Pay Attention To Solve Jigsaw Puzzles: An Exploration of Image Piece Permutation

Dilan Dinushka*
diland.sa.2023@nongrad.smu.edu.sg
School of Computing and Information Systems
Singapore Management University
Singapore, Singapore

Manusha Karunathilaka*
gmik.vidana.2023@phdcs.smu.edu.sg
School of Computing and Information Systems
Singapore Management University
Singapore, Singapore

Nicole Teo*
nicolet.2023@engd.smu.edu.sg
School of Computing and Information Systems
Singapore Management University
Singapore, Singapore

Nipuni Karumpulli*
nipunih.ka.2023@phdcs.smu.edu.sg
School of Computing and Information Systems
Singapore Management University
Singapore, Singapore

Abstract

This paper explores the prediction of permutation which was applied to an image by leveraging a dataset of 2,944 puzzles, each containing 36 pieces. The methodology to solve this problem utilizes deep convolutional neural networks (CNNs) and attention mechanisms to better understand the input embedding correlations. To further enhance the model performance, through pre-processing and feature extraction techniques, we assess the similarity of puzzle pieces and use them as the input for our proposed model. The effectiveness of the approach can be seen by the high accuracy of 97% on both the training dataset and the provided public dataset.

CCS Concepts: • **Computing methodologies** → Neural networks.

Keywords: datasets, neural networks, resnet

ACM Reference Format:
Dilan Dinushka, Manusha Karunathilaka, Nicole Teo, and Nipuni Karumpulli. 2023. Pay Attention To Solve Jigsaw Puzzles: An Exploration of Image Piece Permutation. In *Proceedings of Solving a Jigsaw Puzzle (CS712 Project '23).* ACM, Singapore, SG, Singapore, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

The area of image processing and computer vision has numerous problems that require complex methodologies and architectures to solve. One such problem is the automated solving of jigsaw puzzles. As a first step in solving this problem, this paper tries to determine the precise permutation order that has been applied to the puzzle pieces of an image. For example, the system should predict the permutations 13 and 28 used to create the left sample image and a right sample image of Figure 1 respectively. Traditional methods, while providing foundational insights, often fall short in the face of intricate puzzles involving numerous pieces with subtle distinguishing features.

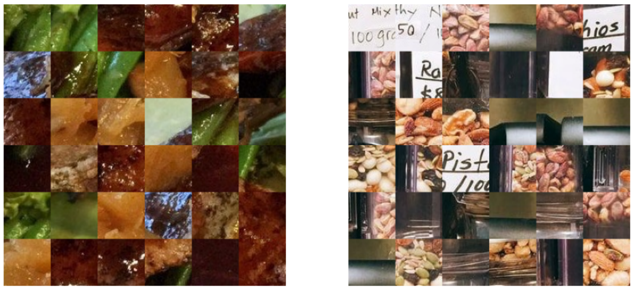

Figure 1. Sample images displaying two permutations of puzzle pieces

Our research aims to overcome these limitations by leveraging deep convolutional neural networks (CNNs) in conjunction with attention mechanisms. This allows us to craft a model adapted to navigating the complexities found within jigsaw puzzle permutations. Our data set comprises 2,944 puzzles. Each puzzle is an image disassembled into 36 labeled pieces associated with one of the 50 unique permutations.

Our methodology revolves around assessing the similarity of neighboring images within the puzzle. Higher similarity values between images indicate their proximity in the

[*] All authors contributed equally to this research.

CS712 Project '23, November 04, 2023, Singapore, Singapore
© 2023 Association for Computing Machinery.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

context of the jigsaw puzzle. This approach aligns with the conventional understanding that pieces exhibiting greater visual similarities, such as edges or patterns, are more likely to be positioned adjacent to one another within the puzzle's structure. By using techniques like deep CNN and attention mechanism, we are able to analyze and infer the image similarities. This allows our model to predict potential adjacency among puzzle pieces, aiding in finding the permutation method applied to the puzzle pieces of an image. Further, this method can be explored in the research direction of reconstructing and solving the overall puzzle.

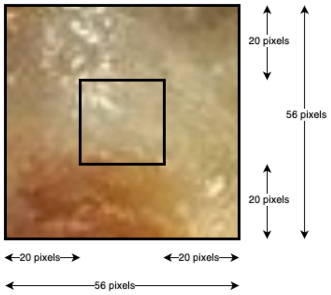

Figure 2. A puzzle piece with a margin of 20 pixels on all sides

2 Related Work

Various approaches and methodologies have conducted the exploration of automated jigsaw puzzle solving. Traditionally, edge-matching and color consistency strategies have been used, introducing limitations, especially for larger and more complex puzzles. However, recent research has incorporated deep learning to improve the solving of jigsaw puzzles.

For example, CNNs can enhance models to capture and handle key attributes from puzzle pieces. In addition, attention mechanisms have the ability to selectively concentrate on pertinent features, which improves their predictive accuracy and robustness. Our work is inspired by these advancements. In this paper we introduces a novel methodology that can be enhanced to create an accurate and efficient automated jigsaw puzzle-solving solution.

In their paper, Noroozi et al. [4] used unsupervised learning for modeling visual representations in puzzle-solving, and leveraged jigsaw puzzles as a pretext task for complex computational tasks. Moreover, Deepzzle [6] looked at image reassembly, and focused on puzzles with disrupted patterns and colors. This approach aimed to handle puzzles with considerable gaps between fragments, thereby improving algorithmic learning from fragment content.

Combining modern computational techniques such as vision transformers and deep reinforcement learning, Dosovitskiy et al. [2], Chen et al. [1] and Song et al. [7] were able to solve the traditional puzzle-solving challenges. These improvements highlight the evolving and diverse approaches within the discipline, with each offering a distinctive perspective to understand and address the inherent challenges of automated jigsaw puzzle-solving.

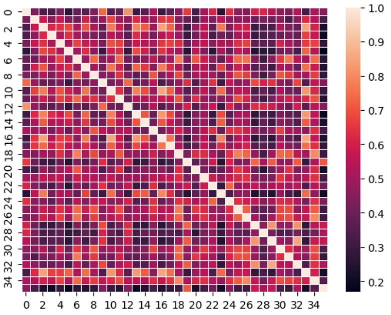

Figure 3. Similarity between puzzle piece embeddings

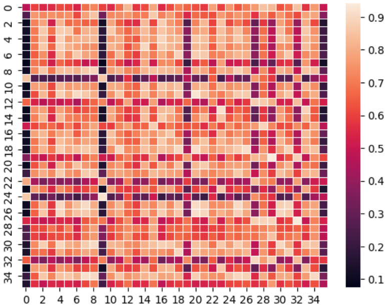

Figure 4. Similarity between left border embeddings with right border embeddings

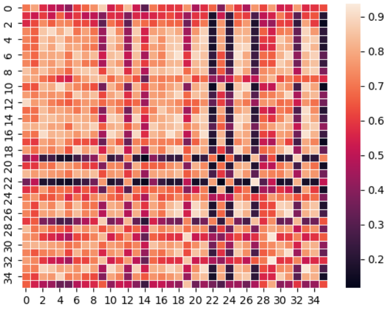

Figure 5. Similarity between top border embeddings with bot- tom border embeddings

By leveraging Generative Adversarial Networks (GANs) in puzzle solving, JigsawGAN [3] highlighted the fusion of geometric shapes and deep learning in the puzzle reconstruction process. Furthermore, Paumard et al. [5] was able to manage missing and extra fragments, showcasing advanced approaches. All of the studies mentioned above have shown the research area of automated jigsaw puzzle solving is constantly evolving, and emphasizes deep learning's role in it.

3 Proposed Approach

Our research revolves around a dataset consisting of 2,944 puzzles, each comprising 36 individual pieces. The dataset consists of specific labels indicating the permutations applied to shuffle the puzzle pieces.

3.1 Pre-processing

The pre-processing stage involves taking each image piece of a puzzle and extracting 20 pixels from the borders (left, right, top, bottom) of each piece [fig:2]. These puzzle pieces and their border pixels are then run through pre-trained ResNet-50 to generate image embeddings. The output of this generates 5 sets of embeddings, with each set representing the complete image pieces and left, right, top, and bottom borders of pieces respectively. These serve as numerical representations capturing the essence of each puzzle piece.

The next step involves computing the cosine similarity between these embeddings.

- Similarity between puzzle piece embeddings[fig:3]
- Similarity between left border embeddings with right border embeddings.[fig:4]
- Similarity between top border embeddings with bottom border embeddings.[fig:5]

This results in three separate 36x36 matrices and these matrices are stacked together, resulting in the creation of a consolidated 3x36x36 matrix. This combined matrix provides an overview of the relationships and alignment patterns among the different puzzle piece embeddings.

The model[fig:6] is fed with two core inputs: the 36 puzzle piece embeddings labeled as (`img_vecs`) and the stacked similarity matrix (`similarity_features`) derived from the preprocessing stage.

3.2 Model Architecture

The model's architecture consists of a sequential flow of operations for the inputs starting from the image piece embeddings. It begins by subjecting the tensor to three convolutional layers reducing the 36x2048 image piece embedding to a resulting 36x36x1 matrix. This output is then re-permuted back to a 1x36x36 configuration and subsequently concatenated with the initially stacked similarity matrix, yielding a 4x36x36 matrix. Following this step, a depth enhancement process, referred to as `map_extender` is executed to increase the depth of the matrix to 64x36x36. The matrix undergoes a reshaping operation, transforming it into a 2D matrix of dimensions 288x288, laying the groundwork for subsequent attention mechanisms.

Following the Vision Transformer encoder layers proposed by Dosovitskiy et al. [2], we implemented the attention mechanism to selectively highlight or suppress specific elements based on their relevance within the output matrix. Then the model proceeds by flattening the transformed matrix into a 1D array. This flattened data is then sequentially processed through three fully connected layers in the neural network architecture. Each layer plays a role in extracting and transforming the features present in the data, contributing to the creation of a refined and condensed representation.

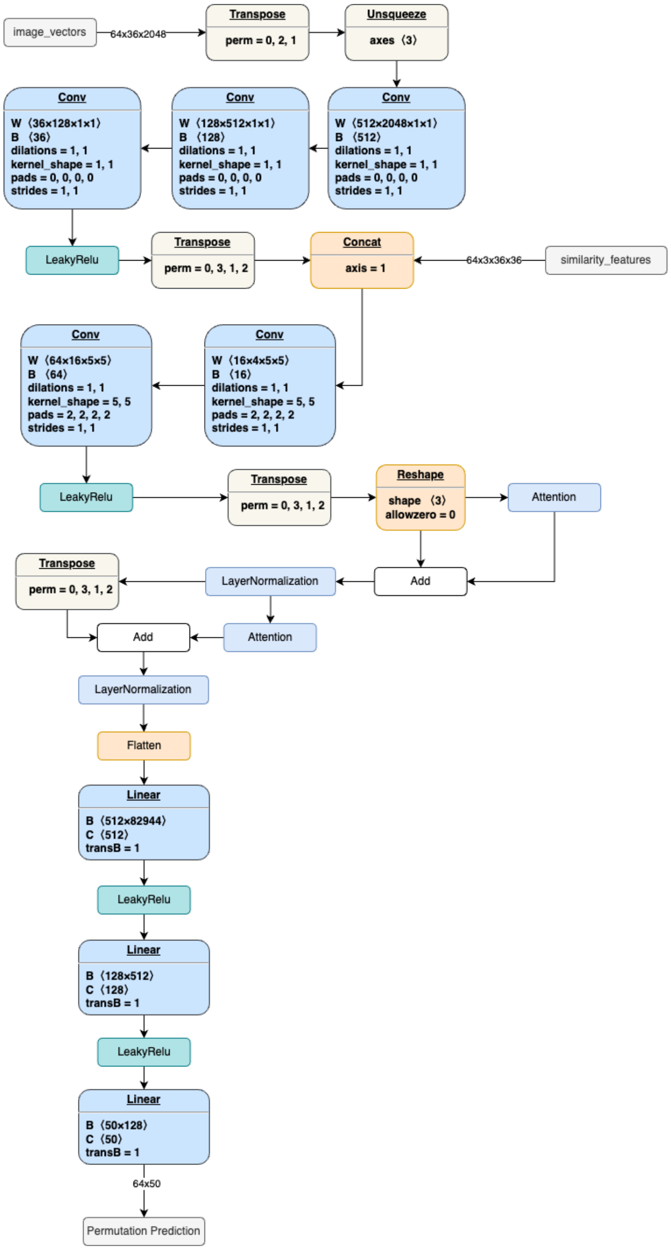

Figure 6. High-level model architecture

3.3 Training Process

In our proposed method instead of using outputs of a single model, we build 20 models that undergo 20 training iterations each, with unique random seeds and distinct training sets, aiming to encompass a wide array of representations and intricate data patterns. After training all those models individually, they collectively produce the outputs through a Softmax function which will then be used to provide a refined, probabilistic interpretation of the ensemble prediction. This final step offers a comprehensive understanding of the data, marking the conclusive phase of the methodology.

In our methodology, we employ the cross-entropy loss function [1] as our measure to assess the model's performance. Additionally, for optimizing the model's parameters, we utilize the Adam optimizer, a popular choice due to its adaptive learning rate methodology. By employing an initial learning rate and adjusting it based on the model's performance, the optimizer helps in fine-tuning the model. Furthermore, we incorporate a learning rate scheduler, specifically the ReduceLROnPlateau scheduler, which dynamically adjusts the learning rate based on a predefined threshold, optimizing the training process [fig:7] [fig:8]. This strategy involves decreasing the learning rate if the model performance reaches a plateau, enhancing the model's learning capacity.

$$\frac{1}{N}\sum_{n=1}^{N}[y_i log(p_i) + (1-y_i)log(1-p_i)] \tag{1}$$

4 Experimentation and Results

4.1 Dataset

The training data contains 2,944 36-image puzzles. The public data contains 1,466 puzzles and the private data contains 1,463 puzzles. For both data, we need to predict the permutation used to shuffle the puzzle pieces.

4.2 Metrics

The performance of the proposed approach is measured based on complete accuracy, positional accuracy, and relative accuracy. Complete accuracy is determined by the percentage of correctly predicted sequences (in its whole). Positional accuracy is determined by the percentage of correctly anticipated images, or predicted images that are in the right location even when the sequence may not be correct overall. Relative accuracy is determined by the percentage of photos that are positioned correctly in relation to their immediate neighbors. After computing complete accuracy, positional accuracy, and relative accuracy, the overall ranking is determined by a harmonic mean of the above.

During the training, we use the accuracy, overall F1 score, and label-wise F1 score to measure the individual model performances. The same metrics are used to validate the ensemble model performance as well.

4.3 Training and Validation

For the training process, we split the training data set with a 70:15:15 ratio for the training, validation, and test set respectively. To determine the validation/test accuracy, while training each model we check their performance to validation data using the mentioned metrics as well. To forecast the labels for the test set, we employ the model with the highest validation accuracy.

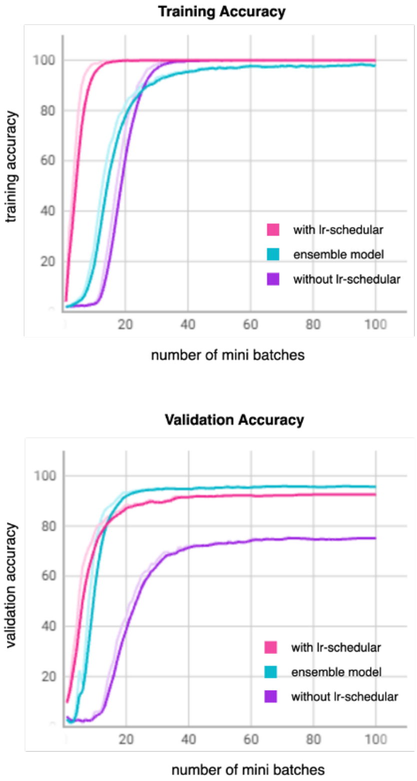

Figure 7. Comparison of training and validation accuracy for models with and without a learning rate scheduler, and an ensemble model over mini-batches

4.4 Pretrained model Selection

The model we use is simple because we are using extracted characteristics to train the model. As complicated neural network structures tend to overfit the training data set, we exclusively use simple neural network architectures. We decided to use ResNet-50.

4.5 Hyperparameters

The following hyperparameters are used in the training process to optimize for improved performance: learning rate, batch size, number of heads, and dropout.

4.5.1 Learning rate.

The fluctuation in the validation accuracy diminishes as it declines. We set the final learning rate as $1 \times e^{-4}$.

4.5.2 Batch size.

Increased batch size leads to faster training times. We set the batch size to 64.

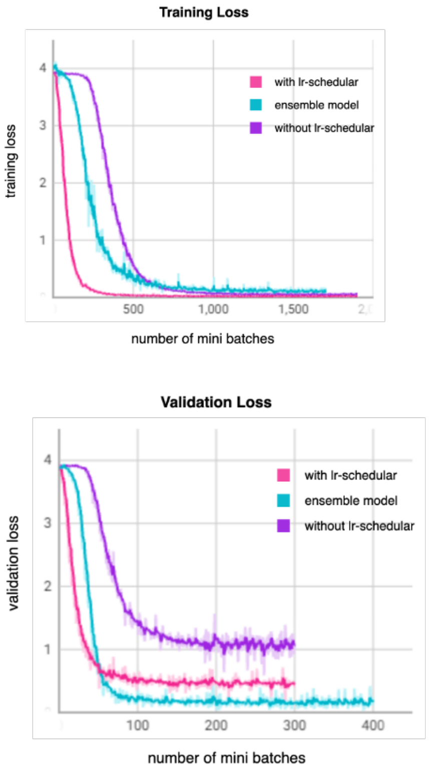

Figure 8. Comparison of training and validation loss for models with and without a learning rate scheduler, and an ensemble model over mini-batches

4.5.3 Number of heads. Having several heads per layer independent of each other means the model can learn different parameters with each head. We set the number of heads as 8.

4.5.4 Dropout. With dropout, we randomly ignore some nodes in a layer while training. This ensures that no units are codependent on one another, which inhibits overfitting. We set the dropout as 0.4.

4.6 Public Dataset

We apply the proposed model, which is based on the validation accuracy, to the public set, which consists of 52,776 sequences. On average, our selected model achieved a test accuracy of 97% on the public dataset. This is lower than the training accuracy of 99-100%. However, the validation accuracy is 97-99% as well. This indicates that the proposed approach is not likely to be overfitting.

5 Conclusion

In this paper, we have looked into predicting the image permutation given the arrangement of image pieces of an image. The proposed methodology involves a comprehensive pre-processing stage, which involves considering individual image pieces and their border pixel segments to build similarity matrices that determine the closeness between each other. The model's input preparation and convolution processes are designed to prioritize attention mechanisms, thereby enhancing feature extraction.

Through a robust training process employing thorough hyper-parameter tuning, learning rate scheduling and model ensemble technique, final predictions are optimized to provide robust prediction capabilities. The experimentation with the model showcases its efficiency through various metrics. This is evident as we have achieved a test accuracy of 97% on a public dataset containing over 52,000 sequences. These findings highlight the potential of this approach in addressing complex challenges across diverse problem-solving domains.

The positive findings of this study suggest several possible paths for future research. Firstly, exploring how well the model works with more complex puzzles that have different shapes presents an interesting challenge. Secondly, mixing the model with unsupervised learning might represent that the model does not require labels on the data, making it more usable in practical scenarios. Thirdly, looking into more complicated neural network designs and combining CNNs with other types of networks could make it work even better. Finally, testing how well the model works in different areas, like medical imaging or analyzing satellite pictures, is also worth investigating.

Acknowledgments

We extend our appreciation to Prof. Hady Wirawan Lauw for his insightful guidance and support. Our gratitude also goes to Ezekiel Ong Young, our teaching assistant, for his dedication and hands-on assistance with this work.

References

[1] Yingyi Chen, Xi Shen, Yahui Liu, Qinghua Tao, and Johan AK Suykens. 2023. Jigsaw-ViT: Learning jigsaw puzzles in vision transformer. *Pattern Recognition Letters* 166 (2023), 53–60.

[2] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929* (2020).

[3] Ru Li, Shuaicheng Liu, Guangfu Wang, Guanghui Liu, and Bing Zeng. 2021. Jigsawgan: Auxiliary learning for solving jigsaw puzzles with generative adversarial networks. *IEEE Transactions on Image Processing* 31 (2021), 513–524.

[4] Mehdi Noroozi and Paolo Favaro. 2016. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European conference on computer vision*. Springer, 69–84.

[5] Marie-Morgane Paumard. 2020. *Solving Jigsaw Puzzles with Deep Learning for Heritage.* Ph. D. Dissertation. CY Cergy Paris Université.

[6] Marie-Morgane Paumard, David Picard, and Hedi Tabia. 2020. Deepzzle: Solving visual jigsaw puzzles with deep learning and shortest path optimization. *IEEE Transactions on Image Processing* 29 (2020), 3569–3581.

[7] Xingke Song, Jiahuan Jin, Chenglin Yao, Shihe Wang, Jianfeng Ren, and Ruibin Bai. 2023. Siamese-discriminant deep reinforcement learning for solving jigsaw puzzles with large eroded gaps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 37. 2303–2311.