COMPREHENSIVE STUDY OF CNN REGRESSORS

Tajwar Abrar Aleef, Marius Staring

Division of Image Processing (LKEB), Leiden University Medical Center
Leiden, Netherlands
E-mail : tajwar.aleef@gmail.com

Abstract— The use of deep convolutional networks is leading to many series of breakthroughs in the field of computer vision. The main motive of this work is to use convolutional neural networks to help in assisting the physician to cut down the time it takes in treatment planning during the process of cervical brachytherapy. Currently, manual delineation of the brachytherapy applicator is done slice by slice by expert radiologists. After the lengthy process of delineation, the transformation parameters are estimated. This process is time-consuming and the patient must remain still with the applicator inserted. The goal of this work is to automate this treatment planning process in order to cut down the time spent during treatment planning. The neural network models used in this paper are solely trained by using synthetically generated 2D data. Several sets of synthetic dataset were generated starting from elementary level to highly realistic data. Next, a set of experiments are conducted in this paper to conclude that it is possible to get back the transformation parameters directly from the MRI scans as the input to the convolutional neural network.

1. Introduction:

Brachytherapy is a kind of cancer treatment strategy where radioactive seeds or sources are placed directly near the tumour. This close proximity of the radiation source to the cancerous region allows treating the patient with a high concentrated radiation dose. During this treatment, thin catheters are placed in the tumour allowing High-Dose Rate (HDR) radiation to be delivered directly into the tumour region. Brachytherapy is used to treat a variety of types of cancers such as prostate cancer, breast cancer, lung cancer, cervical cancer, esophageal cancer, and rectum cancer. A special kind of non-metallic applicator is used in Leiden University Medical Center which uses two circular capsule-like plastic projectors connected through two heavily insulated pipes from which the radiation is delivered. For the treatment of cervical cancer, dose specification to the targeted region has evolved over time. Patients having more serious cases of cervical cancer generally requires brachytherapy even after external beam radiotherapy (EBRT) in order to increase the likelihood of survival [4]. From a survey conducted in 2014 by the American Brachytherapy Society, eighty percent of the physicians who responded said they use a high dose rate (HDR) for image-guided cervical brachytherapy [1]. Generally, MRI is accepted to be the gold standard imaging modality for image-guided cervical brachytherapy treatment because it provides the best visualization in terms of the tumor volume and the cervix [2][3]. When performing brachytherapy, the applicator is left inside the patient's body for a long period of time during which the doctor performs MRI and then goes to the treatment planning stage. In the treatment planning stage, they segment the cancerous region along with the applicator device from the MRI images. This delineation is done slice by slice which takes quite a while during which the patient must stay still in order to minimize dislocating the applicator's position from the time the MRI scan was taken. As it can be understood, this process is not comfortable at all for the patient. The main goal of this project is to minimize the time it takes to manually segment and find the position of the applicator with respect to the tumour region. The idea is to use deep learning to train a network that can learn to output the transformation parameters of the applicator with respect to the tumor region automatically. A more detailed explanation of the treatment planning that goes into brachytherapy starting from preparing the patient, performing an examination and the standard insertion procedures are given in [5].

This report presents an approach to use deep learning to automate the treatment planning process executed during brachytherapy. The following parts of the report are organized as follows. Section II explains the dataset generation process and the test bench used for training the neural networks. In section III, a discussion is made on the results from different experiments conducted in the paper. Section IV summarizes the results and presents conclusions.

2. Data Preparation & Test bench:

For the series of experiments conducted, a large set of synthetic data was generated for training and testing purposes. Since the geometric shape of the applicator used during brachytherapy is known, the cross-sectional view of the applicator was masked out from an MRI scan taken during one of the treatment sessions. Only 2D sagittal images were used to train the neural networks. One assumption that was taken was that the applicator is a rigid object, although in reality the applicator can be expected to flex. After masking out the shape of the applicator from the MRI scan, a binary image was generated. Zero padding was done on the image in order to form a square image positioning the applicator exactly in the center. After that, the whole image was resized to 64×64 pixels. Next, a combination of random values of rotations and X & Y axis translations were introduced to the resized image. The range of rotation was between -15 to 15 degrees and the range of x and y translations was in the range between -15 to 15 pixels. No scaling or shearing was applied to the applicator. The main goal of this work was hence to find back the rotation, X translation, and Y translation parameters. 10,000 purely binary sets of images were generated with random combinations of rotations and translations from the above-mentioned range. 8000 of those images were used for training the network while 1000 of the images were used as a validation set during training and the last 1000 were used for testing purposes.

The networks from the first three experiments were trained on an Inspiron machine with 16GB of ram and 4GB of NVidia GTX 960M graphics card. The last network was trained on a machine with 256GB ram and 12 GB of NVidia TitanX GPU.

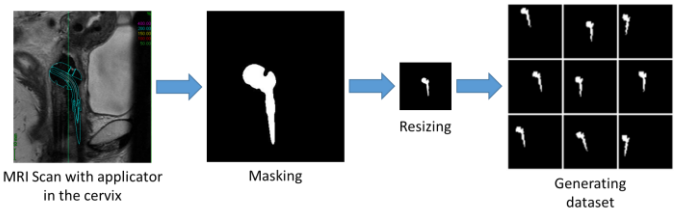

Figure 1: Dataset generation steps

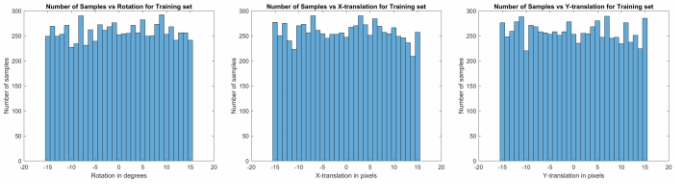

Figure 2: Histogram showing the homogeneity of the dataset in turns of different ranges of transformation parameters

Figure 2 shows the histogram of the number of samples against the range of each transformation parameter in the training set. From Figure 2, it can be observed that the number of data for every value of rotations and translations introduced in the training dataset was homogenous in nature. This means the network saw roughly an equal number of samples for every possible rotations and translation.

3. Experiments:

Experiment 1: Simple CNN architecture

Initially, to start with, the simple purely binary dataset was used to explore the efficacy of finding the transformation parameters from planar images. TensorLayer library, which is an extension of the Googles TensorFlow, was used on top of TensorFlow to design the convolutional neural networks. Starting off, a few different type of networks were explored including networks with just a few layers to networks with much more complex architecture. It was observed a simple network with fewer parameters learns to generalize the dataset better and provides robust predictions on the testing set. The final network of choice was a convolutional neural network containing 4 convolutional layers followed by 2 fully connected layers as shown in Figure 3. The convolutional layers all had a convolutional kernel of 3x3. As the problem at hand is a regression problem, no activation function was used on the final layer of the fully connected layer. The loss function used to optimize the network was Mean Squared Difference of the three output parameters. The network was optimized using Adam Optimizer with a learning rate of 0.0001, batch size of 200, and full training of 250 epochs.

Figure 3: CNN Regressor architecture

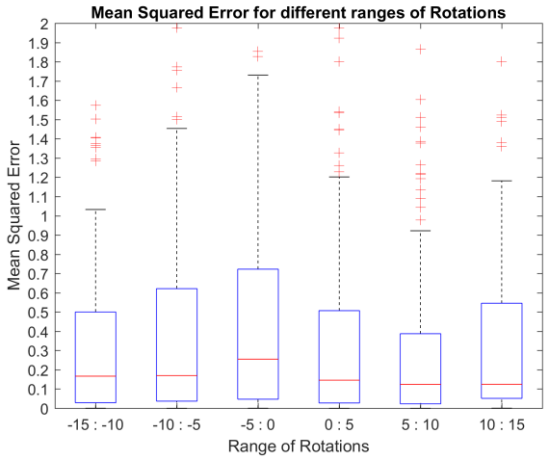

Figure 4: Quantitative evaluation for the Mean Squared Error at different range of Rotations in the testing set

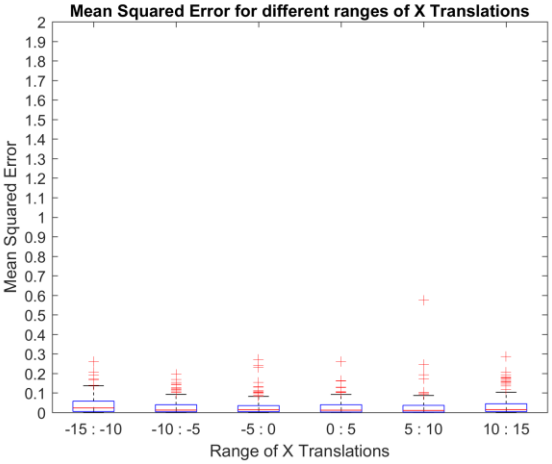

Figure 5: Quantitative evaluation for the Mean Squared Error at different range X-Translation in the testing set

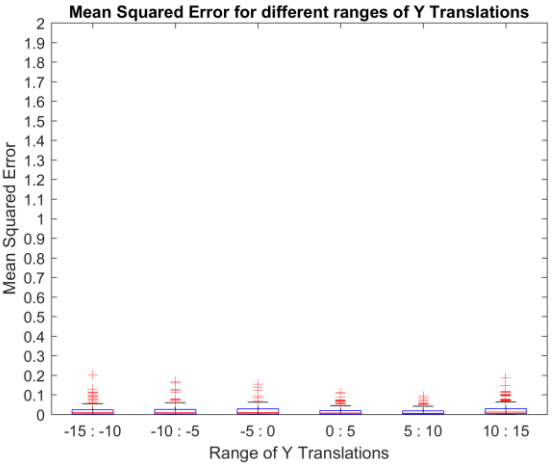

Figure 6: Quantitative evaluation for the Mean Squared Error at different range of Y-Translation in the testing set

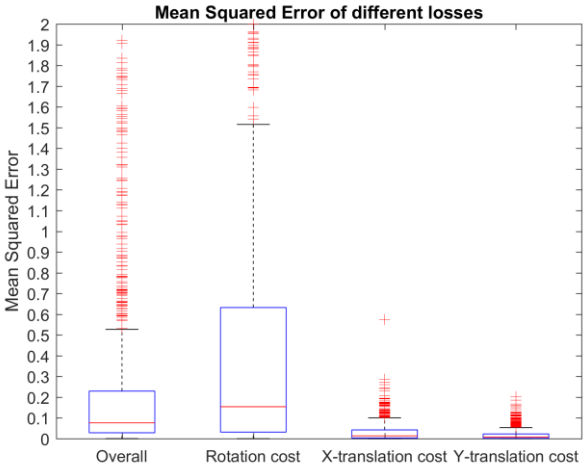

Figure 7: Quantitative evaluation of the Mean Squared Error for the overall cost functions compared to individual cost of Rotation and X &Y-translation in the testing set

After optimizing the network, the testing set was used to evaluate the performance of the network. The loss was recorded for all 1000 images from the testing set. Figure 7 gives the box plot of the overall and individual parametric loss for the testing set. From Figure 5 it can be seen that rotation parameter contributes the most to the overall MSE loss in the training set. It can also be observed that Y-translation contributes a little bit more to the overall loss than the X-translation loss. Overall, it can be concluded that the network learns the X-translation the best followed by the Y-translation and then the Rotation parameter. For a more in-depth understanding, the box plots of MSE for different ranges of rotations and translations were also plotted as shown in Figure 6, Figure 7, and Figure 8. It can be observed that the MSE of the translation parameters was homogenous over its entire range of translations and the median MSE is very low for both the translation parameters which is less than 0.1. The majority of the outliers are also seen to be well within 1. However, for the rotation, it's a different case. It can be observed that overall, the MSE is not homogenous over the range of the rotations in the testing set. It can also be seen that the network struggled more in predicting the rotations in the range of -5:0 degrees.

Figure 8: Box plot showing the Dice Index of the predicted images in the testing set

After getting the predicted parameters, the output images were generated and evaluated against the ground truth using dice index. The mean dice index was calculated over the testing set which yielded 0.9890. Overall, there were 715 images that had a dice index of 1 with the lowest dice index among the test set being 0.8427. Overall, out of 1000 images, 995 images had a dice index greater than 0.9. The box plot of the dice index of the testing set is shown in Figure 8.

Four images ranging from the best results to the worst were selected. From the testing set, image number 1 had the best dice index which was 1, and image number 850 had the worst dice index which was 0.8427. Two more images were chosen in between with dice index of 0.9778 and 0.9392. A comparison of the predicted and ground truth for the four cases was made which is given in Table 3. Figure 9 shows the pair images of the ground truth and the predicted output of the four selected images. From the figure, the false positives and the false negatives in the predicted image can be visualized. Table 1 and 2 gives the ground truth of the transformation parameters and their corresponding predictions for the four selected images. By comparing the two tables, it can be observed that the network was able to detect the transformation parameters with minimal error. Even for the worst case which is for image number 850, the rotation was off by only 1 degree and the X translation was off by only one pixel. The mean squared error for each individual loss is also given in Table 3 for comparison. Moreover, it can be concluded that the simple CNN network was able to learn to predict the transformation parameters quite robustly.

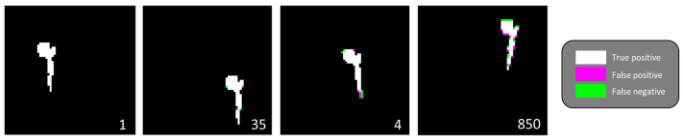

Figure 9: Some examples from the testing set ranging from the best result to the worst result in the set

Table 1: Ground Truth of few examples

Image Index	Rotation	Trans X	Trans Y
1	-6	-2	-12
35	-5	14	13
4	3	1	4
850	-14	-14	12

Table 2: Predicted output from the network

Image Index	Rotation	Trans X	Trans Y
1	-6.064	-2.171	-12.026
35	-5.515	14.101	12.919
4	1.4911	0.7322	4.011
850	-13.480	-13.490	12.027

Table 3: Dice Index and losses of the example images

Image Index	Dice Index	Rotation loss	Trans X loss	Trans Y loss	Mean MSD
1	1.00	0.0041	0.0291	0.000675	0.0113
35	0.9778	0.2653	0.0101	0.0065	0.0940
4	0.9392	2.2769	0.0717	0.0001161	0.7829
850	0.8427	0.2703	0.2599	0.00072	0.1770

Experiment 2: Inclusion of Noisy Dataset

Next, Gaussian noise was introduced in the dataset to reflect a more realistic scenario where there will be noise and background information of other organs in the MRI scan. Uniform Gaussian noise with standard deviation of 0.01, 0.05 and 0.1 was introduced in the purely binary dataset. Figure 10 shows the visual representation of introducing the three standard deviations of Gaussian noise in the dataset.

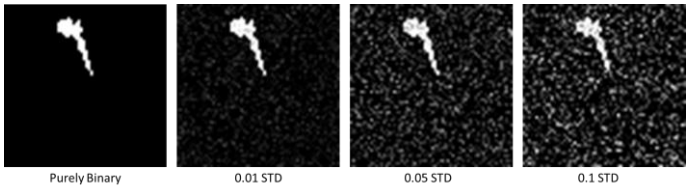

Figure 10: New dataset synthesized by adding different values of Gaussian noise

The same simple CNN network was again trained and optimized using the same hyperparameters with the same batch size of 200 and training epoch of 250. Table 4 gives the mean dice index and the mean plus the individual MSE of each cost function. It can be observed as expected that as the level of noise in the dataset increased, the dice index fell while the mean squared error started to increase. Even though the noisy data had little effect on predicting the transformation parameters, the effect of the noise on predicting the rotational parameter was very large as can be seen from the box plot of all the overall MSE for different levels of noise in the testing dataset from Figure 11. Figure 12 contains a box plot of the dice index of the different levels of noisy test set. This box plot also confirms the degradation of the dice index as the noise in the dataset is increased. From this experiment, it is clear that the simple CNN architecture loses its performance as more noisy images are used in the dataset.

Table 4: Dice Index and losses of the example images

Dataset	Dice index	Rotation MSE	Trans X MSE	Trans Y MSE	Mean MSD
Binary	0.9890	0.6698	0.0310	0.0179	0.2396
0.01 STD	0.9846	0.7144	0.0366	0.0318	0.2609
0.05 STD	0.9570	1.1317	0.0891	0.0645	0.4284
0.1 STD	0.9012	2.8437	0.2101	0.1225	1.0588

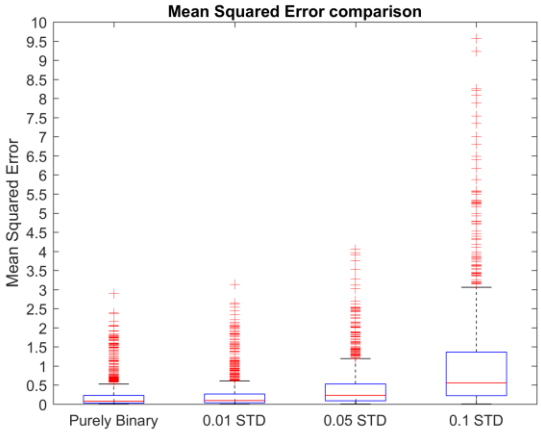

Figure 11: Box plot showing the Mean Squared Error for the overall MSE for different levels of noisy data

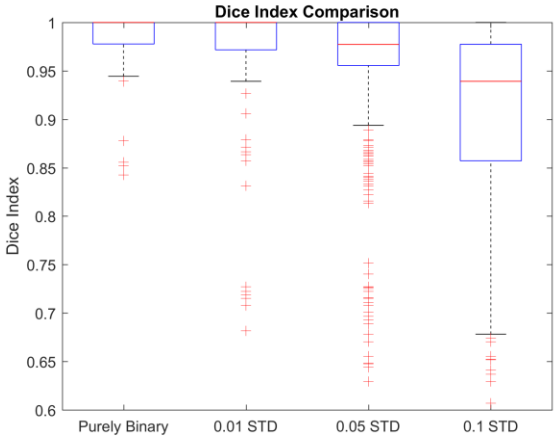

Figure 12: Box plot showing the Dice Index for different levels of noisy data

Experiment 3: Including Spatial Transformers

From the observed results, the higher the amount of noise introduced in the images, the difficult it becomes for the proposed network to predict the three transformation parameters. Even though the results of the noisy dataset can be further improved by optimizing the hyperparameters, a better approach would be including spatial transfer nodes along with the proposed network. This is because from the previous results it is clear that the network struggles to predict the rotational parameter more than the translation one. Spatial transformers networks (STN) which were designed to make CNN spatially invariant can come in handy as it can split the big rotations into small portions where the localization network of the STN learns a portion of the total transformation parameters and the fine-tuning network learns the rest of the transformation parameters [6]. The two can be added up to get the total transformation parameters from the input image. Initially, a single STN node was included with the proposed network as shown in Figure 13. For the next series of experiments, only the noisiest dataset with Gaussian standard deviation of 0.1 was used.

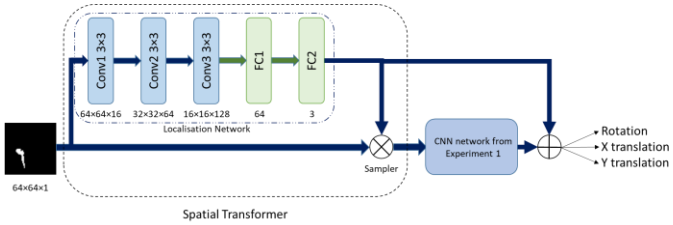

Figure 13: CNN regressor with one STN

Figure 13 shows the network including one spatial transformer network concatenated with the previously designed CNN network. The localization network used in the STN node consists of three layers of convolutional network with a kernel of 3x3 followed by two fully connected layers. The localization network outputs the three transformation parameters which are fed to the sampler which generates a grid and performs the predicted transformation. The transformed image is then fed as an input to the CNN network previously designed in experiment 1. The CNN network predicts and fine-tunes the rest of the remaining transformation parameters. The two results, one from the localization network and the final one from

the CNN network, are added to give the total transformations that were present in the original input image.

More than one spatial transformer nodes can be added in a pipeline to allow more special invariance in the neural network. The next sets of experiments were conducted on a network with 3 STNs in a pipeline. Simple addition was performed from the output of every localization network to get the overall transformation parameters instead of composing the parameters. Batch normalization which allows using higher learning rates were added before the activation function as just by adding Batch normalization, even the state-of-the-art classification models output a substantial speedup in training [7]. Following the practice in [8], no dropout [7] is used in the architecture of the networks. The architecture of the new network with 3 STN is shown in Figure 14.

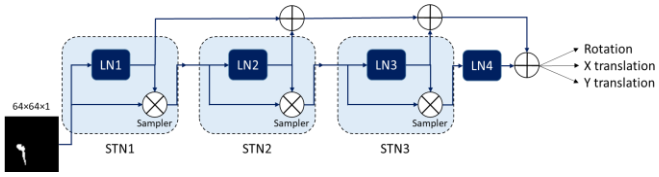

Figure 14: CNN regressor with three STN

Next, a new approach as taken from paper [9], was implemented to see how it performs on the same 0.1 standard deviation dataset. Inverse Compositional Spatial Transformer Networks (IC-STN) solve the problem of boundary conditions that can be observed from using STN in a pipeline. The boundary condition occurs because the output from an STN is fed to the next STN as an input. This means some of the boundary information gets lost as the image gets transformed during each corresponding STN node. In IC-STN, the transformation is always applied to the original input image. And during each corresponding IC-STN, the transformation parameters are added and then again applied to the original image again. This results in input with no loss of boundary condition. The same network as Figure 14 was implemented using IC-STN as shown in Figure 15.

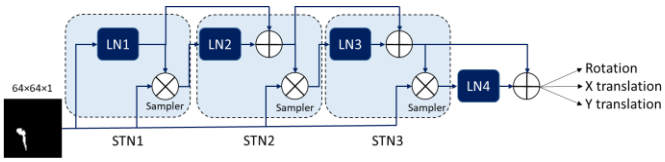

Figure 15: CNN Regressor with three IC-STN

The three new networks were trained using the noisiest dataset generated and a comparison was made to see which network performs the best in such a case. Only 150 epochs were needed to train these networks compared to 250 epochs for the simple CNN network. From Table 5, a comparison of all the networks is made in terms of the dice index, individual MSE, and overall mean MSE. It can be seen that using the 3 STN network yields the best dice index and the least error in the noisiest testing set.

Table 5: Comparison of dice index and both mean and individual losses for the networks trained with the noisy dataset

Dataset	Dice index	Rotation MSE	Trans X MSE	Trans Y MSE	Mean MSE
CNN	0.9012	2.8437	0.2101	0.1225	1.0588
STN	0.9590	1.4164	0.0947	0.0383	0.5165
3 STN	0.9734	1.0811	0.0613	0.0390	0.3938
3 IC-STN	0.9611	1.1237	0.0623	0.0775	0.4212

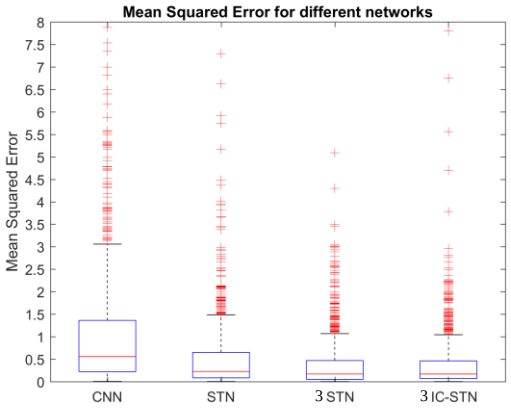

Figure 16: Box plot showing the Mean Squared Error for the overall MSE for the different networks with different STN

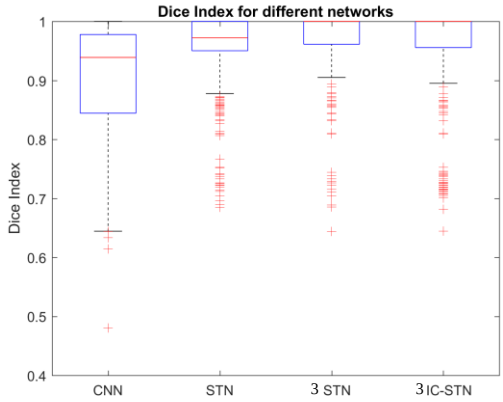

Figure 17: Box plot showing the mean dice index of the 0.1 STD noisy dataset for the different networks with different STN

Experiment 4: Realistic MRI dataset

From the previous experiments it was indeed clear that even with the noisiest synthetic data, all the networks were able to converge. For this part of the experiment, a more realistic dataset was used to further check the ability of a convolutional neural network with spatial transformers to predict transformation parameters. 192 cases of real brachytherapy treatment were obtained from Netherlands Cancer Institute (NKI) database. Several sagittal slices were extracted from these 192 cases where the applicator is not visible and then the applicator was synthetically placed on top of the MRI image with combinations of random rotations and translations. The data was augmented, and 150,000 images were generated; among which 120,000 images were used for training and the rest 15000 for validation and 15000 for testing. Figure 18 shows a few samples of the dataset generated with the synthetically placed applicator. The size of the new images in the dataset was 160×90 pixels.

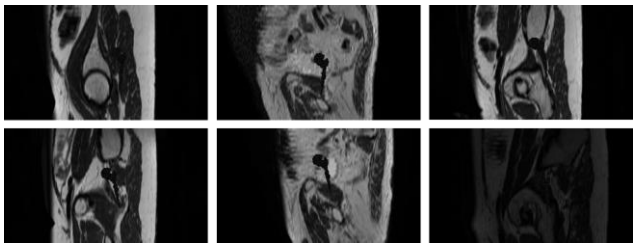

Figure 18: MRI dataset with synthetically inserted applicator

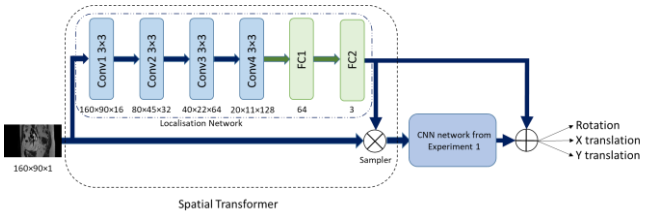

Figure 19: CNN regressor with a single STN

A higher number of training images were generated for this experiment as there was only a limited number of data available to work with. Augmenting the number of images provided better results as the network was able to learn to generalize the data better. STN was used instead of IC-STN as the network with three STN provided a better result in the last experiment. The localization network was again modified for this model but only one STN node was used between the localization network and the fine-tuning one. The architecture of the final model is given in Figure 19.

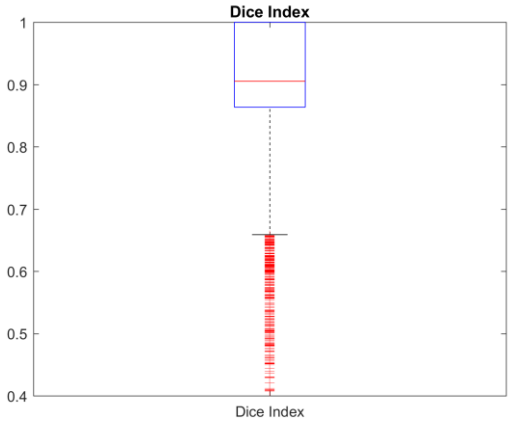

Figure 20: Box plot of Dice index for realistic MRI data

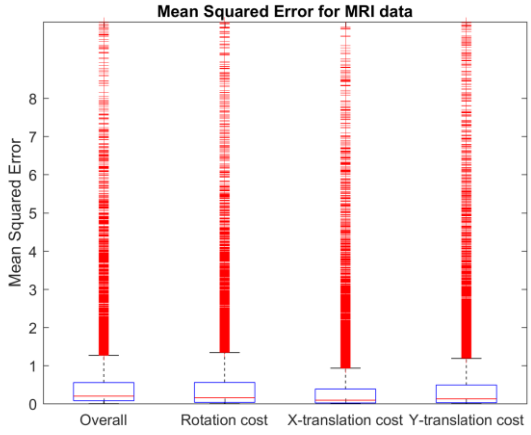

Figure 21: Box plot of MSE for realistic MRI data

The network was able to converge within 100 epochs. The output parameters were again used to generate back the 15,000 test images. The predicted output and the ground truth were evaluated against each other using dice index. From Figure 20, the box plot of the dice index on the testing set is shown. The mean dice index on the overall 15,000 images was 0.9027. The MSE of the overall cost and each individual cost was also calculated and shown as a box plot in Figure 21. Overall, it can be seen that a lot of outliers are present which contributes to an overall poor result. This is because as can be seen from Figure 18 that finding the applicator in the generated dataset is quite difficult. Of course, further tuning and preprocessing of the image can be added in the pipeline to generate better results.

4. Conclusions: In this paper, the use of deep learning to automate the treatment planning phase of cervical brachytherapy is explored. The main goal of this work was to find back the transformation parameters from MRI scans taken directly during brachytherapy. Starting with the generation of the synthetic dataset, 4 experiments were conducted. The point of the first experiment was to prove that a simple CNN can learn a regression problem of a simple dataset quite easily. The second experiment was about using the same network but now with different levels of noise introduced into the dataset. This experiment concluded that the noisier the data the worse the output result. Then in the third experiment, the main purpose was to improve the results of the noisiest dataset by introducing different combinations of STNs and IC-STNs into the networks. Three new networks were trained, and a comparison was made where it was seen that the network with 3 STNs had the best result on the noisiest testing set. Finally, the fourth experiment was all about using a more realistic MRI dataset with a synthetically inserted applicator. The final experiment yielded that the network was still able to output the transformation parameters quite efficiently even with the more challenging MRI dataset.

Future works that can be done are running more experimentation with the realistic MRI data such as adding more STNs to see if results can be improved further. Moreover, this work should extend to exploring the ability of the network to predict the 6 transformation parameters from a 3D scan directly instead of using 2D planar images.

6. References:

[1] Grover, Surbhi, et al. "Image guided cervical brachytherapy: 2014 survey of the American Brachytherapy Society." *International Journal of Radiation Oncology* Biology* Physics* 94.3 (2016): 598-604.

[2] Pötter, Richard, et al. "Clinical outcome of protocol based image (MRI) guided adaptive brachytherapy combined with 3D conformal radiotherapy with or without chemotherapy in patients with locally advanced cervical cancer." *Radiotherapy and Oncology* 100.1 (2011): 116-123.

[3] Tanderup, Kari, et al. "Magnetic resonance image guided brachytherapy." *Seminars in radiation oncology*. Vol. 24. No. 3. WB Saunders, 2014.

[4] Viswanathan, Akila N., et al. "Computed tomography versus magnetic resonance imaging-based contouring in cervical cancer brachytherapy: results of a prospective trial and preliminary guidelines for standardized contours." *International Journal of Radiation Oncology* Biology* Physics* 68.2 (2007): 491-498.

[5] Kirisits, Christian, et al. "Dose and volume parameters for MRI-based treatment planning in intracavitary brachytherapy for cervical cancer." *International Journal of Radiation Oncology* Biology* Physics* 62.3 (2005): 901-911.

[6] Jaderberg, Max, Karen Simonyan, and Andrew Zisserman. "Spatial transformer networks." *Advances in Neural Information Processing Systems*. 2015.

[7] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." *International Conference on Machine Learning*. 2015.

[8] Hinton, Geoffrey E., et al. "Improving neural networks by preventing co-adaptation of feature detectors." *arXiv preprint arXiv:1207.0580* (2012).

[9] Lin, Chen-Hsuan, and Simon Lucey. "Inverse Compositional Spatial Transformer Networks." *arXiv preprint arXiv: 1612.03897*(2016).