Supplements

Image annotation process

Patients were randomly divided into three groups and the images were sent to one of the three experts for analysis. Each expert annotated one group of the images independently and reviewed annotations from the other two experts. If a clear view of the brachial plexus was not observed in an image, the annotation would be blank, indicating there was no brachial plexus in the image. If an image contained only one root of the brachial plexus or reached the supraclavicular level, the image was discarded and another image was sampled from the same patient from the same side of the neck if such an additional image was available. Only images with annotations agreed by all three experts were included in the training dataset. Annotations with different opinions were discussed and revised by the three experts until they reached an agreement, otherwise these images were discarded.

Model development

A convolutional neural network was developed using the Tensorflow package (version 1.8.0) of Python (version 3.6.8) to automatically locate the interscalene brachial plexus in ultrasound images. An overview of the constructed network is

shown in Supplemental Figure 1. The network was based on the U-Net architecture with multiple modifications in an attempt to train the model more efficiently and to improve model performance. The network was trained and ran on an Intel Xeon CPU (E5-2678 V3) 12 cores×2.50GHz, 64GB RAM, and Nvidia GeForce RTX 2080Ti CUDA (Version 9.0.176).

Input images first underwent preprocessing, including cropping and transforming images into uniform size and resolution, padding, and gray histogram equalisation. Subsequently, the images were transformed into a set of feature maps by a convolution layer and passed through a pre-encoding residual neural network (ResNet) block to extract various features for subsequent layers.

The network was composed of 23 convolutional layers, which were organised as the visual cortex that each layer detected features at different levels, from local features, such as lines and dots, to more global features, such as shapes and contours. Corresponding de-convolutional layers at different levels were added after convolutional layers to combine feature maps back into an output image. Both the encoding and decoding paths consisted of four ResNet blocks dealing with features at different levels. Supplemental Figure 2 shows the architecture of a ResNet block. Each ResNet block included two 2D convolution layers with 3 × 3 filters and a stride of 1. Both convolution layers were followed by a concatenated rectified linear unit (Concat-ReLU) activation and batch normalization (BatchNorm) layer. An average pooling layer was applied over a 2 x 2 window

with a stride of 2 to the end of each ResNet block of the encoding path, except for the last one. Dropout with a random probability of 0.3 was utilised after each convolutional layer in the encoding path to avoid overfitting. Skip connections were added to limit the gradient degradation. The decoding path operated in the opposite direction of the encoding path. A concatenation layer with a feature map from the encoding path with the same dimensions was copied to the deconvolution layer, with 3 × 3 filters and a stride of 2 at the start of each ResNet block. A final 1×1 convolutional layer was applied to combine all feature maps into an output image. A detailed introduction of the operations used in the model is presented in Supplemental Table 1. Supplemental Figure 3 illustrates a typical example of the image processing flow from the input image to the output results. The model offers another type of output: a red dot with a diameter of 2 mm in the centre region of the predicted brachial plexus. The entire processing time for a single image was approximately 800 ms. The original code is available on ResearchGate (www.researchgate.net).

We calculated the width of the minimum bounding rectangle of the annotations for the entire training dataset, and found that the minimum width was 4 mm. Thus, the model was set not to output any segmentation on the original image if the width of the minimum bounding-rectangle of it <4 mm.

Supplementary Table 1 An introduction of the operations in the model

Operations	Function	Description
Convolution	Extract features from images	Extract the main features in the image to form a new feature map. Multiply the pixel value on the image point with the value on the corresponding convolution kernel, add all the multiplied values as the pixel value of the new feature map, and finally use the convolution kernel to slide all points of the image process.
Pooling	Compress images	Use a certain screening strategy to retain the main features while reducing redundant parameters and calculations, preventing over-fitting and improving the generalization ability of the model.

Down-sampling	Expand the receptive field	Through operations such as pooling, the main features of the image are retained and redundant information is removed while the size is reduced to prevent over-fitting.
Up-sampling	Enlarge the feature maps	Enlarge the compressed image and restore it to its original size.
Skipping connections	Alleviate the problem of vanishing gradient	Build a deeper network by skipping connections, make better use of the information of sub-feature maps, and perform gradient updates more evenly.
Relu	Improve model learning ability	A kind of piece-wise linear function. Add nonlinear factors to better fit nonlinear functions.

Batch Normalization	Highlight the relative differences of images and speed up the model learning rate	Normalise the feature maps of different data distribution ranges in different batches in the model learning process to reduce absolute differences, avoid problems such as gradient explosion, and speed up the learning rate.
Dropout	Alleviate over-fitting and increase training speed	Use the preset probability to randomly freeze some neuron nodes, reduce the model parameters that need to be updated during training, and reduce the dependence of the model on some neuron nodes to improve generalization ability.

Image augmentation

The extracted images were first denoised and adjusted to achieve optimal contrast, exposure and brightness. Afterward, the original images were flipped horizontally or vertically, rotated 5, 9, 13, 17, or 21 degrees to the left or right, scaled to 110%, 120%, 130%, 90%, 80%, or 70% of the original size, moved 10, 20, 30, 40, or 50 pixels to the left or right, 10, 20, or 30 pixels up, 10, 20, 30, 40, or 50 pixels down, 10, 20, or 30 pixels left and up, 10, 20, 30 pixels right and up, 10, 20, 30, 40, or 50 pixels left and down, or 10, 20, 30, 40, or 50 pixels right and down. Every method described above generated an augmentation of an original image, so 52 augmented images were generated for an original image.

Training and optimisation

An adaptive moment (Adam) optimiser was used to update the network weights and minimise the total loss. Soft Dice, a transform of the original Dice Similarity Coefficient (also known as F1 score) with the formula Soft Dice = 1 − Dice, was used as the loss function of the network during the training process. K-fold cross-validation (k=10) and the dropout technique were applied to improve the accuracy and minimise overfitting. All weights were initialised using a normal distribution with means of 0 and standard deviations of 0.01. All biases were initialised as 0. The network was trained with an initial learning rate of 0.00001, a batch size of 64, and a dropout probability of 0.3 between layers. We ran 10 epochs of tests for each stage with a maximum epoch number of 100. Basic

network optimisation was generally achieved by grid searching and manual tuning.

Evaluation measures

The intersection over union (IoU) and the F1 score evaluate the location agreement between two regions by calculating their overlap ratio. The intersection refers to the area of overlap between two regions, whereas the union describes the combined area of two regions. The formulas for IoU and F1 score are illustrated in Supplemental Figure 5. The values of IoU and F1 score range from 0% to 100% (0~1). A higher score indicates a better location agreement. (Supplemental Figure 6)

The average symmetric surface distance (ASSD) and Hausdorff distance are measures for evaluation of the distance between two region contours. A region contour is composed of a set of points. Let the distance from a point on the contour of one region to the nearest point on the contour of the other region be the shortest distance from the point to the contour of the other region. In this way, we can measure a set of shortest distances from all points on the contour of the predicted segmentation to the contour of the ground truth, as well as a set of shortest distances from all points on the contour of the ground truth to the contour of the predicted segmentation. ASSD is the average distance of the two sets of shortest distances, whereas the Hausdorff distance is the maximal

distance of the two sets (Supplemental Figure 7).

Supplemental Results

Heat maps overlaying 100 predictions or ground truth of images in the test dataset are presented in Supplementary Figure 8. Sensitivity analysis was done by varying the number of training samples, training epochs, and testing samples. (Supplemental Table 2 and 3)

Supplemental Table 2. Sensitivity Analysis 1

Model prediction results of randomly sampled images from the test dataset

Number of images (n)	Sensitivity (%)	Specificity (%)	Accuracy (%)
100	97.7	84.6	96
90	97.5	81.8	86
80	98.6	90.9	78
70	96.7	88.9	67

Supplemental Table 3. Sensitivity Analysis 2

Prediction results of the test dataset by models with different training conditions

Number of training images (n)	Number of training epochs (n)	Sensitivity (%)	Specificity (%)	Accuracy (%)
10000	138	97.7	84.6	96
9000	138	100	53.8	94
9000	103	98.9	30.8	90