Region Mutual Information Loss for Semantic Segmentation

Kobee Raveendran

January 20, 2020

1 Summary

In this paper, the authors propose a novel loss function that takes advantage of dependencies and relationships between neighboring pixels in semantic segmentation tasks. Compared to many previous well-performing methods, which consider each individual pixel in isolation when calculating loss, the authors' method, called RMI (Region Mutual Information loss) considers the pixel in question as well as its 9 neighboring pixels. They then use the neighboring pixels and the central pixel to estimate mutual information (MI), which is a measure of the amount of information the two entities contain about each other. Since it is infeasible to calculate MI directly, the authors estimate a lower bound then optimize with respect to MI to approach the actual value.

2 Good points

One point I found highly desirable about RMI is that it does not require much additional computational resources and is easily transferrable to existing models. RMI requires no extra overhead during inferencing unlike some previous methods, and requires little extra compute resources during training (accomplished via downsampling), unlike memory-intensive pixel affinity methods. Possibly the best feature, however, is that RMI can be integrated into already-existing semantic segmentation frameworks without requiring tweaks to the base model architecture at all, which is perfect for making this type of loss function commonplace (as the other major loss functions all exhibit this "plug and play" trait).

3 Weak points

While the improvements on previous methods with respect to compute resources required is definitely impressive, the improvement in performance during evaluation is not so impressive. When adopting a loss function such as this, which **should** make more sense (as neighboring pixels are related), one would expect a much greater increase in performance. However, on each of the benchmarks, using RMI typically resulted in an improvement of less than 1%.

4 Questions

One doubt I had was how edges of objects in the image were handled. For pixels within an object, its neighbors have a strong correlation with their neighbors, but for boundaries between objects, they have little to no effect on each other, or even oppose each other. I'm curious to see what effect this condition has on the loss function.

5 Ideas

I'd used a similar technique in one of my past research projects (though without mutual information) for efficient image compression, and we ran into problems with pixels on the border of the image cascading their influence deeper into the central parts of the image (corner and side pixels were ignored as they'd have less than 9 neighbors). It's apparent that the authors' method does not suffer from this problem, since the neighbors only influence one pixel, and this 'influence' is calculated on a pixel-by-pixel basis, and likely isn't applied when the process is repeated for the next pixels. However, it would be interesting to see what would happen in the authors' method if the neighboring pixels had a similar diminishing cascading effect instead of keeping the influence within each individual pixel.