DEEP Q-LEARNING IN SAGITTAL MAGNETIC RESONANCE IMAGING OF THE FOOT AND ANKLE

BY

MICHAEL J. ROBINSON

A THESIS SUBMITTED TO THE SCHOOL OF COMPUTING, COLLEGE OF COMPUTING AND DIGITAL MEDIA OF DEPAUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN ARTIFICIAL INTELLIGENCE

DEPAUL UNIVERSITY
CHICAGO, ILLINOIS

2024

DePaul University
College of Computing and Digital Media

MS Thesis Verification

This thesis has been read and approved by the thesis committee below according to the requirements of the School of Computing graduate program and DePaul University.

Name: Michael J. Robinson

Title of dissertation: DEEP Q-LEARNING IN SAGITTAL MAGNETIC RESONANCE IMAGING OF THE FOOT AND ANKLE

Date of Dissertation Defense: October 25, 2024

Jacob Furst, PhD

Advisor*

Noriko Tomuro, PhD

1st Reader

John Rogers, PhD

2nd Reader

** A copy of this form has been signed, but may only be viewed after submission and approval of FERPA request letter.*

Abstract: Fracture is a common occurrence, with approximately 6.8 million fractures in the United States being treated yearly (Lehigh Valley Health Network, 2024). A tool that could identify these in medical images by proposing candidate image patches (that contain pathologies) could save radiologists time and provide them with vital assistance. Such tools could also serve in landmark identification, a crucial step in surgical planning. To this effect, there are many prominent works in the computer science medical imaging community that use reinforcement learning (RL), supervised learning or traditional imaging processing techniques for surgical planning, fracture classification and detection, edema detection, landmark identification or organ (or pathology) segmentation (Ackermann et al., 2021; Alansary et al., 2019; Ghesu et al., 2019; Ma et al., 2024; Navarro et al., 2020; Rahmaniar & Wang, 2019; Ribeiro et al., 2023).

This work focuses on two problems: edema localization in Sagittal-oriented Inversion Recovery (SAGIR) images and calcaneus (heel) bone segmentation in Sagittal-oriented T1 (SAGT1) images. Edema often occurs with fracture, and although it can occur without fracture, identifying it could be a first step in indicating regions on the foot where a fracture is located (Zubler et al., 2007). Significant training and experimentation of a deep Q-learning agent to identify and localize edema in SAGIR Magnetic Resonance Imaging (MRI) fails to yield satisfactory result. The heterogenous location and appearance of edema complicate its segmentation. Future work that uses more complex state representation combined with allowing the agent to stop mid-search via a trigger action, an external classifier and multiple start locations could lead to better results. Some of these ideas are explored independently in this work, while others are found in other related works (Caicedo & Lazebnik, 2015).

In terms of calcaneus segmentation, given that approximately two percent of all fractures are calcaneal fractures, being able to localize the bone could aid in diagnosis, as one paper creates a bone mask (for the knee) to find edema (Pranata et al., 2019; Ribeiro et al., 2023). I achieve limited success in using reinforcement learning (RL) to locate the calcaneus in SAGT1 MRIs with an intersection over union (IoU) score of 0.2. for a set of 288 slices. Further investigation and experimentation are needed to see if improvement is possible.

This works demonstrates proof that RL, especially with the use of deep neural networks has potential to learn structural patterns in MRI. Further work to explore fracture directly, and for cases where both edema and fracture are present is warranted. Additional experimentation for slices that have no pathologies at all, or when pathologies cannot be easily bounded by a single rectangular region is needed. Incorporating various view planes such as axial or coronal imaging of MRIs or including other scan types could also be explored.

1 Introduction

1.1 The need for tools to aid in diagnosis of foot fractures

As the number of medical images continues to grow every year, it is crucial that we continue to develop machine learning tools to assist radiologists. Many tools and algorithms have already been developed to aid in diagnoses or surgeries. These tools can detect pathologies, identify anatomical landmarks, segment organs or perform view-plane planning. They can increase both the efficiency and the accuracy of radiologists, who often have to read scores of images daily. Human radiologists can tire with long shifts, and their performance accuracy can decline as the day progresses (Tanzi et al., 2020). Often times, there is not an additional colleague that can help verify the diagnoses, and CAD tools can provide assistance to help improve consistency (Tanzi et al., 2020). It is vital that CAD technology continues to improve, which can help lead to better patient outcomes and a better quality of life for radiologists. Even tools that could segment bones and create masks for them could help a radiologist pinpoint key regions of interest (ROIs) relevant to diagnoses. This is especially important given the large number of injuries that occur every year.

Annually, 2.7 millions cases of fracture occur in the EU6 countries: France, Spain, Germany, Sweden, Italy and the UK alone (Tanzi et al., 2020). Specifically for foot fracture, estimates are around 139 per 100,00 people, yearly (Chen et al., 2024). Fig. 1 shows the fracture rate for 2019 and 1990 and Fig. 2 shows common causes of these fractures. Another paper, states that 2% of all fractures are of the calcaneus (Pranata et al., 2019). As the prognosis of the fracture often rests on a successful diagnosis, it is critical that fractures are not over or under diagnosed (Meena & Roy, 2022). Overdiagnosis could lead to unnecessary downtime for patients, while a missed diagnosis could complicate or extend the recovery process.

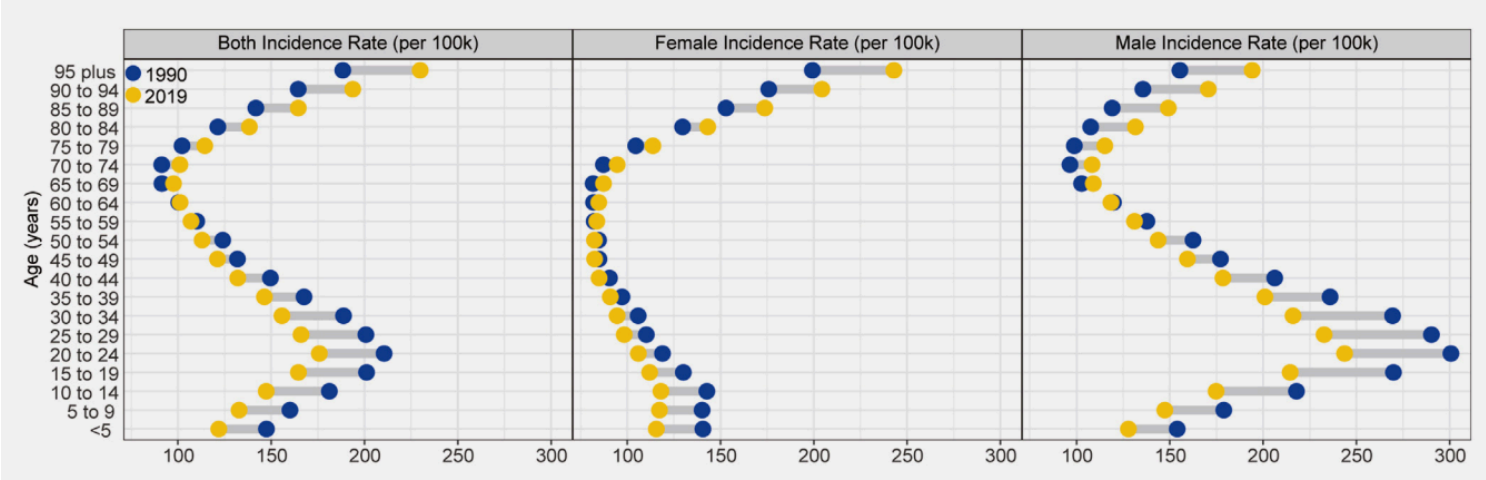

Fig. 1. Fracture rates in 1990 and 2019, globally. Reprinted from Chen et al. (2024).

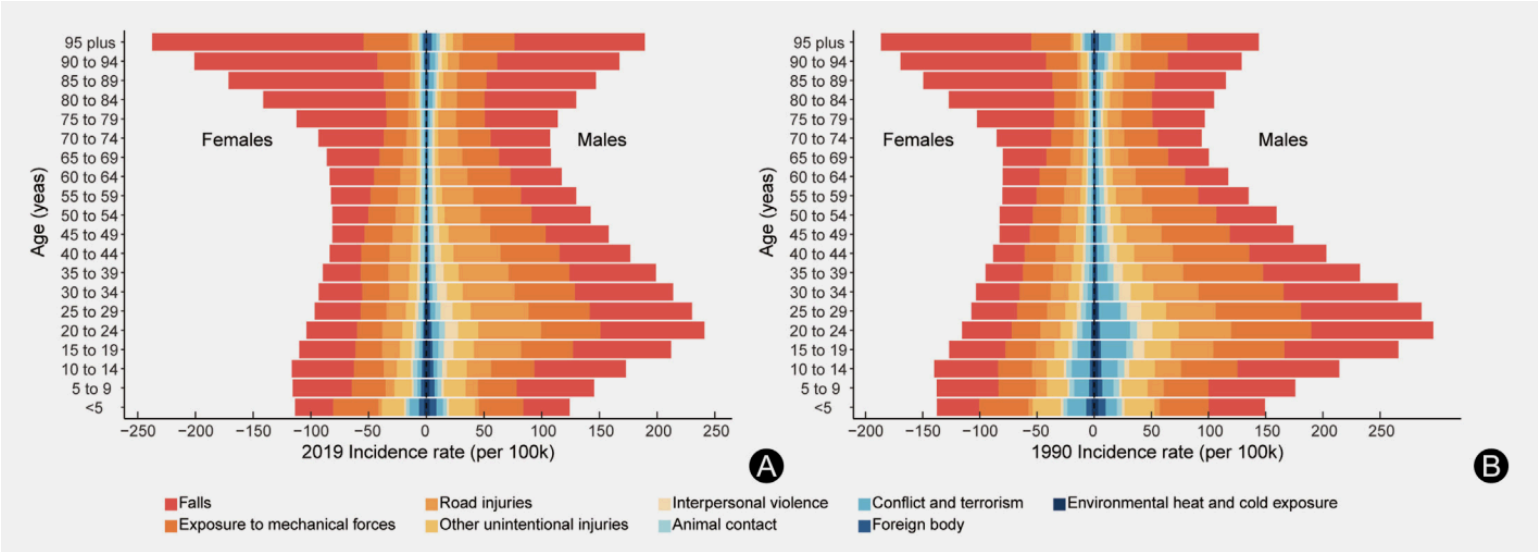

Fig. 2. Fracture rates for various causes. Reprinted from Chen et al. (2024).

1.2 Why use MRI and why examine edema?

One part of this work specifically examines edema in MRI. Although radiograph is the most common imaging modality for fracture in the emergency room, as it is quick and less costly than MRI, nevertheless MRI can be an important tool when radiograph results are unclear (Szaro, Polaczek et al., 2020). MRI can provide additional context that radiograph cannot. Especially in the case of stress fracture, radiograph lacks sensitivity (usually from 12% to 56%) but has high specificities (88% to 96%); MRI on the other hand has sensitivities typically from 68% to 99% with widely varying specificities (Crönlein et al., 2015). In general, MRI can detect pathologies that radiograph cannot – Szaro, Polaczek, et al. (2020) in examining accessory foot bones, find additional benefits in using MRI. They find that MRI can detect changes not shown on radiograph, to include contour, sclerosis, osteonecrosis. These more subtle changes may be detected in machine learning algorithms and can help lead to better diagnosis results. It is however significant to note that radiograph is still necessary and important for fracture detection and is a good baseline (Szaro, Polaczek, et al., 2020).

While Szaro, Polaczek, et al. discuss the importance of examining MRI scans for more complicated foot pain cases, one common symptom of fracture, edema, occurs frequently without fracture or even foot pain (Szaro, Geijer, et al., 2020; Szaro, Polaczek, et al., 2020). With bone marrow edema, in particular it is important to consider a patient's history, as youth often have this as it occurs with normal growth (Ribeiro et al., 2023; Szaro, Geijer, et al., 2020; Zubler et al., 2007). All in all, a tool that could help improve sensitivity by alerting a radiologist that there are potential abnormalities in certain regions of the foot could ultimately lead to less fractures missed. Fig. 3 shows a SAGIR and a SAGT1 MRI of the foot and ankle.

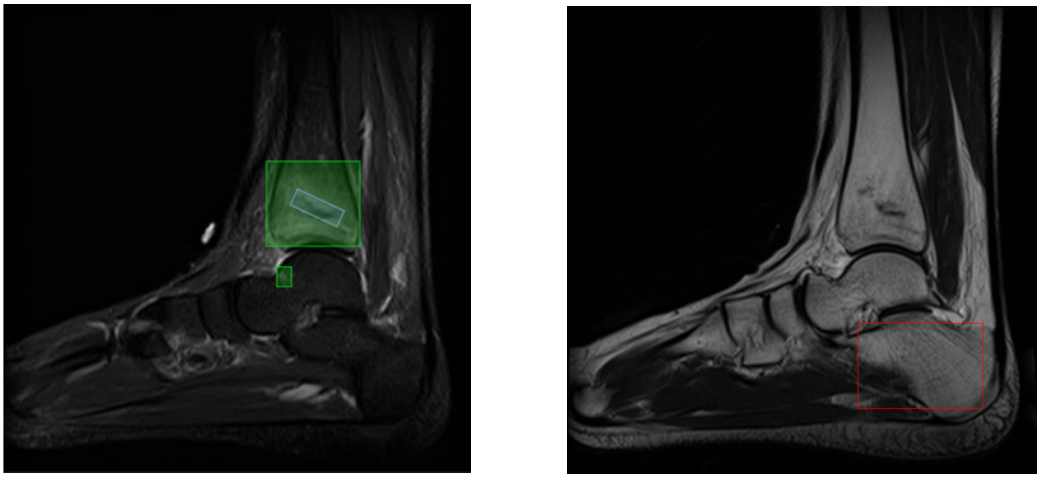

Fig. 3. The green boxes localize edema regions and the blue box bounds a fracture in a SAGIR image (left). A SAGT1 image with the calcaneus bone segmented in red (right).

1.3 The case for reinforcement learning over supervised learning

RL has been applied successfully in many fields (Sutton & Barto, 2020). Given how successful its applications have been, it is only natural to extend this to medical imaging. Publications in the field use RL to perform hyperparameter tuning, meta-learning in supervised training, thresholding, noise removal, organ localization, landmark detection, registration and pathology segmentation (Hu et al., 2023; Zhou et al., 2021). To the author's knowledge this is the first attempt at edema segmentation using deep reinforcement learning. Supervised learning

techniques such as CNNs have achieved comparable results to radiologists in fracture or edema identification already (Ribeiro et al., 2023; Tanzi et al., 2020).

Deep reinforcement learning has several elements that make it substantially more suited to difficult problems than traditional supervised learning, to include: making sequential decisions, learning even when rewards are scarce and optimizing network architectures (a non-differentiable task) (Zhou et al., 2021). It can also be memory efficient by only needing to see parts of an image at one time and can require far less images to train than a supervised classifier – a substantial advantage meaning less data augmentation is needed. (Zhou et al., 2021). With less labeled data, less time can be spent acquiring radiologists' annotations or using crowdsourcing for large dataset labeling. In addition, RL can (when trained sufficiently) be less biased than human annotators (Hu et al., 2023; J. N. Stember & Shalu, 2022). Reinforcement learning agents can find their own patterns, ultimately searching more optimally than traditional exhaustive search methods (Alansary et al., 2019).

1.4 Problem approach

I tackle the problem of identifying edema in MR images of the foot by the use of Q-learning (Watkins, 1989; Watkins & Dayan, 1992). I use the version of this that approximates the Q-function with a deep neural network, called a deep Q-network (DQN) (Mnih et al., 2015). PyTorch is used for training an agent to move a bounding box that is overlaid on a MRI slice. I utilize two networks, one to serve for policy prediction and the other one, that is updated slower, to serve as a label generator for the policy network. The final output layer outputs the Q-value for each possible action that the agent can use to modify its bounding box. The agent moves its bounding box around the image and takes in a zoomed-in version of it when in its Fit mode (see Appendix for more details on Fit mode). See Fig. 4.

Several papers use more recent reinforcement learning methods in their training, such as multi-agent approaches, Actor-Critic (AC) or Proximal Policy Optimization (PPO) (Ackermann et al., 2021; Gao et al., 2024; Kasseroller et al., 2021; Vlontzos et al., 2019). I opt to use deep Q-learning for its simplicity and ease in implementation. I acknowledge that other methods could lead to better performance. In future work I intend to explore these options. Although ultimately I did not successfully segment edema and was only relatively successful at localizing the calcaneus bone, my experiments help me better understand the limitations of RL and what future work will be most promising in achieving results.

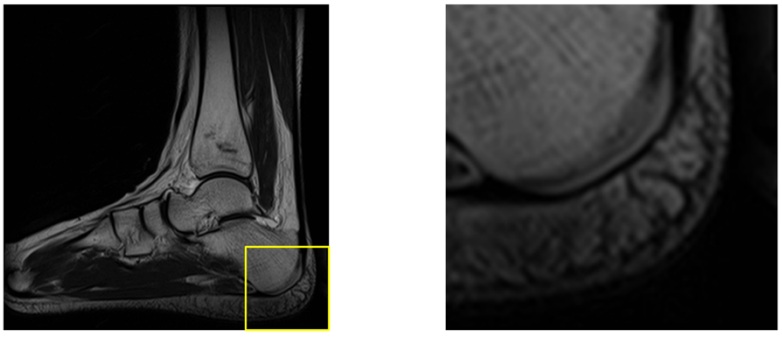

Fig. 4. The agent's bounding box in gold (left). The higher resolution input of the agent of the bounding box (right).

2 Background

2.1 Convolutional Neural Networks (CNNs)

Partially inspired by neuroscience and biology, the artificial neural network (ANN) is the backbone of supervised machine learning. A variant of it, the CNN is used to extract information from images (Russel & Norvig, 2021). A two (or three) dimensional filter (also called a kernel) is slid over an image pixel-wise in strides. The element-wise multiplication of the kernel weights and the pixel intensities are computed and then summed. A bias value is added, and then the final pixel values of the new image are used in the next layer. Typical filter sizes range anywhere from 1x1 to teens (or even more). Smaller filters detect details of the image, while bigger ones extract larger regions (Russel & Norvig, 2021). Nonlinear activations are used after convolutional layers. Rectified linear unit (ReLU) and leaky-ReLU are two of the most common, as shown in Equations 1 and 2. For Equation 2, the value for l is small, and always between zero and one. After, a max-pooling layer is used, which takes the maximum intensities of the pixels within the confines of a kernel (usually 2x2) as it slides across the modified image. These three steps are repeated several or many times before fully-connected (FC) layers. Batch normalizations are sometimes used to handle distribution shifts and mitigate vanishing gradients. Techniques to avert overfitting, such as dropout and regularization are common.

ReLU activation function

$$ReLU(x) = \max(0, x) \qquad (1)$$

Leaky ReLU activation function

$$leaky\ ReLU = \max(lx, x) \qquad (2)$$

2.2 Reinforcement Learning

2.2.1 General considerations

RL has an outstanding variety of uses and has solved problems in a wide variety of domains (Sutton & Barto, 2020). In RL, one or more agents are situated in an environment with a given goal. The environment consists of states and allowed actions. Rewards internal to the agent often signal the overall progress of external goal achievement. A Markov Decision Process (MDP) characterizes how the agent interacts with the environment and makes decisions to reach this goal. RL can work in both model-based and model-free situations. Model-based agents know the transition probabilities between states for each action and use this information to construct a world-view. Model-free ones learn from the environment solely by the given rewards and create a policy (what action to take in a state) (Sutton & Barto, 2020).

RL methods can be generally categorized as on-policy or off-policy. In on-policy learning, the agent updates the policy as it learns, and it uses this policy to make decisions. In off-policy learning, the agent follows one policy while it updates a different one. One off-policy method is (ε-greedy) Q-learning, where the agent determines actions based on the state it is in a greedy manner some of the time, but then at other times (ε percent), in a random manner.

There are various types of RL variants. Monte Carlo methods, rely on episodes finishing before updating and thus can be slower to learn. Policy iteration is quite common and used in various forms. In policy iteration, a policy is evaluated and then updated – this process is repeated until eventually the policy stops changing. Similarly value iteration, where each state is evaluated by an update equation, such as the Bellman Optimality Equation (BOE), via divide-and-conquer approaches like Dynamic Programming (DP) is also popular for small state spaces.

2.2.2 Q-learning

Proposed by Watkins in his 1989 thesis and convergence-proved in a 1992 paper, Q-learning uses Q-values to assign values to state-action pairs (Watkins, 1989; Watkins & Dayan, 1992). This precisely scores both states and actions together, while state-value-based methods only assign values to states. While state values represent how desirable being in a given state is, Q-values represent how desirable it is to take an action in a given state. Updates to these can be made iteratively in cases where there are a small number of them, and they can be stored in tabular form. This can be done synchronously where they are all updated together in a given order, which guarantees their convergence. Asynchronous updates (called backups) are typically used when there are more states and generally have less convergence stability. Many RL algorithms use the Bellman Optimality Equation (BOE). The version used in Q-learning is shown in Equation 3.

Bellman Optimality Equation in Q-learning

$$Q(s,a) = r + \gamma max_{a'} Q(s',a') \qquad (3)$$

In the equation, Q(s, a) is the Q-value of an action, *a*, in state, *s*, *r* is the reward from taking a while in *s*, γ is the discount factor which ranges from 0 to 1 and indicates how future rewards compare to current rewards. The term $max_{a'} Q(s',a')$ represents the (highest) Q-value of the best action taken in the proceeding state (s'). Since the future state depends on the past state, iterative approaches like DP can be used to solve the BOE.

2.2.3 DQN

In cases where there are a very large number of states, a neural network can be used to predict values for them. Usually two different networks are used, where one network is dictated a target network and updates slower. The other network, called the policy network decides what actions the agent will take. While the target weights are simply copied over from the policy networks every C iterations, the policy network trains using an optimizer such as Adam, Stochastic Gradient Descent or Root Mean Squared Propagation. As in Q-learning, the BOE is used when updating the weights (as it appears in the loss function). Specifically, the right side of Equation 4 is used as the ground truth (y) in the loss equation.

The BOE used in DQN

$$Q(s,a,\theta) = r + \gamma max_{a'} Q(s',a',\theta^{-}) \qquad (4)$$

Where:

- θ represents the weights of the policy network
- θ^{-} are for the target network's weights

I use the Huber Loss function because it is more immune to outliers than the Mean Squared Error or Mean Absolute Error (MAE), since it is linear for large differences between the ground truth and the output, while quadratic for small values. In Equation 5, the Huber loss function is shown (PyTorch, 2023).

The Huber Loss function

$$L(y, \hat{y}) = \begin{cases} \frac{(y-\hat{y})^2}{2}, & |y - \hat{y}| < 1 \\ |y - \hat{y}| - 0.5, & otherwise \end{cases} \quad (5)$$

Batches of previously experienced transitions are sampled randomly from the agent's memory to train the policy network. A transition consists of a state, action, reward, subsequent state and whether a terminal state was reached. The best action (a') from the subsequent state (s') is determined when the transition is selected by the target network. Fig. 5 outlines the DQN training.

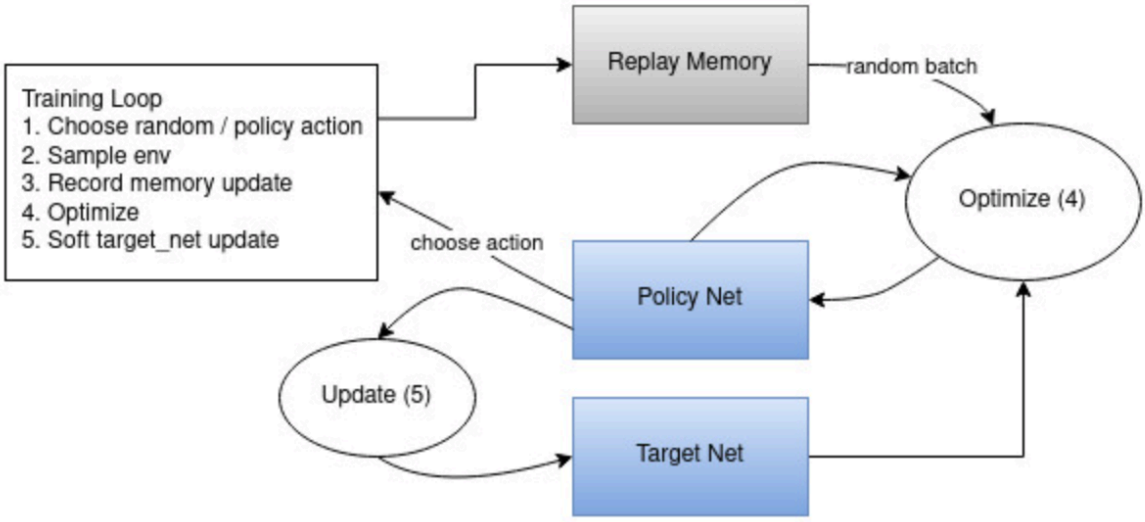

Fig. 5. DQN training. Reprinted from PyTorch (2024).

2.2.4 Double Deep Q-learning Network (DDQN), Dueling DQN, Dueling DDQN

There are several variants of DQNs that have been used in the medical imaging community (Alansary et al., 2019). In DDQN, the action is separated from the Q-value. network. In DQN, the term $max_{a'}Q(s', a',\ \theta^-)$ is determined entirely by the target network. DDQN decouples the next action, a' from the Q-value: the policy network determines the best a', while the target network determines the Q-value for this action (Alansary et al., 2019). This adds stability. This update equation is shown in Equation 6. While my earlier experiments (1-7) use a DQN in edema segmentation, my later experiments (8-10) use a DDQN for calcaneus bone localization.

BOE used in DDQN

$$Q(s, a, \theta) = r + \gamma Q(s', argmax_{a'}Q(s', a', \theta),\ \theta^-) \quad (6)$$

There are other variants of DDQN where each network is updated randomly 50% of the time. This variant is not explored in my work or by others in the medical imaging community. Finally, in dueling DQN the state value is separated from the Q-value as shown in Equation 7. In Dueling DQNs, the networks have parallel dense layers, and these layers are not connected. These represent the advantage function, A(s, a). Thus the state features are represented by the

convolutional layers as a value function (V(s)). Again, this feature is used to add stability. Dueling DDQN use the aforementioned architecture and in addition, lets the policy network select a' as described before. It simply combines the independent action values of dueling agents and the separation of the Q-value from the best subsequent action (a') of DDQN.

Dueling DQN

$$Q(s, a) = V(s) + A(s, a) \quad (7)$$

2.2.5 Actor critic (AC) and Policy Gradient methods

Similar to the vanilla DQN, a target network provides guidance for the policy network in AC methods. AC methods take it a step further than just providing target labels for each action in each state. Instead, the critic network takes in the actions performed by the policy network, in addition to the state, and outputs a single Q-value based on how good this action is. Although more unstable than the DQN, AC allows a continuous action space rather than a finite number of discrete actions (Kasseroller et al., 2021). This can significantly decrease the number of actions an agent needs to take per episode.

The REINFORCE algorithm is an older policy gradient method that tracks rewards from a series of trajectories. After computing the derivative of these trajectories, gradient ascent is used to modify the network weights in order to improve the policy (Pesce et al., 2019). This algorithm is relatively uncommon in the RL medical imaging community.

One of the newer policy optimization algorithms, Proximal Policy Optimization (PPO) keeps track of an advantage function, which measures the reward obtained from a given state. It also uses a ratio function which measures how much the policy changes since the last iteration. A clipping function of the product of the two aforementioned quantities is compared to the unclipped product, and the minimum value is chosen (Ackermann et al., 2021). This method ensures policy changes are slow, stable and beneficial for the agent. Being one of the state-of-the-art (SOTA) algorithms, I anticipate it will become more common in the field.

2.2.6 Other RL considerations and hyperparameters

RL is known for its instability and the vast number of finicky hyperparameters used. The length of the transition memory which the agent randomly samples during training has an effect on the stability. The target network, which determines labels when training the policy network, is 'frozen' and lags behind usually a fixed number of steps or episodes. Epsilon, ε, the percent of the time a random action is taken is also an important factor. It is common to decrease this over time to allow the agent to slowly become more independent as it learns. The discount factor, γ, which signifies how important immediate rewards to future rewards is typically set to high values (generally 0.9 to 0.99) in medical imaging RL. Guided exploration is used too (Caicedo & Lazebnik, 2015). The agent is told which action to perform and is only allowed to take actions that result in a positive reward. This is done only a fraction of ε percent of the time (part of the random exploration becomes guided) and is usually reduced over training.

I use several other hyperparameters in my experiments. Given that the agent tries to move a bounding box to the proper location, I use α (a value between 0 and 1) to determine how much to move or scale the box, where the amount to move or scale equals α times the minimum side length of the box. Please see the appendix for additional details on hyperparameters.

3 Related work

3.1 Reinforcement learning outside of medical imaging

Watkins's invention of Q-learning in 1989 and subsequent work in 1992 provide the basis for many papers to come that use this model-free, off-policy agent to approximate solutions to Markov Decision Processes (Watkins 1989; Watkins & Dayan 1992). Over 20 years later, Google DeepMind expanded upon the algorithm. They introduce the Deep Q-Network (DQN) in the mid 2010's, and a deep CNN is used as a nonlinear approximator for the Q-function, which maps states to action-values (Mnih et al., 2015). They train models (of the same architecture and hyperparameters) to play 49 Atari games and introduce an experience replay buffer to combat correlations between sequential observations and the update-delayed target network. This decreases the connection between the targets and predicted Q-values. Their models are overwhelmingly successful in the majority of games, substantially outplaying even professional game-testers (Mnih et al., 2015). This DQN design is the basis of many future works relevant to medical imaging's landmark detection, pathology detection, and view-planning tasks. It is also the basis for a highly influential paper in non-medical image object detection by Caicedo and Lazebnik (2015).

In 2015, Caicedo and Lazebnik lay the foundation for many landmark detection papers in medical imaging. They use a DQN to identify objects in images. It has two other notable contributions: the reward structure and guided exploration. For the reward, they use the sign of the change in Euclidean distance to a landmark (rather than the change itself) due to shrinking rewards as the agent approaches the ground truth. Additionally, they use guided exploration (where some percentage of the training time the agent is guided to take a certain action in order to learn the path to get to the goal), to speed up training. Their use of a pre-trained CNN is mimicked in other future works (Maicas et al., 2017).

Caicedo and Lazebnik's paper has some notable differences that are not used in later papers in the medical image domain. The trigger action to signal that the object (the landmark or pathology) is found is only used in Maicas et al. (2017). However, Maicas et al. do not continue search after the trigger action as Caicedo and Lazebnik do (in their search for multiple objects). This search-after-triggering method seems to be a relatively simply approach that could work in multi-landmark or multi-pathology detection, and it could be far less complicated than a multi-agent RL model. In addition, their state includes the last 10 actions taken. This provides trajectory stability, while not overly complicating the state representation (such as by using additional frames).

They also discuss the potential downsides of setting the IoU threshold too high – occluded or partial objects might be missed, and mention the difficulties in detecting small objects, which can be quite relevant to the medical domain. Sometimes pathologies or landmarks appear on the edges of images, and both landmarks and pathologies can appear different across patients. Finally, the most substantial difference that seems lost to the other relevant medical RL papers is the use of an external classifier. Caicedo & Lazebnik (2015) use a linear SVM for two modes of localization: examining stopping states and examining all reached states. Most medical imaging papers in RL simply look for oscillation between neighboring states and either pick the average position of them or look for the one with a higher or lower Q-value (Alansary et al., 2017; Ghesu et al., 2019; Navarro et al., 2020; Riedmiller, 1998).

3.2 Reinforcement learning in medical imaging

3.2.1 Early works

Even by the early 2000s, Q-learning makes its way into image processing. Most of these tasks are thresholding-determinations such as in Shokri's and Tizhoosh's 2003 work. Soon after, it makes additional appearances in the medical imaging community. Besides thresholding it is also used for determining the size of filters in morphological changes such as opening, which is one step of segmenting Ultrasound (US) images of the prostate (Sahba et al., 2006). These early works lay the foundations for future work to come. Fig. 6 shows many of the medical imaging works that use RL.

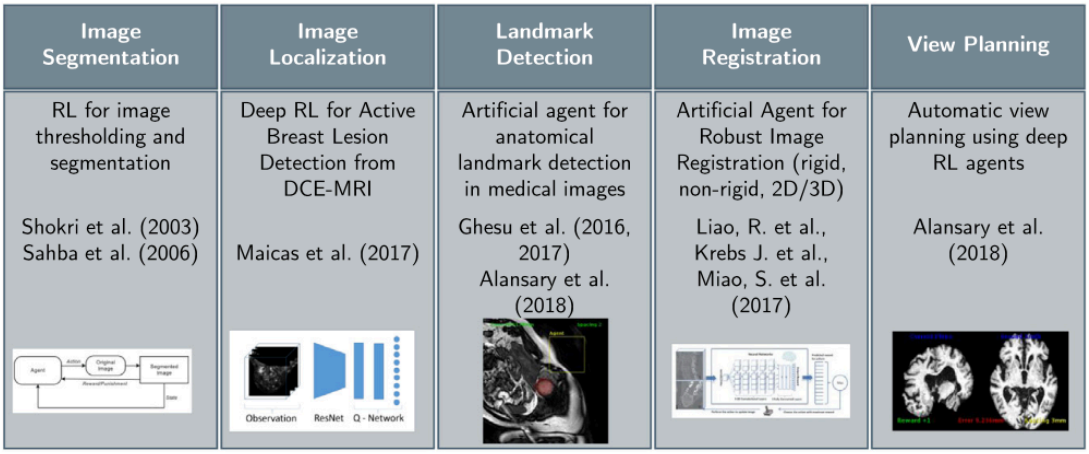

Fig. 6. Some early (and later) works in RL for medical imaging. Reprinted from Alansary et al. (2019).

3.2.2 Landmark localization

The paper by Ghesu et al. (2016) is one of the earliest to use reinforcement learning in landmark detection for medical imaging. The paper focuses on landmarks in three different scan types: MRI, Computed Tomography (CT) and US. They use a larger dataset compared to other RL in medical imaging problems: 700 MRI, 1000 cardiac US and 350 CT scans in three separate problems. A small network of just three convolutional layers and three FC layers is implemented in a DDQN fashion. An impressive, 1 x 10^5 transitions are tracked for their experience replay memory. With four translational actions (up, down, left, right) the agent navigates to the ground truth. They identify when the landmark is not in the image by looking at the Q-values during oscillation. They achieve results consistent with SOTA techniques at faster speeds (80 times faster in 2D, and 3100 times faster in 3D).

Extending their work from the prior year, they introduce a coarse-to-fine scaling approach to achieve impressive results in a 2017 paper (Ghesu et al., 2017). They use a similar architecture and hyperparameters as the earlier work (specifically, a DDQN with two convolutional layers and three FC layers). Six translational actions that modify a 25 x 25 x 25 voxel ROI are learned, and a reward metric of the improvement in Euclidean distance to the ground truth is unchanged. This time however, the agent 'looks' at different resolutions of the scans. After determining a convergence point in the coarsest (16mm) CT scan, they double the resolution and continue the process. They examine four different levels of granularity and using shape-modeling, determine the final location of the landmark. In each level of granularity, they set the starting ROI bounding volume to the location of where it stopped in the last scale.

Stopping is again determined by the midpoint of the oscillation. Their dataset is 855 patients with 1887 3D CT volumes. Impressively, they detect volumes in under 40 milliseconds on average and achieve a 0% false positive and a 0% false negative rate for detecting if the landmark is in the scan or not. For localization, they boast improvements of 20-30% to all landmarks (in terms of distance) with exception to the kidney.

Although quite similar to the 2017 paper, their 2019 paper has several additions (Ghesu et al., 2019). One is that they perform outlier analysis on their data and others' works: specifying outliers by a distance of 10 mm from the ground truth for all organs, with exception to the kidney, where 30 mm is used instead. In addition, they are able to substantially decrease training time by capping the number of steps per episode. They anneal the step limit per episode from 1000 to 50 over training, postulating that convergence happens faster late in training and that increasing the number of unique trajectories in the memory is more beneficial than having a small number of longer trajectories. It is noteworthy that they still maintain a rather large memory size of 1×10^5 transitions and implement a rather small network of only two convolutional and four FC layers. Like the 2017 paper, the model is explicitly trained on instances where the landmark is not present. Their results are superior in both distance and computation time relative to other SOTA methods.

Alansary et al. (2019) explore three datasets (fetal US, brain MRI and cardiac MRI) and use four deep Q-learning variants in both a fixed-scale and multi-scale approach. The state is limited to the voxels within a 45 x 45 x 45 cube over the last four frames (to give stability in the trajectories), with the actions being six translational ones. Rewards are the change in Euclidean distance to the landmark. The network is relatively small consisting of four convolutional and four FC layers. The episodes start in a random spot centered within 80% of the entire volume and are stopped during training when the volume gets to within one millimeter of the landmark (or 1500 steps are taken). During testing, oscillation is used instead. They use a memory of 1×10^4 transitions and a decreasing ε of 0.9 to 0.1.

Despite the theorization of better separation of state and action values with duel DQN and duel DDQN, for only one experiment and one landmark did a duel DDQN perform the best in Alansary et al. (2019). In the majority of the experiments, DDQN or just vanilla DQN perform the best. In all cases however, it is a multi-scale implementation that is superior. This further emphasizes how scaling greatly helps the model learn the environment from coarse-to-fine grain.

Their multi-scale approach beats SOTA methods in the US and cardiac MRI datasets, it performs inferior on the brain MRI dataset however. Although they brag of 4-5 times faster during inference mode, training takes between one-two days on average per landmark. In addition, they point out that if the volume is initialized in the air around the image then it sometimes does not find the landmark at all. This problem can be pertinent as many of the backgrounds in medical imaging contain large swaths of meaningless pixels.

Expanding on previous work, the work by Vlontzos et al., (2019) uses a DQN to find landmarks. Their method uses multiple agents that can communicate – thus saving computational time and decreasing detection errors at a rate of 50%. Their network only uses three convolutional layers and three FC layers, where each agent has their own FC layers in parallel, while the convolutional layers are shared so agents can work together to learn optimal parameters. Episodes are terminated for an agent during training when the proposed volume reaches within 1 mm of the ground truth. During inference, episodes are stopped at a set number of steps, or when the oscillation criterion is met. They use three datasets to train and test on: approximately 800 brain MRIs, 450 cardiac MRIs and 70 fetal brain USs. Nineteen different

starting volume locations are used with their proposed stopping locations averaged. With the exception of one landmark, they beat the SOTA methods (by Euclidean distance to the ground truth). They also reduce the number of steps needed to reach the landmarks by 25-50 time steps.

Kasseroller et al. (2021) are one of the first to use a continuous action space in landmark localization. They do this through the use of an AC network, where the actor network chooses to move an agent or multiple agents (in the multi-agent case) in any direction in 2D space up to fifty pixels, and the critic network predicts the Q-value of this action. In the multi-agent, multi-landmark case, a parallel convolutional path is used for each agent, but they all share the same FC layers. This is exactly opposite of the multi-agent network of Vlontzos et al. (2019). To the first FC layer (in both actor and critic networks), the distances between each pair of agents is inputted. The terminal state determination for each landmark is made when the actor chooses to move an agent by less than a single pixel.

They train and test on nearly 900 hand radiographs, in a 80:20 split. Training is extensive: 30k episodes over 10 days. Despite a reduction in the number of steps it takes to get to the landmark, their error is higher than other works, including those by Payer et al. (2016) who use spatial configuration networks, which are similar to the action heatmaps from several works: Li et al. (2018), Noothout et al. (2018) and Xu et al. (2017).

3.2.3 Organ localization

Navarro et al. (2020) use a deep RL agent to learn organ localization (for CT volumes) on the VISCERAL dataset. They achieve a mean IoU score of 0.63, a median wall distance of 2.25 mm and a median distance between centroids of 3.65 mm. Their agent moves inside a three-dimensional environment and can translate along either of the three axes. In addition, it can zoom in or zoom out, or make the volume thinner, flatter or taller. The Q-values of the eleven actions are determined by a surprisingly small network, consisting of three convolutional layers and four FC layers. They utilize the same architecture as Alansary et al. (2019) and implement a DDQN. In their transitions, they store a' in addition to the usual s, s' and r. This seems odd, as this means these experience replay memories are outdated (with potentially old a' values) and thus the target network's calculation of Q(s', a') is affected. Their state is defined as the last four volume positions, similar to Alansary et al., and it is isotropically resampled to 1 cubic millimeter consisting of 45 x 45 x 45 voxels. They follow Caicedo's and Lazebnik's (2015) recommendation to not use a change (in Euclidean distance or IoU) in itself as the reward due to shrinking IoU deltas and rewards when approaching the goal destination. Instead Navarro et al. reward the agent +1 when moving towards the goal state and -1 if not. It is important to note that this reward is non-Markovian as it depends on both the current state and the penultimate one. They stick with the standard approach of terminating an inference episode when oscillation is reached. Training episodes are terminated when the threshold is reached. Fig. 7 shows some of their results visually.

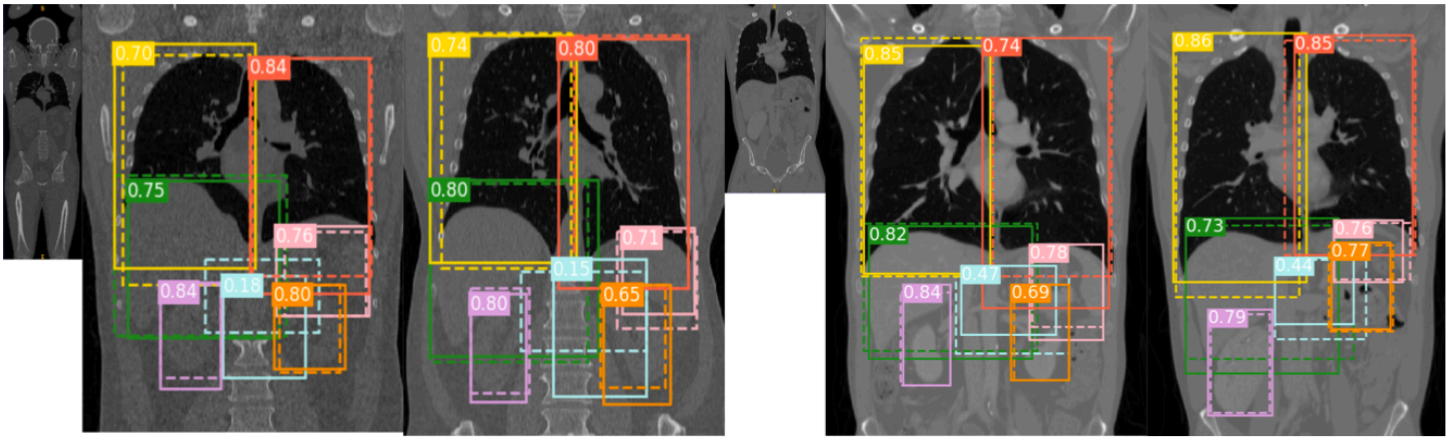

Fig. 7. Results of various organs segmented using a DDQN**.** Dashed lines show ground truths, while solid ones show predictions. Reprinted from Navarro et al. (2020).

3.2.4 Pathology detection and localization

Similar to Navarro et al. (2020), other papers also change the volume size to avert the need for multi-scaling approach. Maicas et al. (2017) use a DQN to segment breast lesions. With an original dataset of 117 patients of 512 x 512 x 128 x 4, 4D Dynamic Contrast-Enhanced MRIs, the training and testing sets consist of 58 patients with 72 lesions, and 59 patients with 69 lesions, respectively. The fourth dimension (time) consists of one volume before contrast is applied and three after. The initial pre-contrast volume is subtracted from the first post contrast volume to eliminate this fourth dimension. Like Ghesu et al. (2016), the state is the voxel intensities within the volume. However, the actions are different. Maicas et al. add in a trigger action that the agent can use to stop an episode and indicate that a terminal state has been reached. They also add in two scaling actions (by $1/6^{th}$ of the volume) which do not change the aspect ratio. The translational actions move the volume by $1/3^{rd}$ of the its size (which is more compared to Ghesu et al.'s translations which moved the volume by only one voxel). Rewards are +1 or - 1 depending if the volume moves closer to the pathology or not. The trigger action, if selected, results in a +10 or -10 reward for the agent based on if the IoU of the volume met or exceeded 0.2 (in comparison with the ground truth volume). The agent network consists of five residual blocks (ResNet) that is trained on 8k positive and 8k negative volumes from the training dataset, where positive volumes have an IoU of 0.2 or more (and negatives do not).

Training occurs over 300 epochs with a learning rate (LR) of $1x\ 10^{-6}$, Adam optimization and an ε annealed from 1 to 0.1, with a 50% guided exploration rate of ε. By setting guided exploration to only part of ε, they modify Caicedo's and Lazebnik's 2015 guided exploration implementation to include some random exploration. During inference, the agent selects the trigger action to indicate to stop. Maicas et al. (2017) achieve a result of 0.8 True Positive Rate per 3.2 False Positive Rate and an average time of 92 +/- 21 seconds to find the lesion.

The paper by Stember & Shalu, (2021) uses RL to determine which slices of T1 MRI Brain slices (from the 2020 BraTS Challenge) contain gliomas (tumors). They use the entire slice as the state and overlay it with a green or red filter based on if the last state is predicted correctly by the agent. Actions for the agent are simply to identify the slice as positive or negative for one or more gliomas. Each episode (30 for each train and test) consists of five steps,

and the agent is either rewarded (+1) or punished (-1) based on the action it takes. Out of the 30 images in the train and test set, each contains 15 positives and 15 negatives. They compare their results (100% accuracy with RL) to 57% accuracy of supervised learning.

In 2022, Stember and Shalu extend their previous work from the year before by including a large language model (LLM) to extract labels from radiological reports and then classifying T1 post-contrast brain volumes as pathological or not (Stember & Shalu, 2022). They select 64 brain volumes, 32 normal (negatives) and 32 positives. Similar to their previous work, they use the entire image. They pass this volume along with the network's correctness boolean of its previous prediction into a 3D CNN. The previous prediction (of the prior volume) correctness value (0 or 1) is concatenated to the penultimate FC layer and rewards are allocated based on if the agent classifies a volume correctly. The agent's two actions are to determine a slice as normal (0) or not (1). Each episode consists of five steps, and 2000 episodes are used. Like before and in many papers, a (linearly) decreasing ε is annealed, from 0.7 to 1 x 10^{-4}. They also add salt and pepper noise (to 2.5-5% of the image) as they found previously that this can help the agent learn better. They achieve impressive results of 100% accuracy with RL and only 50% for a baseline supervised learning model that utilizes the same architecture and is trained for 100 epochs.

Pesce et al. (2019) employ two different methods to detect pulmonary lesions. Pulmonary lesions can be difficult to identify due to their small size and because other abnormalities in the lungs usually exist comorbidly. In the first part of the work, they utilize a convolutional neural network with attention feedback (CONAF) to identify lesions. This network is trained on a large dataset of 430,067 radiographs of the chest. The scans are trifurcated into "healthy", "lesion", and "other" (pathological, but without lesions) classes. After, 2,196 scans from the lesion class are annotated (localized) by an experienced radiologist and the remaining are left only with their class labels. The convolutional part of CONAF is similar to the VGG-13 network. The model both localizes and classifies the image. In the cases of the scans with annotations, a MAE loss is calculated from the ground truth and added to a binary cross entropy (BCE) loss (in equal proportions). In the other cases where annotations are not available, only the classification part of the loss is used. Fig. 8 shows the CONAF architecture.

RAMAF (recurrent attention model with attention feedback) is the other method applied to locate lesions by Pesce et al. RAMAF use a dual glance system (two patches of 70 x 70 and 140 x 140 pixels) with the same central point. After a CNN encoder and a LSTM, a locator determines in which way the agent should move. This agent trains using the REINFORCE algorithm. A train, validate, test split of 80:10:10 is used for both models. CONAF outperforms other SOTA models in all classification metrics, with the exception of precision (for both lesion versus other, and lesion versus healthy paradigms). RAMAF successfully segments 82% of lesions (when counting a hit if there is at least 25% overlap with the ground truth ROI).

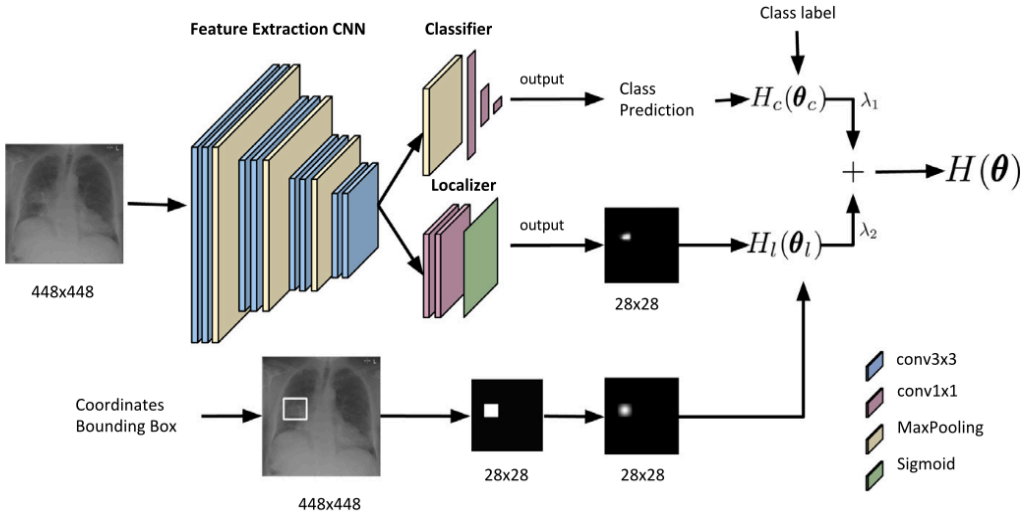

Fig. 8. CONAF architecture**.** Reprinted from Pesce et al. (2019).

3.2.5 Surgical and radiation planning

Ackermann et al. (2021) utilize one of the newest SOTA implementations of deep reinforcement learning, PPO to plan cutting-planes in surgery that correct femoral osteotomies. The complexity of the proscribed surgical plan is notable as there are five key criteria that need to be met. In addition, the environment is a noisy three dimensional CT scan. The agent tracks the advantage function (which is the sum of the discounted rewards from a given state and action), in addition to the policy ratio which keeps track of how the policy has changed since last iteration. Two surgeons participate in the work, to give their professional opinion. Based on real data from eight patients, the agent formulates solutions at or above the human-level for 80% of the patients (in the opinion of one surgeon) and for 100% of patients (in the other surgeon's opinion).

Gao et al., 2024 in a very recent paper published in June, use a multi-agent PPO model to guide radiation sequences in oncology treatments. Their method beat SOTA techniques in both dose score and dose-volume histogram score. Florin Ghesu, who published the earlier series of papers in landmark localization, is the second author on this paper.

3.3 Supervised learning outside of medical imaging

He et al. (2015) introduce ResNet architectures that are very deep and use residual connections. Their networks won them the famed ImageNet competition in 2015 with an impressive 3.57% top-5 error on the test set.

Although vanishing and exploding gradients are minimized by batch normalization layers (and the normalized initialization of weights), training accuracy can degrade as additional layers are added to an ANN. Even more so, a more complex, larger model without residual connections can actually have higher training losses than smaller models, even if the larger model is essentially a superset of the smaller one. He et al. (2015) work to mitigate this problem by the use of residual (also called skip) connections. In their 2015 work, their use of skip connections in the ResNet models lets a layer be the combination of the last layer(s) plus the skip connection (an identity mapping of a past layer). Multiple nonlinear layers might have trouble creating

identity mappings by typical optimization methods. They also find that the larger networks (of 50, 101 or 152 layers) perform better than the ResNet-34 model. The larger number of nodes in the flatten layer (2048 versus 512) proves better at representing the internal states of the inputs.

Ronneberger et al. introduce another important model in 2015 called U-net. U-net has become wildly popular for its use in segmentation and localization. Fig. 9 shows the U-net architecture. As an image-to-image network, U-net reduces an image to a small latent space and then uses upsampling to convert these select features into the desired output. It uses skip connections to help with identity mappings similar to ResNet. Despite its successes, it shares the archetypal downside of many supervised learners as it requires a hefty dataset for training. In addition, its output images are sometimes quite large to temporarily store on the GPU, and can demand large amounts of memory and lengthen the training time.

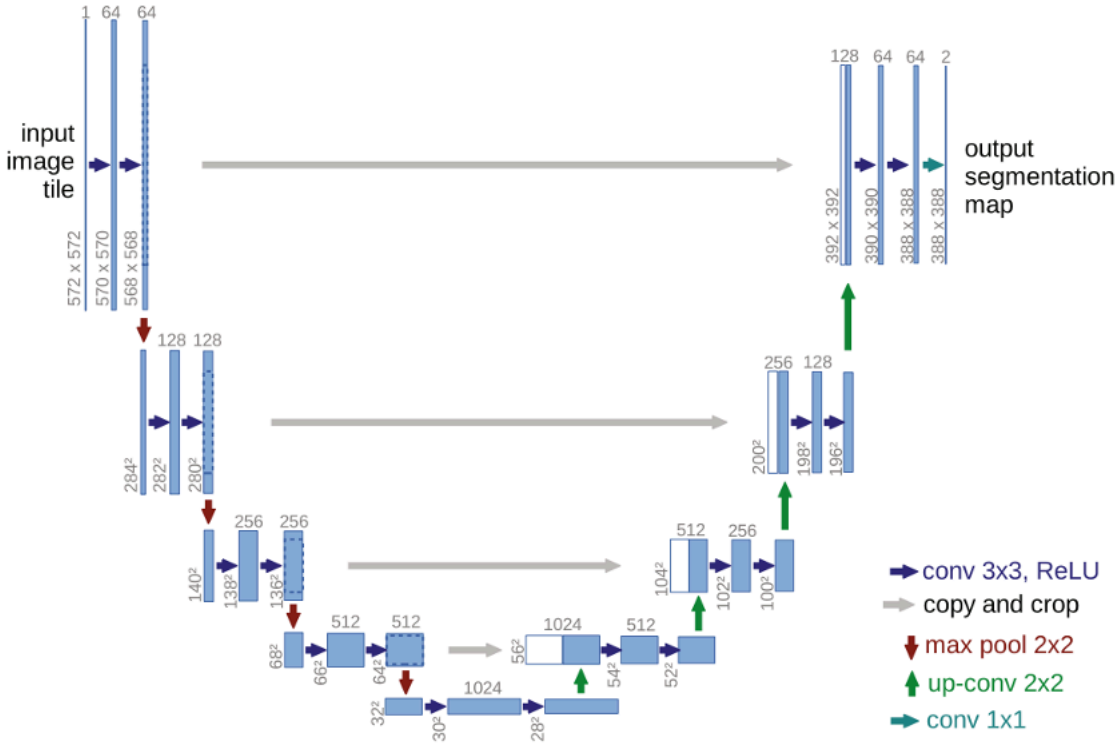

Fig. 9. The U-net architecture. Reprinted from Ronneberger et al. (2015).

Inspired by Meta's segment anything model, the paper by Ma et al., (2024) uses a version of the model specifically for medical imaging (Kirillov et al., 2023). Given a bounding box, it can detect pathologies, landmarks and other structures inside it. Trained on over 1.5M images with their respective masks, MedSAM is robust and can handle diverse structures occurring in the human body (Ma et al., 2024). It uses SOTA visual transformers (ViTs) in the encoding and embedding process for the input images. Further work extends MedSAM to localize bones in MRIs, using the Segment Any Bone model (Gu et al., 2024). I use the Segment Any Bone model to prepare masks of the calcaneus bone for experiments 8-10.

In regards to transfer learning, Alzubaidi et al. describe some of the problems with it in their 2021 work. Transfer learning is often unsuccessful between medical and natural images because models are originally trained on different shapes, colors and textures than in medical imaging (Alzubaidi et al. 2021). Training a network from scratch however requires large labeled

datasets which are often difficult to acquire. In their work, they assign random labels to different lesions in a large but unlabeled dataset, emphasizing that classification is not important in early training stage, but only that the lower convolutional layers learn to extract relevant features. Later on after the convolutional layers have been trained, the downstream FC layers are trained for classification on a small labeled dataset. They find this technique of training the network in sections to be highly effective. Although their work is on skin lesions, this technique could be generalized to various medical imaging domains.

3.4 Supervised learning in medical imaging

3.4.1 Landmark localization

Inspired by deep RL methods, several researchers explore supervised learning approaches to create action maps. These vector (directional) fields indicate which way (up, down left or right) the landmark is from the pixel or image patch. Payer et al. (2016) use regressing heatmaps also called spatial configuration networks (SCNs) which perform the same task.

Xu et al. (2017) are the first to generate a complete action map of the entire search space, similar to the internal one represented by a DQN. They quickly point out potential flaws in deep Q-learning: sometimes insufficiently trained agents traverse inferior paths that do not lead to the landmark when they are initialized in less sampled parts of the image. Especially in the case when there is less variation in the training images, it is quite possible that the agent's path never converges to the landmark.

Xu et al. (2017) avert these issues by generating an exhaustive action map, where every pixel is assigned a directional class (action). This is done by using a fully convolutional network. The general method is shown in Fig. 10. The convolutional part of VGG-16 is used with its pre-trained weights from ImageNet and then upsampling convolutions are placed downstream. This results in an architecture quite similar to U-net, sans the skip connections. After this image-to-image mapping is complete, pixels of the same directional class that share the same row or column are grouped together. This helps to eliminate some of the noise and misclassified pixels in the image. Their results on two US datasets (obstetric and cardiac) are superior to both RL and patch-wise binary classification methods.

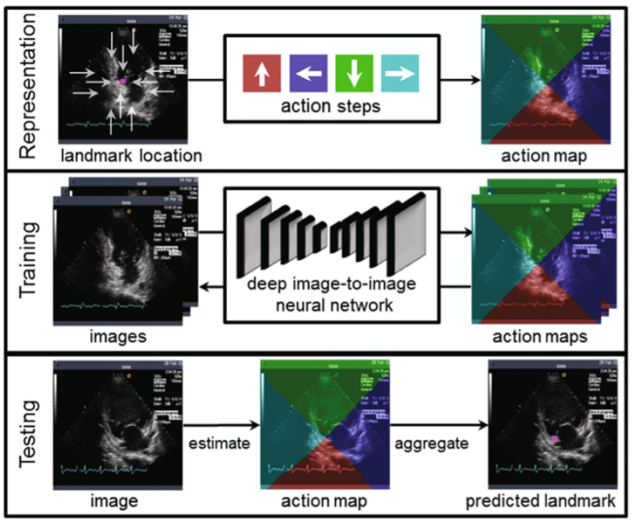

Fig. 10. Action map for landmark detection. A mapping is made by assigning an action to every pixel. This can also be done patch-wise. Reprinted from Xu et al. (2017).

Li et al. (2018) expand on the work of Xu et al. (2017) and Ghesu et al. (2016) by classifying image patches into six classes (by the direction in which the volume needs to move to get to the ground truth landmark). In addition, they implement a regression task to predict how far the volume needs to move. By taking the product of these predictions (the directional label and the regression distance) for the highest class, Li et al. receives the best results. Their Patch Iterative Network (PIN) methodology outperforms previous RL localization results by Ghesu et al. in terms of accuracy. This use of sparse sampling, rather than via exhaustive searching means less time to locate the landmark compared to both Xu et al. (2017) and Ghesu et al. (2016).

Li et al. (2018) also use Principal Component Analysis to reduce the dimensional space in multi-landmark localization. All of their work is in 3D, where three orthogonal planes are assigned to three input channels of the CNN (in a 2.5D fashion). This allows 2D convolutions to be used, which are significantly faster than their 3D counterparts. The architecture consists of five 2D convolutional layers followed by two branches of three FC layers. Single landmark detection uses 19 initializations averaged to find a final proposed spot, while the multi-landmark prediction uses six. Their loss function combines cross entropy and mean squared error with a 50:50 ratio. Training on 70% of 72 3D fetal head US and testing on the reaming volumes yields a single landmark identification result of 5.47 +/- 4.23 mm. Multi-landmark prediction (on ten landmarks) yields a result of 5.59 +/- 3.09 mm.

Noothout et al. (2018) utilize methods that are close to both Xu et al. (2017) and Li et al. (2018). Similar to Xu et al., an entire volume is fed into a network, but like Li et al., two tasks are implemented to determine where the landmark is located. Again, regression and classification are applied. However, to actually find the landmark, only the regression results of patches that contain the landmark are used. This is similar to the idea of Hough Forests in (Donner et al., 2013). Also, the model is trained to predict the log of the distance to the landmark. This ensures patches that are far from the landmark and likely less accurate will not have an unduly large effect on the model. Their 72 x 72 x 72 pixel patches are computationally heavy as they are passed through a six layer convolutional network. After these layers, the network splits into a regression part and classification part that uses 1 x 1 x 1 convolutions. Sixty-thousand patches are trained via Adam optimization using a combination of BCE and MAE loss functions. Out of the 198 CCTA scans, only 8 are used for validation and 40 for testing, while the other 150 are for training. Of the six landmarks, the average coronary ostia error is a mere 2.54 mm, and only 1.94 mm for the three aortic valve commissures.

3.4.2 Edema detection

Four fully-supervised models (U-net, Unet++, HRNet, Attention-Unet) are trained end-to-end on 1945 STIR/ T2 MRI slices of the hip in five-fold cross-validation (Zheng et al., 2023). They all yield similar results, with 89% accuracy in segmenting the femoral head and 69% accuracy for inflammatory lesions.

Deep learning models have the advantage being able to more objectively determine the presence of edema or other pathologies, are cheaper than training additional high-level clinical specialists and can be used in large-scale screening efficiently (Zheng et al., 2023). The U-net model is most thoroughly experimented with, and they find U-net scores highly correlated with manual MRI evaluations from radiologists in BME score, synovitis score and HIMRISS score. Overall, this shows the potential this model has to play a key role in early diagnosis and prognosis in hip inflammation.

Ribeiro et al. (2023) look at edema in the knees of adolescents in a highly insightful paper. They point out that MRIs are often used as the next imaging modality for clinicians when radiographs are inconclusive. In addition, they note that bone marrow edema can be associated with pathologies, but that it can also be naturally occurring during growth in youth. Also, the heterogeneity in imaging BME, makes it easy to miss. Transfer learning is used in their work and they examine the ROIs of bone segmentation and intensity masks.

Their dataset of 36 non-edema patients and 28 edema patients has three planes (coronal, axial and sagittal), and a radiologist annotated the images in the coronal plane. The femur, tibia, along with BME and a reference intensity circle in the quadriceps muscle are drawn in. Overall, there are 228 slices with edema and 121 without. Given the scarcity of data, it is important that the model only sees the relevant parts of an image, with as little noise and extraneous information as possible. They propose intensity-based masking as a method to do this as described below. Preprocessing consists of segmentation followed by normalization. The annotations from clinicians are used to segment the bones. Then, each image is normalized by dividing by the maximum pixel intensity of the image. High intensity masking is performed (thresholding) because the edema occurs at higher intensities in the bone. These modified images are passed into Resnet-18. Some layers are frozen, and they progressively unfreeze them over the course of training. They also use data augmentation. Training yields a balanced accuracy of 0.852 and an AUC of 0.964. Overall, on the test set (via a five-fold split, an average balanced accuracy of 0.792 is reported for edema positive patients (who still have some negative slices) and 0.672 for all patients slices.

3.4.3 Ankle and foot fracture

In this study, Kitamura et al. (2019) use 596 ankle radiographs and train the following models in single and multi-view: Inception V3, Resnet, Xception. They achieve 81% accuracy by using three views (frontal, oblique and lateral) on each model in a 5-network ensemble. The Resnet and Xception networks each have two models (one normal and one with an auxiliary tower and dropout). Auxiliary towers compute additional losses and branch off from the main network to help provide additional learning and faster convergence. In addition, PPV, NPV, sensitivity and specificity all exceed 80%.

A different work by Rahmaniar & Wang (2019) identify calcaneus fractures in CT images. Impressively, they are able to identify the type of fracture (using Sanders classification). They use a series of image processing techniques such as connected components, dilation and erosion, histogram equalization, contour detection, in addition to AdaBoost. It is notable that they do not use a CNN in the process and their methods work for coronal, sagittal and axial views. Their recall of 0.89, precision of 0.86 and F1 score of 0.88 are comparable with SOTA methods. Fig. 11 shows their methodology for determining calcaneus fracture types.

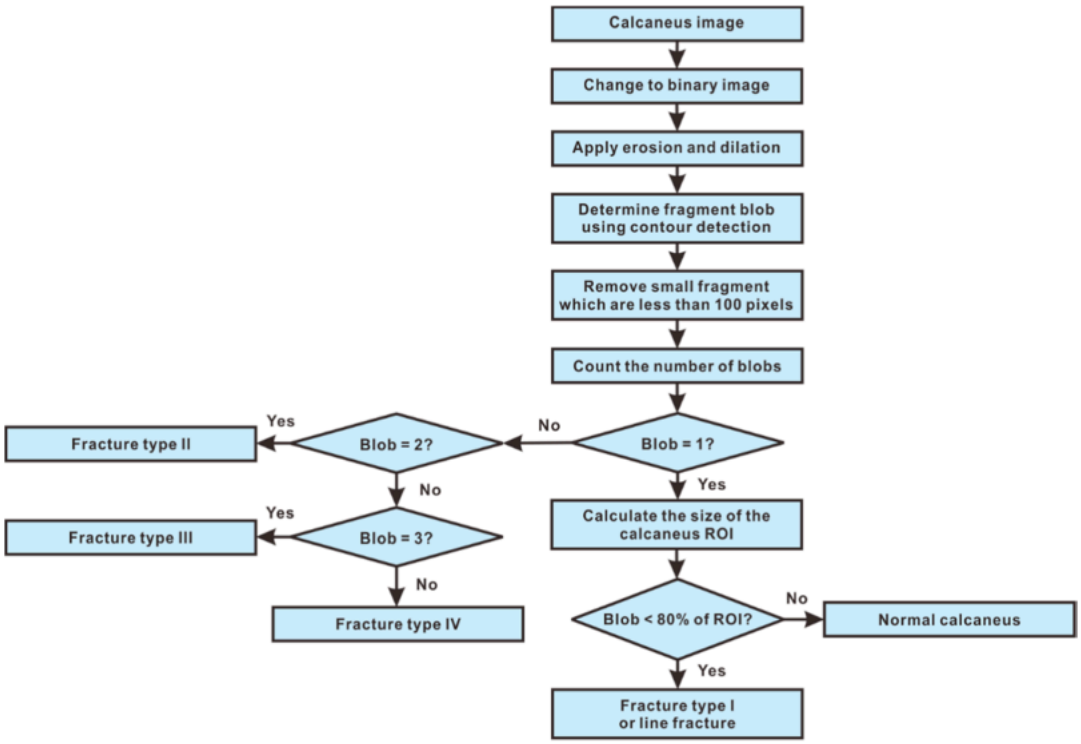

Fig. 11. Fracture classification flowchart. Reprinted from Rahmaniar & Wang (2019).

4 Methods

4.1 Topic selection

I opt for RL as my thesis topic because of my interest in it, and its successes in related problems in medical imaging (Alansary et al., 2019; Ghesu et al., 2019; Navarro et al., 2020). Recent papers tackle various problems, such as slice-wise pathology detection, landmark identification, or pathology segmentation (Maicas et al., 2017; J. Stember & Shalu, 2021; Vlontzos et al., 2019). I examine pathology segmentation via deep Q-learning because it is most appropriate to my labeled dataset and not complex to implement. Some architectures in the literature are very technical, such as RAMAF from Pesce et al., (2019) or surgical planning for the hip via PPO as done by Ackermann et al. (2021). I use Navarro et al. (2020) as an example paper to follow for methodology and hyperparameter values, while the DQN tutorial by PyTorch for the actual Q-learning implementation (PyTorch, 2024).

4.2 Dataset

The dataset consists of 115 adult patients who experienced foot or ankle pain. Imaging of the foot and ankle regions is taken at the Captain James A. Lovell Federal Health Care VA Center located in North Chicago, IL. In two MRI scan types, sagittal T1 and sagittal IR, radiologist students (overseen by an experienced radiologist) segment regions of edema and fracture. Two MRI machines are used, one of 1.5T and another of 3T. The slice thickness, field of view, along with relative times (such as repetition and echo) are unknown to the author. The matrix size of the SAGIR images varies, but the most common sizes are 256 x 256 pixels and 512 x 512 pixels. The most prevalent matrix size for the SAGT1 images is 864 x 864 pixels.

4.3 Experiments

I perform three sets of experiments: segmentation of edema in SAGIR images via a DQN (Experiments 1-7), training ResNet models for calcaneus detection using supervised learning (Experiments A-D) and localization of the calcaneus bone in sagittal T1 images via a DDQN (Experiments 8-10).

4.3.1 Markov Decision Process

I define the MDP for the RL experiments (1-10) as below. Please see the appendix for a more detailed explanation of hyperparameters.

4.3.1.1 The state space

The state is defined in several different ways:

1. A 2D slice of the MRI with a drawn-in bounding box of where the agent is in the image (essentially the slice and the agent's current position). I call this slice mode.
2. The pixels inside the agent's proposed box. Everything outside of the box is made black. I call this pad mode as the box is padded.
3. The fit mode is when the agent's proposed box is zoomed in on. This was done isotropically to fit my network. This method is nearly exclusively used in related works (Alansary et al., 2019; Maicas et al., 2017; Navarro et al., 2020). The position of the agent within the slice is thus not readily available to the agent.
4. Any combination of the above.
5. Any of 1. 2. or 3. with various frame lengths (previous positions) to help the agent understand directionality.

Passing in an entire slice with a proposed box begs the question of why to use RL in the first place as any supervised model should be able to tackle the task in this manner. RL is used because of its more efficient search and lower memory requirements. Slice mode alone is less appropriate.

4.3.1.2 The action space

Actions are defined in three ways:

1. Four translational actions (up, down, left, right).
2. Four translational actions in addition to a zoom in, widen the box, and make the box taller, totaling 7 actions.
3. All the aforementioned actions in addition to zoom out, make the box more narrow and make the box shorter, totaling 10 actions.

4.3.1.3 Rewards

There are three main reward schemas used in the shown experiments:

1. The agent receives a positive reward when the IoU increases. The agent receives a negative reward if not.
2. The agent receives a positive reward when the Euclidean distance is reduced between the ground truth box center and the proposed bounding box center. The agent receives a negative reward if not.
3. The combination of both of the above with priority going to IoU and then resorting to Euclidean distance when IoU is unchanged.

4.3.2 The agent

The agent consists of two identical networks (a target and a policy network) consisting of CNN followed by a Q-network as described below.

4.3.2.1 The network

I train a deep Q-network to identify the goal state (edema or the calcaneus bone) and the path to it. A policy network uses an ε-greedy policy to navigate the environment and collect transitions. I save these transitions and utilize them in an experience replay fashion where minibatches are used to train the policy network after every network step. The random selection of transitions helps the network to overcome the inherent codependence of the transitions (as neural network convergence and generalization typically rely on independent samples taken from a distribution). A lagging target network is updated after every C steps and its weights are taken from the policy network based on τ (the fraction of policy weights used to update the target network). I use the DDQN architecture to separate the selected a' from its Q-value in the calcaneus segmentation experiments (8-10) which helps to provide stability.

Usually $\max_{a'}Q(s', a')$ changes as the target network updates. Recall that the target network provides labels for the policy network, however as the weights of the target network change, so does its output (for a given Q-value). Although the target network updates slower than the policy network, its changes still cause a moving-target effect. The true converged (and stationary) Q-values can be computed however in the case that I only use translational actions. We can calculate how many steps the agent will take minimally to reach the goal state. The agent moves α x *box size* pixels per step, and the Euclidean distance between the current agent's box and the ground truth box is known during training. Since the minimum number of steps can be computed, and the reward it will get for each (correct) step is a fixed constant, the discounted rewards can be summed as follows if there is a minimum of *n* steps to the terminal state (see Equation 8).

Calculating the target values for the subsequent state and action

$$max_{a'}Q(s', a') = r\gamma^0 + r\gamma^1 + r\gamma^2 \ldots + r\gamma^{n-1} \qquad (8)$$

Recalling the BOE (from Equation 3) we arrive at Equation 9, where *r* is the positive reward. Then using the sum of a finite geometric series we arrive at Equation 10. Again, *r* is the reward from taking the best action (a positive reward). Equation 10 can then be assigned as the target or label for the policy network for a given Q-value.

The target value for a state

$$Q(s, a) = transition\ reward + \gamma(r\gamma^0 + r\gamma^1 + r\gamma^2 \ldots + r\gamma^{n-1}) \qquad (9)$$

The target value for a state (reduced)

$$Q(s, a) = transition\ reward + \gamma r\left(\frac{1-\gamma^n}{1-\gamma}\right) \qquad (10)$$

4.3.2.2 Network architecture

My architecture varies with the experiments. The architecture for experiments 1-2 is shown below in Fig. 12. In experiments 3-5, I replace the 4x4 filter by a 3x3 filter. I use ReLU between convolutional layers along with max pooling layers of size 2x2 (with stride 2). I use leaky ReLU activations between the FC layers. The input size varies between, 64, 128 or 256

pixels. The number of output nodes varies with the number of allowed actions. For experiment 1, there are ten actions, for experiments 2-5 there are seven. Fig. 13 shows the architecture for experiments 6-10. These experiments use ResNet for a feature extractor and have four actions.

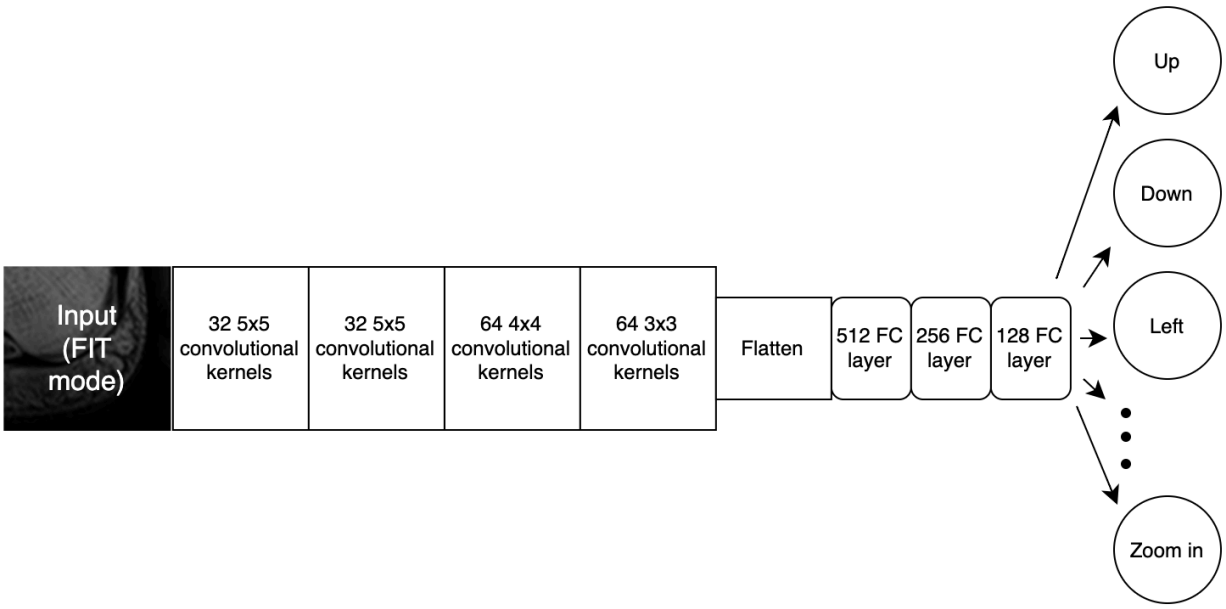

Fig. 12. The general architecture I use for experiments 1-5.

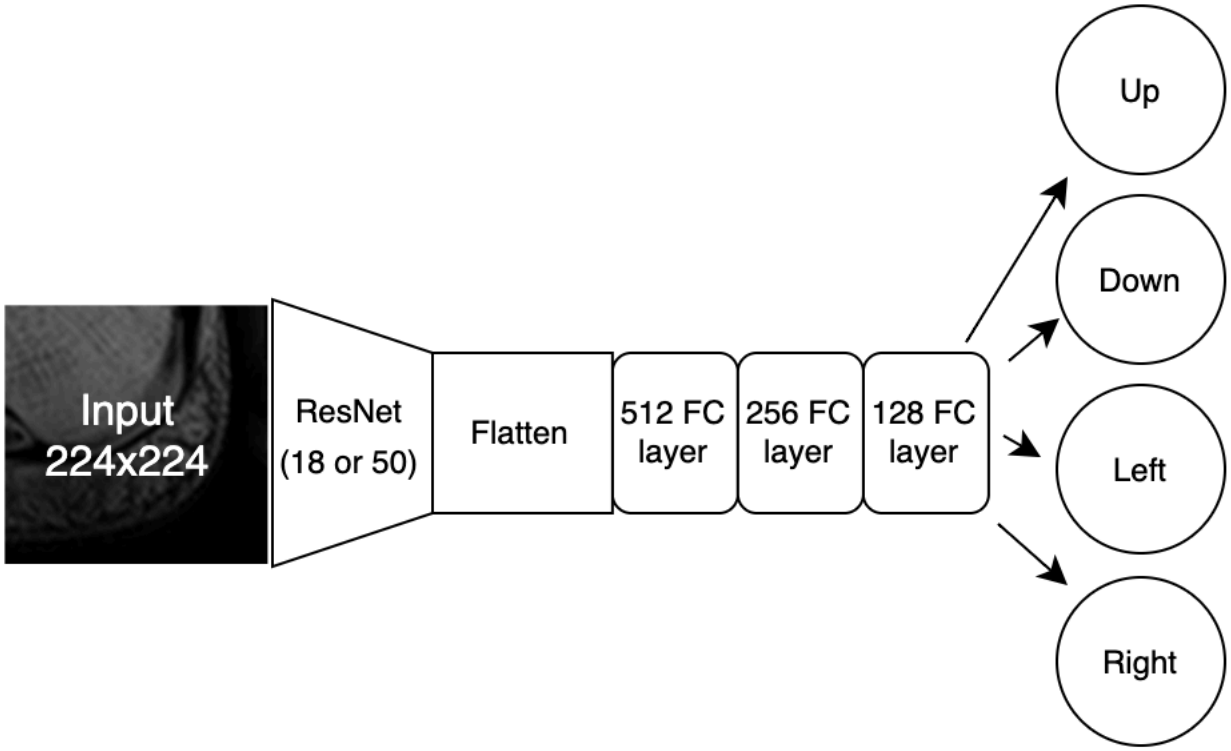

Fig. 13. The architecture used for experiments 6-10. Experiment 7 uses three parallel ResNets prior to concatenation and flattening (this is not shown).

4.3.2.3 Network Training

Both the convolutional layers and FC layers of the network are trained together for experiments 1-7. The main training loop is shown in Fig. 5 of the Background. Although I begin to use ResNet in experiment 6, pretrained ResNets trained on everyday images do not generalize well to medical imaging (Alzubaidi et al., 2021). Following the lead of Maicas et al. (2017), I train ResNet on medical images in experiments A-D in a supervised manner, in order for use in later RL experiments 8-10. To gather training data, I randomly select a box from the MRI scan that is far from the ground truth box and another that is close to the ground truth box (based on the Euclidean distance between the box centers).

After the ResNet is trained independently in this supervised manner (experiments A-D), it is then inserted upstream of the Q-network for experiments 8-10. The Q-network is trained during the ϵ-greedy algorithm of the DQN/ DDQN agent for all experiments. As discussed in the background (Fig. 5), the agent selects an action from a state (either greedily, randomly or with guided exploration). The percentages of greedy choice, random choice and guided exploration are determined prior to the experiment and (in some cases are linearly decreased over training). Then the initial state, the aforementioned action, the new state and the reward, in addition to if the new state is terminal are saved into the experience replay memory. Then the policy network is updated by selecting 32 of these experiences via random sampling. The policy network takes in the initial state and outputs a Q-value for each action. The Q-value of only the transition's action is compared to the label, and this Huber loss (see equation 5) value is backpropagated for this action alone. The Q-value outputs for the other actions are not backpropagated or compared to any labels. The outputs of the Q-network are raw logits (as there is no activation layer prior to the loss function). The aforementioned labels are calculated via Equation 4 for experiments 1-6, Equation 6 for experiments 7-8 and Equation 10 for experiments 9-10. For the target network, the hyperparameter C dictates how often and the hyperparameter τ how much to update the target network weights using the policy network weights. This is not applicable for experiments 9-10, which calculate labels directly from the environment during training (via equation 10) and do not use a target network.

4.3.2.4 Network Policy

I add in guided exploration from Caicedo & Lazebnik (2015) where a small portion of the time a guided action is taken rather than a random action. This ensures some positively rewarded transitions are entering into the memory early on. As ε is linearly decreased over time, the percentage of guided actions goes down as well. I use another hyperparameter to determine whether to increase guided exploration temporarily during an episode to help the agent reach the goal state faster.

4.3.2.5 Episode initiation

Several different methods are used:

1. Starting the agent's box in the center of the image, with the box smaller than the image.
2. Starting the agent's box at the average ground truth location of the training set.
3. Starting the agent's box in a random location.
4. Starting the agent's box as the full size.
5. Starting the agent's box centered and three quarters of the image size.

4.3.2.6 Episode termination

Based on the metric (Euclidean distance or IoU score), when the threshold or the maximum number of steps is reached, the training episode terminates. For inference mode, episodes end when oscillation occurs, a set number of repeated states happen (in a given time frame) or the maximum number of inference steps is reached.

4.3.3 Inference evaluation

Results for inference mode are based on IoU scores and Euclidean distances. For experiments 1-5 when the agent is able to resize the box, IoU was the primary means of evaluation. Experiments 6-10 rely on Euclidean distance for evaluation. Both the ending values and intermediate values are examined during inference mode over the agent's training. Fluctuations in scores indicate instability in training. As the agent learns, IoU score should increase and Euclidean distance should drop. Given that the agent performed inference mode on several slices every epoch, standard deviations and the mean scores are able to be calculated. In addition, median IoU scores and median Euclidean distances are recorded for every inference epoch. The ending median score and best score (that occurred sometime during training) are recorded in Tables I and III.

5 Results

5.1 Edema detection in SAGIR images (Experiments 1-7)

The experiments 1-7 are overall unsuccessful in locating edema in new patients. Some experiments are successful in segmenting edema when training and validation occur on the same slice or set of slices. This shows that the overall deep Q-learning algorithm has limitations and careful environmental construction is needed to obtain satisfactory results. Out of the scores of experiments performed, I show seven of the most important ones in Table I. All experiments shown use the Huber loss function (Equation 3). A batch size of 32 is used for all experiments. The positive reward is +1 and the negative reward is -1. The target values of the policy network are not calculated in these experiments (the *calc target* hyperparameter is set to False) and are determined by the target network in a traditional DQN fashion. The translation and resizing factor, α, is 0.1 for all experiments with the exception of the last one, which uses a value of 0.4 instead. All IoU and Euclidean distance values are for the median value in a given epoch.

The first two shown experiments have a bug where transitions are being overwritten and the experience replay memory is reset after every validation slice. They are included due to interesting results. Experiment 7 uses a DDQN approach with 50% guided exploration that increases as an episode progresses. The significance of each experiment is analyzed in section 6.

TABLE I
DEEP Q-LEARNING EXPERIMENTS FOR EDEMA SEGMENTATION IN SAGIR IMAGES (EXPERIMENTS 1-7)

Exp. No.	box mode; box size	mega epochs; train epochs	no. slices	val. on train	no. of actions	network arch.	input size	no. frames	train term. threshold	max train states; max infer states	start mode; stopping	memory; warm up	C; τ	LR	ε (Δ)	OOB	Max IoU	End IoU	Min Euclid.	End Euclid.
1	slice; changing	10; 4	1	yes	10	Navarro et al.	256	1	IoU=0.8	1k; 1k	original; 4/10	10k ; 0	10; 0.05	1.E-04	0.50 (0)	100	0.8	0.6	0	5
2	slice; changing	10; 3	8	yes	7	Navarro et al.	128	2	IoU=0.8	1k; 1k	original; 5/10	10k ; 0	1; 0.005	1.E-04	1 (0.02)	100	0.3	0.2	12	12
3	fit; changing	12; 3	8	yes	7	3x3 filter	64	4	IoU=0.8	1k; 500	original; osc.	14k ; 0	750; 1	1.E-04	1 (0.02)	5	0.6	0.3	2	4
4	fit; changing	12; 3	8	yes	7	3x3 filter	128	4	IoU=0.8	1k; 500	scaling of Exp. No. 3; osc.	14k ; 0	750; 1	1.E-04	1 (0.02)	5	0.6	0.6	4	4
5	pad; changing	10; 4	28 train, 24 val.	no	7	3x3 filter	256	2	IoU=0.85	1k; 500	original; osc.	14k ; 0	750; 1	1.E-03	.75 (0)	2	0	0	42	42
6	fit; 40	10; 3	28 train, 24 val.	no	4	ResNet18	224	3	Euclid.=10	100; 100	random; max steps	10k ; 2800	1k; 1	1.E-04	0.5 (0)	1	0	0	60	100
7	all 3; 32	10; 4	~20 train, ~10 val.	no	4	3 ResNet50s	224	1	Euclid.=10	500; 50	random; max steps	12k ; ~2000	1k; 1	1.E-06	1 (0.02)	next	0	0	80	110

5.2 Supervised training of ResNets (Experiments A-D)

I train various versions of ResNet in order to be used as a feature extractor for the Q-Network. The final FC layer of ResNet is stripped away and replaced with a single neuron. The BCE loss function is used in training the ResNet to identify if an image patch from an MRI slice is within a certain pixel distance from the center of the calcaneus bone. The PyTorch pretrained ImageNet weights are used as the starting weights with the exception of the bottom-most convolution because this layer is modified due not having three input channels, which is status quo for ResNets. Table II shows some initial experimentation. The box mode for input into these ResNets is Pad with Fit and the box is 32 pixels in length. A LR of 1×10^{-4} is used with a training batch size of 32 and a validation batch size of 64. Adam optimization yields a relatively successful classification. Training is performed on 568 slices and validation is done on 288 slices. These numbers include augmented slices which are created by 90, 180 and 270 degree rotations for each original MRI slice. The Euclidean distance between the patch (box) and the center of the calcaneus is less than or equal to 10 pixels for positive patches, and greater than this distance for negative patches. Twenty positive and twenty negative patches are randomly selected from each of the 568 training slices.

TABLE II
SUPERVISED TRAINING OF FOUR RESNET MODELS (EXPERIMENTS A-D)

Experiment	ResNet Version	f1 score	accuracy	precision	recall	specificity
A	18	0.499	0.742	0.500	0.498	0.982
B	34	0.760	0.758	0.756	0.767	0.751
C	50	0.744	0.764	0.764	0.728	0.799
D	101	0.608	0.738	0.696	0.575	0.890

5.3 Calcaneus detection in SAGT1 (Experiments 8-10)

For these three experiments, I examine calcaneus bone detection via a DDQN approach. The target network is not used as the *calc target* hyperparameter is set to True (see Equation 10 of the Methods). The Huber loss function and a 32-sized batch are used again. Out-of-bounds (OOB) behavior is rectified by choosing the next best action (in terms of next highest Q-value). The discount factor is set to 0 for the final two experiments, but is the typical 0.99 value for experiment 8. The hyperparameter α is set to 0.5 for all three experiments. Table III shows the results of these three experiments. Guided exploration is used 50% of the time (of ε) in experiments 8 and 10, and 52% of the time in experiment 9. Guided exploration increases up to ε as an episode progresses.

TABLE III
CALCANEUS BONE SEGMENTATION IN SAGT1 IMAGES (EXPERIMENTS 7-10)

Exp. No.	box mode; box size	mega epochs; train epochs	no. slices	val. on train	no. of actions	network arch.	input size	no. frames	train term. threshold	max train states; max infer states	start mode; stopping	memory; warm up	LR	ε (Δ)	rewards	Max IoU	End IoU	Min Euclid.	End Euclid.
8	slice and pad; 32	10; 1	568 train, 288 val.	no	4	trained ResNet50	224	1	Euclid=10	50; 50	random; max steps	12k ; ~6500	5.E-05	1 (0.05)	+/-1	0	0	60	70
9	fit; 56	8; 3	568 train, 288 val.	no	4	trained ResNet50	224	1	Euclid=15	50; 50	random; max steps	12k ; 6816	5.E-05	1 (.02)	+1/0	0.1	0.1	50	50
10	fit; 56	10; 6	568 train, 288 val.	no	4	trained ResNet50	224	1	Euclid=15	30; 30	random; max steps	12k ; 6816	1.E-06	1(.04)	+0.5/0	0.2	0.2	40	40

6 Discussion

6.1 Initial experimentation prior to experiment 1

Early experiments contain bugs and are omitted in the results section, except for experiments 1 and 2. Nevertheless, their (poor) results prove an impetus for additional hyperparameter experimentation. One of my earliest observations is that Q-values appear to endlessly increase, which is likely a symptom of a diverging trajectory in the algorithm.

I experiment with removing biases from the final layer and implement *slice* mode so the agent receives the whole slice as input and knows where the box is in the image. Given the box is so small compared to the whole image, it can be lost in the noise. I experiment with decreasing α (similar to how ε is decreased) and introducing Euclidean distance as a metric (because IoU does not change when there is no overlap even if the box moves closer to the ground truth). Early failures lead me to believe the network is not learning because of one or more of the following causes: too few training epochs, the backpropagation is not working, the network architecture is not robust enough, the state representation (input) is poor, the target network is updating too fast or too slow or that edema is exceedingly heterogenous in appearance and thus difficult to identify.

6.2 Experiment 1

Experiment 1 consists of training the agent on a single MRI slice which is passed into the network in its entirety with the current location of the agent's bounding box drawn in. Validation is performed on this same slice. The network architecture is the same as Navarro et al. (2020). All ten actions are available to the agent. Oscillation occurs as referenced by Alansary et al. (2019); Ghesu et al. (2019) and Navarro et al. (2020). This is shown in Fig. 14. The final position of the box has an IoU of 0.6, which is comparable with literature results in organ localization (Navarro et al., 2020). It is noteworthy that the agent performs this well despite a bug which causes some transitions to be overwritten inadvertently. The agent stops in inference mode when it reaches 1000 steps or when four out of the last ten states are identical.

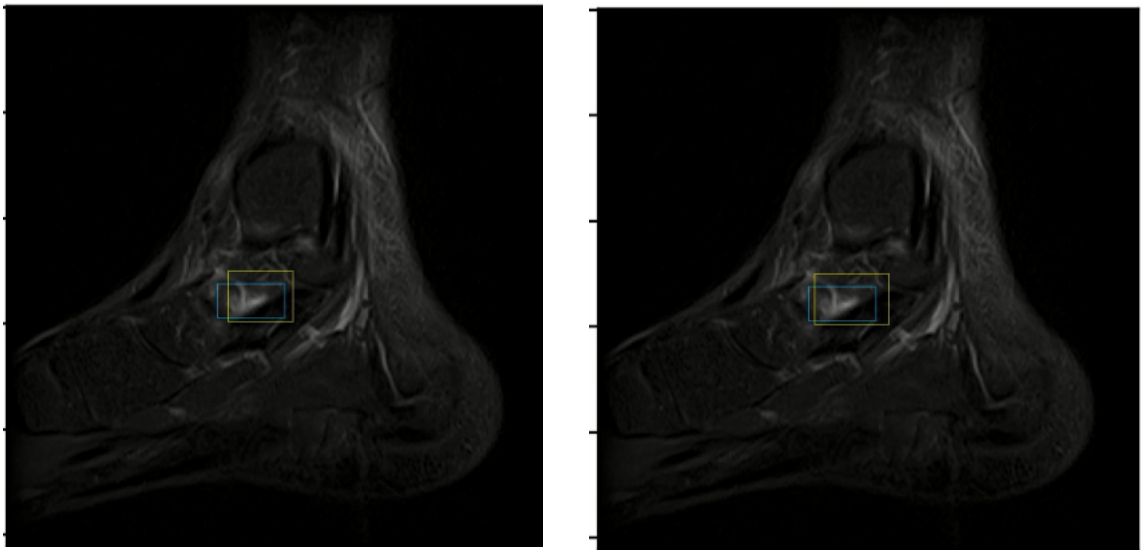

Fig. 14. Oscillation in Inference mode in experiment 1. The blue box is the ground truth which the agent does not "see" during validation (inference mode). The yellow box belongs to the

agent. The agent takes the *wider* action to get from the left to the right state. In the right state, the agent takes the *narrower* action to get back to the left state. This oscillatory behavior signals a stopping point.

6.3 Experiment 2

I expand this experiment to include all single-occurring edema slices of one patient, and I both train and validate on these slices. I limit the action space to seven options, to include the four translational actions, zoom in, taller and wider. I also input the image in smaller dimensions (128 x 128 pixels) and include two frames as the state representation. Fig. 15 shows the graphs of IoU and Euclidean distance. The upward trendline of IoU and downward slope of Euclidean distance shows promise. However, the final values of IoU and Euclidean Distance are inferior to the single slice experiment 1. Results for this experiment and possibly the last experiment could be artificially high due to random pushes, which move the agent away from the image boundary if it tries to move outside the picture (controlled by the OOB hyperparameter). The results in this experiment are comparable to other unreported experiments performed by me in this time frame. I observe rather complex equilibrium behavior for terminal state oscillation (due to multiple frames per state) and the network still does not seem to be learning.

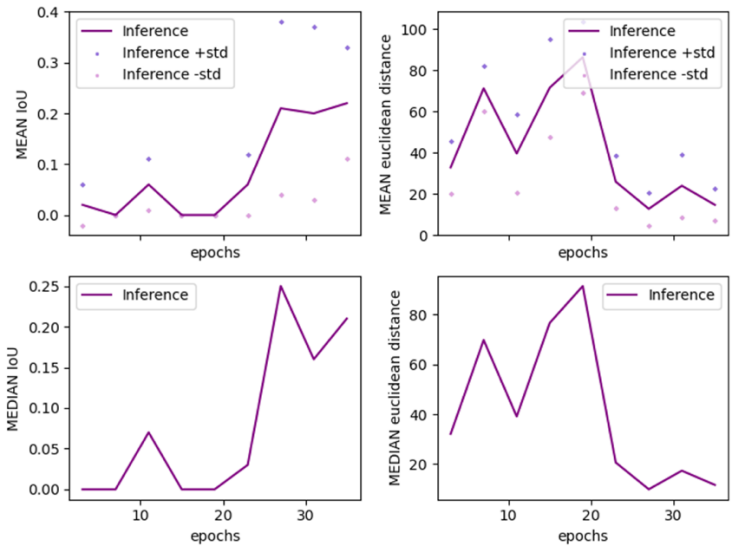

Fig. 15. The mean and median IoU and Euclidean distance for experiment 2. The epoch numbers include both training and validation epochs. Standard deviation (std) is also included.

6.4 Experiments 3 and 4

Experiments 3 and 4 entail a multi-scaling approach, where the ending positions of experiment 3 (which uses 64 x 64 pixel images) are used for experiment 4 (128 x 128 pixels images) starts. Given the agent seems resistant to learn better, I modify the network architecture slightly by reducing the 4 x 4 kernel size to a 3 x 3 one. This 3 x 3 kernel is used in both experiments 3 and 4. For both experiments, I also significantly limit the number of free pushes that move the agent away from an edge down to five (OOB behavior) and introduce the Fit mode, where the agent is given the pixels inside its bounding box as input. The frame number is set to four, so in addition to these pixels, the network also is given the pixel values of the previous three boxes. I increase the update period of the target network (C) and increase τ (the

fraction of policy network weight copied to the target network). I try using a bigger memory to fix this and reduce the number of maximum states for validation to 500. I scale the image to 64 x 64 pixels. The final IoU score is 0.3 with a Euclidean distance of four pixels for experiment 3. However, during the experiment, the IoU score gets as high as 0.6 before decreasing.

For experiment 4, I keep the ending positions of each slice and start the box at this place in the next experiment where I use an image size of 128 x 128. This multi-scale approach has shown great success in the works of Alansary et al. (2019) and Ghesu et al. (2017; 2019). Despite some inconsistency in the trendlines of IoU and Euclidean distance, the agent ends with impressive scores for experiment 4. See Fig. 16.

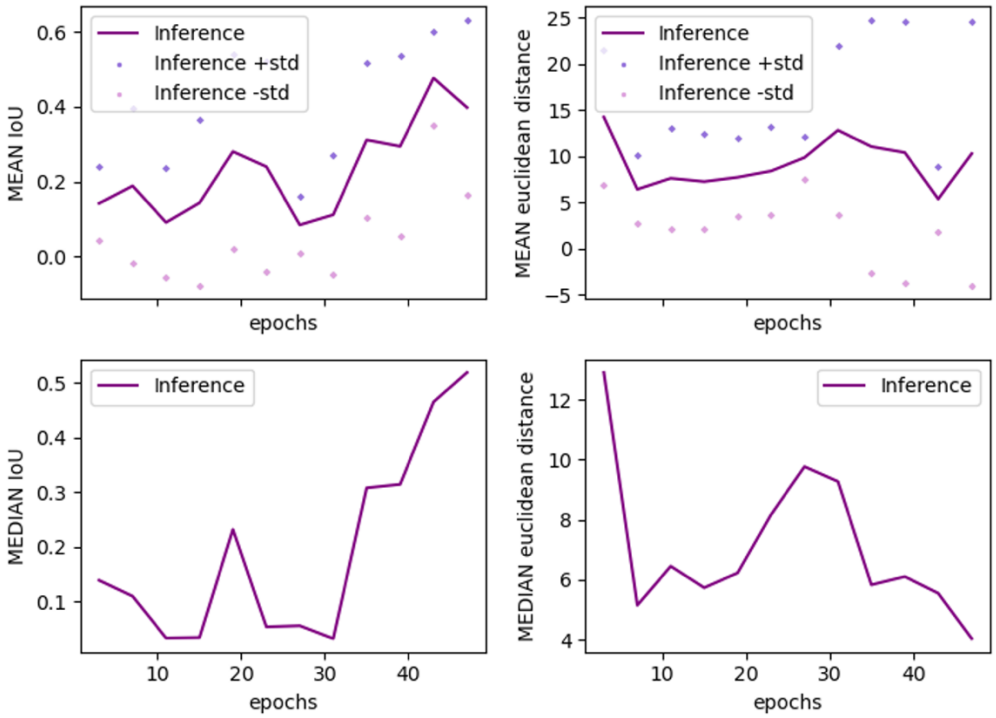

Fig. 16. The mean and median IoU and Euclidean distance for experiment 4. The epoch numbers include both training and validation epochs. Standard deviation (std) is also included. This is the first multi-scale experiment, where the ending positions of experiment 3 (that utilizes 64 x 64 pixel images) are used as the starting positions for this experiment (which utilizes the same images in the higher resolution of 128 x 128 pixels).

6.5 Experiment 5

This is the first experiment that I train and validate on separate patients and slices. I train on 28 slices and validate on 24 different ones. I experiment with Pad mode as well. The IoU score fluctuates wildly as does Euclidean distance (with a possible slight negative slope, showing minor improvement). Additional experiments (not detailed in the results) show that the IoU score is not improving. Although the mean IoU score is around 0.3 for validation, the median is near 0. This shows that in some slices, there is good learning while in most slices, almost none. It is hypothesized that in slices where the edema is localized to the same area in both training and validation, learning occurs, but in dissimilar slices the agent goes to the wrong spot entirely. Changing the number of frames per state proves unsuccessful.

In addition, I try training on a lower IoU threshold that increases over time. I combine this with different size images in a multi-scale approach: 0.4 IoU for 64 x 64 , 0.7 IoU for 128 x 128 then 0.9 for 256 x 256 ones. I experiment with removing the Euclidean metric in rewards, as it is not used in some works such as Navarro et al. (2020). I implement the shuffling of training data and train on more data (validating on other data). These further experiments show that the network is not improving, especially on larger training or validation sets.

6.6 Experiment 6

I change the network architecture to use ResNet18, introduced by He et al. in 2015, as done by Maicas et al. (2017) in their breast lesion detection RL agent. I make this switch after realizing that the zoom in action is overlearned because the agent is always going to the same place and that it cannot distinguish between states sufficiently. The agent seems to be favoring certain actions regardless of what state it is in and keeps stopping at a fixed point, (89, 89), (134, 134), corresponding to several sequential zoom in actions. This leads me to remove the biases from the final layer which choose the action and to simplify the action space down to the four translational actions. I also only use the Euclidean distance as a metric and attempt to just find the center of the pathology. I prevent the agent from leaving the image by making it choose the second best action if the highest Q-valued action takes it outside of the image. Unlike Caicedo & Lazebnik (2015) and Ghesu et al. (2016; 2017) who consider images that do not contain the goal, my MRI slices always contain a single pathology and thus the agent leaving the image is always a mistake.

I make the box size a fixed 40 x 40 pixels and reduce the maximum number of steps in both training and inference episodes. I use action maps showing the direction that the agent will move in each patch of the image. This idea mainly comes from the works of Li et al. (2018), Noothout et al. (2018) and Xu et al. (2017). Looking at the action maps that I generate, I observe that many of them are homogeneous or are bifurcated into two opposing actions. See Fig. 17. When training on a single slice, the Euclidean distance result does not stabilize and better results are not obtained, see Fig. 18.

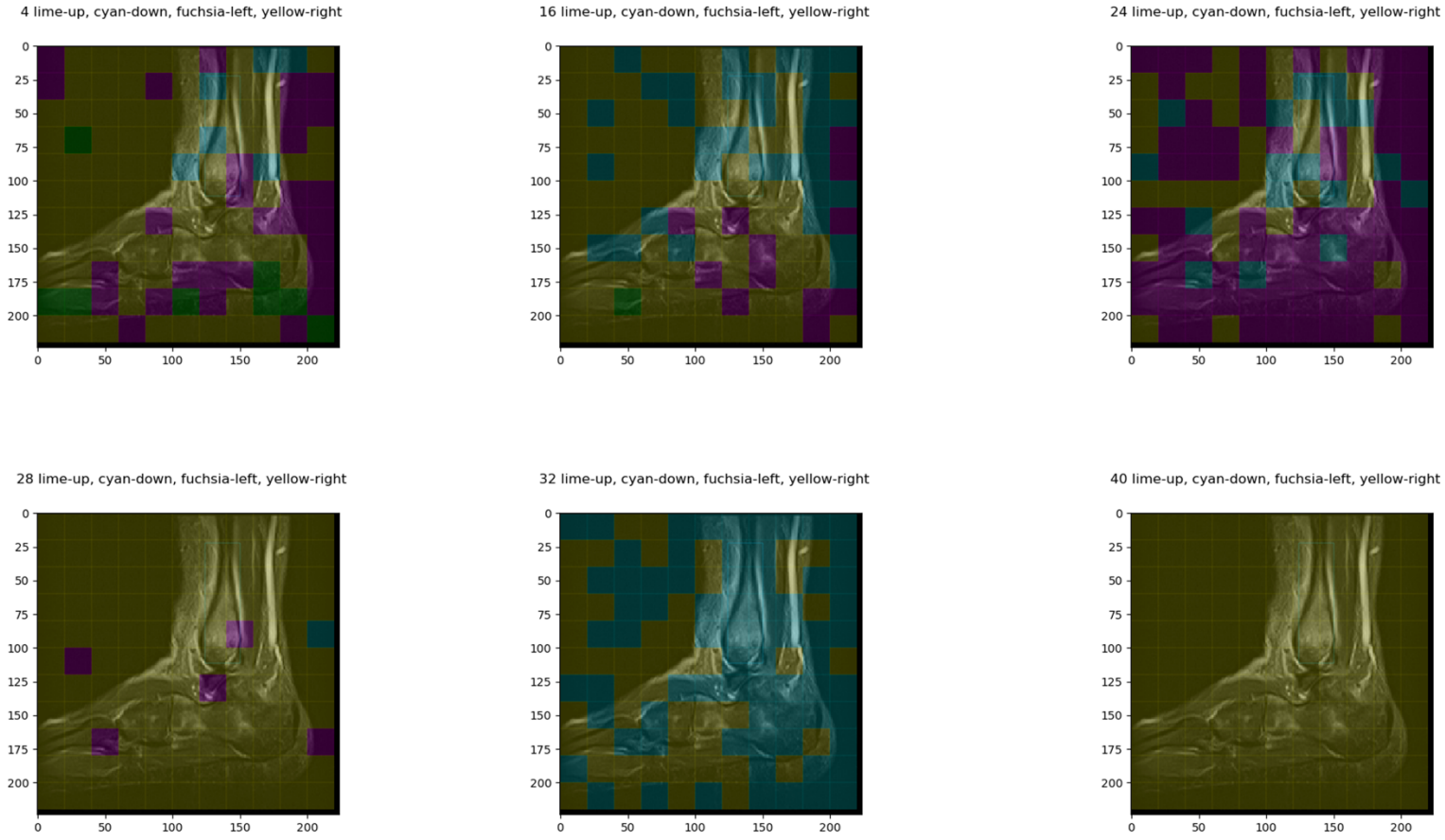

Fig. 17. Patch-wise action map for the same slice at different points in the training during experiment 6. The colors lime, cyan, fuchsia and yellow correspond to the preferred actions of up, down, left and right, respectively. The numbers in the titles correspond to the epoch number (which includes both training and validation). Note that the model overfits to eventually favoring the move right action above all others. The early epochs show more variation in preferred actions, while the later epochs show less.

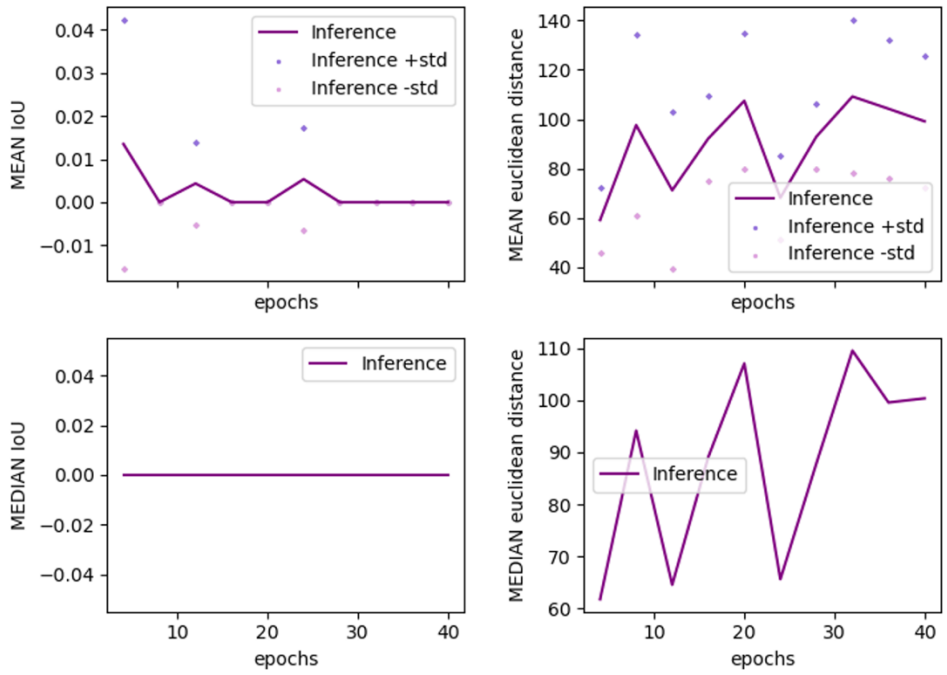

Fig. 18. The mean and median IoU and Euclidean distance for experiment 6. The epoch numbers include both training and validation epochs. Standard deviation (std) is also included. The wavering in the Euclidean distance and flat IoU score make it clear that learning is not occurring in the agent.

6.7 Experiment 7

I introduce a DDQN, by letting the policy network pick the best action from the next state. This does not improve results as the agent seems to keep forgetting any learning as it trains (despite the use of a fairly large experience replay of 12k transitions).

Q-values rise uncontrollably – ending around 30 when training finishes. Multi-frame methods are not working so I make major changes again. I pass in the entire slice, pad mode and fit mode in their own ResNet50 prior to the Q-Network. I also increase α to 0.4 so that the agent does not have to move as many times to get to the goal state. Training occurs on approximately 20 slices and validation on approximately 10. Results are again not satisfactory, with an ending Euclidean distance of over 100 pixels and an ending median IoU of 0.

6.8 Experiments A, B, C and D

I shift course here and move to calcaneus segmentation as it occurs in a consistent place in every image. To do this, I copy the approach of Maicas et al. (2017) and train various ResNets separate from the Q-Network in a supervised manner. Given that the weights of the first layer are randomly initialized during training and do not come from the PyTorch ResNet weights, their values could play a large role in which ResNets converge the best. As shown in the results section, ResNet34 and ResNet50 yield the best results after training on 23,520 instances. I elect to use ResNet50 in further exploration and experiments due to its robustness in the five metrics (from Table II), in addition to its larger state representation capacity. ResNet50 has 2048 nodes upon flattening the final convolutional layer (assuming a single channel input of 224 x 224 pixels), while ResNet34 only has 512. Further training, in a similar manner is performed, and the

final product is used as a feature extractor upstream to the Q-Network in subsequent RL experiments 8-10.

6.9 Experiment 8

Given that the T1 images have better resolution than the IR ones, I investigate those for calcaneus localization. I use the Segment Any Bone tool in addition to traditional image processing techniques (such as dilation and erosion) in combination with the connected components algorithm to draw in a bounding box for the calcaneus bone (Gu et al., 2024). Figs. 19 and 20 show this. I use outlier detection to remove components that are clearly in the wrong place and only choose images in which the mask fits the calcaneus well. I arrive at 284 segmented calcanei. This could be relevant to finding edema, given that one work (Ribeiro et al., 2023) localizes bone marrow edema by first extracting the bones in the knee.

Fig. 19. The Segment Any Bone tool by Gu et al. (2024) is used to create a bone mask (shown in blue in the middle image). The original image is on the far left and the third image is an attention map.

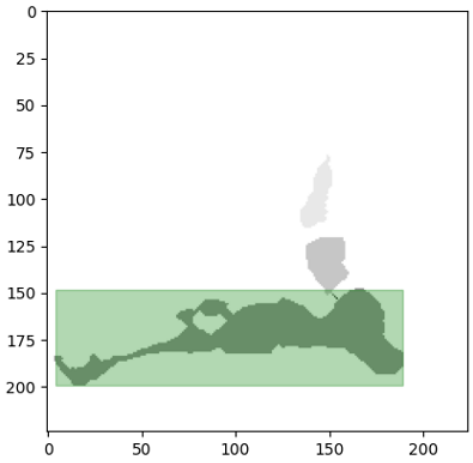

Fig 20. The calcaneus bone conglomerated with other bones. Erosion and outlier detection is needed in cases similar to the above in order to ensure the calcaneus is separated correctly for the connected components algorithm. This is from a different slice than that used in Fig. 19.

I use these segmentations as ground truths and then apply my DDQN to them. I also use guided exploration (which increases as each episode progresses) and set α to 0.5 for the

following three experiments. I reduce the maximum training and validation episodic steps to only 50. This first experiment to segment the calcaneus in 288 (augmented) validation images after training on 568 images is unsuccessful. Augmentation via 90, 180, 270 rotations on every image on both training and validation is used without yielding satisfactory results.

6.10 Experiments 9 and 10

This difference between experiments 9 and 10 are, different rewards, epochs, decreases in exploration and the learning rate (see Table III in the Results section for more details). When the action space is restricted to translation (and α does not change) the converged Q-values can be calculated (see Equation 10 of the Methods); the target network is not needed. I implement a hyperparameter that controls whether or not to use these calculated values.

In addition to experiments A-D, I train ResNet 50 with a 56 x 56 pixel box and more data than before for 3 different modes: Fit (f1 of 0.789 and 0.791 accuracy), Slice with Pad (0.717 f1 and 0.745 accuracy) and Pad (f1 0.616 and accuracy of 0.689). Finally, I run two additional RL experiments with the discount factor set to 0 and the agent appears to stabilize with 0.2 IoU scores for the calcaneus. See Figs. 21 and 22. This discount factor being 0 seems to be a critical factor in stabilization, as it ensures that the myopic agent focuses on getting the next reward only. Further experimentation will be needed to ensure this.

In later experiments (not included) I postulate that the path-dependent reward structure used in almost every paper might lead to Q-values being indistinguishable in noisy environments. I implement a -1 reward for every step to mitigate this issue. The Q-values for -1 reward and for a +1 / -1 reward scenario are shown in the proceeding section.

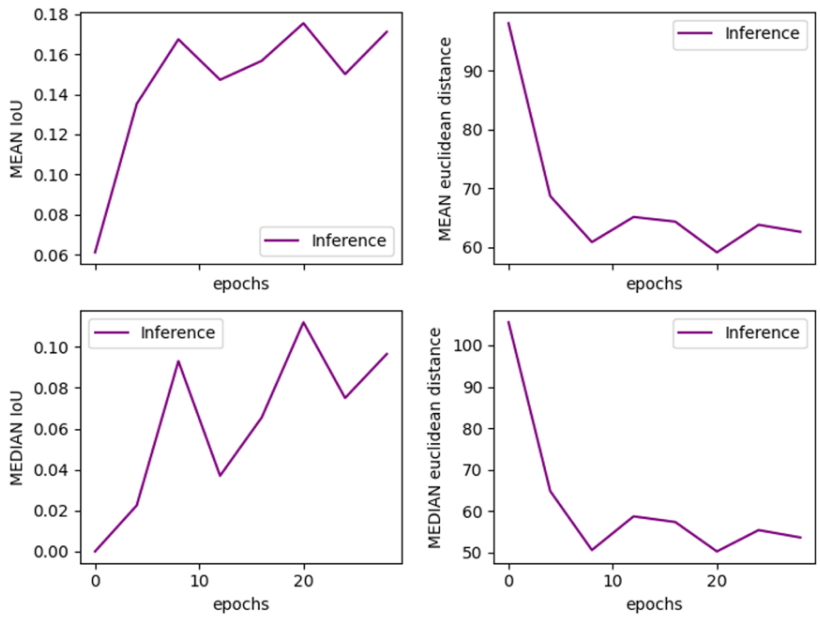

Fig. 21. The mean and median IoU and Euclidean distance for experiment 9.

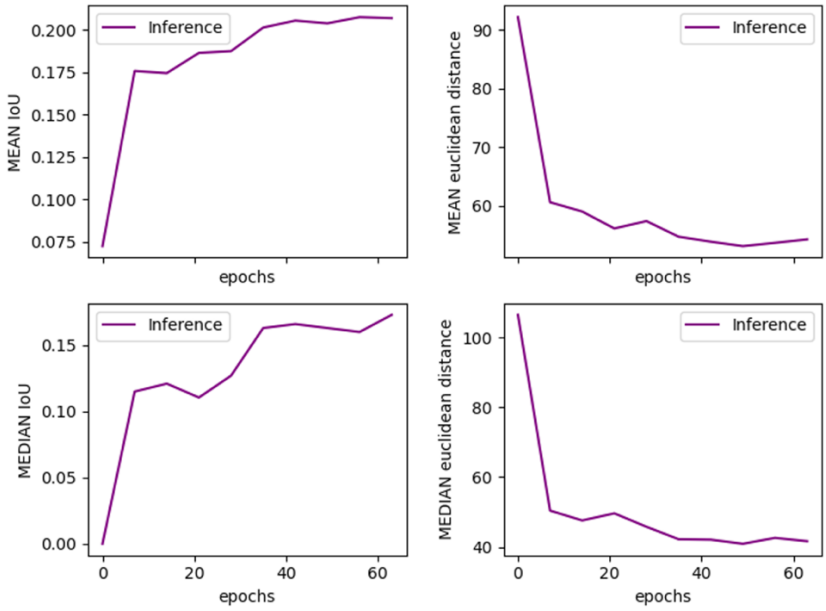

Fig. 22. The mean and median IoU and Euclidean distance for experiment 10.

In retrospect, it is clear that differences in anatomy between patients and especially the differences in fracture locations between patients makes learning quite difficult. On a large training set, the edema is in several different locations and when the agent sees an unknown MRI slice, it has little idea of whether to check the phalanges first or the talus bones.

6.11 Why to use a -1 reward per step exclusively: A problem simplification (Gridworld)

Neglecting box resizing and trigger actions, my problem simplifies down to Gridworld from Sutton & Barto (2020). The agent is bound to a finite board (the image) and moves around in it, trying to reach the goal state (edema or the calcaneus bone). Actions that take the agent off the board, leave the state unchanged. The agent starts at a given location and must navigate to either bold terminal state in as few steps as possible. A terminal state is also indicated by being grey-filled. Fig. 23 shows the original Gridworld scenario from Sutton & Barto (2020) and Fig. 24 shows the problem with a single terminal state.

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	**15**

Fig. 23. Adapted from Sutton & Barto, (2020).

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Fig. 24. My problem reduced to the Gridworld scenario. While their example has two terminal goal positions, mine only has one.

The position of the terminal state is not always in the top left and rather than an integer or coordinate for the state, I use pixel intensities. Running value iteration, the state values converge to the below (Fig. 25), if a discount factor of 0.99 and a reward of -1 per step is assumed. Q-values are shown in Fig. 26.

0	-1.00	-1.99	-2.97
-1.00	-1.99	-2.97	-3.94
-1.99	-2.97	-3.94	-4.90
-2.97	-3.94	-4.90	-5.85

Fig. 25. The converged state values of Gridworld. As expected, these state values are the maximum Q-values of the actions for each state as shown in Fig. 26.

		⬆ -1.99 ⬅ -1.00	➡ -2.97 ⬇ -2.97	⬆ -2.97 ⬅ -1.99	➡ -3.94 ⬇ -3.94	⬆ -3.94 ⬅ -2.97	➡ -3.94 ⬇ -4.90
⬆ -1.00 ⬅ -1.99	➡ -2.97 ⬇ -2.97	⬆ -1.99 ⬅ -1.99	➡ -3.94 ⬇ -3.94	⬆ -2.97 ⬅ -2.97	➡ -4.90 ⬇ -4.90	⬆ -3.94 ⬅ -3.94	➡ -4.90 ⬇ -5.85
⬆ -1.99 ⬅ -2.97	➡ -3.94 ⬇ -3.94	⬆ -2.97 ⬅ -2.97	➡ -3.94 ⬇ -3.94	⬆ -3.94 ⬅ -3.94	➡ -5.85 ⬇ -5.85	⬆ -4.90 ⬅ -4.90	➡ -5.85 ⬇ -6.79
⬆ -2.97 ⬅ -3.94	➡ -4.90 ⬇ -3.94	⬆ -3.94 ⬅ -3.94	➡ -5.85 ⬇ -4.90	⬆ -4.90 ⬅ -4.90	➡ -6.79 ⬇ -5.85	⬆ -5.85 ⬅ -5.85	➡ -6.79 ⬇ -6.79

Fig. 26. Q-values for Gridworld with -1 reward for every step.

The vast majority of papers employ a different reward structure where the agent is rewarded +1 for moving the right way and -1 if not. This non-Markovian reward structure produces Q-values that are significantly closer. In a noisy environment when the Q-value is determined by an approximator function (a CNN + Q-network), it is possible that these subtle difference will be nearly insignificant and the agent might choose the wrong action. See Fig. 27. For this reason, I suspect a -1 reward for every step might lead to better results.

		⬆ -0.01 ⬅ 1.00	➡ 0.97 ⬇ 0.97	⬆ 0.97 ⬅ 1.99	➡ 1.94 ⬇ 1.94	⬆ 1.94 ⬅ 2.97	➡1.94 ⬇ 2.90
⬆ 1.00 ⬅ -0.01	➡ 0.97 ⬇ 0.97	⬆ 1.99 ⬅ 1.99	➡ 1.94 ⬇ 1.94	⬆ 2.97 ⬅ 2.97	➡ 2.90 ⬇ 2.90	⬆ 3.94 ⬅ 3.94	➡ 2.90 ⬇ 3.85
⬆ 1.99 ⬅ 0.97	➡ 1.94 ⬇ 1.94	⬆ 2.97 ⬅ 2.97	➡ 2.90 ⬇ 2.90	⬆ 3.94 ⬅ 3.94	➡ 3.85 ⬇ 3.85	⬆ 4.90 ⬅ 4.90	➡ 3.85 ⬇ 4.79
⬆ 2.97 ⬅ 1.94	➡ 2.90 ⬇ 1.94	⬆ 3.94 ⬅ 3.94	➡ 3.85 ⬇ 2.90	⬆ 4.90 ⬅ 4.90	➡ 4.79 ⬇ 3.85	⬆ 5.85 ⬅ 5.85	➡ 4.79 ⬇ 4.79

Fig. 27. The Q-values for Gridworld with rewards dependent on the path (closer +1, farther -1).

7 Conclusion

7.1 Successful and unsuccessful experiments

Both experiments 9 and 10 show a consistent IoU score of around 0.2 for calcaneus segmentation, and this is for a fixed box size of 56 pixels in length. Experiments in localizing the calcaneus are for the most part preliminary, as the bulk of my time was used in exploring edema. Further experiments in locating the heel bone are needed to confirm results. In addition, adding back in box size-changing actions would almost certainly improve the IoU score.

Overall, I suspect that my edema-based experiments fail to yield good results because of the inconsistency in pathology location. With the exception of two papers, all other explored papers use segmentation and localization in a consistent place in the image. The organ localization paper looks at abdominal imaging, where the lungs are always towards the upper part of the torso, and the pancreas somewhere in the middle-bottom of the image (Navarro et al., 2020). Alansary et al.'s (2019) landmark identifications has specific locations and substructures that are relatively fixed. The original paper, that inspired much of the work, deals with structures that are mainly in the foreground and are not in the medical imaging domain (Caicedo & Lazebnik, 2015). The variation in edema appearance between patients, combined with its different locations might demand additional methodologies in order to segment it correctly.

7.2 Future work

The issue with a pathology that is not in a consistent location across patients, is that the agent only sees one part of the image at a time. A greedy agent is not stochastic and simply moves to maximize the Q-values (Sutton & Barto, 2020). It would not know to check the phalanges for fracture if there is no fracture in the heel, it simply would move towards one of them in a two-dimensional unimodal action map. In addition to exploring -1 rewards per step (as aforementioned), I propose four additional ideas that might circumvent this problem: implementing a trigger action, using an external classifier, utilizing multiple start locations, and using a more complex state (which includes previous actions or frames).

Caicedo & Lazebnik, (2015) use a trigger action to signal when the agent has found an object, and Maicas et al. (2017) use this technique with success to find breast lesions in 3D volumes. This is the closest case to my work, as the lesion appearance and location within the volume can be heterogenous. Once the agent believes it reaches edema, it could hit the trigger action and restart in a new part of the image. This could allow for multi-object search or help to provide more consistent results and outlier detection. Both Caicedo & Lazebnik, (2015) and

Alansary et al. (2019) use this approach. Without a trigger action all the agent can do is move to the most likely spot if the state representation is not multi-frame.

Caicedo & Lazebnik, (2015) use a Support Vector Machine (SVM) to classify if the object is in the terminal state or in any state reached along the way. The SVM could be trained in a supervised manner (similar to experiments A-D). It could learn the appearance of edema and could provide backup to the agent, potentially improving the segmentation accuracy.

Edema localization might be achieved better in MRI slices by modifying the state. Caicedo & Lazebnik, (2015) successfully use a history of the previous ten actions as an addition to the state. An agent might be able to modify its behavior if it observes that the pathology is not present at a likely location, if it has previous frames or actions in its state. In addition, many papers use multiple frames, as I also do (Alansary et al., 2019; Mnih et al., 2015; Navarro et al., 2020). This could be combined with an action history to give the agent a better idea of where it has gone within the image, potentially leading to improved results.

References

Ackermann, J., Wieland, M., Hoch, A., Ganz, R., Snedeker, J. G., Oswald, M. R., Pollefeys, M., Zingg, P. O., Esfandiari, H., & Fürnstahl, P. (2021). A New Approach to Orthopedic Surgery Planning Using Deep Reinforcement Learning and Simulation. In M. De Bruijne, P. C. Cattin, S. Cotin, N. Padoy, S. Speidel, Y. Zheng, & C. Essert (Eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021* (Vol. 12904, pp. 540–549). Springer International Publishing. https://doi.org/10.1007/978-3-030-87202-1_52

Alansary, A., Oktay, O., Li, Y., Folgoc, L. L., Hou, B., Vaillant, G., Kamnitsas, K., Vlontzos, A., Glocker, B., Kainz, B., & Rueckert, D. (2019). Evaluating reinforcement learning agents for anatomical landmark detection. *Medical Image Analysis*, *53*, 156–164. https://doi.org/10.1016/j.media.2019.02.007

Alzubaidi, L., Al-Amidie, M., Al-Asadi, A., Humaidi, A. J., Al-Shamma, O., Fadhel, M. A., Zhang, J., Santamaría, J., & Duan, Y. (2021). Novel Transfer Learning Approach for Medical Imaging with Limited Labeled Data. *Cancers*, *13*(7), 1590. https://doi.org/10.3390/cancers13071590

Caicedo, J. C., & Lazebnik, S. (2015). Active Object Localization with Deep Reinforcement Learning. *2015 IEEE International Conference on Computer Vision (ICCV)*, 2488–2496. https://doi.org/10.1109/ICCV.2015.286

Chen, C., Lin, J.-R., Zhang, Y., Ye, T.-B., & Yang, Y.-F. (2024). A systematic analysis on global epidemiology and burden of foot fracture over three decades. *Chinese Journal of Traumatology*, S1008127524000270. https://doi.org/10.1016/j.cjtee.2024.03.001

Crönlein, M., Rauscher, I., Beer, A. J., Schwaiger, M., Schäffeler, C., Beirer, M., Huber, S., Sandmann, G. H., Biberthaler, P., Eiber, M., & Kirchhoff, C. (2015). Visualization of stress fractures of the foot using PET-MRI: A feasibility study. *European Journal of Medical Research*, *20*(1), 99. https://doi.org/10.1186/s40001-015-0193-6

Donner, R., Menze, B. H., Bischof, H., & Langs, G. (2013). Global localization of 3D anatomical structures by pre-filtered Hough Forests and discrete optimization. *Medical Image Analysis*, *17*(8), 1304–1314. https://doi.org/10.1016/j.media.2013.02.004

Fracture (2024). Lehigh Valley Health Network. https://www.lvhn.org/conditions/fracture

Gao, R., Ghesu, F. C., Arberet, S., Basiri, S., Kuusela, E., Kraus, M., Comaniciu, D., & Kamen, A. (2024). *Multi-Agent Reinforcement Learning Meets Leaf Sequencing in Radiotherapy* (arXiv:2406.01853). arXiv. http://arxiv.org/abs/2406.01853

Ghesu, F. C., Georgescu, B., Grbic, S., Maier, A. K., Hornegger, J., & Comaniciu, D. (2017). Robust Multi-scale Anatomical Landmark Detection in Incomplete 3D-CT Data. In M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, & S. Duchesne (Eds.), *Medical Image Computing and Computer Assisted Intervention − MICCAI 2017* (Vol. 10433, pp. 194–202). Springer International Publishing. https://doi.org/10.1007/978-3-319-66182-7_23

Ghesu, F. C., Georgescu, B., Mansi, T., Neumann, D., Hornegger, J., & Comaniciu, D. (2016). An Artificial Agent for Anatomical Landmark Detection in Medical Images. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, & W. Wells (Eds.), *Medical Image Computing and Computer-Assisted Intervention—MICCAI 2016* (Vol. 9902, pp. 229–237). Springer International Publishing. https://doi.org/10.1007/978-3-319-46726-9_27

Ghesu, F. C., Georgescu, B., Zheng, Y., Grbic, S., Maier, A., Hornegger, J., & Comaniciu, D. (2019). Multi-Scale Deep Reinforcement Learning for Real-Time 3D-Landmark Detection in CT Scans. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *41*(1), 176–189. https://doi.org/10.1109/TPAMI.2017.2782687

Gu, H., Colglazier, R., Dong, H., Zhang, J., Chen, Y., Yildiz, Z., Chen, Y., Li, L., Yang, J., Willhite, J., Meyer, A. M., Guo, B., Shah, Y. A., Luo, E., Rajput, S., Kuehn, S., Bulleit, C., Wu, K. A., Lee, J., … Mazurowski, M. A. (2024). *SegmentAnyBone: A Universal Model that Segments Any Bone at Any Location on MRI* (arXiv:2401.12974). arXiv. http://arxiv.org/abs/2401.12974

He, K., Zhang, X., Ren, S., & Sun, J. (2015). *Deep Residual Learning for Image Recognition* (arXiv:1512.03385). arXiv. http://arxiv.org/abs/1512.03385

Hu, M., Zhang, J., Matkovic, L., Liu, T., & Yang, X. (2023). Reinforcement learning in medical image analysis: Concepts, applications, challenges, and future directions. *Journal of Applied Clinical Medical Physics*, *24*(2), e13898. https://doi.org/10.1002/acm2.13898

Kasseroller, K., Thaler, F., Payer, C., & Štern, D. (2021). Collaborative Multi-agent Reinforcement Learning for Landmark Localization Using Continuous Action Space. In A. Feragen, S. Sommer, J. Schnabel, & M. Nielsen (Eds.), *Information Processing in Medical Imaging* (Vol. 12729, pp. 767–778). Springer International Publishing. https://doi.org/10.1007/978-3-030-78191-0_59

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). *Segment Anything* (arXiv:2304.02643). arXiv. http://arxiv.org/abs/2304.02643

Kitamura, G., Chung, C. Y., & Moore, B. E. (2019). Ankle Fracture Detection Utilizing a Convolutional Neural Network Ensemble Implemented with a Small Sample, De Novo Training, and Multiview Incorporation. *Journal of Digital Imaging*, *32*(4), 672–677. https://doi.org/10.1007/s10278-018-0167-7

Li, Y., Alansary, A., Cerrolaza, J. J., Khanal, B., Sinclair, M., Matthew, J., Gupta, C., Knight, C., Kainz, B., & Rueckert, D. (2018). Fast Multiple Landmark Localisation Using a Patch-Based Iterative Network. In A. F. Frangi, J. A. Schnabel, C. Davatzikos, C. Alberola-López, & G. Fichtinger (Eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018* (Vol. 11070, pp. 563–571). Springer International Publishing. https://doi.org/10.1007/978-3-030-00928-1_64

Ma, J., He, Y., Li, F., Han, L., You, C., & Wang, B. (2024). Segment anything in medical images. *Nature Communications*, *15*(1), 654. https://doi.org/10.1038/s41467-024-44824-z

Maicas, G., Carneiro, G., Bradley, A. P., Nascimento, J. C., & Reid, I. (2017). Deep Reinforcement Learning for Active Breast Lesion Detection from DCE-MRI. In M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, & S. Duchesne (Eds.), *Medical Image Computing and Computer Assisted Intervention − MICCAI 2017* (Vol. 10435, pp. 665–673). Springer International Publishing. https://doi.org/10.1007/978-3-319-66179-7_76

Meena, T., & Roy, S. (2022). Bone Fracture Detection Using Deep Supervised Learning from Radiological Images: A Paradigm Shift. *Diagnostics*, *12*(10), 2420. https://doi.org/10.3390/diagnostics12102420

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

Navarro, F., Sekuboyina, A., Waldmannstetter, D., Peeken, J. C., Combs, S. E., & Menze, B. H. (2020). *Deep Reinforcement Learning for Organ Localization in CT*.

Noothout, J. M. H., de Vos, B. D., Wolterink, J. M., Leiner, T., & Išgum, I. (2018). *CNN-based Landmark Detection in Cardiac CTA Scans* (arXiv:1804.04963). arXiv. http://arxiv.org/abs/1804.04963

Payer, C., Štern, D., Bischof, H., & Urschler, M. (2016). Regressing Heatmaps for Multiple Landmark Localization Using CNNs. In S. Ourselin, L. Joskowicz, M. R. Sabuncu, G. Unal, & W. Wells (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016* (Vol. 9901, pp. 230–238). Springer International Publishing. https://doi.org/10.1007/978-3-319-46723-8_27

Pesce, E., Joseph Withey, S., Ypsilantis, P.-P., Bakewell, R., Goh, V., & Montana, G. (2019). Learning to detect chest radiographs containing pulmonary lesions using visual attention networks. *Medical Image Analysis*, *53*, 26–38. https://doi.org/10.1016/j.media.2018.12.007

Pranata, Y. D., Wang, K.-C., Wang, J.-C., Idram, I., Lai, J.-Y., Liu, J.-W., & Hsieh, I.-H. (2019). Deep learning and SURF for automated classification and detection of calcaneus fractures in CT images. *Computer Methods and Programs in Biomedicine*, *171*, 27–37. https://doi.org/10.1016/j.cmpb.2019.02.006

PyTorch. (2023). *HuberLoss*. Retrieved August 11, 2024. https://pytorch.org/docs/stable/generated/torch.nn.HuberLoss.html
PyTorch. (2024). Reinforcement Learning (DQN) Tutorial. Retrieved August 11, 2024. https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html
Rahmaniar, W., & Wang, W.-J. (2019). Real-Time Automated Segmentation and Classification of Calcaneal Fractures in CT Images. *Applied Sciences*, *9*(15), 3011. https://doi.org/10.3390/app9153011
Ribeiro, G., Pereira, T., Silva, F., Sousa, J., Carvalho, D. C., Dias, S. C., & Oliveira, H. P. (2023). Learning Models for Bone Marrow Edema Detection in Magnetic Resonance Imaging. *Applied Sciences*, *13*(2), 1024. https://doi.org/10.3390/app13021024
Riedmiller, M. (1998). Reinforcement learning without an explicit terminal state. *1998 IEEE International Joint Conference on Neural Networks Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98CH36227)*, *3*, 1998–2003. https://doi.org/10.1109/IJCNN.1998.687166
Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. In N. Navab, J. Hornegger, W. M. Wells, & A. F. Frangi (Eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015* (Vol. 9351, pp. 234–241). Springer International Publishing. https://doi.org/10.1007/978-3-319-24574-4_28
Sahba, F., Tizhoosh, H. R., & Salama, M. M. A. (2006). A Reinforcement Learning Framework for Medical Image Segmentation. *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, 511–517. https://doi.org/10.1109/IJCNN.2006.246725
Shokri, M., & Tizhoosh, H. R. (2003). Using reinforcement learning for image thresholding. *CCECE 2003 - Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No.03CH37436)*, *2*, 1231–1234. https://doi.org/10.1109/CCECE.2003.1226121
Stember, J., & Shalu, H. (2020). Deep reinforcement learning to detect brain lesions on MRI: A proof-of-concept application of reinforcement learning to medical images. arXiv. https://arxiv.org/abs/2008.02708
Stember, J., Hrithwik Shalu, & Balaraman Ravindran. (2021). *Comparison of Contextual Bandits versus Markov Decision Process Reinforcement Learning for MRI brain classification*. https://doi.org/10.13140/RG.2.2.24944.58884
Stember, J. N., & Shalu, H. (2022). Deep Reinforcement Learning with Automated Label Extraction from Clinical Reports Accurately Classifies 3D MRI Brain Volumes. *Journal of Digital Imaging*, *35*(5), 1143–1152. https://doi.org/10.1007/s10278-022-00644-5
Stember, J., & Shalu, H. (2021). *Deep reinforcement learning-based image classification achieves perfect testing set accuracy for MRI brain tumors with a training set of only 30 images* (arXiv:2102.02895). arXiv. http://arxiv.org/abs/2102.02895
Sutton, R. S., & Barto, A. G. (2020). *Reinforcement learning: An introduction* (2nd ed.). MIT Press.
Szaro, P., Geijer, M., & Solidakis, N. (2020). Traumatic and non-traumatic bone marrow edema in ankle MRI: A pictorial essay. *Insights into Imaging*, *11*(1), 97. https://doi.org/10.1186/s13244-020-00900-8
Szaro, P., Polaczek, M., Świątkowski, J., & Kocoń, H. (2020). How to increase the accuracy of the diagnosis of the accessory bone of the foot? *La Radiologia Medica*, *125*(2), 188–196. https://doi.org/10.1007/s11547-019-01104-x
Tanzi, L., Vezzetti, E., Moreno, R., & Moos, S. (2020). X-Ray Bone Fracture Classification Using Deep Learning: A Baseline for Designing a Reliable Approach. *Applied Sciences*, *10*(4), 1507. https://doi.org/10.3390/app10041507
Vlontzos, A., Alansary, A., Kamnitsas, K., Rueckert, D., & Kainz, B. (2019). *Multiple Landmark Detection using Multi-Agent Reinforcement Learning* (arXiv:1907.00318). arXiv. http://arxiv.org/abs/1907.00318

Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. Machine Learning, 8(3), 279-292. https://doi.org/10.1007/BF00992698

Watkins, C. J. C. H. (1989). *Learning from delayed rewards* (Doctoral dissertation). University of Cambridge.

Xu, Z., Huang, Q., Park, J., Chen, M., Xu, D., Yang, D., Liu, D., & Zhou, S. K. (2017). Supervised Action Classifier: Approaching Landmark Detection as Image Partitioning. In M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, & S. Duchesne (Eds.), *Medical Image Computing and Computer Assisted Intervention − MICCAI 2017* (Vol. 10435, pp. 338–346). Springer International Publishing. https://doi.org/10.1007/978-3-319-66179-7_39

Zheng, G., Lai, S., Braverman, V., Jacobs, M. A., & Parekh, V. S. (2023). *Multi-environment lifelong deep reinforcement learning for medical imaging* (arXiv:2306.00188). arXiv. http://arxiv.org/abs/2306.00188

Zhou, S. K., Le, H. N., Luu, K., V Nguyen, H., & Ayache, N. (2021). Deep reinforcement learning in medical imaging: A literature review. *Medical Image Analysis*, *73*, 102193. https://doi.org/10.1016/j.media.2021.102193

Zubler, V., Mengiardi, B., Pfirrmann, C. W. A., Duc, S. R., Schmid, M. R., Hodler, J., & Zanetti, M. (2007). Bone marrow changes on STIR MR images of asymptomatic feet and ankles. *European Radiology*, *17*(12), 3066–3072. https://doi.org/10.1007/s00330-007-0691-1

Appendix
Definitions and explanations of hyperparameters (from Tables I and III)

Box mode – How to input the box to the network (slice – the entire slice, pad – the box surrounded by black padding in early experiments it was not positioned in the slice, in later experiments it was, fit – a zoomed in (higher resolution display of the image that fit to the network input size). All means all three of the above.

Box size – how big the box is (in pixels).

C – The target network update period. How often the weights of the policy network get copied to the target network.

Δ – The amount ε decreases per episode.

ε – Epsilon. The percent of the time exploration occurs.

End Euclid. - The ending median Euclidean distance during validation.

End IoU - The ending median IoU score during validation.

Epochs: Training epochs per mega epoch, validation epochs per mega epoch, number of mega epochs. A mega epoch is a set of training and validation epochs. The number of validation epochs is always 1 per mega epoch.

Exp. No. – The experiment number. Additional experiments were performed. Only the most prominent experiments were included.

Input size – The number of pixels input to the network in one dimension. Only square images were used. Thus 128 signifies an image of 128 x 128 pixels.

LR – Learning rate for the policy network

Max infer states - The maximum number of validation states in an episode.

Max IoU – The highest median IoU score for an epoch that occurred during validation.

Max train states – The maximum number of training states in an episode before termination (in case the training term. threshold is not reached).

Memory – Size of the experience replay buffer.

Min Euclid. – The lowest median Euclidean distance for an epoch that occurred during validation.

Network arch. – The architecture of the network. Navarro et al.'s architecture was used for many of the experiments. Later, the 4 x 4 filter was replace by a 3x3 filter. The later experiments use one or more ResNets (in parallel).

No. frames – Number of frames. How many frames make up a single state.

No. of actions – The number of actions.

No. slices – The number of slices used.

OOB – Out of bounds behavior. The agent is either given a random action to move it if it tries to leave the image, or the next best action is chosen so it does not leave the slice.

Rewards – The agent is given rewards based on if it takes an action that improves the IoU score or Euclidean distance, or not.

Start mode – How the starting box is initialized by the agent: original (starts as the entire image), scaling (based on the ending location of the other scale), random (random start location), center (starts at the center of the image), mean (starts at the mean location of the training boxes).

Stopping – Signals the termination of a validation episode. Can either be the number of repeated states in a given period, an oscillation pattern or the maximum number of validation steps.

τ – The fraction of each policy network weight that modifies the target network weights. When τ is one, the networks are directly copied.

To calc. targets – Whether to use the target network or use the calculated converged values for labels for the policy network.

Train term. threshold - Training termination threshold. The criteria used to stop a training episode. Euclid. is when the center of the agent's box gets closer to the center of the ground truth than the threshold. Same applies for IoU score.

Val. on train – Whether or not the validation slices were the same as the training slices.

Warm up – Number of random transitions taken to fill up the memory prior to training.