Semantic segmentation for learning hockey broadcast image representation

Philippe Blouin-Leclerc and Stéphane Caron
Laval University
{philippe.blouin-leclerc.1, stephane.caron.9}@ulaval.ca

May 10th 2019

Abstract

In this project, we propose a novel way to recognize key locations within hockey broadcast images using semantic segmentation and convolutional neural networks (CNN). The semantic representation of an image could then be used for many applications such as mapping a broadcast image into a 2D plan. All the codes that aimed to realized that project and that article are hosted on that GitHub repository.

1 Introduction

Computer vision is a growing field that changed the faces of many applications in robotic, security or even health sectors. Another field that is currently having more and more interests in such computer vision applications is the sports analytics. The professionnal sports clubs are using analytics and data to understand as more as they can the game and the performances of their players and their opponents. To increase that understanding, you often need a bunch of experts analyzing a bunch of data. Computer vision is well suited to extract that data because it allows the detection of many events simultaneously, which otherwise may have been done by many humans. In order to extract that data properly, it's way more interesting to map those events on the field (or the ice). To do so, it's often necessary to understand the general representation of the moment (or the image), which could be done by a computer vision task called semantic segmentation.
In his work, Homayounfar et al. (2017) present a methodology where he uses different cues on the field such as lines, corners, circles and so on to train another model that position the field in a 2 dimensionnal plan. In our project, we tried to improve the cues detection technic by training different semantic segmentation models and gain insights on how to train such models. The next step following that project will be to also use that representation in order to map players into a 2 dimensionnal plan.
In the next section, we will start by giving some background about the task of semantic segmentation (section 2.2), then we'll present our methodology (section 3), the dataset we used for our experiments (section 4.2), the results we had (section 5) and finally a short conclusion (section 6).

2 Background

2.1 Define the task

Before getting into more details in the 3 section, let's first define the task we will try to learn in this project. Semantic segmentation is a computer vision task where the model learns the general representation of an image by attributing a label to each and every pixels. Intuitively, to make pixel-wise predictions, you first need to have attributed a class to each pixel in the image (see figure 1), this is called a **mask**.

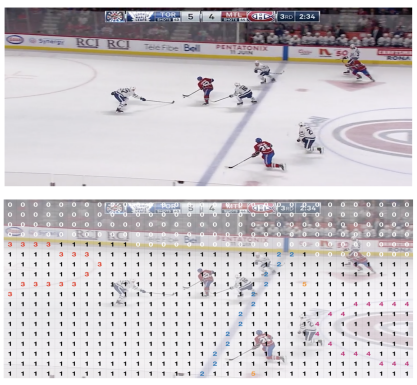

Figure 1: Example of mask where each pixel of an image is associated with one class.

2.2 Complete workflow

Once we have a class to every pixel, we now have the output we would like our model to predict. Similarly to other multiclass learning problems, we will have to *one-hot encode* the output of the model so that the output for every pixel will be a vector with the same length as the number of classes. If we look at it matrix-wise instead of pixel-wise, we can say that we will have *one-hot encoded matrix* as the output of the model. Each of those matrix. have the same width and height as the input image, will be related to a specific class and the values predicted by the model will correspond to probabilities to be part of that class.

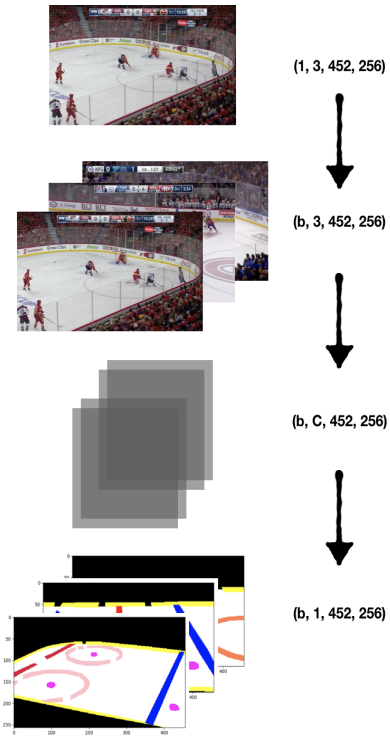

Figure 2: Summary of the workflow behind the semantic segmentation learning task.

The figure 2 summarizes this workflow where we start from multiple images and ends up with multiple predictions. As such, we start with one RGB image, then we have b images inside one minibatch. At the end of the model, each of those images are *one-hot* encoded to C dimensions, corresponding to the number of classes to predict. Finally, we apply a `softmax` or any other function that select a predicted class, so that we no longer have C dimensions, but only 1.

3 Methodology

- Semantic segmentation (architerures, loss, data aug)
- mapping (plus short)

4 Dataset

4.1 Dataset creation

We applied the methodology described earlier to hockey broadcast images. We generated our dataset, which contains images (inputs) and masks (outputs), by ourselves. To do so, we annotated a total of 43 NHL broadcast images. For the annotation part, we used an open source tool called cvat tool. That tool allowed us to draw polygons around our labels and then associate a class to each pixels in the image.
To choose our labels, we used an iterative approach where we first used our judgment to decide arbitrary categories (ex: ice, crowd, corners, horizontal lines, vertical lines). After that, we used one single image and tried to fit a model using those labels. We then noticed the bahviour of the model regarding those categories and adapted them. For example, we noticed the corners, the horizontal and vertical lines such as the board were difficult to distinguish for the model. As such, we decided to merge them and use one single category called "board". Here are the **9 categories** we selected at the end: ice, boards, crowd, red line, blue lines, goal lines, circles in end zone, circles in neutral zone and dots. The figure 3 is an example of image we extracted and then labeled using our 9 categories.

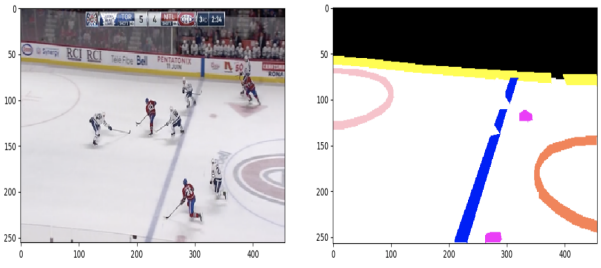

Figure 3: Example of extracted image (left) and the labeling output (right) using the 9 categories.

In the methodology section, we mentionned we had a class imbalance problem. In that section, we tackled this problem by adapting the loss. In the labeling task, we also tackled that class imbalance problem

by subjectively drawing larger polygons for rare categories such as dots or circles.

4.2 Data augmentation

The labeling task was quite time consuming so that's mainly why we were not able to extract a large amount of data (only 43 images). However, we used data augmentation technics to increase that dataset. There are a lot of different augmentation technics for images, such as rotation, flip, scale or crop. Because we did not had a large amount of data, we tried to keep our augmented images as more realistics as we can. For this reason, we used horizontal rotations only. That kind of transformation makes sense in our context because a hockey ring is symmetric, so that way, we conserve the nature of the image. The figure 4 shows an example of horizontal flip transformation.

Figure 4: Example of augmented image (bottom) by doing a horizontal flip of the original image (top).

Allo

5 Results

Training details de la pancarte + les results

6 Conclusion

In conclusion, we can first say that we need more images. The lack of images is even more problematic for a non pre-trained model such as the U-Net model. We can also say that a pre-trained model seems to be a must in our learning task, even if we could have a larger dataset. After a lot of training attempts, we also noticed that those can of models seems to learn slowly and over a long period of time.
The next steps would be to extract more images. Also, because the pre-trained model was quite successful, we may need to put more focus on using a more specific encoder for that task. In the same idea, we may need to put more efforts on the decoder part of the model, which can hardly be pre-traine in our case. Finally, if we are able to build a adequate semantic segmentation model, we could then use that model to map our images into a 2 dimensional plan.

References

Homayounfar, N., Fidler, S., & Urtasun, R. (2017). Sports field localization via deep structured models. In *2017 IEEE conference on computer vision and pattern recognition, CVPR 2017, honolulu, hi, usa, july 21-26, 2017* (pp. 4012–4020).