AttentionBoost: Learning What to Attend for Gland Segmentation in Histopathological Images by Boosting Fully Convolutional Networks (Supplementary Material)

Gozde Nur Gunesli, Cenk Sokmensuer, and Cigdem Gunduz-Demir

***Abstract*—This technical report contains the supplementary material for an error-driven multi-stage model that we developed for gland instance segmentation in histopathological images [1].**

***Index Terms*—Deep learning, attention learning, adaptive boosting, gland instance segmentation, instance segmentation**

I. Introduction

We recently developed a new error-driven multi-attention learning model, which we call *AttentionBoost*, for instance segmentation. This model proposes to design a multi-stage network and adaptively learn what image parts (pixels) each stage network needs to attend and the level of this attention directly on image data. For this purpose, it introduces a new loss adjustment mechanism that uses adaptive boosting for a dense prediction task [1]. This technical report provides supplementary material for additional definitions and experiments used in the evaluation of the proposed model.

II. Evaluation Metrics

In order to quantitatively evaluate the results of the proposed model and the comparison methods, three criteria are used. These are the object-level F-score, the object-level Dice index, and the object-level Hausdorff distance metrics, which were also used in the GlaS Challenge Contest [2]. The definitions of these metrics are given below.

1) F-score: A segmented gland object is considered as true positive (TP) if it intersects with at least 50 percent of a ground truth object, and as false positive (FP) otherwise. A ground truth object is considered as false negative (FN) if at least its 50 percent does not intersect with any segmented gland object. The object-level F-score is defined as:

$$\begin{aligned} \textit{F-score} &= \frac{2 \cdot \textit{precision} \cdot \textit{recall}}{\textit{precision} + \textit{recall}} \\ \textit{precision} &= |TP| / (|TP| + |FP|) \\ \textit{recall} &= |TP| / (|TP| + |FN|) \end{aligned} \tag{1}$$

2) Dice index: Let $S = \{s_i\}$ be a set of segmented gland objects in all images of a given dataset and $G = \{g_j\}$ be a set of ground truth objects in these images. To calculate the object-level Dice index on these two sets, the objects in S and G are first matched: Each $s_i \in S$ is matched with a ground truth object $\gamma(s_i) \in G$ that maximally overlaps s_i. Similarly, each $g_j \in G$ is matched with a segmented gland object $\sigma(g_j) \in S$ that maximally overlaps g_j. Then, by accumulating the Dice indices calculated for all matching object pairs, the object-level Dice index is defined as follows:

$$\textit{Dice}(S,G) = \frac{1}{2} \begin{pmatrix} \sum\limits_{s_i \in S} \omega(s_i) \cdot DI(s_i, \gamma(s_i)) \\ + \\ \sum\limits_{g_j \in G} \omega(g_j) \cdot DI(g_j, \sigma(g_j)) \end{pmatrix} \tag{2}$$

where $\omega(s_i) = |s_i| \ / \sum\limits_{s_m \in S} s_m$ and $\omega(g_j) = |g_j| \ / \sum\limits_{g_m \in G} g_m$. Here $DI(x,y) = 2 \cdot |x \cap y| / (|x| + |y|)$ is the Dice index of a pair of objects x and y, one from the segmented gland objects and the other from the ground truth objects. If there is no matching ground truth object of a segmented gland object (or vice versa), the contribution of this object to the Dice index is zero.

3) Hausdorff distance: Likewise, the objects in S and G are matched to calculate the object-level Hausdorff distance. Each $s_i \in S$ is matched with $\gamma(s_i) \in G$ that maximally overlaps s_i. If there is no overlap, $\gamma(s_i)$ is the ground truth object that has the minimum Hausdorff distance from s_i. Similarly, each $g_j \in G$ is matched with $\sigma(g_j) \in S$ that maximally overlaps g_j. If there is no overlap, $\sigma(g_j)$ is the segmented gland object that has the minimum Hausdorff distance from g_j. Then, by accumulating the Hausdorff distances calculated for all matching object pairs, the object-level Hausdorff distance is defined as follows:

$$\textit{Hausdorff}(S,G) = \frac{1}{2} \begin{pmatrix} \sum\limits_{s_i \in S} \omega(s_i) \cdot HD(s_i, \gamma(s_i)) \\ + \\ \sum\limits_{g_j \in G} \omega(g_j) \cdot HD(g_j, \sigma(g_j)) \end{pmatrix} \tag{3}$$

$HD(x,y) = \max\{\sup\limits_{p_x \in x} \inf\limits_{p_y \in y} ||p_x - p_y||, \sup\limits_{p_y \in y} \inf\limits_{p_x \in x} ||p_x - p_y||\}$ is the Hausdorff distance between a pair of objects x and y,

This work was supported by the Scientific and Technological Research Council of Turkey under the project number TÜBİTAK 116E075.

G. N. Gunesli is with the Department of Computer Engineering, Bilkent University, Ankara TR-06800, Turkey (e-mail: nur.gunesli@bilkent.edu.tr).

C. Sokmensuer is with the Department of Pathology, Medical School, Hacettepe University, Ankara TR-06100, Turkey (e-mail: csokmens@hacettepe.edu.tr).

C. Gunduz-Demir is with the Department of Computer Engineering and Neuroscience Graduate Program, Bilkent University, Ankara TR-06800, Turkey (e-mail: gunduz@cs.bilkent.edu.tr).

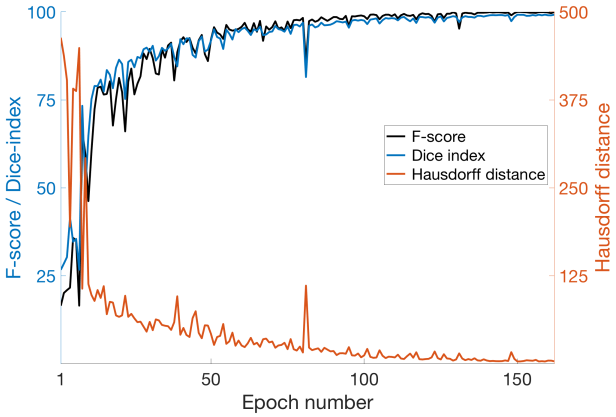


Fig. 1. F-scores, Dice indices, and Hausdorff distances as a function of the epoch number. These metrics are calculated for the training images.

one from the segmented gland objects and the other from the ground truth objects. Note that $\sup_{p_x \in x} \inf_{p_y \in y} ||p_x - p_y||$ gives the maximum of the minimum distances calculated from every pixel p_x of object x to any pixel p_y of object y.

III. Multi-Stage Network Training

We analyze the qualitative and quantitative results obtained during network training. For this purpose, for an exemplary network, the segmentation (probability) maps are generated for each training image at the end of each epoch and glands are located on these segmentation maps. Afterwards, the object-level F-score, Dice index and Hausdorff distance are calculated. Fig. 1 shows these performance metrics as a function of the epoch number. Moreover, for two selected training images (one containing normal glands and one containing cancerous glands), qualitative results are obtained at different epochs during training. These qualitative results are illustrated in Figs. 2 and 3.

IV. Parameter Analysis

AttentionBoost uses two external parameters in its gland instance segmentation step: confidence parameter α and area threshold A_{thr}. For the gland instance segmentation task, we analyze the effects of these parameters on the model's performance. To this end, for each parameter, we fix the value of the other parameter and measure the test set performance as a function of the parameter of interest.

This step inputs the average probability map $\widehat{\mathcal{Y}}_{avg}(I) = \{\hat{y}_{avg}(p)\}_{p \in I}$ for image I and locates gland objects (instances) on this map. For that, it first identifies certain gland and background pixels on which seed regions are defined. The confidence parameter α determines which pixels are considered as certain, see Eqn. 4 of the main paper [1]. When it is selected too large, only pixels p for which $\hat{y}_{avg}(p)$ is very close to 1 and very close to 0 are selected for the gland and background seed regions, respectively. Such average posteriors can only be obtained when the networks at all stages give the same output with high confidence. However, this is not an expected output of our multi-stage network, especially for hard-to-learn pixels, since it is designed with the purpose of correcting mistakes of one stage by another. Thus, larger α values result in selecting a smaller number of certain gland pixels, which decreases the number of gland seed regions to be grown. This, in turn, greatly lowers the model's performance. On the other hand, when it is selected too small, almost all pixels are considered as certain. This also lowers the performance, by leading to more undersegmented gland objects, since pixels whose $\hat{y}_{avg}(p)$ is around 0.5 are typically found on gland boundaries and these pixels are considered as certain when smaller α values are used. This analysis is depicted in Fig. 4(a).

The area threshold A_{thr} is used to eliminate small gland and background seed regions to be grown. Too small A_{thr} values cannot eliminate noisy gland seed regions, leading to false positives. On the other hand, too large A_{thr} values eliminate seed regions corresponding to small gland instances, leading to false negatives. Both lower the F-score. Note that this parameter only slightly affects the Dice index and the Hausdorff distance since they are weighted averages of these measures calculated for individual gland objects where the weights are determined by their areas. Since this elimination typically affects small glands, it does not change these measures too much. This analysis is depicted in Fig. 4(b).

This step has also an internal parameter, f_{size}, which is the size of the majority filter applied on the grown gland regions to smooth their boundaries. Although its selected value affects the appearance of gland boundaries, it only very slightly affects the performance measures since the number of boundary pixels is low. Thus, for the sake of simplicity, the smallest filter size $f_{size} = 3$ is used in the experiments.

V. Number of Parameters

In our experiments, we use the same network architecture as the base models of the comparison methods. This network architecture is given in Fig. 3 of the main paper [1]. However, for fair comparisons, we keep the number of their parameters (network weights) on par with ours by selecting an appropriate number of feature maps in their first convolutional layers. For each comparison method, Table I provides this number of feature maps as well as the number of its total network parameters. It also provides these numbers for the ablation studies, referring the corresponding table numbers in the main paper [1]. Note that the total number of network parameters also affect the computational time required for training a network. For each comparison method and ablation study, the average computational time required for the network training is also given in Table I.

VI. Effects of Using Shared Weights

In our experiments, we conduct two ablation studies to understand the effects of using different weights at each stage network of the *AttentionBoost* model. For that, we implement variants of the proposed model, in which all stage networks share weights. In the first variant, each stage network uses the same base model with the original *AttentionBoost* model.

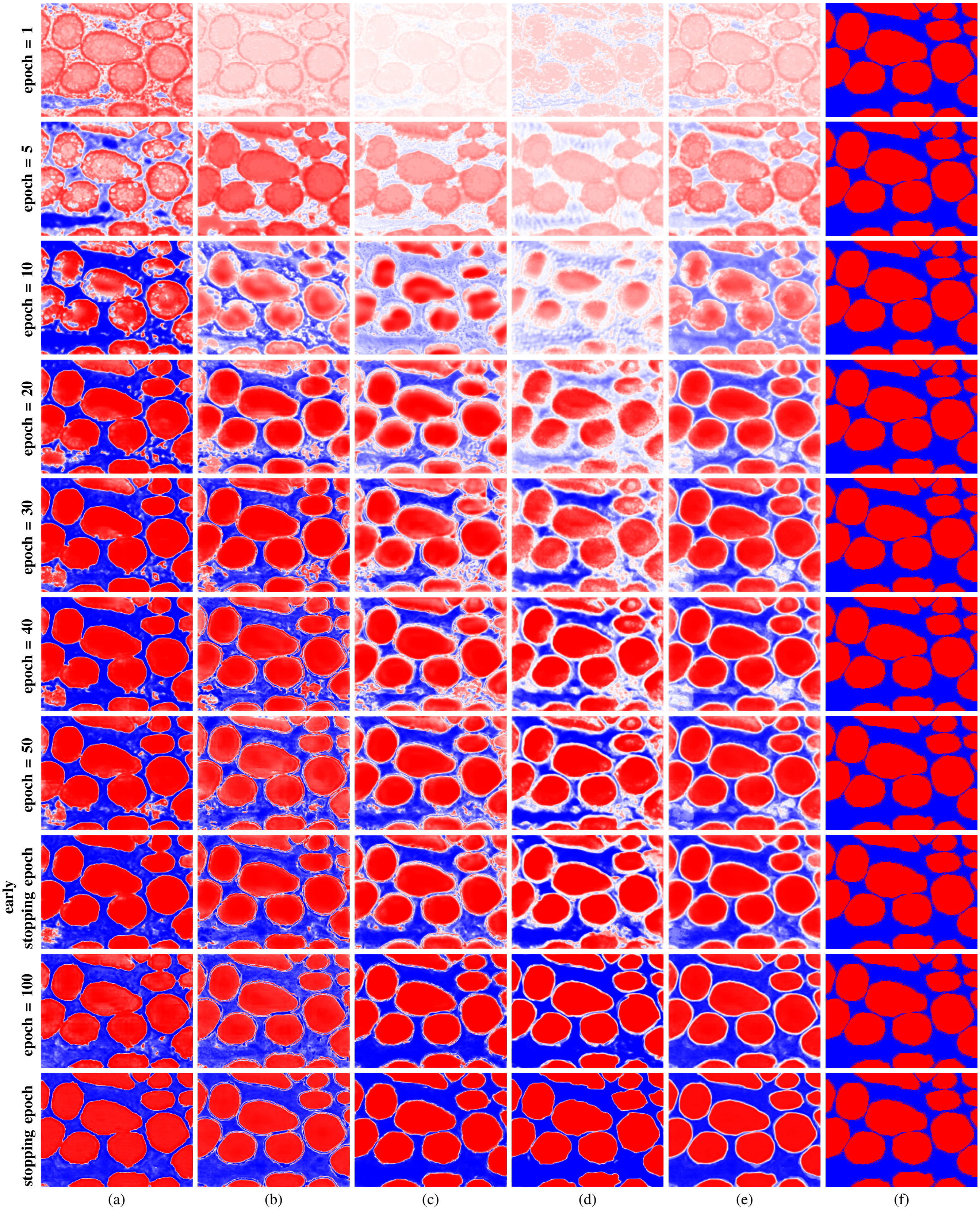


Fig. 2. Probability maps obtained for a training image containing normal glands at different epochs. (a) Posterior map $\widehat{\mathcal{Y}}_1(I)$ generated by the first stage. (b) Posterior map $\widehat{\mathcal{Y}}_2(I)$ generated by the second stage. (c) Posterior map $\widehat{\mathcal{Y}}_3(I)$ generated by the third stage. (d) Posterior map $\widehat{\mathcal{Y}}_4(I)$ generated by the fourth stage. (e) Average posterior map $\widehat{\mathcal{Y}}_{avg}(I)$ obtained by aggregating the posterior maps of all stages. (f) Posterior map $\mathcal{Y}(I)$ produced by the ground truth segmentation. These maps include pixel posteriors where 1 indicates that a pixel belongs to the gland class and 0 indicates that it belongs to the background. Posteriors between 1 and 0.5 are shown with increasing tints of red and posteriors between 0 and 0.5 are shown with increasing tints of blue. Note that in these images posteriors close to 0.5 seem whitish.

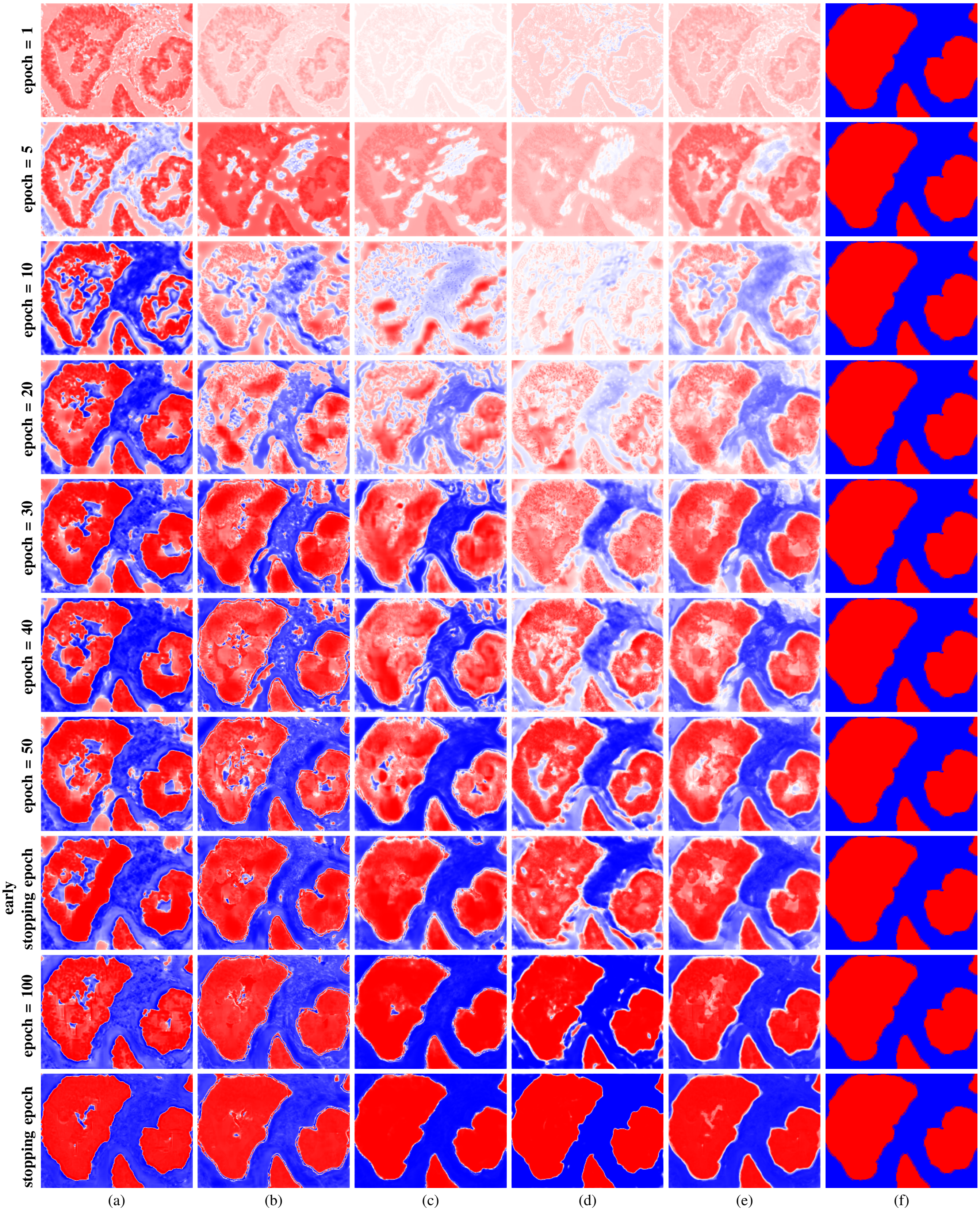
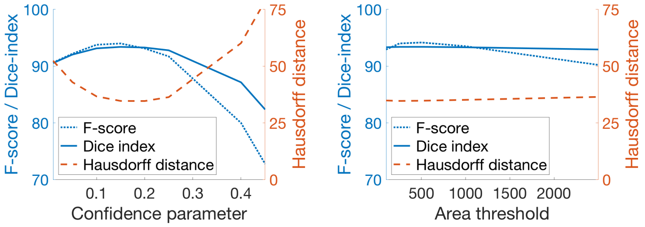


Fig. 3. Probability maps obtained for a training image containing cancerous glands at different epochs. (a) Posterior map $\widehat{\mathcal{Y}}_1(I)$ generated by the first stage. (b) Posterior map $\widehat{\mathcal{Y}}_2(I)$ generated by the second stage. (c) Posterior map $\widehat{\mathcal{Y}}_3(I)$ generated by the third stage. (d) Posterior map $\widehat{\mathcal{Y}}_4(I)$ generated by the fourth stage. (e) Average posterior map $\widehat{\mathcal{Y}}_{avg}(I)$ obtained by aggregating the posterior maps of all stages. (f) Posterior map $\mathcal{Y}(I)$ produced by the ground truth segmentation. These maps include pixel posteriors where 1 indicates that a pixel belongs to the gland class and 0 indicates that it belongs to the background. Posteriors between 1 and 0.5 are shown with increasing tints of red and posteriors between 0 and 0.5 are shown with increasing tints of blue. Note that in these images posteriors close to 0.5 seem whitish.

TABLE I

Number of feature maps used in the first convolutional layers, number of total network parameters, and computational time for network training. Gray rows indicate the table numbers in the main paper that report the results of the methods given in the rows below.

	Number of feature maps in the first convolutional layer	Number of total network parameters	Training time (seconds)
Tables II and III			
AttentionBoost	32	31,387,780	4844 ± 403
Boundary-loss-adjustment	64	31,378,945	2887 ± 155
Multi-task	54	31,264,870	3480 ± 122
Iterative	32	31,387,780	4412 ± 474
Table IV			
AttentionBoost	32	31,387,780	4844 ± 403
AttentionBoost (shared weights)	32	7,846,945	3569 ± 157
AttentionBoost (shared weights × 2)	64	31,379,521	19936 ± 955
AttentionBoost (w/o normalization)	32	31,387,780	3910 ± 165
Table V			
AttentionBoost (2-stages)	32	15,693,890	1971 ± 139
AttentionBoost (3-stages)	32	23,540,835	3595 ± 460
AttentionBoost (4-stages)	32	31,387,780	4844 ± 403
AttentionBoost (5-stages)	32	39,234,725	5277 ± 251
AttentionBoost (6-stages)	32	47,081,670	6199 ± 601
AttentionBoost (7-stages)	32	54,928,615	6662 ± 431
Table VI			
SingleStage-two classes (U-Net)	64	31,378,945	2844 ± 176
SingleStage-two classes (G-Conv)	32	31,372,161	7502 ± 743
SingleStage-two classes (G-Res)	32	33,130,721	8051 ± 202
SingleStage-three classes (U-Net)	64	31,379,075	2909 ± 156
SingleStage-three classes (G-Conv)	32	31,372,227	7678 ± 530
SingleStage-three classes (G-Res)	32	33,130,787	7983 ± 218
AttentionBoost (U-Net)	32	31,387,780	4844 ± 403
AttentionBoost (G-Conv)	16	31,373,636	8814 ± 454
AttentionBoost (G-Res)	16	33,150,020	9942 ± 188
Table VII			
AttentionBoost (same model: *4b32f*)	32, 32, 32, 32	31,387,780	4844 ± 403
AttentionBoost (different models: *4b32f*, *4b32f*, *4b64f*, and *4b64f*)	32, 32, 64, 64	78,452,932	8820 ± 592
AttentionBoost (different models: *3b32f*, *3b64f*, *4b32f*, and *4b64f*)	32, 64, 32, 64	60,755,140	7896 ± 582



(a) (b)

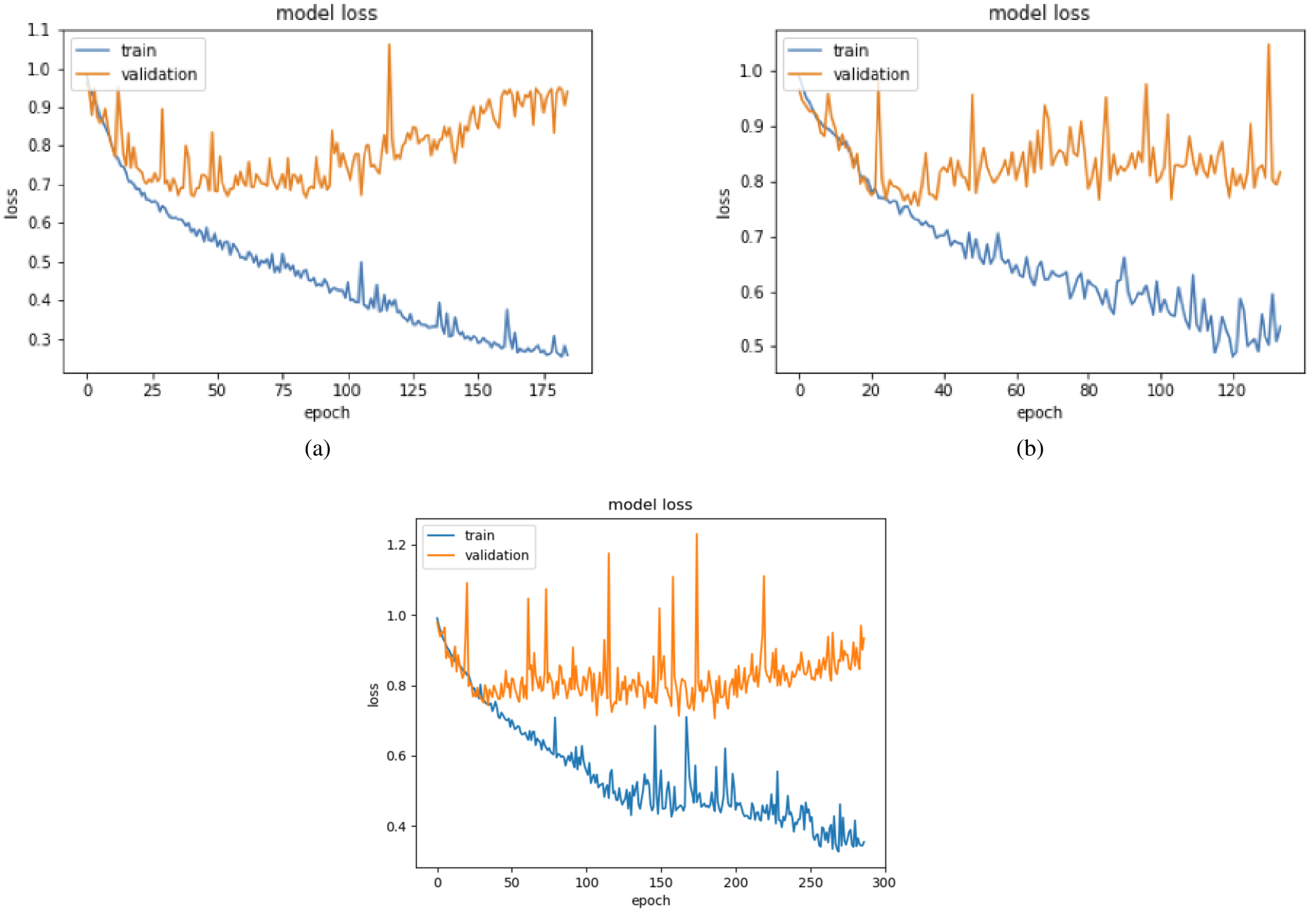
Fig. 4. Test set F-scores, Dice indices, and Hausdorff distances as a function of (a) the confidence parameter α and (b) the area threshold A_{thr}.

For this first variant, the changes in the model and in network training are summarized as follows.

- The number of parameters to be learned decreases from 31,387,780 to 7,846,945.
- The number of epochs at the stopping time (convergence point) decreases from 79.4 to 32.2 on the average (over five folds). The convergence plots for the first fold are provided in Fig.5. Note that since the training procedure uses an early stopping approach, these convergence plots are obtained until the end of the 180^{th} and 130^{th} epochs, respectively, although the convergence times are less than these numbers of epochs.
- The computational time required by each epoch remains almost the same (approximately 26-27 seconds) since the training procedure unfolds the network to be learned.
- The same training and validation sets are used for this ablation study.

Our experiments show that the first variant leads to lower performance measures. In order to understand whether the performance decrease is a result of weight sharing or due to the decrease in the parameter number, we implement the second variant, which doubles the number of feature maps in the base model. For the second variant, the changes in the model and in network training are summarized as follows.

- Doubling the number of feature maps gives a network with 31,379,521 parameters.
- The number of epochs at the stopping time greatly increases; it becomes 184.8 on the average. The convergence plot for the first fold is also provided in Fig 5.
- The computational time required by each epoch becomes approximately 70 seconds since the training procedure unfolds the network.
- The same training and validation sets are also used for this ablation study.



(a)

(b)

(c)

Fig. 5. Convergence plots for (a) the proposed *AttentionBoost* model and (b)-(c) two variants that use shared weights for all of their networks. The first variant uses the same base model with the proposed *AttentionBoost* model whereas the second variant doubles the number of the feature maps in the base model. These convergence plots are obtained for the first trained network (for the first fold). Since the training procedure uses an early stopping approach, these convergence plots are obtained until the end of the 180^{th}, 130^{th}, and 287^{th} epochs, respectively, although the convergence times (stopping points) are 80, 30, and 187 epochs for these three methods.

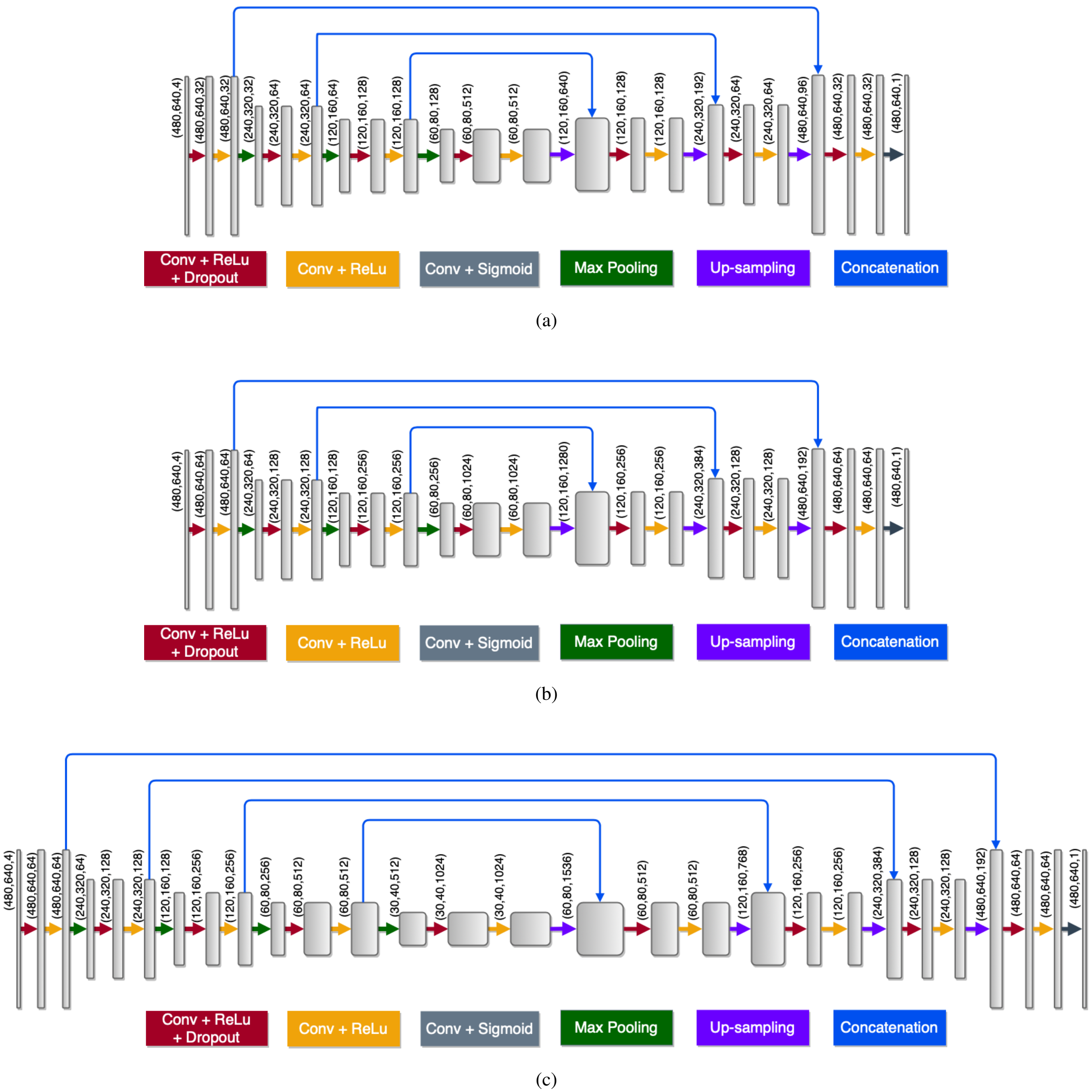
VII. Use of Different Base Models

We conduct additional experiments to investigate the effects of using different base models at different stages of our multi-stage network. For that, we use three more U-Net like networks whose architectures contain different numbers of layers and feature maps. The architectures of these networks (base models) are illustrated in Fig. 6. Similar to the one used in [1], all these networks have convolutional layers with 3×3 filters, pooling/upsampling layers with 2×2 filters, and dropout layers with a drop-out factor of 0.2. They use the sigmoid activation function at their last layers and the ReLu activation function elsewhere. Likewise, they are trained from scratch with an early stopping approach. The learning rate and the momentum value are adjusted using the AdaDelta optimizer. The selected batch size is 1.

References

[1] G. N. Gunesli, C. Sokmensuer, and C. Gunduz-Demir, "*AttentionBoost*: Learning what to attend for gland segmentation in histopathological images by boosting fully convolutional networks," *IEEE Trans. Med. Imaging*, submitted 2020.

[2] K. Sirinukunwattana *et al.*,, "Gland segmentation in colon histology images: The GlaS Challenge Contest," *arXiv preprint arXiv:1603.00275v2*, 2016.



(a)

(b)

(c)

Fig. 6. Architectures of the FCNs, referred as (a) *3b32f*, (b) *3b64f*, and (c) *4b64f*. Each box represents a feature map with its dimensions and number of channels being indicated in order on its right. Each arrow corresponds to an operation which is distinguishable by its color.