Pattern Recognition on Abstraction and Reasoning Challenge (ARC) Dataset using Convolutional Neural Networks

Anonymous authors
Paper under double-blind review

1 The Problem Statement

The ARC (Abstraction and Reasoning Corpus) dataset poses a significant challenge to machine learning models due to its focus on abstract reasoning and pattern recognition, which go beyond conventional feature extraction and classification tasks. Traditional neural networks often struggle to generalize to unseen tasks in this dataset because they are prone to overfitting and rely heavily on learning statistical patterns rather than abstract rules. This project aims to address these challenges by leveraging advanced deep learning architectures, such as Convolutional Neural Networks (CNNs) and Residual Networks (Res-Net), to detect and generalize patterns in the ARC dataset. The goal is to evaluate the effectiveness of these architectures in solving ARC tasks, identify their limitations, checking how well does the models generalize for abstractreasoning tasks. The broader objective is to bridge the gap between neural network performance on traditional datasets and their application to abstract reasoning, thereby understanding the generalization in deep learning models. The hypothesis that we will be considering is whether increasing the number of layersin a convolutional neural network can lead to higher training errors in small and complex datasets due to overparameterization or overfitting regularization effects.

2 Methodology

2.1 Convolutional Neural Network

The ARC dataset, consisting of input-output task pairs, is first processed to extract training and testing sets. These tasks are visualized using custom color maps to understand their structure and transformations. The core models used are Convolutional Neural Networks (CNNs) and Residual Networks (ResNet). CNNs extract spatial features from the grid-based dataset through convolutional and pooling layers, while ResNet leverages skip connections to enable deeper learning without gradient degradation. The models are trained using the Adam optimizer over 50 epochs with a batch size of 128, optimizing a loss function that quantifies the difference between predicted and target outputs. The trained models are then evaluated on unseen test tasks to measure generalization, reflecting their ability to learn abstract rules and patterns effectively. This approach combines visualization, advanced architectures, and systematic training to tackle the complex nature of the ARC dataset.

2.2 Residual Networks (Res-Net)

ResNet-18 model is to detect and recognize patterns in the ARC dataset by using task-specific preprocessing, training, and evaluation. Input grids are transformed into 10-channel binary images, resized to 224x224, and fed into the model. The ResNet-18 architecture is adapted with a custom first convolutional layer to handle 10 input channels and specialized fully connected layers for prediction. Training employs Mean Squared Error loss and the Adam optimizer, with processes for forward and backward passes, loss computation, and weight updates tracked across epochs. Predictions involve resizing inputs, passing them through the trained model, and extracting outputs. The pipeline evaluates tasks by training on matching input-output shapes and visualizes loss trends to assess performance, offering insights into optimization and generalization.

3 Experimental Results

The CNN model achieved lower error rates compared to Res-Net on the ARC dataset. While Res-Net's skip connections enable deeper learning, the simpler CNN architecture proved more effective in capturing ARC's spatial patterns. Both models faced challenges with highly abstract transformations, indicating dataset complexity.

3.1 Convolutional Neural Network

We utilized Convolutional Neural Networks (CNNs) to recognize patterns in the ARC dataset by designing a model capable of extracting spatial and abstract features from input-output grids. The process began with pre-processing the dataset, including normalization and resizing, to ensure compatibility with the network. The CNN architecture comprised convolutional layers for feature extraction, pooling layers for dimensionality reduction, and fully connected layers for decision-making.

During training, the CNN identified spatial hierarchies and relationships within the grid-based tasks, learning how input grids transform into output grids. Techniques such as ReLU activation functions and dropout were employed to enhance learning and prevent overfitting. The model's performance was iteratively optimized using back-propagation and a suitable loss function, enabling it to generalize patterns and transformations across diverse tasks in the ARC dataset. This approach showcased the effectiveness of CNNs in addressing the dataset's abstract reasoning challenges.

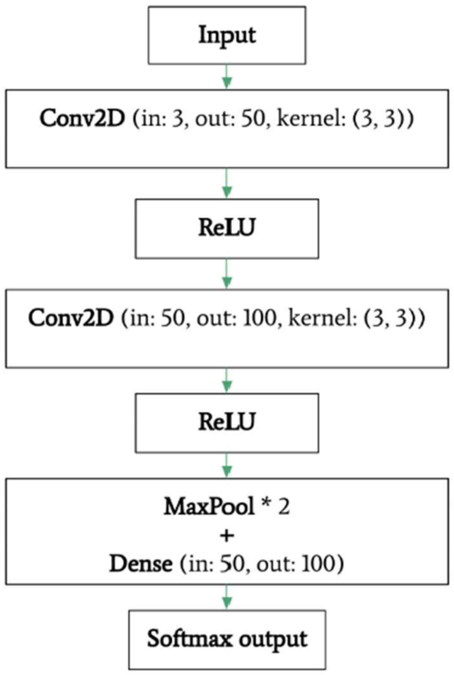
Input
Conv2D (in: 3, out: 50, kernel: (3, 3))
ReLU
Conv2D (in: 50, out: 100, kernel: (3, 3))
ReLU
MaxPool * 2
+
Dense (in: 50, out: 100)
Softmax output

Figure 1: Structure of CNN

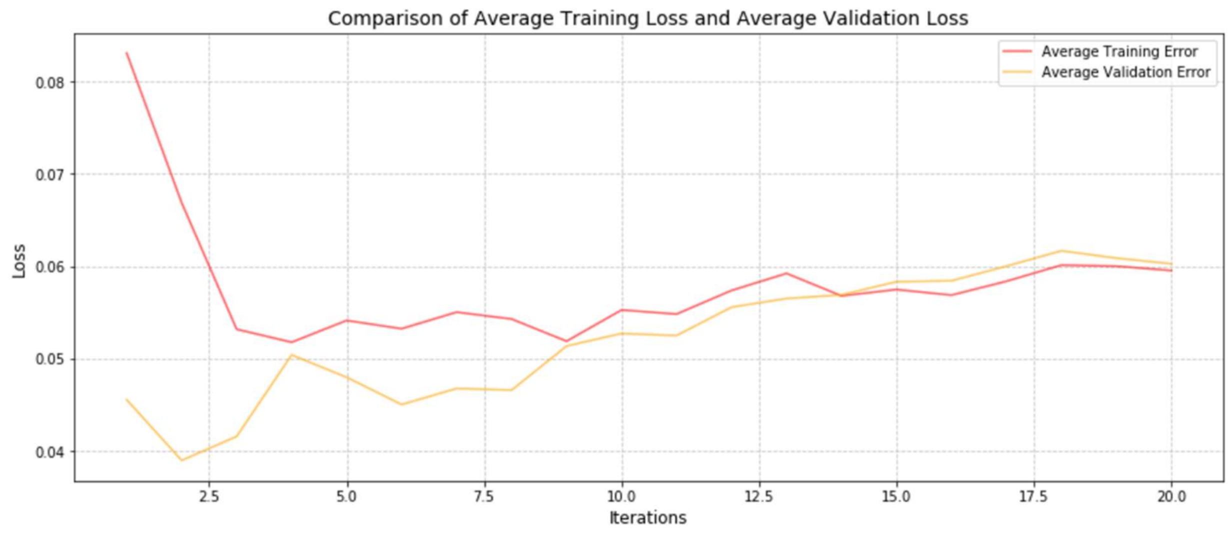

Figure 2: Validation loss V/S Train Loss of CNN

3.1.1 Outputs

The following are the Train input vs Train output and Test input vs Test output.

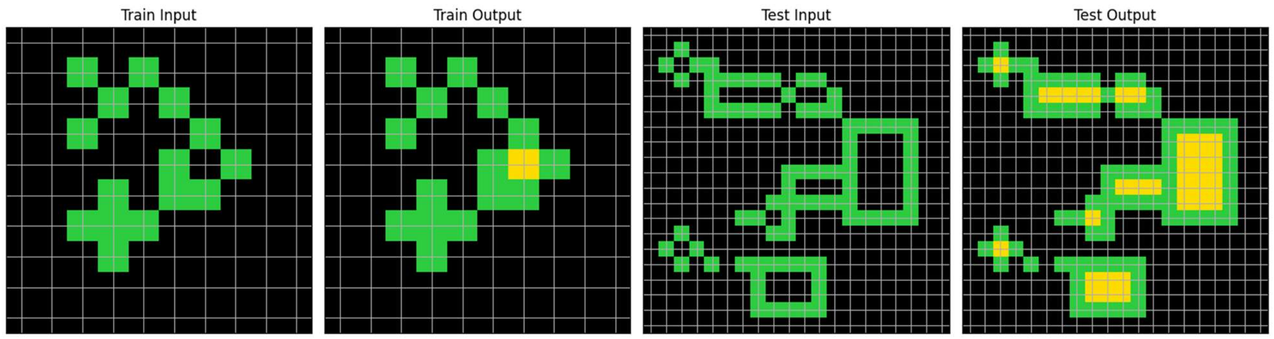

Figure 3: Pattern 1

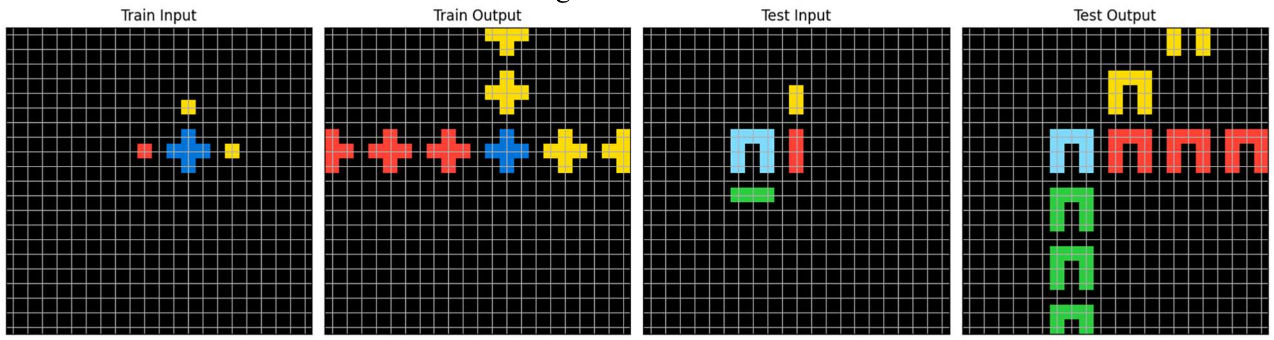

Figure 4: Pattern 2

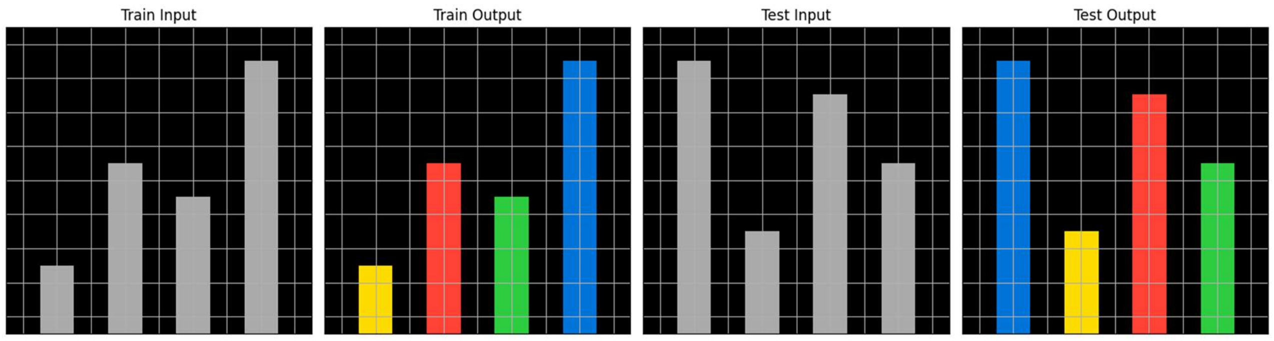

Figure 5: Pattern 3

3.2 Residual Network (Res-Net)

The input tasks, represented as grid-based images, are processed through the network's convolutional layers to extract spatial and structural features. Residual blocks in ResNet-18 allow the model to efficiently learn these patterns by bypassing certain layers, ensuring that essential information is not lost and gradients flow smoothly during backpropagation. This architecture enables the network to capture both low-level patterns (e.g., colors, shapes) and high-level relationships (e.g., transformations and rules) essential for predicting outputs. Despite its depth, ResNet-18 struggles with some complex transformations in the ARC dataset, highlighting the challenges of generalizing abstract reasoning tasks.

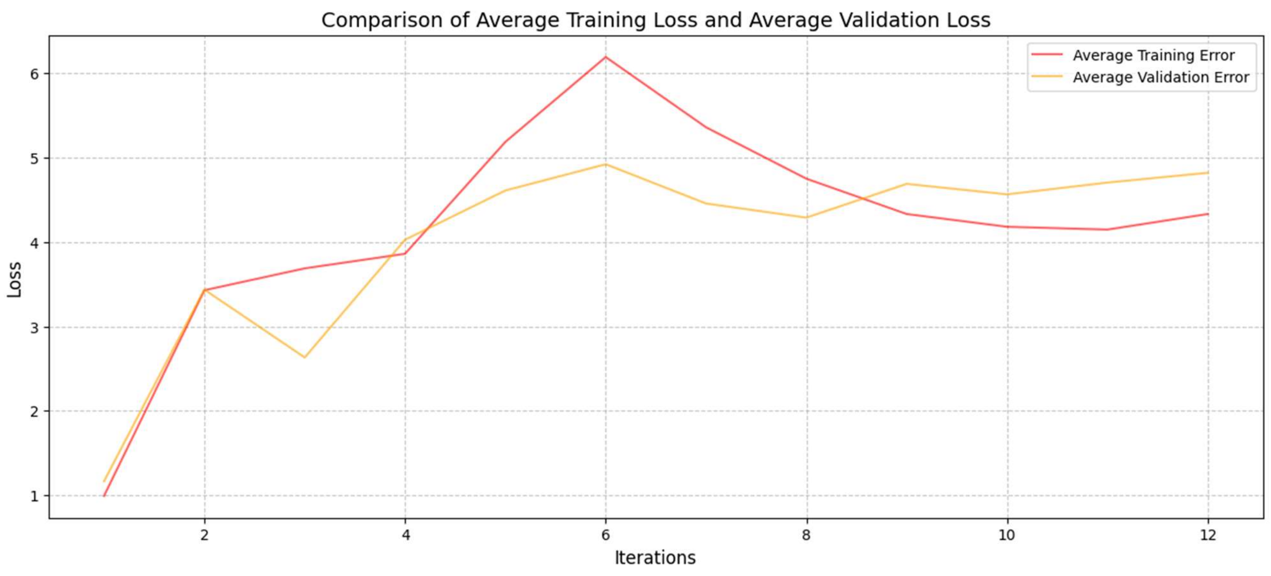

Figure 6: Validation Loss V/S Train Loss of Res-Net

4 Conclusion and Future Works

This study explored the impact of model complexity on pattern recognition and generalization using convolu- tional neural networks (CNNs). A 2-layer CNN and a ResNet-18 model were trained on a small dataset, with results showing significantly lower training error for the 2-layer CNN (MSE: 0.06) compared to ResNet-18 (MSE: 4). These findings support the hypothesis that increasing the number of layers in a CNN can lead to higher training errors on small or simple datasets due to overparameterization and the effects of built-in regularization.

The deeper ResNet-18, designed for large and complex datasets, was unable to leverage its full potential due to the dataset's limited size and simplicity. The insufficient data prevented the model from effectively generalizing and achieving lower training errors. In contrast, the 2-layer CNN, with its simpler architecture, effectively fit the dataset, demonstrating that smaller networks are often better suited for simpler tasks or smaller datasets.

This experiment highlights the importance of tailoring model complexity to the dataset size and task at hand. While deeper architectures like ResNet-18 excel in large-scale scenarios, simpler models may perform better when data is limited, offering faster training and lower error rates. Future work could extend this analysis to larger datasets or explore data augmentation techniques to evaluate the generalization capability of deeper networks under different conditions.

Thinking more on the Deep Learning side, CNN Autoencoders could significantly improve performance in this case by addressing challenges related to small or simple datasets. By learning compressed, meaningful feature representations through unsupervised training, Autoencoders help extract key patterns from the data without requiring labeled information. This pretraining can benefit both shallow and deep networks, as it provides a more efficient starting point for downstream tasks. For the ResNet-18 model, using Autoencoders for feature extraction would reduce the risk of overfitting by offering more informative inputs, potentially improving generalization on the small dataset. Similarly, for the 2-layer CNN, dimensionality reduction or denoising via Autoencoders could make the data easier to process, leading to faster and more effective training. Additionally, Autoencoders can augment the dataset by generating synthetic samples, further enhancing the model's ability to learn from limited data.

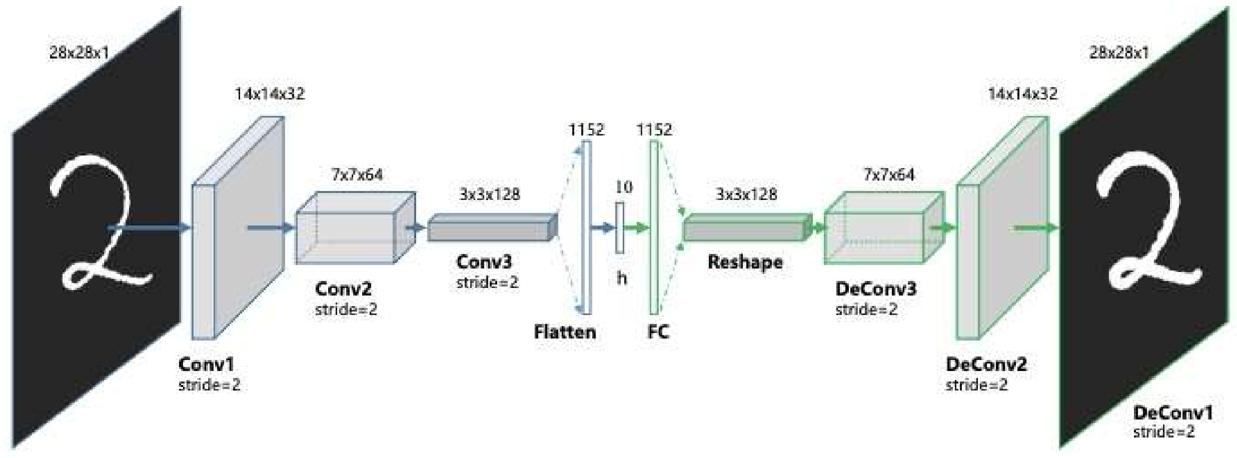

Figure 7: CNN Autoencoder