J-Net: A Low-Resolution Lightweight Neural Network for Semantic Segmentation in the Medical field for Embedded Deployment

No Author Given

No Institute Given

Abstract. When deploying neural networks in real-life situations, the size and computational effort are often the limiting factors. This is especially true in environments where big, expensive hardware is not affordable, like in embedded medical devices, where budgets are often tight. State-of-the-art proposed multiple different lightweight solutions for such use cases, mostly by changing the base model architecture, not taking the input and output resolution into consideration. In this paper, we propose the J-Net architecture that takes advantage of the fact that in hardware-limited environments, we often refrain from using the highest available input resolutions to guarantee a higher throughput. Although using lower-resolution input leads to a significant reduction in computing and memory requirements, it may also incur reduced prediction quality. Our J-Net architecture addresses this problem by exploiting the fact that we can still utilize high-resolution ground-truths in training. The proposed model inputs lower-resolution images and high-resolution ground truths, which can improve the prediction quality by 5.5% while adding less than 200 parameters to the model. We conduct an extensive analysis to illustrate that J-Net enhances existing state-of-the-art frameworks for lightweight semantic segmentation of cancer in MRI images. We also tested the deployment speed of state-of-the-art lightweight networks and J-Net on Nvidia's Jetson Nano to emulate deployment in resource-constrained embedded scenarios. The framework is open-source and accessible online at `https://BlindedLinkForReview`.

Keywords: Semantic Segmentation · Lightweight · Embedded Deployment · CAD · Computer Vision

1 Introduction

Over time, AI has found many use cases in the medical field, like EEG analysis [1], improvement of MRI image resolution [2,3], and detecting diseases in medical images. Above all, the detection and localization of cancer have gotten much attention from the AI research community. Hence, many methods were proposed for Computer Aided Diagnostics (CAD), starting exclusively with image classification in the early days [4] to the more fine-grained bounding boxes [5,6], and semantic segmentation predictions [7]. Semantic Segmentation predictions consist of a pixel-wise classification for the input image, giving the most

detailed degree of localization and, thus, the most helpful information for the doctors.

The U-Net architecture has established itself as the most popular deep-learning model for semantic segmentation in the medical field. This architecture proved especially successful in medical applications since the network's encoding part efficiently captures the relevant information in the image, like most segmentation architectures. However, the subsequent decoding part up-samples the compressed input back to a higher resolution, which preserves the spatial information. Nevertheless, if we strive for real-world applicability, we can not focus only on prediction quality but also need to put emphasis on reducing the cost of application. On the one hand, it is convenient to make predictions on video streams, which require a throughput rate of at least one image per second and preferably around 20 images per second. On the other hand, being able to make predictions on small, efficient, and cheaper hardware is more than desired. Research has already proposed several improvements and iterations on U-Net [8] like U-Net3+ [9] or ELU-Net [10], which partly focus on throughput rate. In particular, ELU-Net achieved higher throughput at lower overall memory consumption while improving the original network's prediction quality.

However, the proposed lightweight networks are often frigid. When they fail the set target, the end-user does not have much freedom to change the architecture to turn the dial more toward throughput speed at the cost of prediction quality. The go-to strategy in this scenario is to reduce the resolution of the input images. Reducing the input image resolution is very effective in meeting strict hardware limitations. For example, by reducing the input resolution from 320×320 to 160×160, we cut the memory consumption and number of operations by 75%. However, this comes also a considerable loss in prediction quality.

To address this issue, we exploit two observations. First, most computations in a U-Net-like architecture stem from encoding the input image and not so much from the up-scaling in the decoder part of the network. Second, since we down-scaled the original input, we still have access to the high-resolution ground-truths. Hence, we propose our J-Net architecture designed to take advantage of this scenario. J-Net extends any U-Net-like architecture by adding more up-scaling layers at the end of the U-Net, therefore outputting a higher resolution than the input image. Those additional layers do not contribute much toward the overall computational complexity of the model but help the model to leverage the more informative high-resolution ground truth, thus improving the prediction quality of the network compared to only using low-resolution input and ground truth. To further boost the prediction quality, we propose a loss function using the output from all up-scaling steps, from the input resolution to the ground truth resolution. The loss values generated from the different up-scaling steps are helpful in guiding the model while not adding any significant computations.

Our experiments have shown that J-Net significantly improves the prediction quality of the baseline when using the same input resolution while not adding any significant computations. Moreover, we compared J-Net's performance to comparable U-Net architectures on Nvidia's Jetson Nano to show its viability

in resource-constraint embedded scenarios. Additionally, we compare the memory usage and computational complexity of the U-Net variants to our proposed architecture. Moreover, we conducted extensive experiments on the Decathlon prostate dataset [11] and the BraTs 2020 dataset [12] to prove the effectiveness of the proposed framework in various experimental settings and compare them with state-of-the-art techniques to illustrate the benefits of our approach.

The key contributions of this work are:

1. J-Net network improves the prediction quality on 16×16 input resolution on the Decathlon dataset by 5.5% and on the BraTS dataset reaches almost the same scores as the baseline using 32×32 input resolution while increasing the throughput rate only by 0.1 images per second.
2. We can extend any pre-existing U-Net-like architecture to a J-Net to use higher-resolution ground truths efficiently.
3. We present detailed ablation studies and analysis of the results compared our framework to comparable segmentation methods on the Decathlon prostate dataset and BraTS 2020 to evaluate our method's efficacy.
4. This work is accessible online at `https://BlindedLinkForReview`.

2 Related Works

This section discusses the current state-of-the-art U-Net variations.

First, the basic U-Net gained enormous popularity in the biomedical field and thus is usually the starting point for any researcher trying to perform semantic segmentation of medical images. Compared to other popular semantic segmentation networks, like ResNet [13], U-Net [8] not only consists of an encoder part, but also introduces a decoder part, which up-samples the compressed feature maps.

Table 1. Comparison of state-of-the-art U-Net based semantic segmentation for medical imaging.

Methods	Efficiency	Resolution	Loss	Architecture
ACU-Net [14]	✗	✗	✗	✓
MH U-Net [15]	✗	✗	✗	✓
E1D3 U-Net [16]	✗	✗	✗	✓
Peiris et al. [17]	✗	✗	✓	✗
Zabihollah et al. [18]	✗	✗	✗	✓
U-Net++ [19]	✗	✗	✗	✓
U-Net3+ [9]	✓	✗	✓	✓
ELU-Net [10]	✓	✗	✓	✓
J-Net (Ours)	✓	✓	✓	✓

However, the original U-Net was proposed in 2015, and since then, much progress has been made in deep learning. Therefore, several extensions to U-Net were proposed: ACU-Net [14] replaces the convolutional layers of U-Net

with deep separable convolutional layers. Moreover, Tan et al. utilize residual skip connections and an active contour model. MH U-Net [15] is a multi-scale hierarchical-based architecture that introduces a hierarchical block between encoder and decoder for acquiring and merging features to extract multi-scale information. E1D3 U-Net [16] extends the 3D U-Net to use three decoder branches instead of one, where each decoder segments one of the hierarchical regions of interest. Peiris et al. [17] used dual reciprocal adversarial learning approaches. The authors followed the Virtual Adversarial Training approach by generating more adversarial examples by adding some noise to the original patient data. Zabihollahy et al. [18] used a separate U-Net per modality on a prostate dataset. One was tasked to segment the whole prostate gland, and the other to segment the central gland.

Another extension is U-Net++ [19], which replaced the simple skip connection with dense nested connections. The dense nested skip connection on U-Net++ added multiple convolutional layers between the source and target layer, and the output of every previous layer will be used in the input of the next layer. Thus, U-Net++ achieves higher prediction quality at the cost of much more computational effort. U-Net3+ [9] was proposed to address the high computational costs of U-Net++. Instead of dense nested skip connections, it uses full-scale skip connections, which means that every layer in the decoding part of the network also uses the output of the corresponding encoder layer and the downscaled output of encoder layers before the corresponding layer. Moreover, each decoder layer has skip connections to all previous decoder layers. ELU-Net, a recent U-Net iteration, uses a reversed approach by adding skip connections between decoder layers with the corresponding encoder layers and all subsequent encoder layers. However, they do not utilize the skip connection between the encoder layers. Hence, their network reaches higher prediction quality than the original U-Net with less computation. Table. 1 highlights aspects of improvement in the state-of-the-art for U-Nets. We observed only one method made use of low resolution inputs and very few concentrated on the cost of deployment. However, most state-of-the-art approaches rely optimizing the U-Net base architecture and its loss function. Moreover, we see that our J-Net is the only approach with a focus on minimizing the cost of deployment by making use of lower resolution inputs.

3 Our Framework

Fig. 1 presents an overview of our J-Net architecture and proposed loss function. We can use any U-Net as a basis for J-Net since all U-Nets share the standard structure of encoder and decoder.

First, the U-Net input image runs through the encoder. The purpose of the encoder is to extract high-level features, which is achieved by down-sampling the input through each level. The decoder's purpose is to retrieve spatial information by up-sampling its input through each decoder level.

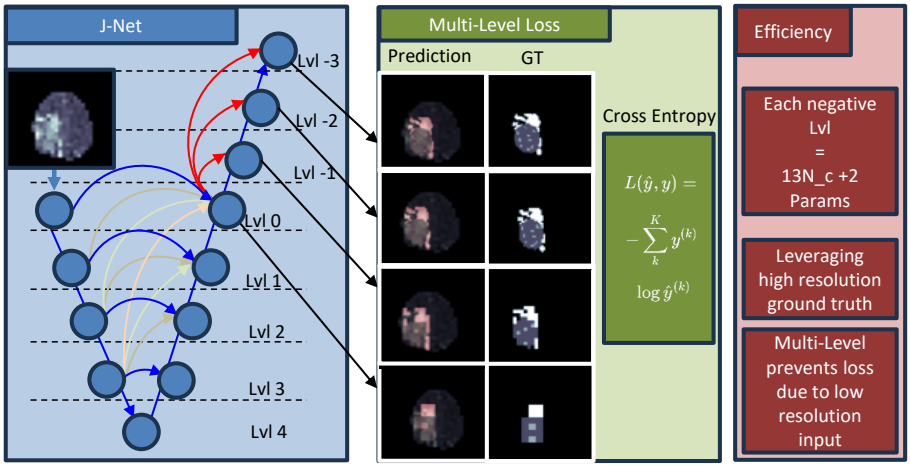

Fig. 1. Overview of J-Net. J-Net extends the conventional "U" shape of the network to produce higher-resolution outputs. In training, we compare the output at different sizes to the correspondingly reshaped ground truths.

However, as we already saw in Section 2, the original U-Net has undergone several iterations over the years. Moreover, several hardware and software optimizations were proposed to accelerate neural networks [20]. Nevertheless, to the best of our knowledge, they did not target their performance by reducing the input resolution in an efficient way with regard to the prediction quality.

In application, users often reduce the input resolution if the chosen architecture does not reach the desired performance requirements. Nonetheless, the reduction of the input resolution comes with a dramatic loss in prediction quality, although the available data quality would allow higher quality predictions. Moreover, in our experiments, we observed that the bulk of computational load stems from the input resolution rather than the output resolution. In other words, reducing (or increasing) the number of decoder layers does not affect the network's overall complexity by a considerable degree. Conversely, any change in the number of encoding layers has vast implications. Furthermore, higher-resolution ground truths are still available when reducing input resolution for performance reasons.

Thus, we propose our J-Net architecture. The J-Net architecture does not interfere with the baseline U-Net in any way but instead adds more up-scaling layers to the decoder part. This way, we can extend any U-Net-like architecture to a J-Net without much effort. The additional layers help the J-Net to predict a higher resolution than the input to compensate for the lost detail in the initial input compression. Our experiments showed that it is most efficient if we utilize simple transposed layers and up-scaling of the input only by a factor of two at the time. Therefore, we will need more up-scaling layers if we significantly reduce the input resolution. Furthermore, we added skip connections between the

additional up-scaling operations, as those would make it easier for the gradient to traverse the now deeper network and thus increase the prediction quality. We will compute the number of additional parameters here to prove that the proposed addition of up-scaling layers does not increase the computational load significantly. The up-scaling layer consists of a ConvTranspose2d and a Conv2d, which we define as C^T and C, respectively. The trainable weights $W_t(\cdot)$ of both layers can then be described as:

$$W_t(C) = W_t(C^T) = \\ (H_{kernel} \times W_{kernel} \times c_{in} + 1) \times c_{out},$$

where c_{in} is the number of channels if the layer input, c_{out} is the number of channels after the convolutional layer, H_{kernel} and W_{kernel} are the dimensions of the used convolutional kernel. In J-Net c_{in} and c_{out} are equal to the number of classes N_c, and for the ConvTranspose2d layer we use a 2×2 kernel, and for the Conv2d layer a 3×3 kernel. That means each up-sampling operation will add the following number of parameters:

$$\#Params = (4N_c + 1) \times N_c + (9N_c + 1) \times N_c = 13N_c^2 + 2N_c.$$

However, we perform the up-scaling layer multiple times until we reach the desired output resolution. For example, suppose we want to train with a compact resolution of 16×16, and we have a ground truth available of at least 256×256 resolution. In that case, we must perform four up-sampling operations in J-Net, doubling the output resolution each time. The number of classes we used for the Decathlon prostate dataset is $N_c = 1$. Altogether the additional parameter for a input resolution of 16×16 are calculated as follows:

$$\begin{aligned} \#Params = &[13 \times 1^2 + 2 \times 1^2]2 \\ &+ [4 \times 1^2 + 1]7 + [9 \times 1^2 + 1]2 \\ &+ [(4 \times 2 + 1) + (9 \times 2 + 1)]2 \\ &= 159, \end{aligned}$$

where the first term refers to normal two up-scaling layers, and the second term describes the stretching of the intermediate result for the skip connections, where we performed transposed convolution with a higher up-scaling factor each followed by one normal convolution, to reach the desired resolution. The higher up-scaling factor translates to performing the ConvTranspose2D operation multiple times, in our case 3 and *times*. The last term describes the two up-scaling layers, that used the additional skip connections, thus the input channels increase from one to two. Considering that the baseline models consist of several million trainable parameters, those 159 are negligible for additional computing efforts. However, regarding memory consumption, we must pay a higher price to store and load higher-resolution outputs and ground truths. We will provide an in-depth analysis of additional memory costs in Section 4.

Furthermore, to take more advantage of our architecture, we propose a new loss function:

$$L_{sum} = \sum_{i=0}^{m} L_i(F_i(Y), \hat{Y}_i)$$

Where L_i denotes our loss of choice, for example, Cross Entropy, Y is the input image, $\hat{Y}_i$ is the output of the i_{th} transposed layer, where $\hat{Y}_0$ denotes the prediction with the input resolution, m is the number of up-scaling layers we need, to reach the desired output resolution defined as M. We observed the best results when choosing M as high as possible and therefore strive to have at least one term compared to the most detailed ground truth to our availability. Incorporating all losses for each up-scaling stage does not add much complexity since the Cross-Entropy Loss is relatively simple, and extracting the output of each up-scaling layer comes at no cost.

We also experimented with other methods of reducing the resolution. Extending J-Net with only one up-sampling operation that increases the input resolution by a factor of 16 instead of four up-sampling operations with a factor of 2 did lead to predictions, which are similar to the quality of the predictions generated without extra up-sampling. We also tried adding skip connections from the encoder to the new up-scaling layers, as ELU-Net and U-Net3+ showed that those skip connections can significantly improve prediction quality while simultaneously being cost-efficient when compared to dense connections.

However, we observed that adding such skip connections did not contribute to better model predictions but added almost as many parameters as using higher input resolution from the start.

4 Results and Discussion

This section will review the experimental setup used, the results obtained, and the ablations and insights we draw from them.

For our performance evaluation, we also ran the networks on Nvidia's Jetson Nano, a small edge device for deep learning deployment. We used ELU-Net [10] as the backbone for J-Net.

We used the Jaccard Coefficient and the Dice Coefficient as the evaluation metrics for all experiments.

We evaluate our method on two medical datasets the Decathlon prostate dataset, and the Brain Tumor Segmentation (BraTS) Challenge 2020 dataset.

Decathlon: The Decathlon segmentation challenge is a multi-modal prostate dataset with the task of localizing and detecting the prostate peripheral and transition zones. The dataset includes 32 volumetric scans. Each scan is available in two modalities for each scan: the MRI-T2 and the Apparent Diffusion Coefficient (ADC). Furthermore, each scan also includes corresponding segmentation masks. We separated the scans into 2D slices, resulting in 1806 slices overall. We split the dataset into train and validation sets in a 2/3 to 1/3 ratio. In our experiments, we observed that adding the ADC modality to the model input had little to no effect on the model's prediction quality. Therefore, we only used

Table 2. Comparison of U-Net, ELU-Net, and J-Net on the Decathlon dataset at different input resolutions. The lower the input resolution, the higher is the prediction quality of J-Net compared to the baseline.

Method	Resolution	Dice	Jaccard
U-Net [8]		**98.8**	**97.7**
ELU-Net [10]	320x320	98.4	96.9
J-Net (Ours)		98.4	96.9
U-Net		**98.8**	**97.6**
ELU-Net	160x160	98.7	97.4
J-Net (Ours)		98.7	97.4
U-Net		98.4	96.9
ELU-Net	80x80	98.7	97.4
J-Net (Ours)		**98.7**	**97.4**
U-Net		96.0	92.4
ELU-Net	32x32	96.5	93.3
J-Net (Ours)		**98.0**	**96.1**
U-Net		93.2	87.3
ELU-Net	16x16	93.0	86.9
J-Net (Ours)		**96.2**	**92.6**

the MRI-T2 maps for training, which further reduces computational complexity. Moreover, we combined the prostate peripheral zone and the transition zone into a single 'positive' class as the transition zone is barely present in the dataset, and the models could not detect this class. Most 2D slices were available in 320×320 resolution. We stretched the image to this size using OpenCV's inter-linear interpolation scheme in cases where the provided resolution was lower than 320×320.

In Table 2, we compare the results on the Decathlon datasets. We compared a basic U-Net, ELU-Net, and J-Net. We list the Jaccard and Dice results on the validation set and perform the experiments with different input resolutions. For J-Net, we always used the 320×320 resolution ground truth, except for input resolution 32×32 and 16×16, for which we used a 258×258 ground truth resolution. At input resolution 320×320, the J-Net extension is not applicable anymore and thus J-Net is the same as ELU-Net.

On the highest input resolution U-Net achieves a 0.8% Jaccard improvement over ELU-Net. However, training the ELU-Net with halfed resolution seems to affect the overall prediction quality not at all. Nevertheless, if we further reduce the input resolution, the prediction quality of ELU-Net (and U-Net) diminishes, as expected. We observe that J-Net can maintain higher prediction quality on lower input resolutions than the other networks. Furthermore, we notice that the more we reduce input resolution, the more significant the difference between J-Net and ELU-Net. The difference between ELU-Net at 80×80 input resolution is non-existent, whereas, with 32×32 input resolution, the difference is almost 3% Jaccard score. At 16×16 input resolution, J-Net and ELU-Net are separated by 5.5% Jaccard score. We can explain this because ground truth information

diminishes greatly by reducing its resolution, while J-Net keeps using the high resolution and thus high information ground truths. Therefore, J-Net is not as much affected by the lower input resolution.

Table 3. Comparison of U-Net, ELU-Net, and J-Net on the BraTs dataset at different input resolutions. J-Nets results are on the same level as U-Net at the next higher input resolution.

Method	Input	ED Dice	ED Jacc.	NCR Dice	NCR Jacc.	ET Dice	ET Jacc.
U-Net [8]		86.9	82.1	88.3	85.6	89.2	85.6
ELU-Net [10]	256x256	**88.3**	**83.5**	**88.5**	**85.9**	**90.0**	**86.4**
J-Net (Ours)		**88.3**	**83.5**	**88.5**	**85.9**	**90.0**	**86.4**
U-Net		85.4	79.9	87.4	84.5	87.6	83.7
ELU-Net	128x128	85.4	80.0	86.9	84.3	87.7	83.8
J-Net (Ours)		**87.6**	**83.5**	**88.2**	**85.9**	**89.2**	**86.2**
U-Net		80.6	74.7	84.7	82.1	84.0	80.0
ELU-Net	64x64	81.4	75.4	85.3	82.6	84.8	80.8
J-Net (Ours)		**85.4**	**80.9**	**86.6**	**84.4**	**87.5**	**84.2**
U-Net		74.7	68.9	82.1	79.6	80.0	76.4
ELU-Net	32x32	75.2	69.3	82.7	80.3	80.7	77.2
J-Net (Ours)		**82.2**	**77.4**	**84.7**	**82.5**	**84.7**	**81.3**
U-Net		67.3	62.6	78.8	77.0	75.5	73.0
ELU-Net	16x16	67.5	62.8	79.5	77.7	76.8	74.0
J-Net (Ours)		**75.5**	**70.7**	**81.6**	**79.8**	**78.7**	**76.0**

BraTs: The BraTS dataset is a multi-modal brain tumor segmentation in Magnetic Resonance Images. The dataset includes 369 multi-modal scans with their corresponding expert segmentation masks. The available modalities are T1, T1c, T2, and T2 Fluid Attenuated Inversion Recovery (FLAIR). The dataset contains annotations for the GD-enhancing tumor (ET), the peritumoral edema (ED), and the necrotic and non-enhancing tumor core (NCR). We separated each scan into its set of slides, resulting in 50,764 slides, and cropped them to the size 256×256. We used a 85% to 15% train validation split for our experiments.

In Table 3, we compare the results of a basic U-Net, ELU-Net, and J-Net on the BraTs datasets. We used 256×256 resolution ground truth for all J-Net input resolutions. Again, at 256×256 input resolution the is no difference between ELU-Net and J-Net. As typical in this dataset, we evaluate the results for each class separately. This time, J-Net outperforms ELU-Net with the same input resolution and can achieve better results than ELU-Net with the next higher input resolution for the ED class and almost the same for the other two classes. For the input resolution of 128×128, J-Net improves on ELU-Net by 2.5% Jaccard on average and loses to ELU-Net with the input of 256×256 resolution 0.07% Jaccard on average. Similar to the evaluation on Decathlon, J-Net shows the most significant improvement with input resolution 16×16, achieving an average 4% better Jaccard prediction result. Note that when evaluating before

saturation, J-Net has an even higher lead than ELU-Net, showing that the up-scaling layers and high-resolution ground truths lead to fast learning.

Next, we will evaluate our model's hardware requirements compared to the baseline models' requirements, as this was the central motivation of the work.

Table 4. Complexity and memory consumption of the tested models with respect to the input resolution of one image with one channel.

Method	Resolution	Complexity (GMac)	Memory (mb)
U-Net [8]		71.16	1191.41
ELU-Net [10]	320x320	**29.98**	**868.36**
J-Net (Ours)		**29.98**	**868.36**
U-Net		17.79	297.85
ELU-Net	160x160	**7.5**	**217.09**
J-Net (Ours)		**7.5**	219.43
U-Net		4.45	74.46
ELU-Net	80x80	**1.87**	**54.27**
J-Net (Ours)		1.88	58.57
U-Net		0.71	11.91
ELU-Net	32x32	**0.30**	**8.68**
J-Net (Ours)		**0.30**	13.68
U-Net		0.18	2.98
ELU-Net	16x16	**0.07**	**2.17**
J-Net (Ours)		0.08	5.5

In Table 4, we compare the complexity and memory requirements, this will be followed by the time needed to evaluate one image with one channel on Jetson Nano for the different networks, as this was the setting we used with the Decathlon dataset. Starting with the complexity analysis in the number of the Giga Multiply and Accumulate (GMac) operations at 16×16 input resolution, J-Net has 0.08 GMacs compared to ELU-NETs 0.07 GMacs and U-Nets 0.18 GMACs. The additional 0.01 GMacs add relative 14% more complexity. At 32×32, both J-Net and ELU-Net have 0.3 GMacs, while U-Net has more than double with 0.71. The same goes for 80×80, with 1.88, 1.87 and 4.45 and for and 160×160, with 7.5, 7.5 and 17.79 for J-Net, ELU-Net and U-Net, respectively. For 320×320 the architecture of J-Net and ELU-Net are the same, as it is not useful to no up-sample beyond the original resolution but ELU-Net only uses 29.98 GMacs compared to U-Nets 71.16 GMacs.

Going over to the analysis of the memory requirements. We observe that as we increase the input resolution size, the relative difference in memory consumption between J-Net and the other networks decreases; At 16×16 input size J-Net uses 5.5 mb, ELU-Net 2.17 mb and U-Net 2.98 mb, which shows a huge difference between J-Net and the other U-Nets. This is mainly because, in J-Net, we need to store the full resolution prediction, while the other networks only deal with predictions and ground truths that are 20 times smaller. However, the gap shrinks with increasing resolution. At 32×32 input size J-Net uses 13.68 mb, ELU-Net

8.68 mb and U-Net 11.91 mb, and at 80×80 input size J-Net uses 58.57 mb, ELU-Net 54.27 mb and U-Net 74.46 mb, meaning that the difference between J-Net and U-Net is very small or even in favor of J-Net, at higher resolutions. For 160×160 inputs, J-Nets uses with 219.43 mb less than 1% more memory than ELU-Net (217.09 mb) and considerably less than U-Net (297.85 mb). When we compare J-Net's memory requirements to the other networks of the next higher input resolution class, we notice that even for the 16×16 inputs, J-Net uses 40% less memory than ELU-Net with 32×32 inputs resolution but reaches on decathlon a prediction quality closer to the results of ELU-Net with 32×32 inputs than with 16×16, which is in our eyes an acceptable trade-off.

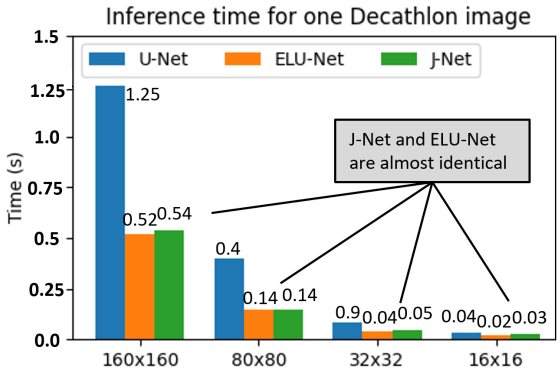

Fig. 2. Inference time in seconds for evaluating one decathlon image in Nvidia's Jetson Nano. ELU-Net and Jet-Net achieve almost the same speed-up compared to U-Net, while J-Net has much higher prediction quality.

Fig. 2 shows the number of seconds it takes to pass one Decathlon image with one channel on Nvidia's Jetson Nano. In deployment on Nvidia's Jetson Nano, it becomes clear why we should reduce the input resolution. With input resolution 80×80, we achieve a throughput of one image in 0.4 seconds for the U-Net, translating to 2.5 images every second. The more efficient ELU-Net and J-Net need around 0.15 seconds per image, which is almost seven images per second. A desirable throughput rate would be closer to 20 or 30 images per second; in this case, a live evaluation of a video stream would be possible. Rescaling the input to 16×16 would yield us 0.02 seconds per image for ELU-Net, translating to 50 images per second, which is better than our goal. The additional complexity of J-Net reduces the throughput rate to 0.25 seconds per image or 40 images per second, which is still above our optimal target of 30 images per second. However, the prediction quality of J-Net with 16×16 input resolution is comparable to ELU-Net with 32×32 input resolution. In this case, ELU-Net has a throughput rate of 0.04 seconds per image or 25 images per second,m40% slower compared to J-Net at 16×16.

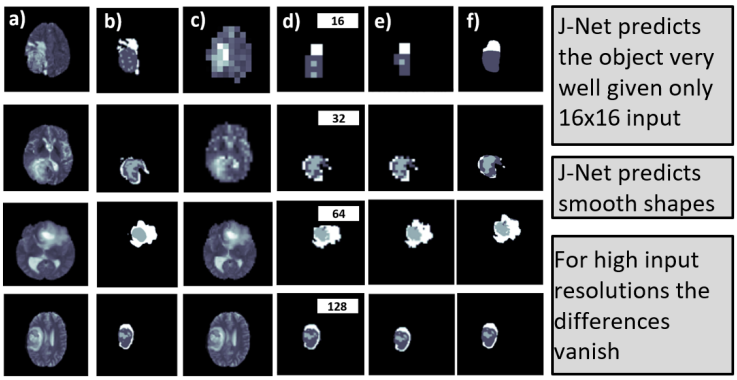

Fig. 3. Example results of J-Net and ELU-Net for comparison. The a) column shows the T2 brain MRIs in their original resolution. The b) column shows the ground truth in the original resolution and the c) and d) columns the T2 brain MRIs and ground truths in input resolution respectively, the resolution is specified in the top left corner in column d), and the e) column shows the prediction of ELU-Net), and the f) column shows the prediction of J-Net.

Fig. 3 and Fig. 4 give a qualitative overview of J-Nets results compared to ELU-Net. Each row of the figures depicts an example image at a different input resolution from BraTs and Decathlon, respectively. For both figures, column 'a)' shows the MRI-T2 modality, and column 'b)' indicates the corresponding ground truth in the original resolution. Columns 'c)' and 'd)' show the MRI-T2 modality and ground truth in reduced resolution as indicated in each ground truth. Column 'e)' and column 'f)' depicts the prediction of ELU-Net and J-Net given the low-resolution input image in column 'a)'. We chose the T2 modality for Fig. 3 as untrained eyes can spot cancerous regions relatively easily. Note that ELU-Net was trained only using ground truths of the same resolution as the input, while J-Net was trained using original resolution ground truths. Fig. 3 elucidates how the difference between ELU-Net and J-Net is most noticeable at the lowest input resolution. J-Nets low resolution predictions are very detailed, which shows that the network can extract significant amounts of useful information using highly compressed images. Similarly, in Fig. 4, even with the 16×16 input image, J-Net predicts a shape quite like the high-resolution ground truth. We can summarize that networks can extract much more information from their input than their output suggests and are limited by their output resolution. Moreover, we observe that J-Net allows us to overcome this limitation.

5 Conclusion

In this paper, we have proposed our J-Net, which provides a novel way to combine low resolution input images with high-resolution ground-truths efficiently. The J-Net adds multiple up-scaling layers with skip connections at the end of

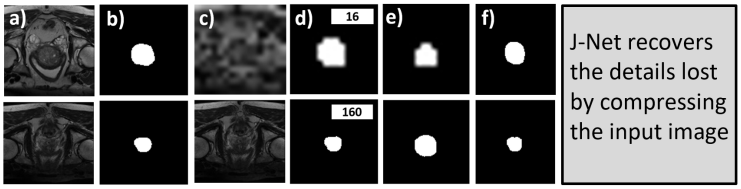

Fig. 4. Example results of J-Net and ELU-Net for comparison. The a) column shows the T2 brain MRIs in their original resolution. The b) column shows the ground truth in the original resolution and the c) and d) columns the T2 brain MRIs and ground truths in input resolution respectively, the resolution is specified in the top left corner in column d), and the e) column shows the prediction of ELU-Net), and the f) column shows the prediction of J-Net.

a U-Net-like architecture to leverage the power of high-resolution ground truths with minimal compromises for the lightweight nature of low-resolution inputs. The J-Net up-scaling layers can be added to any conventional U-Net-like network without making any changes to the base network. We showed that the predictions generated by J-Net significantly improve the performance on the Decathlon and BraTs datasets. This shows that J-Net is a valuable alternative to maintain prediction quality if resolution reduction is necessary because of hardware limitations. J-Net is open-source to ensure reproducible research and accessibility. The source code is online at `https://BlindedLinkForReview`.

References

1. S. L. Oh, Y. Hagiwara, U. Raghavendra, R. Yuvaraj, N. Arunkumar, M. Murugappan, and U. R. Acharya, "A deep learning approach for parkinson's disease diagnosis from eeg signals," *Neural Computing and Applications*, vol. 32, pp. 10927–10933, 2020.
2. E. M. Masutani, N. Bahrami, and A. Hsiao, "Deep learning single-frame and multiframe super-resolution for cardiac mri," *Radiology*, vol. 295, no. 3, pp. 552–561, 2020.
3. S. Zhang, G. Liang, S. Pan, and L. Zheng, "A fast medical image super resolution method based on deep learning network," *IEEE Access*, vol. 7, pp. 12319–12327, 2018.
4. K. Suzuki, "Pixel-based machine learning in medical imaging," *Journal of Biomedical Imaging*, vol. 2012, pp. 1–1, 2012.
5. C.-W. Wang, C.-T. Huang, M.-C. Hsieh, C.-H. Li, S.-W. Chang, W.-C. Li, R. Vandaele, R. Marée, S. Jodogne, P. Geurts, *et al.*, "Evaluation and comparison of anatomical landmark detection methods for cephalometric x-ray images: a grand challenge," *IEEE transactions on medical imaging*, vol. 34, no. 9, pp. 1890–1900, 2015.
6. B. D. De Vos, J. M. Wolterink, P. A. De Jong, M. A. Viergever, and I. Išgum, "2d image classification for 3d anatomy localization: employing deep convolutional neural networks," in *Medical imaging 2016: Image processing*, vol. 9784, pp. 517–523, SPIE, 2016.

7. M. Marsousi, K. N. Plataniotis, and S. Stergiopoulos, "Shape-based kidney detection and segmentation in three-dimensional abdominal ultrasound images," in *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 2890–2894, IEEE, 2014.
8. O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pp. 234–241, Springer, 2015.
9. H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, "Unet 3+: A full-scale connected unet for medical image segmentation," in *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 1055–1059, IEEE, 2020.
10. Y. Deng, Y. Hou, J. Yan, and D. Zeng, "Elu-net: an efficient and lightweight u-net for medical image segmentation," *IEEE Access*, vol. 10, pp. 35932–35941, 2022.
11. M. Antonelli, A. Reinke, S. Bakas, K. Farahani, A. Kopp-Schneider, B. A. Landman, G. Litjens, B. Menze, O. Ronneberger, R. M. Summers, *et al.*, "The medical segmentation decathlon," *Nature communications*, vol. 13, no. 1, p. 4128, 2022.
12. B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest, *et al.*, "The multimodal brain tumor image segmentation benchmark (brats)," *IEEE transactions on medical imaging*, vol. 34, no. 10, pp. 1993–2024, 2014.
13. C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, 2017.
14. L. Tan, W. Ma, J. Xia, and S. Sarker, "Multimodal magnetic resonance image brain tumor segmentation based on acu-net network," *IEEE Access*, vol. 9, pp. 14608–14618, 2021.
15. P. Ahmad, H. Jin, R. Alroobaea, S. Qamar, R. Zheng, F. Alnajjar, and F. Aboudi, "Mh unet: A multi-scale hierarchical based architecture for medical image segmentation," *IEEE Access*, vol. 9, pp. 148384–148408, 2021.
16. S. T. Bukhari and H. Mohy-ud Din, "E1d3 u-net for brain tumor segmentation: Submission to the rsna-asnr-miccai brats 2021 challenge," in *International MICCAI Brainlesion Workshop*, pp. 276–288, Springer, 2021.
17. H. Peiris, Z. Chen, G. Egan, and M. Harandi, "Reciprocal adversarial learning for brain tumor segmentation: a solution to brats challenge 2021 segmentation task," in *International MICCAI Brainlesion Workshop*, pp. 171–181, Springer, 2021.
18. F. Zabihollahy, N. Schieda, S. Krishna Jeyaraj, and E. Ukwatta, "Automated segmentation of prostate zonal anatomy on t2-weighted (t2w) and apparent diffusion coefficient (adc) map mr images using u-nets," *Medical physics*, vol. 46, no. 7, pp. 3078–3090, 2019.
19. Z. Zhou, M. M. Rahman Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*, pp. 3–11, Springer, 2018.
20. M. Capra, B. Bussolino, A. Marchisio, G. Masera, M. Martina, and M. Shafique, "Hardware and software optimizations for accelerating deep neural networks: Survey of current trends, challenges, and the road ahead," *IEEE Access*, vol. 8, pp. 225134–225180, 2020.