MR-based synthetic CT generation using a deep convolutional neural network method

Xiao Han

Elekta Inc., Maryland Heights, MO 63043, USA

(Received 18 October 2016; revised 31 January 2017; accepted for publication 5 February 2017; published 21 March 2017)

Purpose: Interests have been rapidly growing in the field of radiotherapy to replace CT with magnetic resonance imaging (MRI), due to superior soft tissue contrast offered by MRI and the desire to reduce unnecessary radiation dose. MR-only radiotherapy also simplifies clinical workflow and avoids uncertainties in aligning MR with CT. Methods, however, are needed to derive CT-equivalent representations, often known as synthetic CT (sCT), from patient MR images for dose calculation and DRR-based patient positioning. Synthetic CT estimation is also important for PET attenuation correction in hybrid PET-MR systems. We propose in this work a novel deep convolutional neural network (DCNN) method for sCT generation and evaluate its performance on a set of brain tumor patient images.

Methods: The proposed method builds upon recent developments of deep learning and convolutional neural networks in the computer vision literature. The proposed DCNN model has 27 convolutional layers interleaved with pooling and unpooling layers and 35 million free parameters, which can be trained to learn a direct end-to-end mapping from MR images to their corresponding CTs. Training such a large model on our limited data is made possible through the principle of transfer learning and by initializing model weights from a pretrained model. Eighteen brain tumor patients with both CT and T1-weighted MR images are used as experimental data and a sixfold cross-validation study is performed. Each sCT generated is compared against the real CT image of the same patient on a voxel-by-voxel basis. Comparison is also made with respect to an atlas-based approach that involves deformable atlas registration and patch-based atlas fusion.

Results: The proposed DCNN method produced a mean absolute error (MAE) below 85 HU for 13 of the 18 test subjects. The overall average MAE was 84.8 ± 17.3 HU for all subjects, which was found to be significantly better than the average MAE of 94.5 ± 17.8 HU for the atlas-based method. The DCNN method also provided significantly better accuracy when being evaluated using two other metrics: the mean squared error (188.6 ± 33.7 versus 198.3 ± 33.0) and the Pearson correlation coefficient (0.906 ± 0.03 versus 0.896 ± 0.03). Although training a DCNN model can be slow, training only need be done once. Applying a trained model to generate a complete sCT volume for each new patient MR image only took 9 s, which was much faster than the atlas-based approach.

Conclusions: A DCNN model method was developed, and shown to be able to produce highly accurate sCT estimations from conventional, single-sequence MR images in near real time. Quantitative results also showed that the proposed method competed favorably with an atlas-based method, in terms of both accuracy and computation speed at test time. Further validation on dose computation accuracy and on a larger patient cohort is warranted. Extensions of the method are also possible to further improve accuracy or to handle multi-sequence MR images. © 2017 American Association of Physicists in Medicine [https://doi.org/10.1002/mp.12155]

Key words: convolutional neural network, deep learning, MRI, radiation therapy, synthetic CT

1. INTRODUCTION

Computed Tomography (CT) imaging has been traditionally used as the primary source of image data in the planning process of external radiation therapy. CT images offer accurate representation of patient geometry, and CT values can be directly converted to electron densities for radiation dose calculation. On the other hand, CT images have limited soft tissue contrast and result in extra radiation to the patients.

In recent years, interests in replacing CT with magnetic resonance imaging (MRI) in the treatment planning process have grown rapidly.1,2 This is mainly because MRI is free of ionizing radiation and offers superior soft tissue contrast that allows more accurate target and structure delineation. A complete workflow based solely on MRI can further eliminate image registration uncertainties for combining MRI with CT and reduce clinical workload. The main challenges in replacing CT with MRI are that the MRI intensity values are not directly related to electron densities and conventional MRI sequences cannot obtain signal from bone. It is therefore desirable to have a method that can derive CT-equivalent information from MR images for the purposes of dose calculation during treatment planning and DRR-based patient setup. Such MR-based CT-equivalent data are often referred
to as pseudo-CT or synthetic CT (sCT). In addition to MR-only radiotherapy, generating accurate sCTs from MR images is also important for PET attenuation correction in hybrid PET/MRI systems.\(^3\) It should be noted that the inferior geometric fidelity of MRI can still be an issue when applying sCT for these applications, since the sCT generated will inherit any geometric distortion that exists in the source MR image.

Many different methods have been proposed in the literature for automatic generation of sCTs from MR images. They can be roughly divided into three categories: tissue-segmentation-based, learning-based, and atlas-based approaches. Tissue segmentation approaches first segment or classify MR image voxels into a discrete set of tissue types, such as air, fat, soft tissue, and bone, and then use bulk density assignments to assign a different CT number for each tissue type.\(^4\)–\(^11\) As pointed out in Hsu et al.,\(^8\) tissue segmentation is not an easy problem and one MR image volume is usually insufficient to separate all major tissue types. In addition, conventional MR sequences cannot reliably differentiate bone from air. Consequently, most tissue segmentation methods require the use of multiple MR sequences, especially the specialized ultrashort echo time (UTE) sequences. This, however, may lead to longer image acquisition time and more complicated workflow.

Existing learning-based approaches employ statistical learning or model fitting techniques to construct a mapping function that associates MR voxel intensities (or intensity patterns) with corresponding CT numbers.\(^12\)–\(^19\) Similar to the case of tissue segmentation, using conventional MR images alone is insufficient to make reliable predictions. Some methods require manual delineation of the bone volume so that separate models could be built for different regions.\(^13\),\(^14\) Alternatively, multiple sequences, especially UTE sequences, are often used to help distinguish bone voxels from air.\(^16\),\(^17\) In addition to MR intensity values, various local intensity pattern and coordinate-related features were also designed and used as extra inputs to improve the prediction accuracy of a sCT model.\(^15\),\(^17\),\(^18\)

Atlas-based approaches apply image registration to align a target MR image to an atlas MR, the correspondence information can then be used to warp the associated atlas CT image to the target MR to generate a sCT.\(^20\)–\(^33\) Such approaches have become very popular due to their potential of producing reliable sCT estimates using conventional MR images, as demonstrated in the literature.\(^20\)–\(^33\) Pure atlas-based methods require accurate deformable image registration of atlas and patient MR images. It, however, can be difficult to accurately register a patient image with an arbitrary atlas, especially when there are large anatomical variations or pathological differences. These difficulties can be partially overcome through the use of multiple atlases and the design of atlas fusion methods that are robust to atlas registration errors (such as patch-based fusion and sparse coding).\(^20\)–\(^22\),\(^24\),\(^29\),\(^30\),\(^32\) Hybrid methods have also been proposed that combine atlas-based methods with model-based prediction.\(^33\),\(^34\)

In this work, we propose a novel learning-based approach for sCT generation that exploits recent developments in deep learning and convolutional neural networks (CNNs). CNNs\(^35\)–\(^37\) are multi-layer, fully trainable models that can capture and represent complex, high-dimensional input-output relationships. They have become the method of choice in many fields of computer vision, and also been applied for computer-aided detection and medical image segmentation problems.\(^38\)–\(^41\) One appealing property of CNNs is their ability to learn complex models end-to-end instead of hand-crafting individual components (such as what image features to compute and how to map the features to the desired outputs).

We thus apply CNN to learn a direct image-to-image mapping between MR images and their corresponding CTs. This is different from all existing learning-based sCT methods, which require hand-designed features and predict CT values voxel-by-voxel. To our knowledge, this is the first time that a deep CNN method is developed for sCT generation. It should be noted, however, that a non-convolutional neural network method was proposed recently for cross-modality image synthesis.\(^42\) But unlike what is proposed here, the method was patch-based and need be applied in a sliding window fashion to generate image synthesis voxel-by-voxel. A small image patch (\(3 \times 3 \times 3\) in Nguyen et al.\(^42\)) cannot provide sufficient image context and also limits the model capacity. Hence, the authors\(^42\) found it necessary to incorporate spatial coordinates as extra features, which would require all images to be registered or spatially normalized.

Due to limited training data and computer memory restrictions, the CNN model in this study is designed for 2D, in the sense that it takes a 2D MR slice as input and outputs a corresponding 2D sCT slice. A 3D MR image can be processed slice-by-slice to generate a corresponding 3D sCT volume. This 2D procedure is still much more efficient than voxel-by-voxel predictions, since a typical 3D image can have millions of voxels but only a couple hundred slices at most. Our experiments showed that a full 2D slice already provides sufficient contextual information for the model to reliably separate locally similar voxels and produce very accurate sCT results. The method can, however, be extended in the future to take multiple slices as input or directly process a 3D volume once data and computer resources become available.

2. MATERIALS AND METHODS

2.A. Data acquisition and image preprocessing

The reported study was performed on real image data from 18 patients, which were randomly selected from a collection of patients having undergone stereotactic radiosurgery with the Leksell Gamma Knife®. The MR image for each patient was acquired with a 1.5 T Siemens Avanto scanner using a T1-weighted 3D spoiled gradient recalled echo sequence without contrast agent (echo time 4.6 ms, repetition time 11 ms, flip angle 20°, voxel size \(1 \times 1 \times 1\) mm\(^3\), field-of-view \(256 \times 256 \times 160\) mm\(^3\), total scan time 4 min). The corresponding CT image was acquired on a Siemens
Sensation 16 scanner with tube voltage 120 kV, exposure 300 mAs, in-plane resolution $0.5 \times 0.5 \text{ mm}^2$, and slice thickness of 1 mm.

Each patient's MR/CT image pair was aligned rigidly using a mutual information rigid registration algorithm, and the CT was resampled to match the resolution and field of view of the MR image. Each MR image was further corrected for intensity non-uniformity using the N3 bias field correction algorithm. All MR images were then histogram-matched to a randomly chosen template to help standardize image intensities across different patients using the method of Cox et al.

To better evaluate the accuracy of sCT estimation, a binary head mask was automatically derived from each MR image to separate the head region from the non-anatomical, background region of the image. This was achieved by applying the Otsu auto-thresholding method on each MR image. A morphological closing operator was employed to fill in gaps around the nasal cavities and the ear canals, the largest connected component of which then produced the head mask. Figs. 1(a) and 1(b) show one example MR image and the head mask computed.

It should be noted that every patient had a stereotactic head frame that was used in conjunction with Gamma Knife treatment. The head frame is totally invisible in the MR image but only present in the CT image. Its position also differs in each patient. It is infeasible to expect that a model based on the MR image alone can reliably predict the head frame. To avoid any adverse impact of the head frame on model training, we artificially removed the head frame from each CT image by setting all voxels outside the previously computed head mask region to a HU of -1000. Figs. 1(c) and 1(d) show axial views of a CT image before and after the head frame was removed. The head frame caused streaking artifacts in the CT image of every patient, as can be seen in these figures. It should be noted that these random artifacts in the “ground truth” CT images inevitably cause some over-estimation of the error when evaluating the accuracy of the predicted sCTs.

2.B. Deep CNN (DCNN) model for sCT estimation

As mentioned in the introduction section, we design a 2D DCNN model in this work to directly learn a mapping function to convert a 2D MR slice to its corresponding 2D CT. The model can be trained by collecting all 2D MR slices with corresponding 2D CT slices from each training subject’s 3D MR/CT pair. Once the model is trained, it can be applied on a new MR image slice-by-slice and the results can be assembled to get the final 3D sCT. Directly training a full-3D DCNN model is infeasible due to limitations in GPU memory of commodity GPU cards and due to limited training data in this study. It may also be unnecessary since a 2D slice already contains rich contextual information.

Many different CNN models have been proposed in the computer vision literature, and their architectures can be very flexible. In this work, we build upon recent developments in semantic image segmentation where a deep CNN model can be trained from end-to-end to directly produce a dense label map for object segmentation in a 2D image. In particular, we adopt and modify from the U-net architecture that was proposed in Ronneberger et al. and the resulting network architecture is shown in Fig. 2.
Similar to Ronneberger et al.\(^3\) (see also Noh et al.\(^4\) and Badrinarayanan et al.\(^4\)), the model can be seen as consisting of two main parts: an encoding part (left half) and a decoding part (right half). The encoding part behaves as traditional CNNs that learn to extract a hierarchy of increasingly complex features from an input MR image. The decoding part transforms the features and gradually reconstructs the sCT prediction from low to high resolution. The final output of the network is a 2D image with the same size as the input image. A key innovation from Ronneberger et al.\(^3\) that we borrow here is to introduce direct connections (shown as white arrows in Fig. 2) across the encoding part and the decoding part so that high resolution features from the first part can be used as extra inputs for the convolutional layers in the second part. This design makes it easier for the decoding part to generate high resolution predictions. These short-cuts also make the model more flexible. For example, the model can automatically learn to skip coarse level features (at bottom of the network) if high resolution features (at top of the network) are sufficient to produce accurate CT predictions.

Different from Ronneberger et al.,\(^3\) we design the encoding part to follow the same architecture as the popular VGG 16-layer net model\(^5\) that was proposed for image classification. We keep the first 13 convolution layers but remove the three-fully connected layers. Removing the fully connected layers reduces the number of parameters by 90% (from 134 million to 14.7 million), which makes the final model easier to train. Besides, the fully connected layers correspond to global image features that are critical for image classification tasks but not very relevant for dense pixel-wise prediction. By duplicating an existing architecture, we can initialize the feature extraction part of our model by copying existing VGG model weights that were trained on a very large set of nonmedical image data (i.e., 1.3 million natural images consisting of 1000 different object categories as explained in Simonyan et al.\(^4\)). Although the VGG model was trained on totally different images, low- to mid-level features learned from a CNN model are quite generic and initializing from a well-trained model is an effective way to help get better solutions for a new task with limited training data as we face here. In the next, we explain individual components of the network in Fig. 2 in more details.

The encoding part applies a series of convolutional and pooling layers to extract a hierarchy of features. The convolutional layer is the core building block of a CNN model. Each convolutional layer performs 2D convolutions of its input with a set of filters, the results of which are passed through a nonlinear activation function. Mathematically, the operation can be expressed as follows:

\[
h_k = \max(0, W_k \ast X + b_k), k \in [0, K - 1],
\]

where \(W_k\) and \(b_k\) represent the weights and bias of the \(k\)-th filter, respectively, and \(\ast\) denotes the convolution operation. The subscript \(k\), \(k \in [0, K - 1]\), denotes the index of the filter in the set of total \(K\) filters. \(X\) denotes the input, and \(h_k\) is the output of the \(k\)-th filter, which is also known as the \(k\)-th channel of the output feature map of the convolutional layer.

The max(0,\(\cdot\)) operation that is applied element-wise on the convolution output corresponds to the use of a Rectified Linear Unit (ReLU) as the nonlinear activation function. For the first convolutional layer, the input \(X\) is the MR image itself. For subsequent layers, the input comes from the output feature map of the previous layer.

The number of filters in each convolution layer is predetermined, whereas the weights and the biases \(W_k\)’s and \(b_k\)’s are the free parameters. A key to the success of CNNs is their ability to learn the weights and biases of individual feature maps to get data-driven, task-specific feature extractors instead of relying on a fixed set of hand-crafted features. Following the VGG model, we use 3 \(\times\) 3 filters for all convolutional layers in the encoding path. It should be noted that the size 3 \(\times\) 3 only refers to the spatial size of each convolution filter. Each weight is a vector that has the same number of components as the number of channels in the input. Suppose there are \(R\) input channels, the total number of free parameters in a convolution layer is thus equal to \(K \times 3 \times 3 \times R + K\). We apply zero-padding when performing the convolution so that the output size (spatial) stays the same as the input. The 2D size and the number of channels of the output feature map of each convolutional layer are shown in Fig. 2 on top of the blue boxes.

It should be clear from Fig. 2 that a lot of computer memory is needed to store the feature maps, since each feature map has tens or hundreds of channels and each channel has a size directly proportional to the input image size. The memory requirement is one limiting factor that prevents directly building a 3D DCNN model, especially since the number of filters or channels also needs to be increased to extract features in 3D. In addition, a 3D DCNN model will have more parameters and hence require more training data.

After performing 2 to 3 convolutional layers, a max-pooling operation layer is applied to reduce the spatial resolution of the feature maps so that consequent convolutional layers can learn features of larger image context. The pooling operation also helps to generate features that are invariant to local perturbations of input images. The VGG model uses max-pooling with a 2 \(\times\) 2 window and stride 2 (non-overlapping window). The operation consists of taking the maximum feature value over non-overlapping sub-windows of size 2 \(\times\) 2 within each feature channel. This can be formalized as follows:

\[
z_{k,ij} = \max_{p,q \in [0,1]} h_{k,2i+p,2j+q},
\]

where \((i, j)\) denotes the spatial index of the output feature map and \(k\) is the channel index. The max-pooling operation shrinks the size of the feature map by half in each spatial dimension. This is clearly seen in Fig. 2. It should be noted that there are no free parameters to learn for the max-pooling layer. But the locations where the maximum value is achieved in each pooling window need to be saved for later unpooling operation.\(^5\) This information can be stored as a binary mask with the same size as the input feature map, as illustrated in Fig. 3.
The decoding part of our DCNN model is chosen to be a mirrored version of the encoding part as similar to Ronneberger et al.\textsuperscript{39} and Noh et al.,\textsuperscript{47} with an extra $1 \times 1$ convolutional layer added in the end that is used to map each 64-component feature vector from the previous layer to a CT number. Contrary to the feature extraction part that uses max-pooling layers to gradually reduce the spatial resolution of the feature maps, the decoding part applies unpooling layers to propagate information from coarser to fine resolution. The unpooling layer can be considered as performing a reverse operation of the corresponding max-pooling layer. It doubles the spatial size of its input feature map, and copies the feature value directly from the input to the output at the maximal position for each pooling window using the recorded binary mask as illustrated in Fig. 3. The rest of the output is filled with zeros.

The output of an unpooling layer is very sparse. The subsequent convolutional layers then transform the sparse output into denser representations through convolution operations. Note that each of these convolutional layers in the decoding part has its own set of learnable parameters that are independent of the weights in the encoding part. As suggested in Ronneberger et al.,\textsuperscript{39} to make it easier to reconstruct image details, extra connections are made that copy over high resolution features from the encoding part and combine them with the unpooling output before feeding to the subsequent convolutional layers, as shown Fig. 2. Putting the encoding and decoding parts together, the complete network has 27 convolutional layers in total and about 34.9 million parameters.

The full network can be considered as representing a complex end-to-end mapping function that transforms an input MR image to its corresponding CT image. Learning the end-to-end mapping function requires the estimation of network parameters $\theta = \{W_1, b_1, W_2, b_2, \ldots\}$, which is achieved through minimizing a loss or prediction error between the predicted images $F(X; \theta)$ and the corresponding ground truth CT images $Y$. Given a set of MR images $\{X_i\}$ and their corresponding CT images $\{Y_i\}$, we use the Mean Absolute Error (MAE) as the loss function:

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} ||Y_i - F(X_i; \theta)||,$$

where $N$ is the number of training images (2D in this study). Using MAE ($l_1$-norm) as the loss function makes the learning more robust to outliers in the training data, such as noise or other artifacts in the images or due to imperfect matching between MR and CT images.

**2.C. Model implementation details**

The proposed pseudo-CT DCNN model is implemented using the publicly available Caffe package\textsuperscript{51} with a newly added input layer in order to handle 16-bit image data. The MAE loss function is also implemented. All computations are performed on GPU.

The DCNN model is trained using backpropagation\textsuperscript{52} with the Adam stochastic optimization method\textsuperscript{53} as implemented in Caffe, which offers faster convergence than standard stochastic gradient descent methods. The stochastic optimization method randomly selects a subset of training samples (each sample consists of a MR slice and the corresponding CT slice) at each iteration of the optimization, which is known as a mini-batch. Larger mini-batches are often preferred in order to reduce variances in gradient estimation and to fully leverage the parallel computing power of the GPU. But the mini-batch size is also limited by the available GPU memory. We use an effective mini-batch size of 48 for model training, as permitted by the GPU card used in this study. As mentioned earlier, the model parameters for the encoder part of the network are initialized using corresponding weights from a pretrained 16-layer VGG model as available from the Caffe ModelZoo repository.\textsuperscript{*} It is noted that the first layer of the VGG model expects 3-channel color images as input. We simply sum the VGG weights over the color channels to get the initial weights for our model for this layer. The weights in the decoding part of the network are initialized randomly with zero-mean Gaussian distribution $N(0, 1e^{-4})$,\textsuperscript{52} and the bias parameters are initialized to all zeros. To accelerate training, batch normalization is performed after each convolutional layer to reduce internal covariant shift.\textsuperscript{54} Simple data augmentation\textsuperscript{36} is also performed to artificially increase the number of training data during model training, which applies a random translation of up to 20 pixels in each spatial dimension for each pair of MR and CT images or randomly flips the images.

**2.D. Cross-validation of DCNN model**

The performance of the proposed DCNN method is evaluated using a sixfold cross-validation procedure. The 18 subjects are randomly divided into six equal-sized groups. At
each time, one group is retained as the test set, and the remaining five groups are used as training data to train a DCNN model. Once the model is trained, it is applied on each test subject’s MR image to generate the sCT.

The accuracy of the predicted sCT for each subject is evaluated against the real CT using a voxel-wise mean absolute error computed within the head region:

\[
MAE = \frac{1}{n} \sum_{i=1}^{n} |CT(i) - pCT(i)|,
\]

where \(n\) is the total number of voxels inside the head region of the MR. The head region is defined through a binary mask that is automatically derived from the MR image as described in Section 2.A. Similarly, the voxel-wise mean error (ME) and mean squared error (MSE) can also be calculated:

\[
ME = \frac{1}{N} \sum_{i=1}^{N} (CT(i) - pCT(i)),
\]

and

\[
MSE = \frac{1}{N} \sum_{i=1}^{N} |CT(i) - pCT(i)|^2.
\]

A fourth metric, the Pearson correlation coefficient is also computed between the CT and sCT for voxels within the head region.

For comparison purposes, we also compute sCT results using an atlas-based sCT generation method that we previously developed, which is also similar in principle to the method of Andreasen et al. The method uses the training data as atlases and computes deformable image registration to align each atlas MR to the test subject’s MR image. The registration transformation is applied to deform each atlas’ MR/CT pair to the test subject. At every voxel of the test subject’s MR, a local patch is extracted and compared against nearby patches in each warped atlas MR image. The top 10 most similar patches are found and the corresponding atlas CT numbers are then averaged using the patch similarity as weighting factors to get the predicted CT number at each test MR image voxel.

3. RESULTS

3.A. Computation time

Each MR volume has about 160 slices, so a group of 15 training subjects provide 2400 training samples. With the data being divided into mini-batches of size 48, it takes 50 iterations to go over all training samples once, which is known as one epoch in neural network training. Empirically, we found that it requires roughly 600 epochs or 30K iterations for the model to converge, which takes about 2.5 days of computation time using a single NVIDIA Titan X GPU with 3584 cores and 12 GB memory. Once the model is trained, it takes approximately 9 s to process all 160 slices of a new MR image to get the 3D sCT result. This leads to very efficient sCT generation at test time. In comparison, the atlas-based method took about 10 min on the same GPU.

3.B. Qualitative and quantitative evaluation of sCT images

Figures 4 and 5 show cross-sectional views of two representative sCT results (corresponding to subjects #5 and #12 in Fig. 6) as produced by the proposed DCNN method, along with the corresponding CT and MR images and the difference map between each sCT and the corresponding ground truth CT. Results from the atlas-based approach are also shown for comparison purposes. For most parts of the head region, the DCNN method predicts very accurate CT values as can be seen in the figures. Large errors mainly occur at interfaces between different tissue types, especially around the borders of bones and air. This is due to high intensity gradients at these areas, but it may also be partially due to non-perfect alignment between the MR and ground truth CT images for each patient, especially since a linear registration is unable to fully account for geometric distortions commonly exist in MR scans. Large errors are also present at the sinus and mastoid process regions where the CT images show fine bony details whereas the MR images are rather homogeneous. Comparing to the DCNN method, the atlas-based results tend to be noisier and have larger errors around skull and dura regions due to atlas registration difficulties.

The MAE values for all 18 subjects are plotted in Fig. 6, which compares the DCNN method with the atlas-based approach. It is seen that the DCNN method produced smaller MAE than the atlas-based approach for every subject. In particular, 13 of the 18 subjects had an MAE less than 85 HU for the proposed DCNN method, whereas only five for the atlas-based method.

The overall statistics of the four quantitative metrics computed over the 18 test subjects are summarized in Table I for the two different methods. Both the DCNN and the atlas-based methods gave unbiased sCT estimations, as evident from a mean ME close to zero. Although the mean ME of the DCNN results is slightly larger than that of the atlas-based results, the difference is not statistically significant with \(P \geq 0.05\). For all the other three metrics, the proposed DCNN was found to be better than the atlas-based approach, and the differences were also found to be statistically significant \((P < 0.05)\) based on paired \(t\)-tests.

4. DISCUSSION

A DCNN model is proposed in this work for synthetic CT generation using conventional, T1-weighted MR images. Although a deep neural network model typically requires a large amount of training data, very good performance is achieved with our limited data by making use of an existing, pretrained model through the principle of transfer learning. The DCNN method does not require any inter-subject image registration, either linear or deformable, and directly learns the mapping from the space of MR images to the
corresponding CT images. Evaluation results showed that the DCNN method offered significantly better accuracy than an atlas-based method with patch refinement and fusion. This result is expected since the atlas-based method relies on patch comparison to find similar atlas candidates, as also common in other atlas- or patch-based methods proposed in the literature. A small, local patch has limited image information and using raw image intensities of a patch as features may suffer from large redundancy in the data and reduce the discrimination power. On the contrary, the DCNN model automatically learns a hierarchy of image features at different scales and complexity from a full image slice.

**FIG. 4.** Qualitative comparison of sCTs and real CT for subject #5. The image type that each column represents is indicated at the bottom of the corresponding column. First column: MR; second column: sCTs (rows 1, 3, and 5 show the DCNN results, and rows 2, 4, and 6 show the atlas-based results); third column: real CT; fourth column: difference maps (rows 1, 3, and 5 correspond to the DCNN results, and rows 2, 4, and 6 correspond to the atlas-based results). The color bar is associated with the difference maps. First and second rows: axial slices; third and fourth rows: coronal slices; fifth and sixth rows: sagittal slices.
The proposed method produced an overall average MAE of 84.8 HU and an MSE of 188.6 when tested on data from 18 brain tumor patients, which compares favorably with other reported results in the literature for the head region. For example, the fuzzy $c$-means clustering method of Su et al. produced an MAE of 130 HU based on UTE MR sequences. Hofmann et al. reported an MAE of 100.7 HU for their hybrid method that combines pattern recognition and atlas registration. Uh et al. obtained an MSE of 207 using a similar approach as Hofmann et al. but with more atlases. Johansson et al. reported an MAE of 130 HU using a Gaussian mixture regression model. Another Gaussian
mixture classification method\textsuperscript{11} obtained an MAE of 147 HU. Gudur et al.\textsuperscript{34} achieved an MAE of 126 HU using a single atlas and Bayesian regression. The multi-atlas method of Burgos et al.\textsuperscript{23} produced an MAE of 102 HU. The atlas-regression method of Sjölund et al. yielded an MAE of 113.4 HU. The best MAE reported in the literature was from Andreasen et al.\textsuperscript{21}, who obtained an average MAE of 85 HU but only tested on five subjects. The particular approach\textsuperscript{21} consists of atlas registration followed by patch fusion, which is similar in principle to the atlas-based method that we compared in this study. Of course, these numbers are not directly comparable since the image data used in each work were all different.

A major advantage of the DCNN method is the fast computation time at model deployment. Although training a model can take days, the training only need be done once and acceleration is possible through the use of multiple GPUs. Applying the model to create sCT for each new patient only takes a few second on a single GPU. On the other hand, model-based or atlas-based methods can be rather slow. For example, the hybrid method of Uh et al.\textsuperscript{31} was reported to take 2 h when using six atlases and 4.5 h using 12 atlases, where the Gaussian process model fitting itself took over an hour when using 12 atlases. Andreasen et al.\textsuperscript{21} reported a computation time of 15 h for their atlas-based method with patch fusion. The authors later proposed an accelerated but approximate patch fusion procedure\textsuperscript{30} that reduced the computation time to 38 min. It was, however, acknowledged that the approximation would increase the MAE by 9 HU.\textsuperscript{21} Even without any atlas registration and with full GPU acceleration, another patch-based method of Torrado-Carvajal et al.\textsuperscript{30} still took 8.86 min when using 10 atlases.

A second major advantage of the DCNN method is that it can easily accommodate a large number of training data. In fact, it is well known that deep CNN models can greatly benefit from big data due to their high model capacity. We hence expect that the accuracy of the proposed DCNN method can further improve when more training data become available. The only burden from an increased number of training data is longer model training time, but the size of the final model and the test speed will remain the same. On the other hand, it may be cumbersome to increase training data for atlas-based methods or other model-based methods such as the Gaussian Process model, since they require all training data be kept

Table I. Overall statistics of four quality measures: mean absolute error (MAE), mean error (ME), mean squared error (MSE), and the Pearson correlation coefficient (CC). Average value and standard deviation ($\sigma$) are shown. The bottom row shows the $p$-values from paired $t$-tests of the differences between the two methods.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAE</th>
<th>ME</th>
<th>MSE</th>
<th>CC</th>
</tr>
</thead>
<tbody>
<tr>
<td>DCNN</td>
<td>84.8 ($\sigma = 17.3$)</td>
<td>$-3.1$ ($\sigma = 21.6$)</td>
<td>188.6 ($\sigma = 33.7$)</td>
<td>0.906 ($\sigma = 0.03$)</td>
</tr>
<tr>
<td>Atlas-based</td>
<td>94.5 ($\sigma = 17.8$)</td>
<td>$-1.5$ ($\sigma = 24.2$)</td>
<td>198.3 ($\sigma = 33.0$)</td>
<td>0.896 ($\sigma = 0.03$)</td>
</tr>
<tr>
<td>$p$-value</td>
<td>$4.7 \times 10^{-7}$</td>
<td>0.54</td>
<td>$7.0 \times 10^{-4}$</td>
<td>0.0013</td>
</tr>
</tbody>
</table>

Fig. 6. The MAEs computed within the head region for all 18 subjects, comparing two different approaches.
around all the time. In addition, the computation time of these methods is often directly proportional to the number of atlases used. It is also observed that the accuracy of atlas-based methods can quickly saturate and some atlas selection procedure is often needed to avoid degradation in accuracy when the number of atlases is large.20,21,31

Similar to other atlas- or model- based methods20, accurate intra-patient alignment of MR and CT images is an important prerequisite and can become a bottleneck for developing accurate DCNN models. Misalignments in training data cause inaccuracy in the model since the model will be trained to make wrong predictions. Inaccurately aligned ground truth MR and CT also negatively influence the validation study. We noticed that there were often some residue deformations in each MR/CT pair in our data after the initial rigid alignment, which might be due to both uncorrected geometric distortions in the MR images and inaccuracy in rigid registration. When the CT and MR images are acquired days or weeks apart, there can also be real anatomical changes taking place. This was the case for subject #17, whose sCT had the largest MAE, as illustrated in Fig. 7. Correcting the residue deformation is not a simple task since it is difficult to achieve sub-voxel accuracy everywhere for inter-modality deformable image registration. One potential approach to improve MR/CT alignment of the training data is to employ the sCT generated as a substitute for the MR so as to convert the inter-modality registration problem into a CT-to-CT registration one, for which many more registration methods are available. A trained DCNN model is usually only robust to data variations covered in the training data. Thus, a DCNN model trained using data from one scanner may not be directly applicable to data from a different scanner or a different imaging sequence. To address this issue, one can apply data preprocessing such as the histogram-matching in Section 2.A to help standardize MR image intensities before training and applying a DCNN model. Alternatively, one can simulate contrast changes as a part of data augmentation during the model training if a reliable contrast simulation procedure can be designed. When multi-sequence MR images are available and if it is possible to estimate MR tissue parameters (e.g., T1, T2, and PD) from these sequences, one can also train a DCNN sCT model based on MR parameter maps instead of the raw MR images. All these topics will be part of our future work.

Finally, note that unlike CT, MR image intensity and image contrast can vary significantly across field strengths and scanner types, even if the same imaging sequence is used. A trained DCNN model is usually only robust to data variations covered in the training data. Thus, a DCNN model trained using data from one scanner may not be directly applicable to data from a different scanner or a different imaging sequence. To address this issue, one can apply data preprocessing such as the histogram-matching in Section 2.A to help standardize MR image intensities before training and applying a DCNN model. Alternatively, one can simulate contrast changes as a part of data augmentation during the model training if a reliable contrast simulation procedure can be designed. When multi-sequence MR images are available and if it is possible to estimate MR tissue parameters (e.g., T1, T2, and PD) from these sequences, one can also train a DCNN sCT model based on MR parameter maps instead of the raw MR images. All these topics will be part of our future work.

5. CONCLUSION

In this work, we developed a deep CNN method for synthetic CT generation from conventional, single-sequence MR images. Experimental results showed that the method produced accurate synthetic CT results in near real time and competed very favorably with other state-of-the-art atlas-based approaches. It can also easily handle large training data without sacrificing test speed. It is thus a very promising
method that can facilitate MRI-only radiation therapy. Future work will evaluate the method on larger data sets and other anatomical regions. Extensions of the method to further improve accuracy and handle multi-sequence MR images are also warranted.

ACKNOWLEDGMENTS

The author would like to thank Dr. Jens Sjölund for providing the experimental data used in this work.

CONFLICTS OF INTEREST

The author is a full-time employee of Elekta Inc.

a)Author to whom correspondence should be addressed. Electronic mail: xiao.han@elekta.com.

REFERENCE
