On the Implementation of ALFA – Agglomerative Late Fusion Algorithm for Object Detection

Iuliia Saveleva(✉) and Evgenii Razinkov

Institute of Computational Mathematics and Information Technologies, Kazan Federal University, Kazan, Russia
JuOSaveleva@stud.kpfu.ru, Evgenij.Razinkov@kpfu.ru

Abstract. The paper focuses on implementation details of ALFA – an agglomerative late fusion algorithm for object detection. ALFA agglomeratively clusters detector predictions while taking into account bounding box locations and class scores. We discuss the source code of ALFA and another late fusion algorithm – Dynamic Belief Fusion (DBF). The workflow and the hyperparameters necessary to reproduce the published results are presented. We also provide a framework for evaluation of late fusion algorithms like ALFA, DBF and Non-Maximum Suppression with arbitrary object detectors.

Keywords: Object detection · Late fusion · Agglomerative clustering

1 Introduction

Object detection is an important and challenging computer vision problem. State of the art object detectors, such as Faster R-CNN, YOLO, SSD and DeNet, rely on deep convolutional neural networks and show remarkable results in terms of accuracy and speed. Fusing results of several object detection methods is a common way to increase accuracy of object detection. In the companion paper [1] a new late fusion algorithm for object detection called ALFA was proposed. ALFA relies on agglomerative clustering and shows state of the art results on PASCAL VOC 2007 and 2012 object detection datasets.

We also implemented Dynamic Belief Fusion – state of the art late fusion algorithm for object detection proposed in [2] – as our baseline, since the implementation from authors is not available.

Here we describe our implementation of ALFA and DBF providing pseudocode for the key functions of these methods. We also provide hyperparameter values required to reproduce results from [1] on PASCAL VOC 2012 dataset. Results on PASCAL VOC 2007 are not reproducible due to randomness of a cross-validation procedure.

Link to our implementation: http://github.com/IuliiaSaveleva/ALFA. All the details required to successfully run the code are provided in README.md.

B. Kerautret et al. (Eds.): RRPR 2018, LNCS 11455, pp. 98–103, 2019.
https://doi.org/10.1007/978-3-030-23987-9_9

2 Implementation

Assume object detection task for K classes and N trained object detectors $D_1, D_2, ..., D_N$. Given an image I object detector produces a set of predictions:

$$D_i(I) = \{p_1, ..., p_{m_i}\}, \quad p = (r, c),$$

where m_i is the number of detected objects, r represents four coordinates of the axis-aligned bounding box and c is class scores tuple of size $(K + 1)$, including "no object" score $c^{(0)}$.

2.1 ALFA Implementation

The steps of ALFA are given below.

2.1.1 Agglomerative Clustering of Base Detectors Predictions

We assume that prediction bounding box r_i and class scores c_i should be similar to other prediction bounding box r_j and class scores c_j if they correspond to the same object. Let C_i and C_j be two clusters and $\sigma(p, \tilde{p})$ – similarity score function between predictions p and $\tilde{p}$. We define the following similarity score function with hyperparameter τ for prediction clusters:

$$\sigma(C_i, C_j) = \min_{p \in C_i, \tilde{p} \in C_j} \sigma(p, \tilde{p}), \quad \text{while} \quad max_{i,j}\sigma(C_i, C_j) \geq \tau. \tag{1}$$

We propose the following measure of similarity between predictions:

$$\sigma(p_i, p_j) = IoU(r_i, r_j)^{\gamma} \cdot BC(\bar{c}_i, \bar{c}_j)^{1-\gamma}, \tag{2}$$

where $\gamma \in [0, 1]$ is a hyperparameter, BC – Bhattacharyya coefficient as a measure of similarity between class scores ($\bar{c}$ is obtained from class score tuple c by omitting the zeroth *"no object"* component and renormalizing):

$$BC(\bar{c}_i, \bar{c}_j) = \sum_{k=1}^{K} \sqrt{\bar{c}_i^{(k)} \bar{c}_j^{(k)}}, \quad \bar{c}^{(k)} = \frac{c^{(k)}}{1 - c^{(0)}}, \quad k = 1, ...K, \tag{3}$$

IoU – intersection over union coefficient which is widely used as a measure of similarity between bounding boxes:

$$IoU(r_i, r_j) = \frac{r_i \cap r_j}{r_i \cup r_j}. \tag{4}$$

See Algorithm 1.

2.1.2 Class Scores Aggregation

Assume that predictions from detectors $D_{i_1}, D_{i_2}, ..., D_{i_s}$ were assigned to object proposal π. We assign an additional low-confidence class scores tuple to this object proposal for every detector that missed:

$$c_{lc} = \left(1 - \varepsilon, \frac{\varepsilon}{K}, \frac{\varepsilon}{K}, ..., \frac{\varepsilon}{K}\right), \tag{5}$$

where ε is a hyperparameter.

Each method uses one of two class scores aggregation strategies:

– *Averaging fusion*:

$$c_{\pi}^{(k)} = \frac{1}{N}\left(\sum_{d=1}^{s} c_{i_d}^{(k)} + (N - s) \cdot c_{lc}^{(k)}\right), k = 0, ..., K. \tag{6}$$

– *Multiplication fusion*:

$$c_{\pi}^{(k)} = \frac{\tilde{c}_{\pi}^{(k)}}{\sum_i \tilde{c}_{\pi}^{(i)}}, \quad \tilde{c}_{\pi}^{(k)} = \left(c_{lc}^{(k)}\right)^{N-s} \prod_{d=1}^{s} c_{i_d}^{(k)}, \quad k = 0, ..., K. \tag{7}$$

2.1.3 Bounding Box Aggregation

All methods have the same bounding box aggregation strategy:

$$r_{\pi} = \frac{1}{\sum_{i \in \pi} c_i^{(l)}} \sum_{i \in \pi} c_i^{(l)} \cdot r_i, \quad \text{where} \quad l = \underset{k \geq 1}{\operatorname{argmax}}\, c_{\pi}^{(k)}. \tag{8}$$

Best ALFA parameters are provided in Table 1:

Table 1. Best ALFA parameters.

Detectors	Methods	Confidence threshold	mAP	τ	γ	Scores aggregation strategy	ε	δ
SSD + DeNet	Fast ALFA	0.05	mAP	0.73	0.25	Averaging	0.26	True
	ALFA	0.015						
	Fast ALFA	0.05	mAP-s	0.48	0.22	Multiplication	0.56	True
	ALFA	0.015						
SSD + DeNet + Faster R-CNN	Fast ALFA	0.05	mAP	0.74	0.3	Averaging	0.39	False
	ALFA	0.015						
	Fast ALFA	0.05	mAP-s	0.75	0.28	Multiplication	0.17	True
	ALFA	0.015						

Algorithm 1. Agglomerative Clustering

Data: $D = D_1(I) \cup ... \cup D_N(I)$; Hyperparameters: γ, $\tau \in [0, 1]$, $\delta = \{False, True\}$

begin

Set $\sigma(p_i, p_j) = IoU(r_i, r_j)^{\gamma} \cdot BC(\bar{c}_i, \bar{c}_j)^{1-\gamma}$

$G = \{g_{ij}\}$, $g_{ij} = 1$ **if** $((label_i = label_j)$ or $\delta = False)$, 0 **otherwise**

$U = \{u_{ij}\}$, $u_{ij} = 1$ **if** $\nexists t : p_i, p_j \in D_t(I)$, 0 **otherwise**

$S = \{s_{ij}\}$, $s_{ij} = \sigma(p_i, p_j)$ **if** $\sigma(p_i, p_j) > \tau$ **else** 0

$Q = G \circ U \circ S$

$k = 0$; $W_0 = Sign(Q)$

do

$k = k + 1$

$W_k = Sign(W_{k-1} \cdot W_{k-1})$

while $W_k \neq W_{k-1}$;

$M = UniqueRows(W_k)$

for $i := 1$ **to** $|M|$ **do**

$Clusters_i = \{C_j = \{p_j\} | m_{ij} = 1\}$

do

$sim = \max\limits_{C, C'} \min\limits_{p \in C, p' \in C'} \sigma(p, p'), \quad C, C' \in Clusters_i$

if $sim > \tau$ **then**

$C_i = C_i \cup C_j$

$Clusters_i = Clusters_i \setminus C_j$

while $sim > \tau$;

return $\cup_i Clusters_i$

2.2 DBF Implementation

Our implementation of DBF consists of the following steps:

1. Compute PR-curves PR_i^k for each class k and each detector D_i, $i = 1, ..., N$;
2. Construct detection vectors for each $p \in D_i(I)$, $i = 1, ..., N$, and calculation of basic probabilities of hypothesis according to label l and PR_i^k. See Algorithm 2;
3. Join basic probabilities by Dempster-Shaffer combination rule:

$$m_f(A) = \frac{1}{N} \sum_{X_1 \cap X_2 ... \cap X_K = A} \prod_{i=1}^{K} m_i(X_i),$$

where $N = \sum_{X_1 \cap X_2 ... \cap X_K \neq \varnothing} \prod_{i=1}^{K} m_i(X_i)$, to determine fused basic probabilities $m_f(T)$ and $m_f(\neg T)$;
4. Get fused score as $\bar{s} = m_f(T) - m_f(\neg T)$;
5. Apply NMS to bounding boxes r and scores $\bar{s}$. In order to help DBF more on NMS step we sort detections by score $\bar{s}$ and precision from PR_i^k, k = l, if detections had equal $\bar{s}$ values.

Algorithm 2. DBF algorithm: Constructing detection vectors and calculating basic probabilities of hypothesis

Data: $p = (r, c)$, $D_i(I)$, $i = 1, ..., N$, PR^k; Hyperparameter: n
begin
  for $i := 1$ **to** N **do**
    if $p \notin D_i(I)$ **then**
      Find $\bar{p_i} = (\bar{r_i}, \bar{c_i})$ that $l = l_i$ and $max(IoU(r, r_i))$, $l = \text{argmax}_{k \geq 1}\, c^k$
      if $IoU(r, \bar{r_i}) > 0.5$ **then**
        $d_i = \bar{c}_i^{\bar{l}_i}$
      else
        $d_i = -\infty$
    else
      $d_i = c^l$
  Calculate $\{m(T), m(\neg T), m(I)\}$ for each component of detection vectors d:
  for $i := 1$ **to** N **do**
    Get precision p and recall r from PR^k, $k = \text{argmax}_{k \geq 1}\, c^k$, using score from d_i
    $m(T)_i = p$
    $p_{bpd} = 1 - r^n$ – precision of best possible detector
    $m(\neg T)_i = 1 - p_{bpd}$
    $m(I)_i = p_{bpd} - p$
return $m(T)$, $m(\neg T)$, $m(I)$

Best DBF parameters are provided in Table 2:

Table 2. Best DBF parameters.

	SSD + DeNet		SSD + DeNet + Faster R-CNN	
	mAP	mAP-s	mAP	mAP-s
n	16	16	18	14
Confidence threshold	0.015			

3 Conclusion

This paper had presented implementation details of ALFA and DBF late fusion methods for object detection. We provide source code and hyperparameter values that allow one to reproduce results from [1] on PASCAL VOC 2012.

Acknowledgment. I. Saveleva was funded by the Russian Government support of the Program of Competitive Growth of Kazan Federal University among World's Leading Academic Centers and by Russian Foundation of Basic Research, project number 16-01-00109a.

References

1. Razinkov, E., Saveleva, I., Matas, J.: ALFA: agglomerative late fusion algorithm for object detection. In: 2018 24th International Conference on Pattern Recognition (ICPR), pp. 2594–2599. IEEE, August 2018
2. Lee, H., Kwon, H., Robinson, R., Nothwang, W., Marathe, A.: Dynamic belief fusion for object detection. In: 2016 IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 1–9. IEEE (2016)