Academic Journal of Computing & Information Science
ISSN 2616-5775 Vol. 6, Issue 11: 35-39, DOI: 10.25236/AJCIS.2023.061105

Difficulty Level Classification of Sudoku Puzzles Based on Convolutional Neural Network

Xuan Wei[1,*]

[1]*Department of Computer Science, The University of Manchester, Manchester, The United Kingdom*
[*]*Corresponding author: wx2192768229@outlook.com*

Abstract: *Sudoku is a classic logic puzzle that many people love to play. Dividing the difficulty of Sudoku puzzles helps provide Sudokus with different levels suitable for new or skilled Sudoku players. This paper proposes a model using convolutional neural networks to distinguish difficulty levels of Sudoku puzzles. Firstly, this paper uses traditional depth-first search algorithms to measure the solving steps of Sudoku, thereby labelling the difficulty of Sudoku puzzles. Then, these difficulty-labelled Sudoku data are used as training data to enable the convolutional neural network-based model to distinguish Sudokus with difficulty levels. Finally, this neural network model can correctly classify approximately 80% of the difficulty levels in testing Sudoku datasets.*

Keywords: *Deep learning, Convolutional neural network, Supervised learning, Sudoku puzzle, Depth-first search algorithm*

1. Introduction

Sudoku is a logical problem that filling other numbers according to Sudoku rules in a grid of 9 rows and 9 columns, which partially fills with numbers. Generating, solving, and verifying Sudoku puzzles can be achieved using computer algorithms. Traditional algorithms for solving Sudoku puzzles are based on deep-first search algorithms [1], and in recent years, deep learning [2] has also been proven to solve Sudoku puzzles [3]. This paper explores using deep learning networks to classify the difficulty level of Sudoku puzzles, which can help generate Sudoku puzzles with different difficulty levels.

2. Method

2.1. Dataset

The data used in this paper comes from the public dataset of 9 Million Sudoku Puzzles and Solutions. The Sudoku puzzles in this publicly available dataset are solvable, making it suitable for this paper to explore using deep learning models to classify the difficulty level of Sudoku puzzles.

2.2. Data preprocessing

This paper aims to classify three different level types of Sudoku puzzles, and it is necessary to know the difficulty of the Sudoku puzzle in the training set. The traditional method of calculating the difficulty of a Sudoku puzzle can use the depth-first search algorithm. In the process of the depth-first search algorithm, each step to assign a value to a grid is counted as one attempt. When the total number of assignment attempts in solving a Sudoku puzzle is low, the Sudoku puzzle is considered relatively simple. In contrast, when the total number of assignment attempts in solving a Sudoku puzzle is high, the Sudoku puzzle is considered relatively difficult. After using the depth-first search algorithm to count the total number of assignment attempts of all Sudoku puzzles in the dataset, the statistical results indicate that the difficulty distribution of these Sudoku puzzles ranges from 100 attempts to around 1000 attempts. Next, randomly select an equal amount of Sudoku data from those with fewer attempts number, those with medium attempts number, and those with larger attempts number as the dataset used in this experiment. Sudoku data with fewer attempts number is labelled as easy level, Sudoku data with medium attempts number is labelled as medium level, and Sudoku data with larger attempts number is labelled as difficult level. Finally, this dataset will be divided into training, validation, and test datasets for subsequent model training and evaluation experiments.

ISSN 2616-5775 Vol. 6, Issue 11: 35-39, DOI: 10.25236/AJCIS.2023.061105

2.3. Convolutional neural network

The core part of the neural network model used to classify the difficulty of Sudoku puzzles in this paper is convolutional neural networks [4]. Convolutional neural networks are a classic network structure commonly used for processing computer vision problems [5]. One of its advantages over multi-layer perceptrons [6] is that it can process spatial information, making it suitable for processing data such as images.

In theory, Sudoku puzzles can also be solved using convolutional neural networks. Although Sudoku is a logical problem rather than an image problem, there are some similarities between Sudoku and images. Firstly, each Sudoku data is a fixed-size grid structure with 9 rows and 9 columns, and there is a number or a blank mark at each grid point, which means that both Sudoku data and image data belong to dense data. Secondly, due to the rules of Sudoku, the content in each grid will have some relationship with the grid content around it, so Sudoku data also needs to consider the relationship between spatial information, the same as an image. Therefore, the Sudoku puzzle can be seen as a special image with 81 pixels, with each pixel having a specific value of 1 to 9 that has already been filled in or a blank pixel that needs to be filled. The blank pixel can be set to 0 as its pixel value. In this way, convolutional neural networks can also learn feature information from Sudokus in a similar way as images.

Figure 1, 2, and 3 shows some neural network structures designed in this paper to identify the difficulty level of Sudoku puzzles. These three models have different numbers of convolutional layers [4] and max pooling layers [7]. A Sudoku puzzle is considered a picture with very few pixels; after two pooling layers, only two rows and two columns of pixel feature maps are left. Therefore, the experiment only compares the situation between one max pooling layer and two max pooling layers. The effectiveness of these three neural network models on the Sudoku puzzle dataset used in this paper will be evaluated in the CNN structure comparison experiment section.

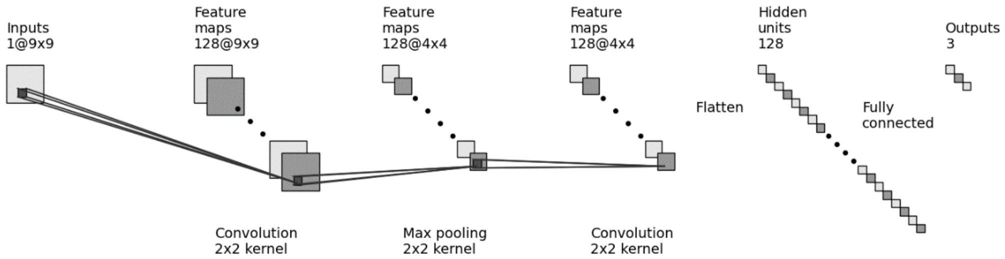

Figure 1: Neural network structure 1.

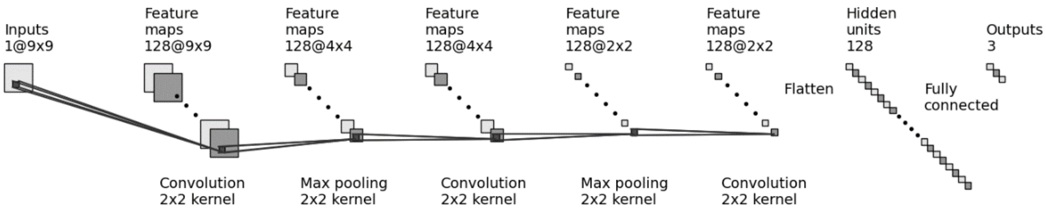

Figure 2: Neural network structure 2.

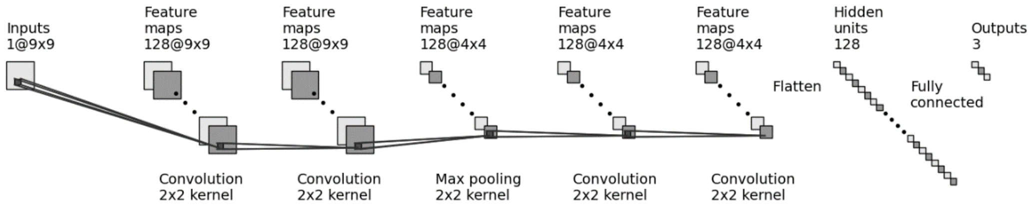

Figure 3: Neural network structure 3.

3. Experiment

3.1. Convolutional neural network structures choice experiment

The convolutional neural network experiment compared the effectiveness of the three neural network

ISSN 2616-5775 Vol. 6, Issue 11: 35-39, DOI: 10.25236/AJCIS.2023.061105

structures designed in this paper on Sudoku difficulty level classification problems. Figure 4 shows the classification performance of three trained convolutional neural network (CNN) structures on the validation set.

After approximately 20 epochs of training, the accuracy of CNN structure 1 and CNN structure 2 tends to be the same, indicating that adding more pooling layers in the neural network structure does not significantly improve the model's performance. This may be because the number of pixel features in Sudoku images is only 9 rows with 9 columns, and after two pooling layers, it becomes 2 rows with 2 columns. Although pooling layers can gather feature information, too few features can also reduce the model's effectiveness. Thus, CNN structure 2 dose not perform better than CNN structure 1.

CNN structure 3 has an improvement in accuracy compared to CNN structure 1 and 2, indicating that adding more convolutional layers can improve the model's performance. Therefore, CNN Structure 3 is selected as the neural network model for conducting post hyperparameter selection experiments and verifying the final model classification effectiveness.

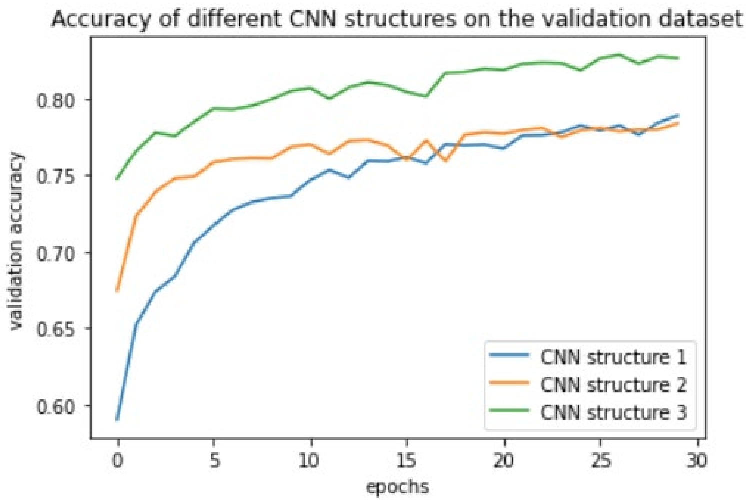

Figure 4: Neural network structure choice experiment result.

3.2. Hyperparameters choice experiment

The hyperparameter experiment is used to test and select some better parameters in the neural network model. This paper conducted the hyperparameter experiment on four hyperparameters: the training optimizer, the number of filters in the convolutional layer, the kernel size of the convolutional layer, and the batch size during training. Some possible candidate values for these four hyperparameters are shown in Table 1. Each hyperparameter experiment will be conducted independently and record the accuracy of the model with different hyperparameter candidate values on the same Sudoku validation dataset.

Table 1: Hyperparameters and their candidate values.

Hyperparameters	***Candidate values***
Optimizer	SGD (learning rate = 0.01) [8], SGD (learning rate = 0.001), Adam [9], RMSprop [10]
Filters number of convolution layers	16, 32, 64, 128
Kernel size of convolution layers	2, 3, 4, 5
Batch size	64, 128, 256

Figure 5 summarizes the results of the four hyperparameter choice experiments. First of all, the accuracy of the model using Adam or RMSprop optimizer is higher than that using SGD methods. This is because Adam and RMSprop optimizer can adjust the learning rate during the training process to better adapt to the training dataset. Therefore, the Adam algorithm is selected as the training optimizer because its performance is slightly more stable than that of the RMSprop algorithm. Secondly, the filter number of the convolutional layer is selected as 128, as the model training performance is better when the filter number is 128 than when other filter number values are selected. Thirdly, in the first half of the training epochs, the effect of different kernel sizes of convolutional layers cannot be clearly distinguished. After 15 epochs of training, the accuracy of models with kernel sizes 3, 4, and 5 decreases slightly, while the accuracy of models with kernel size 2 continues to increase. Therefore, the final model set the kernel size

ISSN 2616-5775 Vol. 6, Issue 11: 35-39, DOI: 10.25236/AJCIS.2023.061105

of the convolutional layer to 2. Finally, the performances of the model training with different batch sizes are insignificant. The final batch size setting for model training is 128, as its accuracy is slightly higher than when the batch size is set to 256, and its accuracy curve is more stable than when the batch size is set to 64.

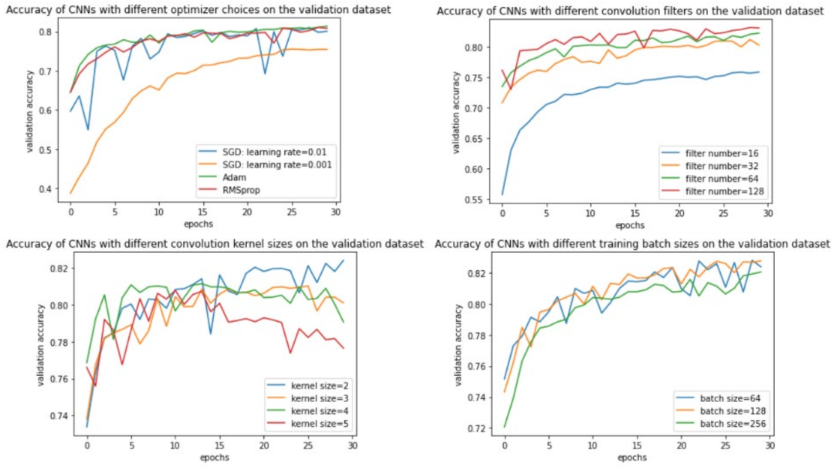

Figure 5: Hyperparameter choice experiment result.

4. Result

4.1. Training result

After selecting the appropriate neural network structure and hyperparameters, the model is trained with the training dataset for Sudoku difficulty level classification, and the validation dataset is used to monitor the changes in model performance during the training process. Figure 6 shows the model's accuracy and loss changes for both the training and validation datasets during the model training process. After several epochs of training, the model's accuracy on the training set reached over 80% and was relatively stable. The performance trend of the model on the validation set is consistent with that on the training set, and it ultimately achieved an accuracy of about 80%, only slightly lower than the accuracy on the training set by about 2%. This indicates that the model can gradually classify the difficulty level of Sudoku puzzles correctly during the training process.

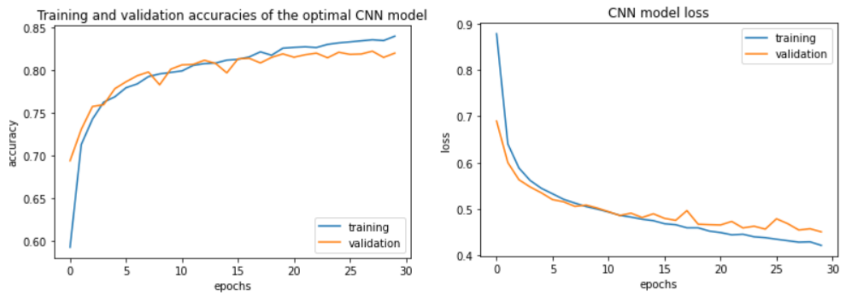

Figure 6: Accuracy and loss of the Sudoku difficulty level classification model on the training and the validation dataset during training processing.

4.2. Confusion matrix

The confusion matrix is usually used to evaluate the performance of a model in classification problems. The model used in this paper was evaluated for classification performance on the test dataset. Figure 7 shows the model's classification result of the Sudoku test dataset. Firstly, the classification accuracy is around 80%, indicating that this model can correctly classify the difficulty level of most Sudoku puzzles. Secondly, most samples are correctly classified in each difficulty level, meaning that this model has a relatively balanced classification ability for Sudoku puzzles of different difficulty levels. Finally, the evaluation results indicate that the Sudoku difficulty level classification method based on the convolutional neural network used in this paper can be used for classifying the difficulty level of Sudoku puzzles.

ISSN 2616-5775 Vol. 6, Issue 11: 35-39, DOI: 10.25236/AJCIS.2023.061105

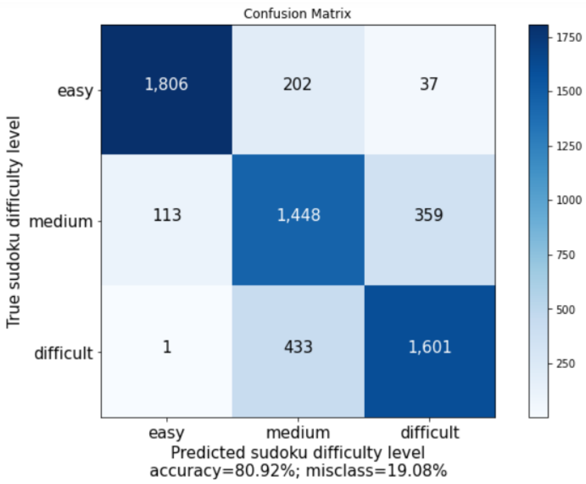

Figure 7: Confusion matrix of the Sudoku difficulty classification result on the test dataset.

5. Conclusion

The Sudoku difficulty level classifier proposes in this paper can classify Sudoku puzzles into three categories based on difficulty: simple, medium, and difficult. This Sudoku classifier can help design Sudoku puzzles suitable for people with different Sudoku-solving abilities by distinguishing the difficulty of Sudoku. In the evaluation experiment, this Sudoku classifier can correctly classify the difficulty level of approximately 80% of Sudoku puzzles. Therefore, this Sudoku classifier can be applied to distinguish the difficulty level of Sudoku puzzles. However, this Sudoku classifier does not perform well in a wide range of classification situations, such as the ten-difficulty level classification task. This may be because when there are too many classifications, Sudoku itself does not have enough features for the classification model to classify, so this model cannot be used to classify Sudoku puzzles with more difficulty level categories. Finally, future research can consider using transfer learning [11] methods to classify Sudoku puzzles to obtain better results.

References

[1] Job D, Paul V. Recursive backtracking for solving 9 9 Sudoku puzzle[J]. Bonfring International Journal of Data Mining, 2016, 6(1): 7-9.*

[2] LeCun Y, Bengio Y, Hinton G. Deep learning[J]. nature, 2015, 521(7553): 436-444.

[3] Vamsi K S, Gangadharabhotla S, Sai V S H. A Deep Learning approach to solve sudoku puzzle [C]//2021 5th International Conference on Intelligent Computing and Control Systems (ICICCS). IEEE, 2021: 1175-1179.

[4] Gu J, Wang Z, Kuen J, et al. Recent advances in convolutional neural networks[J]. Pattern recognition, 2018, 77: 354-377.

[5] Rawat W, Wang Z. Deep convolutional neural networks for image classification: A comprehensive review [J]. Neural computation, 2017, 29(9): 2352-2449.

[6] Noriega L. Multilayer perceptron tutorial[J]. School of Computing. Staffordshire University, 2005, 4(5): 444.

[7] Gholamalinezhad H, Khosravi H. Pooling methods in deep neural networks, a review[J]. arXiv preprint arXiv:2009.07485, 2020.

[8] Jordan M I, Mitchell T M. Machine learning: Trends, perspectives, and prospects[J]. Science, 2015, 349(6245): 255-260.

[9] Kingma D P, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.

[10] Ruder S. An overview of gradient descent optimization algorithms[J]. arXiv preprint arXiv:1609.04747, 2016.

[11] Pan S J, Yang Q. A survey on transfer learning[J]. IEEE Transactions on knowledge and data engineering, 2009, 22(10): 1345-1359.