Convolutional Neural Networks for Image Registration of Head and Neck Cancer Patients

Dónal McSweeney

9654610

School of Physics and Astronomy

University of Manchester

MPhys Report

January 2019

This project was performed in collaboration with *Thomas Heys*.

Abstract

Although current image registration algorithms are accurate, the computational cost makes them inadequate for large datasets. We seek to implement a convolutional neural network to perform fast and accurate supervised deformable image registration of head and neck cancer patients. In this paper, we discuss our efforts to register PET-CT images onto Planning CT images. Currently, our results seem promising. Our network performs registration of volume pairs in (5.76 ± 0.05) seconds, nearly 30x faster than current methods. However, the accuracy of our results is poor compared to those produced by other means. Future work will seek to solve this issue as well as developing an unsupervised alternative.

1 Introduction

Cancer is still one of the leading causes of death internationally. According to the International Agency for Cancer Research (IARC) an estimated 9.6 million deaths worldwide were attributed to cancer with 18.1 million new cases reported in 2018 [1]. However, over the last 15 years, the cancer death rate has declined by 1.5% annually [2]. This progress can be attributed to a reduction in smoking and advances in detection and treatment methods. Further research is vital to reducing the cancer death rate worldwide.

Due to the anatomy of the region, head and neck cancers can greatly affect a patient's quality of life. These cancers can greatly hinder respiration, ingestion of food, salivation and speech [3]. It is therefore extremely important to carry out precise treatment. Common cancer treatments methods include radiation therapy, chemotherapy, immunotherapy, hormonal therapy and surgery [4]. Treatment methods are often combined for optimal results. Indeed, alongside surgery, radiation therapy can be used to shrink the tumour and facilitate its removal or it can be used to eliminate unremoved malignancies post-surgery [4]. Here, we will discuss radiation therapy for head and neck cancer patients.

Radiotherapy is one of the most common methods of cancer treatment, with 50% of all cancer patients receiving radiation therapy at some point during the course of their treatment [4] [5]. Over the last century, great advancements have been made in the field. From improved dose delivery methods to a better understanding of cancer biology, radiation therapy is now responsible for 40% of all curative cancer treatments [4]. Due to the targeted nature of radiotherapy, it is important to accurately determine the target site. To do so, it is useful to image the region using different modalities. Image registration is central to combining the information produced by different methods.

Traditional image registration algorithms are slow and computationally expensive. They are also task specific and applying them in different contexts requires manual intervention and parameter tweaking [6]. An alternative approach seeks to implement deep neural networks, specifically convolutional neural networks (CNN). These types of networks were first applied to cancer research in 1996 by Sahiner et al [7] to perform classification of breast cancers. However, convolutional neural networks were not applied to deformable image registration until 2013. Wu et al [8] were able to perform fast and accurate deformable image registration of magnetic resonance brain images.

Deformable image registration produces a dense displacement vector field (DVF) containing information on the displacement of each pixel in three dimensions. However, most machine learning based image registration algorithms are unsupervised and thus operate by minimising a specified similarity metric [9]. Our goal is to produce a supervised convolutional neural network capable of performing deformable image registration of 3D volumes. We aim to train the network on PET-CT and planning CT volume pairs. This will be achieved by minimising the difference between the output DVFs from the network and DVFs generated by traditional means.

In this report, we will discuss radiation oncology and describe the pre-treatment process under-

gone by a head and neck cancer patient. Then, we will introduce the field of image registration and describe deep learning alternatives to traditional methods. This will lead into a description of our work and presentation of our current results. Finally, we will explore future plans for our project.

2 Theory

2.1 Radiotherapy

Radiation therapy involves delivering ionizing radiation to the tumour in an attempt to kill cancer cells. By delivering high-energy radiation to the target, DNA-damage can occur resulting in cell death or the inability of the cell to proliferate [10]. Many types of radiation have been applied and can be classified as either electromagnetic (EM) or particulate [11]. Examples of EM radiation are x-ray beams and gamma ray beams. On the other hand, electron beams, proton beams and neutron beams are considered particulate radiation [4]. This distinction is important as both types interact with cells differently. Here, we will discuss the processes by which EM radiation lead to cell death as it is the most commonly used form of radiation. EM radiation ionizes the target through the photoelectric effect, Compton effect or pair production [11].

The photoelectric effect involves the interaction of an incident photon with a bound electron. In doing so, the photon transfers *all* its energy to the electron causing it to escape its orbital and begin to ionize surrounding molecules [12]. This interaction is dependent on the energy of the photon and the atomic number of the atom. A lower energy photon and higher atomic number increases the probability of the photoelectric effect occurring. In tissue, the photoelectric effect dominates at photon energies in the 10-25 keV range [12].

The Compton effect involves the interaction of a photon with a free electron. Unlike the photoelectric effect, both the photon and electron are scattered. Thus the photon can undergo further interactions (at a lower energy) whilst the electron can ionize surrounding molecules [12]. The cross section for this process is inversely proportional to the energy of the incident photon and independent of atomic number. The Compton effect is the most common interaction in radiation therapy as treatment often involves photon energies in the 6-20 MeV range, the region where this process dominates [12].

Pair production involves the interaction of the incident photon with an atomic nucleus. The energy transferred leads to the creation of an electron-positron pair ($e^- + e^+$). The positron ionizes surrounding molecules until it recombines with a free electron, emitting two photons in opposite directions [12]. The probability of this interaction is proportional to the photon energy and is dependent on the atomic number. This process is only observed clinically when using high-energy beams ($\geq$25 MeV).

It is important to note that radiation damage is not limited to cancer cells. Therefore, an important aspect of radiation therapy involves developing an understanding the optimal way of delivering radiation to the tumour whilst reducing the dose delivered to healthy tissue. Cancer

cells however, tend to be less effective at repairing radiation damage than healthy cells, a process which facilitates selective treatment [5]. The radiation therapy procedure involves numerous steps to optimise tumour irradiation whist limiting exposure of healthy tissue.

2.2 Radiotherapy Procedure

Certain imaging procedures must take place prior to treatment. Firstly, a positron emission tomography-computed tomography (PET-CT) scan must be taken of the patient. This method combines a positron emission tomography (PET) scanner and an x-ray computed tomograpy (CT) [13].

PET scans excel at imaging metabolic processes. The PET procedure involves the injection of radiolabelled molecular probes (tracers) into the patient which become localized in the malignancy. The tracers used are specific to the process wanting to be monitored, 2-^{18}F-fluoro-2-deoxy-D-glucose (FDG) accumulates in cancer cells due to their altered metabolism and is therefore of great use in cancer imaging [14]. Three dimensional images can be reconstructed showing the location and concentration of the tracers within the patient, providing vital information about the tumour [14]. However, PET scans alone return poor representations of healthy anatomical structures.

In the early 1990's, Townsend et al [13] proposed combining PET with CT. The CT scan makes use of a number of images, commonly x-rays, taken from different angles and combines them into slices of the patient's anatomy [15]. In doing so, the metabolic processes imaged by PET can be more accurately aligned with the anatomic imaging obtained by CT [13]. PET-CT is used to monitor the spread of cancer cells and improve understanding of the tumour's location.

Subsequently, a planning CT scan must be performed in order to image the tumour alongside the surrounding organs whilst recreating the patient's position during treatment [12]. The planning CT allows the radiation oncologist to delimit the tumour and determine the most effective method of delivering the dose to the target site whilst minimising the damage to healthy tissue. Due to the localised nature of radiation therapy, target volume definition is an important procedure. An example is shown in Fig 1. The patient target volume (PTV) accounts for patient motion and variations in the size of internal organs during treatment. It is used to select an appropriate treatment plan to ensure that an adequate radiation dose is delivered to the clinical target volume (CTV).

It is important to immobilize the patient during this procedure as it allows for more accurate replication of the position the patient will be in during treatment [17]. It is especially important for head and neck cancer patients as many sensitive organs are located in close proximity to one another. For example, damage to the spinal cord could lead to huge long term complications for the patient. In the case of the planning CT, the patient is immobilized for approximately 15 minutes through the use of a perspex shell fixed in at least five places [17]. An example of this is shown in Fig 2. However, PET-CT is a longer procedure. Therefore, immobilising the patient

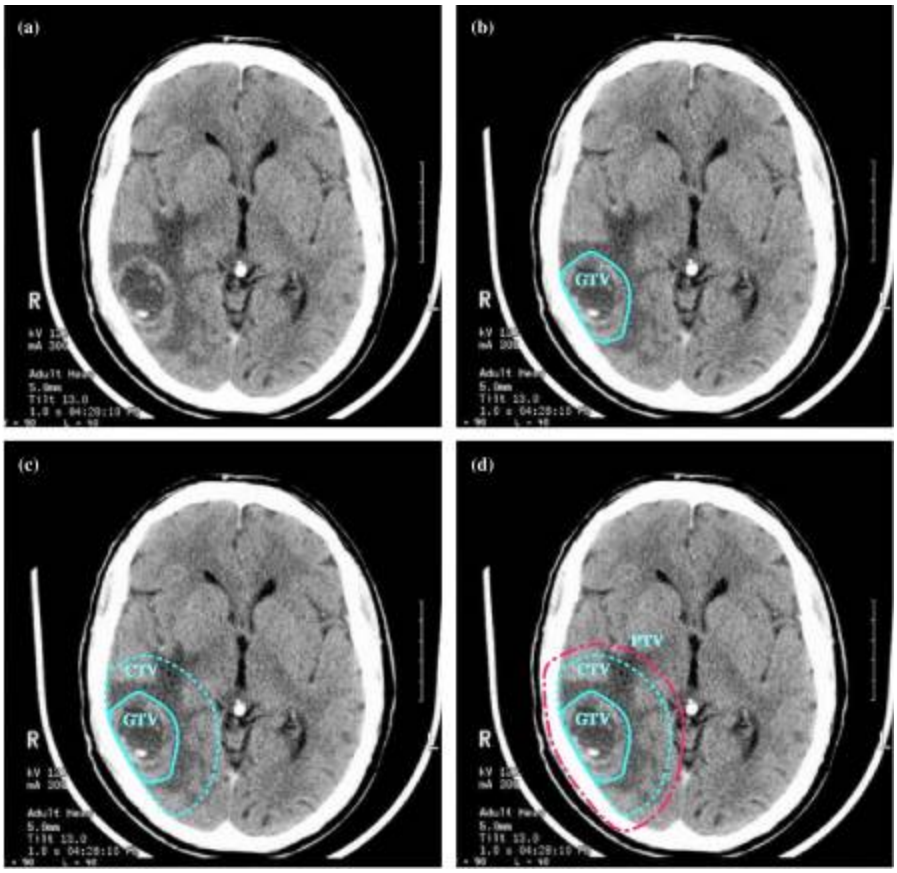

Figure 1: Target volume definition of a brain cancer patient. (a) Planning CT displaying contrast-enhancing tumour. (b) Gross tumour volume (GTV) outlines the visible tumour. (c) Clinical target volume (CTV) accounts for microscopic spread, uniform in all directions. (d) Planning target volume (PTV) accounts for uncertainties in planning and execution of treatment [16].

Figure 2: Perspex shell used to immobilise a head and neck cancer patient [17].

for up to an hour is inappropriate. A head rest is used instead to improve the patient's comfort. It is often advantageous to combine image modalities to extract further information. Due to the more accurate tumour imaging capabilities provided by the PET-CT, image registration is beneficial. By mapping the PET-CT scan onto the Planning CT scan, clinicians can improve the treatment plan by improving the accuracy of the PTV.

2.3 Image Registration

Image registration is the process of mapping one image onto another. It has been one of the main challenges in modern image analysis. Image registration seeks to determine the transformation $T(x')$ which minimises the difference between a fixed image, $F(x)$, and a moving image, $M(x')$. Formally, this is expressed as follows:

$$F(x) = M(T(x')) = M(x' + u(x')) \tag{1}$$

where $u(x')$ is the displacement vector applied to the moving image [18]. A number of image registration algorithms exist, with these three components in common: a transform, a similarity function (metric) and an optimisation method [19]. The registration process is shown schematically in Fig 3. Instead of applying the transformation to every pixel of the moving image, modern algorithms will apply the transformation to a sparse field of points to decrease registration time. Therefore, to calculate an accurate metric, an interpolator must be used to populate the image pixels from the transformed points [20]. Image registration can be classified into one of two categories: rigid and non-rigid.

Both have clinical applications but are applied in different contexts. Rigid registrations involve global transformations of an image such as rotations, translations and scaling. In this case, the pixel-to-pixel relationship remains constant throughout the process [18]. For example, rigid

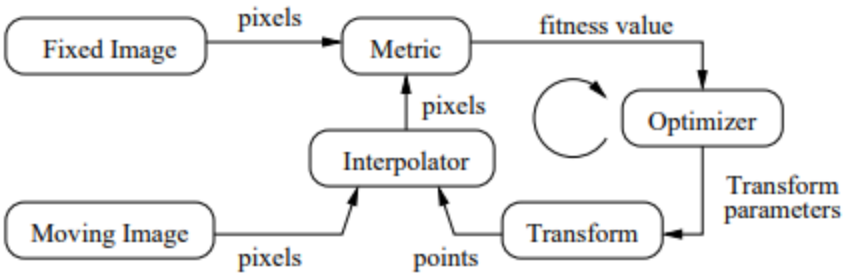

Figure 3: Basic components of an image registration algorithm. Two images are used as inputs, a transformation is applied to the moving image. A metric quantifies the similarity between the transformed image and the fixed image. The optimizer seeks to infer the optimal transformation based on the metric [19] [20].

registration is not the appropriate method for performing the task displayed in Fig 4 as image (B) can not be mapped onto image (A) through the application of global transformations as it requires a local transformation of the corners. However, non-rigid registration, known as deformable image registration can be applied in this context.

Deformable registration is described by a densely populated deformation vector field [19]. In the case of 3D deformable registration, this is a matrix where each element is a 3D vector describing the transformation applied to that corresponding pixel. As we aim to register PET-CT data onto Planning CT data, non-rigid registration is the most suited to our needs. This is mainly due to the capture method of the scans. Indeed, the resting position of the patient is different in each imaging case affecting the relative position of the internal organs. This aspect is clearly visible in Fig 5. However, due to different reference positions, a rigid transformation is also needed to initially align the images.

Although modern algorithms can be extremely accurate if they have been parametrised appropriately, they remain slow and computationally expensive [6]. Further, these algorithms are task specific and require manual adjustments to be applied to other tasks. Eppenhof et al [6] propose an alternative approach, which implements aspects of machine learning to perform deformable image registration.

2.4 Machine Learning

Machine learning is widely used in modern data analysis as it is extremely effective at analysing large datasets extremely quickly. This is achieved by training a neural network, through the use of input-output pairs; a form of machine learning known as supervised learning. In its simplest form, a neural network is a system built from layers of nodes each connected by links. A link from node i to node j, propagates the activation a_i from i to j. Associated with each link is a weight, w_{ij}. Each node j computes an output, a_j by applying an activation function, g to the

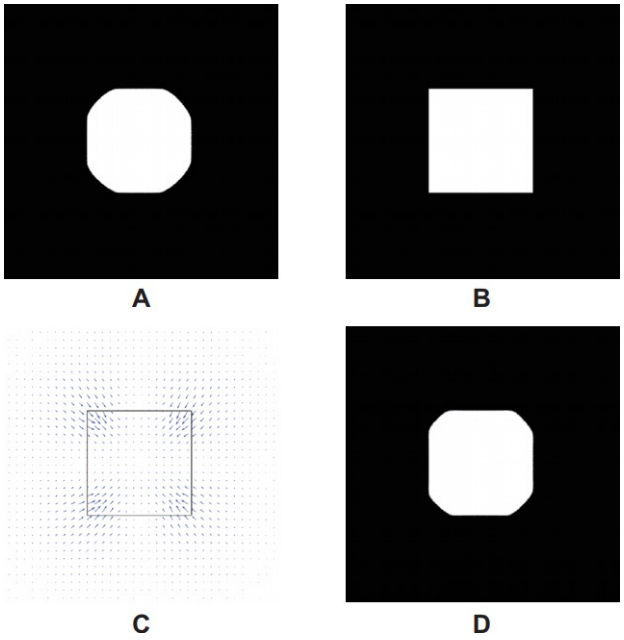

Figure 4: 2D non-rigid image registration process. (A) Fixed image, the image which will be mapped onto. (B) The moving image, the image which will be deformed. (C) Deformation vector field (DVF) displayed as blue arrows. The direction and size of the arrows represents the direction and magnitude of the deformation. (D) The resulting deformed image [18].

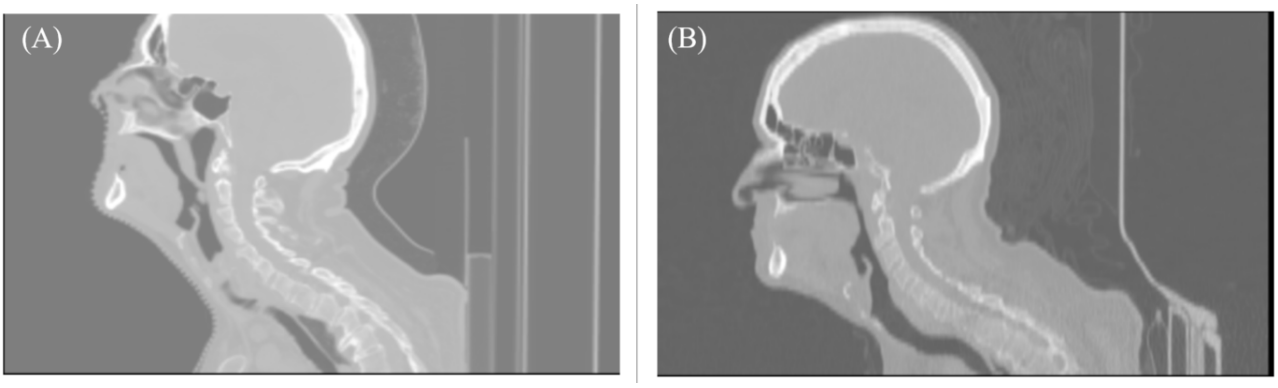

Figure 5: Clinical scans of a head and neck cancer patient retrieved from the Cancer Imaging Archive [21]. (A) Planning CT. (B) PET-CT. Notice the variation in the compression of the spine and the tilt of the head due to the different head rests used in each procedure.

weighted sum of all its inputs:

$$a_j = g(\sum_{i=0}^{N} w_{ij} a_i)$$

where there are N nodes in a layer. The activation function either produces a binary output (0 or 1) or a value between 0 and 1 [22]. The output from node j is used as an input to the following layer and so on. An entire network of these nodes and links can be designed for various purposes. This requires training the neural network. Training involves giving an input and explicitly telling the network what output should be produced. In doing so, the weights on each link are adjusted in order to match the given output. When trained correctly from input-output pairs (this requires thousands of pairs), the network can then predict an accurate output when only given the corresponding input [22]. This is known as supervised learning. Simple network designs can be very effective at performing relatively simple tasks such as sales forecasting and data validation. However, to perform more complicated tasks, such as image registration, deep neural networks are favoured [23].

Deep neural networks consist of more layers, allowing for improved results and higher levels of abstraction [24]. This makes them especially useful for image analysis tasks. In particular, a subcategory of deep neural networks, known as convolutional neural networks (CNN) have proven to be extremely powerful image analysis tools [25]. The convolutional layer is the foundation of the CNN and is designed to receive images as inputs.

Convolutional layers operate on the principle that local variations within an image can be classified into a small number of categories (e.g. edges, corners.) [26]. Each node in a layer receives as input a set of nodes in a small region of the previous layer. Feature detection occurs by applying a filter (kernel) on the previous layer. Fig 6 shows this process schematically. The kernel is applied across the input image. The dot product of the kernel and the input region produce an output, known as the feature map. Increasing the number of convolutional layers allows for the detection of higher-order features.

A CNN's ability to quickly and accurately detect features has made them extremely attractive to medical research. They have been applied to a variety of tasks from malignancy detection to image registration [25]. These networks have been applied to perform both rigid and non-rigid image registration and have proven to be much more efficient than the alternatives that do not implement machine learning [6].

Common CNN image registration networks such as VoxelMorph are unsupervised [28]. They do not require input-output pairs but rather learn to minimise a given function. Although we seek to take an alternative approach to performing deformable registration of 3D volumes, we implement a similar network structure (Fig 7). Initially, this structure, known as U-Net was proposed by Ronneberger et al [29] to perform image segmentation.

The architecture involves a number of convolutional layers at various resolutions. The downsampling path (left) has the purpose of extracting feature maps at various resolutions. The skip connections represented by grey arrows serve to reintroduce local information to the global information while upsampling (right) [29]. We believe a similar architecture can be of use for

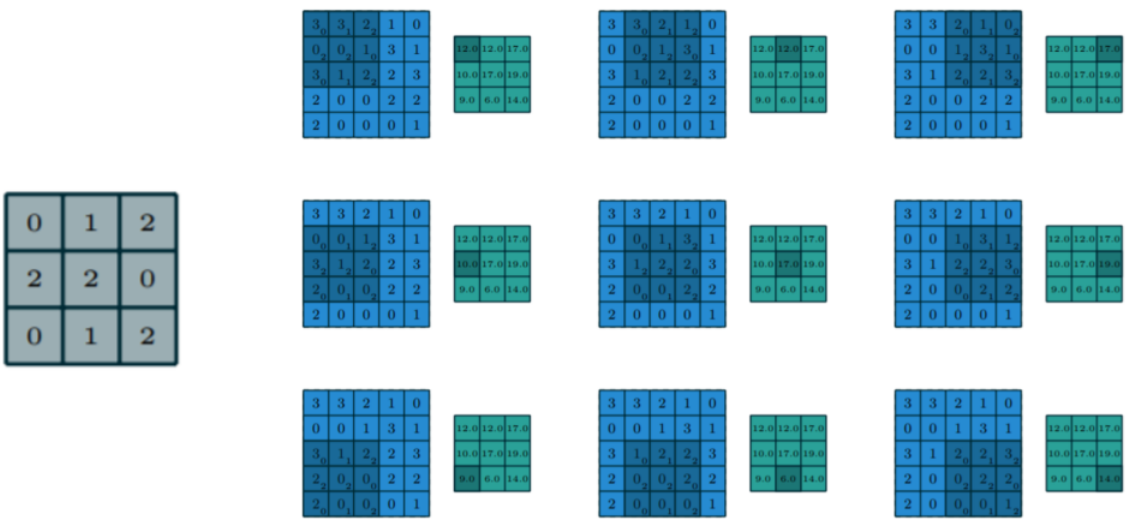

Figure 6: Convolution operation. The kernel (left) is applied to the input image in blue. The resulting dot product forms the feature map in green [27].

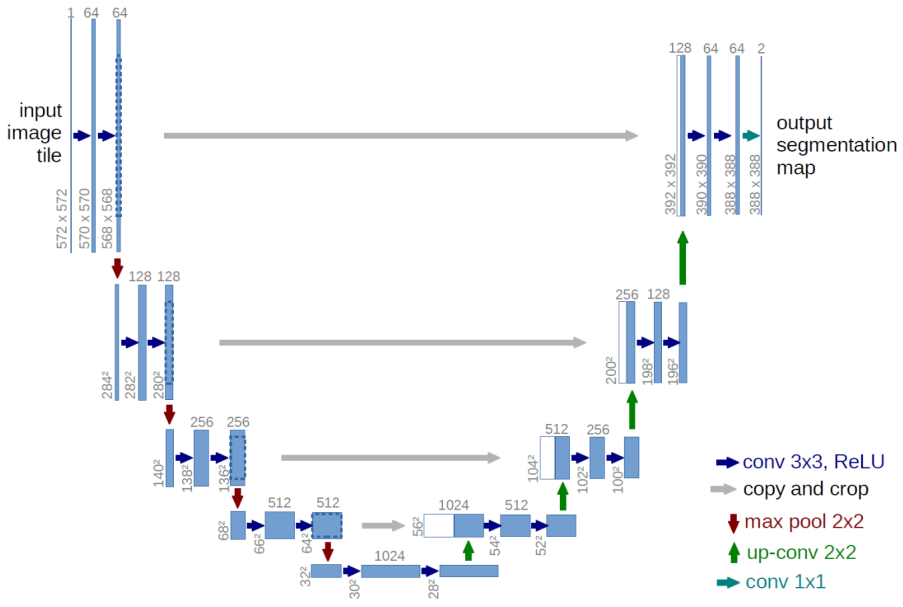

Figure 7: Illustration of U-Net network architecture [29].

supervised deformable image registration.

3 Method

3.1 Data pre-processing

Initially, we sought to implement our method using open-source data available from the Cancer Imaging Archive (CIA) [21]. This archive contains PET-CT and radiotherapy planning CT (PCT) imaging data for 298 head-and-neck cancer patients from various institutions around Québec. The imaging data is available in the form of DICOM files (.dcm extension) with various images associated with each patient. These include PET-CT and PCT image series, the displacement vector fields (DVF) and the structure set which contains contours of various organs, useful for segmentation purposes. The image volumes are saved in the form of image series where each image represents a slice of the volume in the x-y plane.

Our first task involved using the *pydicom* python library in order to load in the DICOM files in a comprehensible format. The first challenge involved understanding the structure of DICOM datasets and how to access the elements which were of interest to us. Indeed, these DICOM files contain a large amount of irrelevant information for our purposes, such as the time the image was taken, the manufacturer's model name and so on. To further understand the organisation of such files, we read in the datasets using *pydicom* and wrote them to text files. After analysing the data, we understood that *keywords* were used to index elements and *dicom sequence* type elements were lists of keywords. An improved understanding of the dataset structure made our work more efficient as we would only be reading in relevant information, thus decreasing the computational load.

To test our understanding we attempted to isolate the information which was important to us. For the image volumes, these included: the image position (patient), image orientation (patient), slice thickness, slice location, pixel spacing and pixel array. The image position and image orientation describe the position (x, y, z coordinates) of the upper left voxel of the image in mm and the direction cosines (i.e orientation) of the slice relative to the patient, respectively. The slice thickness and slice location informed us of the distance in mm along the z-axis between slices and the relative position of the image plane in mm (i.e distance along the z-axis relative to an arbitrary origin), respectively. Finally, the pixel spacing and pixel array define the size of each voxel and the greyscale value associated with each voxel, respectively.

The image position and image orientation were also of use when reading the DVFs but we needed additional information such as the grid dimensions, grid resolution and vector grid data. The grid dimensions and grid resolution are used to define the grid used for performing the registration of the images; they describe the size of the grid in the x,y and z directions and the size of each element in mm, respectively. The vector grid data element contains the 3D vectors used to define the non-rigid deformation. Further, the dataset contains the information related to the rigid transformation which preceeds the non-rigid deformation.

After inspection we noticed that these elements could be accessed through a combination of calling class variables and list indexing. As an example, the code used to access the image position data is shown in Listing 1. A similar process was used to extract the other important information. The ability to extract specific information and manipulate it according to our needs was extremely useful in implementing the data alongside other packages which could not read DICOM files. Before training the network, we needed to visually check the accuracy of the DVFs which accompanied the scans. We aimed to do this by applying the DVFs to the PET-CT and overlaying the planning CT. Enabling us to visually assess the accuracy of the registration.

Listing 1: Code used to access specific elements within DICOM dataset

```
# Creates a dataset object associated with the image
dataset = pydicom.dcmread(image)
# Extract image position (patient), Sequence elements are lists of
    keywords
image_position = dataset.DeformableRegistrationSequence[1].
    DeformableRegistrationGridSequence[0].ImagePositionPatient
```

At first, we attempted to resample the PET-CT onto the Planning CT using the python binary for the *SimpleITK* C++ package. As there was non-rigid deformation in three dimensions, we needed to resample the image volumes, not the individual slices. Fortunately, SimpleITK offers a method to convert image series into one image volume. The output was saved in the MHA format, widely used in the field. We attempted to resample the volumes by implementing the *ResampleImageFilter* method alongside the *DisplacementFieldTransform* method where we used the DVFs from the DICOM files as arguments. This can be seen in Listing 2. However, after trying multiple formats and altering input arguments we were unable to use these methods to perform resampling and visually check the accuracy of the deformation vectors. It was thought that this could be due to starting the non-rigid transform without first initiliazing the affine (rigid) transform. Further tests were implemented following the initial rigid transform but the results persisted. The source of the issue was not definitively determined but we believe it could be caused by the format used when saving deformation fields in the DICOM files.

Listing 2: Code snippet used to implement SimpleITK for image registration

```
import SimpleITK as sitk
# Initialize filter
resample = sitk.ResampleImageFilter()
# Define the fixed image
resample.SetReferenceImage(fixed_img)
# Initialize transform
dis_tx = sitk.DisplacementFieldTransform(sitk.Cast(dis, sitk.
    sitkVectorFloat64))
resample.SetTransform(dis_tx)
# Perform transformation
out = resample.Execute(moving_img)
```

As an alternative, we tried to perform registration by implementing the *SimpleElastix* python package. A python wrapper for *SimpleITK* which includes the elastix C++ package [30]. We believed that implementing the python package would allow for easier integration with the rest of our code. However, this task proved to be challenging. We were unable to apply the DVFs and visually assess the accuracy of the registration. At this point, we decided to disregard the deformation vector fields included in the CIA dataset and perform registration ourselves.

We attempted to use *SimpleITK*, *SimpleElastix* and *pyelastix*, a python wrapper for elastix, to register the volumes as we were more experienced with python and thought that using these packages would enable further control of the process. Once again, we encountered difficulties with *SimpleITK* and *SimpleElastix*. We succeeded in performing registration with *pyelastix* but were unable to extract the DVFs in a usable format. We also performed registration using *3D Slicer*, a medical imaging data processing environment [31]. Although, we were able to register two volumes for one patient, applying this process to the whole dataset would have been impractically time-consuming due to the need to use *3D Slicer*'s GUI. It seemed that the best alternative was to write a bash script using the native elastix executable to perform registration ourselves.

In an attempt to perform registration ourselves, we tried to automate a process by which each patient's PET-CT and Planning CT image series were converted to an image volume. The idea was to write all image volumes into one directory to improve the speed of our registration and organise our data more effectively. In doing so, we encountered an issue with the way the CIA dataset was organised. Directory names across patients varied, the number of sub-directories also differed and the number of files within each directory changed. An example of the encountered file trees and our desired organisation are shown in Fig 8. This made automating a sorting process difficult.

We wrote the code shown in Listing 3 to organise the CIA dataset more effectively. This function sorts patients with two directories within the patient folder and ignores other structures. We expected one of these folders to contain PET-CT data and the other Planning CT data.

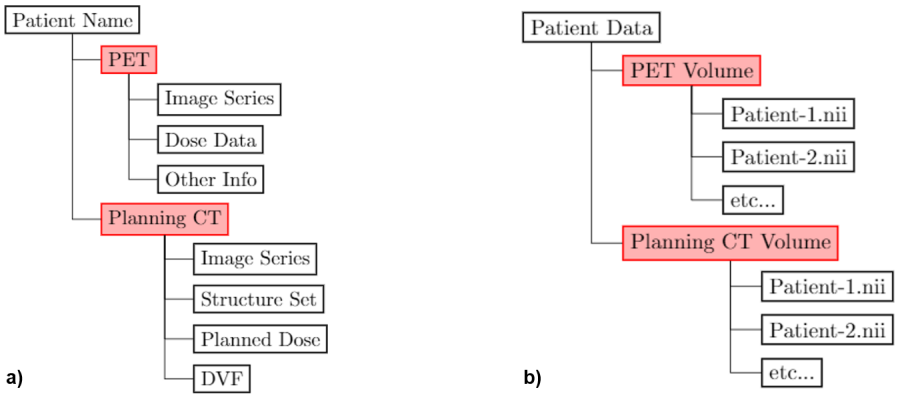

Figure 8: Example file trees. a) Before sorting. Variations occured frequently. b) After sorting

However, some patients only had one directory, others had as many as four. We decided to skip these exceptions as they further complicated our algorithm and would not affect our results. The function then sorts these two directories by size (in memory). The largest folder is usually the one containing the PET-CT as these images tend to have a greater field of view and thus more images. Subsequently, the sub-directories associated with each modality are then ranked according to the number of files within them. The folders with the greatest number of files contain the scans since there is one file per slice. The paths of the image series for each patient were written to a dictionary with the key being the patient's name. Subsequently, the image series were written in the form of image volumes in Nifty format (.nii extension), a widely used format in the community.

Listing 3: Code used to sort dataset

```
pet_paths = {}
pct_paths = {}
# Iterate over folders in patient directory
for f in os.listdir(patient_dir):
        # Skip entry if number of subdirectories != 2
        try:
                patient_path_list = os.path.join(patient_dir, f)
                folders = [os.path.join(patient_path_list, file)
                for file in os.listdir(patient_path_list)]
                if len(folders) == 2:
                        # Sort by size, largest = PET, other = PCT
                        folders.sort(key=get_size, reverse=True)
                        pet_path = folders[0]  # Inside patient folder
                        pct_path = folders[1]
                        # Organise PET and CT folders, by number of
                           files per folder
                        pet_files = [os.path.join(pet_path, folder) for
                           folder in os.listdir(pet_path)]
                        pct_files = [os.path.join(pct_path, folder) for
                           folder in os.listdir(pct_path)]
                        pet_files.sort(key=get_number_files, reverse=
                           True)
                        pct_files.sort(key=get_number_files, reverse=
                           True)
                        pet_paths[f] = pet_files[0]
                        pct_paths[f] = pct_files[0]
                else:
                        print("Can't␣extract␣scans")
        except IndexError:
                print("Too␣few␣files")
                pass
```

Due to the variations in the file organisation for each patient, some patients were skipped. Ideally, we would have developed a more general algorithm, capable of identifying all paths for every patient. However, our method resulted in 110 patients being accessed. Thus we had 110 PET-CT/Planning CT pairs to register, a number we deemed to be sufficient to train our network. If more patients were required, we would have optimised our sorting function. Further, the new organisation system made registration more accessible.

A snippet of the bash script using *elastix* to register the two volumes is shown in Listing 4. The directory paths have been omitted here for conciseness. The *transformix* module accompanies *elastix* and is mainly used to apply the generated transforms to other images. Here, we implemented it to generate the deformation vector fields in NIFTY format from the non-rigid

transform parameters. To perform image registration, *elastix* requires an input parameter file which contains general information about the desired transformation, the parameters are then fine-tuned to the specific image pair and a distinct parameter file is written. Fortunately, a database is available containing initial parameter files for various applications. We implemented the rigid and b-spline parameter files developed by Brouwer et al [32] for 3D CT scans of head-and-neck cancer patients. The generated DVFs were applied to the PET-CT volumes in *3D Slicer* where we visually checked their accuracy. The results were deemed acceptable so we began developing our convolutional neural network.

Listing 4: Code used to perform registration with elastix

```
# Loop over all patients saved in Nifty Patients
for filepath in `(ls -f ${base_dir}/${fixed_ref}/*.nii )`; do
        #  Get patient name from filename
        filename=$(basename -- "$filepath")
        image="${filename%.*}"
        # Perform Registration, first affine then b-spline
        elastix -f ${base_dir}/${fixed_ref}/${image} -m ${base_dir}/${
           moving_ref}/${image} -p ${cwd}/Affine-parameter-file.txt  -
           out ${out_dir}/Rigid/${image}
        elastix -f ${base_dir}/${fixed_ref}/${image} -m ${out_dir}/Rigid
           /${image}/result.0.nii -p ${cwd}/B-Spline-parameter-file.txt
            -out ${out_dir}/Non-Rigid/${image}
        # Generate deformation field from b-spline transform file
        transformix -def all -tp ${out_dir}/Non-Rigid/${image}/
           TransformParameters.0.txt -out ${out_dir}/DVF/${image}
done
```

3.2 Machine Learning

Having overcome the issues with pre-processing our data, we began work towards training a network. To speed up this process we looked into using NiftyNet, a deep-learning platform for medical imaging [33]. NiftyNet offers a number of networks used for segmentation including U-Net, a deep neural network which has been applied to image registration tasks [34]. VoxelMorph is an example of an algorithm which implements the U-net architecture to perform image registration of 3D MR scans of brain cancer patients [35]. Included in NiftyNet is a weakly-supervised convolutional neural network developed by Hu et al [36] which implements the U-Net architecture. We believed it could be modified to fit our needs, so we used it as the foundation to our work. Due to the memory allocation requirements and the need for powerful GPUs when training a neural network we needed to find an alternative to working on our personal computers. We were given access to the Tesla V100 GPUs belonging to the High Energy Physics Group in the School of Physics. Having installed all our dependencies in a python 3.6.6 virtual environment, we began modifying the network originally developed by Hu and colleagues. A few modifications

needed to be made in order to get the network functioning.
The weakly-supervised method seeks to generate DVFs and apply them to known organ masks. Masks are regions which delimit the volume of a specific organ. Subsequently, a similarity metric is calculated between the Planning CT mask and the deformed PET-CT mask. The network is trained to minimise this metric and thus provide accurate deformation fields. Our idea was to directly compare the output DVFs from the network to the ones generated by our registration with *elastix*. Thus, we needed to implement a root-mean-square loss function instead of the aforementioned similarity metric. Our modified loss function is shown in Listing 5.

Listing 5: Modified loss function.

```
# ddf_label = an array representing the dvfs generated by elastix
# ddf = an array representing the dvfs generated by the network
# Remove empty dimensions
ddf_label = tf.squeeze(ddf_label)
# Flatten array: ddf_flat[0] = batch size, ddf_flat[1] = size
ddf_flat = tf.contrib.layers.flatten(ddf)
ddf_label_flat = tf.contrib.layers.flatten(ddf_label)
ddf_error = []
# Find difference between each element
ms = tf.subtract(ddf_flat, ddf_label_flat)
# Take absolute value of the diference
rms = tf.abs(ms)
ddf_error.append(tf.reduce_sum(rms, axis=1))
return tf.reduce_mean(ddf_error)
```

Further modifications were required to work with our inputs and our lack of organ masks. We encountered issues when we tried to train the network as it required inputs of identical sizes. In addition, smaller inputs were required as our volumes were too large for the GPUs to process. *SimpleITK* was used to resample all of our image volumes to a [64,64,64] array to avoid these issues. This was implemented as seen in Listing 6.

Listing 6: Code used to resample images to 64x64x64

```
startImage = sitk.ReadImage(inputImage)
size = [64, 64, 64]
# factors to downsize by
factors = [a/b for a, b in zip(startImage.GetSize(), size)]
# Calculate the new pixel spacing and image size
new_spacing = [a*b for a, b in zip(startImage.GetSpacing(), factors)]
new_size = [int(a/b) for a, b in zip(startImage.GetSize(), factors)]
# do some in-plane blurring
sig = 2.0/(2.0*np.sqrt(np.pi))
# Blur in each plane separately
blurredImage = sitk.RecursiveGaussian(startImage, sigma=sig*new_spacing
    [0], direction=0)
blurredImage = sitk.RecursiveGaussian(blurredImage, sigma=sig*
    new_spacing[1], direction=1)
# Build the resampling filter, and then apply it
resampleFilter = sitk.ResampleImageFilter()
downsampledImage = resampleFilter.Execute(blurredImage,
                                  new_size,
                                  sitk.Transform(),
                                  sitk.sitkBSpline,
                                  startImage.GetOrigin(),
                                  new_spacing,
                                  startImage.GetDirection(),
                                  0,  # Default pixel value
                                  startImage.GetPixelIDValue())
# Write the result to check
sitk.WriteImage(downsampledImage, outputImage)
```

Elastix registration needed to be performed again as the resampling would affect the deformation field. Our resampled data was then used to train the network. Our results were tested by using our CNN to register two pairs of PET-CT and Planning CT volumes. The outputs were viewed in *3D Slicer*. Further tweaking of the network parameters (e.g learning rate, batch size, training epochs,..) was performed to optimise the network. Many features of the network were analysed using Tensorboard, a visualisation tool which accompanies Tensorflow.

4 Results

Initially, we performed registration with a traditional algorithm, namely *elastix*. An example of the output from such a registration is shown in Fig 9. Colours were selected to increase contrast between the fixed Planning CT and the moving PET-CT. *Elastix* took (161.3 ± 5.6) seconds per

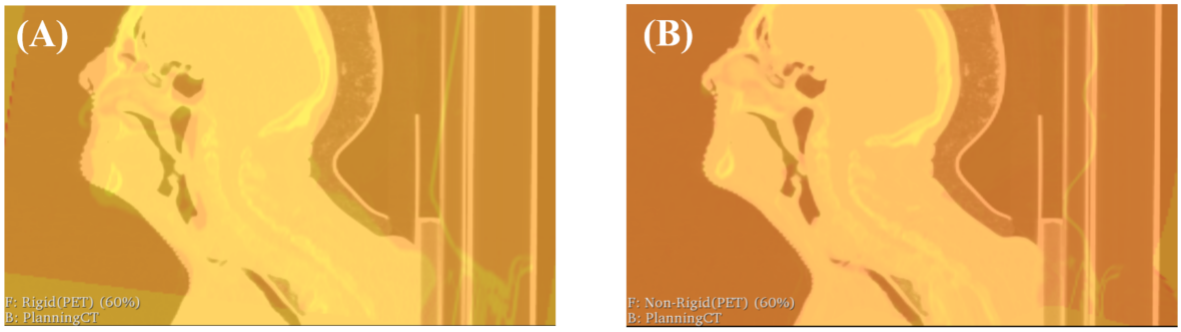

Figure 9: *Elastix* registration output. (A) Initial rigid transformation. PET-CT is the least opaque image, discrepancies clearly visible around the chin and nose. (B) Non-Rigid transformation. No clear differences.

Figure 10: CNN registration output. (A) Result of resampling Fig 9(A). Used as inputs. (B) Non-rigid registration output. Note that the output has been translated away from the fixed CT.

image pair to perform non-rigid registration, where we have assumed iterations were independent.
One of the results from our CNN is shown in Fig 10. Clearly, mistakes have been made. Our registration took (5.76±0.05) seconds per image., again we have assumed iterations are independent. A plot of our loss function over 5000 iterations is shown in Fig 11.

5 Discussion & Future Work

It is obvious that our network is not yet working as intended. We believe that this could be due to a number of factors. Firstly, this could be the result of resampling the inputs to a lower resolution. In doing so, we may have lost some vital local information (e.g. gland definition, individual vertebrae,..) which could affect the results of our registration. This issue may have been magnified by the fact that PET-CT and planning CT volumes have different slice thicknesses. By resampling both images to identical sizes, we may have lost some important

Figure 11: CNN loss value vs number of iterations. Graph was plotted using Tensorboard. The second less opaque plot appears due to smoothing, an in-built feature in Tensorboard.

information. If this is the case, we can tackle this issue by first slicing the initial full resolution volumes to select a region of interest. Therefore, when resampling, the effects would be less drastic and would maintain an adequate level of local information.

Another possibility is that our loss function has not be defined appropriately. This could cause training the CNN to perform an unwanted task. Investigating the effectiveness of a variety of loss functions seems like the most effective solution to this problem.

Despite this, our results are promising. Our loss is decreasing as the number of iterations increases as is expected from a trained network. A loss value of the order 10^6 is expected due to the large number of parameters being optimised. We were also able to train the network and produce an output. Further, our CNN is nearly 30x faster than a traditional algorithm at performing deformable registration of 3D volumes. This indicates that if we produce outputs comparable to those produced by *elastix*, our method would be much more appropriate for performing fast and accurate registration of 3D volumes.

In addition, our network does not require any parameter files to perform registration. It is believed that it could be applied in a number of settings as long as a large training set is available.

In future, the main focus will be to improve the accuracy of our image registration algorithm. Subsequently, we will need to verify our results across a number of patients and determine whether our results are comparable to those achieved via slower traditional algorithms. Finally, it may be interesting to explore an unsupervised method, which will work without needing large training datasets. Hopefully, making this algorithm more accessible. Comparing the speed and accuracy of the two methods may be interesting to future research.

References

[1] International Agency for Research on Cancer, "Latest global cancer data: Cancer burden rises to 18.1 million new cases and 9.6 million cancer deaths in 2018," tech. rep., 2018.

[2] R. L. Siegel, K. D. Miller, and A. Jemal, "Cancer statistics, 2018," *CA: A Cancer Journal for Clinicians*, vol. 68, pp. 7–30, jan 2018.

[3] J. L. Barrera-Franco, J. F. Gallegos-Hernández, M. Granados-García, and H. Gurrola-Machuca, "Multidisciplinary approach in head and neck cancer," *Gaceta Mexicana de Oncologia*, vol. 16, sep 2018.

[4] R. Baskar, K. A. Lee, R. Yeo, and K.-W. Yeoh, "Cancer and radiation therapy: current advances and future directions.," *International journal of medical sciences*, vol. 9, no. 3, pp. 193–9, 2012.

[5] A. C. Begg, F. A. Stewart, and C. Vens, "Strategies to improve radiotherapy with targeted drugs," *Nature Reviews Cancer*, vol. 11, pp. 239–253, apr 2011.

[6] K. A. J. Eppenhof, M. W. Lafarge, P. . Moeskops, M. . Veta, and J. P. W. Pluim, "Deformable image registration using convolutional neural networks," *Event: SPIE Medical Imaging*, vol. 10574, no. 2, 2018.

[7] B. Sahiner, Heang-Ping Chan, N. Petrick, Datong Wei, M. Helvie, D. Adler, and M. Goodsitt, "Classification of mass and normal breast tissue: a convolution neural network classifier with spatial domain and texture images," *IEEE Transactions on Medical Imaging*, vol. 15, no. 5, pp. 598–610, 1996.

[8] G. Wu, M. Kim, Q. Wang, Y. Gao, S. Liao, and D. Shen, "Unsupervised deep feature learning for deformable registration of MR brain images.," *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention*, vol. 16, no. Pt 2, pp. 649–56, 2013.

[9] S. Shan, W. Yan, X. Guo, E. I.-C. Chang, Y. Fan, and Y. Xu, "Unsupervised End-to-end Learning for Deformable Medical Image Registration,"

[10] S. P. Jackson and J. Bartek, "The DNA-damage response in human biology and disease," *Nature*, vol. 461, pp. 1071–1078, oct 2009.

[11] J. Ward, "DNA Damage Produced by Ionizing Radiation in Mammalian Cells: Identities, Mechanisms of Formation, and Reparability," *Progress in Nucleic Acid Research and Molecular Biology*, vol. 35, pp. 95–125, jan 1988.

[12] M. J. Gazda and L. R. Coia, "Principles of radiation therapy," tech. rep.

[13] D. W. Townsend, "Combined positron emission tomography-computed tomography: the historical perspective.," *Seminars in ultrasound, CT, and MR*, vol. 29, pp. 232–5, aug 2008.

[14] S. S. Gambhir, "Molecular imaging of cancer with positron emission tomography," *Nature Reviews Cancer*, vol. 2, pp. 683–693, sep 2002.

[15] G. T. Herman, *Fundamentals of Computerized Tomography Image Reconstruction from Projections.* Springer-Verlag New York Inc, 2012.

[16] N. G. Burnet, S. J. Thomas, K. E. Burton, and S. J. Jefferies, "Defining the tumour and target volumes for radiotherapy.," *Cancer imaging : the official publication of the International Cancer Imaging Society*, vol. 4, pp. 153–61, oct 2004.

[17] J. Dobbs and A. Barrett, "Practical radiotherapy planning," 1985.

[18] S. Oh and S. Kim, "Deformable image registration in radiation therapy.," *Radiation oncology journal*, vol. 35, pp. 101–111, jun 2017.

[19] A. Sotiras, C. Davatzikos, and N. Paragios, "Deformable medical image registration: a survey.," *IEEE transactions on medical imaging*, vol. 32, pp. 1153–90, jul 2013.

[20] H. J. Johnson and M. M. McCormick, "The ITK Software Guide Book 1: Introduction and Development Guidelines Fourth Edition Updated for ITK version 5.0.0," tech. rep., 2018.

[21] M. Vallières, E. Kay-Rivest, L. J. Perrin, X. Liem, C. Furstoss, H. J. W. L. Aerts, N. Khaouam, P. F. Nguyen-Tan, C.-S. Wang, K. Sultanem, J. Seuntjens, and I. El Naqa, "Radiomics strategies for risk assessment of tumour failure in head-and-neck cancer," *Scientific Reports*, vol. 7, p. 10117, dec 2017.

[22] S. J. S. J. Russell, P. Norvig, and E. Davis, *Artificial intelligence : a modern approach.* Prentice Hall, 2010.

[23] L. V. Fausett, *Fundamentals of neural networks : architectures, algorithms, and applications.* Prentice-Hall, 1994.

[24] L. Yann, B. Yoshua, and G. Hinton, "Deep Learning," *Nature*, 2015.

[25] H. Greenspan, B. van Ginneken, and R. M. Summers, "Guest Editorial Deep Learning in Medical Imaging: Overview and Future Promise of an Exciting New Technique," *IEEE Transactions on Medical Imaging*, vol. 35, pp. 1153–1159, may 2016.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," *IEEE*, p. 46, 1998.

[27] V. Dumoulin, F. Visin, and G. E. P. Box, "A guide to convolution arithmetic for deep learning," tech. rep., 2016.

[28] G. Balakrishnan MIT, A. Zhao MIT, M. R. Sabuncu, J. Guttag MIT, and A. V. Dalca MIT, "An Unsupervised Learning Model for Deformable Medical Image Registration," tech. rep.

[29] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," may 2015.

[30] K. Marstal, F. Berendsen, M. Staring, and S. Klein, "SimpleElastix: A user-friendly, multi-lingual library for medical image registration," tech. rep.

[31] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, J. Finet, J.-C. Fillion-Robin, S. Pujol, C. Bauer, D. Jennings, F. Fennessy, M. Sonka, J. Buatti, S. Aylward, J. V. Miller, S. Pieper, and R. Kikinis, "3D Slicer as an image computing platform for the Quantitative Imaging Network.," *Magnetic resonance imaging*, vol. 30, pp. 1323–41, nov 2012.

[32] C. L. Brouwer, R. G. Kierkels, A. A. van 't Veld, N. M. Sijtsema, and H. Meertens, "The effects of computed tomography image characteristics and knot spacing on the spatial accuracy of B-spline deformable image registration in the head and neck geometry," *Radiation Oncology*, vol. 9, p. 169, jul 2014.

[33] E. Gibson, W. Li, C. Sudre, L. Fidon, D. I. Shakir, G. Wang, Z. Eaton-Rosen, R. Gray, T. Doel, Y. Hu, T. Whyntie, P. Nachev, M. Modat, D. C. Barratt, S. Ourselin, M. J. Cardoso, and T. Vercauteren, "NiftyNet: a deep-learning platform for medical imaging," *Computer Methods and Programs in Biomedicine*, vol. 158, pp. 113–122, may 2018.

[34] C. Ozgün, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation," tech. rep.

[35] G. Balakrishnan, A. Zhao, M. R. Sabuncu, J. Guttag, and A. V. Dalca, "VoxelMorph: A Learning Framework for Deformable Medical Image Registration," tech. rep.

[36] Y. Hu, M. Modat, E. Gibson, W. Li, N. Ghavami, E. Bonmati, G. Wang, S. Bandula, C. M. Moore, M. Emberton, S. Ourselin, J. A. Noble, D. C. Barratt, and T. Vercauteren, "Weakly-Supervised Convolutional Neural Networks for Multimodal Image Registration," tech. rep.

A Appendix

Our work was completed using python 3.6.6, bash 4.4.19, elastix 4.9, pydicom 1.2.1, SimpleITK 1.1.0, pyelastix 1.1, 3D Slicer 4.10 and SimpleElastix 1.1.
All versions of our code are available at: https://github.com/DMcSweeney/MPhys