related to the difference.

$$Loss_{Diff}(y^{true}, y^{pred}) = \frac{1}{11} * \frac{1}{69}\sum_{i=1}^{11}\sum_{t=1}^{69} (\log(y'^{true}_{i,t} + 1) - log(y'^{pred}_{i,t} + 1))^2 \quad (5)$$

In equation (5), y' represents the difference with the adjacent temperature. Here, when phase fractions are given for 70 discrete temperatures, the difference in phase fractions for adjacent temperatures can be obtained at 69 points, which is 70-1.

The loss function for phase fraction, $L_{Phase}$ was defined as in equation (6) by combining equations (1) through (5).

$$L_{Phase} = C_1 * L_{MAE} + C_2 * L_{MSLE} + C_3 * L_{CrossEntropy} + C_4 * L_{Temp} + C_5 * L_{Diff} \quad (6)$$

Here, the subscripts $C_1$ through $C_5$ are coefficients for adjusting the scale of each loss.

Fig. 4 shows examples of the results learned with Phase Loss as the loss functions. The model trained with Phase Loss as the loss function was found to be very close to the results calculated by the CALPHAD method. Table 2 lists the results calculated for each loss function on the validation data for each model. It can be observed that the model trained with Phase Loss produces the best results in all cases.

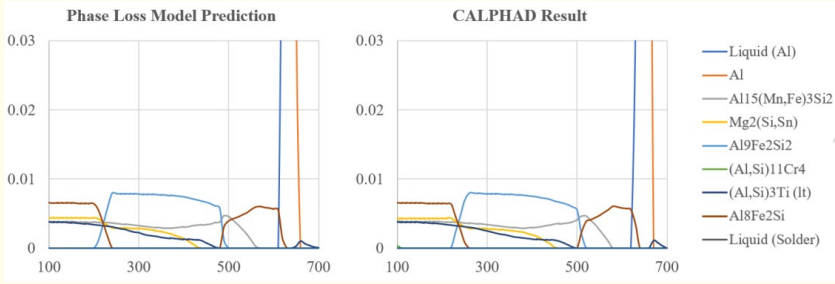


***Figure 4:*** The predictive outcome of model trained with Phase Loss.

The vertical axis represents the phase fraction, while the horizontal axis represents temperature.

| | ***MAE Loss Model*** | ***MSLE Loss Model*** | ***Phase Loss Model*** |
|---|---|---|---|
| MAE Loss | $6.17 * 10^{-4}$ | $5.96 * 10^{-4}$ | $1.73 * 10^{-}4$ |
| MSLE Loss | $3.64 * 10^{-5}$ | $3.14 * 10^{-5}$ | $1.88 * 10^{-}5$ |
| Diff Loss | $1.61 * 10^{-7}$ | $1.57 * 10^{-7}$ | $7.30 * 10^{-}8$ |
| Entropy Loss | 16.0 | 14.6 | 14.5 |
| Temp Loss | $8.24 * 10^{-5}$ | $4.66 * 10^{-5}$ | $1.58 * 10^{-5}$ |

***Table 2:*** Comparison of Losses Among Models on Validation Data.

### *Architecture other than Transformer*

As noted in the previous section, Phase Loss is highly effective in learning phase fractions. Although the Transformer is a effective model, when the sequence length is denoted as L, its computational complexity is known to increase at $O(L^2)$. This results in longer computational times and increased memory requirements when processing extensive data sequences. In this context, we evaluated the accuracy of architectures lighter than the Transformer when trained with Phase Loss. Specifically, we assessed the Seq2Seq [3] model and a network composed solely of fully connected layers.

Table 3 lists the results calculated for each loss function on the validation data for each architecture. It can be observed that the Transformer architecture with Phase Loss produces the best results in all cases. The models of Full Connected and Seq2Seq generally exhibit high loss values. Therefore, even when trained with Phase Loss, applying them for phase fraction predictions is challenging.

| | *Full Connected* | *Seq2Seq* | *Transformer* |
|---|---|---|---|
| MAE Loss | $4.73 * 10^{-3}$ | $1.89 * 10^{-3}$ | $1.73 * 10^{-}4$ |
| MSLE Loss | $3.80 * 10^{-4}$ | $3.64 * 10^{-4}$ | $1.88 * 10^{-}5$ |
| Diff Loss | $5.81 * 10^{-7}$ | $1.33 * 10^{-6}$ | $7.30 * 10^{-}8$ |
| Entropy Loss | 18.8 | 16.3 | 14.5 |
| Temp Loss | $1.82 * 10^{-4}$ | $6.48 * 10^{-4}$ | $1.58 * 10^{-5}$ |

***Table 3:*** Comparison of Losses Among Architecture on Validation Data.

### *Time required for model inference*

For models trained with Phase Loss, we measured the time required for predictions on both GPU and CPU. We then compared these results with those of CALPHAD. The findings are documented in Table 4. We confirmed that when using a GPU, the Transformer model can calculate phase fractions more than 100 times faster than ThermoCalc.

| *Model* | *Calculation Time using GPU* | *Calculation Time using CPU* |
|---|---|---|
| CALPHAD | 51 | 51 |
| Transformer | 0.49 | 4.0 |

***Table 4:*** Comparison of Calculation Time (seconds).

## Conclusion

Firstly, transformer architecture was applied to reproduce the phase fractions calculated by the CALPHAD method. Secondly, the deep learning method based on Transformer architecture reproduces the phase fractions calculated by the CALPHAD method. Finally, our trained model can calculate thermodynamic equilibrium states more than 100 times faster than the CALPHAD method.

## References

1. L Kaufman and H Bernstein. “Computer Calculation of Phase Diagrams”. Academic Press Inc (1970).
2. A Vaswani., et al. “Attention is all you need”. In Advances in Neural Information Processing Systems (2017): 6000-6010.
3. Ilya Sutskever, Oriol Vinyals and Quoc V Le. “Sequence to sequence learning with neural networks”. Advances in neural information processing systems 27 (2014).