Aplicacoes

Created	@May 20, 2025 9:20 PM
Tags	

O documento explora como as Redes Neurais Convolucionais (CNNs) são aplicadas em diversas tarefas de visão computacional, utilizando a arquitetura **VGG16** como exemplo principal.

VGG16: Uma Arquitetura Exemplo de CNN

- A VGG é uma arquitetura de CNN que ainda é utilizada hoje. Foi proposta em 2013, com um primeiro protótipo em 2014.
- Essa arquitetura teve grande sucesso no desafio **ILSVRC (ImageNet Large Scale Visual Recognition Challenge)**, vencendo em 1º e 2º lugar em diferentes categorias em 2014. A VGG16, uma implementação comum equivalente à variante "D" do artigo original, possui 16 camadas com parâmetros treináveis (13 convolucionais e 3 densas).
- O problema que a VGG (e outras CNNs de classificação) busca resolver no ImageNet é transformar uma imagem de entrada com resolução de 224×224 pixels e 3 canais (RGB), representada como um tensor (224, 224, 3), em uma das 1000 classes pré-definidas.

A arquitetura VGG16 alcançou uma taxa de erro no ImageNet de 6.8% para Top-5 e 23.7% para Top-1. O treinamento da VGG16 envolve aproximadamente 140 milhões de parâmetros treináveis.

Componentes da VGG16 e seus Papéis:

A VGG16 utiliza a combinação de diferentes tipos de camadas:

- **Convoluções + ReLU:** Estas camadas, que na VGG16 usam kernels de tamanho 3×3 com stride 1, aumentam o número de canais. A maioria dos parâmetros treináveis, no entanto, se encontra nas camadas densas. As camadas convolucionais, atuando em conjunto, funcionam como um **extrator de features** da imagem de entrada.

- **Agrupamentos (Pooling):** Essas camadas são usadas para **subamostrar a dimensão espacial** da imagem ou dos mapas de ativação.
- **Camadas Densas (Fully Connected - FC):** Localizadas no final da rede, essas camadas realizam a **classificação** com base nas features extraídas pelas camadas convolucionais.

Aplicações Além da Classificação Pura:

As CNNs, como a VGG16, podem ser adaptadas ou estendidas para outras tarefas de visão computacional:

1. **Classificação + Localização**
    - Além de classificar um objeto, podemos também **localizar sua posição** na imagem, geralmente desenhando uma "bounding box" ao redor dele.
    - Para definir uma bounding box, são necessários 4 valores (por exemplo, coordenadas x, y do centro ou canto, e altura e largura). Esta tarefa pode ser modelada como um **problema de regressão multivariada**, onde a rede prediz esses 4 valores.
    - A saída da rede para localização pode ter 4 unidades com ativação Sigmoid. A **loss MSE (Mean Squared Error)** pode ser usada para treinar a rede a prever esses valores de regressão.
    - É possível **reaproveitar os pesos** das camadas convolucionais de uma rede já treinada para classificação (por exemplo, no ImageNet) e treinar apenas as camadas densas adicionais para a tarefa de localização. Isso é eficiente, especialmente com conjuntos de dados menores.
    - A qualidade da localização é avaliada por métricas como **Intersection over Union (IoU)** e **Dice Score**, que medem a sobreposição entre a bounding box predita e a real (ground truth). Um valor ideal para ambas é 1.
2. **Classificação e Localização Simultâneas**
    - Uma única rede pode ser projetada para realizar ambas as tarefas (classificação e localização) ao mesmo tempo.
    - As camadas convolucionais iniciais continuam servindo como um **extrator de features** compartilhado. Camadas totalmente conectadas separadas

podem ser adicionadas na ponta para cada tarefa específica (uma para classificação, outra para localização).

3. **Segmentação Semântica**
    - Esta tarefa envolve uma **classificação em nível de pixel**. Cada pixel da imagem recebe uma probabilidade de pertencer a uma determinada classe.
    - Ideias iniciais envolviam classificar o pixel central de pequenos "patches" da imagem, o que era extremamente ineficiente.
    - Uma abordagem mais eficiente é usar uma arquitetura **encoder-decoder**, onde o "encoder" (por exemplo, a VGG-16) diminui a imagem (fazendo downsampling) para extrair features de alto nível, e o "decoder" (usando, por exemplo, convolução transposta) aumenta a imagem (fazendo upsampling) de volta à resolução original para classificar cada pixel.
4. **Detecção de Objetos**
    - É similar à classificação + localização, mas lida com a possibilidade de **múltiplos objetos** de diferentes classes em uma única imagem. O desafio principal é que não se sabe quantos objetos estarão presentes antecipadamente.
5. **Segmentação de Instâncias**
    - É similar à detecção de objetos, mas em vez de bounding boxes, a saída é um **valor por pixel**. A diferença em relação à segmentação semântica é que a segmentação de instâncias **diferencia objetos individuais** da mesma classe (por exemplo, se houver dois cachorros na imagem, a segmentação de instância irá separá-los como duas instâncias distintas).

O documento menciona que a discussão sobre detecção de objetos, segmentação semântica e segmentação de instâncias será aprofundada após a introdução das camadas residuais.

As referências listadas no documento incluem o Capítulo 10 de "Understanding Deep Learning", o artigo original da VGG, o artigo OverFeat (sobre reconhecimento, localização e detecção integrados), e um artigo sobre redes de deconvolução para segmentação semântica.