Fuzzy Pooling in Convolutional Neural Networks

Aryan Lath
Data Science and Artificial Intelligence
Indian Institute of Technology Guwahati
Assam, India
l.aryan@iitg.ac.in

***Abstract*—In recent years, convolution neural networks (CNNs) have seen remarkable success in image processing and computer vision tasks. Pooling is a vital operation in CNNs that reduces the spatial dimensions of feature maps while retaining important information. Traditional max pooling, however, discards much of the feature map data, which may lead to information loss. Fuzzy pooling applies fuzzy logic principles to preserve more nuanced information from the feature maps, resulting in enhanced learning capacity. This paper discusses the methodology, applications, and advantages of fuzzy pooling over traditional pooling in CNN architectures.**

***Index Terms*—fuzzy logic, pooling, convolutional neural networks, image processing, deep learning**

I. Introduction

Pooling layers are a critical component in convolutional neural networks (CNNs), serving the purpose of reducing the spatial dimensions of feature maps and thus, controlling overfitting by decreasing the number of parameters. Traditional pooling methods, such as max pooling and average pooling, operate by selecting the highest value in each pooling window or computing the average value, respectively. However, these methods may lead to a loss of important information within the feature map, which could limit the model's learning potential.

Fuzzy logic, with its capability to handle uncertainty and partial truths, offers a promising alternative. Fuzzy pooling, as introduced by Diamantis and Iakovidis in medical image analysis [1], integrates fuzzy logic principles into the pooling operation, enabling CNNs to retain more nuanced information. Another study by Sharma et al. [2] demonstrated the potential of fuzzy-based pooling in general image classification tasks, further validating the approach's robustness and accuracy improvements.

Fuzzy pooling uses fuzzy sets to classify pixel values within a pooling window into categories such as 'small', 'medium', and 'large', and then combines them based on these membership values. This approach enables the network to make more informed decisions during pooling, preserving details that are often discarded by traditional pooling techniques.

II. Fuzzy Pooling Methodology

Fuzzy pooling leverages fuzzy set theory to create a more nuanced pooling operation within convolutional neural networks (CNNs). Unlike max pooling or average pooling, which only consider one aggregated value (either the maximum or the mean) within a pooling window, fuzzy pooling applies fuzzy logic principles to retain and utilize more information from the feature maps. This approach increases the representational capacity of CNNs and helps to capture subtle variations that could otherwise be lost. The fuzzy pooling process involves four key steps: fuzzification, fuzzy aggregation, selection of the dominant fuzzy set, and defuzzification.

A. Fuzzification

The first step in fuzzy pooling is to assign each value in the pooling window a degree of membership in linguistic categories such as 'small', 'medium', and 'large'. This is achieved by defining fuzzy membership functions for each category. A common approach is to use triangular membership functions, which are simple yet effective in mapping values to degrees of membership.

Each value in the pooling window, denoted by $p_{i,j}$, is mapped to a membership value using these functions. For example, let's define three membership functions for the categories 'small', 'medium', and 'large'. The membership function for the 'small' category might be:

$$\mu_{\text{small}}(p_{i,j}) = \begin{cases} 1, & p_{i,j} \le c \\ \frac{d-p_{i,j}}{d-c}, & c < p_{i,j} \le d \\ 0, & p_{i,j} > d \end{cases}$$

Similarly, the membership functions for 'medium' and 'large' values can be defined as follows:

$$\mu_{\text{medium}}(p_{i,j}) = \begin{cases} 0, & p_{i,j} \le a \\ \frac{p_{i,j}-a}{m-a}, & a < p_{i,j} \le m \\ \frac{b-p_{i,j}}{b-m}, & m < p_{i,j} \le b \\ 0, & p_{i,j} > b \end{cases}$$

$$\mu_{\text{large}}(p_{i,j}) = \begin{cases} 0, & p_{i,j} \le r \\ \frac{p_{i,j}-r}{q-r}, & r < p_{i,j} \le q \\ 1, & p_{i,j} > q \end{cases}$$

Here, c, d, a, m, b, r, and q are threshold values that define the ranges for each membership function, based on the distribution of values in the feature maps. Each $p_{i,j}$ in the pooling window thus obtains a degree of membership in each fuzzy set ('small', 'medium', and 'large'), with values ranging between 0 and 1.

B. Fuzzy Aggregation

Once the membership values for each pixel in the pooling window are determined, fuzzy aggregation is performed to quantify the overall membership score for each category across the pooling window. This step involves summing the membership values for each category ('small', 'medium', and 'large') within the window.

For a pooling window with dimensions $k \times k$, the aggregated scores s_{small}, s_{medium}, and s_{large} for each category are calculated as follows:

$$s_{\text{small}} = \sum_{i=1}^{k} \sum_{j=1}^{k} \mu_{\text{small}}(p_{i,j})$$

$$s_{\text{medium}} = \sum_{i=1}^{k} \sum_{j=1}^{k} \mu_{\text{medium}}(p_{i,j})$$

$$s_{\text{large}} = \sum_{i=1}^{k} \sum_{j=1}^{k} \mu_{\text{large}}(p_{i,j})$$

The aggregated scores s_{small}, s_{medium}, and s_{large} reflect the prominence of each category within the pooling window, providing a richer representation of the values than a single maximum or average.

C. Selection of Dominant Fuzzy Set

The next step is to determine the dominant fuzzy set for the pooling window. This selection is based on the aggregated scores from the previous step. The category with the highest score is chosen as the dominant fuzzy set, representing the most significant characteristic of the values within the pooling window.

For example, if s_{medium} is the highest score among s_{small}, s_{medium}, and s_{large}, the 'medium' fuzzy set is selected as the dominant fuzzy set for that pooling window. The choice of the dominant set enables the pooling operation to focus on the most representative range of values within the window, preserving meaningful variations that traditional max or average pooling would discard.

D. Defuzzification

Once the dominant fuzzy set is identified, the defuzzification process is used to obtain a single representative value for the pooling window. Defuzzification translates the fuzzy representation back into a crisp value, which serves as the output of the fuzzy pooling operation for that window.

A common defuzzification method is the Center of Gravity (CoG) approach, which calculates a weighted average of the values in the pooling window based on their membership values in the selected fuzzy set. The pooled value p' is computed as:

$$p' = \frac{\sum_{i=1}^{k} \sum_{j=1}^{k} \mu'_{i,j} \cdot p_{i,j}}{\sum_{i=1}^{k} \sum_{j=1}^{k} \mu'_{i,j}}$$

where $\mu'_{i,j}$ represents the membership values of $p_{i,j}$ in the dominant fuzzy set. This approach retains more information from the original values, as it considers the overall distribution within the pooling window instead of reducing it to a single extreme or average value.

The result of this fuzzy pooling operation is a more informative and representative feature map, preserving subtle variations that may be important for downstream tasks such as classification or object detection. By utilizing fuzzy logic, CNNs with fuzzy pooling can potentially achieve better performance on tasks requiring fine-grained details, as they retain a broader spectrum of information compared to traditional pooling methods.

III. Results

To evaluate the performance of fuzzy pooling in convolutional neural networks, we compared it with traditional max pooling using training, validation, and test accuracy and loss metrics.

A. Dataset

The experiments in this study were conducted using the MNIST dataset, a widely used benchmark for evaluating image classification models. The MNIST dataset consists of 70,000 grayscale images of handwritten digits (0 through 9), with each image being 28x28 pixels. The dataset is split into 60,000 training images and 10,000 test images, making it suitable for assessing the performance of convolutional neural networks (CNNs).

Each image in the dataset represents a single digit centered within the frame, with pixels scaled between 0 (black) and 255 (white). The MNIST dataset is balanced, containing roughly an equal number of samples for each of the ten digit classes, ensuring no class imbalance that could affect model evaluation.

B. Model Overview

The convolutional neural network (CNN) architecture used in this study is a relatively simple yet effective model for the MNIST dataset, designed to highlight the differences between max pooling and fuzzy pooling. The architecture consists of several convolutional layers followed by pooling layers, with a final fully connected layer for classification. The simplicity of this architecture ensures that any performance differences can be attributed to the pooling methods, rather than complexities in the network design.

1) Architecture Details: The model architecture is structured as follows:

- **Input Layer**: The input to the model is a 28x28 grayscale image, corresponding to the size of each MNIST image.
- **Convolutional Layers**: The model has two convolutional layers:
  - **Conv Layer 1**: A convolutional layer with 16 filters of size 3x3 and ReLU activation. This layer captures low-level features such as edges and textures.
  - **Conv Layer 2**: A convolutional layer with 32 filters of size 3x3 and ReLU activation. With more filters,

this layer learns more complex patterns, which are crucial for distinguishing digits.

- **Pooling Layers**: The architecture includes two pooling layers, which are either max pooling or fuzzy pooling, depending on the experiment:
  - **Pooling Layer 1**: A 2x2 pooling layer applied after the first convolutional layer. This layer reduces the spatial dimensions, retaining essential information while lowering the number of parameters.
  - **Pooling Layer 2**: Another 2x2 pooling layer applied after the second convolutional layer. This further reduces the dimensions and allows the network to focus on more abstract features in the feature maps.
- **Fully Connected Layer**: After flattening the output of the final pooling layer, a fully connected layer with 32 units and ReLU activation is used to learn a compact representation of the input features.
- **Output Layer**: The final layer is a fully connected layer with 10 units and softmax activation, producing a probability distribution over the 10 possible digit classes.

2) Rationale for Model Architecture: The architecture was chosen to balance simplicity and representational power, making it well-suited for the MNIST dataset while being efficient enough to compare pooling methods. The primary considerations for this architecture are as follows:

- **Feature Extraction and Dimensionality Reduction**: The convolutional layers progressively extract features from the input images, starting with low-level details in the first layer and building up to more complex features in the second. Pooling layers follow each convolutional layer to reduce spatial dimensions, lowering computational costs and controlling overfitting.
- **Comparative Focus on Pooling**: The model's simplicity ensures that the primary difference between experimental setups is the type of pooling layer used. By limiting the number of layers and filters, any performance difference observed can be largely attributed to the pooling method rather than architectural complexity.
- **Efficiency for Fuzzy Pooling**: Fuzzy pooling, while beneficial for feature retention, involves additional computations compared to max pooling. This architecture is compact enough to accommodate fuzzy pooling without excessive computational overhead, making it feasible to explore its advantages within a manageable training time.

This model architecture allows for an effective comparison between max pooling and fuzzy pooling, providing insights into how feature retention and pooling strategies impact CNN performance on digit classification tasks. The design highlights the role of pooling in dimensionality reduction while enabling the CNN to learn relevant patterns in the MNIST dataset.

C. Accuracy Comparison

Table I provides a summary of the training, validation, and test accuracy achieved by the models with max pooling and fuzzy pooling. As shown, the fuzzy pooling model achieves a slightly higher accuracy across all metrics, indicating improved feature retention and robustness in the fuzzy pooling model.

TABLE I
TRAINING, VALIDATION, AND TEST ACCURACY COMPARISON

Pooling Method	Training Acc	Validation Acc	Test Acc
Max Pooling	98.6%	98.2%	98.27%
Fuzzy Pooling	98.7%	98.6%	98.60%

D. Loss Comparison

The training, validation, and test loss for each model are presented in Table II. The results indicate that the fuzzy pooling model has lower loss values across all metrics, suggesting better convergence and generalization. The lower test loss of fuzzy pooling demonstrates its robustness to overfitting.

TABLE II
TRAINING, VALIDATION, AND TEST LOSS COMPARISON

Pooling Method	Training Loss	Validation Loss	Test Loss
Max Pooling	0.04	0.05	0.05
Fuzzy Pooling	0.03	0.04	0.04

E. Performance Analysis with Plots

To better understand the performance differences between max pooling and fuzzy pooling, we present the training and validation accuracy and loss curves for both models in Figures 1 and 2.

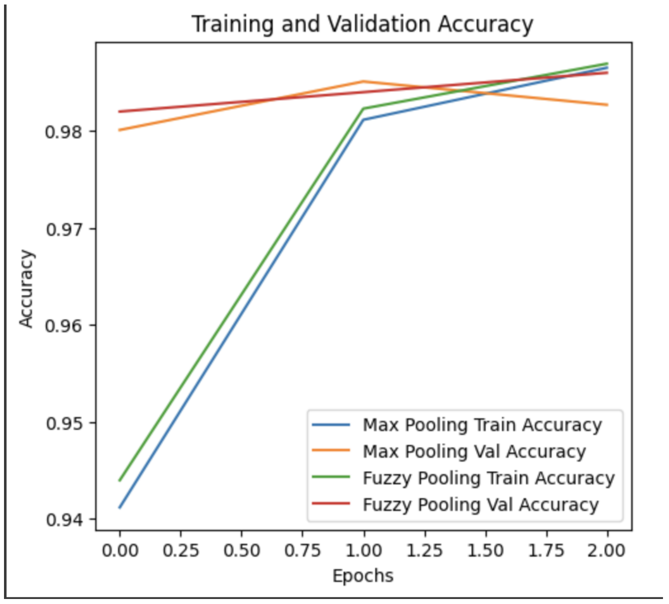

Fig. 1. Training and Validation Accuracy Comparison between Max Pooling and Fuzzy Pooling

Figures 1 and 2 illustrate the training and validation accuracy and loss curves for both models. The fuzzy pooling model demonstrates consistently higher accuracy and lower loss compared to the max pooling model. The accuracy curve for fuzzy pooling converges more smoothly and reaches a

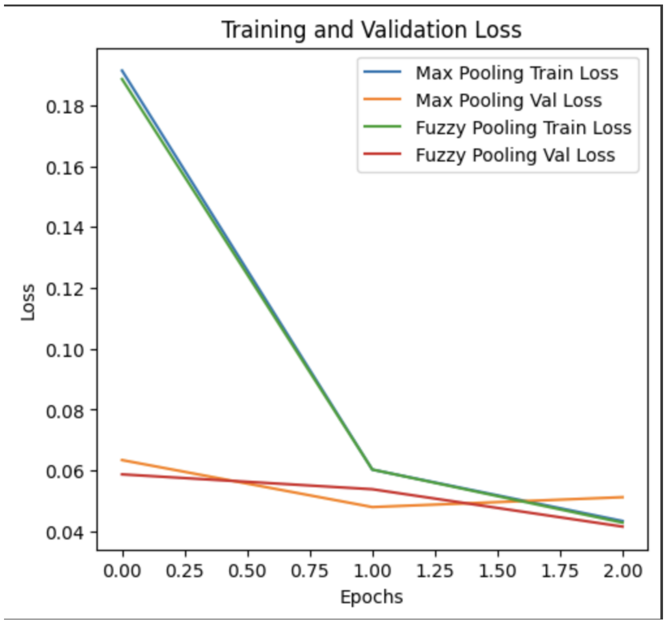

Fig. 2. Training and Validation Loss Comparison between Max Pooling and Fuzzy Pooling

slightly higher final value, suggesting that fuzzy pooling improves feature retention. Additionally, the fuzzy pooling model shows lower validation loss, which may indicate better generalization performance.

The improved performance of fuzzy pooling can be attributed to its ability to capture a broader range of feature values by incorporating fuzzy logic principles. This enables the model to retain more relevant information from the input feature maps, leading to more accurate predictions.

IV. Applications and Advantages of Fuzzy Pooling

Fuzzy pooling has several advantages over traditional pooling methods, particularly in applications that benefit from enhanced feature retention and robustness to noise.

A. Enhanced Feature Retention

Unlike max pooling, which only retains the highest value in each pooling window, fuzzy pooling uses fuzzy logic to retain more information about the overall distribution of values. This results in feature maps that better represent the input, which can improve the CNN's performance on tasks requiring fine-grained detail.

B. Robustness to Noise

Fuzzy pooling's use of fuzzy sets makes it more robust to noise in the input data. Since values are not solely dependent on the maximum intensity in the pooling window, the network can ignore outliers or noisy pixels, resulting in a cleaner feature representation.

C. Applications in Image Classification

In image classification tasks, fuzzy pooling can improve the CNN's ability to differentiate between classes by preserving more meaningful patterns within feature maps. For instance, in medical imaging or satellite image processing, retaining subtle differences in feature maps can be crucial for accurate classification.

V. Conclusion

Fuzzy pooling is a powerful alternative to traditional pooling methods in convolutional neural networks. By leveraging fuzzy logic principles, it offers enhanced feature retention and robustness to noise, which are critical in applications requiring high precision. Future work may explore the combination of fuzzy pooling with other advanced CNN techniques, as well as its potential applications in diverse fields such as medical imaging, object detection, and remote sensing.

References

[1] D. E. Diamantis and D. K. Iakovidis, "Fuzzy pooling in convolutional neural networks for medical image analysis," *IEEE Transactions on Medical Imaging*, vol. 38, no. 2, pp. 490–500, 2019.

[2] T. Sharma, V. Singh, S. Sudhakaran, and N. K. Verma, "Fuzzy based pooling in convolutional neural network for image classification," in *2021 International Conference on Artificial Intelligence and Signal Processing (AISP)*. IEEE, 2021, pp. 1–6.