Ain Shams University
Computer and Systems Engineering Department
CSE477s : Fundamentals of Deep Learning

Course Project: Paper Overview
RANet: Ranking Attention Network for Fast Video Object Segmentation

Yassa Seifen Ayed	2001307
Karim Walid Fawzy	2002080

Under Supervision of :
Prof. Hazem Abbas
Prof. Mahout Khalid
Eng. Ahmed Elgazwy

Table of Contents

[I] GitHub Link

You can click link below to refer to Github ...
https://github.com/yassa9/RANet-paper-overview

[II] Introduction

Video Object Segmentation (VOS) is important for editing videos and other applications like augmented reality. Traditional methods are accurate but slow, and faster methods often make mistakes, especially when objects change appearance. The paper introduces RANet, a new system that processes videos fast without losing the quality of segmentation. It uses a special method called “ranking attention” to improve how it tracks objects over time. This makes RANet quick and accurate, suitable for real-time applications.

This report will go over how RANet works and why it’s effective, showing that it’s a significant step forward for video processing technology.

[III] What is the problem statement of the paper ?

The paper “RANet: Ranking Attention Network for Fast Video Object Segmentation” discusses how to make video object segmentation (VOS) both faster and more accurate.

Main issues the paper is addressing:

- Online Learning Costs
- Problems with Current Methods
- Need for Speed and Accuracy

1. **Online Learning Costs:** The paper tells that online learning methods (where the model learns as it processes the video) have made VOS more accurate, they are also slow and use a lot of computing power. This is because the model needs to be continuously updated for each frame of the video.

2. **Problems with Current Methods:** Other faster methods that don't use online learning, like matching-based and propagation-based methods, often aren't as accurate. They can lose track of the object when its appearance changes quickly or moves too fast.
3. **Need for Speed and Accuracy:** There's a strong need for a VOS method that works fast without losing accuracy. The challenge is to create a method that can handle both demands better than the current ones.

The goal of the paper is to introduce a new network design called RANet. This network combines the best features of both fast and accurate methods. It uses a special technique called a ranking attention mechanism to better identify and follow objects in videos, aiming to work quickly and accurately. This could help make VOS more practical for real-world applications.

[IV] What are the objectives of the paper and do you think the authors managed to achieve these goals ?

The paper "RANet: Ranking Attention Network for Fast Video Object Segmentation" aims to improve the speed and accuracy of video object segmentation (VOS). This is important for applications like video editing and analysis where quick and precise segmentation is crucial.

The objectives of the paper:

- Develop a new network architecture called RANet that integrates benefits of existing VOS methods.
- Propose a novel component, the Ranking Attention Module, to enhance segmentation by effectively using similarity maps.
- Achieve real-time VOS performance without sacrificing accuracy.

The paper claims success in these areas, showing that RANet performs well on standard datasets, delivering a good balance of speed and accuracy. The experiments suggest the authors likely achieved their goals, offering a solution that outperforms many existing methods in terms of both speed and segmentation accuracy.

[V] What is the DL method used in this paper ?

1. **Encoder-Decoder Framework:** This structure is used to learn pixel-level similarities and segmentation from video frames in an end-to-end manner, which means it processes the video frames from input to output without needing to break down the task into separate parts.
2. **Siamese Networks:** These are used within the encoder to extract detailed features from each video frame, ensuring that the network recognizes what's important in each frame, especially in relation to the object that needs to be segmented.
3. **Ranking Attention Module (RAM):** This is the key innovation of their approach. The RAM sorts and selects the most relevant features from the similarity maps generated by the encoder. This process helps in focusing the model's attention on significant features for more accurate segmentation.
4. **Integration of Techniques:** The RANet combines matching-based and propagation-based methods. It uses the learned similarities to guide the segmentation process effectively.

[VI] What are the other state-of-the-art methods that can be applied to the same problem ?

1. **Online Learning Techniques:** These methods adjust the model as new video frames come in, which helps in maintaining high accuracy but at the cost of speed. Examples include OSVOS, OnAVOS, and OSVOS-S.
2. **Matching-Based Methods:** These methods work by comparing new video frames with the initial frame and trying to find similar features to guide segmentation. They are faster than online learning techniques but may not always be as accurate because they can lose track of the object if its appearance changes too much. Examples include Pixel-Wise Metric Learning and VideoMatch.
3. **Propagation-Based Methods:** These methods use the information from previous frames to predict the segmentation in the current frame.

They can be quick and work well if the object's motion is consistent, but they struggle with rapid movements or when the object gets blocked by something else. Examples include MaskTrack and RGMP.

[VII] Would you apply any of the other methods other than the DL method used in this paper? Explain your answer ?

1. **If speed is crucial:** If the main requirement is to process videos as quickly as possible, perhaps for real-time applications like live sports analysis, using simpler **`propagation-based`** or **`matching-based methods`** might be useful. These methods are generally faster than deep learning approaches that use online learning because they require less computation.
2. **If accuracy is more important:** If the highest possible accuracy is necessary, for example in medical video analysis or in scenarios where precise object segmentation can determine the success of the application, then going with deep learning approaches like **`RANet`** would be advisable.
3. **If computational resources are limited:** In cases where computational resources are a limiting factor, such as on mobile devices or in embedded systems, simpler matching or **`propagation methods`** might be preferred over a complex deep learning model.

[VIII] What datasets have been used in this paper? Do you think the result is generalizable for any datasets ?

1. **DAVIS 2016:** DAVIS stands for “Densely Annotated VIdeo Segmentation.” It is a benchmark dataset widely used in the computer vision community, particularly for evaluating the performance of video object segmentation algorithms and focuses on the quality of the segmentation. It includes various videos that present challenging scenarios for segmentation such as occlusions, motion-blur, and appearance changes.
2. **DAVIS 2017:** An extension of DAVIS 2016, this dataset includes multiple objects per video, increasing the complexity and testing the ability of the segmentation models to handle scenarios with more than one object of interest.
3. **YouTube-VOS:** This is a large-scale dataset featuring a diverse set of videos sourced from YouTube. It includes a wide variety of objects and scenarios, which helps in testing the robustness and generalizability of the segmentation models.

Generalizability of the Results:

The use of these diverse datasets supports the generalizability of the results to some extent. However, while these datasets are comprehensive, they may still not cover all possible real-world scenarios.

1. **Domain Specific Challenges:** If the model were to be applied in a highly specialized area (like underwater imagery or medical video analysis), the results might not generalize well.
2. **Extreme Conditions:** The datasets used mainly cover common challenges in video object segmentation. In cases of extreme environmental conditions or low-quality video inputs, additional preprocessing image processing techniques may be necessary.

[IX] Discuss the results presented in the paper. Compare the results with other stateof-the art methods used to solve this problem.

First, we want to discuss important metric used in this paper and that is (J&F), as in the context of video object segmentation, J&F is used to evaluate the performance of segmentation methods.

J(Jaccard Index): This measures the overlap between the predicted segmentation and the ground truth segmentation. It's also known as the Intersection-over-Union (IoU). This metric evaluates how much the segmented object area predicted by the model matches the actual object area in the video frame.

F(Boundary Accuracy): This evaluates how accurately the boundaries of the predicted segmentation align with the boundaries of the actual object. It's a measure of the preciseness of the object's outline as predicted by the segmentation method.

Results from the Paper:

- The RANet achieves a high accuracy **(J&F=85.5%)** and runs very fast (33 milliseconds per frame) on the DAVIS16 dataset.
- With additional online learning, the performance slightly improves **(J&F=87.1%)**.
- The method is efficient because it integrates matching and propagation methods, using a ranking attention module to enhance the use of similarity maps for better segmentation.

Comparison with Other Methods:

- Traditional online learning methods, while accurate, are slower due to the need for fine-tuning during segmentation (e.g., OSVOS and OnAVOS).
- Other fast methods like SiamMask are quicker but have lower accuracy compared to RANet.
- RANet offers a good balance, being nearly more accurate but much faster.

[X] What would you like to criticize about the paper? Could you suggest any improvements.

1. **Dependency on Pre-trained Models:** The method relies heavily on a pre-trained network (ResNet). While this is common, it could limit the model's adaptability to new kinds of video content that are significantly different from what the network was originally trained on.
2. **Evaluation on More Diverse Datasets:** The evaluation mainly focuses on the DAVIS datasets, which kinda makes overfitting.
3. **Resource Usage:** The paper mentions speed a lot but doesn't talk much about the computational resources required, like memory and power consumption. It would be helpful to include a detailed analysis of the trade-offs between speed, accuracy, and resource usage, especially for applications on mobile devices or other resource-constrained environments like Embedded Systems.

[XI] Have you implemented the paper using your own code? Do your results agree with the authors? What are the differences and why ?

Although it is extremely large project to implement from scratch, or even clone the github repo `PyTocrh` formal implementaion by authors that is done in almost 2 years.

We gonna implement simplified version of RANet in TensorFlow, focusing on the major modules like the Siamese encoder, correlation layer, and decoder.

The code of course lacks many complex features like handling Ranking Attention Module (RAM).

The output of the RANet model, as outlined in the TensorFlow prototype, is a segmentation mask. This mask is a binary or probabilistic image that highlights the regions of interest—typically objects—within each frame of the video.

```
import tensorflow as tf
from  tensorflow.keras.layers  import  Conv2D,  BatchNormalization,
ReLU, MaxPooling2D, UpSampling2D
from tensorflow.keras.models import Model

# Build the Siamese Encoder.
def siamese_encoder(input_shape):
    inputs = tf.keras.Input(shape=input_shape)

    # Example of a simple convolutional block
    x = Conv2D(64, (3, 3), padding='same')(inputs)
    x = BatchNormalization()(x)
    x = ReLU()(x)
    x = MaxPooling2D((2, 2))(x)

    return Model(inputs, x, name='siamese_encoder')

# Build the correlation layer to compare features from two frames.
def correlation_layer(featuresA, featuresB):
  # This is a placeholder function for correlation, as it is complex
to handle.
  # Implementing a full correlation layer in TensorFlow might require
custom operations.
    return tf.multiply(featuresA, featuresB)

# Build the decoder part of the network.
def decoder(input_shape):
    inputs = tf.keras.Input(shape=input_shape)

    # Example of a simple upscaling block
    x = Conv2D(64, (3, 3), padding='same')(inputs)
    x = UpSampling2D((2, 2))(x)
    x = ReLU()(x)

    # Final layer to predict segmentation mask
    outputs = Conv2D(1, (1, 1), activation='sigmoid')(x)

    return Model(inputs, outputs, name='decoder')

# Build the full RANet model with Siamese Encoders and a Decoder.
def RANet(input_shape):
    frame_t = tf.keras.Input(shape=input_shape, name='frame_t')
    frame_t_minus_1 = tf.keras.Input(shape=input_shape,
                                     name='frame_t_minus_1')

    # Siamese encoders
    encoder = siamese_encoder(input_shape)
    features_t = encoder(frame_t)
    features_t_minus_1 = encoder(frame_t_minus_1)

    # Correlation layer
    correlation_output = correlation_layer(features_t,
                                           features_t_minus_1)

    # Decoder
    decoder_model = decoder(correlation_output.shape[1:])
    segmentation_mask = decoder_model(correlation_output)

    return Model(inputs=[frame_t,
                         frame_t_minus_1],
                         outputs=segmentation_mask,
                         name='RANet')

# Example input shape (H, W, Channels)
model = RANet((256, 256, 3))
model.summary()
```

[XI.1] Prototype Model HYperparameters

1. <u>**Siamese Encoder**</u>
   - **Number of Filters in Conv2D:** Set to 64.
   - **Kernel Size in Conv2D:** Set to (3, 3). This is the dimension of the convolution window.
   - **MaxPooling2D Pool Size:** Set to (2, 2). This reduces the spatial dimensions (height, width) of the input volume by a factor of 2, which helps in reducing the computation processing.
2. <u>**Decoder**</u>
   - **Number of Filters in Conv2D:** Set to 64.
   - **Kernel Size in Conv2D:** Also (3, 3), consistent with the encoder.
   - **Upsampling Factor in UpSampling2D:** Set to (2, 2), which doubles the dimensions of the input feature map, essentially reversing the effect of max pooling.
   - **Activation Function in Final Conv2D:** 'sigmoid', used for binary segmentation tasks to predict the probability that each pixel belongs to the object of interest.