قسم هندسة الحاسبات و المنظومات

كلية الهندسة - جامعة الزقازيق

2025/2024

الفرقة: الثالثة هندسة الحاسبات و المنظومات

المقرر: الإشارات و النظم

الإسم: كريم أسامة السيد عبدالرحمن

رقم السكشن: 2

الرقم في السكشن: 11

رقم الجلوس: 6347

رقم الجروب وإسم الموضوع : **Group 8 Cancer Detection**

Skin Cancer Detection CNN Model

Convolution and Its Role in Image Processing:

1. Introduction to Convolution

Convolution is a fundamental mathematical operation used extensively in various fields such as engineering, physics, statistics, and particularly in signal and image processing.

In the context of digital signal processing, convolution is a tool that enables the transformation of signals and images through filtering, smoothing, sharpening, edge detection, and more. The process of convolution involves integrating two functions to produce a third function that expresses how the shape of one is modified by the other.

2. Convolution in Image Processing

Image processing relies heavily on convolution to perform a wide variety of tasks. These include enhancing image quality, detecting features, extracting information, and transforming images for better understanding by machines or human observers.

The kernel, also called a mask or filter, is typically a small matrix that is slid across the image, and the convolution operation calculates a weighted sum of the neighborhood of each pixel.

Common operations in image processing that utilize convolution include:

- Blurring and Smoothing: Reducing noise and detail in an image.
- Sharpening: Enhancing the edges and fine details.
- Edge Detection: Identifying the boundaries within images.
- Embossing: Creating a 3D shadow effect.
- Feature Extraction: Detecting patterns or objects in images.

MaxPooling and Its Role in Image Processing:

MaxPooling (short for Maximum Pooling) is a crucial operation in image processing, particularly in convolutional neural networks (CNNs).

Its primary role is dimensionality reduction, noise suppression, and feature selection. it's a downsampling technique. It works by partitioning an input image (or feature map) into a set of non-overlapping or overlapping rectangles (usually called windows or kernels), and then outputs the maximum value from each region.

How Does MaxPooling Work?

Parameters:

- Window size (Kernel size): The area over which you apply the max operation. Common sizes are 2×2 or 3×3.
- Stride: How many pixels the window moves at a time. A stride of 2 reduces size by half.

Process:

1. Slide the window over the input matrix.
2. For each region covered, compute the maximum value.
3. Store the result in the output matrix.

Effect on Feature Maps

Let's say you have a **feature map** of size 32×32, and apply 2×2 MaxPooling with stride 2:

- Output size becomes 16×16.
- You keep only 25% of the original values.
- But most of the **essential information is preserved**.

MaxPooling vs Other Pooling Types

- **Average Pooling:** Takes average instead of max. Less aggressive, but may retain noise.
- **Global MaxPooling:** Applies max operation to the entire feature map, reducing it to a single value per feature channel.
- **L2 Pooling:** Uses the Euclidean norm; less common.

MaxPooling tends to work better for **classification tasks**, where the presence of a feature matters more than its exact value or location.

Biological Analogy

In the human visual system, receptive fields in the brain pool the responses from multiple neurons to **summarize information** from a region of the visual field. MaxPooling simulates a similar idea by summarizing feature activations.

Role in a CNN Pipeline

1. **Input Image**
2. **Convolution Layers:** Detect patterns (edges, textures).
3. **Activation (ReLU):** Introduce non-linearity.
4. **MaxPooling:** Reduce size, retain key features.

Summary

Feature	Description
Purpose	Downsample while preserving important features
Operation	Selects the **maximum** value in each pooling window
Benefits	Reduces size, prevents overfitting, adds robustness to shifts
Common Size	2×2 window with stride 2
Used In	Convolutional Neural Networks (CNNs)
Preserves	Translational invariance and dominant activations

Model Design and Layer Functions with Implementation Code

Model Overview

This project presents a Convolutional Neural Network (CNN) model tailored for the binary classification of skin lesion images specifically differentiating between benign and malignant cases.

With the increasing prevalence of skin cancer worldwide, automated diagnosis has become a crucial area of research. CNNs are widely used in medical image analysis due to their superior ability to extract hierarchical features directly from images, reducing the need for manual feature engineering.

The architecture follows a layered, sequential design, wherein each layer gradually transforms the raw pixel data into a more abstract and meaningful representation. This transformation is achieved through a combination of two fundamental operations: **convolution** and **max pooling**.

Convolution captures features, while max pooling condenses and highlights the most relevant data. The output from these operations is then passed to dense layers that make the final prediction.

Understanding Convolution in Image Processing

Convolution is at the core of how CNNs perceive images. Rather than analyzing an image as a whole, convolution allows the network to examine smaller, more manageable regions—similar to how humans focus on specific sections of a scene.

Each convolutional operation uses a small matrix called a **kernel** or **filter**, which slides across the image and performs element-wise multiplication with the portion of the image it covers. The result of this operation highlights specific features in that region. Depending on the learned values of the kernel, this can emphasize edges, corners, textures, or even specific shapes.

Initially, these filters might detect simple visual cues like straight lines or corners. However, as the network deepens, the filters evolve to identify more complex and abstract patterns. This hierarchical feature learning enables CNNs to distinguish intricate details in medical images, such as the irregular borders of a malignant lesion.

One of the key benefits of convolution is **spatial awareness**. By preserving the arrangement of pixels, convolution ensures the model maintains a sense of 'where' a feature occurs in the image. This is essential in medical applications, where the position of certain features can be diagnostically significant.

In essence, convolution helps the model "look" at an image in layers—extracting progressively more abstract and detailed insights at each step.

1st Layer: Convolution Block (Feature Extraction)

Job: The first convolutional layer receives a color image of shape (224, 224, 3), representing the height, width, and RGB color channels. It applies 32 filters of size 3x3 to this image. Each filter learns to recognize a specific visual feature, such as edges or gradients, through training. The ReLU activation function introduces non-linearity, which helps the model learn more complex patterns.

Code Responsible:

```
layers.Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3))
```

Understanding Max Pooling in Image Processing

Max pooling is another essential operation used in CNNs, typically placed after convolutional layers. While convolution helps discover features, max pooling helps distill these features by **reducing the spatial size** of the representation.

It works by scanning the image in small non-overlapping windows (usually 2x2) and selecting the maximum value in each window. This reduces the overall number of pixels, allowing the network to process data more efficiently. Importantly, it retains the strongest activations—those that are most relevant to identifying the feature of interest.

Max pooling also improves the network's ability to handle variations in the input image. Whether a lesion appears slightly to the left or right in an image, max pooling makes the model more robust by reducing sensitivity to small translations or distortions.

This operation essentially filters the image to keep only the most essential and confident signals, helping the model to focus better and generalize well to unseen data.

Max Pooling Block

Job: This layer performs a 2x2 downsampling operation. It scans over the output from the previous convolutional layer and extracts the most prominent

feature within each region. This reduces the image dimensions by half and helps in decreasing computational load.

Code Responsible:

layers.MaxPooling2D((2, 2))

2nd Layer: Convolution Block

Job: This layer continues the process of feature extraction by applying 64 filters of size 3x3 to the downsampled feature maps from the previous layer. These filters detect more refined and deeper spatial features—possibly curved lines, small patterns, or even specific combinations of edges.

Code Responsible:

layers.Conv2D(64, (3, 3), activation='relu')

Max Pooling Block

Job: Again, max pooling is used to shrink the feature maps while preserving their most essential components. By focusing on the dominant signals, this pooling layer ensures that only the most meaningful parts of the image are carried forward.

Code Responsible:

layers.MaxPooling2D((2, 2))

3rd Layer: Convolution Block

Job: This layer deepens the feature abstraction by using 128 filters. At this level, the model starts to capture high-level patterns—those that could indicate the malignancy of a lesion, such as complex textures or asymmetrical regions.

Code Responsible:

layers.Conv2D(128, (3, 3), activation='relu')

Max Pooling Block

Job: Performs another downsampling operation using a 2x2 window, further reducing the spatial dimensions and compacting the most relevant features from the 128-filter output.

Code Responsible:

layers.MaxPooling2D((2, 2))

Summary So Far

By the end of these convolution and pooling blocks, the model has effectively transformed a 224x224 pixel image into a compact and abstract set of features that encode meaningful patterns relevant to the classification task. Convolution layers discover patterns, and pooling layers filter and compress them.

These features can now be flattened and passed to fully connected (dense) layers for decision-making, which is typically the next phase in a CNN architecture.

Full Code:

[6]:
```
from tensorflow.keras import layers, models

# Build the CNN model
model = models.Sequential([
    # 1st Convolution Block
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(224, 224, 3)),
    layers.MaxPooling2D((2, 2)),

    # 2nd Convolution Block
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    # 3rd Convolution Block
    layers.Conv2D(128, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),
```

Output:

Layer (type)	Output Shape	Param #
conv2d_3 (Conv2D)	(None, 222, 222, 32)	896
max_pooling2d_3 (MaxPooling2D)	(None, 111, 111, 32)	0
conv2d_4 (Conv2D)	(None, 109, 109, 64)	18,496
max_pooling2d_4 (MaxPooling2D)	(None, 54, 54, 64)	0
conv2d_5 (Conv2D)	(None, 52, 52, 128)	73,856
max_pooling2d_5 (MaxPooling2D)	(None, 26, 26, 128)	0

References

Goodfellow, I., Bengio, Y., & Courville, A. (2016).
Deep Learning. **MIT Press.**
https://www.deeplearningbook.org/
📘 **A foundational textbook that provides in-depth explanations of convolution and pooling operations.**

- **Chollet, F. (2017).**
  Deep Learning with Python. Manning Publications.
  https://www.manning.com/books/deep-learning-with-python
  📗 Written by the creator of Keras; includes practical CNN examples for image classification.
- **Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012).**
  ImageNet Classification with Deep Convolutional Neural Networks.
  https://papers.nips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf
  📄 The landmark paper introducing deep CNNs (AlexNet) for large-scale image classification.

- **TensorFlow & Keras Documentation:**
  https://www.tensorflow.org/guide/keras
  📘 Official documentation explaining Conv2D, MaxPooling2D, and other relevant layers.
- **ISIC Archive: International Skin Imaging Collaboration**
  https://www.isic-archive.com/
  📂 A public repository of annotated skin lesion images used in many research papers and projects.
- **Towards Data Science – CNN Tutorials:**
  https://towardsdatascience.com/tagged/convolutional-neural-networks
  📰 A rich source of beginner to advanced CNN guides and case studies.

Textbooks & Foundational Reading

1. **Goodfellow, I., Bengio, Y., & Courville, A. (2016).**
   Deep Learning. MIT Press.
   🔗 https://www.deeplearningbook.org/
   ✅ Comprehensive theoretical background on CNNs, convolution, and pooling layers.
2. **Chollet, F. (2017).**
   Deep Learning with Python. Manning Publications.
   🔗 https://www.manning.com/books/deep-learning-with-python
   ✅ Hands-on guide using Keras with explanations and working CNN examples.
3. **Ian Goodfellow et al. (2021).**
   Deep Learning (Adaptive Computation and Machine Learning series). MIT Press.
   (Updated print version of above with more modern insights.)