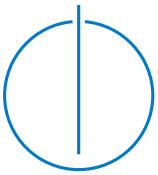
DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Biomedical Computing

'Project & Excite' Modules for Segmentation of Volumetric Medical Scans

Anne-Marie Rickmann



DEPARTMENT OF INFORMATICS

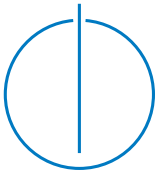
TECHNICAL UNIVERSITY OF MUNICH

Master's Thesis in Biomedical Computing

'Project & Excite' Modules for Segmentation of Volumetric Medical Scans

'Project & Excite' Module für die Segmentierung von Volumetrischen Medizinischen Scans

Author:	Anne-Marie Rickmann
Supervisor:	Prof. Dr. Nassir Navab
Advisor:	MSc. Ignacio Sarasua
Advisor:	MSc. Abhijit Guha Roy
Submission Date:	April 15th, 2019



I confirm that this master's thesis is my own work and I have documented all sources and material used.

April 15th, 2019 Anne-Marie Rickmann

Acknowledgments

I would like to thank everyone who supported me during my studies and the duration of this project. My supervisors Abhijit Guha Roy and Ignacio Sarasua, for their ideas and encouragement and the guidance to focus on the important aspects of the project. Shayan Siddiqui and Abhijit Guha Roy for providing baseline implementations, which saved me a lot of time. Prof. Christian Wachinger for taking the time to discuss details of the project and provide valuable feedback. Everyone at AI-Med, especially for helping me write my first research paper. My parents for the emotional and financial support during the 5 years of my studies. All my friends, who I met through the BMC program, for all the fun we have had in the last years. Especially Agnieszka Tomczak, for all the discussions and honest feedback, for motivating me and for reminding me to set priorities.

Abstract

Fully Convolutional Neural Networks (F-CNNs) achieve state-of-the-art performance for image segmentation in medical imaging and computer vision. Recently, squeeze and excitation (SE) modules and variations thereof have been introduced to recalibrate feature maps channel- and spatial-wise, which can boost performance while only minimally increasing model complexity. So far, the development of SE has focused on 2D images. In this thesis, I propose 'Project & Excite' (PE) modules that base upon the ideas of SE and extend them to operating on 3D volumetric images. 'Project & Excite' does not perform global average pooling, but squeezes feature maps along different slices of a tensor separately to retain more spatial information that is subsequently used in the excitation step. I demonstrate that PE modules can be easily integrated in 3D U-Net, boosting performance by $4-5\%$ Dice points, while only increasing the model complexity by 2%. I evaluate the proposed module on two challenging tasks, whole-brain segmentation of MRI scans and whole-body segmentation of CT scans.

Contents

Acknowledgements **iv**

Abstract **v**

List of Figures **viii**

List of Tables **ix**

I. Introduction and Background Theory 1

1. Introduction 2

2. Background Theory 5

2.1. Deep Learning for Medical Image Segmentation 5

2.1.1. U-net 5

2.1.2. Dice Loss 5

2.1.3. Whole Brain Segmentation 6

2.1.4. 3D Segmentation 7

II. Methods and Experiments 9

3. Methodology 10

3.1. Squeeze and Excitation Modules 10

3.1.1. 3D channel SE 10

3.1.2. 3D spatial SE 12

3.1.3. Project & Excite . . . 12
3.2. Baseline Network Architecture . . . 15
3.2.1. Loss Function . . . 17
3.3. Implementation Details . . . 17

4. Experiments **19**
4.1. Datasets . . . 19
4.1.1. Whole-Brain Segmentation . . . 19
4.1.2. Whole-Body Segmentation . . . 20
4.2. Training Setup: . . . 21
4.3. Training Parameters: . . . 22

III.Results and Conclusion **23**

5. Results **24**
5.1. Position of PE Blocks . . . 24
5.2. Model Complexity: . . . 25
5.3. Segmentation Results: . . . 26

6. Conclusion **31**

Bibliography **32**

List of Figures

1.1. Projections in 'PE' block 3

2.1. U-net architecture 6

3.1. Illustration of the 3D cSE block. 11
3.2. Illustration of the 3D sSE block. 13
3.3. Illustration of the proposed 'Project& Excite' block. 14
3.4. 3D U-net used as baseline architecture. 16

4.1. Class frequencies of the left hemisphere of the brain 20
4.2. Class frequencies of the 14 labels of visceral dataset 21

5.1. Input scans, ground truth segmentation and results for 3D U-Net, 3D cSE and the proposed PE model 29
5.2. Boxplots comparing 3D U-Net, cSE, sSE and the proposed PE model for MALC and Visceral datasets. 30

List of Tables

5.1. Mean Dice score on MALC dataset due to placement of 'PE' blocks within 3D U-Net architecture. 24
5.2. Mean Dice vs model complexity measured in number of trainable parameters 25
5.3. Comparison of 3D U-Net with 3D cSE, 3D sSE and the proposed PE block. 26

Part I.

Introduction and Background Theory

1. Introduction

Fully convolutional neural networks (F-CNNs) have been widely adopted for semantic image segmentation in computer vision [8, 12] and medical imaging [13, 3, 11]. As computer vision tasks mainly deal with 2D natural images, most of the architectural innovations have focused towards 2D CNNs. For medical image segmentation, 2D F-CNNs are used to segment 3D medical scans slice-wise. In such an approach the contextual information from adjacent slices remains unexplored, which might lead to imperfect segmentations, especially if the target class is small. Hence, the natural choice of segmenting 3D scans would be to use 3D F-CNN architectures. However, there exist some practical challenges in using 3D F-CNNs. 3D F-CNNs require large amount of GPU RAM space for training and the number of learnable weight parameters are much higher than for its 2D counter-part, which can make the models prone to over-fitting with limited training data. Although the first issue can be effectively addressed using recent GPU clusters, the second issue still remains. This problem is prominent in medical applications, where training data is commonly very limited. Most datasets often contain only about 15-20 annotated training scans. To overcome the problem of over-fitting, 3D F-CNNs are carefully engineered for a task to minimize the model complexity by reducing the number of convolutional layers or by decreasing the number of channels per convolutional layer. Although this might aid training models with limited data, the exploratory capacity of the 3D F-CNN gets limited. In such a scenario, it is necessary to ensure that the learnable parameters within the F-CNN are maximally utilized to solve the task at hand. Recently, a computational module termed 'Squeeze and Excite' (SE) block [5] has been introduced to recalibrate CNN feature maps, which boosts the performance while increasing model complexity marginally. This is performed by modeling the interdependencies between the channels of feature maps, and learning to provide attention

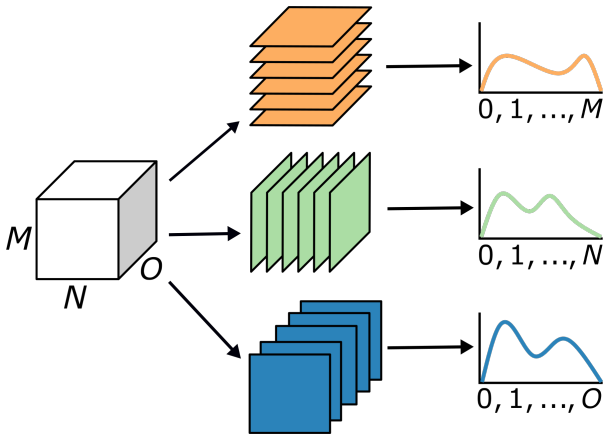


Figure 1.1.: Projections in 'PE' block

on specific channels depending on the task. This idea was also extended to medical image segmentation [16], where it was demonstrated that such light-weight blocks can be a better architectural choice than extra convolutional layers. Although, SE blocks were customarily designed for 2D architectures, they have recently been extended to 3D F-CNNS to aid volumetric segmentation [18].

In this thesis, I discuss state-of-the-art F-CNN architectures for medical image segmentation, with a focus on whole-brain segmentation. I will then dive deeper into the topic of squeeze and excite based attention modules and propose the 'Project & Excite' (PE) module, a new computational block custom-made to recalibrate 3D F-CNNs. Zhu et al. [18] directly extended the concept of SE to 3D by averaging the 4D tensor over all spatial dimensions to generate a channel descriptor for recalibration. I hypothesize, that removing all spatial information leads to a loss of relevant information, particularly for segmentation, where we need to exactly localize anatomical structures. In contrast, PE blocks aim at preserving the spatial information without any excess model complexity or FLOP operations, which is relevant for fine-grained volumetric segmentation. I draw my inspiration from traditional tensor slicing techniques, by averaging along the three principle axes of the tensor as indicated in Fig. 1.1. I term this operation the 'Projection' operation. By this, we get three projection-vectors indicating the relevance of the slices along the three axes. A spatial location is important if all the corresponding slices associated with it provide higher estimates. So, instead of learning the dependencies of the

scalar values across the channels as in [18], the dependencies of these projection-vectors across the channels are learned for excitation. Also, PE blocks provide a global receptive field to the network at every stage.

2. Background Theory

2.1. Deep Learning for Medical Image Segmentation

2.1.1. U-net

Although Convolutional Neural Networks (CNNs) have been very successful in Computer Vision tasks for several years, they were mostly built for classification tasks, assigning one or multiple labels to the whole image. For the task of semantic segmentation, FCNNs became popular after publication of [8]. In medical image segmentation the most popular network architecture is U-net [13] on which most recent architectures are based on. The U-net architecture consists of an contracting encoder path, that performs several steps of convolutions followed by downsampling, using max-pooling, of the image to achieve a global receptive field. The encoder path is followed by an expanding decoder path, which upsamples the feature maps by using transpose convolutions. Each encoder block forwards the last feature map to the corresponding decoder block by a so-called skip-connection. This way the important fine grained spatial information doesn't get lost. The final segmentation is achieved by a kernel size 1×1 convolution with the channel size being the number of classes, followed by a softmax layer. In state-of-the-art segmentation networks, this original U-net arcitecture is commonly extended by batch normalization. Figure 2.1 shows the original illustration of the U-net architecture, taken from the paper from Ronneberger et al. [13].

2.1.2. Dice Loss

In medical image segmentation, the most commonly used loss function is the Dice Loss, which was first introduced in [11]. It is based on the Dice Similarity Coefficient, which is

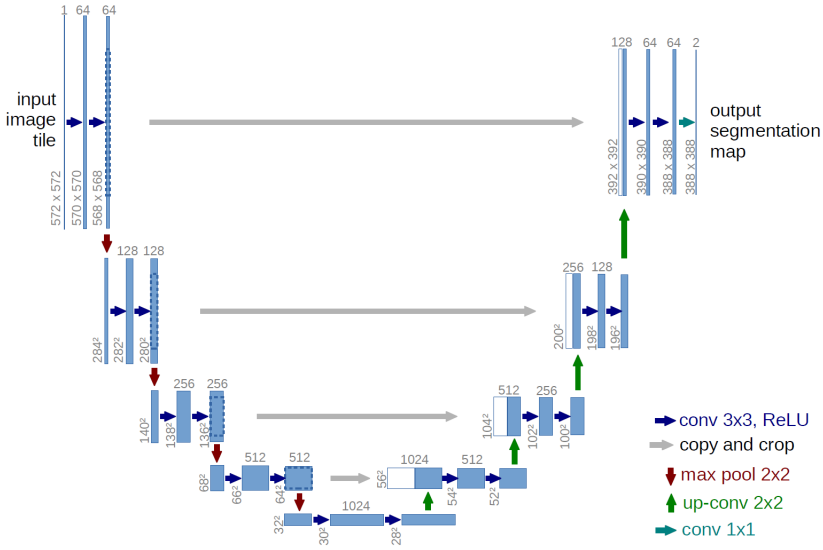


Figure 2.1.: U-net architecture[13]

a common evaluation metric for medical image segmentation. The Dice score lies in the range between 0 and 1, with a score of 1 meaning perfectly overlapping segmentations. The Dice Coefficient is defined as:

$$Dice(\mathbf{X}, \mathbf{Y}) = \frac{2\sum_{i=1}^{N} x_i y_i}{\sum_{i=1}^{N} x_i^2 + \sum_{i=1}^{N} y_i^2} \tag{2.1}$$

where $\mathbf{X}, \mathbf{Y}$ are 2 binary segmentations with N voxels and x_i, y_i the i_th voxels of $\mathbf{X}$ and $\mathbf{Y}$. Since this function is differentiable [11], the Dice Loss function can be defined as $1 - Dice(\mathbf{X}, \mathbf{Y})$, where $\mathbf{X}$ is the predicted segmentation and $\mathbf{Y}$ the ground truth segmentation. For multi-class segmentation tasks, the Dice Coefficient is calculated separately for each class and then averaged over all classes to obtain a global similarity descriptor.

2.1.3. Whole Brain Segmentation

Whole brain segmentation is important for studying morphological changes in the brain for diagnosis and monitoring disease progression in diseases like Alzheimers, schizophre-

nia or Parkinson. Manual segmentation by radiologists is time consuming, so there is a need for automatic segmentation algorithms. F-CNNs have shown promising results in brain segmentation [4]. One of the challenges of whole brain segmentation is, like in all medical segmentation tasks, the lack of large annotated datasets for training. Secondly, the number of classes to be segmented is high, unlike in other tasks, which are often binary. The class frequencies are also highly unbalanced, meaning there are some very small structures that make up only a very small percentage of the volume. To tackle this class imbalance, the loss function needs to be weighted accordingly, giving higher weights to smaller classes. There exist various weighting schemes, in this thesis I used median frequency balancing, which will be explained in the Methods section 3.2.1.

2.1.4. 3D Segmentation

Although most medical images are inherently volumetric, 2D network architectures are still mostly used. 2D architectures have some obvious advantages over 3D networks, like lower computational complexity and less GPU memory requirements. The advantage of 3D networks is, that they make use of all the anatomical information, while 2D networks, don't exploit the information in the directions orthogonal to the image plane [10]. Some methods eliminate this problem by training networks on slides from different views (sagittal, coronal and axial) and fuse the predictions to obtain a 3D segmentation [14]. Despite the challenges of 3D networks, there exist some common architectures, which i will describe in the following section.

3D Network Architectures

3D-Unet is a 3D extension of the previously mentioned U-net, using 3D convolutions instead of 2D convolutions. The original version only uses 3 encoding and 3 decoding blocks, also they added batchnormalization after the convolutional layers. The networks building blocks use convolutional layers with kernel size $3 \times 3 \times 3$, max-pooling with size $2 \times 2 \times 2$ and stride 2. A similar, popular 3D architecture is V-net [11]. V-net uses a similar encoder-decoder based structure, but has some differences. i) they use strided convolutions instead of max-pooling for downsampling, ii) they introduced the previously

mentioned Dice loss, iii) they use additional residual connections in each encoder/decoder block, iv) they use PRelu nonlinearity instead of Relu, v) convolutional layers have kernel size $5\times5\times5$. 3D U-net and V-net are both very general architectures that can be adapted for individual tasks, by changing the number of encoders/decoders, convolutional kernel-size or number of channels in convolutional layers. One example of a modified 3D U-net architecture is AnatomyNet [18], which is used to segment full volume CT scans of head and neck anatomies. The architectural changes include having only one downsampling layer in the encoder path and consequently only one upsampling layer in the decoder path. They substantiate this modification by showing that multiple downsampling layers can hurt the segmentation accuracy of small structures. Further, they reduce the number of channels per convolutional layer, to keep the number of learnable parameters low. In addition, AnatomyNet is the first network to introduce Squeeze and Excite modules into a 3D F-CNN architecture. In the following chapter, I will explain the concept of original Squeeze and Excite blocks, with the recent modifications and 3D extension, used in AnatomyNet, and introduce the novel 'Project & Excite' module.

Part II.

Methods and Experiments

3. Methodology

3.1. Squeeze and Excitation Modules

'Squeeze & Excite' (SE) blocks were first introduced in [5]. SE blocks recalibrate the channels of a feature map, by learning channel weights, that model the interdependencies between the channels. The idea of SE has been further developed in [16], where it was used for the task of medical image segmentation. Roy et al. termed the original SE block channel SE block and proposed another SE block, termed spatial SE, which recalibrates spatially instead of channel-wise. In the following sections I will follow this suggestion of the terms channel SE and spatial SE. To be consistent with the literature I will follow the notations used in [5, 16].
SE blocks $\mathbf{F}_{se}(\cdot)$ take a feature map $\mathbf{U}$, which usually is the output of a convolutional layer, as input and recalibrate it to $\hat{\mathbf{U}} = \mathbf{F}_{se}(\mathbf{U})$. Let $\mathbf{U}, \hat{\mathbf{U}} \in \mathbb{R}^{H \times W \times D \times C}$, with height H, width W, depth D, and number of channels C. This section details the extension of channel and spatial SE to 3D F-CNNs and finally introduces the novel 'Project & Excite' blocks.

3.1.1. 3D channel SE

This 3D SE block is a direct extension of the original SE block, proposed in [5], to a 3D version, which was first introduced in [18]. The transformation $\mathbf{F}_{se}(\cdot)$ is divided into the squeeze operation $\mathbf{F}_{sq}(\cdot)$ and excite operation $\mathbf{F}_{ex}(\cdot)$. The squeeze operation $\mathbf{F}_{sq}(\cdot)$ performs a global average pooling operation that squeezes the spatial content of the input $\mathbf{U}$ into a scalar value per channel $\mathbf{z} \in \mathbb{R}^C$. The excitation operation $\mathbf{F}_{ex}(\cdot)$ takes in $\mathbf{z}$ and adaptively learns the inter-channel dependencies by using two fully-connected

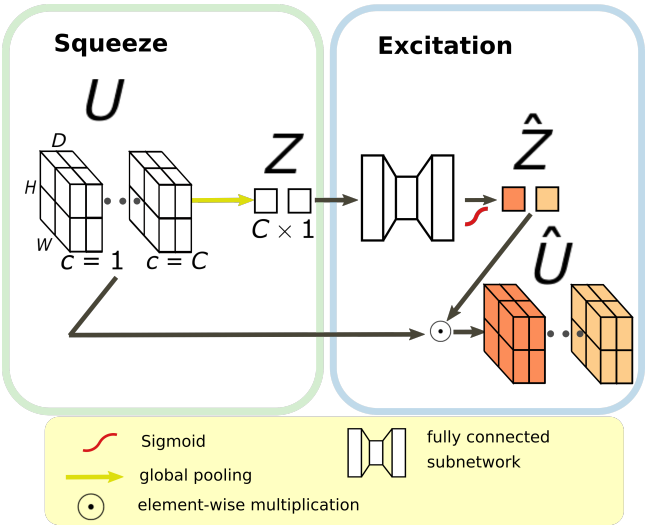


Figure 3.1.: Illustration of the 3D cSE block.

layers. The operations are defined as:

$$z_c = \mathbf{F}_{sq}(\mathbf{u}_c) = \frac{1}{H}\frac{1}{W}\frac{1}{D}\sum_{i=1}^{H}\sum_{j=1}^{W}\sum_{k=1}^{D}\mathbf{u}_c(i,j,k), \tag{3.1}$$

$$\hat{\mathbf{z}} = \mathbf{F}_{ex}(\mathbf{z}, \mathbf{W}) = \sigma(\mathbf{W}_2\delta(\mathbf{W}_1\mathbf{z})), \tag{3.2}$$

with δ denoting the ReLU nonlinearity, σ the sigmoid layer, $\mathbf{W}_1 \in \mathbb{R}^{\frac{C}{r}\times C}$ and $\mathbf{W}_2 \in \mathbb{R}^{C\times\frac{C}{r}}$ the weights of the fully-connected layers and r is the channel reduction factor similar to [5]. The output of the 3D cSE module is defined by a channel-wise multiplication of $\mathbf{U}$ with $\hat{\mathbf{z}}$. The cth channel of $\hat{\mathbf{U}}$ is defined as: $\hat{\mathbf{u}}_c = \mathbf{F}_{ex}(\mathbf{F}_{sq}(\mathbf{u}_c))\mathbf{u}_c = \hat{z}_c\mathbf{u}_c$. Figure 3.1 illustrates the 3D cSE module.

It has been shown in [5, 16, 18], that adding this block into existing F-CNN architectures, can improve the performance of the network comparable to the performance of much deeper networks, while keeping the model complexity low. Further, it was observed in [5], that cSE blocks in shallower layers are exciting low-level features which are class agnostic, while in deeper layers, class specific features are emphasized.

3.1.2. 3D spatial SE

The original 2D spatial SE (sSE) module was introduced in [16]. The idea behind it is, that channel SE focuses on learning channel relationships by providing a global receptive field using average pooling, but for segmentation tasks, the fine grained local information is important, so sSE modules learn a spatial attention map, without changing the receptive field, to lead the network to focus on specific regions. The spatial SE module is similar to the channel SE module, but it squeezes along the channel dimension and excites spatially. It was shown [16], that this is very efficient for segmentation tasks and outperforms the channel SE module. A 3D extension is straight-forward, by replacing all operations by their 3D counter-parts. The channel squeeze operation is performed using a convolution with kernel size $1 \times 1 \times 1$ and output channel size 1, followed by a Sigmoid layer. This way every element in the squeezed feature map encodes a linear combination of the same location over all channels. For an input feature map $\mathbf{U}$, the squeeze operation $F_{sq}(\cdot)$ is defined as:

$$F_{sq}(\mathbf{U}) = \sigma(\mathbf{W} \star \mathbf{U}) \tag{3.3}$$

with σ being the sigmoid function, $\star$ the convolution operation and $\mathbf{W}$ the convolutional weights. Excitation follows by multiplying all channels of the input feature map $\mathbf{U}$ with the squeezed feature map $F_{sq}(\mathbf{U})$. An illustration of the 3D sSE block can be found in Figure 3.2.

3.1.3. Project & Excite

The 3D cSE module squeezes spatial information of a volumetric feature map into one scalar value per channel. Especially in the first and last layers of a typical architecture, these feature maps have a high spatial extent. The hypothesis is, that a volumetric input of large size holds relevant spatial information which might not be properly captured by a global pooling operation. Hence, I introduce the 'Project & Excite' module that retains more of the valuable spatial information within the proposed *projection operation instead of spatial squeeze operation*. This follows the excite operation, which learns interdependencies between the projections across the different channels. Thus, it combines

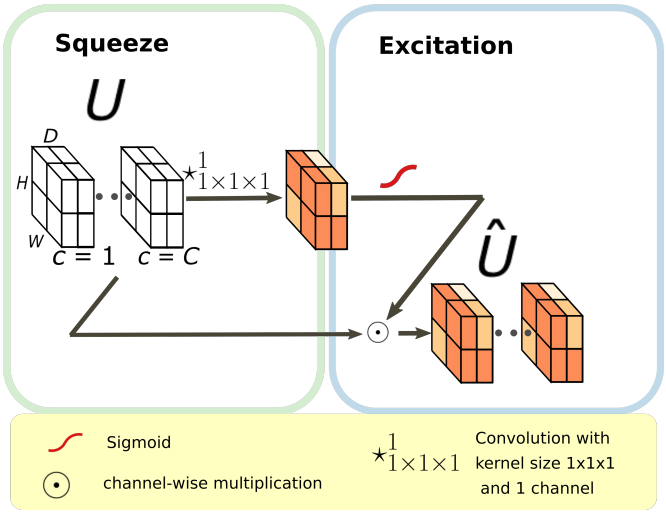


Figure 3.2.: Illustration of the 3D sSE block.

spatial and channel context for recalibration. The architectural details of the 'PE' block is illustrated in Fig. 3.3. The projection operation $\mathbf{F}_{pr}(\cdot)$ is separated into three projection operations ($\mathbf{F}_{pr_H}(\cdot)$, $\mathbf{F}_{pr_W}(\cdot)$, $\mathbf{F}_{pr_D}(\cdot)$) along the spatial dimensions with outputs $\mathbf{z}_{h_c} \in \mathbb{R}^{C\times H}$, $\mathbf{z}_{w_c} \in \mathbb{R}^{C\times W}$ and $\mathbf{z}_{d_c} \in \mathbb{R}^{C\times D}$. The projection operations are done by average pooling defined as:

$$\mathbf{z}_{h_c}(i) = \mathbf{F}_{pr_H}(\mathbf{u}_c) = \frac{1}{W}\frac{1}{D}\sum_{j=1}^{W}\sum_{k=1}^{D}\mathbf{u}_c(i,j,k), \quad i \in \{1,\ldots,H\} \tag{3.4}$$

$$\mathbf{z}_{w_c}(j) = \mathbf{F}_{pr_W}(\mathbf{u}_c) = \frac{1}{H}\frac{1}{D}\sum_{i=1}^{H}\sum_{k=1}^{D}\mathbf{u}_c(i,j,k), \quad j \in \{1,\ldots,W\} \tag{3.5}$$

$$\mathbf{z}_{d_c}(k) = \mathbf{F}_{pr_D}(\mathbf{u}_c) = \frac{1}{H}\frac{1}{W}\sum_{i=1}^{H}\sum_{j=1}^{W}\mathbf{u}_c(i,j,k), \quad k \in \{1,\ldots,D\}. \tag{3.6}$$

The outputs $\mathbf{z}_c$ are tiled to the shape $H \times W \times D \times C$ and added to obtain $\mathbf{Z}$, which is then fed to the excitation operation $\mathbf{F}_{ex}(\cdot)$, which is defined by two convolutional layers followed by a ReLU and sigmoid activation respectively. The convolutional layers have kernel size $1 \times 1 \times 1$, to aid modelling of channel dependencies. The first layer reduces

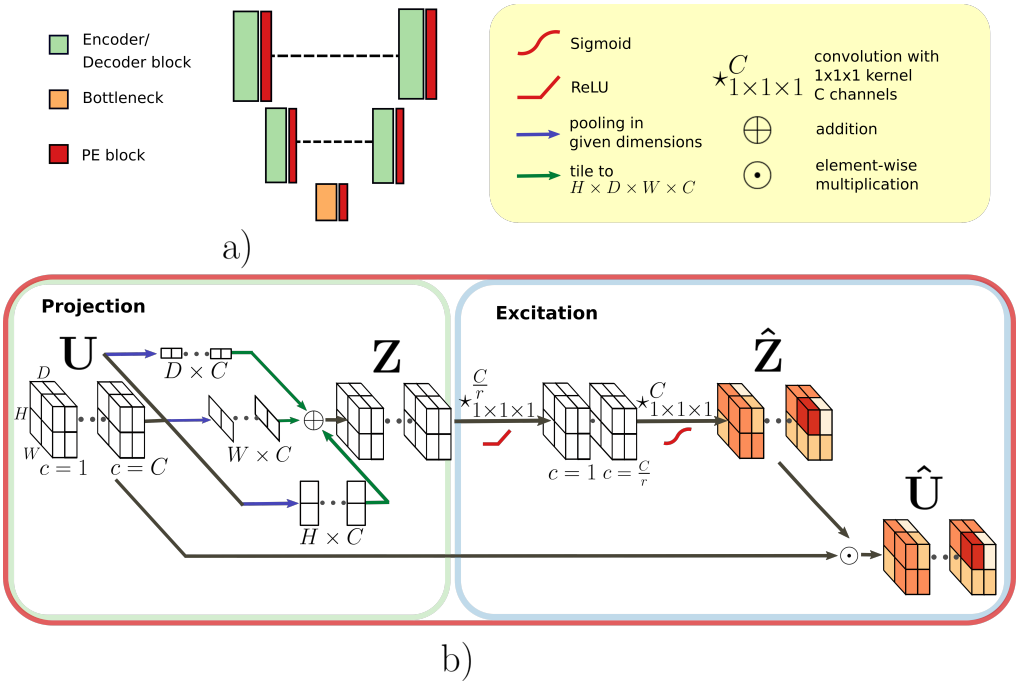


Figure 3.3.: a): typical encoder/decoder based F-CNN architecture with PE blocks placed after each block. b): Illustration of the proposed 'Project& Excite' block. Projection operation, with the 3 different pooling operations and Excitation operation with 2 convolutional layers and recalibration of the feature map.

the number of channels by r, and the second layer brings the channel dimension back to the original size. The excite operation is defined as:

$$\hat{\mathbf{U}} = \hat{\mathbf{Z}} \odot \mathbf{U} = \mathbf{F}_{ex}(\mathbf{Z}) \odot \mathbf{U} = \sigma(\mathbf{V}_2 \star \delta(\mathbf{V}_1 \star \mathbf{Z})) \odot \mathbf{U}, \tag{3.7}$$

where $\star$ describes the convolution operation, $\odot$ indicates point-wise multiplication, $\mathbf{V}_1 \in \mathbb{R}^{1\times1\times1\times\frac{C}{r}}$ and $\mathbf{V}_2 \in \mathbb{R}^{1\times1\times1\times C}$ the convolution weights, σ the sigmoid and δ the ReLU activation function. The final output of the PE block $\hat{\mathbf{U}}$ is obtained by an element-wise multiplication of the feature map $\mathbf{U}$ and $\hat{\mathbf{Z}}$.

3.2. Baseline Network Architecture

In the next chapter 4, I report several experiments conducted to compare the performance of the different squeeze & excite blocks. Here I define the baseline network architecture used in the experiments.

The architecture is based on 3D U-net [3], which was described in the Introduction chapter 1. The 3 encoder blocks consist of 2 convolutional layers with kernel size of $3 \times 3 \times 3$, followed by an Instance normalization layer [17]. The original 3D U-net uses batch-normalization, but since batch size is set to 1 in all experiments, batch normalization makes training unstable. Since Instance normalization normalizes each feature map of the batch separately it is agnostic to batch size. Let $\mathbf{X} \in \mathbb{R}^{N\times C\times D\times H\times W}$ be the output of a convolutional layer in the network. With N, C, D, H, W being batch size, channel size, depth, height and width of the feature map. x_{nijkl} be its $nijkl_{th}$ element. Following the notations from [17], instance normalization is defined as:

$$\hat{x}_{nijkl} = \frac{x_{nijkl} - \mu_{ni}}{\sqrt{\sigma_{ni}^2 + \epsilon}} \tag{3.8}$$

$$\mu_{ni} = \frac{1}{DHW} \sum_{j=1}^{D} \sum_{k=1}^{H} \sum_{l=1}^{W} x_{nijkl} \tag{3.9}$$

$$\sigma_{ni}^2 = \frac{1}{DHW} \sum_{j=1}^{D} \sum_{k=1}^{H} \sum_{l=1}^{W} (x_{nijkl} - \mu_{ni})^2 \tag{3.10}$$

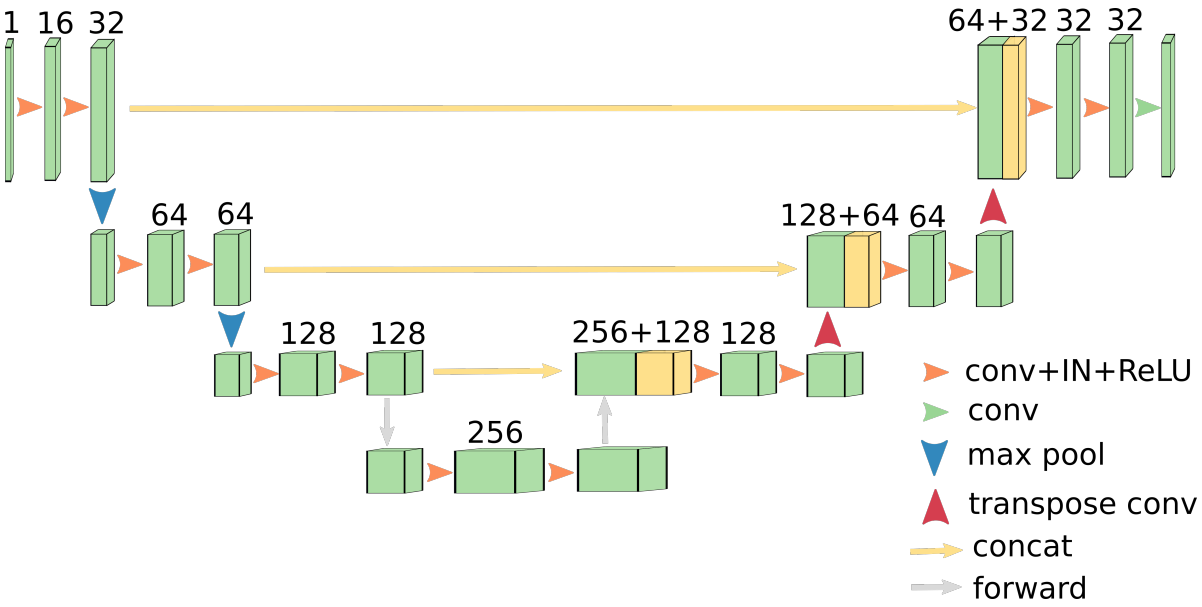


Figure 3.4.: 3D U-net used as baseline architecture. Numbers above the feature map denote the number of channels.

The instance normalization layer is followed by a ReLU layer. The first and second encoder block perform downsampling as the last step, using $2 \times 2 \times 2$ max-pooling with stride 2. The decoder blocks follow a similar set-up. First, the input feature map and the forwarded feature map from the corresponding encoder block get concatenated in the channel dimension. In the second and last decoder block, the input feature map first needs to be upsampled using transpose convolutions, to match the size of the forwarded feature map. Then the concatenated feature map is passed to 2 convolutional layers followed by instance normalization and ReLU.
The bottleneck layer doesn't perform any up- or downsampling, it just contains 2 convolutional layers + instance norm + ReLU.
The number of output channels at every encoder/decoder block was reduced to half of original size used in 3D U-Net to keep the model complexity low. For example, the two convolutions in encoder 1 have number of channels $\{16, 32\}$ instead of $\{32, 64\}$ and so on. The last layer of the network is a convolutional layer with kernel size $1 \times 1 \times 1$ and output channel size equal to the number of classes to predict, followed by a softmax layer. Figure 3.4 illustrates the baseline architecture. The changes to the original 3D

U-net architecture are justified by experiments in the next chapter 4.

3.2.1. Loss Function

A combined weighted loss function composed of Cross Entropy loss and Dice loss was used, inspired by the work in [15, 14], which is defined as:

$$L = -\sum_{\mathbf{x}} w(\mathbf{x}) g_l(\mathbf{x}) \log(p_l(\mathbf{x})) - \frac{2\sum_{\mathbf{x}} g_l(\mathbf{x}) p_l(\mathbf{x})}{\sum_{\mathbf{x}} g_l^2(\mathbf{x}) + \sum_{\mathbf{x}} p_l^2(\mathbf{x})}, \tag{3.11}$$

where $p_l(\mathbf{x})$ denotes the predicted probability of voxel $\mathbf{x}$ belonging to class l and $g_l(\mathbf{x})$ the ground truth probability. The first term describes the multi class cross entropy loss and the second term is the multi class dice loss. Where $w(\mathbf{x})$ is a weighting factor, described in [15, 14]. For the vector of label frequencies $\mathbf{f} = [f_1, ..., f_C]$, the frequency of class l in the training volume f_l and the ground truth segmentation S, the weighting factor $w(\mathbf{x})$ is defined as:

$$w(\mathbf{x}) = \sum_{l} I(S(\mathbf{x} = l)) \frac{median(\mathbf{f})}{f_l} + w_0 I(|\nabla S(\mathbf{x})| > 0), \tag{3.12}$$

where I is the indicator function and ∇ the gradient operator. The first term tackles the problem of class imbalance by giving higher weights to smaller classes using median frequency balancing. The second term gives higher importance to boundary regions, where the absolute gradient is higher than 0. The factor w_0 balances both terms and is set to $\frac{2 \cdot median(\mathbf{f})}{f_{min}}$.

3.3. Implementation Details

The implementation was done using python 3.6.8 and pytorch version 1.0.
To obtain segmentation maps that match the input size I used zero-padded convolutions in all layers. For implementation of the Squeeze and Excitation modules I used the existing 2D implementation by Roy et al. [16, 2] as baseline and extended it to 3D. Implementations for all 3D SE models developed in this thesis can be found in the same

repository [2].
I had access to two Nvida Titan Xp GPUs with 12GB RAM and one Nvidia quadro P6000 GPU with 24GB RAM.
To be able to train the 3D networks with full volume inputs on the Titan Xp GPUs I used the pytorch checkpointing functionality in order to stay below the RAM limit of 12GB. This functionality works by trading compute for memory [1]. It's done by not saving the intermediate results in the forward pass, but recomputing the forward pass during backpropagation. This way memory is saved, but computational time is increased. Processing a whole volume of the brain using the proposed 3D U-net with PE blocks, including loading the volume and saving the prediction, takes on average 0.8 seconds on a Titan Xp GPU.

4. Experiments

4.1. Datasets

Although the main focus of the thesis is brain segmentation, I used two different datasets to analyze the performance of the proposed PE blocks, to show the module is general and can be used for different segmentation tasks.

4.1.1. Whole-Brain Segmentation

For whole-brain segmentation I used the Multi-Atlas Labelling Challenge (MALC) dataset [7], which is part of OASIS [9]. It contains 30 T1 MRI volumes of the brain. I segment the brain volumes into 32 cortical and subcortical structures. The main challenge associated with the dataset is the limited training data with severe class imbalance between the target structures. Manual segmentations for MALC were provided by Neuromorphometrics, Inc.. 15 scans were used for training, 3 scans for validation and the remaining 12 scans for testing. The 32 classes to segment are: Left White Matter, Left Cortex, Left Lateral Ventricle, Left Inferior Lateral Ventricle, Left Cerebellum White Matter, Left Cerebellum Cortex, Left Thalamus, Left Caudate, Left Putamen, Left Pallidum, 3rd Ventricle, 4th Ventricle, Brain Stem, Left Hippocampus, Left Amygdala, CSF (Cranial), Left Accumbens, Left Ventral DC, Right White Matter, Right Cortex, Right Lateral Ventricle, Right Inferior Lateral Ventricle, Right Cerebellum White Matter, Right Cerebellum Cortex, Right Thalamus, Right Caudate, Right Putamen, Right Pallidum, Right Hippocampus,Right Amygdala, Right Accumbens and Right Ventral DC. Figure 4.1 shows the severe class imbalance exemplary on labels of the left hemisphere.

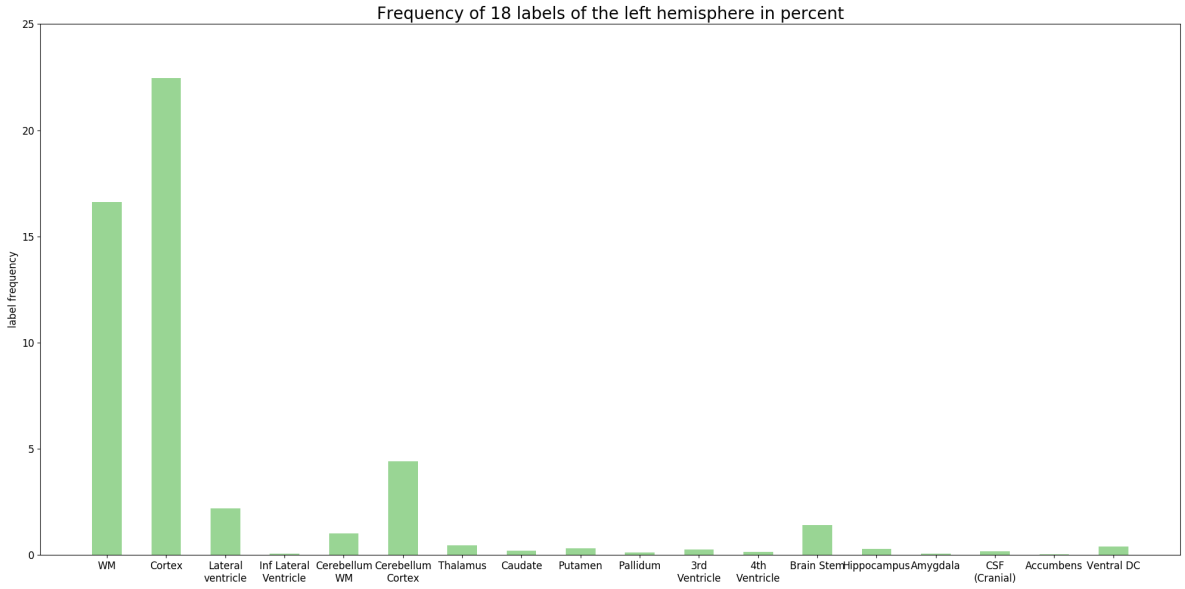


Figure 4.1.: Class frequencies of the left hemisphere of the brain

4.1.2. Whole-Body Segmentation

For this task, I used the contrast enhanced CT scans of the Visceral dataset [6]. Training and Testing was done using the gold corpus of the dataset, which contains 20 annotated scans. I performed 5-fold cross-validation, where one scan from the test fold was kept as validation set. I segment 14 organs from thorax and abdomen. I chose these 14 classes, since other classes are not annotated in all 20 scans. Both datasets have common challenges w.r.t the limited amount of training scans and severe class-imbalance across the target classes. The 14 segmented classes are: Liver, Spleen, Urinary bladder, Aorta, Trachea, Right Lung, Left Lung, Sternum, Right Kidney, Left Kidney, Right Psoas Muscle, Left Psoas Muscle, Right Rectus Abdominis Muscle and Left Rectus Abdominis Muscle. Figure 4.2 demonstrates the class imbalance problem of the visceral dataset.

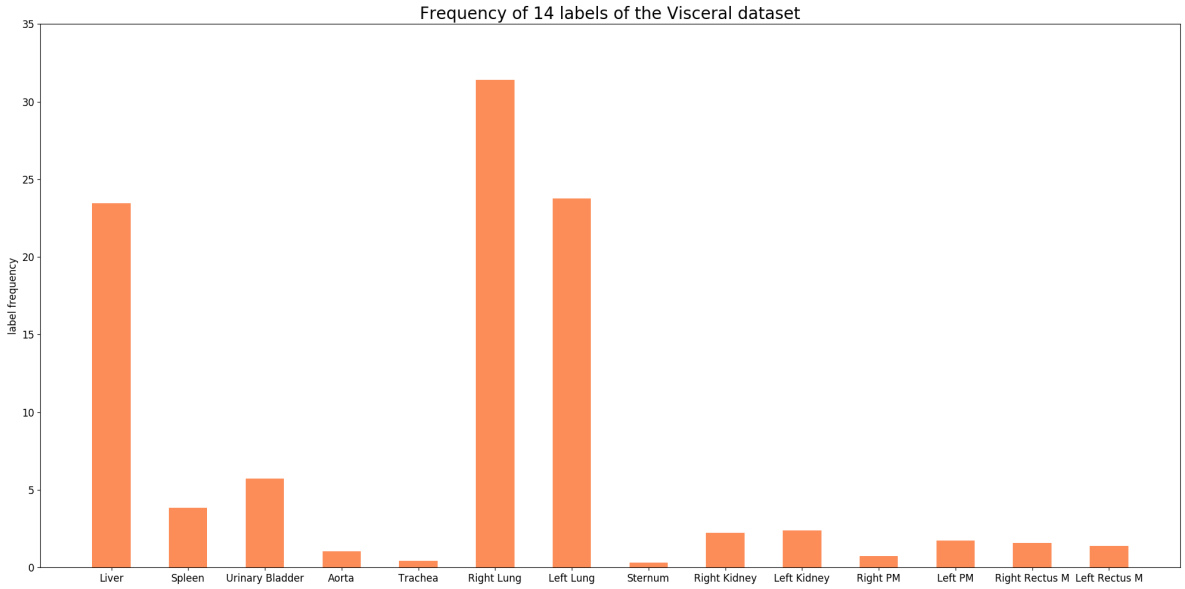


Figure 4.2.: Class frequencies of the 14 labels of visceral dataset

4.2. Training Setup:

I used the modified 3D U-Net [3] architecture that was described in chapter 3.2 for experimental purposes. The design consists of 3 encoder and 3 decoder blocks, with only the first two encoders performing downsampling, and the last two decoders performing upsampling. I performed experiments on MALC Dataset to verify this architectural configuration. With 3 up/down sampling configuration a mean Dice score of 0.774 was observed. With only 1 up/down sampling configuration a mean Dice score of 0.768 was observed. With 2 up/down sampling configuration the mean Dice score observed was **0.802**. Thus I chose this configuration in all experiments. When training the 3D U-net using batch-normalization, an average Dice Score of 0.777 was achieved on the test set. During training I observed the validation loss and accuracy to be very noisy. When using instance normalization instead, the average Dice Score increased to **0.802** and the validation loss and accuracy curves were much smoother, so instance normalization was used in all following experiments.

4.3. Training Parameters:

Due to the large and variable dimensions of the input volumes I chose a batch size of 1 for training purpose. Also, this configuration totally occupied the 12 GB RAM of the TITAN Xp GPU. Optimization was done using SGD with momentum of 0.9. The learning rate was initially set to 0.1 and was reduced by a factor of 10 when validation loss plateaued. On-the-fly data augmentation using elastic deformations and random rotations was performed on the training set. I used a combined Cross Entropy and Dice loss with the Cross Entropy loss being weighted using median frequency balancing to tackle the high class imbalance, similar to [15].

Part III.

Results and Conclusion

5. Results

5.1. Position of PE Blocks

In this section, I investigate the positions at which our proposed 'Project & Excite' (PE) blocks need to be placed within the 3D U-Net architecture. I explored 6 possibilities by placing them after every encoder block (P1), after every decoder block (P2), after the bottleneck block (P3), after both encoder and decoder blocks (P4), after each encoder block and bottleneck (P5), and finally after all the blocks (P6). I present the results of all these configurations for MALC dataset in Tab. 5.1 and compared against having no 'PE' blocks. I observed that placing the blocks after every encoder, decoder and bottleneck provided the best accuracy, boosting by 4% Dice points.

Table 5.1.: Mean Dice score on MALC dataset due to placement of 'PE' blocks within 3D U-Net architecture.

	Position of 'PE' block			
	Encoders	Bottleneck	Decoders	Mean Dice ± std
3D U-Net	✗	✗	✗	0.802 ± 0.171
P1	✓	✗	✗	0.828 ± 0.111
P2	✗	✗	✓	0.796 ± 0.215
P3	✗	✓	✗	0.822 ± 0.144
P4	✓	✗	✓	0.819 ± 0.159
P5	✓	✓	✗	0.818 ± 0.156
P6	✓	✓	✓	$\mathbf{0.843} \pm 0.079$

Also, placing it after encoder and bottleneck blocks improves the Dice score by 2% points, whereas placing it after decoder blocks does not affect the performance. I conclude that 'PE' blocks are most effective in encoder and bottleneck positions of F-CNN. In the following experiments, I place the 'PE' blocks after every encoder, decoder and bottleneck blocks.

5.2. Model Complexity:

Here I investigate the increase of model complexity due to addition of 'PE' blocks within 3D U-Net architecture. I compare the PE blocks with 3D cSE and 3D sSE blocks complexity-wise and report them in Tab. 5.2. The experiments have been conducted using the MALC dataset. It can be observed, that both PE blocks and cSE blocks cause the same fraction of 1.97% increase in model complexity, whereas when using sSE blocks model complexity stays almost constant. One might think that the boost in performance is due to the added complexity, which might also be gained by adding more convolutional layers. I investigated this matter by conducting two more experiments. First, I added an extra encoder and decoder block within the architecture. This immensely increased the model complexity by almost 40% and I observed a drop in Dice performance. One possible reason might be due to over-fitting given the limited data samples and sudden increase in model complexity.

Table 5.2.: Mean Dice vs model complexity measured in number of trainable parameters

	Dice	Complexity
3D-Unet	0.802	$5.57 \cdot 10^6$
+ 3D cSE	0.825	+1.97%
+ 3D sSE [1]	0.831	+0.01%
+ PE	**0.843**	+1.97%
+ Encoder/Decoder	0.779	+39.7%
+ 2 conv layers	0.826	+3.97%

So, next I only added two additional convolutional layers at the second encoder and second decoder to make sure that the increase in model complexity is only marginal ($\sim 4\%$), not risking over-fitting. Here, I did observe a boost in performance similar to cSE with double the increase in parameters, but still failed to match the performance of our PE blocks. Thus, I conclude that PE blocks are in fact more effective than simply adding convolutional layers.

5.3. Segmentation Results:

The results of whole-brain segmentation and whole-body segmentation are presented in Tab. 5.3. I compared 'PE' blocks to the 3D channel SE (cSE) blocks [18], 3D spatial SE (sSE) blocks [16] and the baseline 3D U-Net. The placement of the cSE and sSE blocks in the architecture was kept identical to the placement of PE blocks, discussed in section 5.1. For brain segmentation, the results are consistent with [16], showing the overall mean Dice score increases by 2% by using cSE blocks, while sSE blocks lead to a higher increase of 3%.

Table 5.3.: Comparison of 3D U-Net with 3D cSE, 3D sSE and the proposed PE block. Mean Dice scores for selected classes of MALC and Visceral datasets. In the top table WM stands for white matter and GM for grey matter. In the bottom table L. stands for left and R. for right.

	MALC dataset					
	Mean Dice ± std	WM	GM	Inf. Lat. Vent.	Amygdala	Accumbens
3D U-Net	0.802 ± 0.171	0.906	0.887	0.242	0.761	0.483
+ 3D cSE	0.825 ± 0.119	0.907	0.888	0.403	0.761	0.704
+ 3D sSE [1]	0.831 ± 0.108	0.909	0.894	0.454	0.782	0.734
+ Project & Excite	$\mathbf{0.843} \pm 0.079$	**0.916**	**0.899**	**0.604**	**0.789**	**0.735**
	Visceral dataset					
	Mean Dice ± std	Liver	R. Lung	R. Kidney	Trachea	Sternum
3D U-Net	0.810 ± 0.137	0.922	0.965	0.907	0.815	0.438
+ 3D cSE	0.797 ± 0.168	0.930	0.966	0.919	0.491	0.427
+ 3D sSE [1]	0.765 ± 0.269	0.928	0.966	0.921	0.145	0.162
+ Project & Excite	$\mathbf{0.846} \pm 0.095$	0.931	0.966	**0.929**	**0.845**	**0.699**

This confirms the advantage of spatial recalibration for segmentation tasks. The proposed PE blocks however result in an increase of 4%, outperforming even the sSE blocks, substantiating its efficacy in 3D architectures. For whole body segmentation, the mean Dice score by using 3D cSE even decreases by 1% and using 3D sSE leads to an decrease of 4%, while, when using PE blocks, it increases by 3.5%. The decrease in average dice score for cSE and sSE is due to the missing segmentation of trachea and sternum. The performance on other structures is similar to the baseline U-net. Further, I explored the impact of PE blocks on some selected structures. Firstly, I selected bigger structures, white and grey matter for brain segmentation, and liver and right lung for whole-body segmentation. The boost in Dice score for white and grey matter was very marginal by using either cSE, sSE or PE blocks ranging within 1%. For liver and right lung the performance using cSE, sSE or PE blocks is comparable to the baseline 3D U-Net. Next, I analyzed some smaller structures, namely inferior lateral ventricles, amygdala and accumbens for brain segmentation, and right kidney, trachea and sternum for whole body segmentation, which are difficult to segment. These smaller structures seem to be affected more by the choice of SE blocks which perform spatial recalibration. While using cSE blocks also increases the performance in inferior lateral ventricles and accumbens, they even lead to decreasing performance for trachea and sternum. The spatial SE blocks lead to an even higher increase in performance for inferior lateral ventricles (20%) and accumbens (25%), indicating the efficacy of these blocks for segmenting very small structures. PE blocks, however result in an immense boost in performance in these structures ranging between $3 - 36\%$, outperforming cSE and sSE blocks. The segmentation accuracy of inferior lateral ventricles has to be emphasized here, where PE blocks lead to an increase of 36% Dice points. In whole-body segmentation the spatial SE model misses segmentation of trachea and sternum mostly, which leads to poor results on these structures. This is unexpected, since sSE blocks are known to be highly effective for small structures and needs to be further investigated in future work. In Fig. 5.1, I present visualizations of the segmentation performance of PE models in comparison to baseline 3D U-Net and 3D cSE models. In the top row, white arrows indicate the region of left inferior lateral ventricle, which was missed by both 3D U-Net and 3D csE models. The proposed PE model, however, was able to segment this very

small structure. In the bottom row, white arrows point to the bifurcation of the trachea, where the 3D U-Net is oversegmenting the right lung and 3D cSE model is missing the trachea completely. In conclusion, I observed similar trends in both, whole-brain and whole-body segmentation, demonstrating the efficacy of PE blocks for segmentation of small structures in 3D scans. Figure 5.2 presents box-plots of the segmentation results on MALC and Visceral datasets for all classes with Dice score as the underlying evaluation metric. For MALC dataset structures of left and right hemisphere were combined into one class by averaging, due to space constraints.

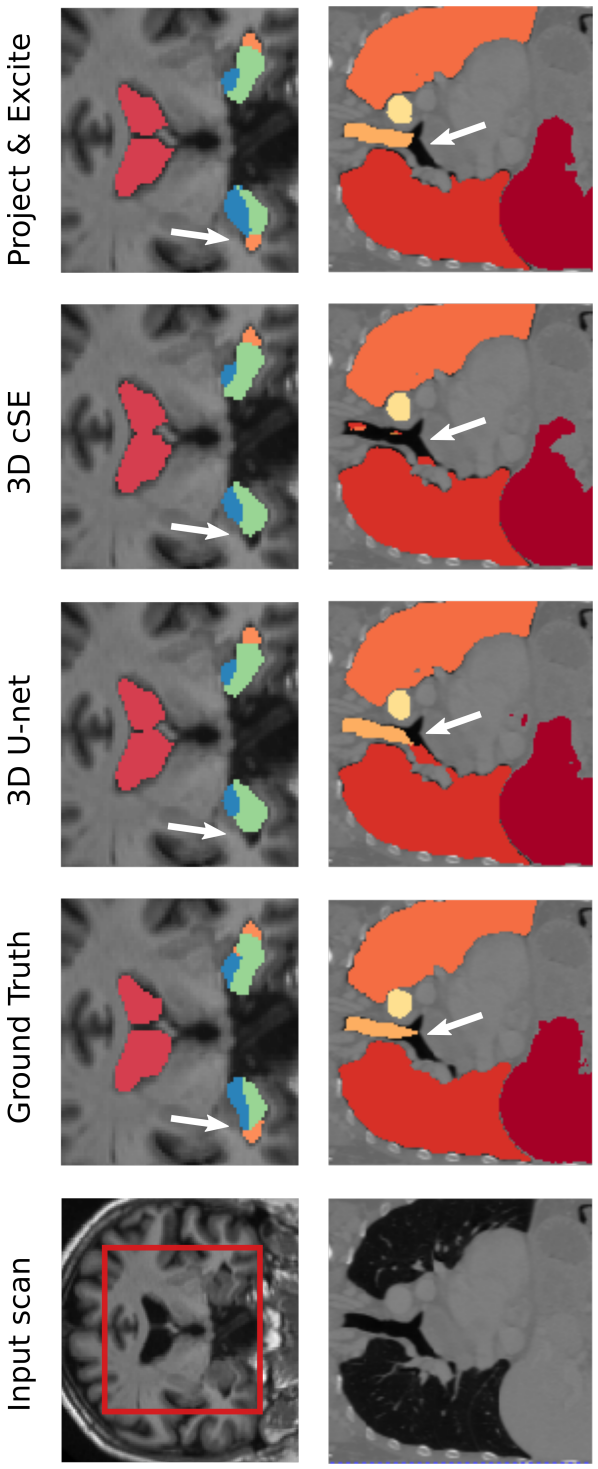


Figure 5.1.: Input scans, ground truth segmentation and results for 3D U-Net, 3D cSE and the proposed PE model, for MALC (top row) and Visceral (bottom row) datasets. Red box defines a ROI for segmented structures, white arrows point to the structures where our PE model improved the performance.

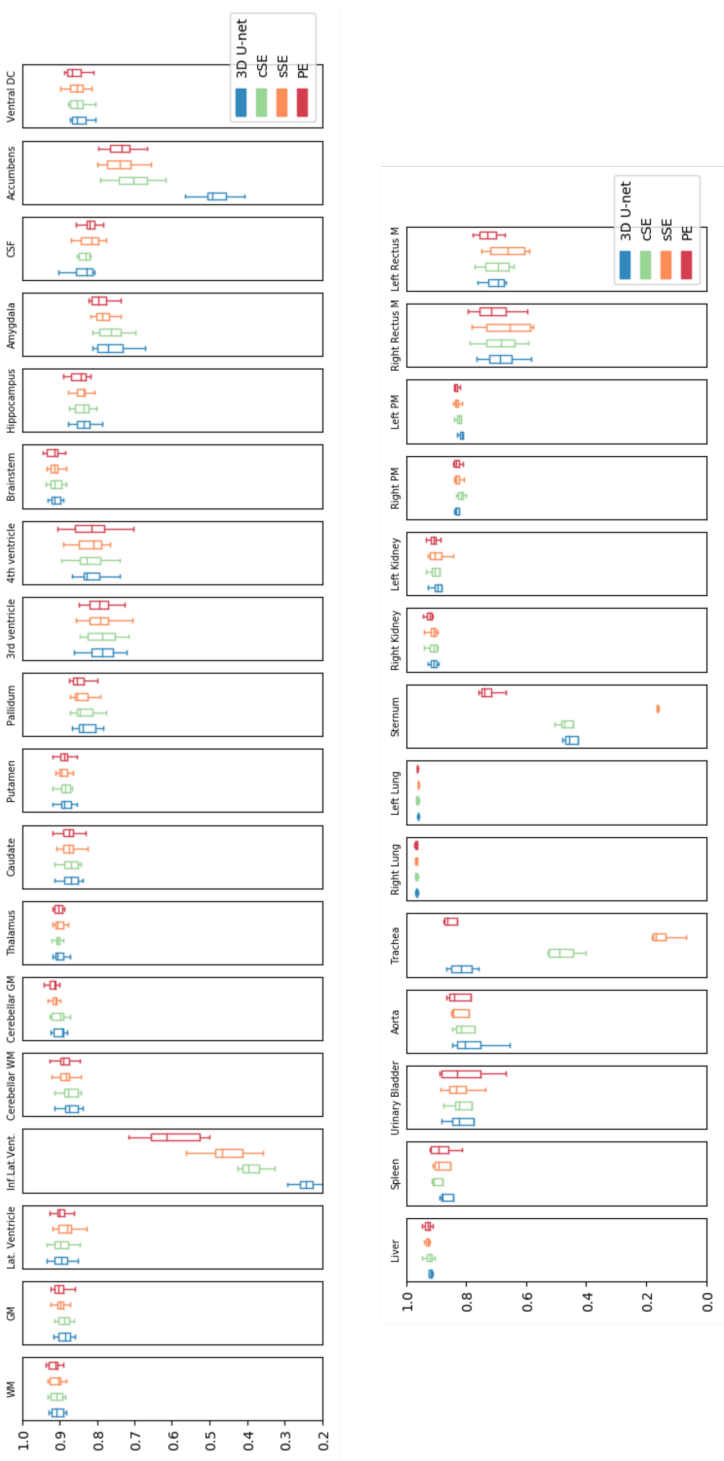


Figure 5.2.: Boxplot comparing 3D U-Net, cSE, sSE and the proposed PE model, for MALC dataset (top) and Visceral dataset (bottom). Due to space constraints structures of left and right hemisphere have been averaged and treated as one class.

6. Conclusion

In this thesis I proposed 'Project & Excite', a light-weight recalibration module that can be easily integrated within any 3D F-CNN architecture and boosts segmentation performance while only increasing model complexity by a small fraction. I demonstrated that PE blocks can be an attractive alternative to adding more convolutional layers in 3D F-CNNs, especially in situations where training data and GPU resources are limited. I exhibited the effectiveness of 'PE' blocks by conducting experiments on two challenging tasks of whole-brain and whole-body segmentation, comparing them to 3D extensions of previously proposed Squeeze and Excite modules. PE blocks have outperformed the baseline 3D U-net model without and with other 3D SE blocks, indicating the need for blocks that are customarily designed for 3D architectures. In contrast to previous work, cSE and sSE blocks have not increased performance on the visceral dataset. This is mainly due to completely missing segmentations of few small classes. It would be interesting to conduct further experiments focused on small anatomies to identify possible reasons for this behaviour. The experiments in this thesis have evaluated the performance of 'PE' blocks but not the impact they have in different stages of the network. Although this has been covered by previous work on squeeze and excitation modules, it can not simply be transferred to 'PE' modules, so there is a need for further investigation. In this thesis I have integrated PE blocks into 3D U-net, in future work the performance of PE blocks integrated into other F-CNN architectures can be evaluated. Further, other pooling methods in the Project operation, e.g. max-pooling or combinations of max- and average-pooling could be assessed.

Bibliography

[1] PyTorch Checkpointing. `https://pytorch.org/docs/stable/checkpoint.html`. Accessed: 2019-04-05.

[2] Squeeze and Excitation Implementation. `https://github.com/ai-med/squeeze_and_excitation`. Accessed: 2019-04-05.

[3] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *MICCAI*, pages 424–432. Springer, 2016.

[4] Jose Dolz, Christian Desrosiers, and Ismail Ben Ayed. 3d fully convolutional networks for subcortical segmentation in mri: A large-scale study. *NeuroImage*, 170:456–470, 2018.

[5] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, June 2018.

[6] Oscar Jimenez-del Toro, Henning Müller, Markus Krenn, Katharina Gruenberg, Abdel Aziz Taha, Marianne Winterstein, Ivan Eggel, Antonio Foncubierta-Rodríguez, Goksel, et al. Cloud-based evaluation of anatomical structure segmentation and landmark detection algorithms: Visceral anatomy benchmarks. *IEEE TMI*, 35(11):2459–2475, 2016.

[7] B Landman and S Warfield. Miccai 2012 workshop on multi-atlas labeling. In *MICCAI*, 2012.

[8] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[9] Daniel S Marcus, Anthony F Fotenos, John G Csernansky, John C Morris, and Randy L Buckner. Open access series of imaging studies: longitudinal mri data in nondemented and demented older adults. *Journal of cognitive neuroscience*, 22(12):2677–2684, 2010.

[10] Fausto Milletari, Seyed-Ahmad Ahmadi, Christine Kroll, Annika Plate, Verena Rozanski, Juliana Maiostre, Johannes Levin, Olaf Dietrich, Birgit Ertl-Wagner, Kai Bötzel, et al. Hough-cnn: deep learning for segmentation of deep brain regions in mri and ultrasound. *Computer Vision and Image Understanding*, 164:92–102, 2017.

[11] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571. IEEE, 2016.

[12] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *ICCV*, pages 1520–1528, 2015.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015.

[14] Abhijit Guha Roy, Sailesh Conjeti, Nassir Navab, and Christian Wachinger. Quicknat: A fully convolutional network for quick and accurate segmentation of neuroanatomy. *NeuroImage*, 186:713–727, 2019.

[15] Abhijit Guha Roy, Sailesh Conjeti, Debdoot Sheet, Amin Katouzian, Nassir Navab, and Christian Wachinger. Error corrective boosting for learning fully convolutional networks with limited data. In *MICCAI*, pages 231–239. Springer, 2017.

[16] Abhijit Guha Roy, Nassir Navab, and Christian Wachinger. Recalibrating fully convolutional networks with spatial and channel squeeze and excitation blocks. *IEEE TMI*, 38(2):540–549, 2019.

[17] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*, pages 6924–6932, 2017.

[18] Wentao Zhu, Yufang Huang, Liang Zeng, Xuming Chen, Yong Liu, Zhen Qian, Nan Du, Wei Fan, and Xiaohui Xie. Anatomynet: Deep learning for fast and fully automated whole-volume segmentation of head and neck anatomy. *Medical physics*, 46(2):576–589, 2019.