Advances in Artificial Intelligence and Machine Learning; Research 2 (1) 288-302 Received 14-03-22; Accepted 27-03-22; Published 31-03-22

Adaptable Deep Learning and Probabilistic Graphical Model System for Semantic Segmentation

Matthew Avaylon, Robbie Sadre, Zhe Bai
Talita Perciano, TPERCIANO@LBL.GOV
Scientific Data Division, Lawrence Berkeley National Laboratory, USA.

Corresponding Author: Talita Perciano.

Abstract

Semantic segmentation algorithms based on deep learning architectures have been applied to a diverse set of problems. Consequently, new methodologies have emerged to push the state-of-the-art in this field forward, and the need for powerful user-friendly software increased significantly. The combination of conditional random fields (CRFs) and convolutional neural networks (CNNs) boosted the results of pixel-level classification predictions. Recent work using a fully integrated CRF-RNN layer have shown strong advantages in segmentation benchmarks over the base models. Despite this success, the rigidity of these frameworks prevents mass adaptability for complex scientific datasets and presents challenges in optimally scaling these models. In this work, we introduce a new encoder-decoder system that overcomes both these issues. We adapt multiple CNNs as encoders, allowing for the definition of multiple function parameter arguments to structure the models according to the targeted datasets and scientific problem. We leverage the flexibility of the U-Net architecture to act as a scalable decoder. The CRF-RNN layer is integrated into the decoder as an optional final layer, keeping the entire system fully compatible with backpropagation. To evaluate the performance of our implementation, we performed experiments on the Oxford-IIIT Pet Dataset and to experimental scientific data acquired via micro-computed tomography (μ-CT), revealing the adaptability of this framework and the performance benefits from a fully end-to-end CNN-CRF system on both experimental and benchmark datasets.
Keywords: Convolutional Neural Networks, Probabilistic Graphical Models, Conditional Random Fields, Scientific Data.

1. INTRODUCTION

Over the years there have been great strides in developing new deep learning methods for pixel-level labeling problems. In addition to the advancements in convolutional neural networks (CNNs), probabilistic graphical models (PGM), specifically in the form of conditional random fields (CRF), have been effective additions to CNNs as a post-processing step and later as fully end-to-end training processes [1]. Although there are many advancements in segmentation algorithms, there is a lack of user-friendly tools and packages that not only encompass the state-of-the-art, but also introduces new experimental methods that have demonstrated great potential in the literature. At the same

Citation: Matthew Avaylon et al. Adaptable Deep Learning and Probabilistic Graphical Model System for Semantic Segmentation. Advances in Artificial Intelligence and Machine Learning. 2022;2(1):19.

time, the growth in data size and heterogeneity, especially data coming from scientific data facilities across the U.S. Department of Energy (DOE), overwhelms current statistical learning approaches and software tools for analysis. Consequently, scientific discoveries are falling behind the pace of data generation.

Motivated by these challenges and inspired by previous success in implementing an end-to-end CNN-CRF system [2], our goal is to explore the benefits of CRFs on other highly popular CNNs, targeting future efficient analysis of scientific data. Maintaining the original proposed VGG-16 encoder, we created additional encoders using U-Net and ResNet50. The original decoder in performed well [2], but lacked the ability to scale across different encoder options, while also being difficult to adjust to image sizes beyond the original 500×500. To overcome this issue, we implemented a new decoder based on the U-Net architecture: U-Net Decoder. The U-Net based decoder is able to be scaled for various encoder sizes, while also allowing users more options for image dimensions. As in the original end-to-end setup, the CRF-RNN layer is fully integrated, allowing each of these models to be fully capable of backpropagation [3]. These new encoder-decoder combinations were trained on the Oxford-IIIT Pet Dataset and a synthetic binary dataset that simulates data obtained via micro-computed tomography (μ-CT).

When trained on the Oxford-IIIT Pet Dataset, all models showed a clear increase in validation accuracy when compared to their non-CRF counterparts. All models were initially trained using a CPU; however, our CRF-CNN models are GPU compatible, reducing training time to nearly that of the non-CRF models on the same GPU system. We also successfully apply our framework to segment an experimental dataset acquired via μ-CT, showing a promising path towards developing an efficient, accurate, and easy to use segmentation framework for scientific data that is capable of including physical constraints through the embedded CRF model. Our new encoder-decoder system is not only an initial step towards creating a user-friendly pixel-level segmentation package that includes both convolutional and graphical methods, but also further demonstrates the viability of CRFs to improve performance on traditional CNNs. Our framework is freely available at (`https://github.com/mavaylon1/UNet_Decoder_CRF`).

The remainder of this paper is organized as follows. Section 2 reviews the literature relevant to this work. Section 3 explains in details the proposed deep learning and conditional random fields system. Section 4 describes a series of experiments carried out to evaluate the proposed approach. Finally, we draw our conclusions and discuss future work in Section 5.

2. RELATED WORK

In this section, we cover relevant literature regarding some of the recent progress in convolutional methods for semantic segmentation tasks, explore important stepping stones that brought about our encoder-decoder architecture, and lastly look towards the advantages of having a conditional random fields model integrated with CNNs.

There is a wide array of approaches for pixel-level classification in the literature. The success of the U-Net architecture for biomedical imaging have led to new U-Net based structures and more complex image data applications [4]. A common issue in biomedical image segmentation is the data can have varying scales. Punn *et al.* tackled this problem by replacing the U-Net convolutional

layers with the inception layers [5] and by replacing the max-pooling layers with hybrid pooling layer [6]. Lou *et al.* took it further by using inception blocks with dimensional reduction in their novel inception-U-Net architecture to achieve reduced training times over the original inception block framework [7].

He *et al.* introduced residual skip connections, allowing for the creation of deeper classification models that surpassed the then state-of-the-art over a multitude of performance benchmarks on the PASCAL and MS COCO datasets [8,9]. Shortly after, residual skip connections made their way into segmentation architectures. Drozdzal *et al.* expanded fully convolutional networks (FCN), an important milestone of segmentation that will be discussed shortly, by adding residual skip connections to better maintain spatial features [10]. Beyond biomedical imaging, U-Net variations have made their way into other applications. Zhang *et al.* applied U-Net to road extraction, in which they found promising benefits over compared architectures using a U-Net built with residual blocks [11].

Some have ventured into the world of Natural Language Processing (NLP), where the Transformers architecture has become the mainstay [12]. Dosovistkiy *et al.* integrated the original transformer encoder with a Multi-Layer Perceptron (MLP) head for image classification [13]. Requiring large datasets and sufficient computational power, Visual Transformers [14,15], surpassed ResNets of similar size. However, inductive bias allows CNNs to outperform the Visual Transformer on more modestly sized datasets, such as ImageNet. With Visual Transformers and certain CNN-Visual Transformer hybrids still in their infancy [16], convolutional methods are still the first choice for image analysis problems.

The introduction of FCNs brought about a new standard for semantic segmentation [17]. FCNs can be divided into two stages. The first stage consists of an image classification architecture, such as VGG-16, that is adapted towards semantic segmentation tasks by removing the final pooling layer and changing the fully connected layers to convolutional layers. This stage serves to downsample the image, returning specialized feature maps. However, as the models execute convolutions, information regarding spatial information is lost as the models get deeper. The authors developed a solution to this issue with the second stage, a process consisting of upsampling layers and fusing predictions from multiple pooling stages in order to produce sharper final predictions. This decoder framework presents complications with easily adapting models to new image datasets with different dimensions, while also requiring longer training times compared to other methods.

SegNet was one of the first deep convolutional encoder-decoder architectures [18-21], for image segmentation to provide a framework that has the ability to deliver competitive performance metrics, while also having less memory requirements and lower training times [22]. The encoder adapted VGG-16 for their tasks; however, the pooling layer and the fully connected layers were removed, creating a smaller downsampling stage compared to FCN. The biggest development from SegNet was the introduction of a decoder system whose framework mirrored the structure of the encoder. Decoder blocks were matched to corresponding blocks in the encoder, upsampling pool indices and followed by convolutions. The SegNet architecture was a step forward towards creating a scalable encoder-decoder architecture; however, in our framework, we found better performance adopting the U-Net approach of transferring the entire feature map from the final convolution each encoder block, rather than pooling indices. With U-Net as the base for decoder, a wider range of CNNs can be utilized as encoders, while keeping training times low.

PGMs have been explored in the literature as a way to improve deep learning algorithms by adding structural information. For example, Markov random fields (MRF) combined with deep learning was used for image synthesis in [23] and for image segmentation in [24]. CRFs have been adopted to aid CNNs in order to retain an image's features by favoring assigning the same labels to similar pixels without focusing on purely smoothing the image. Initially, CNNs were trained separately from the CRF stage, using a fully connected CRF as a post-processing step to refine the CNN outputs [1]. Although having a disconnected CRF with CNNs have shown increased performance over their pure CNN cores, Zheng *et al.* demonstrated on the Pascal VOC 2012 dataset [25], that having an end-to-end trainable CNN-CRF architecture will outperform the former.

In this work, we propose an efficient and easy-to-use unified framework that enables the combination of different FCNs with a conditional random field model. Differently from the literature, we adapt multiple CNNs as encoders, allowing for the use of multiple parameter arguments to structure the models according to the problem at hand. Moreover, the CRF layer is conveniently available as an optional final layer, always keeping the entire system fully compatible with backpropagation. Next section explains in details our proposed encoder-decoder system.

3. PROPOSED ENCODER-DECODER SYSTEM

3.1 Convolutional Neural Networks

The proposed framework uses three main deep learning architectures for the encoder part of the network: ResNet-50, VGG-16, and U-Net. Residual networks [8], significantly improved performance of neural network architectures due to the introduction of residual blocks, which make use of residual skip connections to combat the vanishing gradient problem in deep learning. When networks get excessively large, the gradients that are backpropagated to early layers become minuscule, making it difficult to learn. This sets a limit on the depth of a network. On the other hand, with residual blocks allow the model to easily represent the identity function if needed, preserving the computed gradients.

The second encoder architecture used is VGG-16 [26]. This is a classic neural network architecture used for image recognition. It improved upon the then state-of-the-art AlexNet by reducing the number of hyperparameters via smaller kernel filters throughout the convolutional layers. This reduction in parameters allowed for faster training times.

The final encoder architecture is U-Net [27]. Among assorted models used for medical segmentation [28], the architecture can be divided into two key stages: the encoder and the decoder. The encoder is a stack of blocks consisting of convolutional and max-pooling layers, with each block learning more complex features. The decoder consists of blocks that expand the output from the bottleneck. Each block consists of convolutional and upsampling layers; however, the input of each block is a concatenation of the corresponding feature map from the encoder stage and the output from the previous decoder block. The skip connections help to maintain details from the feature maps that were lost through downsampling. This U-shape framework forms the premise of our encoder-decoder system. Besides the use of these three deep learning architectures, we also embed

a conditional random field model to form an end-to-end trainable pipeline. We explain this model in details in the following.

3.2 Conditional Random Fields

A Probabilistic Graphical Model (PGM) [29], is a data structure for encoding probability distributions. The nodes of the graph represent random variables, and the edges represent the dependencies between variables. Undirected PGMs are known as Markov random fields (MRF) or conditional random fields (CRF). Those models have been used to tackle inference, prediction, and analysis problems in computer vision [30], robotics, medicine, natural language processing, and many others. In MRFs and CRFs, the Maximum A Posteriori (MAP) is an inference problem, defined as follows. Let $\mathbf{x} \in \mathcal{X} = \mathcal{X}_1 \times \cdots \times \mathcal{X}_n$ denote an assignment to n discrete random variables $X_1, \ldots, X_n$ where each variable X_i takes values in a finite set of states (or labels) $\mathcal{X}_i$ and it is associated to a pixel i in an image $\mathbf{I}$. The set of labels is pre-defined as $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$. Let $\mathcal{G}$ be a graph of n nodes with the set of cliques C. Consider an MRF that represents a joint distribution $p(\mathbf{x}) := p(X_1, \ldots, X_n)$ factorizing over $\mathcal{G}$, i.e., $p(\cdot)$ takes the form:

$$p(\mathbf{x}) = \frac{1}{Z} \prod_{c \in C} \psi_c(\mathbf{x}_c) \quad \forall \mathbf{x} \in \mathcal{X}, \tag{1}$$

where $\mathbf{x}_c$ is the joint configuration of the variables in the clique c, ψ_c are positive functions called potentials, and $Z = \sum_{\mathbf{x}} \prod_{c \in C} \psi_c(\mathbf{x}_c)$ is a normalization factor called partition function. The MAP inference problem consists of finding the most probable assignment to the variables, i.e.:

$$\mathbf{x}^* \in_{\mathbf{x} \in \mathcal{X}} p(\mathbf{x}) =_{\mathbf{x} \in \mathcal{X}} \prod_{c \in C} \psi_c(\mathbf{x}_c). \tag{2}$$

For each clique c, let $\mathcal{X}_c = \prod_{i \in c} \mathcal{X}_i$ be the set of its joint configurations and define $f_c(\mathbf{x}_c) = -\log \psi_c(\mathbf{x}_c) \quad \forall \mathbf{x}_c \in \mathcal{X}_c$. It is straightforward that the MAP inference problem (2) is equivalent to minimizing the energy of the MRF, $E(\mathbf{x}; \mathbf{I}) = \sum_{c \in C} f_c(\mathbf{x}_c)$. The degree of the potential $\psi_c(\cdot)$ is the size of the corresponding clique c ($|c|$). For example, a pairwise potential has $|c| = 2$. For this type of MRFs, the energy only consists of potentials of degree one and two:

$$E(\mathbf{x}; \mathbf{I}) = \sum_{i \in \mathcal{V}} \psi_i(X_i; \mathbf{I}) + \sum_{(i,j) \in \mathcal{E}} \psi_{i,j}(X_i, X_j), \tag{3}$$

where $\mathcal{V}$ denotes the set of nodes of the graph $\mathcal{G}$ and $\mathcal{E}$ is the set of pairs of variables that interact with each other. The pairwise potential $\psi_{i,j}(X_i, X_j)$ in (3) does not depend on the observed data. If so, then we have a pairwise CRF model where the second term of the equation above is now $\sum_{(i,j) \in \mathcal{E}} \psi_{i,j}(X_i, X_j; \mathbf{I})$.

The unary energy, $\psi_i(X_i; \mathbf{I})$, measures the cost of the pixel i taking the label $\mathcal{X}_i$, and the pairwaise energy, $\psi_{i,j}(X_i, X_j; \mathbf{I})$ measures the cost of pixels i and j taking the labels $\mathcal{X}_i$ and $\mathcal{X}_j$ at the same time. As in [2], in our framework, the unary energy comes from a CNN. The pairwise energy is a smoothing image data-dependent term that encourages assigning similar labels to pixels with similar characteristics. As in [2,31], the pairwise potentials in our framework are modeled as weighted Gaussians:

$$\psi_{i,j}(X_i, X_j; \mathbf{I}) = \mu(X_i, X_j) \sum_{m=1}^{M} w^{(m)} k_G^{(m)}(\mathbf{f}_i, \mathbf{f}_j), \tag{4}$$

where each $k_G^{(m)}$ for $m = 1, \ldots, M$, is a Gaussian kernel applied on feature vectors, which in our case consist of spatial location and pixels gray-level or RGB values [31]. The function $\mu(\cdot, \cdot)$, as in [2], is called the compatibility function and captures the compatibility between different pairs of labels.

In order the find the most probable configuration (pixel-based segmentation), the maximization problem in Eq. (2) can be written as a minimization problem using Eq. (3):

$$\mathbf{x}^*(\mathbf{I}) =_{\mathbf{x} \in \mathcal{X}} E(\mathbf{x}; \mathbf{I}). \tag{5}$$

Finding the exact solution of Eq. (5) is an intractable problem. Consequently, a mean-field approximation to the CRF distribution is used to approximate the MAP marginal inference [2], which approximates the CRF distribution $p(\mathbf{x})$ by a simpler distribution $q(\mathbf{x}) = \prod_i q_i(X_i)$.

The mean-field CRF inference can be reformulated as a Recurrent Neural Network (RNN), as proposed in Zheng, et al [2]. In this case, one iteration of the algorithm can be reproduced by a series of common CNN operations. The bottom of FIGURE 2 shows the operations involved in one iteration of the mean-field algorithm. We summarize below which CNN operations are equivalent to each mean-field operation. For more details, please refer to [2]:

1. *Initialization*: Application of a softmax function over the unary potentials coming from the CNN across all the labels at each pixel;
2. *Message Passing*: Application of M Gaussian filters based on Eq. (4) using permutohedral lattice [32];
3. *Weighting Filter Outputs*: Obtained via convolution with a 1 filter with M input channels and one output channel;
4. *Compatibility Transform*: Uses the Potts model as compatibility function, $\mu(l, l') = [l, l']$, where $[\cdot]$ is the Iverson bracket. This step can be achieved via convolution where the spatial receptive field of the filter is 1×1, and the number of input and output channels are both L.
5. *Unary Addition*: Achieved by subtracting the output from the compatibility transform stage from the unary inputs (coming from the CNN);
6. *Normalization*: Achieved via another softmax operation.

The end-to-end trainable process is implemented by repeating the above stack of layers so that each iteration takes value estimates from the previous iteration and the unary values in their original form as a RNN. This CRF-RNN layer is already implemented in our framework and it is available for the users as an optional step as explained in the next section.

3.3 System Architecture

Aiming to provide a flexible framework for semantic segmentation using CNNs and CRFs, we implemented different convolution networks embedding the CRF model in an end-to-end trainable

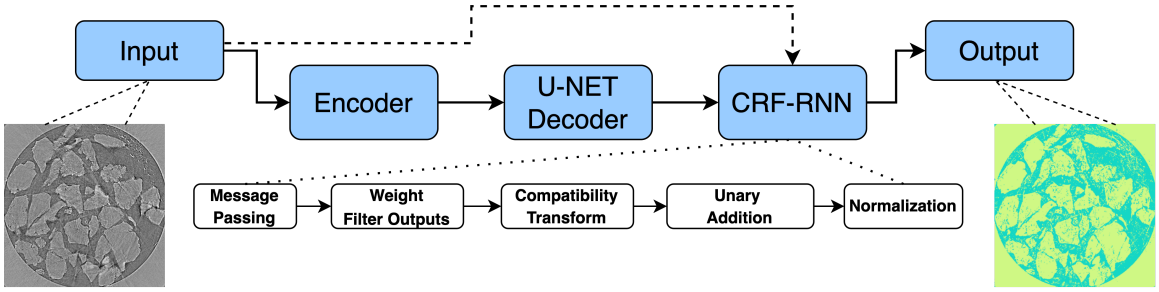

Figure 1: A general depiction of our encoder-decoder framework with an integrated CRF-RNN layer. The input data is fed into a chosen encoder, which then is upsampled by the U-Net based decoder. Both the decoder output and the initial image serve as inputs for the CRF-RNN layer, which produces the final pixel-based prediction.

system by creating a convolutional encoder-decoder system using a U-Net based decoder. The software architecture consists of three main components: **encoders**, **decoders**, and **model utilities**.

There are separate encoders for each supported convolutional method: U-Net, VGG-16, ResNet34, ResNet50, and ResNet101. The encoders for VGG-16, ResNet34, ResNet50, and ResNet101 were built according to the original implementations; however, we discarded the final pooling and fully connected layers of VGG-16 and of all ResNet encoders. This allows for the reduction of the number of parameters in the models. The U-Net architecture is easily scalable, allowing users great freedom to dictate hyperparameters. We have embodied this adaptability with the U-Net encoder, allowing users to set the model's depth, number of filters, and a choice of batch normalization. The Batch Normalization layer normalizes the output from the previous layer, combating the change in the data distribution, i.e internal covariate shift and allowing the network to be trained more efficiently. In the original Batch Normalization paper, Ioffe *et al.* places the layer before the non-linear activation functions [33]. However, it is still a topic of debate of whether it is better to use Batch Normalization after the nonlinear functions. Consequently, in our system, we decided to create a function argument for this placement so that the user has the ability to change it as needed. Each encoder returns a list of feature maps, which is used for concatenation in the U-Net-Decoder. Each encoder supports image shape and channel arguments. In the future, ResNet will be conglomerated into a generalized encoder similar in format to our U-Net encoder.

As mentioned prior, FCNs are a robust tool that take inputs of varying sizes and produce segmentation results. However, it requires users to delve into adjusting Cropping layers to fit different image sizes. The U-Net based decoder have greatly simplified the upsampling process by allowing users to connect the encoder by defining corresponding dimensions within the decoder parameters. The U-Net-Decoder takes in the encoder as an input, creating blocks that will concatenate with the feature maps of corresponding shape as illustrated in FIGURE 1. These blocks are the same blocks from U-Net, consisting of the same flexible setup as defined in the U-Net encoder. However, the decoder function gives the user the choice between using an Upsampling Layer or a Transposed Convolutional Layer, providing users more options when it comes to saving computing time.

The role of model utilities is to act as a reservoir of helper functions and tools. It contains the blocks used to build the encoders and decoders, having these blocks exist as separate functions allows users

to create their own architectures if desired. Model utilities also houses the original adaptation of the CRF as an RNN. Users are not required to implement the CRF-RNN layer into their models; it is set as a boolean argument in the decoder. In the next section, we present a series of experiments using our proposed framework.

4. RESULTS

4.1 Datasets and Training

Although previous related work were trained using the VOC 12 dataset [34,2], we sought to see the benefits of conditional random fields using experimental data, specifically data from micro-computed tomography. Using experimental data is closer aligned with research scenarios in disciplines outside of computer vision benchmarking. For the first set of experiments, we selected the synthetic dataset from the 3D benchmark made available by the Network Generation Comparison Forum (NGCF) [1]. The NGCF datasets are a global, recognized standard to support the study of 3D tomographic data of porous media. The datasets provided are binary representations of a 3D porous media. For the purposes of our experiments, we corrupted the original data with noise (salt-and-pepper) and additive Gaussian with $\sigma = 100$. Additionally, we also simulate ringing artifacts [35], into the sample to closer resemble real-world data. For the pixel-level segmentation analysis, the corrupted data serves as the "original data" and the binary stack as the ground-truth. A full synthetic dataset is 268 MB in size, and consists of 512 image slices of dimensions 512×512. Among the four datasets provided by NGCF, we chose two of them named Castle and BeadPack (FIGURE 2). The Castle data simulate a sample of Castlegate outcrop sandstone, from southeastern Utah, USA. The BeadPack data simulate a packing of silica spheres. In our experiments, we found that an equal mix of cross-sections (2D slices) of BeadPack and Castle returned the highest validation accuracy. The weights from the models on this combined dataset were used to predict the segmentation of an experimental data, Sandstone, described later. The total size of the training and validation sets is 812 and 206 images respectively.

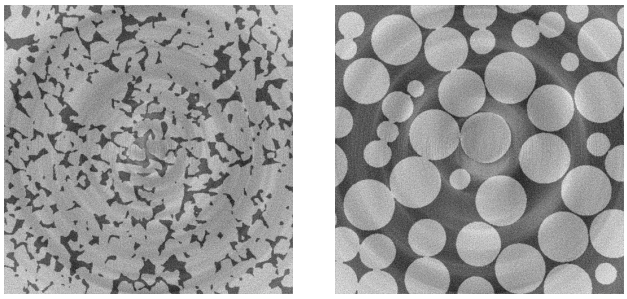

Figure 2: Examples of cross-sections coming from the two simulated datasets (after noise addition) used for the training/validation sets: Castle (a) and BeadPack (b).

The Sandstone data does not have a ground-truth set, an issue quite common in experimental datasets, and so we used the model weights trained from the binary dataset to generate segmentation pre-

[1] `http://people.physics.anu.edu.au/~aps110/network_comparison`

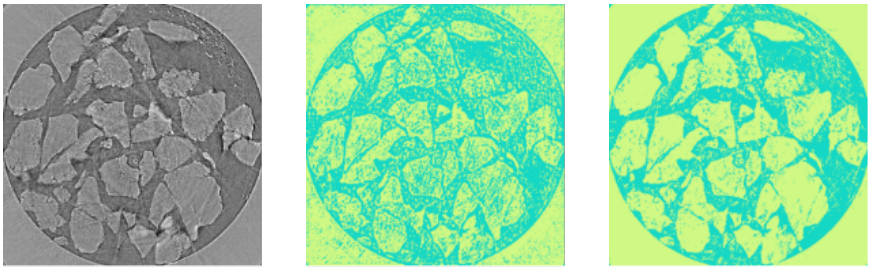

Figure 3: (a) Example of sandstone cross-section. (b) Prediction from the U-Net architecture without the CRF-RNN layer. (c) Prediction from the U-Net architecture with the CRF-RNN layer.

dictions of this experimental data. The data consists of cross-sectional slices (1813×1830) of a geological sample (sandstone), where the gray-scale values corresponds to the x-ray attenuation and density of the material. This data was generated by the Lawrence Berkeley National Laboratory Advanced Light Source X-ray beamline 8.3.2 [2] [36]. As seen in FIGURE 3, we segment this dataset in two classes: the sandstone and the matrix (the background of the sandstone sample). The addition of the CRF model leads to a much better label compatibility (FIGURE 3c), i.e., the segmentation of regions pertaining to the sandstone class is more homogeneous compared to the results without using the CRF model (FIGURE 3b). This is explained by the neighborhood information (prior knowledge) embedded into the CRF model via the graph-based representation: pixels in the same neighborhood are more likely to pertain to the same class.

The final dataset used in our experiments is the Oxford-Pet IIIT [37], which contains images that have large variations in scale and positioning for 37 different animal breeds, providing a sufficient increase in difficulty over the binary case. We used the pixel level trimap as the ground-truth and normalized the images for training. Based on this dataset, we evaluate how adding a CRF-RNN layer affects the model's performance with a difficult border class. This border class is not a structure within the images themselves, rather an artificial region of both foreground and background existing only on the ground-truth set. It can be viewed as a region of uncertainty. This dataset is also publicly available, allowing future users an easy starting point when testing our framework. FIGURE 4 shows an example of an image and its ground-truth coming from this dataset. The segmentation using U-Net with the CRF model (FIGURE 4d) successfully identifies fine details pertained to the foreground class.

All models were trained using the Adam optimizer with a fixed learning rate of 0.001, using the standard softmax loss function as in [38]. The weights for each model were initialized using He normal initialization [39], while the CRF-RNN layer was kept the same as the original implementation from [2]. For the training process, we used categorical cross-entropy for the loss function and both accuracy and IoU as our evaluation metrics. To have a base comparison of our U-Net decoder, we also trained the VGG-FCN-8 model to evaluate its performance against VGG-U-Net. With the trend that deeper ResNets tend to outperform smaller implementations, we trained ResNet101 to see whether a deeper model will outperform a smaller ResNet with the CRF-RNN layer, i.e ResNet50.

[2] `microct.lbl.gov`

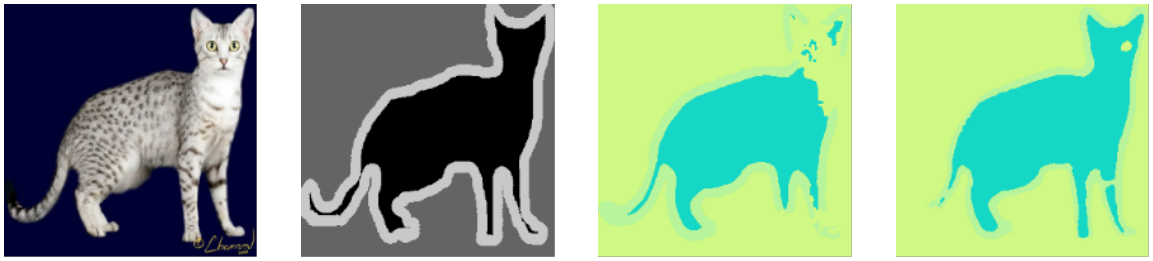

Figure 4: (a) Example of Oxford-Pet IIIT Data input image. (b) Example of Oxford-Pet IIIT Data input image groundtruth. (c) Prediction using the U-Net framework without the CRF-RNN layer. (d) Prediction using the U-Net framework with the CRF-RNN layer.

Users have the option between using the Batch Normalization either before or after the activation function. We found that having Batch Normalization before our activation (ReLU) yields better performance than having it post-activation. We also found that adding Batch Normalization before the CRF-RNN layer resulted in less training accuracy oscillations, allowing for faster convergence.

4.2 Evaluation

We provide two metrics to evaluate the models: accuracy and IoU. We found with the synthetic binary dataset, regardless of the presence of the CRF-RNN layer, all models reported an accuracy above 97%. However, we also found that VGG-16 with our U-Net-Decoder required less than half the epochs of the VGG-16 with the FCN-8 decoder to converge in both the CRF and non-CRF versions. The addition of the CRF-RNN layer, though did not improve accuracy, reduced the number of epochs by half when compared to their core counterparts.

As mentioned prior, we used the weights trained on the synthetic binary dataset to create segmentation predictions for the Sandstone data. As seen in FIGURE 3, the addition of the CRF preserves both the sandstone and the matrix regions more so than the base models. The CRF was also able to create predictions that carry over finer sandstone regions from the original image.

With a more complex dataset consisting of more classes, difficult borders, and variations in scale, we see a clear increase in accuracy of the CRF models over the base architectures when trained on the Oxford-Pet IIIT data. Similar to our preliminary tests with the binary dataset, we compared VGG-16 with our U-Net-Decoder and VGG-16 with the FCN-8 decoder. The VGG-16 with our U-Net decoder reported a higher accuracy for both the CRF and non-CRF versions when compared to the FCN implementation. The addition of the CRF-RNN layer to both decoder versions increased the accuracy by 1.74% and 2.27% respectively (TABLE 1). The increase in mIoU score with the addition of a CRF-RNN layer is similar between the FCN and U-Net decoder methods; however, the VGG-16 model with the U-Net decoder performed 10 points higher than the FCN version (TABLE 1).

For the remaining encoders, we conducted tests using just the U-Net decoder. U-Net performed the best out of all the architectures in both metrics, while ResNet50 saw the greatest increase from non-CRF to the CRF version for both accuracy and mIoU. Additionaly, we explore how the CRF-

Table 1: The validation accuracy and mIoU of each model that was trained using the Oxford-Pet IIIT dataset. ResNet101 was not trained with the CRF-RNN layer.

Model	Non-CRF		CRF	
	Accuracy	mIoU	Accuracy	mIoU
ResNet50	84.52	63.25	86.58	66.22
U-Net	86.29	66.67	87.97	70.36
VGG-U-Net	85.82	67.67	87.56	69.29
VGG-FCN	81.23	58.09	83.50	59.25
ResNet101	85.33	65.66	-	-

Table 2: The individual IoU scores of each class for each of the models that were trained on the Oxford-Pet IIIT dataset. All models, except VGG-FCN, were trained using the U-Net-Decoder. ResNet101 only has scores for the Non-CRF version in order to compare with ResNet50+CRF. Class 0: Foreground, Class 1: Background, Class 2: Border.

Model	Non-CRF			CRF		
	class 0	class 1	class 2	class 0	class 1	class 2
ResNet50	67.86	81.92	40.01	75.67	84.90	38.09
U-Net	73.83	84.01	42.14	78.81	87.76	44.51
VGG-U-Net	73.92	84.15	44.93	79.91	87.69	40.27
VGG-FCN	65.87	78.38	30.02	69.51	81.62	26.63
ResNet101	71.29	82.81	42.96	-	-	-

RNN implementation could compete against pure larger models. Our findings show that Resnet50 with the CRF-RNN layer outperforms the standard ResNet101, reporting an improvement of 1.25% in accuracy and .56 points in mIoU. We see larger benefits when looking at the individual IoU scores for each class: foreground (class 0), background (class 1), and border (class 2). Here we see the smaller ResNet50 with the CRF-RNN layer has an increase of 4.39 and 2.09 points for the foreground and background classes respectively (TABLE 2).

When looking at the individual class IoU scores, the addition of the CRF-RNN layer improves the foreground and background classes; however, the border class score decreases. Within the CRF-RNN layer, the compatibility transform used consists of the Potts model, which penalizes equally across all labels for pixels with similar features. As a result, the penalty of having a mislabeled background and foreground pixel next to each other is the same as having a border pixel next to a pixel that was mislabeled as foreground or background. The images themselves do not have a border, and so the foreground and background are in fact touching each other according to the input images. The CRF with the Potts model was not able to effectively penalize mislabeling the border class provided by the ground-truth, rather it followed the class properties of the input image. The Non-CRF models were able to pick up the border distinction more by at most 4 points. Where the CRF models did succeed over the base implementations is in regards to the foreground and background

classes. The CRF models were able to preserve more of the animals structure, specifically the smaller limb regions as shown in FIGURE 4.

4.3 GPU implementation

The proposed framework can be also executed on GPUs. In order to provide initial results on GPU, we run experiments on the Cori system at NERSC, which contains NVIDIA V100 cards [3]. When comparing the training performance of a dual-socket, 20-core Xeon (Skylake) against eight Nvidia V100 GPUs, we observe that the eight GPUs delivered 3-4× better performance. For instance, on the synthetic binary data, the eight GPUs require an average of 2 seconds per step and 156 minutes of total training time compared to 6 seconds per step and 486 minutes of total time on the Xeons. When it comes to the Oxford-Pet IIIT data, the GPU-accelerated node completed 20 epochs of training in 2050 minutes (34 hours). This reduced training time by two days when compared with the CPU-based implementation while attaining a statistically similar accuracy of 87% for the U-Net-CRF model.

5. CONCLUSION AND FUTURE WORK

In this work, we presented a flexible Python-based encoder-decoder framework, U-Net-Decoder, that allows users to use different types of CNNs along with a CRF-RNN layer in an end-to-end training process for semantic segmentation. Our main goal was to introduce a new framework that can be adapted towards a wide range of CNN architectures and demonstrate the benefits of a fully integrated CRF-RNN layer for semantic segmentation, using both popular datasets in the community and experimental datasets. We demonstrated that our U-Net-Decoder decreases training time, while also surpassing previous encoder-decoder setups in performance. Models with the CRF-RNN layer performed significantly better than their core versions in accuracy, mIoU, and IoU scores for the foreground and background classes, adding credence to the benefits of the CRF-RNN.

In the future, we seek to expand the software to be fully compatible with 3D images and higher-dimensional data to target complex scientific datasets. Larger and more complex datasets will lead to higher computational complexity, which we will overcome by improving our GPU implementation and taking advantage of parallel and distributed training. We aim to provide an approach that can be adopted to solve problems for a range of scientific fields. We also plan to keep usability as a primary goal for the software, adding improvements to how users interact and shape these tools to their needs with each content expansion. The current implementation of conditional random fields as an RNN provides significant benefits; however, adjusting the compatibility transform to account for different weight penalization, in additional to learning the weights of a combination of different filters, will allow for better results over the simplistic Potts model.

[3] `https://docs.nersc.gov/systems/cori/`

6. CONFLICT OF INTEREST

The authors declare no competing interests.

7. ACKNOWLEDGEMENT

This work was supported by the Laboratory Directed Research and Development Program of Lawrence Berkeley National Laboratory under U.S. Department of Energy Contract No. DE-AC02-05CH11231.

References

[1] Chen LC, Papandreou G, Kokkinos I, Murphy K, Yuille AL. Deeplab: Semantic Image Segmentation With Deep Convolutional Nets, Atrous Convolution, and Fully Connected Crfs. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017; 40:834-848.

[2] Zheng S, Jayasumana S, Romera-Paredes B, Vineet V, Su Z, et al. Conditional Random Fields as Recurrent Neural Networks. 2015 IEEE International Conference on Computer Vision (ICCV), Dec 2015.

[3] Lecun Y, Bottou L, Bengio Y, Haffner P. Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE. 1998;86:2278–2324.

[4] Çiçek Ö, Abdulkadir A, Lienkamp SS, Brox T, Ronneberger O. 3D U-Net: Learning Dense Volumetric Segmentation From Sparse Annotation. Medical Image Computing and Computer-Assisted Intervention – MICCAI. 2016; 424–432.

[5] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, et al. Going deeper with convolutions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).2015:1–9.

[6] Punn NS, Agarwal S. Inception U-Net Architecture for Semantic Segmentation to Identify Nuclei in Microscopy Cell Images. ACM Trans. Multimedia Comput. Commun. Appl. 2020;16:1-15.

[7] Lou A, Guan S, Loew M. DC-UNet: rethinking the U-Net architecture with dual channel efficient CNN for medical image segmentation. In Ivana Išgum and Bennett A. Landman, editors, Medical Imaging 2021: Image Processing. volume 11596:758-768. International Society for Optics and Photonics, SPIE, 2021.

[8] He K, Zhang X, Ren S, Jian S. Deep Residual Learning for Image Recognition. In Proceedings of the Ieee Conference on Computer Vision and Pattern Recognition. 2016;770-778.

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona,et al. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision – ECCV 2014, pages 740–755, Cham, 2014. Springer International Publishing.

[10] Drozdzal M, Vorontsov E, Chartrand G, Kadoury S, Chris Pal. The Importance of Skip Connections in Biomedical Image Segmentation. Deep Learning and Data Labeling for Medical Applications. 2016;179–187.

[11] Zhang Z, Liu Q, Wang Y. Road Extraction by Deep Residual U-Net. Ieee Geoscience and Remote Sensing Letters. 2018;15:749–753.

[12] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, et al. Attention Is All You Need. Advances in Neural Information Processing Systems 30 (NIPS 2017). 2017.

[13] Dosovitskiy A,Beyer L, Kolesnikov A, Weissenborn D, Xiaohua Zhai, et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021.

[14] Zheng S, Lu J, Zhao H, Zhu X, Luo Z, et al.Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 6877–6886. IEEE, 2022.

[15] Strudel R, Garcia R, Laptev I, Schmid C. Segmenter: Transformer for Semantic Segmentation. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV).2021:7242–7252.

[16] Chen J, Lu Y, Yu Q, Luo X, Adeli E, et al. TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation, 2021.

[17] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2015: 3431-3440.

[18] Noh H, Hong S, Han B. Learning Deconvolution Network for Semantic Segmentation. In 2015 IEEE International Conference on Computer Vision (ICCV), pages 1520–1528, Los Alamitos, CA, USA, dec 2015. IEEE Computer Society.

[19] Jun Fu, Jing Liu, Yuhang Wang, Jin Zhou, Changyong Wang, and Hanqing Lu. Stacked Deconvolutional Network for Semantic Segmentation. IEEE Transactions on Image Processing.2019:1–1, .

[20] Chaurasia A, Culurciello E. Linknet: Exploiting Encoder Representations for Efficient Semantic Segmentation. 2017 IEEE Visual Communications and Image Processing (VCIP). Dec 2017.

[21] https://arxiv.org/pdf/1711.08506.pdf

[22] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoderdecoder architecture for image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence.2017;39:2481–2495.

[23] Li C, Wand M. Combining Markov Random Fields and Convolutional Neural Networks for Image Synthesis. In Proceedings of the IEEE conference on computer vision and pattern recognition. 2016; 2479–2486.

[24] Liu Z, Li X, Luo P, Loy C, Tang X. Deep Learning Markov Random Field for Semantic Segmentation. IEEE Transactions on Pattern Analysis Machine Intelligence. 2018;40:1814-1828.

[25] Everingham M, Gool LV, Williams C, Winn J, Zisserman A. The Pascal Visual Object Classes (VOC) Challenge. International Journal of Computer Vision. 2010; 88:303-338.

[26] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. In International Conference on Learning Representations, 2015.

[27] Ronneberger O, Fischer P, Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer. 2015; 234–241.

[28] Milletari F, Navab N, Ahmadi SA. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. In 2016 Fourth International Conference on 3D Vision (3DV).2016:565–571.

[29] Daphne Koller and Nir Friedman. Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning. The MIT Press, 2009.

[30] Plath N, Toussaint M, Nakajima S. Multi-Class Image Segmentation Using Conditional Random Fields and Global Classification. In Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09, New York, NY, USA, 2009. Association for Computing Machinery.2009: 817–824.

[31] Krähenbühl P, Koltun V. Efficient Inference in Fully Connected Crfs With Gaussian Edge Potentials. In Proceedings of the 24th International Conference on Neural Information Processing Systems, NIPS'11, , Red Hook, NY, USA, 2011. Curran Associates Inc. 2011;109–117.

[32] Adams A, Baek J, Davis MA. Fast High-Dimensional Filtering Using the Permutohedral Lattice. Computer Graphics Forum, 2010;29.

[33] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. Proceedings of the 32nd International Conference on Machine Learning. 2015;37:448-456.

[34] George Papandreou, Liang-Chieh Chen, Kevin P. Murphy, and Alan L. Yuille. Weakly-and Semi-Supervised Learning of a Deep Convolutional Network for Semantic Image Segmentation. In 2015, IEEE International Conference on Computer Vision (ICCV).2015:1742–1750.

[35] Perciano T, Ushizima D, Krishnan H, Parkinson D, Larson N, et al. Insight Into 3D Micro-CT Data: Exploring Segmentation Algorithms Through Performance Metrics. Journal of Synchrotron Radiation. 2017;24:1065–1077.

[36] Donatelli JJ, Zwart PH, Sethian JA. Iterative Phasing for Fluctuation X-Ray Scattering. Proceedings of the National Academy of Sciences. 2015;112:10286–10291.

[37] OM, Vedaldi A, Zisserman A, Jawahar CV. Cats and Dogs. In 2012 IEEE Conference on Computer Vision and Pattern Recognition, pages. 2012;3498–3505.

[38] Krähenbühl P, Koltun V. Parameter Learning and Convergent Inference for Dense Random Fields. In Proceedings of the 30th International Conference on International Conference on Machine Learning. 2013;28: 513–521.

[39] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. In 2015 IEEE International Conference on Computer Vision (ICCV).2015:1026–1034.