EPJ Web of Conferences **321**, 02009 (2025) https://doi.org/10.1051/epjconf/202532102009
APITECH-VII-2025

Neural network for generating composition and parameters of metal alloys based on a given range of known and unknown parameters

Dmitrii Zhuro*, *Dmitry* Viatkin *and Andrey* Tsykarev

Northern Arctic Federal University, Arkhangelsk, 163002, Russia

Abstract. This paper describes the development of a generative-adversarial neural network for generating metal alloy compounds with given parameters. The resulting alloy is described by 19 parameters: 14 describe the alloy composition and 5 describe the alloy properties. At the stage of data preparation the parameters are normalized to the range from 0 to 1. The generator in the generative-adversarial network has 4 input layers. The first input layer receives noise to generate different realistic parameters for the same input values. The second input layer is a mask describing the known and unknown parameters. To the third input layer, the minimum acceptable parameter values are passed. To the fourth input layer of the generator the maximum allowable values of parameters are transferred. Based on the input parameters, at the output of the generator we get 19 parameters describing the alloy. The result of the generator is checked by the discriminator for the reliability of the prediction. The discriminator has 4 input layers. The first one receives the prediction made by the generator. The other 3 inputs receive data from the 2nd, 3rd and 4th input layers of the generator. The generative-adversarial neural network is capable of generating the composition and properties of alloys with an average absolute error of 0.082 units relative to the normalized range of test data parameters, i.e. with an accuracy of 91.8% relative to the real value.

1 Introduction

Materials science is a broad branch of science that solves many problems. Materials design is one of them. The development of the scientific, technical and social side of progress requires new materials for use in various human activities [1- 4]. And one of the important directions in materials design is the development of new metal alloys [1, 4, 5]. One of the most challenging aspects is the development of a mathematical model for several metal alloys, which takes into account the interaction of different alloy components and the effect of these interactions on the final material properties [5, 6].

Advances in artificial intelligence and machine learning algorithms have made it possible to automate the search for dependencies between alloy components and automate the calculation of the influence of the components on the final material properties [1, 7-10]. For

* Corresponding author: d.zhuro@narfu.ru

a long time, machine learning and artificial intelligence capabilities were used only for predicting the properties of alloys based on their composition, since creating materials based on their chemical composition was a difficult task [1, 10].

The capabilities of generative-adversarial neural networks allow generating data according to given parameters and constraints [11, 12]. This approach can also be applied to the design of materials with given parameters and properties.

Next, the possibility of using generative-adversarial networks to design metal alloys with user-defined properties and composition constraints will be discussed.

2 Experimental section

2.1 Materials

In this paper, an open access dataset describing the properties and composition of metal alloys (RN. Mechanical properties of low alloy steels) is used. This dataset includes 915 rows describing metal alloys containing different compositions and subjected to different operating conditions. Each alloy is characterized by 20 parameters including: alloy identification code, elemental content C, Si, Mn, P, S, Ni, Cr, Mo, Cu, V, Al, N, Ceq, Nb + Ta, temperature (°C), 0.2% test stress (MPa), tensile strength (MPa), elongation (%), and area reduction (%).

Alloy code - a description of the alloy grade. This parameter is important when dividing the dataset into training dataset, test dataset and verification dataset. After splitting, this column is deleted. The alloy grades from the training dataset were not present in the test and validation datasets. There are a total of 95 alloy grades in the dataset. The training dataset contains the description of 76 alloys and 729 samples, the test dataset consists of 9 alloys and 88 samples, and the validation dataset consists of 10 alloys and 98 samples. After splitting into training, test and validation datasets, the column “Alloy Code” is removed from the dataset. Thus, only the numerical parameters of the alloys remain.

Due to the wide range of values of the other 19 parameters, it becomes difficult to analyze such data and train the neural network. To normalize the values of each parameter, a normalization method was applied to the range from 0 to 1 based on the training dataset. In calculating these parameters, test and validation datasets were not used to avoid possible leakage of information from the test set to the training set.

2.2 Methods

Generative adversarial networks (GAN) is a teacherless learning algorithm that consists of two neural networks: a generator and a discriminator. The generator is trained to create new data samples, while the discriminator is trained to distinguish correct samples from the generated ones [11, 12].

A summary of the order of the training steps of the generative-adversarial network for this study is summarized in Figure 1.

During training, the discriminator performs the task of classifying real and false data generated by the generator. The discriminator is trained on two batches of data in one training step of the generative-adversarial network: one batch contains real data labeled “real data”, and the other batch contains generator-generated data labeled “false data”.

Then, after training the discriminator on real and generated data, the generator begins to be trained to generate more realistic data.

After that the discriminator learnability is switched off. The input data together with random noise is fed to the input of the generative-adversarial network. This noise is used to

generate different variants from the same data. Inside the generative-adversarial network, the generator creates data which is immediately sent to the discriminator. At this point, the generated data is labeled as real data and the generator continues training to create more realistic data.

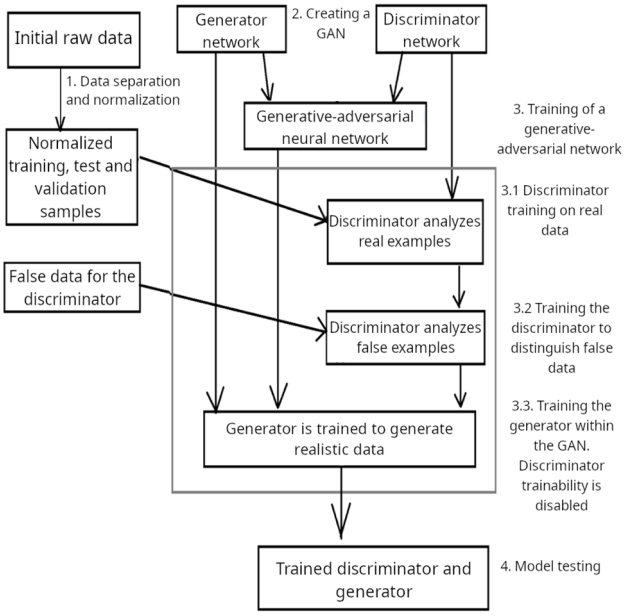

Fig. 1. Stages of training a generative-adversarial network.

The generator obtained after training can be used to generate the required data. The discriminator trained in the process can serve to check the authenticity of the generated data. For realistic objects created by the generator, the discriminator produces a higher score and for less realistic objects, it produces a low probability score.

A generative-adversarial network has two main parts: the generator and the discriminator, which work in a complementary manner to generate and evaluate the data.

The generator contains 4 input layers.

The first input layer receives noise to create a variety of realistic characteristics with the same input data. The noise consists of 100 random parameters generated based on normal distribution. The noise values range from 0 to 1.

The second input layer contains a mask that describes the known and unknown parameters. The value 0 indicates a parameter for which we know the range of values. A value of 1 indicates a parameter whose value is not important or completely unknown. The number of unknown parameters can be up to 10.

The third input layer receives the minimum allowable values of the parameters specified by the user. If a parameter corresponds to mask element 1 in the second input layer, it means that there is no information about this parameter and it must be generated. For this parameter in the third layer the value is set to -1. When using the pre-selected option to normalize the parameter between 0 and 1, the value -1 is unobtainable and can serve as a label for unknown values.

The fourth layer of the generator obtains the maximum allowable parameter values. If a parameter matches mask element 1 in the second input layer, which means that there is no information about this parameter, it must be generated. The value -1 is set for this parameter

in the fourth layer. If you use the option to normalize the parameter between 0 and 1, the value -1 is unobtainable and can serve as a label for unknown values.

In case of absence of information about the parameter, the value -1 for it should be simultaneously set on 3 and 4 input layers.

The generator output generates 19 parameters describing the alloy based on the input parameters. Unknown parameters are recovered by the generator in the normalized range.

The mean absolute error loss function and Adam optimizer are used to train the generator. The learning rate of the optimizer is 0.001 and the batch size is 20.

The discriminator analyzes the forecasts created by the generator and has 4 inputs. The first input of the discriminator is designed to evaluate the realism of the generated data, where in addition to the forecasts from the generator, real data are also added.

The other three inputs of the discriminator are fed from the input layers of the generator 2, 3, 4 without any additional processing.

A binary cross-entropy loss function and Adam optimizer are used to train the discriminator. The learning rate of the optimizer is set at 0.0005. The exponential decay rate parameter for the first moment estimates (beta_1) is 0.5. The batch size is 20.

Discriminator training in the generator-adversarial network is disabled, as only the generator training is being performed at the moment. The discriminator is trained separately. The binary cross-entropy loss function and the Adam optimizer are used to train the generative-adversarial network. The learning rate of the optimizer is set to 0.0005. The exponential decay rate parameter for the first moment estimation (beta_1) is 0.5. The batch size is 20.

Figure 2 shows a plot of the variation of the generator loss function error. It is observed that the generator error decreases, which means that the generator creates more and more realistic objects given the set parameters.

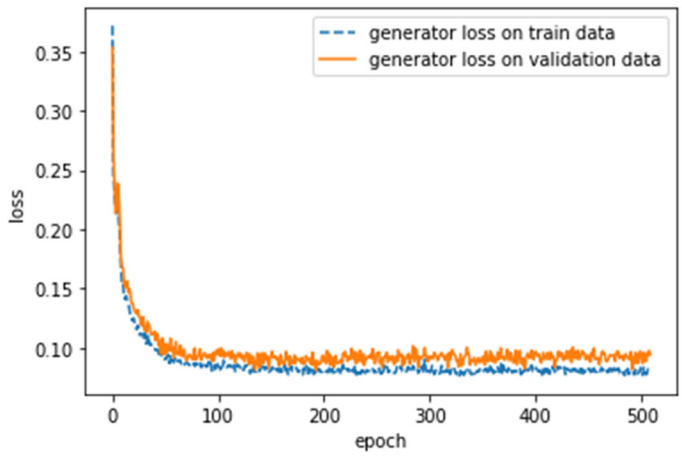

Fig. 2. Generator training process.

Figure 3 displays the learning process, showing that with each new epoch, the generator becomes more and more accurately trained to create examples with sufficient realism to allow it to outperform the discriminator. It becomes more difficult for the discriminator to distinguish the generated data from real data with each new epoch, indicating that the generator is improving the quality of its predictions.

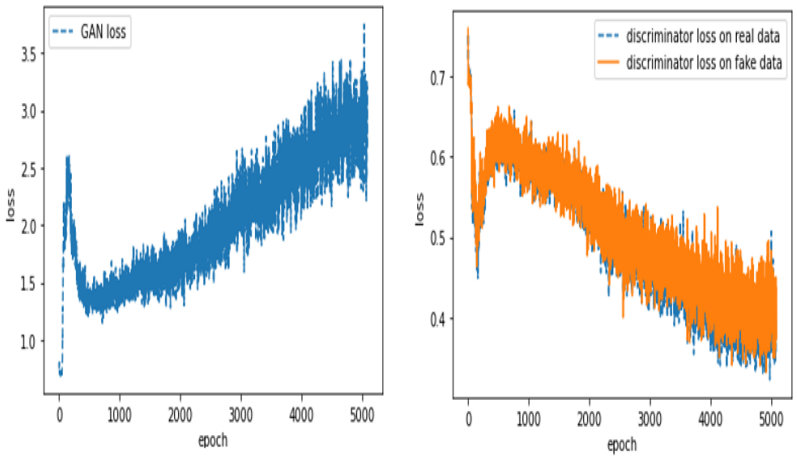

Fig. 3. Discriminator training process. Growth of error (left side) and decrease of accuracy (right side) of the discriminator.

In the initial training phase of the generator, a peak error value is noted, indicating the beginning of the process of detecting dependencies in the data. The discriminator moves from the classification of real data and random noise to the task of classifying real and generated data.

3 Training result

Optimal results were achieved using different weight generation options and neural network architectures ranging from 3,000 to 5,000 epochs. Below is the best variant achieved over 990 epochs of training.

To evaluate the generator error, the training and validation data was checked every 10 epochs where the generated data was compared with the original data. The error was calculated based on the loss function used by the generator, namely the mean absolute error.

The oscillator model with the lowest value of the loss function on the control data was retained. The discriminator was saved together with it.

The generator model with the minimum value of the mean absolute error loss function on the validation data was created, and the discriminator was saved on the same epoch. The minimum value of the mean absolute error on the validation data is 0.082. Since the parameters were normalized between 0 and 1, this value of the generator loss function can be interpreted as an accuracy of 91.8% with respect to the actual values of the calculated parameters.

The use of a lower learning rate led to an increase in the final value of the generator loss function. This may indicate that the model training process stopped at a local minimum. On the other hand, training the model at a higher speed may have decreased the probability of successful convergence of the oscillator and discriminator.

Discriminator performance can be used to assess the degree of realism of the generated material. The value returned by the discriminator increases as the realism of the material suggested by the generator increases.

An additional approach may be to train neural networks to predict a missing material property based on available data. If the generator is trained well enough to produce realistic examples, the error in predicting the missing parameter should not exceed the sum of the generator error and the error of the trained model on the predicted parameter.

By training models to predict missing parameters on real data, such models can be used to assess the realism of the generated materials.

By analyzing each parameter separately using neural networks and predicting its value based on the other known parameters, it is possible to obtain an error value for each parameter. If we summarize these values and look at the discriminator value, we can evaluate how realistic the material was generated. Fig. 4 shows a correlation plot between the discriminator value, which predicts the realism of the material on the X-axis, and the calculated error for the generated alloy under the specified conditions on the Y-axis.

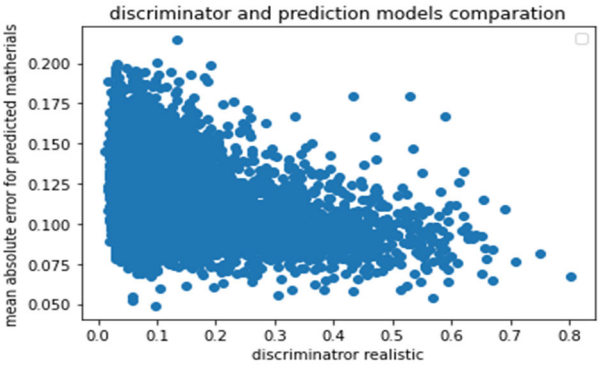

Fig. 4. Correlation of discriminator prediction and neural network error.

During the experiment, the generator created 200000 variants of material with different levels of realism. It is observed that most of the materials have a low level of realism. With the increase in the discriminator value, it is observed that the realism level of the material increases. The generated materials with discriminator value less than 0.5 are less than 50% likely to have properties close to realistic materials. Such materials are not of interest for research.

The materials that have a high level of realism and minimal error, located in the lower right part in Figure 4, attract special attention. These materials have a better chance of demonstrating the required properties and parameters. However, they need to have strict limits on the required parameters with small tolerances on all analyzed parameters. Since the generator is a neural network, it is not able to provide absolutely accurate values. For example, if the upper and lower limits of a parameter's range are 0, the neural network will provide a value close to 0 rather than exactly 0.

4 Conclusion

This paper describes the development of a generative-adversarial neural network for generating metal alloy compounds with certain characteristics.

After training, the generative-adversarial network became capable of generating the composition and properties of alloys with an average absolute error of 0.082 in the normalized range between 0 and 1 on validation data, which corresponds to an accuracy of 91.8% with respect to real values.

Generating objects with desired properties based on available data is applicable in various fields. This method is not limited only to the creation of metal alloys, but can be used in other fields as well. For example, in materials science, one can train a model on data from related fields and combine the trained generators and discriminators into ensembles. This approach will allow a more complete characterization of physical properties of materials and avoid the problem of stopping learning in the local minimum of the feature space.

Adding additional attributes to the data can significantly improve the quality of the created models. For example, such attributes can represent chemical parameters of elements from the periodic table.

Future plans include training the discussed method for related areas of materials science. It is also planned to study the possibilities of combining the trained models into ensembles.

References

1. L. Himanen, A. Geurts, A.S. Foster, P. Rinke, Data-Driven Materials Science: Status, Chal-lenges, and Perspectives, Advanced Science, 1900808 (2019).
2. M.S. Dresselhaus, G.W. Crabtree, M.V. Buchanan, MRS Bulletin **30(07)**, 518-524 (2005).
3. E. Stach, B. DeCost, A.G. Kusne, J. Hattrick-Simpers, K.A. Brown et al., Matter **4(9)**, 2702-2726 (2021).
4. G. Pintsuk, E. Diegele, S.L. Dudarev, M. Gorley, J. Henry, J. Reiser, M. Rieth, Fu-sion Engineering and Design **146(Part A)**, 1300-1307 (2019).
5. G.J. Kipouros, W.F. Caley, D.P. Bishop, Metallurgical and Materials Transactions, **37(12)**, 3429-3436 (2006).
6. R. Hardian, Z. Liang, X. Zhang, G. Szekely, Green Chemistry **22**, 7521-7528 (2020). https://doi.org/10.1039/D0GC02956D
7. P. Ball, MRS Bulletin **44(05)**, 335-344 (2019).
8. W. Sha, Y. Guo, Q. Yuan, S. Tang, X. Zhang et al., Ad-vanced Intelligent Systems **2(4)**, 1900143 (2020). https://doi.org/10.1002/aisy.201900143
9. J. Wei, X. Chu, X. Sun, K. Xu, H. Deng et al., InfoMat **1(3)**, 338-358 (2019).
10. J.S. Huang, J.X. Liew, A.S. Ademiloye, K.M. Liew, Archives of Computational Methods in Engineering **28(5)**, 3399-3413 (2020).
11. A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, A.A. Bharath, IEEE Signal Processing Magazine **35(1)**, 53-65 (2018).
12. K. Wang, C. Gou, Y. Duan, Y. Lin, X. Zheng, F.-Y. Wang, IEEE/CAA Journal of Automatica Sinica **4(4)**, 588-598 (2017).