Deep Learning with Principal Component Wavelet Networks

CS39440 Major Project Report

Author: Samuel Spurling ([sas90@aber.ac.uk](mailto:sas90@aber.ac.uk))

Supervisor: Dr Bernie Tiddeman. My Supervisor (bpt@aber.ac.uk)

29th March 2022

Version 1.0 (Draft)

This report is submitted as partial fulfilment of a BSc degree in  
Computer Science and Artificial Intelligence (With Integrated Year in Industry) (GG47)

Department of Computer Science

Aberystwyth University

Aberystwyth

Ceredigion

SY23 3DB

Wales, UK

Declaration of originality

I confirm that:

* This submission is my own work, except where clearly indicated.
* I understand that there are severe penalties for Unacceptable Academic Practice, which can lead to loss of marks or even the withholding of a degree.
* I have read the regulations on Unacceptable Academic Practice from the University’s Academic Registry (AR) and the relevant sections of the current Student Handbook of the Department of Computer Science.
* In submitting this work, I understand and agree to abide by the University’s regulations governing these issues.

Name …………………………………………

Date ……………………………………………

Consent to share this work

By including my name below, I hereby agree to this project's report and technical work being made available to other students and academic staff of the Aberystwyth Computer Science Department.

Name …………………………………………

Date ……………………………………………

Acknowledgements

I am grateful to…

I’d like to thank…

Abstract

In this report we propose a novel application for Principal Component Wavelet Networks (PCWN) [1]. The PCWN decomposes an image using handcrafted invertible 2D wavelet filters-banks and 1x1 filters learnt through principal component analysis to control the size of decompositions. The reconstruction is accomplished using the inverted filter bank and an approximately inverted PCA. Previously the approach has shown competitive results for linear inverse problems in computer vision such as compressive sensing, super-resolution, and in-painting. The proposed novel application is image segmentation. This is a task where there are objects in an image and through segmentation the image is split into distinct components, or segments, where each segment represents an object in the image.

We use 2 variants of the PCWN, a fully convolutional and a fully connected approach. In experiments, there will be a comparison of the PCWN approach to the U-net model, a standard architecture for image segmentation, to benchmark the PCWN models performance. The results of the experiments demonstrate that PCWN has great potential when applied to the task of image segmentation. Possible future work looks at using PCWN as regularisation in the U-net architecture and replacing PCA with a different kind of invertible dimensionality reduction such as Kernel PCA or UMAP.

[Figure 1: Wavelet decomposition of an image of a bird 12](https://d.docs.live.net/455c1cc37aadd471/Desktop/major_project/MajorProject/Writeup/Writeup/MajorProject.docx#_Toc101287641)

[Figure 2: PCA training process [1] 12](#_Toc101287642)

[Figure 3: Fully connected PCWN architecture 14](#_Toc101287643)

[Figure 4: Fully convolutional PCWN architecture 14](https://d.docs.live.net/455c1cc37aadd471/Desktop/major_project/MajorProject/Writeup/Writeup/MajorProject.docx#_Toc101287644)

[Figure 5: U-net architecture [3] 16](https://d.docs.live.net/455c1cc37aadd471/Desktop/major_project/MajorProject/Writeup/Writeup/MajorProject.docx#_Toc101287645)

[Figure 6: Example of high-resolution features being used to localize low-resolution features [3] 16](https://d.docs.live.net/455c1cc37aadd471/Desktop/major_project/MajorProject/Writeup/Writeup/MajorProject.docx#_Toc101287646)

[Figure 7: Atrous convolution: Atrous rate = 1 17](https://d.docs.live.net/455c1cc37aadd471/Desktop/major_project/MajorProject/Writeup/Writeup/MajorProject.docx#_Toc101287647)

[Figure 8: Atrous convolution: Atrous rate = 2 17](https://d.docs.live.net/455c1cc37aadd471/Desktop/major_project/MajorProject/Writeup/Writeup/MajorProject.docx#_Toc101287648)

Contents

[1 Background, Analysis & Process 9](#_Toc101287609)

[1.1 Background 9](#_Toc101287610)

[1.2 Analysis 9](#_Toc101287611)

[1.2.1 Fundamentals 9](#_Toc101287612)

[1.2.2 Architectures for Segmentation 11](#_Toc101287613)

[1.2.3 Research Questions 17](#_Toc101287614)

[1.3 Process 17](#_Toc101287615)

[2 Experiment Methods 18](#_Toc101287616)

[3 Software Design, Implementation and Testing 19](#_Toc101287617)

[3.1 Design 19](#_Toc101287618)

[3.1.1 Overall Architecture 19](#_Toc101287619)

[3.1.2 Some detailed design 19](#_Toc101287620)

[3.1.3 User Interface 20](#_Toc101287621)

[3.1.4 Other relevant sections 20](#_Toc101287622)

[3.2 Implementation 20](#_Toc101287623)

[3.3 Testing 20](#_Toc101287624)

[3.3.1 Overall Approach to Testing 21](#_Toc101287625)

[3.3.2 Automated Testing 21](#_Toc101287626)

[3.3.3 Integration Testing 21](#_Toc101287627)

[3.3.4 User Testing 21](#_Toc101287628)

[4 Results and Conclusions 22](#_Toc101287629)

[4.1 Issues with experiments 23](#_Toc101287630)

[4.1.1 Memory errors 23](#_Toc101287631)

[4.2 Experimental Results: Determining best parameters for segmentation 23](#_Toc101287632)

[4.2.1 Convolutional Network approach 23](#_Toc101287633)

[4.3 Experimental Results: Convolutional vs Connected PCWN 24](#_Toc101287634)

[4.4 Experimental Results: Connected PCWN vs U-net 24](#_Toc101287635)

[5 Critical Evaluation 24](#_Toc101287636)

[6 References 26](#_Toc101287637)

[7 Appendices 27](#_Toc101287638)

[A. Third-Party Code and Libraries 28](#_Toc101287639)

[B. Code Samples 29](#_Toc101287640)

# Background, Analysis & Process

## Background

The motivation for this project stems from an interest in computer vision, dimensionality reduction techniques, and a curiously for wavelets. Image segmentation is an active field of research, with applications in medical imaging, satellite imagery and image-based search.

The project extends the PCWN method [1] to the task of image segmentation, so naturally PCWN was a key part of the background research. The motivation for this was to understand the novel approach to linear inverse problems and apply my findings to an image segmentation approach. Convolutional scattering wavelet networks [2] is another method which shares similarities to PCWN, especially on methods for mitigating the issue of large channels after multiple applications of wavelet filter banks. To understand the segmentation task, there was research of U-net [3], DeepLabV3 [4] and a comparison of segmentation methods [5].

## Analysis

### Fundamentals

#### Principal Component Analysis

Principal Component Analysis (PCA) is a method of dimensionality reduction that calculates “principal components” and then orients the data using the principal components as basis vectors. The advantage of using PCA is that high dimensional data can be expressed in lower dimensions whilst retaining almost all the variance between data points. Consequently, the curse of dimensionality [6] can be avoided when using PCA in machine learning, where higher dimensions cause an increase in sparsity, causing data objects to appear dissimilar which impacts the algorithms’ ability to learn.

Given an input matrix of data A, this is how to calculate the principal components. First the covariance matrix and the mean of the data is calculated. The mean is subtracted from the data to standardise the data.

The principal components are found by calculating the eigenvectors and eigenvalues of the covariance matrix, and then the eigenvectors are ordered using the eigenvalues. A n-dimensional dataset has n principal components. The sum of the eigenvalues of n principal components is the total variance, therefore we can calculate explained variance by dividing the eigenvalue of a component by the sum of the eigenvalues. The total explained variance is found by summing the explained variance for each component selected. The following is an example of eigenvectors and eigenvalues .

The eigenvectors of the covariance matrix are the direction of the axes which have the highest variance, and the eigenvalues are the amount of variance in the respective components. The feature vector is formed using the eigenvectors like so.

We are applying PCA to the feature maps produced by the filter banks. Finding the dot product of the feature maps (A) and the transpose of the feature vector (B) produces a dimensionally reduced feature map (C) that retains most of the information of the higher dimension feature map.

#### Convolutional Neural Networks

An artificial neural network (ANN) is a directed and weighted graph where neurons are nodes and neuron connections (weights) are edges. ANNs are composed of layers, where every layer is a collection of neurons. The most common architecture is feed-forward. In this architecture every layer is connected to another layer, except the output layer, which only has a incoming connections. ANNs typically consist of three types of layers, input, output and hidden. The hidden layers of an ANN allow them to function as universal function approximators. That is to say, with the correct architecture and parameters an ANN can model any function to a small degree of error. Finding the correct architecture and parameters is a complex task with no general solution, and there are no guarantees of optimality. Typically, ANN’s are trained using vast amounts of data and an error update algorithm that tweaks the parameters of the network towards a more optimal solution. Normally this algorithm is backpropagation [7].

There are many architectures other than feed-forward networks, such as Recurrent Neural Networks (RNNs), transformers and Self-Organising Maps (SOMs). The architecture used for this project is Convolutional neural networks (CNNs).

CNNs are a commonly used architecture in computer vision. Popularised by Kunihiko Fukushima [8] and Yann LeCun [9] between 1980-1990, CNNs have a number of desirable qualities for computer vision tasks. CNN’s work using convolutional filters, which are 2D matrices, or kernels, that slide over the image and “convolve” with patches on the image. A convolution measures the strength of a signal at a particular patch on the image. Multiple kernels can be used in parallel to generate deep feature maps, and the kernels can have any size. The size of the steps taken when sliding the kernel over an image is called the stride. A stride of 2 means that a pixel is skipped with every step. Below is the equation used to calculate the value of a 1D feature map, y, at location [m, n]. The variable x is the image we are convoluting, and h is the feature map. The operator is used as shorthand for a convolution operation.

The kernel values are learnt through error update, the same as any other neural network. Typically, they are initialised randomly, although there are a number of handcrafted filters which can be used to initialise the kernels. For example, there are filters for edge detection and gaussian blurring.

The main advantage of using convolutions instead of a fully connected solution is the massive reduction in connections in the network relative to a standard densely connected network. In a feed-forward network, the number of connections scales exponentially with the resolution of the image. Every one of these connections requires an error update when training, which introduces a massive computational overhead. The network will also struggle to generalise, resulting in overfitting. A CNN uses convolutions and pooling to minimise the connections between layers and lead to more efficient computation. Pooling is an operation which further reduces the size of feature maps. Max-pooling, for example, takes the largest value in a small window, discarding the other values.

#### Wavelets

Appropriate image representation is key to the success of an algorithm. There exist several different representations. The most familiar representation is the spatial domain. The spatial domain is how we view digital images, with resolution, colour channels and pixel intensity values. Other representations include the Fourier domain and wavelets.

Wavelets transforms are similar to Fourier transforms, except that whereas Fourier can only capture frequency information, wavelets can capture frequency and location information. Both Fourier and wavelet transforms measure signal strength. For Fourier transforms, this means which frequencies have the largest amplitudes in the frequency domain of an image. For wavelet transforms, signal strength is a measure of how much of a wavelet is in a signal. This may sound similar to a convolution, and that’s exactly what a wavelet transform is. This is shown by the discrete wavelet transform formula below, where W is a bank of wavelet filters, t is the location in the image, x is the image and y is the output feature map.

Wavelets can be classified as discrete or continuous. Wavelets have been in use since the early 20th century, originating with Harr wavelets [10] which have enjoyed a wide range of applications, notably in Viola-Jones facial detection [11]. Harr wavelets are discrete wavelet transforms. There are many applications of wavelets, for example scattering wavelets are used in convolutional scattering wavelet networks [2]. This method in the paper used sets of different scale Morlet wavelets, an approach which has desirable properties such as deformation, rotation, and scale invariance. Morlet wavelets do not have a simple inversion, requiring direct optimisation [12] or generative networks [13] to solve them. In PCWN the filter banks are based on approximations of gaussian derivatives.

### Architectures for Segmentation

#### PCWN

The PCWN architecture is a hybrid of CNNs and wavelet transforms. The network consists of a contracting path, which decomposes an input, and an expanding path, which reconstructs the input.

##### Decomposition Step

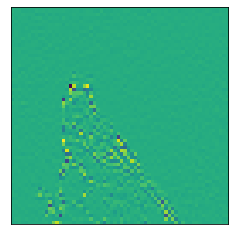
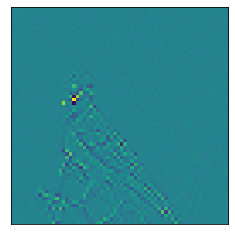
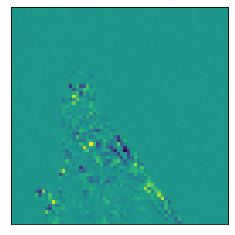
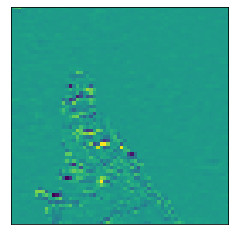
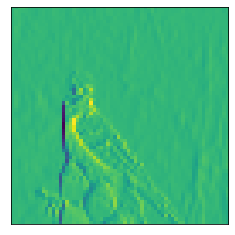
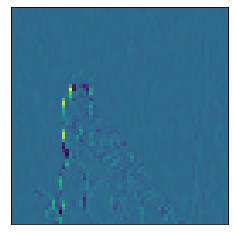
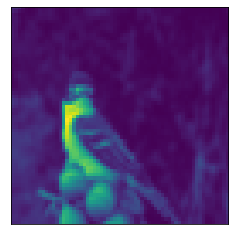
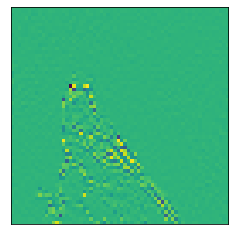
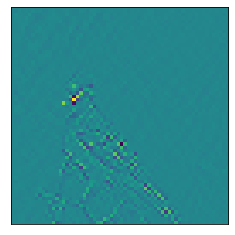
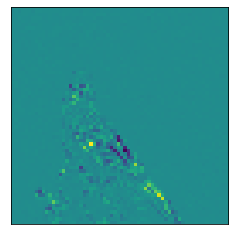
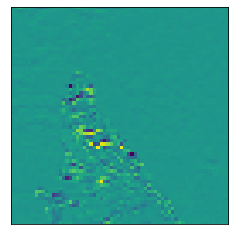
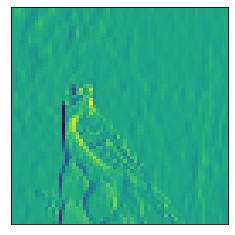
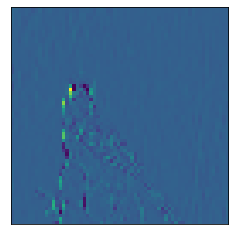
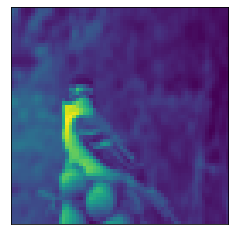
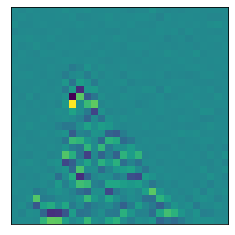
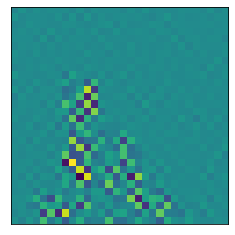
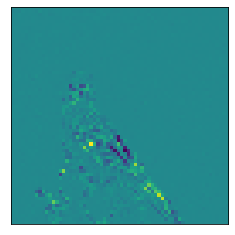
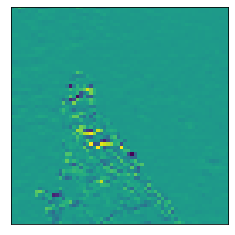
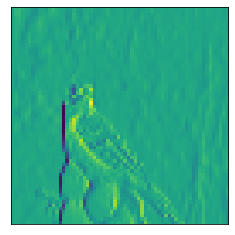
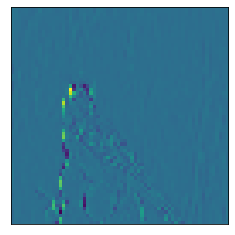
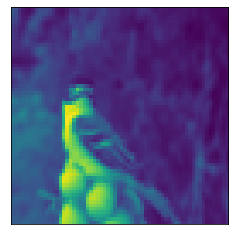
The bank of wavelet filters used in PCWN are discrete and invertible. At every level of decomposition, each filter in the filter bank is applied to each channel in the partial decomposition. The filter bank used is based on approximations of the zeroth, first and second derivatives of the gaussian, applied along the x and y direction. Every input channel outputs 9 more channels in the next decomposition. This process is shown by the following formula:

Where is the decomposition at tensor channel at level before PCA is applied. is the decomposition tensor channel at level of the decomposition, after PCA has been applied, or alternatively it is the input.and are 1D Gaussian derivatives in order and of order and respectively, where and .

Below is an example of one level of wavelet decomposition. The input is an image, which naturally has 3 channels, red, green, and blue. The number of output channels is 27, however many of these channels are redundant. Therefore, PCA is used, to reduce the number of channels, whilst retaining the variance. The full decomposition step applies the filter bank and then projects the output into the PCA subspace

To project the decompositions into the PCA subspace is simple once the principal components are calculated. The projection is accomplished as a weighted sum of the channels in the decomposition.

Figure 1: Wavelet decomposition of an image of a bird



is the PCA decomposition of channel of the tensor. is the number of channels before projection into PCA subspace. is the component of the principal component at level in the decomposition. is the bias term, which is calculated as the dot product of the negative of the mean and the principal component.

##### Training

A point of note is that PCWN does not use stochastic methods for learning such as backpropagation. This means the outcome of a PCWN is deterministic which has the advantage of repeatable results.

The PCWN trains sequentially, layer by layer. The training process loops over every image and feeds it through the filter bank. This creates a feature map, with many redundant channels. Every 2D position in this feature map has a vector called a fiber. The fibers are used to calculate the covariance matrix and the mean of the data. Using the covariance matrix and mean the principal components are calculated

Graphical user interface, application, Word

Description automatically generated

Figure 2: PCA training process [1]

##### Reconstruction Step

##### PCWN and Linear Least Squares

PCWNs have worked excellently for linear inverse tasks in computer vision. For segmentation the PCWN architecture requires some edits. The method used for adapting PCWN segmentation is incredibly versatile and can easily be adapted to a number of tasks, such as crowd counting and point detection.

The method involves training 2 separate PCWN networks. One network trains on the input, in this case an image. The other network trains on the images with the segmented masks applied. Both networks are only able to reconstruct the inputs, or corrupted versions of them. To extend PCWN to the task of segmentation, there needs to be a method for mapping the decomposition of the image network to the decomposition of the segmentation network.

The mapping can be thought of as a linear equation, where y is the segmentation decomposition, x is the image decomposition, and matrix A and vector b are variables that once solved will map x to y.

Using linear least squares, it is possible to find a solution to this equation quite efficiently and simply. The aim is minimising the expression below:

To do this requires finding the derivatives of each element with respect to A and b, the equations to do this end up looking like the two below:

The vector b is eliminated through substitution:

The substitution results in the following equation:

For the readers sake, the equation can be simplified by introducing X and Y as substitutions for and respectively, giving the following equation.

To find A, both sides of the equation are multiplied by the inverse of X. In practice, the Moore-Penrose pseudo-inverse [14, 15] is used, as the system of linear equations being solved may not have a unique solution.

X =

Which simplifies to:

=

It is then trivial to find vector b by substituting A back into the original equation and rearranging:

The method for calculating A and b can be accomplished in 2 ways, a fully connected method, or a fully convolutional method. The method of using linear least squares is the same, but the shape of the tensors is different.

##### Fully Connected Method

Diagram

Description automatically generated

Figure 3: Fully connected PCWN architecture

In the fully connected method, the decompositions are flattened. This approach is location variant, and so is suited to images where the object of interest is in a consistent position, such as the centre of the image. When calculated using the fully connected method, the X and Y matrix can become very large. This can cause memory issues, especially when calculating the Moore-Penrose pseudoinverse. The actual size of A is the product of the product of the dimensions of each decomposition.

The method for solving the linear inverse for the fully connected PCWN is exactly as stated in the previous section. However, when using A and b to transform an image decomposition into a segmentation decomposition, the image decomposition must be flattened, and the output is then reshaped, where the shape is that of a valid segmentation decomposition.

The python code for this operation can be found in the appendix.

##### A screenshot of a computer Description automatically generated with medium confidenceFully convolutional method

Figure 4: Fully convolutional PCWN architecture

The fully convolutional method is slightly more complex. This method is appropriate for images where the object of interest does not have a fixed position. The size of the matrix A is much smaller for a convolutional network approach, reducing computational complexity from calculations such as the pseudo-inverse. This means higher resolution images may be used, and deeper networks.

Solving the linear inverse for fully convolutional PCWN is slightly different to the fully connected method, as the convolutional approach uses matrices rather than vectors.

For the most part, the two approaches are very similar. When calculating the covariance between the decompositions, the matrix multiplication between a transposed decomposition and another decomposition is replaced with an outer product multiplication between decompositions. This is effectively the same operation but for a matrix.

The other difference is how the A matrix and b vector are applied when solving y=Ax+b. This method calculates a 1x1 convolutional filter, so A is reshaped as 1x1 filter and then applied to the image decomposition. The bias is added to the result. Again, this is effectively the same process as before but adapted for a matrix.

The python code for this operation can be found in the appendix.

#### U-Net Architecture

The U-net architecture was conceived in 2015 and is so called because of its U-shaped architecture. The success of U-map comes from the use of skip connections. These are connections in the network that allow layer outputs to bypass layers of the network. The result is then concatenated or reintroduced later in the network. Skip connections, or residual connections, first appeared in highway networks [16]. Highway networks implemented skip connections with gates, similar to a long-short term memory (LSTM) [17] mechanism. Perhaps the most famous architecture to use skip-connections is ResNet [18], an image recognition network that is still used as a benchmark today.

Modern methods for image segmentation use multi-scale feature maps, which means the feature maps encodes information from small resolution features, such as a paw in an image of a cat, as well as large scale features, such as the background of the image. U-net is not an exception. Skip connections allow U-net to use multi-scale feature maps when calculating the segmentation mask.

In a standard fully convolutional network there is a contracting path, which reduces the resolution of the feature map with every successive convolution and pooling operation. This means that feature information is lost, especially information regarding location of the object being segmented. Skip connections means that the network can retain information that would otherwise be lost. This is illustrated in the network architecture below.

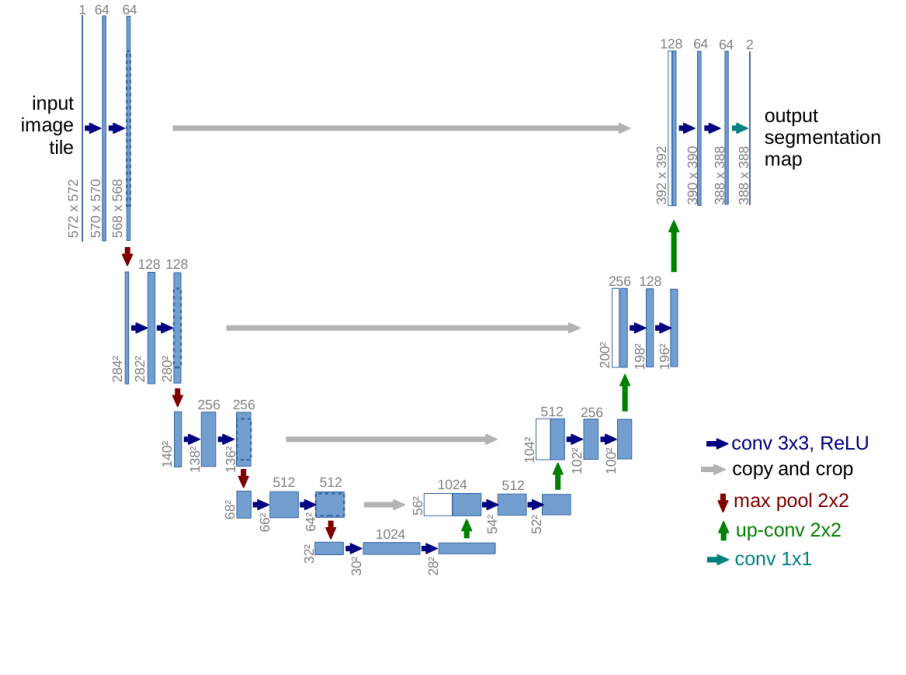
For image segmentation, the location of an image is important in constructing the mask. By using skip connections, it is possible to reuse high-resolution features to supplement up-sampled outputs from deeper layers, resulting in a more precise output. In the below example the blue box shows the location information provided by the high resolution feature to the low resolution feature in the yellow box.

Figure 5: U-net architecture [3]

#### Graphical user interface, text, application, Word Description automatically generatedDeeplabV3 Architecture

Figure 6: Example of high-resolution features being used to localize low-resolution features [3]

The DeepLabV3 is state of the art for semantic segmentation. The approach proposes the use of atrous convolutions, used in a method called Atrous Spatial Pyramid Pooling (ASPP) [19]. ASPP resamples a feature map with multiple atrous filters of increasing dilation and combines the resultant map using a pooling operation. Atrous convolutions are convolutional filters with a dilation rate which determines the padding between values in the filter. Atrous convolutions are excellent for multi-scale features, capturing dense features whilst maintaining the resolution of resultant feature maps. As mentioned, when discussing U-net, deep convolutional networks that use pooling and striding do not maintain resolution of feature maps, meaning information is lost.

The formula for calculating a 1-dimensional atrous convolution is:

Where y is the output feature map, x is the image, i is a location in x and y, r is the atrous rate and w is the kernel. By increasing the atrous rate, we increase the number of zeros between consecutive values in the filter. For example, an atrous value of 1 is a conventional filter. An atrous rate of 2 pads one zero between every value in the filter.





Figure 7: Atrous convolution: Atrous rate = 1

Figure 8: Atrous convolution: Atrous rate = 2

The reason for including details on DeeplabV3 and atrous convolutions is that the ASPP method could be applied to future PCWN work. Atrous convolutions have been used with undecimated wavelet networks before successfully [20] and would be an interesting edit to the PCWN architecture, however ASPP was beyond the scope of this project.

### Research Questions

There were several initial questions posed:

* Can you reimplement the network?
* Can you apply the network to a novel task?
  + Image segmentation
  + Crowd counting
* Can another dimensionality reduction be used instead of PCA?
* What are the effects of changing network parameters?

Of these questions, I have completed 3, and would suggest the remainder for a future project.

#### Can you reimplement the network?

This task is necessary to complete for the success of the rest of the project. Simply, the task is to reimplement the network for a new dataset.

#### Can you apply the network to a novel task? (Image Segmentation)

This task was the main focus of the project. There are a number of technical aspects to answering this question which were expected to take a long time, which is why the remaining questions have been delegated to future research.

#### What are the effects of changing network parameters?

This task is a natural extension of the segmentation task. Some experimentation would be required to determine the best parameters for segmentation. There are several parameters to vary through experiments. For the model, the parameters to edit include layer count, percentage of channels kept, different architectures and different activation functions. For the dataset, the editable parameters are image resolution and training data size.

## Process

The project was an individual effort which meant an exact agile methodology was not an appropriate choice for the task. Waterfall planning was not appropriate either, as the requirements were not clear at the start of the project. The most sensible approach was to select aspects of agile to apply to account for the unknown requirements of the project and the consequent changes. I used Kanban to track and plan the project, through the use of Kanban boards.

A Kanban approach is most effective in a team. As the project was carried out by an individual there required some adaption to the techniques. The changes were mostly simplifications of the method, such as removing work in progress (WIP) limits. Lots of features are only beneficial when working as a team, removing them streamlined development as less time was spent on teamworking features.

Week-long iterations emerged naturally, as there were weekly meetings with the project supervisor. During these meetings there were discussions about current progress, and the next set of features to implement. This customer driven approach, with the project supervisor as the customer, is consistent with feature driven development in the agile methodology.

# Experiment Methods

This section should discuss the overall hypothesis being tested and justify the approach selected in the context of the research area. Describe the experiment design that has been selected and how measurements and comparisons of results are to be made.

You should concentrate on the more important aspects of the method. Present an overview before going into detail. As well as describing the methods adopted, discuss other approaches that were considered. You might also discuss areas that you had to revise after some investigation.

You should also identify any support tools that you used. You should discuss your choice of implementation tools or simulation tools. For any code that you have written, you can talk about languages and related tools. For any simulation and analysis tools, identify the tools and how they are used on the project.

For the parts of your project that need some engineering (hardware, software, firmware, or a mixture) to support the experiments, include details in your report about your design and implementation. You should discuss with your supervisor whether it is better to include a different top-level section to describe any engineering work. In this template, Chapter 3 is suggested as a place for that discussion.

# Software Design, Implementation and Testing

## Design

### Overall Architecture

The purpose of the software was to generate benchmark image segmentation models, and objectively compare the benchmark models to PCWN models.

The requirements of the software are as follows:

1. Implementation of PCWN architecture for segmentation
2. Implementation of a deep-learning method for segmentation
3. Experiments to find best parameters for PCWN and deep learning model
4. Experiments to compare deep learning method and PCWN
5. Dataset loading and pre-processing

### Explanation of features

#### Even more detail

### User Interface

### Other relevant sections

## Implementation

This section should discuss issues you encountered as you tried to implement your experiments. What were the results of running the experiments? What conclusions can you draw from these results?

During the work, you might have found that elements of your experiments were unnecessary or overly complex; perhaps third-party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

If you had multiple experiments to run, it may be sensible to discuss each experiment in separate sections.

## Testing

Detailed descriptions of every test case are definitely not what is required in this section; the place for detailed lists of tests cases is in an appendix. In this section, it is more important to show that you adopted a sensible strategy that was, in principle, capable of testing the system adequately even if you did not have the time to test the system fully.

Provide information in the body of your report and the appendix to explain the testing that has been performed. How does this testing address the requirements and design for the project?

How comprehensive is the testing within the constraints of the project? Are you testing the normal working behaviour? Are you testing the exceptional behaviour, e.g. error conditions? Are you testing security issues if they are relevant for your project?

Have you tested your system on “real users”? For example, if your system is supposed to solve a problem for a business, then it would be appropriate to present your approach to involve the users in the testing process and to record the results that you obtained. Depending on the level of detail, it is likely that you would put any detailed results in an appendix.

Whilst testing with “real users” can be useful, don't see it as a way to shortcut detailed testing of your own. Think about issues discussed in the lectures about until testing, integration testing, etc. User testing without sensible testing of your own is not a useful activity.

The following sections indicate some areas you might include. Other sections may be more appropriate to your project.

### Overall Approach to Testing

### Automated Testing

#### Unit Tests

#### User Interface Testing

#### Stress Testing

#### Other Types of Testing

### Integration Testing

### User Testing

# Results and Conclusions

This section should discuss issues you encountered as you tried to implement your experiments. What were the results of running the experiments? What conclusions can you draw from these results? What graphs or other information have you assessed regarding your experiments? Discuss those.

During the work, you might have found that elements of your experiments were unnecessary or overly complex; perhaps third-party libraries were available that simplified some of the functions that you intended to implement. If things were easier in some areas, then how did you adapt your project to take account of your findings?

It is more likely that things were more complex than you first thought. In particular, were there any problems or difficulties that you found during implementation that you had to address? Did such problems simply delay you or were they more significant?

If you had multiple experiments to run, it may be sensible to discuss each experiment in separate sections.

## Issues with experiments

### Memory errors

By far the most prevalent error was out of memory errors. The cause of this ranged from inefficient programming, insufficient RAM, and poor memory management. To solve the inefficient programming and memory management issues we removed unnecessary objects. The insufficient RAM issue was solved initially by using cloud-based platforms, and then by purchasing a new laptop, upgrading the RAM from 8GB to 16GB.

Memory errors meant initial experiments for the fully connected network were throttled at a 128x128 image resolution. This in turn limited the “count” parameter of the PCWN. This parameter controlled the number of layers in the head, and the inverse head. Every layer reduced the resolution of the composition by half. A 128x128 image would have a feature map with a 4x4 resolution after passing through the head component of a network with 5 layers. This caused an error with the convolutional filters used in reconstruction, as the resolution is too small.

The fully convolutional network had notably fewer memory problems compared to the fully connected version. The reason for this was the calculation of the A matrix, which uses computationally intense methods, such as calculating the pseudo-inverse. The A matrix for the fully connected method was larger than the fully convolutional matrix, meaning the memory usage was much higher for the fully connected variant.

## Experimental Results: Determining best parameters for segmentation

### Convolutional Network approach

Table 1 Results of convolutional PCWN with different parameters

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Exp** | **training data size** | **count** | **keep %** | **resolution** | **dice mean test** | **dice mean train** | **dice std test** | **dice std train** | **seg train time** | **linear inverse train time** |
| **1** | None | 4 | 0.8 | 128 | 0.339 | 0.326 | 0.245 | 0.253 | 1072.086 | 1702.609 |
| **2** | 4000 | 3 | 0.3 | 128 | 0.358 | 0.348 | 0.218 | 0.222 | 168.371 | 242.668 |
| **3** | 2000 | 3 | 0.3 | 128 | 0.360 | 0.353 | 0.218 | 0.220 | 83.499 | 124.600 |
| **4** | 6000 | 3 | 0.2 | 64 | 0.564 | 0.557 | 0.203 | 0.205 | 251.632 | 297.255 |
| **5** | 6000 | 3 | 0.3 | 256 | 0.564 | 0.557 | 0.203 | 0.205 | 256.319 | 355.888 |
| **5** | 6000 | 3 | 0.1 | 128 | 0.565 | 0.558 | 0.204 | 0.206 | 226.381 | 327.403 |
| **6** | 6000 | 3 | 0.1 | 128 | 0.565 | 0.558 | 0.204 | 0.206 | 260.870 | 327.596 |
| **7** | None | 3 | 0.3 | 128 | 0.591 | 0.584 | 0.196 | 0.194 | 328.039 | 433.187 |
| **9** | None | 4 | 0.3 | 128 | 0.602 | 0.595 | 0.182 | 0.183 | 441.355 | 783.941 |
| **10** | None | 4 | 0.1 | 128 | 0.602 | 0.595 | 0.182 | 0.183 | 391.624 | 726.183 |

This experiment tests the strength of the convolutional network approach to PCWN. The results are sorted by the mean dice score. We can make a number of deductions from the results about what factors improve the performance of the model.

Training data size gives a significant improvement in the score (“None” means the full dataset was used, with 100 samples reserved for testing). In experiment number 2 and 3 there appears to be a small decrease in performance after increasing the training data size from 2000 to 4000. However, the result is not statistically significant.

Keep percent has a surprisingly negative effect on the dice score. Comparing result 1 to result 10 there is a very clear increase in dice score, 0.339 to 0.602, corresponding to decreasing the keep percent from 0.8 to 0.1. Furthermore, the training time decreases.

Count also has an effect, however it is not statistically significant unless

A cat sitting on a window sill

Description automatically generated with medium confidence

## Experimental Results: Convolutional vs Connected PCWN

## Experimental Results: Connected PCWN vs U-net

# Critical Evaluation

Examiners expect to find a section addressing questions such as:

* Were the requirements correctly identified?
* Were the design decisions correct?
* Could a more suitable set of tools have been chosen?
* How well did the software meet the needs of those who were expecting to use it?
* How well were any other project aims achieved?
* If you were starting again, what would you do differently?

Other questions can be addressed as appropriate for a project.

The questions are an indication of issues you should consider. They are not intended as a specification of a list of sections.

The evaluation is regarded as an important part of the project report; it should demonstrate that you are capable not only of carrying out a piece of work but also of thinking critically about how you did it and how you might have done it better. This is seen as an important part of an honours degree.

There will be good things in the work and aspects of the work that could be improved. As you write this section, identify and discuss the parts of the work that went well and also consider ways in which the work could be improved.

In the latter stages of the module, we will discuss the evaluation. That will probably be around week 9, although that differs each year.

# References

|  |  |
| --- | --- |
| [1] | M. G. Bernard Tiddeman, "Principal Component Wavelet Networks for Solving Linear Inverse Problems," *Symmetry,* vol. 13, no. 6, p. 1083, 2021. |
| [2] | S. M. Joan Bruna, "Invariant Scattering Convolution Networks," *IEEE transactions on pattern analysis and machine intelligence,* vol. 35, no. 8, pp. 1872-1886, 2013. |
| [3] | O. P. F. T. B. Ronneberger, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, Cham, 2015. |
| [4] | Y. H. Ş. a. G. U. S. C. Yurtkulu, "Semantic Segmentation with Extended DeepLabv3 Architecture," in *2019 27th Signal Processing and Communications Applications Conference (SIU)*, 2019. |
| [5] | S. e. a. Ghosh, ""Understanding deep learning techniques for image segmentation."," *ACM Computing Surveys (CSUR) ,* vol. 52, no. 4 , pp. 1-35, 2019. |
| [6] | R. E. Bellman, Dynamic Programming, Princeton University Press, 1957. |
| [7] | G. E. H. &. R. J. W. David E. Rumelhart, "Learning represntations by back-propagating errors," *Nature,* vol. 323, no. 9, pp. 523-536, 1986. |
| [8] | K. Fukushima, "Neocognitron: A Self-organizing Neural Network Model," *Biological ,* vol. 36, pp. 193-202, 1980. |
| [9] | B. B. J. S. D. D. H. R. E. H. W. H. L. D. J. Y. LeCun, "Backpropagation Applied to Handwritten Zip Code Recognition," AT&T Bell Laboratories, Holmdel, 1989. |
| [10] | A. Haar, "Zur Theorie der orthogonalen Funktionensysteme," *Mathematische Annalen,* vol. 69, no. 3, pp. 331-371, 1910. |
| [11] | M. J. Paul Viola, "Rapid Object Detection using a Boosted Cascade of Simple," in *Computer Vision and Pattern Recognition*, 2001. |
| [12] | E. Oyallon, S. Zagoruyko, G. Huang, N. Komodakis, S. Lacoste-Julien, M. Blaschko and E. Belilovsky, "Scattering networks for hybrid representation learning," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 41, pp. 2208-2221, 2018. |
| [13] | T. Angles and S. Mallat, "Generative networks as inverse problems with scattering transforms," in *ICLR*, 2018. |
| [14] | R. Penrose, "A generalized inverse for matrices," *Mathematical Proceedings of the Cambridge Philosophical Society,* vol. 51, pp. 406-413, 1955. |
| [15] | E. Moore, "On the reciprocal of the general algebraic matrix," *Bulletin of the American Mathematical Society,* vol. 26, pp. 394-395, 1920. |
| [16] | K. G. S. Rupesh Kumar Srivastava, "Highway Networks," 2015. |
| [17] | S. H. a. J. Schmidhuber, "Long short-term memory," *Neural computation,* vol. 9, no. 8, pp. 1735-1780, 1997. |
| [18] | X. Z. S. R. a. J. S. Kaiming He, "Deep Residual Learning for Image Recognition," 2015. |
| [19] | G. P. I. K. K. M. a. A. L. L.-C. Chen, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," 2016. |
| [20] | R. K.-M. J. M. a. M. Holschneider, "A real-time algorithm for signal analysis with the help of the wavelet transform," *Wavelets: Time-Frequency Methods and Phase Space,* pp. 289-297, 1989. |

# Appendices

The appendices are for additional content that is useful to support the discussion in the report. It is material that is not necessarily needed in the body of the report, but its inclusion in the appendices makes it easy to access.

If you have used any 3rd party code, i.e. code that you have not written yourself such as libraries, then you must include Appendix A. In that appendix, you will provide details of the 3rd party code that you have used.

For most other items, it would be better to include them in your technical submission instead of including them as an appendix. For example:

* If you have developed a Design Specification document as part of a plan-driven approach for the project, then it would be appropriate to include that document in the technical work. In this report, you would highlight the most interesting aspects of the design, referring your reader to the full specification for further detail.
* If you have taken an agile approach to developing the project, then you may be less likely to have developed a full requirements specification at the start of the project. Perhaps you used stories to keep track of the functionality and the ‘future conversations.’ If it isn’t relevant to include all those stories in the body of your report, you could detail those stores in a document in the technical work.
* If you have used manual testing, then include a document in the technical work that records the tests that have been done. In this report, you would talk about the use of those tests.

Documents included in the technical work or in the appendices are supporting evidence of the work done. Where you include documents, this report should refer to the documents. You should not be relying on detailed study of those documents in order to understand what is written in this report.

Speak to your supervisor or the module coordinator if you have questions about this.

* 1. Third-Party Code and Libraries

keras

tensorflow

tensorflow\_dataset

numpy

pandas

matplotlib

os

sys

tqdm

seaborn

MajorProject/Code/BaseCode directory

pca\_wavelet\_utils

* 1. Code Samples

Least Squares: Fully Connected

Calculating matrix A and vector b for least squares. This method is used for the fully connected version, where the decompositions are flattened.

Graphical user interface, application, Word

Description automatically generated

Least Squares: Fully Convolutional

Calculating matrix A and vector b for least squares. This method is used for the convolutional variant of PCWN, and reshapes the decompositions into 2D matrices.

Graphical user interface, text, application

Description automatically generated