Multiscale Fully Convolutional Network with Application to Industrial Inspection

Xiao Bian Ser Nam Lim Ning Zhou
GE Global Research
{xiao.bian, limser, ningzhou}@ge.com

Abstract

In recent years, deep learning, particularly Convolutional Neural Network (CNN), has shown great efficacy for solving various vision tasks. In image segmentation, it has been demonstrated that a CNN can greatly outperform other approaches. However, special attention has to be paid towards setting various parameters in the CNN that affects the scale of the feature map generated at the last convolutional layer, where scale here refers to the ratio of the number of pixels in the original input image that correspond to each pixel in the feature map. Quite often, the optimal settings are tied to the specific problem on hand and can be fairly challenging to determine. To overcome such an issue, this paper proposes a multiscale Fully Convolutional Network (FCN) that combines networks trained at various scales, thereby allowing for conducting segmentation more generically. Moreover, such a multiscale architecture allows for incremental fine-tuning as more training images become available later on and new networks can be trained and added to the combined network. Such flexibility has great utility in applications such as industrial inspection, where training images may not be readily available initially, but yet requires a high level of accuracy. This paper will validate our findings by reporting the results that we have obtained by applying multiscale FCN to the inspection of aircraft engine part.

1. Introduction

Deep learning, particularly Convolutional Neural Network (CNN), has drawn keen interest from both academia and industry for its superior performance on various vision tasks. "Coarse inference" problems such as Image classification [8, 14], object detection [5, 18], pose estimation [15], and "fine inference" problems such as semantic segmentation [7, 9] have all greatly benefited from the advances in deep learning. Different from typical two-stage approaches that involve hand-crafted feature extraction and classifier design [12, 17], CNN automatically learns the most informative features from the input images themselves for the task involved by using a stack of convolutional layers [8]. Additionally, the breadth (connectivity) between these convolutional layers and depth (number of convolutional layers) make the CNN very effective for representing and solving highly nonlinear problems, which is typical of most real world problems [14].

While previous work [5, 8, 9, 18] has attempted to address vision problems generically, this paper is concerned with the flexibility with which the success of deep learning can be readily harnessed in different applications. To this end, we need to alleviate the amount of training image annotation requires while producing trained networks that are highly accurate. The availability of annotated datasets such as ImageNet [2] can hardly be assumed in many applications, particularly industrial inspection, which is a focus of this paper. In industrial inspection, we are quite often faced with the challenges of changing environment and short deployment cycles, thereby requiring fine tuning of existing trained network in a timely fashion while achieving the same level of accuracy.

The goal of this paper is to introduce a multiscale Fully Convolutional Network (FCN) that consists of a summarization layer capable of combining different networks that have been trained independently. For a set of annotated training images, we can use it to train different networks at different scales, helping to ease the amount of training images that may be required. Scale is defined as the size of the image patch in the *original input image* which connect to the corresponding pixel in the feature maps of the last convolutional layer of the network. Several parameters influence the scale of a network, including depth of the network, receptive fields used in each convolutional layer, and stride settings. While the smaller the scale of a network, the less susceptible it is to variations in the image appearance, a smaller scale may not be able to sufficiently capture the semantic of the segmentation task. In contrast, a larger scale network is adept at capturing the semantic but faces larger amount of variations, which will require more train-

ing images to overcome. By combining networks trained independently at different scales, the hope is to balance the need for accuracy, model size and time taken in fine-tuning a network.

We will report results of applying such a multiscale FCN to a specific industrial inspection problem, which involves the automatic segmentation of regions in an aircraft engine blade where coating has been lost. In this application, the said blade is coated with thermal barrier coating when newly made to protect the metal alloy underneath from the heat generated during the operation of the aircraft engine. As the blade experiences huge swings in heating and cooling, the coating wears off and exposes the metal alloy underneath. Such a process of losing coating is commonly known as spallation. By being able to accurately quantify the spallation process using image segmentation, it allows for inspectors to detect the problem early on and take corrective action. The proposed multiscale FCN has demonstrated very good accuracy for this application.

The rest of this paper is organized as follows. In Section 2, we review related works on semantics segmentation and specifically focus on a brief review of fully convolutional networks (FCN). In Section 3, we introduce our architecture of mutiscale CNN built upon FCN to accommodate the needs of aircraft engine blade inspection. In Section 4, we present our end-to-end engine blade inspection system and further evaluate the performance of different neural network configurations on engine blade inspection in Section 5. Finally, we conclude in Section 6.

2. CNN for segmentation

2.1. Related work

Deep learning on pixel-wise inference has make impressive progress on general vision tasks as scene labeling [3, 13] and semantic segmentation [1, 6, 9, 7]. In [1] and [3], CNN outputs from different scales of the input image are combined for semantic segmentation. In [6], contour detection and region proposal are used as pre-processing steps. In addition, CNN depth features and CNN RGB features are combined for semantic segmentation. In [7], CNN features from lower layers and higher layers are both used for segmentation. In particular, the low resolution features from higher layers are up-sampled to match the resolution of lower layers. In [9] fully convolutional network is studied on semantic segmentation. The networks is trained end-to-end without any pre-processing steps. Features from both lower layers and higher layers are summed together for a dense prediction. The proposed network exceeds the state of the art on several semantic segmentation dataset.

Compare to prior arts of deep learning on segmentation using patch-wise training [1, 3, 13, 6, 4, 11], our work exploit the translation invariant property and computational efficiency of fully convolutional network (FCN) [16, 9]. There is no pre-processing step in our design such as contour detection and region proposal in [6], neither post-processing as super-pixel projection in [3]. Additionally, in [3], the concept of "multiscale" is for images at different scales. However, in our work, we construct networks at different scale rather than using images at different scales as input to the same network.

Our design has a similar philosophy as [9] and [7] in terms of integrating features across both lower and higher layers. Different from [9] and [7] that fuse features from various layers of the same network, we train individual networks to be expert at their own scales before all information are fused into one network.

2.2. Fully convolutional network

Fully convolutional network (FCN) has been explored starting from [16] and [10] for detection. Recently [9] use FCN on both training and testing, and achieve superior performance on semantic segmentation.

FCN can be seen as a CNN using convolutional layers instead of any fully-connected layers. Specifically, FCN has the translation invariant property to allow any size of image as the input of the network. This property provides the flexibility to process images of different size. Moreover, in our Multiscale FCN(M-FCN) architecture, it allows us to train individual networks using images of different sizes and still able to integrate as one new network. The translation invariant property is the result of the local connectivity of FCN. In particular, the potential layers in FCN, e.g. convolution, ReLu, Pooling and deconvolution/upsampling, etc, all have the property of local connectivity and therefore the output of each neuron only depends on relative locations of a input value rather than its global location.

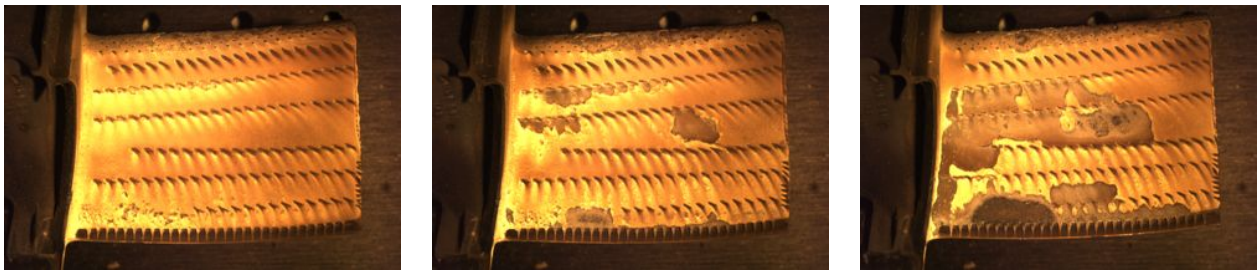
Furthermore, the computation in FCN, i.e. feed-forward pass and back-propagation, is also computationally more efficient compare to treating each patch independently. Processing an image as a whole rather than a set of patches, the overlapped region of these patches will be calculated only once in both feed-forward pass and back-propagation.

3. Multiscale fully convolutional network

We design our network architecture based on FCN to exploit the superior efficiency on both feed-forward pass and back-propagation compared to patch-based processing. In this section, we first introduce the scale of FCN, and then illustrate the design of multiscale FCN for segmentation.

3.1. Scale of FCN

FCNs of various architectures may have different *scales*, the connected pixels in the input image to an output neuron. Intuitively, the scale of a FCN is the "field of vision" of each



(a) (b) (c)

Figure 1: Examples of aircraft engine blades at different distress levels. Note that the cooling holes on each blade are not part of spallation.

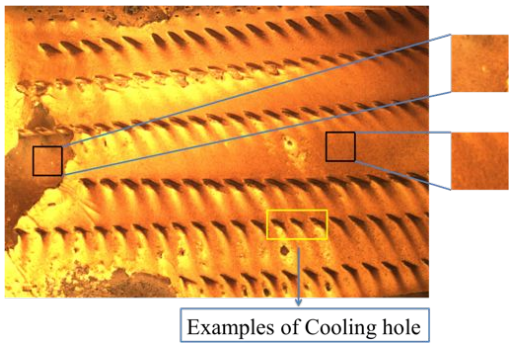


Figure 2: Small patches of spallation/non-spallation is not distinguishable, implying that a larger field of vision is necessary for a successful segmentation. Cooling hole is also indicated here.

output neuron, the local patch in the input image that the inference of each pixel in the output map relies on. The scale size can be as small as one pixel or as big as the entire image. In the case of one-pixel scale, we simply use the very local information, the pixel value, to give a dense inference; and when the scale is as big as the entire image, we predict a label of the given image just like image classification.

In particular, assume a network of T layers, and the kernel size of layer i is (w_i, h_i), s_i the stride of layer i. Let (v_i^w, v_i^h) be the visual field of top layer T to layer i. We can calculate (v_i^w, v_i^h) as

$$v_i^w = v_{i+1}^w s_i + w_i - 1, \quad (1)$$

$$v_i^h = v_{i+1}^h s_i + h_i - 1, \quad (2)$$

$$v_T^w = w^T, v_T^h = h^T. \quad (3)$$

The scale of a network is then defined as the size of the receptive field of the top layer, (v_0^w, v_0^h).

Networks of different scales process image structures of different size. For fine inference tasks like segmentation, information from both small scales and large scales are helpful. Specifically, networks of a smaller scale can use simpler structures and less parameters, since structures at a smaller scale inherently has less degrees of freedom and are hence more resilient to overfitting. This property is particularly helpful when the training data are limited. However, some inference are highly nonlinear and only a larger view can provide sufficient information to make a correct decision. For example, in the problem of spallation segmentation of aircraft engine blade, as shown in Fig 2, it is very hard to tell if a certain area is of spallation until we see the boundary of the entire area of spallation.

3.2. Multiscale architecture

We integrate networks of different scales in order to achieve a better performance on fine inference. In particular, there are two stages in the learning process:

1. Train individual networks at different scales independently.
2. Combine networks of different scales into a new network and fine-tune its last layer.

In Stage 1, the network is configured as shown in Fig. 3(b). As an FCN, all layers, e.g. convolutional layer, pooling layer, Relu layer, are translation invariant. Note that in order to fuse features from networks at different scales, upsampling is necessary for networks of a small output map. In each individual network, we carefully choose the padding parameter of each convolutional layer so that the size of the feature maps only changes during pooling and upsampling.

We train each individual network using images and their corresponding segmentation map. Note that since the networks are trained independently, they can use different dataset for training at different times. A newly trained FCN can be added into the multiscale model very efficiently as described in Stage 2 below as only the weights of the last layer in the multiscale model needs to be updated allowing for the training to converge very quickly. We use logistic regression as the loss function for each network.

In Stage 2, as shown in Fig. 3(a), the feature maps of network at different scales are concatenated as the input to a new logistic regression classifier. Only the parameters of logistic regression are fine-tuned in the new network. Intuitively, these input vectors from each individual network have already provide decent segmentation results. By concatenating these vectors at different scales, we have the information from different scales to do a fine-tuned segmentation. With limited training data, as the scale of a FCN gets larger, the network will inevitably overfit. However, this overfitting problem can be significantly alleviated by integrating networks of different scales and fine-tuning the final model using a different set of samples.

In particular, consider the output layer of FCN_i is a linear classifier $f(\mathbf{w}_i^T \mathbf{x}_i)$, where $\mathbf{w}_i$ is the vector of weights and $\mathbf{x}_i$ is the input vector of the classifier of FCN_i. We can then write the combined feature vectors from all FCN_i as $\mathbf{x}_{all} = [\mathbf{x}_1^T | \mathbf{x}_2^T | \cdots | \mathbf{x}_n^T]^T$. Assume the loss function is convex such as linear regression or SVM, etc, written as $L(y, f(\mathbf{w}, \mathbf{x}))$, where y is the true label. It follows that the fine-tuned result of M-FCN satisfies

$$L(y, f(\mathbf{w}_{all}^T \mathbf{x}_{all})) \leq min_i \{L(y, f(\mathbf{w}_i^T \mathbf{x}_i)\}, \quad (4)$$

since $L(\cdot)$ is convex and

$$\mathbf{w} = [\mathbf{0}^T | \cdots | \mathbf{0}^T | \mathbf{w_i^T} | \mathbf{0}^T | \cdots | \mathbf{0}^T]^T$$

is an obvious solution of $L(y, f(\mathbf{w}_{all}^T \mathbf{x}_{all}))$. This conclusion supports the method we use to combine the feature vectors from FCNs at multiple scales.

In Stage 2, we can use the same dataset as Stage 1 or an entirely new dataset. In our experiments, we use the same dataset as Stage 1 for a fair comparison with other architecture. However, during the deployment of this system to an engine inspection shop, we can use a new dataset in this fine-tune stage.

4. An end-to-end aircraft engine blade inspection system

Aircraft engine blade inspection requires an estimation of the distress level of a given engine blade, i.e. the loss of coating of the engine blade. Currently, human inspectors visually check the condition of each blade, and then give a rank of the distress level of the blade accordingly, i.e. rank 0 - 9. In this section, we introduce the proposed end-to-end system providing an accurate estimation of the distress level of engine blades.

Since the distress level of each blade is determined by the coating spallation, we use M-FCN to provide an estimation of spallation area. Specifically, we define *spallation score* as

$$S = \frac{\sum_{i=1}^{n} f(p_i)}{A}, \quad (5)$$

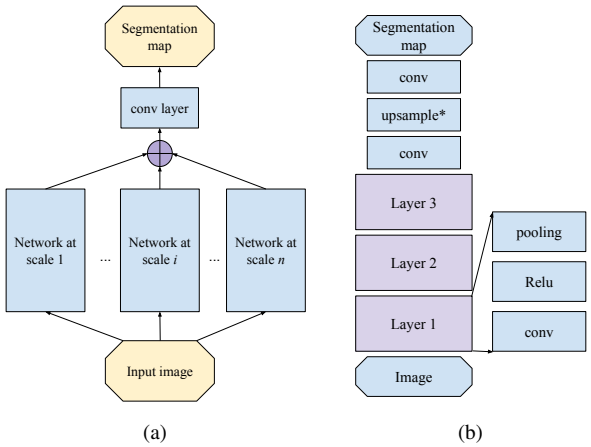


Figure 3: Network architectures

where p_i is the ith pixel of interest, A is the total area of interest and $f(\cdot)$ is the function of a pixel representing the spallation level. Here we set $f(p_i) = 1$ when p_i is classified as spallation and 0 otherwise.

In this system, the front-end camera takes images of blades on a fixture, and then the CNN-based back-end calculates the corresponding spallation map and uses the spallation score to estimate the distress level of the given blade. The whole system using a computer of 2.6GHz Intel Core i7 without GPU support can process an image of 4896×3264 in approximately 0.5s. As long as the CNN-based segmentation algorithm provides an accurate spallation map, the system is capable of giving a much finer estimation of engine blades distress level much faster compare to human labor.

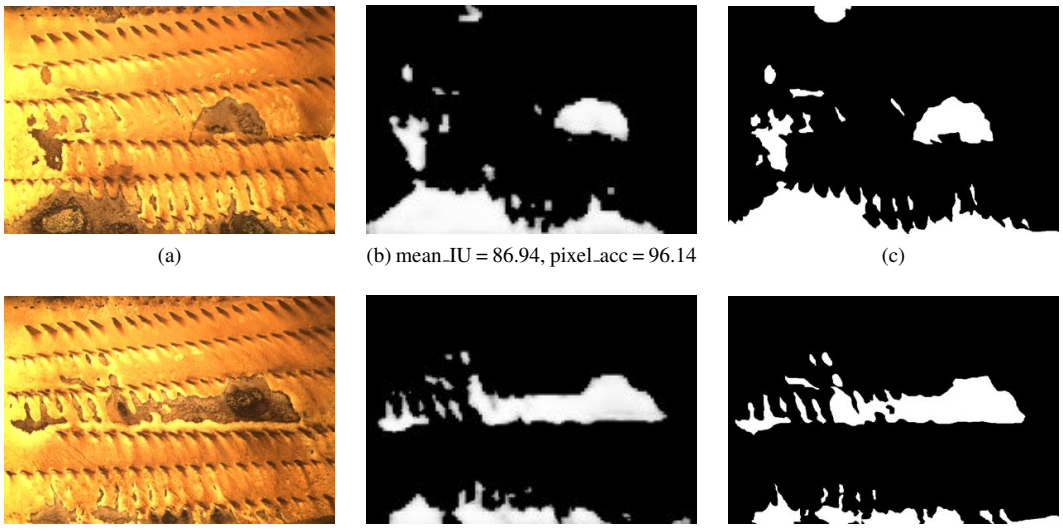
5. System evaluation

The performance of this engine blade inspection system is determined by the accuracy of the segmentation of spallation area. In this section, we build an aircraft engine blade dataset and evaluate our architecture on this segmentation problem this dataset, and further compare the M-FCN architecture with other designs and human annotation.

5.1. A dataset of engine blades at different distress levels

In order to evaluate the performance of this CNN-based engine blade inspection system, we collect images of engine blades from different engines of various conditions. In Fig. 4, we show examples of engine blade from two engines. Note that blades from the same engine can have various distress levels with different spallation map.

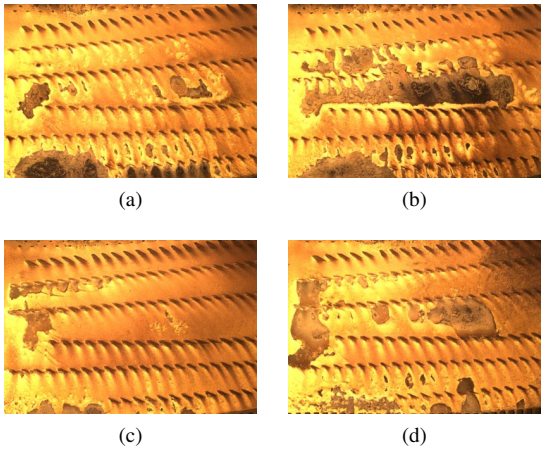
In total we collect 256 images from 4 engines, 64 blades each. Since blades from the same engine share some com-



(a) (b) mean_IU = 86.94, pixel_acc = 96.14 (c)

(d) (e) mean_IU = 88.46, pixel_acc = 95.96 (f)

Figure 5: Some segmentation examples



(a) (b)

(c) (d)

Figure 4: Examples of engine blades. The blades in the same row are from the same engine. Note that the rows of small holes are cooling holes of engine blades. They are not part of the spallation area.

mon features, we use blades from 2 engines as training set and blades from other 2 engines as test set to verify the generalizability of our system. We annotate the spallation area of each blade as the ground truth for training and testing.

5.2. Experimental results

We evaluate the proposed M-FCN and two other FCNs on the task engine blade segmentation. The architectures of the two FCNs are as follows:

- FCN_1: input → conv(24x9x9) → ReLU → pooling(2x2) → conv(24x9x9) → ReLU → pooling(2x2) → conv(200x13x13) → ReLU → conv(2x1x1) → output,
- FCN_2: input → conv(24x9x9) → ReLU → pooling(2x2) → conv(24x9x9) → ReLU → pooling(2x2) → conv(32x5x5) → ReLU → pooling(2x2) → conv(200x11x11) → ReLU → deconv(200x2x2) → conv(2x1x1) → output,

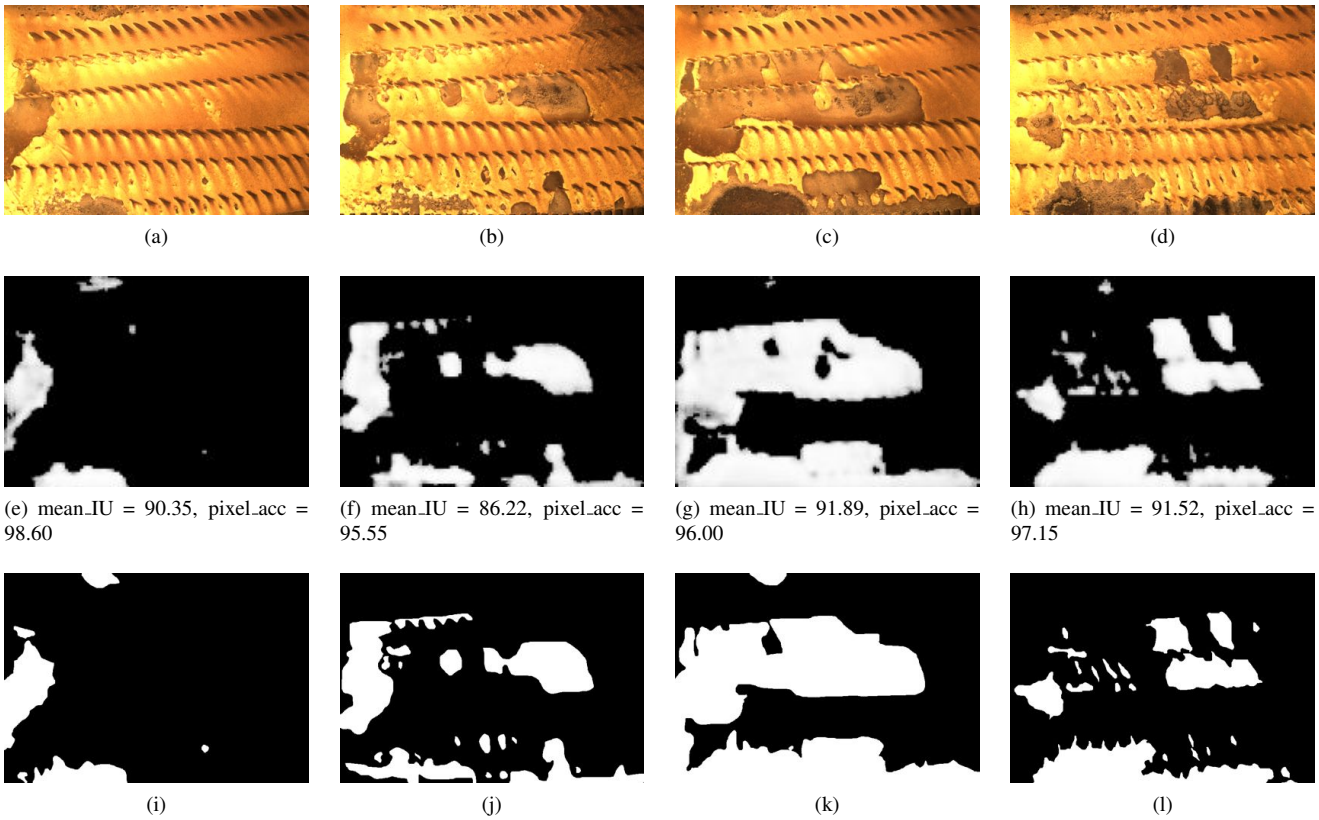
where conv denotes a convolutional layer, ReLU denotes a rectified linear unit, pooling denotes the max pooling layer and deconv denotes the deconvolution layer for upsampling.

The architecture of M-FCN is the combination of FCN_1 and FCN_1, as shown in Fig. 6. In particular, we concatenate the input vectors of logistic regression layers of either network, and use these new feature vector as the input to a new logistic regression layer in M-FCN.

Both FCN_1 and FCN_2 are trained using images of two engines's blades. The training dataset includes 128 original images of blades. We further augment the dataset by adjusting the pixel value by a random constant (between $\pm 10\%$ of each pixel value) to simulate possible change of lighting conditions and to alleviate the overfitting on the training data.

The training of M-FCN, as mentioned in Section 3.2, is essentially fine-tuning the model of FCN_1 and FCN_2 using the same dataset.

We use two metrics to evaluate the performance of our system: mean IU and pixel accuracy. These two metrics have been used in lots of prior arts to measure the performance of semantic segmentation [5, 9]. In particular, mean



(a) (b) (c) (d)

(e) mean_IU = 90.35, pixel_acc = 98.60 (f) mean_IU = 86.22, pixel_acc = 95.55 (g) mean_IU = 91.89, pixel_acc = 96.00 (h) mean_IU = 91.52, pixel_acc = 97.15

(i) (j) (k) (l)

Figure 7: Some segmentation examples

IU is defined as

$$mean\_IU = \frac{1}{C}\sum_i \frac{p_{ii}}{c_i + \sum_j p_{ji} - p_{ii}}, \tag{6}$$

and pixel accuracy as

$$pixel\_accuracy = \frac{\sum_i p_{ii}}{\sum_i c_i}, \tag{7}$$

where C is the number of classes of all pixels, c_i is the number of pixels in Class i and p_{ij} the number of pixels in Class i predicted to be in Class j.

We find that two properties of engine blade are particularly challenging for a successful segmentation: 1) The appearances of cooling holes can be quite similar to spallation at certain scales, and 2) the spallation area in a blade can be quite fractal, as shown in Fig. 4(b).

M-FCN addresses these two issues quite well, as shown in Fig. 5 and Fig. 7. The network can robustly recognize the cooling holes as areas without spallation. Features from small scales and large scales both contribute to the segmentation of fractal regions of spallation.

We show the overall performance of spallation segmentation on FCN_1, FCN_2 and M-FCN in Table 1. Multi-scale FCN has the highest performance in both mean_IU and pixel accuracy.

Network architectures	mean_IU (%)	pixel_acc (%)
FCN_1	85.04	96.78
FCN_2	87.93	96.87
M-FCN	**89.29**	**97.15**

Table 1: Spallation segmentation results for different network architectures

5.3. Comparison to manual annotation

The ground truth on which the results in 5.2 is based is obtained from manual annotation. Manual annotation of spallation area is not an easy task by itself, and warrants further investigation to validate the results obtained. Specifically, we are interested in the efficacy of manual annotation when compared to the multiscale FCN. In Fig. 5 and Fig. 7, we show a few examples of segmentation obtained from multiscale FCN and the corresponding annotations by human. While the results look very similar, it begs the question of how good the multiscale FCN is when compared to

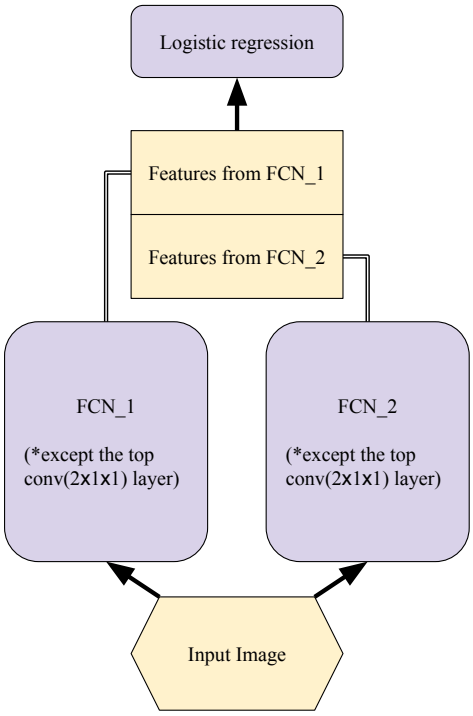


Figure 6: The architecture of M-FCN

	A1	A2	A3	M-FCN
A1 as ground truth	-	87.93	90.22	89.29
A2 as ground truth	94.80	-	92.66	92.60
A3 as ground truth	93.24	88.78	-	90.01

Table 2: The mean_IU of human annotators and M-FCN. A1, A2 and A3 are three human annotators.

human annotators?

To this end, we compare the performance of our system with three human annotators. In particular, we treat each annotator's result as "ground truth" and then evaluate the performance of the system and human annotators. The idea is that if the human annotators can do much better than the M-FCN system, their segmentation results should be consistent with each other and the M-FCN system will produce different results from the annotators. Otherwise, the system should perform similarly as human annotators.

We present the overall result in Table 2. The performance of M-FCN are very similar to human annotators. In particular, M-FCN gives higher performance than Annotator 2 when comparing to Annotator 1 and Annotator 3. This result further shows the potential of the M-FCN based system to replace human labors on the aircraft engine blade inspection task.

6. Conclusion

This paper presents a multiscale FCN that facilitates fine-tuning of a combined network. The proposal of how independently trained networks can be combined to enhance performance as more data becomes valuable is a key contribution towards a more flexible way to harness the power of deep learning. Results obtained from applying the multiscale FCN to segment and quantify the spallation of an aircraft engine blade appears to support the efficacy of the multiscale FCN strongly. Our expectations are that the multiscale FCN should be able to benefit the research community with our findings.

References

[1] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *Advances in neural information processing systems*, pages 2843–2851, 2012.

[2] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[3] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1915–1929, 2013.

[4] Y. Ganin and V. Lempitsky. Nˆ 4-fields: Neural network nearest neighbor fields for image transforms. In *Computer Vision–ACCV 2014*, pages 536–551. Springer, 2014.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 580–587. IEEE, 2014.

[6] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *Computer Vision–ECCV 2014*, pages 345–360. Springer, 2014.

[7] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[9] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. *CVPR (to appear)*, Nov. 2015.

[10] O. Matan, C. J. Burges, Y. Le Cun, and J. S. Denker. Multi-digit recognition using a space displacement neural network. 1995.

[11] F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P. E. Barbano. Toward automatic phenotyping of developing embryos from videos. *Image Processing, IEEE Transactions on*, 14(9):1360–1371, 2005.

[12] E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Computer Vision–ECCV 2006*, pages 490–503. Springer, 2006.

[13] P. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *Proceedings of The 31st International Conference on Machine Learning*, pages 82–90, 2014.

[14] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014.

[15] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems*, pages 1799–1807, 2014.

[16] R. Wolf and J. C. Platt. Postal address block location using a convolutional locator network. *Advances in Neural Information Processing Systems*, pages 745–745, 1994.

[17] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE, 2009.

[18] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014.