ContourNet: Salient Local Contour Identification for Blob Detection in Plasma Fusion Simulation Data

Martin Imre[1], Jun Han[1], Julien Dominski[3], Michael Churchill[3], Ralph Kube[3], Choong-Seock Chang[3], Tom Peterka[2], Hanqi Guo[2], and Chaoli Wang[1]

[1] University of Notre Dame
[2] Argonne National Laboratory
[3] Princeton Plasma Physics Laboratory

Abstract. We present ContourNet, a deep learning approach to identify salient local isocontours as blobs in large-scale 5D gyrokinetic tokamak simulation data. Blobs—regions of high turbulence that run along the edge wall down toward the diverter and can damage the tokamak—are non-well-defined features but have been empirically localized by isocontours in 2D normalized fluctuating density fields. The key of our study is to train ContourNet to follow the empirical rules to detect blobs over the time-varying simulation data. The architecture of ContourNet is a convolutional neural segmentation network: the inputs are the density field and a rasterized isocontour; the output is a set of isocontour encircling blobs. At the training stage, we feed the network with manually identified isocontours and propagated labels. At the inference stage, we extract isocontours from the segmented blob regions. Results show that our approach can achieve both high accuracy and performance, which enables scientists to understand the blob dynamics influencing the confinement of the plasma.

Keywords: XGC plasma fusion · blob detection · local isocontour selection · segmentation

1 Introduction

Fusion energy is a promising future source of energy with the minimal ecological footprint. A leading candidate for controlled fusion technique is to use a *tokamak*, which confines hot plasma in a torus-shaped reactor with a strong magnetic field. The edge of the plasma plays a dominant role in the confinement of the plasma, but needs to be managed to ensure high confinement without ruining the tokamak walls. Scientists conduct large-scale 5D gyrokinetic fusion plasma simulations using XGC [7, 8] in order to study the edge and improve the tokamak design. XGC is a particle-in-cell simulation with the output consisting of scalar fields that characterize the fluctuating level. The output is on a number of discrete 2D planes around the tokamak, with each poloidal plane representing a 2D cross-section of the simulated tokamak reactor.

The focus of this paper is the detection of *blobs*—regions of high turbulence that are thought to be a major cause of lost confinement at the edge—in fusion simulation data. Blob detection is challenging because blobs are *non-well-defined features*. While experienced scientists are able to empirically identify blobs, defining these features

mathematically remains an open challenge [6, 9]. Scientists urge the need for reliable detection methods to better understand the formation and impact of these blobs.

In practice, scientists empirically encircle a blob as a simply-connected and closed isocontour [11], or a *local contour*, in the output 2D scalar field. An example of a blob can be seen in Figure 1 (a). Scientists have been using simple heuristics to find proper local contours based on their intuitions of blobs. In general, blobs have a certain size, shape, value ranges, and could only appear in certain areas of the domain. To this end, Davis et al. [11] and Zweben et al. [29] use half-maximum levels as isovalues to compute local contours for blob detection. In Wu et al. [28], the isovalue is determined by the statistical distribution of the input data in a region of interest. However, all existing methods are based on arbitrary choices that involve manual selections and thus hard to generalize to the time-varying simulation output data.

Based on the discussion with scientists, we transform the problem of *blob detection* to *salient local isocontour detection*. Such a local treatment allows identifying each blob's isovalue independently instead of finding a universally valid heuristic for blob identification. Furthermore, such an isovalue-based detection meets the need of domain scientists. Isovalue selection has been widely studied. However, to the best of our knowledge, there exists no approach to select local, independent isovalues. Current methods for isovalue selection either analyze the entire volume and suggest global values, or perform local isovalue selection using topological analysis while still relying on a universal rule.

We instead propose a new deep-learning-based approach for detecting salient isocontours for blob identification. We present *ContourNet*, a framework that tackles the aforementioned problems using a *convolutional neural network (CNN)* to identify blobs in the XGC plasma fusion simulation data. We inspect 2D cross-sections of the reactor at a given time step. Scientists help us label blobs in these cross-sections so that we can obtain the training data used by ContourNet for training. ContourNet is a CNN that uses a combination of a 2D cross-section with a contour mask to segment blob regions in the cross-section. To avoid the enormous labeling effort, we introduce a label propagation strategy that heuristically extends the labeled data to neighboring time steps. In summary, the contributions of this work are the following:

- a salient local contour identification algorithm based on deep neural network for blob detection in the fusion plasma simulation;
- a label propagation strategy that augments labeled local contours for ContourNet training to overcome the lack of training data;
- a baseline for using CNN in blob detection with minimal labeling effort[4].

2 Background and Related Work

In this section, we formalize the blob detection problem in fusion sciences, and then review the literature on salient isovalue selection and image segmentation.

Blob detection in fusion sciences. Blob detection is regarded as an ill-posed problem in fusion sciences, because there exists no clear definition for blobs. D'Ippolito et

[4] A PyTorch implementation can be found at https://github.com/mimre25/ContourNet

al. [13] characterized blobs as higher-than-background density filaments propagating outwards and lower-than-background density filaments propagating inwards [13], and thus blobs have certain shapes, sizes, value ranges, and could only appear in certain areas of the domain.

Formally, the input of blob detection is the scalar field $f_{t,s} : \mathbb{D} \to \mathbb{R}$ on the sth 2D cross-section of the tokamak at time step t, where $\mathbb{D} \subset \mathbb{R}^2$ is the domain for a 2D cross-section, and the output is a number of regions $\{B^i_{t,s}\}$ that represent blobs. Depending on the context of different experimental and simulation studies, the input scalar field can be temperature, electron densities, scalar derivative of electrostatic potential, and normalized fluctuating density that characterize the fluctuating level. Without loss of generality, we only study blobs that are associated with local maxima. Assuming the smoothness of the input scalar field $F_{t,p}(\mathbf{x})$, we define

$$L(f_{t,s}, \theta) = \{\mathbf{x} \mid F_{t,s}(\mathbf{x}) \geq \theta\} = \bigcup_{k \in \mathbb{I}} R^k_{t,s}(\theta) \tag{1}$$

as the *super-levelset* of $f_{t,s}(\mathbf{x})$ with respect to the threshold θ; $\{R^k_{t,s}(\theta)\}$ are disjoint connected components ($R^k_{t,s}(\theta) \cap R^l_{t,s}(\theta) = \emptyset$, for any $k \neq l \in \mathbb{I}$) of the super-levelset and $\mathbb{I}$ is the index set. We further define a *blob candidate*

$$C(f_{t,s}, \theta, \mathbf{x}) = \{R^k_{t,s} \mid \mathbf{x} \in R^k_{t,s}\} \tag{2}$$

as the simply connected component that contains $\mathbf{x}$ in the super-levelset $L(f_{t,s}, \theta)$. The blobs $\{B^j_{t,s}\}$ are a subset of blob candidates that meet the empirical rules defined by scientists, where j is the index of the blob.

In practice, scientists define empirical rules to filter the candidates that connected to a few selected local maxima locations $\{\mathbf{m}^j_{t,s}\}$ ($\mathbf{m}^j_{t,s} \in \mathbb{D}$) with different super-levelset thresholds $\{\theta^j_{t,s}\}$, that is,

$$B^j_{t,s} = C(N, \theta^j_{t,s}, \mathbf{m}^j_{t,s}). \tag{3}$$

In Kube et al. [16], the threshold $\theta^j_{t,s}$ was set to 60% of the selected maximum value $f_{t,s}(\mathbf{x}_j)$. Davis et al. [11] first found large local maxima in $f_{t,s}(\mathbf{x})$ by thresholding, and then fitted an ellipse to the contour at the 50% maximum level. The same technique was used by Zweben et al. [29] and Churchill et al. [9], but the selection of local maxima is nontrivial and subject to small perturbations of $f_{t,s}(\mathbf{x})$. Based on the statistics of $f_{t,s}(\mathbf{x})$ in a small region of interest, Wu et al. [28] determined the local contour level to only incorporate regions with significantly higher temperatures and densities than the surroundings. A two-phase algorithm was proposed that first selects candidate points and then extracts blobs as the connected components of the candidates.

In this paper, we follow the convention to localize blobs by contouring. Although it is possible to approach blob detection as an object detection problem by segmenting a 2D cross-section into blob and non-blob areas, a local contour has clearer physical meaning than an arbitrary segmentation does. In general, the output segmentation is not necessarily local isocontours and thus could be misleading. Therefore, we aim to segment the image into areas that can be encircled with local isocontours.

For ease of description in terms of image input data, we define $v_{t,s}(i)$ as the $W \times H$ regular 2D image that represents $f_{t,s}(\mathbf{x})$, where $0 \leq i < W \times H$ is the the pixel index. The blob candidate $c_{t,s}(i)$ is defined as the binary (0 or 1) image of the connected component $C(f_{t,s}, f_{t,s}(\mathbf{x}_i), \mathbf{x}_i)$ that contains $\mathbf{x}_i$, the physical coordinates of i. We also define the Boolean function $b_{t,s}(i)$ to denote if the blob candidate $c_{t,s}(i)$ is a blob. The purpose of ContourNet is to learn *blob labels* $\widetilde{b}_{t,s}(i)$, in order to predict $b_{t,s}(i)$ for arbitrary t, s, and i, as detailed in further sections.

Salient isovalue selection. Our study is related to the problem of salient isovalue selection, which aims at highlighting key materials or structures in scalar field data. Salient isovalue selection algorithms can be categorized into *statistics*-, *topology*- and *similarity*-based methods.

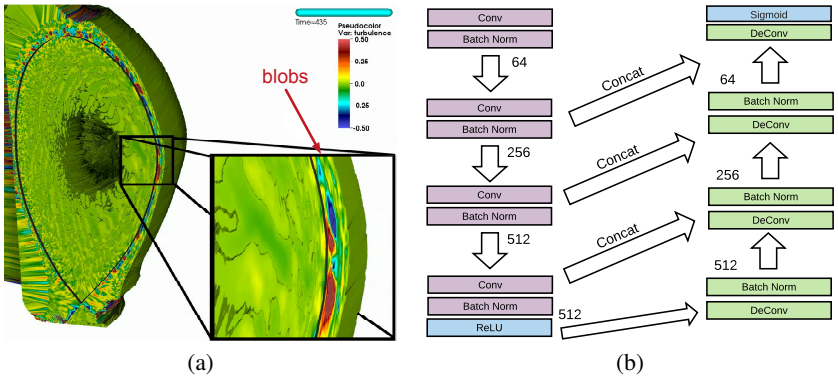
For statistics-based analysis, Bajaj et al. [1] introduced the contour spectrum to facilitate an easier choice of isovalue by displaying several statistics in a 2D curve in relation to the isovalue. Pekar et al. [22] proposed a fast one-pass algorithm to analyze volumetric datasets and find intensity transitions between different surface materials based on gray-value histograms. Carr et al. [3] showed that histogram is inferior to statistics like area, or approximation of those, e.g., triangle count, in terms of representing the underlying function distribution. Scheidegger et al. [19] revisited the relationship between histograms and the underlying geometric structure of a volume dataset. They used geometric measure theory to establish a convergence between the area statics and the histogram.

For topology-based analysis, a prominent technique is using contour trees to inspect the development of underlying geometry based on the change of isovalue. Carr et al. [4] showed that contour trees can be computed in all dimensions, while Pascucci et al. [21] introduced branch decomposition to simplify the analysis of contour trees. Later on, Carr et al. [5] introduced simplification based on local geometric measures of feature importance to allow an interactive and flexible selection of isovalues.

For similarity-based analysis, Bruckner and Möller [2] introduced isosurface similarity maps. These maps are based on the distance fields of the analyzed volume and show the inverse of the mutual information obtained from pair-wise joint histograms. They further introduced a priority-ranking algorithm based on the similarity maps. Other works have extended their approach to multimodal analysis [14, 25] or to identify nearly equally-distanced isosurfaces [15]. A shortcoming of these methods is that they all limit their selection to a single or multiple isovalues for the whole volume.

The purpose of salient isovalue selection is different from our application, because the outputs of salient isovalue selection are a levelset of a given threshold θ in the entire scalar field. We instead need to identify salient local contours defined by different levels $\{\theta^i_{t,p}\}$ locally using deep learning approaches.

Image segmentation using neural networks. Our method is inspired by U-Net [23], a deep CNN for medical image segmentation. U-Net detects cells in light microscopy images by downsampling and upsampling an input image while copying over features between the downward and upward phases. It was later adapted to work in 3D [10, 20]. Other approaches use hierarchical cascaded models [24] or create semantic segmentation of objects [17]. Nabla-net [18] uses an encoder-decoder framework or biomedical image



(a) (b)

Fig. 1. (a) shows a zoomed view of a blob. The dark red area corresponds to the blob. (b) shows the architecture of ContourNet. Similar to U-Net, ContourNet has two phases. However, there is no change of the image size. The numbers in the figure show the number of channels after a given batch normalization.

segmentation. Other researchers proposed interactive segmentation using CNNs [26, 27] to guide users during a segmentation task.

For blob detection, we cannot directly use a segmentation technique to identify salient local contours. The typical output in such a task is a pixel-wise segmentation mask which often does not encompass additional information. In our case, we need super-levelsets that are encircled by closed isocontours, which is not necessarily the case in the output of a straightforward segmentation. However, adopting such an approach to extract isocontours instead of regions with different values can overcome this.

3 Overview

Blob detection workflow. The input data of our blob detection workflow is the scalar field $f_{t,s}(\mathbf{x})$, and the outputs are a number of detected blobs $\{B^j_{t,s}\}$. The core of our approach is ContourNet, a deep neural network that takes a single slice as input and returns a segmentation for blob and non-blob regions. In this study, ContourNet is based on a modified U-Net [23], which is a CNN originally designed for medical image segmentation. For the ease of data handling using ContourNet, we transform the scalar field $f_{t,s}(\mathbf{x})$ into an image and the ground truth blob candidates into a binary mask.

We tailor both the training and inference routines of ContourNet, in order to detect blobs accurately and efficiently. At the training stage, we gather as many labeled blobs as possible through expert labeling and label propagation. We worked closely with scientists on the user interface design for expert labeling, and co-designed the automatic label propagation strategy, in order to minimize the burden of manual labeling and to maximize the blob detection accuracy.

ContourNet architecture. The original U-Net is designed for image segmentation: the input is a fixed-size image and the output is a mask that segments the different areas

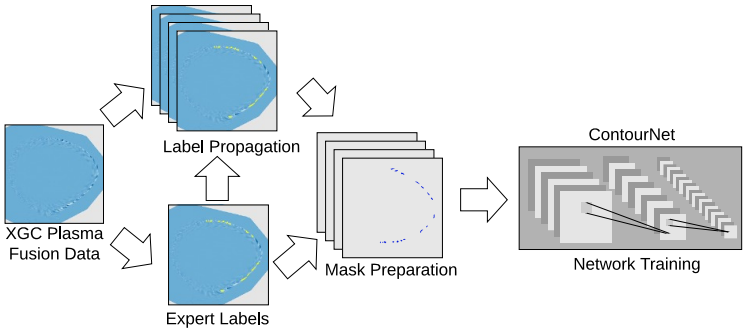


Fig. 2. The training stage for ContourNet. Expert labels are first propagated to increase the amount of training data. The data slice is then combined with a blob mask as input for network training.

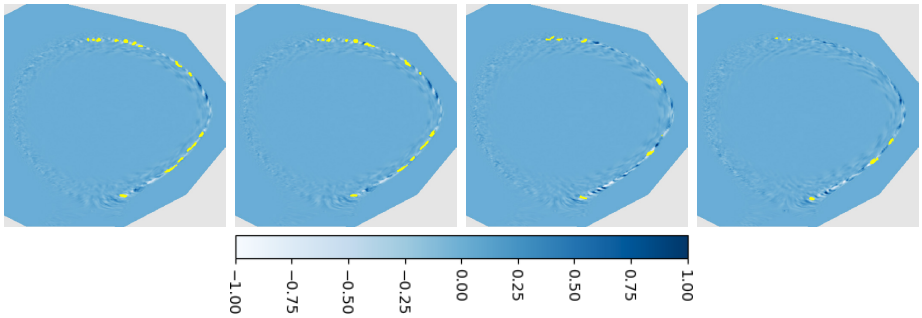
of the image. U-Net contains a downward phase and an upward phase. At each level of the downward phase, a convolutional layer halves the size of the image and doubles the size of the feature maps. Within each level of the upward phase, the exact opposite happens by applying deconvolution operations. Additionally, the features in a given level of the downward phase are copied over as additional input into the same level of the upward phase.

The modified U-Net design for ContourNet is illustrated in Figure 1 (b). ContourNet consists of four levels in both the downward and upward phases. The biggest difference to the original U-Net architecture is that ContourNet does not perform any downscaling of the input. We do this as blobs are small-scale features in comparison to the image, which could get missed in an early layer and not be recovered later. The input to each (de-)convolutional layer varies depending on its position in the network. The ith level has the same size as the input image but with a different number of features in the feature map. To copy over the features from the downward to the upward phase, we concatenate them to the already existing feature map.

The last layer produces a binary segmentation mask that we then perform the following to further transform into a mask of isocontours. First, we compute the connected components for every blob in a segmentation mask. Second, we generate a super-levelset for every point of a given connected component. Third, we compare the given super-levelset to the area from the segmentation using the Dice coefficient and pick the best fit. Finally, we extract the isocontour from the super-levelset and store the seed point for future use.

4 ContourNet Training

The training stage of ContourNet is shown in Figure 2. During training, we preprocess the simulation data into input image $v_{t,s}(i)$ and obtain labeled samples $\widetilde{b}_{t,s}(i)$ from scientists. We further improve the accuracy of ContourNet by introducing propagated labels.



(a) $(t, s) = (60, 10)$ (b) $(t, s) = (59, 10)$ (c) $(t, s) = (65, 10)$ (d) $(t, s) = (70, 10)$

Fig. 3. Four different time steps showing the same slice with the expert-labeled (a) or propagated (b-d) blobs. (t, s) are for time step ID and slice ID, respectively. We use the underlying density values $\rho_{t,s}$ as the background and overlay them with the blobs in yellow. The density values are clamped to $[-1, 1]$. The background is light gray. The images use a 400×400 pixel resolution.

Data preprocessing. Because the input image $v_{t,s}(i)$ is not directly available from the simulation, we need to preprocess the simulation data. Based on the instructions from scientists, we first derive $f_{t,s}(\mathbf{x})$ by normalizing the electrostatic potential field with the plasma temperature. The field data, which are defined on a 2D unstructured mesh, are stored in multiple HDF5 files. Because the current CNN implementations do not inherently support unstructured mesh data, we interpolate and convert the data into a 400×400 regular mesh, which is fine enough to capture blobs for this study. During mask preparation, we collect all the labeled blobs for a time step and slice combination and generate another regular 400×400 grid with blob areas set to 1 and the rest to 0.

Label propagation. We propagate the blob labels $\widetilde{b}_{t,s}(i)$ to a few neighboring time steps to increase the amount of training data and ease the amount of time that scientists need to spend on labeling. Although the propagation is an approximation and cannot be used to detect newly appeared or disappeared blobs, based on the verification from scientists, the propagated blobs are acceptable for the training purpose. A set of propagated labels can be seen in Figure 3. In each image, we show blobs as filled regions rather than contours as we used a region-based coefficient to select the best matches. In the figure, (a) shows the expert labels at time step 60, (b) is the propagation to time step 59 which shows very good quality. The propagation to time step 65 is shown in (c). Here we can already see the quality deteriorating as fewer blobs are propagated, but the output is still acceptable. In (d), we can see that the propagation to time step 70 showing even fewer blobs, resulting in a misleading training sample.

Formally, we propagate the labels $\widetilde{b}_{t,s}(i)$ to the next n time steps $\{t + 1, \ldots, t + n\}$. The propagation to preceding time steps can be done similarly. Based on the verification from scientists, we set $n = 5$ to achieve a balance between the propagation precision and the number of propagated labels. With this setting, the propagation enables us to generate ten times the labeled blobs that we obtained from the scientists.

The propagation is an iterative process over the consequent time steps. In the kth iteration, we find all blob candidates $c_{t+k,s}(i)$ that are already labeled as blobs, that is, $\widetilde{b}_{t+k,s}(i) = 1$ for all i. We then compare each $c_{t+k,s}(i)$ with all possible blob candidates $c_{t+k+1,s}(i')$ using the Dice coefficient [12]. The Dice coefficient for two binary images is defined as

$$\mathrm{dice}(I_1, I_2) = \frac{2 \times |V(I_1) \cap V(I_2)|}{|V(I_1)| + |V(I_2)|}, \tag{4}$$

where I_1 and I_2 are the two blob candidates being compared, and $V(\cdot)$ is the number of nonzero elements in the image. We then find the blob candidate $c_{t+k+1,s}(i')$ with the max Dice coefficient

$$\arg\max_{i'} \mathrm{dice}\left(c_{t+k,s}(i), c_{t+k+1,s}(i')\right), \tag{5}$$

and label $\widetilde{b}_{t+k+1,s}(i')$ as 1. If there is no overlap found, we disregard all candidates.

To generate possible blob candidates, we exhaustively sample the area close to the outer wall at a given slice s at time step t. To do so, we first filter the ring-like structure of the cross-section. Starting from the remaining area, we inspect every pixel and use it as a seed point to compute an isocontour with its corresponding value and fill the area for propagation.

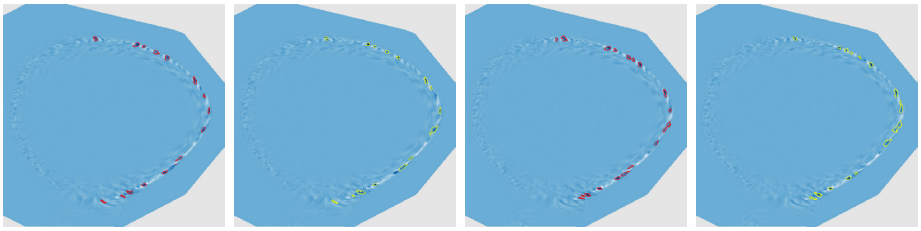
Training process. We use all labels $\widetilde{b}_{t,s}(i)$ including both expert labels and propagated labels to train ContourNet. Domain scientists labeled every slice for time step 60, yielding a total of 225 labeled blobs for 15 slices. We extend this set with our label propagation strategy to obtain a total of 165 slices label with about 15 blobs for each slice. We then train ContourNet for 100 epochs on these sets of training data. One epoch took about 52 seconds on a server with an Intel Core i7-7700K 4.2GHz quad-core processor, 32GB main memory, and an NVIDIA Titan Xp GPU. Every input slice comes with a ground-truth mask which is used to compare the weighted cross-entropy loss defined as follows

$$L = -\left(\delta B \log\left(p\right) + \gamma(1 - B) \log\left(1 - p\right)\right), \tag{6}$$

where B is the ground truth segmentation mask, p is the predicted likelihood for each pixel to be part of a blob, and δ and γ are, respectively, the weighting factors for the blob and non-blob classes. For the results shown in Section 5, we set δ to 5 and γ to 1.

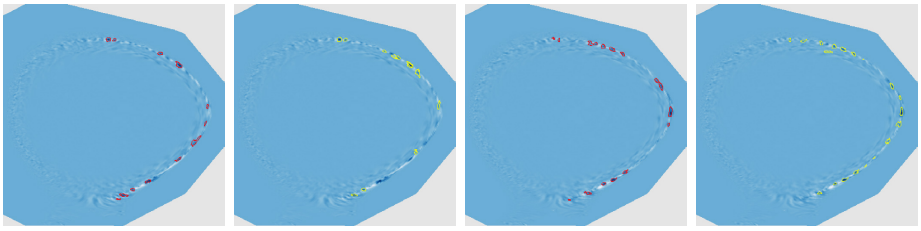
5 Results and Discussion

We used 90% of our data for training and 10% for inference. Among the testing set, the prediction yields a Dice score of 0.628. A commonly acceptable Dice score for segmentation tasks is ≥ 0.7. However, in our scenario, we lower the requirement to a score of ≥ 0.6 for two reasons. First, we compare inferred results to heuristically generated and noisy "ground truth" (propagated results). Second, small-scale blobs are very difficult to segment and jointly, they only account for a fraction of the image. The



(a) $(t, s) = (60, 6)$ (b) $(t, s) = (60, 6)$ (c) $(t, s) = (60, 7)$ (d) $(t, s) = (60, 7)$

Fig. 4. Comparing ContourNet's segmentation results to the ground truth. (a) and (c) show the predictions (red), whereas (b) and (d) show the ground truth (yellow).



(a) $(t, s) = (56, 8)$ (b) $(t, s) = (56, 8)$ (c) $(t, s) = (56, 12)$ (d) $(t, s) = (56, 12)$

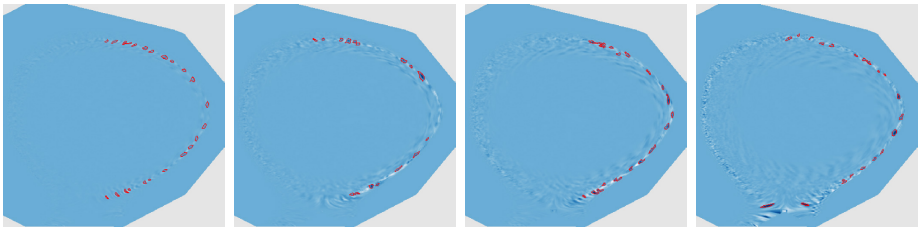
Fig. 5. Comparing ContourNet's segmentation results to the propagated ground truth. (a) and (c) show the predictions (red), whereas (b) and (d) show the ground truth (yellow).

average prediction time was 7.56 seconds (segmentation 0.52s, contour extraction 7.04s). In the following, we discuss selective results in detail.

Comparison to the ground truth. Figure 4 shows the comparison of our prediction and the ground truth at time step 60. Note that we show the images side by side to avoid the overlap of contours. In (a), we can see that the overall prediction for slice 6 is very good. While some of the blobs differ in the shape or position from the ones shown in (b), the differences are relatively low. In (c), we can see that ContourNet detects more blobs than shown in the ground truth for slice 7. Besides that, some of the blobs have different shapes (bottom right) and there are a few missing ones (top center and right). These results are mostly false positives. We point out that the goal of ContourNet is to assist scientists in identifying blobs, and therefore, false positives are better than a false negatives.

Figure 5 shows a comparison of our prediction and the propagated labels at time step 56. (a) and (b) show slice 8 with ContourNet predicting some false positives (center right, bottom). However, the shapes of the detected blobs seem fairly accurate. In (c) and (d), we can see that the prediction for slice 12 is more accurate but misses some of the blobs (top center, right middle).

Expert evaluation. We conduct an expert evaluation of ContourNet with fusion scientists. Their feedback is as follows. All in all, ContourNet correctly identifies relevant potential blob candidates across a large spectrum of different simulation data. Figure 6



(a) $(t, s) = (30, 7)$ (b) $(t, s) = (55, 9)$ (c) $(t, s) = (70, 10)$ (d) $(t, s) = (170, 6)$

Fig. 6. ContourNet segmentation results where contours are highlighted in red.

shows toroidal slices of the simulation data at four different time steps. These instances show a variety of situations that typically occur in simulations. In (a), a large number of potential blobs can be seen at the magnetic separatrix. ContourNet identifies the relevant large-amplitude regions in the data and visually, no other region would be a blob candidate. The situation in (b) is more difficult. Again, ContourNet correctly identifies the relevant large-amplitude regions of the image. Note that these regions appear more stretched out along the poloidal direction than in (a). On the other hand, the negative regions in the outboard mid-plane region might be relevant for the physics of the system. In (c), perturbations of the electric potential can be seen all along the separatrix. Here ContourNet identifies contour regions only on the rightmost half, the region where blob dynamics are most relevant to plasma confinement. This inference result is significantly better than the corresponding propagated result shown in Figure 3 (d). The simulation data shown in (d) are a difficult test case. Again, small-scale perturbations of the potential appear all along the magnetic separatrix. These stretch out over an X-shaped region at the bottom of the simulation domain. ContourNet identifies multiple blob instances, two of them in the X-shaped bottom region. This area is relevant for the physics underlying blob motion. There are no regions identified in the left half of the image, a region that is not very relevant for blob motion.

6 Conclusions and Future Work

We have presented ContourNet, a deep CNN to detect blobs as local isocontours in the XGC plasma fusion simulation. ContourNet learns from expert labeled data and profits from a label propagation strategy that allows us to heuristically label neighboring time steps from the ones labeled by experts. We further present expert evaluation agreeing with the performance of ContourNet. Our work provides a baseline for future improvement that uses only minimal labeling effort.

For future work, an interesting addition to ContourNet would be semi-supervised learning, or better, an active learning concept. Using ContourNet to detect blobs and incorporating the feedback from domain experts during training could be beneficial for the performance, especially when considering that it mostly produces false positives and suffers from a low amount of training data. A further extension would be the discrimination between different kinds of blobs. In fusion sciences, there are distinctions

among different kinds of turbulent eddies, such as blobs and streamers. Finally, using ContourNet for in-situ blob detection in coupled simulation codes would be beneficial for scientists to gain better insights into the underlying physics of blobs.

Acknowledgments

This research was supported in part by the U.S. National Science Foundation through grants IIS-1455886, CNS-1629914, and DUE-1833129, and the U.S. Department of Energy through grant DE-AC02-06CH11357 and the Exascale Computing Project (17-SC-20-SC).

References

1. Bajaj, C.L., Pascucci, V., Schikore, D.R.: The contour spectrum. In: Proceedings of IEEE Visualization Conference. pp. 167–173 (1997)
2. Bruckner, S., Möller, T.: Isosurface similarity maps. Computer Graphics Forum **29**(3), 773–782 (2010)
3. Carr, H., Brian, D., Brian, D.: On histograms and isosurface statistics. IEEE Transactions on Visualization and Computer Graphics **12**(5), 1259–1266 (2006)
4. Carr, H., Snoeyink, J., Axen, U.: Computing contour trees in all dimensions. In: Proceedings of ACM Symposium on Discrete Algorithms. pp. 918–926 (2000)
5. Carr, H., Snoeyink, J., Van De Panne, M.: Flexible isosurfaces: Simplifying and displaying scalar topology using the contour tree. Computational Geometry **43**(1), 42–58 (2010)
6. Chang, C.S., Ku, S., Tynan, G.R., Hager, R., Churchill, R.M., Cziegler, I., Greenwald, M., Hubbard, A.E., Hughes, J.W.: Fast low-to-high confinement mode bifurcation dynamics in a tokamak edge plasma gyrokinetic simulation. Physical Review Letters **118**(17), 1–6 (2017)
7. Chang, C.S., Ku, S.: Spontaneous rotation sources in a quiescent tokamak edge plasma. Physics of Plasmas **15**(6), 062510 (2008)
8. Chang, C.S., Ku, S., Diamond, P., Lin, Z., Parker, S., Hahm, T., Samatova, N.: Compressed ion temperature gradient turbulence in diverted tokamak edge. Physics of Plasmas **16**(5), 056108 (2009)
9. Churchill, R.M., Chang, C.S., Ku, S., Dominski, J.: Pedestal and edge electrostatic turbulence characteristics from an XGC1 gyrokinetic simulation. Plasma Physics and Controlled Fusion **59**(10), 105014 (2017)
10. Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T., Ronneberger, O.: 3D U-Net: Learning dense volumetric segmentation from sparse annotation. In: Proceedings of International Conference on Medical Image Computing and Computer Assisted Interventions. pp. 424–432 (2016)
11. Davis, W.M., Ko, M.K., Maqueda, R.J., Roquemore, A.L., Scotti, F., Zweben, S.J.: Fast 2-D camera control, data acquisition, and database techniques for edge studies on NSTX. Fusion Engineering and Design **89**(5), 717–720 (2014)
12. Dice, L.R.: Measures of the amount of ecologic association between species. Ecology **26**(3), 297–302 (1945)
13. DIppolito, D., Myra, J., Zweben, S.: Convective transport by intermittent blob-filaments: Comparison of theory and experiment. Physics of Plasmas **18**(6), 060501 (2011)
14. Haidacher, M., Bruckner, S., Groller, E.: Volume analysis using multimodal surface similarity. IEEE Transactions on Visualization and Computer Graphics **17**(12), 1969–1978 (2011)

15. Imre, M., Tao, J., Wang, C.: Identifying nearly equally spaced isosurfaces for volumetric data sets. Computers & Graphics **72**, 82–97 (2018)
16. Kube, R., Garcia, O.E., LaBombard, B., Terry, J., Zweben, S.: Blob sizes and velocities in the alcator c-mod scrape-off layer. Journal of Nuclear Materials **438**, S505–S508 (2013)
17. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition. pp. 3431–3440 (2015)
18. McKinley, R., Wepfer, R., Gundersen, T., Wagner, F., Chan, A., Wiest, R., Reyes, M.: Nabla-net: A deep dag-like convolutional architecture for biomedical image segmentation. In: Proceedings of International Workshop on Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries. pp. 119–128 (2016)
19. Meyer, M., Scheidegger, C.E., Schreiner, J.M., Duffy, B., Carr, H., Silva, C.T.: Revisiting histograms and isosurface statistics. IEEE Transactions on Visualization and Computer Graphics **14**(6), 1659–1666 (2008)
20. Milletari, F., Navab, N., Ahmadi, S.A.: V-Net: Fully convolutional neural networks for volumetric medical image segmentation. In: Proceedings of International Conference on 3D Vision. pp. 565–571 (2016)
21. Pascucci, V., Cole-McLaughlin, K., Scorzelli, G.: Multi-resolution computation and presentation of contour trees. In: Proceedings of IASTED Conference on Visualization, Imaging, and Image Processing. pp. 452–290 (2004)
22. Pekar, V., Wiemker, R., Hempel, D.: Fast detection of meaningful isosurfaces for volume data visualization. In: Proceedings of IEEE Visualization Conference. pp. 223–230 (2001)
23. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: Proceedings of International Conference on Medical Image Computing and Computer Assisted Interventions. pp. 234–241 (2015)
24. Seyedhosseini, M., Sajjadi, M., Tasdizen, T.: Image segmentation with cascaded hierarchical models and logistic disjunctive normal networks. In: Proceedings of IEEE International Conference on Computer Vision. pp. 2168–2175 (2013)
25. Tao, J., Imre, M., Wang, C., Chawla, N.V., Guo, H., Sever, G., Kim, S.H.: Exploring time-varying multivariate volume data using matrix of isosurface similarity maps. IEEE Transactions on Visualization and Computer Graphics **25**(1), 1236–1245 (2019)
26. Wang, G., Zuluaga, M., Li, W., Pratt, R., Patel, P., Aertsen, M., Doel, T., David, A., Deprest, J., Ourselin, S., et al.: DeepIGeoS: A deep interactive geodesic framework for medical image segmentation. arXiv preprint arXiv:1707.00652 (2017)
27. Wang, G., Li, W., Zuluaga, M.A., Pratt, R., Patel, P.A., Aertsen, M., Doel, T., David, A.L., Deprest, J., Ourselin, S., et al.: Interactive medical image segmentation using deep learning with image-specific fine tuning. IEEE Transactions on Medical Imaging **37**(7), 1562–1573 (2018)
28. Wu, L., Wu, K.J., Sim, A., Churchill, M., Choi, J.Y., Stathopoulos, A., Chang, C.S., Klasky, S.: Towards real-time detection and tracking of spatio-temporal features : Blob-filaments in fusion plasma. IEEE Transactions on Big Data **2**(3), 262–275 (2016)
29. Zweben, S., Davis, W., Kaye, S., Myra, J., Bell, R., LeBlanc, B., Maqueda, R., Munsat, T., Sabbagh, S., Sechrest, Y., Stotler, D., the NSTX Team: Edge and SOL turbulence and blob variations over a large database in NSTX. Nuclear Fusion **55**(9), 093035 (2015)