Attention-Based Techniques for Target Detection in Synthetic Aperture Radar Images

AttentionDETECT

Hadrien Bontemps

Table des matières

Abstract 3
Introduction 4
1. Algorithms and Techniques 5
2. Recent advances 5
3. Challenges and Limitations 7
4. Future directions 7
5. Summary 8
Problem tackled 8
1. Feature Extraction Module 9
2. Attention Module 9
3. Scene Recognition Module 10
4. Detection Module 10
5. Algorithm flow 11
Algorithm implementation 12
1. The MiniSAR dataset 12
1.1. Reading and cropping MiniSAR images 13
1.2. Labelling sub-images 13
1.3. Retrieving all the labels 15
2. Networks implementation in PyTorch 15
2.1. Feature Extraction network 16
2.2. Attention network 17
2.3. Scene Recognition network 17
2.4. Detection network 18
Results 19
Conclusion 19
References 20

Abstract

After giving an overview of the state-of-the-art in target detection in SAR images, this report focuses on a 2021 research paper written by Di Wei *et al.*, which proposes a novel approach based on semi-supervised learning and attention mechanism. The MiniSAR dataset from Sandia National Laboratories is at the heart of this study. 588 training sub-images get preliminarily labeled, and an implementation of the network is proposed in PyTorch.

Introduction

Synthetic aperture radar (SAR) technology was developed in the 1950s as a way to improve the resolution of radar images. Traditional radar systems were limited in their ability to produce high-resolution images, because they relied on the physical size of the antenna to determine the resolution of the image. SAR technology overcomes this limitation by using signal processing techniques to synthesize a large antenna aperture, effectively creating a virtual antenna with a much larger size than the physical antenna. This allows SAR systems to produce high-resolution images with improved accuracy and detail.

SAR technology was initially developed for military applications, such as for detecting and tracking targets and for mapping terrain. However, it has since been applied to a wide range of fields, including environmental monitoring, meteorology, and geology. The ability of SAR systems to produce high-resolution images and to operate in a variety of weather and lighting conditions has made them an important tool for many applications.

The field of SAR imaging has made significant progress in recent years, with advances in technology and algorithms enabling the development of new applications and capabilities. SAR systems have been miniaturized and optimized over the years and have flown on many spacecraft like Magellan, which made mapping the surface of Venus possible despite its obscuring clouds. SAR imaging is a powerful tool for remote sensing and surveillance, allowing for the detection and characterization of objects and features in the environment based on their radar reflectivity and scattering properties. SAR imaging is particularly useful for applications where optical imaging is not feasible, such as in poor visibility conditions or at night. SAR systems can work at night and through clouds with almost same resolution as same-sized optical satellites. Additionally, SAR imaging can provide unique information about the structure and composition of objects, such as their shape, size, and material properties.

SAR technology is an important tool for earth imagery and astronomy, providing high-resolution images and detailed information about the surfaces of the Earth and other celestial bodies. SAR images can be used to monitor changes in the Earth's surface over time, such as to track the growth of vegetation, the movement of glaciers, or the spread of urbanization. They can also be used to produce high-resolution images of the surfaces of planets, moons, and other celestial bodies, allowing scientists to study their topography, composition, and other characteristics. Many companies operating in the field of earth imagery have started to invest in SAR technology. In August 2020, Rocket Lab dedicated the mission *I Can't Believe It's Not Optical* for Capella Space, an information services company providing Earth observation data on demand, to launch their first publicly available satellite in the company's commercial SAR constellation.

However, the interpretation of SAR images is challenging due to the presence of noise, clutter, and other factors that can affect the accuracy and reliability of target detection algorithms. The signal-to-noise ratio in SAR images is typically low, and the presence of clutter, such as buildings, trees, and other natural and man-made objects, can make it difficult to identify and isolate targets. Additionally, the resolution of SAR images is often limited, making it challenging to distinguish small or occluded targets. Moreover, SAR images are subject to many errors known as artifacts, such as foreshortening and layover phenomena, which make mountains appear as if they were leaning towards the viewer. SAR images at different wavelengths can show different brightness and properties depending upon the type of surface being hit. Finally, SAR technology is not effective in tracking moving objects, and

can produce incorrect results in case something moves in the scene. As a result, effective algorithms and techniques are needed for target detection in SAR images.

In this state-of-the-art review, we provide an overview of the field of target detection in SAR images, including the principles of SAR imaging, the algorithms and techniques that have been developed for target detection, and the challenges and limitations of using these methods. We begin by providing a brief history of the development of SAR technology, and discuss the key applications and challenges of using SAR images for target detection. We then introduce the various algorithms and techniques that have been developed for target detection in SAR images, including feature extraction, clustering, and classification methods. We discuss the strengths and limitations of each approach, and provide examples of how they have been used in previous research. Next, we discuss recent advances in the field, including the use of attention mechanisms and machine learning techniques to improve the performance and robustness of target detection algorithms.

1. Algorithms and Techniques

The field of target detection in SAR images has been the subject of intense research over the past decades, with the development of a variety of algorithms and techniques for identifying and characterizing targets in these challenging images. Considerable progress has been made by using traditional algorithms, but they still suffer many shortcomings, including low detection accuracy, high missed or false detection rate and poor robustness **[8]**. In recent years, artificial intelligence, especially with the development of deep learning techniques, has revolutionized target detection, offering high accuracies, reduced workload and improved robustness.

For the past few years, SAR satellites have been launched all over the world, which has significantly promoted the progress of SAR image research. Based on this, many experts and scholars have constructed some SAR image data sets, including the SAR ship detection dataset (SSDD), the High-Resolution-SAR Images (HRSID), the SAR-Ship-Dataset (OpenSARShip) and the Moving and Stationary Target Acquisition and Recognition (MSTAR). The MSTAR dataset has been particularly used for training, as it includes a large number of labeled SAR images of military vehicles and environmental scenes, collected under different conditions and viewing angles **[8] [7]**.

In order to get the most out of target detection algorithms, data preprocessing is an essential very first step. The images that are gathered in the aforementioned datasets need to be denoised and enhanced. Speckle noise, which is the main source of noise in SAR images, can be suppressed via spatial filtering (smoothing, sharpening…), transform domain (Fourier, wavelets…) and deep learning image-denoising algorithms (CNNs, GANs…).

At present, CNN-based SAR image target detection primarily concentrate on improving detection accuracy and reducing missed and false alarm rate in complex scenes, performing few shots learning to behave well on small datasets, reducing the networks' sizes, detecting small target and combining CNNs with traditional methods. Several types of methods have been developed. CNN Patch-based methods divide the image into smaller patches and use a CNN to classify each patch as containing a target or not. Region proposal-based methods use a CNN to generate a set of candidate regions that may contain a target and another CNN to classify these regions as containing a target or not. Classification methods use a CNN to classify the entire image as containing a target or not. Detection by regression methods use a CNN to predict the location and size of targets in the image.

2. Recent advances

In recent years, the field of target detection in SAR images has seen significant advances, with the development of new algorithms and techniques that have improved the accuracy, robustness, and

efficiency of target detection methods. One of the key developments in this field has been the use of attention mechanisms, also known as attention models or mechanisms, which allow for the selective focus on specific regions or features in the image, allowing for improved performance and robustness of target detection algorithms. Attention mechanisms have been used in a variety of tasks, including image recognition, natural language processing, and computer vision, and have been shown to be effective at improving the performance of deep learning models **[4]**.

The attention mechanism, like other methods based on neural networks, tries to mimic the human vision behavior when processing data by focusing on specific parts and minimizing the weight given to their surrounding. Attention models were first developed in 2014 in the field of natural language processing, and have since been widely used for many more applications. In 2017, a group of researchers at Google DeepMind published a now-famous paper "Attention is All You Need" in which they introduced the attention-based concept of Transformer which has been another revolution in the field of NLP **[1]**.

Attention mechanisms can be integrated into a variety of target detection algorithms, including feature extraction, clustering, and classification methods. For example, attention mechanisms can be used to weight different features in the image, allowing the algorithm to focus on the most relevant and informative features for target detection. This can improve the accuracy and robustness of the algorithm, particularly in the presence of noise and clutter. Additionally, attention mechanisms can be used to adaptively adjust the focus of the algorithm over time, allowing it to track moving targets or adapt to changing conditions in the image.

Recent research has also focused on developing methods for multi-target detection and tracking in SAR images, allowing for the analysis of complex scenarios with multiple targets. These methods can be used to identify and track multiple targets in the image, providing information about the location, orientation, and motion of the targets. These methods can be particularly useful for applications such as surveillance and situational awareness, where the ability to detect and track multiple targets is critical **[2] [3] [4]**.

Novel works have also started to develop methods for adaptive and self-learning algorithms, which can adapt to changing conditions and improve their performance over time. These methods can be used to improve the accuracy and robustness of target detection algorithms, allowing them to adapt to changing conditions in the image, such as the presence of noise and clutter, and to improve their performance over time. These methods can be particularly useful for applications where the environment and target characteristics are dynamic and may change over time, such as in surveillance and military operations **[6] [7]**.

The potential impact of recent advances in target detection in SAR images on applications such as remote sensing and surveillance is significant. Improved algorithms and techniques for target detection in SAR images can provide more accurate and reliable information about the location and characteristics of targets, which can be critical for a variety of applications.

For example, in remote sensing applications, improved target detection in SAR images can provide more accurate and detailed information about the environment and its features, such as buildings, roads, and other infrastructure. This information can be used for a variety of purposes, including disaster response, environmental monitoring, and mapping. Improved target detection can also provide information about the location and characteristics of targets, such as vehicles, ships, or other objects, which can be used for transportation and logistics, as well as for security and defense applications **[4] [6]**.

In surveillance applications, improved target detection in SAR images can provide more accurate and timely information about the location and movements of targets, allowing for more effective monitoring and tracking. This information can be used for a variety of purposes, including surveillance and situational awareness, as well as for security and defense applications. Additionally, the ability to detect and track multiple targets in complex environments can provide valuable information about the movements and interactions of targets, which can be critical for decision making and response in dynamic situations.

3. Challenges and Limitations

One of the key challenges and limitations of using attention mechanisms for target detection in SAR images is the selection of attention regions. Attention mechanisms typically require the specification of a set of regions or features in the image that the algorithm should focus on, and the choice of these regions can significantly affect the performance of the algorithm. In some cases, the correct attention regions may not be known *a priori*, and may need to be learned or adapted over time. Additionally, the computation complexity of attention mechanisms can be a significant challenge, particularly for large images or datasets, and may require significant computational resources to produce accurate results.

Another challenge of using attention mechanisms for target detection in SAR images is the need for large amounts of labeled data for training. Attention mechanisms typically require a significant amount of labeled data in order to learn the correct attention regions and improve the performance of the algorithm. This can be difficult and time-consuming to obtain, particularly for complex or dynamic environments. Furthermore, the quality and representativeness of the training data can significantly affect the performance of the algorithm, and may require careful consideration and analysis in order to produce accurate and reliable results.

4. Future directions

One of the current and future directions for research in the field of target detection in SAR images is the development of methods for multi-target detection and tracking. The ability to detect and track multiple targets in complex environments is critical for many applications, such as surveillance and situational awareness, and requires the development of new algorithms and techniques that can accurately identify and characterize multiple targets in the image.

Recent research has focused on developing methods for multi-target detection and tracking, including the use of multiple instance learning (MIL) and multiple hypothesis tracking (MHT) algorithms. MIL algorithms can be used to identify and classify multiple instances of the same target class, allowing for the detection of multiple targets in the image. MHT algorithms can be used to track multiple targets over time, providing information about the location, orientation, and motion of the targets. These methods can be particularly useful for applications where the environment and target characteristics are dynamic and may change over time, such as in surveillance and military operations **[4] [5]**.

Another direction for future research in the field of target detection in SAR images is the development of adaptive and self-learning algorithms. These algorithms can be used to improve the performance of target detection algorithms over time, allowing them to adapt to changing conditions in the image and to improve their performance. This can be particularly useful for applications where the environment and target characteristics are dynamic and may change over time, such as in surveillance and military operations.

Additionally, there is a need for research on the integration of target detection algorithms into larger systems for remote sensing and surveillance. This will require the development of methods for combining the output of multiple algorithms, as well as for integrating the output of target detection algorithms with other sensors and systems. This will enable the development of more comprehensive and effective systems for remote sensing and surveillance, with the ability to accurately detect and track targets in complex environments **[8]**.

5. Summary

The field of target detection in SAR images has seen significant progress in recent years, with the development of new algorithms and techniques that have improved the accuracy, robustness, and efficiency of target detection methods. The use of attention mechanisms has been particularly effective at improving the performance of target detection algorithms, and is a promising direction for future research in this field.

The current challenges and limitations of target detection in SAR images include issues related to image resolution, signal-to-noise ratio, and the presence of clutter and noise. To address these challenges, researchers have developed a variety of techniques for improving the accuracy and robustness of target detection algorithms, including image processing and machine learning methods.

The potential impact of recent advances in target detection in SAR images on applications such as remote sensing and surveillance is significant. Improved algorithms and techniques for target detection in SAR images can provide more accurate and reliable information about the location and characteristics of targets, which can be critical for a variety of applications.

One of the current and future directions for research in the field of target detection in SAR images is the development of methods for multi-target detection and tracking, as well as adaptive and self-learning algorithms. Additionally, there is a need for research on the integration of target detection algorithms into larger systems for remote sensing and surveillance.

Overall, the field of target detection in SAR images is an active and promising area of research, with significant potential for impact on a wide range of applications. The continued development of new algorithms and techniques, including the use of attention mechanisms and machine learning, holds great promise for improving the accuracy and robustness of target detection methods, and will continue to be a focus of research in the coming years.

Problem tackled

Since the field of target detection in SAR images is very large, I decided to focus on one specific recent research paper : Target Detection Network for SAR Images Based on Semi-Supervised Learning and Attention Mechanism **[7]**. This work was motivated by the observation its authors made that although existing methods based on convolutional neural networks (CNNs) can achieve great results, most of rely on fully-supervised learning and require a large number of SAR images to be labeled at target-level, meaning that each target must be manually identified and located beforehand, which gets very time-consuming as the number of SAR images used for training the models grows. The authors of this article also noticed that classical CNN-based methods too often fail to differentiate the targets from clutter in complex SAR images. From these two observations, they introduced a novel SAR target detection method based on semi-supervised learning and attention mechanism to both save time in labelling training samples and guide the network towards regions of interest, where targets most likely lie.

In order to achieve great performance with a small number of target-level labeled images, this method still relies on a large number of images labeled at image level. In fact, all SAR images used for training must be manually marked as containing any target or not. Because such a task is easy and quick to get done, the authors still refer to their method as semi-supervised, since the image-level labeled images are weakly labeled and can be considered unlabeled for that matter.

The CNN proposed by Di Wei *et al.* consists of four modules : a feature extraction module, an attention module, a scene recognition module, and a detection module. An input SAR image goes first into the feature extraction module which can extract its deep features, and then the attention module can guide the network to focus on the targets of interest within that image while ignoring their surroundings. After an attention map is obtained from the attention module, the features obtained from feature extraction and the attention map can pass through either the scene recognition module or the detection module depending on at which level the original input SAR image was labeled: target-level labeled training samples will pass through the detection branch, while the image-level labeled training samples will pass through the scene recognition branch.

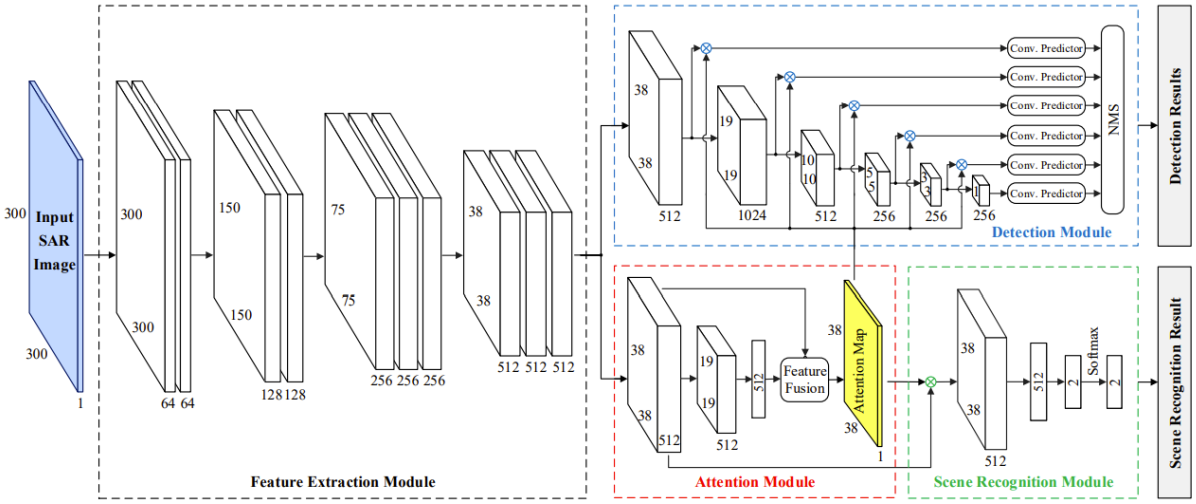

Figure 1. Global architecture of the model

1. Feature Extraction Module

The feature extraction module is the first module of the entire network, and is employed to extract the deep features of an input SAR image. The model is a modified VGGNet with four convolution stages, the first two consisting of two convolutional layers and the other consisting of three convolutional layers. A small kernel size of 3 × 3 was chosen for each convolution layer and the ReLU activation function was set to follow each one of them. Maximum pooling is used after each convolutional stage to decrease computational cost and reduce the risk of overfitting.

2. Attention Module

The attention module takes the deep features obtained through the feature extraction convolutional stages and outputs an attention map (a scalar matrix representing the relative importance of layer activations at different 2D spatial locations).

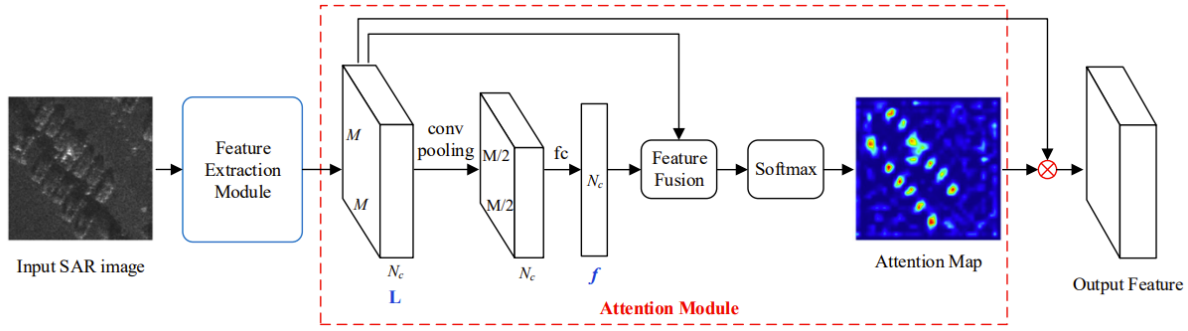

Figure 2. Focus on the attention module

After a first convolutional layer, a max pooling layer with a pixel window of 2 × 2 and a stride of 2 is used to down-sample the feature maps, and a fully connected layer is adopted to obtain the global descriptor of the input SAR image. Each local feature is then added to the global descriptor and multiplied by learnable weights to finally obtain the attention map after softmax activation. The fusion between the local features and the global descriptor is a novel approach. The idea is to force the module to consider both local and global information to output a richer and more accurate attention map.

3. Scene Recognition Module

The scene recognition module is used to classify the input SAR image at image level. It takes the deep features and attention map of the input SAR image, applies a convolution to their spatial dot product which then fed into fully connected layers, and eventually outputs whether the image contains any target or not.

4. Detection Module

The detection module is the most important part of the proposed method. It takes the deep features and attention map of the input target-level labeled SAR image and predicts the bounding boxes of all found targets. The model is inspired by Single Shot Detection (SSD), with the particularity that each feature map is first multiplied by the attention map before being fed into convolution predictors.

The Single Shot MultiBox Detector approach was introduced in 2016 by Wei Liu *et al* **[9]**. The idea is to generate a set of default boxes over different aspect ratios and scales per feature map location, attribute a score for each default box and adjust them to better match the ground truth bounding boxes in the image. The method is called “Single Shot” because it predicts the bounding boxes of the targets (and the classes to which they belong in classification problems) a single forward pass of the network, by opposition to other approaches that are based on region proposal networks.

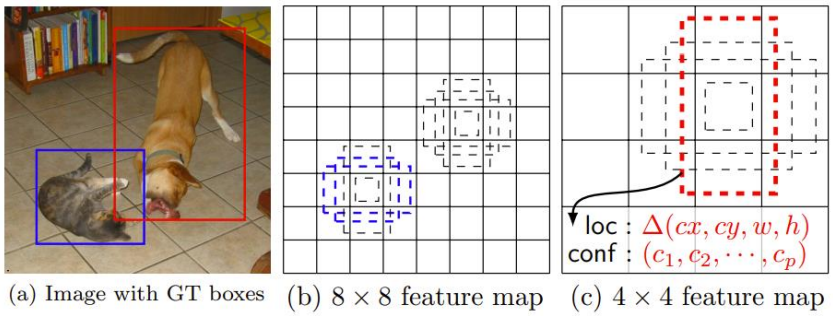

Figure 3. Illustration of the Single Shot Detection approach [9]

Finally, the results of each convolution predictor are fed into a Non-Maximum Suppression (NMS) algorithm to remove redundant targets. The reference algorithm in this paper is an efficient implementation of the NMS introduced in 2006 by A. Neubeck and L. Van Gool **[10]**. The objective of this algorithm is to select one bounding box out of all overlapping bounding boxes, in our case predicted by SSD.

5. Algorithm flow

The target detection network can be summarized with the following flowchart highlighting its two-branch organization :

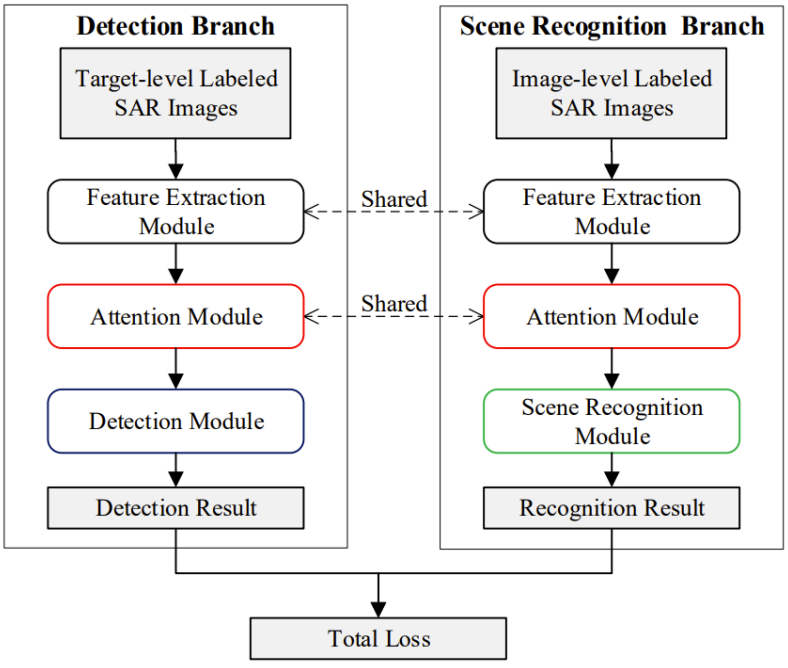

Figure 4. Semi-supervised learning flowchart

The network learns from all image-level labeled SAR images to estimate if an image contains at least one target or not, which consequently improves the detection results since the two branches share both feature extraction and attention modules.

Algorithm implementation

1. The MiniSAR dataset

In order to compare my results to the findings of Di Wei *et al.*, I worked with the same dataset. The MiniSAR dataset was acquired by U.S. Sandia National Laboratories in 2005 in the Kirtland Airforce Base. It contains 20 images of size 1638×2510 corresponding to different scenes, from a golf course to a helicopter park and a baseball field. The images were acquired by a 4-inch (0.1m x 0.1m) resolution radar in Spotlight mode in the Ku band, at 16.8 GHz center frequency with a grazing angle within the range 26-29°. **[12] [13] [14]**

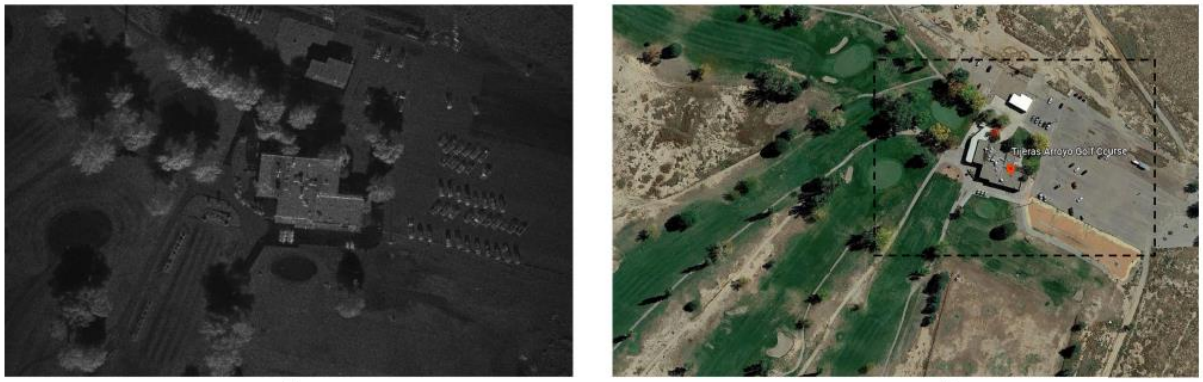

Figure 5. Sample of the MiniSAR dataset (left) / Corresponding optical scene (right)
Tijeras Arroyo Golf Course, Kirtland Air Force Base [11]

The MiniSAR images contains various types of objects, such as trees, lawns, buildings, military vehicles… Up to now, the MiniSAR is one the most complex SAR publicly available image datasets, with hundreds of targets in each image, most of them very hard to distinguish from clutter.

Figure 6. Example of a complex scene in the MiniSAR dataset

1.1. Reading and cropping MiniSAR images

The MiniSAR dataset can be downloaded on SNL official website in the Complex SAR data section **[14]**. The images are only available in Sandia's own GFF format. I struggled a lot to find a way to open those images, since I could not get the Maltab GFF reader available on the same website to work. I eventually found a working piece of code in a pdf present in the MiniSAR archive **[15]**. I first converted the 20 SAR images in grayscale JPEG format in Matlab.

As in the article, I chose 9 images out of the 20 images in the MiniSAR dataset, since some of them did not contain interesting targets and many images were redundant. 7 images where arbitrary chosen for training and 2 for testing. I cropped each of the 7 training images into 300 × 300 sub-images, for data augmentation purposes and because the network takes 300 x 300 input SAR images.

The cropping was done in Python with the PIL library, with a sliding window moved every 200 pixels, meaning two consecutive sub-images had an overlap of 100 pixels. These sub-images consist of sub-images that contain the targets and sub-images that only contain background clutter. The targets I chose to detect were the vehicles, ranging from cars, trucks, tanks, helicopters and planes. I ended up with 588 training samples.

1.2. Labelling sub-images

Labelling all the 588 training sub-images can take a while. All the images must be marked as "Target" or "Clutter" at image level, and only a small percentage of all these images must be labelled at target level. Out of the 588 training images, I found that about 183 of them contained at least one target. The proposed method uses only 30% randomly selected target-level labeled training samples, which gives us 54 images to mark at target level. The authors of the article actually labelled all the target-level training samples and compared their results over ten selections of these 30% to check whether their method was dependent on the choice of the images or not. I did the same and thus ended up labelling the 183 SAR sub-images that did contain targets.

To create all the labels, I took advantage of an online labelling service called Labelbox, free for students, which offers convenient tools to facilitate this task **[16]**. I created an ontology, with a bounding box object and a 2-option classifier ("Target" or "Clutter").

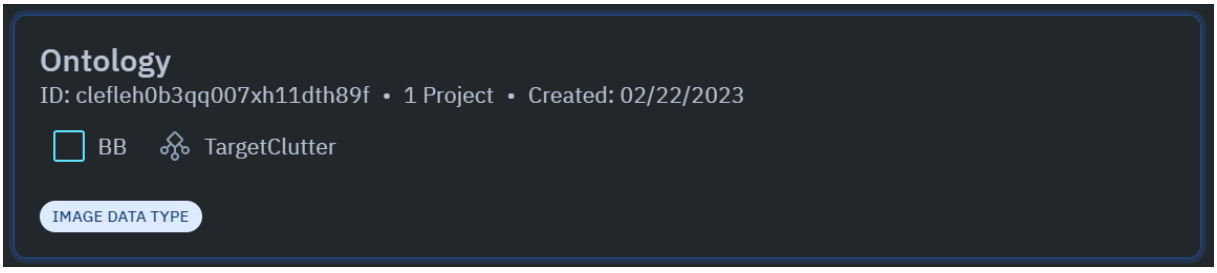

Figure 7. Ontology for SAR target detection

Then I created a queue for the 588 images to be labelled. For each image I had to mark it at image level by selecting a choice between "Target" or "Clutter" in the classifier, and also at target level if the image did belonged to the "Target" class.

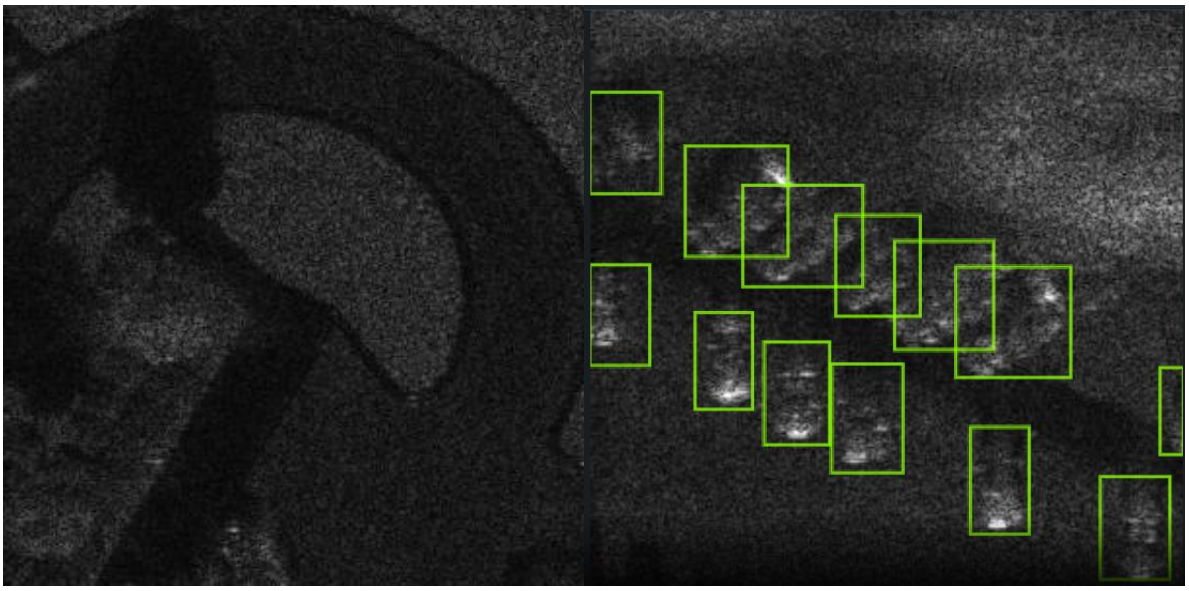

Figure 8. Example of an image labeled as 'Clutter' (left) / Example of an image labeled as 'Target' (right)

There were two problems with the bounding boxes: many targets were close and slanted, which created a lot of overlapping between neighboring boxes, and many targets were very hard to distinguish from buildings or clutter, which could cause labelling mistakes. The ideal scenario would have been to have access to satellite views of each SAR image at the same time they were acquired but I did not.

Figure 9. Glimpse of the labeled dataset on Labelbox

1.3. Retrieving all the labels

Another advantage of the Labelbox tool is that it is possible to export all the labels defined in a project thanks to their SDK. I just had to install the labelbox library and generate an API key:

```
# !pip3 install labelbox[data]
import labelbox

LB_API_KEY = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOiJjbGVma3pqcjUwengyMDd4aWF

lb = labelbox.Client(api_key=LB_API_KEY)
project = lb.get_project('clefrpsjn0hpr07y7cp3b096x')
labels = project.export_labels(download=True, start="2023-02-22", end="2023-02-22")
```

Figure 10. Labels import

The obtained labels variable is a list of dictionaries with all the information about the image and the labels for each training sample. In particular, all the bounding boxes are stored as dictionaries of the form {'top': 235, 'left': 163, 'height': 42, 'width': 56}, locating the top-left corner of the bounding box and the height and width of the rectangle in pixels (between 0 and 300 here).

2. Networks implementation in PyTorch

In the paper, the experiments were implemented using the Caffe deep learning framework, on a computer with Intel Xeon E5-2630 v4 CPU of 2.2 GHz, an NVIDIA GeForce GTX 1080 Ti GPU, and 128 GB of memory on Ubuntu 18.04 Linux system.

I personally worked with the PyTorch machine learning framework on a much more modest Intel Core i7-1065G7 CPU of 1.3GHz, an integrated Intel Iris Graphics Plus GPU, and 16GB of memory on Windows 11.

We will now see the results of the summary function from torchsummary for each module with the right input sizes.

2.1. Feature Extraction network

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 64, 300, 300]             640
              ReLU-2         [-1, 64, 300, 300]               0
            Conv2d-3         [-1, 64, 300, 300]          36,928
              ReLU-4         [-1, 64, 300, 300]               0
         MaxPool2d-5         [-1, 64, 150, 150]               0
            Conv2d-6        [-1, 128, 150, 150]          73,856
              ReLU-7        [-1, 128, 150, 150]               0
            Conv2d-8        [-1, 128, 150, 150]         147,584
              ReLU-9        [-1, 128, 150, 150]               0
        MaxPool2d-10          [-1, 128, 75, 75]               0
           Conv2d-11          [-1, 256, 75, 75]         295,168
             ReLU-12          [-1, 256, 75, 75]               0
           Conv2d-13          [-1, 256, 75, 75]         590,080
             ReLU-14          [-1, 256, 75, 75]               0
           Conv2d-15          [-1, 256, 75, 75]         590,080
             ReLU-16          [-1, 256, 75, 75]               0
        MaxPool2d-17          [-1, 256, 38, 38]               0
           Conv2d-18          [-1, 512, 38, 38]       1,180,160
             ReLU-19          [-1, 512, 38, 38]               0
           Conv2d-20          [-1, 512, 38, 38]       2,359,808
             ReLU-21          [-1, 512, 38, 38]               0
           Conv2d-22          [-1, 512, 38, 38]       2,359,808
             ReLU-23          [-1, 512, 38, 38]               0
================================================================
Total params: 7,634,112
Trainable params: 7,634,112
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 0.34
Forward/backward pass size (MB): 382.73
Params size (MB): 29.12
Estimated Total Size (MB): 412.20
----------------------------------------------------------------
```

Figure 11. Summary of the Feature Extraction network

For an input SAR image of size (1, 300, 300), with 1 referring to the number of channels in the image, the desired three convolutional stages of the Feature Extraction Module were correctly implemented, with a 1-padding on each convolutional layer and on the last Max Pooling layer.

2.2. Attention network

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 512, 38, 38]       2,359,808
              ReLU-2         [-1, 512, 38, 38]               0
         MaxPool2d-3         [-1, 512, 19, 19]               0
           Flatten-4              [-1, 184832]               0
            Linear-5                 [-1, 512]      94,634,496
              ReLU-6                 [-1, 512]               0
================================================================
Total params: 96,994,304
Trainable params: 96,994,304
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 2.82
Forward/backward pass size (MB): 14.11
Params size (MB): 370.00
Estimated Total Size (MB): 386.93
----------------------------------------------------------------
```

Figure 12. Summary of the Attention network

2.3. Scene Recognition network

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
           Flatten-1              [-1, 739328]               0
            Linear-2                 [-1, 512]     378,536,448
              ReLU-3                 [-1, 512]               0
            Linear-4                   [-1, 2]           1,026
           Softmax-5                   [-1, 2]               0
================================================================
Total params: 378,537,474
Trainable params: 378,537,474
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 4072.53
Forward/backward pass size (MB): 5.65
Params size (MB): 1444.01
Estimated Total Size (MB): 5522.19
----------------------------------------------------------------
```

Figure 13. Summary of the Scene Recognition network

Flattening the dot product between (512, 38, 38)-sized deep features and a (1, 38, 38)-sized attention map results in a column vector of 512*38*38 = 739328 rows.

2.4. Detection network

```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1         [-1, 1024, 38, 38]       4,719,616
              ReLU-2         [-1, 1024, 38, 38]               0
         MaxPool2d-3         [-1, 1024, 19, 19]               0
            Conv2d-4          [-1, 512, 19, 19]       4,719,104
              ReLU-5          [-1, 512, 19, 19]               0
         MaxPool2d-6          [-1, 512, 10, 10]               0
            Conv2d-7          [-1, 256, 10, 10]       1,179,904
              ReLU-8          [-1, 256, 10, 10]               0
         MaxPool2d-9            [-1, 256, 5, 5]               0
           Conv2d-10            [-1, 256, 5, 5]         590,080
             ReLU-11            [-1, 256, 5, 5]               0
        MaxPool2d-12            [-1, 256, 3, 3]               0
           Conv2d-13            [-1, 256, 3, 3]         590,080
             ReLU-14            [-1, 256, 3, 3]               0
        MaxPool2d-15            [-1, 256, 1, 1]               0
================================================================
Total params: 11,798,784
Trainable params: 11,798,784
Non-trainable params: 0
----------------------------------------------------------------
Input size (MB): 4072.53
Forward/backward pass size (MB): 29.19
Params size (MB): 45.01
Estimated Total Size (MB): 4146.73
----------------------------------------------------------------
```

Figure 14. Summary of the Detection network

For the detection network, I used an existing implementation of the SSD algorithm **[17]** which can be integrated as follows:

```
import torchvision
from torchvision.models.detection import ssd300_vgg16, SSD300_VGG16_Weights

ssd_model = ssd300_vgg16(weights=SSD300_VGG16_Weights.DEFAULT)
ssd_model.eval()
```

Figure 15. SSD model integration

I also fell back on different implementations of the NMS algorithm **[18]**.

```
from torchvision.ops import nms
```

Figure 16. NMS import

To avoid shape mismatches between the results of each convolution and the attention map, I down sampled the latter each time using the interpolate function from torch.nn.functional with a syntax of the form:

```
torch.matmul(L2, interpolate(A, size=L2.shape[-2:]))
```

Figure 17. Downsampling operation

Results

The whole network can be divided into the Scene Recognition and Detection branches with shared Feature Extraction and Attention modules as follows:

```
class SceneRecognitionBranch(nn.Module):
    def __init__(self):
        super(SceneRecognitionBranch, self).__init__()
        self.criterion = nn.BCELoss()

    def forward(self, image_level_labeled_image):
        L = FeatureExtraction().forward(image_level_labeled_image)
        L = L[None, :, :, :]
        A = Attention().forward(L)
        outputSR = SceneRecognition().forward(L, A)
        return outputSR


class DetectionBranch(nn.Module):
    def __init__(self):
        super(DetectionBranch, self).__init__()
        self.criterion = nn.SmoothL1Loss()

    def forward(self, target_level_labeled_image):
        L = FeatureExtraction().forward(target_level_labeled_image)
        L = L[None, :, :, :]
        A = Attention().forward(L)
        outputDT = Detection().forward(L, A)
        return outputDT
```

Figure 18. Branches outline

Then, training the network consists of browsing the whole training dataset at each epoch, calculating the output of the Scene Recognition branch each time and the output of the Detection branch if applicable, and fine tuning the model parameters via back propagation and optimization.

I could not manage to get results in time, out of lack of organization but also because training this whole network takes a very long time especially on such a weak machine as mine. This is also the conclusion of the paper that although the proposed method can significantly improve the performance of target detection in SAR images, the computational complexity is huge and the detection speed very slow. With more time I would have tried to lighten the whole network as the authors did in their ablation study.

Conclusion

The novel approach proposed by Di Wei *et al.* to detect targets in SAR images based on semi-supervised learning and attention mechanism looks promising. They out-performed all previous approaches on both precision and F1-score, leading to less missing and false alarms and higher robustness, with only 30% of the training samples labeled at target level. As mentioned in their own conclusion, the next step would be to find a way to simplify the proposed network in order to speed up the training process.

References

[1] Chen, Lifu, Ting Weng, Jin Xing, Zhouhao Pan, Zhihui Yuan, Xuemin Xing, and Peng Zhang. 2020. “A New Deep Learning Network for Automatic Bridge Detection from SAR Images Based on Balanced and Attention Mechanism”. *Remote Sensing* 12(3):441. doi: 10.3390/rs12030441.

[2] Gao, Fei, Aidong Liu, Kai Liu, Erfu Yang, and Amir Hussain. 2019. “A Novel Visual Attention Method for Target Detection from SAR Images”. *Chinese Journal of Aeronautics* 32(8):1946-58. doi: 10.1016/j.cja.2019.03.021.

[3] Gao, Fei, Wei Shi, Jun Wang, Amir Hussain, and Huiyu Zhou. 2019. “A Semi-Supervised Synthetic Aperture Radar (SAR) Image Recognition Algorithm Based on an Attention Mechanism and Bias-Variance Decomposition”. *IEEE Access* 7:108617-32. doi: 10.1109/ACCESS.2019.2933459.

[4] Ghaffarian, Saman, João Valente, Mariska van der Voort, and Bedir Tekinerdogan. 2021. “Effect of Attention Mechanism in Deep Learning-Based Remote Sensing Image Processing: A Systematic Literature Review”. *Remote Sensing* 13(15):2965. doi: 10.3390/rs13152965.

[5] Liu, Shuo, and Zongjie Cao. 2014. “SAR Image Target Detection in Complex Environments Based on Improved Visual Attention Algorithm”. *EURASIP Journal on Wireless Communications and Networking* 2014(1):54. doi: 10.1186/1687-1499-2014-54.

[6] Shi, Baodai, Qin Zhang, Dayan Wang, and Yao Li. 2021. “Synthetic Aperture Radar SAR Image Target Recognition Algorithm Based on Attention Mechanism”. *IEEE Access* 9:140512-24. doi: 10.1109/ACCESS.2021.3118034.

[7] Wei, Di, Yuang Du, Lan Du, and Lu Li. 2021. “Target Detection Network for SAR Images Based on Semi-Supervised Learning and Attention Mechanism”. *Remote Sensing* 13(14):2686. doi: 10.3390/rs13142686.

[8] Zhang, Ying, and Yisheng Hao. 2022. “A Survey of SAR Image Target Detection Based on Convolutional Neural Networks”. *Remote Sensing* 14(24):6240. doi: 10.3390/rs14246240.

[9] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg. “SSD: Single Shot MultiBox Detector”. 2016. *Springer International* Publishing(21-37). doi: 10.1007/978-3-319-46448-0_2.

[10] A. Neubeck and L. Van Gool, "Efficient Non-Maximum Suppression," *18th International Conference on Pattern Recognition* (ICPR'06), Hong Kong, China, 2006, pp. 850-855, doi: 10.1109/ICPR.2006.479.

[11] Y. Shi, L. Du and Y. Guo, "Unsupervised Domain Adaptation for SAR Target Detection," in *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 14, pp. 6372-6385, 2021, doi: 10.1109/JSTARS.2021.3089238.

[12] Zi-shuo Han, Chun-ping Wang, Qiang Fu. “Arbitrary-oriented target detection in large scene sar images,” *Defence Technology*, Volume 16, Issue 4, 2020, Pages 933-946, ISSN 2214-9147, doi: 10.1016/j.dt.2019.11.014.

[13] Liao, L.; Du, L.; Guo, Y. “Semi-Supervised SAR Target Detection Based on an Improved Faster R-CNN”. *Remote Sensing*. 2022, 14, 143. doi: 10.3390/rs14010143.

[14] Sandia National Laboratories, Complex SAR data. https://www.sandia.gov/radar/pathfinder-radar-isr-and-synthetic-aperture-radar-sar-systems/complex-data/ (Feb. 2023)

[15] SANDIA REPORT / SAND2006-xxxx, Unlimited Release, Printed April 2006. “Viewing GFF format SAR images with Matlab”, William H. Hensley, Jr. and Armin W. Doerry.

[16] Labelbox, "Labelbox," Online, 2023. [Online]. Available: https://labelbox.com

[17] Nvidia Deep Learning Examples
https://pytorch.org/hub/nvidia_deeplearningexamples_ssd/
https://github.com/NVIDIA/DeepLearningExamples/tree/master/PyTorch/Detection/SSD

[18] Non-Maximum Suppression Implementation
https://learnopencv.com/non-maximum-suppression-theory-and-implementation-in-pytorch/
https://pytorch.org/vision/main/generated/torchvision.ops.nms.html