Analysis of Unsupervised Feature Learning in Image Segmentation

Mennatullah Siam, Min Tang, Sepehr Valipour, Sepideh Hosseinzadeh, Xuebin Qin, Zichen Zhang
Co-coach: Dana Cobzas
University of Alberta
116 St 85 Ave, Edmonton
valipour@ualberta.ca

Abstract

Unsupervised feature learning was proved to be a potentially powerful tool for image segmentation as pixel-wise classification. However, there is no comprehensive study on the importance of each module of image segmentation pipeline. In this project we aim to understand the formulated variability of performance of feature learning methods in the context of image segmentation. A generic test framework was developed, then two segmentation tasks from two different domain were studied and analyzed. Through extensive experiments on buildings segmentation and multiple sclerosis lesions segmentation, different parameters are compared. Discussions about the preprocessing settings, the impact of dictionary learning, encoding and classification is presented. Our results conform in some parts with the analysis previously reported on image classification, but also new conclusions are drawn specific to the segmentation task.

1 Introduction

One of the main challenges in machine learning is the lack of labeled data for a particular task. Labeling data is usually expansive, both in money and time. One approach for using machine learning algorithms in such a situation is to utilize the abundance of cheap unlabeled data for the learning task. A variety of methods have been proposed recently that use unlabeled data as a source of information [12][16][2]. A successful methods is feature learning in which a dictionary is trained as an overcomplete basis for the data.

Many efforts have been made on parameter tuning, researches suggest to identify most prominent ones, then limit the search scope. Nowak et al. [19], investigates effects of feature representation algorithm and values of involved parameters. They conclude that highly overcomplete codebook/dictionary tends to be more successful, randomly sampled patches are superior to point of interest based sampler and choice of dictionary learning algorithm has negligible effect on outcome. Jarrett et al. [14], studies the encoding part and also the effect of a multilayer architecture compared to a single layer. They found encoding is the most influential part of the algorithm and having a second layer benefits the method. Boureau et al. [3], particularly look into encoding and pooling steps in image classification and tries to find best combination of methods for these two tasks. Coates et al. [6] goes a step further and explores if the search space contains global maxima or not. He states that, the better results can be achieved on the boundaries of some parameters. All these findings are mostly consistent and point to the same direction.

In contrast to these comprehensive researches about unsupervised feature learning for image classification, there is no peer study for pixel-wise image segmentation. Even though most of the procedures for pixel-wise image segmentation and image classification are the same, a few differences can drastically change the outcome which is sensitivity to some parameters. For example, the pooling step in image classification can provide an opportunity for smoothing the features [3] but it is absent in segmentation task. Besides, the proposed methods for improvement, such as increasing the dictionary size, is much more expansive for segmentation and not always feasible. Furthermore, increasing popularity of unsupervised methods for segmentation, asks for a deeper understanding of each module.

Compared with traditional methods for image segmentation, unsupervised feature learning poses advantage. Most of existing methods use, handcrafted features combined with an energy function or a classifier for segmentation. Even though these methods achieve impressive results in many applications, they are not widely applicable. Because, for each new task, appropriate filters should be carefully chosen and finely tuned to get decent results such as works in [8]. But as recent development in deep learning has shown a generic learning algorithm could be used for a wide range of applications with extremely good performance. Kiros et al. [15], achieved state of the art performance on vessel segmentation in brain MRI images using sparse encoders and Deshpande et al. [7] achieves best results in MS lesion segmentation. Rigamonti et al. [20] work on filter learning using sparse methods for road segmentation in aerial images.

Accordingly, in this study, we seek answers for questions about unsupervised feature learning in image segmentation. The same questions have been asked in image classification. Specifically, we investigate four major components of the learning system, pre-processing, dictionary learning, encoding and classifier. Some parameters in each component have been thoroughly studied and empirical result is gathered for comparison and conclusion. To make the finding independent from the subject, two tasks from very different domains were chosen. One is building segmentation in high resolution satellite images and the other is multiple sclerosis lesion segmentation. Even though they are different in nature they share common characteristics. Collecting labeled data for both tasks is quite expensive while unlabeled data could be found freely in abundance. Besides, intensive effort to solve the task with hand crafted features produced mediocre results. All these make both tasks suitable for unsupervised methods.

The proposed image segmentation pipeline is shown in Fig. 1. The rest of the paper is organized as follows: first we give a review on each module of the pipeline, then we show the datasets. Afterwards, the experiments and result analysis are presented and finally we have the conclusion and future work.

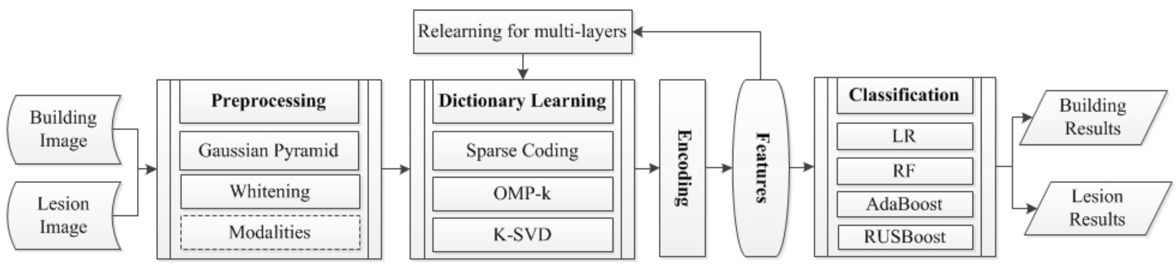

Figure 1: Framework for analysis of feature learning in image segmentation

2 Methodology

2.1 Preprocessing

Gaussian Pyramid: In order to have a compact and efficient multi-scale representation for an image, we apply Gaussian Pyramid to the image. A Gaussian pyramid contains the original image and subsequent images in different scales that are built by repeatedly smoothing and subsampling the original. The smoothing is done by a Gaussian blur (average) and scaled down. Each pixel contains a local average of neighborhood pixels on a lower level of the pyramid. [15]

Whitening: Whitening makes the raw input images less redundant. The goal is to make the features less correlated with each other and have the same variance. [5] We performed whitening on each patch instead of the entire image. Interestingly, whitening did help with buildings but did not make a difference for lesions. A possible reason is that the correlation between the pixels on MRI images are low even without applying whitening.

2.2 Unsupervised Feature Learning

For given data $X \in \mathbb{R}^{m\times n}$ (m is the number of instance and n is the attribute dimension of instance), the main idea of unsupervised feature learning is to find a basis(or dictionary) $D \in \mathbb{R}^{n\times d}$ which can be used to represent this X more efficiently by codes S based on D, $S \in \mathbb{R}^{m\times d}$. $\boldsymbol{x}^{(i)} \in \mathbb{R}^n$ is an instance of X and $\boldsymbol{s}^{(i)}$ is its corresponding codes. $D^{(j)}$ represents a column element of D.

2.2.1 Dictionary Learning

In this work, three kinds of unsupervised dictionary learning algorithms, sparse coding, OMP-k and K-SVD, were used.

(1) Sparse Coding (SC): The cost function of sparse coding is shown in equation (1) [17].

$$\min_{D,\mathbf{s}} \sum_i \|D\boldsymbol{s}^{(i)} - \boldsymbol{x}^{(i)}\|_2^2 + \lambda \sum_i \phi(\boldsymbol{s}^{(i)}) \quad \text{subject to } \|D^{(j)}\|_2^2 = 1\, \forall j \tag{1}$$

$$\phi(\mathbf{s}^{(j)}) = \begin{cases} \|\mathbf{s}^{(j)}\|_1 & \text{L}_1 \text{ penalty function} \\ ((\mathbf{s}^{(j)})^2 + \epsilon)^{\frac{1}{2}} & \text{epsilon L}_1 \text{ penalty function} \\ \log(1 + (\mathbf{s}^{(j)})^2) & \text{log penalty function.} \end{cases} \tag{2}$$

D and $\boldsymbol{s}$ are computed alternatively by minimizing cost function[17]. Equation (2) is the penalty function and we mainly use epsilon L_1 penalty function in this work.

(2) Orthogonal Matching Pursuit (OMP-k): The cost function of OMP-k is the same with sparse coding except for the removing of penalty function and addition of constrain as is shown in equation (3) [4].

$$\min_{D,\boldsymbol{s}} \sum_i \|D\boldsymbol{s}^{(i)} - \boldsymbol{x}^{(i)}\|_2^2 \quad \text{subject to } \|D^{(j)}\|_2^2 = 1\, \forall j \quad \text{and} \quad \|\boldsymbol{s}^{(i)}\|_0 \leqslant k\, \forall i \tag{3}$$

$\boldsymbol{s}^{(i)}$ are approximately computed by Orthogonal Matching Pursuit and $\|\boldsymbol{s}^{(i)}\|_0$ is the number of non-zero elements in $\boldsymbol{s}^{(i)}$. $\|\boldsymbol{s}^{(i)}\|_0$ are not greater than k that's why this method is called OMP-k. In our study, OMP-1 was the most commonly used because it is easy to solve for optimal dictionary. It chooses $k = \text{argmax}_j |D^{(j)T}\boldsymbol{x}^{(i)}|$, then makes $\boldsymbol{s}_{j=k}^{(i)} = D^{(j)T}\boldsymbol{x}^{(i)}$ and $\boldsymbol{s}_{j\neq k}^{(i)} = D^{(j)T}\boldsymbol{x}^{(i)} = 0$.

(3) K-SVD: The cost function of K-SVD is almost the same with OMP-k. The difference exists in the process of computing the dictionary. The name K-SVD stems from the fact that K-SVD operations are used to update the dictionary. It updates one column at a time with each time computing SVD on the restricted error matrix [1].

$$\min_{D,\boldsymbol{s}} \sum_i \|D\boldsymbol{s}^{(i)} - \boldsymbol{x}^{(i)}\|_2^2 \quad \text{subject to } \|\boldsymbol{s}^{(i)}\|_0 \leqslant T_0\, \forall i \tag{4}$$

2.2.2 Encoding

After training the dictionary D, original input data $\boldsymbol{x}$ need to be coded by dictionary D and features $\boldsymbol{s}$. The encoding process is to find $\boldsymbol{s}$ when given D. In this work, sparse coding, OMP-k and thresholding were implemented and tested.

(1) Sparse Coding: Given trained dictionary D, encoding by sparse coding minimizes the same equation (1). But here D is fixed and $\boldsymbol{s}$ is computed iteratively. In general , parameters of encoding process should be the same as training process except for the coefficient (λ) of penalty function. λ can be changed appropriately. In this work, we use the same parameters saved in dictionary learning and encoding. When we use dictionary trained by OMP-k or k-SVD to encoding, parameters, such as patch size, were modified to match the dictionary. $\boldsymbol{s}$ here is the feature (we did not split the positive and negative components of $\boldsymbol{s}$ to form features [4]).

(2) Orthogonal Matching Pursuit (OMP-k): For endcoding by OMP-k, $D^{(j)T}\boldsymbol{x}$ was used to compute $\boldsymbol{s}$ when $k = 1$. When $k \neq 1$, $\boldsymbol{s}$ was computed by minimizing equation (3).

(3) Thresholding (T): Soft thresholding works on top of OMP-1 by applying the following non-linearity to compute features [4]. In our experiment, we set α to zero and name it Thresholding to distinguish from soft thresholding.

$$\begin{aligned} f_j &= \max\{0, D^{(j)T}\boldsymbol{x} - \alpha\} \\ f_{j+d} &= \max\{0, -D^{(j)T}\boldsymbol{x} - \alpha\} \end{aligned}$$

2.3 Multi modality

In this project, we considered both Gaussian pyramids and modalities of images. Given images with modality number of c, patches (size: $m \times (w \times h)$, m represents the number of patch. w and h denote the width and height of patch.) were randomly extracted from image pyramids in dictionary learning. Each patch contains information of corresponding area of c modalities. So the size of X is $m \times (w \times h \times c)$,where, m is the number of intances and $(w \times h \times c)$ denotes the attribute dimension of instance, and the size of dictionary D is $((w \times h) \times d) \times c$. In the process of encoding, features considered with modality and pyramid scale were extracted by (5) [15]. T_j^{γ} represents a volume slice of modality j and scale γ. $D_j^{(l)} \in \mathbb{R}^{w\times h}$ is the l-th basis for modality j of D. where * denotes convolution. Finally, feature maps of each octave of pyramid were upsampled to the same size of original image.

$$f_l^\gamma = \sum_{j=1}^{c} T_j^\gamma * D_j^{(l)} \tag{5}$$

2.4 Classification

2.4.1 Logistic Regression

Logistic Regression finds the linear model that fits the data in sense of minimizing the cost function below with respect to θ [9].

$$J(\theta) = -\sum_i (y^{(i)} log(h_\theta(x^{(i)})) + (1-y^{(i)}) log(1-h_\theta(x^{(i)})) \tag{6}$$

where $\boldsymbol{x}$ is the input data, $\boldsymbol{y}$ is labels and h_θ is the linear hypothesis for the model.

2.4.2 Random Forest

Random Forests [13] is an ensemble learning method for classification. Each tree is trained with a random subset of the training data, this process is called bagging. And the output of different trees are merged using majority voting. Also each tree is sampling from the original features and using them for the tree splitting. The main advantage of random forests, is in handling mislabeled data because trees are trained on subsets of the data. Another advantage is that it can easily be parallelizable. The parameters of the random forest can be tuned using out of bag error, which is the error rate for observations left out of the bootstrap sample for each tree.

2.4.3 AdaBoost/RUSBoost

AdaBoost is an algorithm for building a stronger classifier from a simple one, as linear combination of simple classifier [10]. A variation of Adaboost for handling imbalanced data is called RUSBoost. RUS stands for Random Under Sampling. It randomly undersamples the majority class and then run AdaBoost on the undersampled data [21].

3 Experimental Analysis

3.1 Datasets

3.1.1 MS Lesion

For MS lesion segmentation, we used the MICCAI Grand Challenge 2008 dataset. The data were acquired from Children's Hospital Boston (CHB) and University of North Carolina (UNC). The labeled data is composed of 20 MRI images that represents a range of patients and pathology. Each MRI has 3 modalities: a T1-weighted image, a T2-weighted image and a FLAIR image. All data has been rigidly registered [1].

3.1.2 Building

The Massachusetts Buildings Dataset [18] consists of 151 aerial images of the Boston area, with each of the images being 1500 by 1500 pixels for an area of 2.25 square kilometers. The entire dataset covers roughly 340 square kilometers. The data used for the experiment is randomly split into a training set of 137 images, a test set of 10 images and a validation set of 4 images. The target maps were obtained by rasterizing building footprints obtained from the OpenStreetMap project. Unlike the Greater Toronto Area (GTA) Buildings dataset, this data was restricted to regions with an average omission noise level of roughly 5% or less. The dataset covers mostly urban and suburban areas and buildings of different sizes. Individual houses and garages are also included in the labels [2].

[1] http://www.ia.unc.edu/MSseg/index.html
[2] https://www.cs.toronto.edu/ vmnih/data/

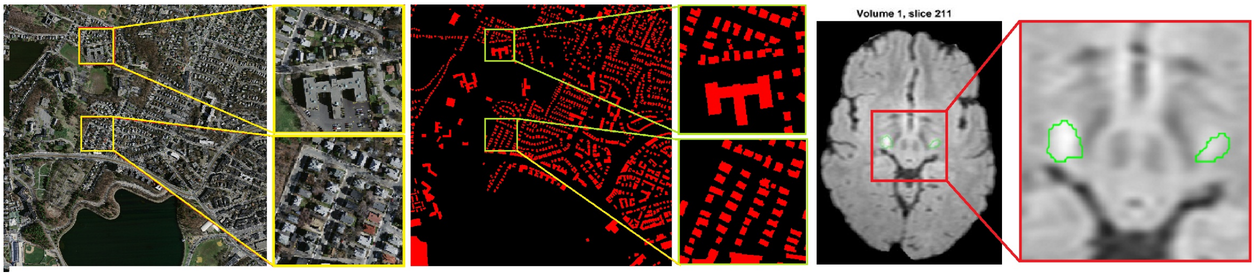

Figure 2: Sample images of the datasets. From left to right:
Aerial Image and the labels (red area denotes buildings), MRI and the labels (lesions are inside the green boundary)

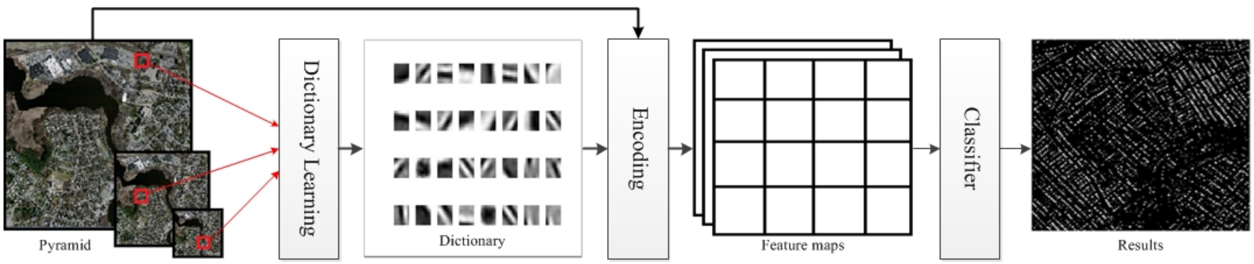

Figure 3: Visualization of pipeline and final output for building segmentation

3.2 Evaluation

Quantitative evaluation is carried out for the segmentation tasks (pixel-wise classification). With skewed data (0.1% positives in the case of MS lesions), the accuracy is 99.9% even if simply predicting all negatives. Thus we use Precision, Recall and F1-score as evaluation metrics which are summarized in Table 1. Due to time constraints and memory usage, the experiment is carried out with a subset of the training data, which is summarized in Table 2. Note that the experiments presented for building segmentation is trained with 4500000 samples that is from 2 training images, and testd on 2250000 samples from 1 image. For MS lesions segmentation, we trained on one volume and tested on another one. Specifically, we used the features learned from the 42 slices that contain lesions. The testing was done on all the 512 slices in the test volume.

Table 1: The evaluation metrics.
TN: true negatives. TP: true positives.
FN: false negatives. FP: false positives.

Name	Definition	Unit	Best	Worst
Precision	$\frac{TP}{TP+FP}$	%	100	0
Recall	$\frac{TP}{TP+FN}$	%	100	0
F1-score	$\frac{2\times Precision\times Recall}{Precision+Recall}$	%	100	0

Table 2: Training and Testing data

	Lesion	Building
Training	UNC_train_01	MassachusettsB train
Testing	UNC_train_10	MassachusettsB test

3.3 The state of the art

Ezequiel Geremia et al. built a discriminative random decision forest framework for MS lesion segmentation which provide a voxel-wise probabilistic classification of the volume [11]. The method used three kinds of 3D features based on multi-channel intensity, prior and context- rich information and got a performance of 39.8% in precision, 39.4% in recall and 39.6% in F1-score.

Volodymyr Mnih implemented deep neural networks on building segmentation which can efficiently learn highly discriminative image features [18]. The method introduced new loss functions for training neural networks that are partially robust to incomplete and poorly registered target maps. It proposed two ways of improving the predictions by introducing structure into the outputs of the neural networks and achieved a performance of 92% in precision, recall and F1-score.

3.4 Results and Discussion

In this section we show the experimental results obtained on both buildings segmentation and lesions segmentation, followed by a discussion of the results. The learning and performance task is illustrated in Fig. 3. For each component in the pipeline, we experimented with different parameters and discuss how they impact the segmentation result.

The experiments results are presented in two sections: one that is concerned with tuning the preprocessing parameters, and the other was focused on experimenting with different dictionary learning, encoding techniques and classifier. Table3 summarizes the components and parameters that were tuned in the experiments. All the results shown in this section, are obtained by tuning one particular parameter while keep the other parameters the same.

Table 3: Components and parameters used in the experiment
SC: Sparse Coding, RF: Random Forests, LR: Logistic Regression, T: Thresholding

Preprocessing		**Main experiments**	
Patch Size	5x5 / 9x9 / 15x15	**Dictionary Type**	K-SVD/ OMP-1/ SC
Modalities	Multi / Single	**Dictionary Size**	32 / 100 / $>$ 400
Gaussian Pyramid	1 / 3 / 6 scales	**Encoder**	$D^T x$ / SC / OMP-K/ T
		Classifier	LR / RF / Adaboost / RUSBoost
		Data	Balanced/Imbalanced.

3.4.1 Preprocessing Parameters

The first set of experiments is to decide on the preprocessing parameters to use during the main experiments.

We compared the result between using single and multi modalities. On one hand, multi modalities when fused together is expected to provide more information to the classifier. But on the other hand it will increase the computation and merging the features may blur out the distinctive features. The "modalities" is T1, T2 and FLAIR for the lesion data and RGB channels for the building data. The multi-modalities used in buildings yielded better results as shown in table 4. But for lesions segmentation it yielded worse result. Our interpretation is that the features of lesions are less distinctive on some modalities so taking the average of all modalities makes it worse.

Table 4: Modalities Results

Modalities	Single	Multiple
Lesions	FLAIR	T1,T2,FLAIR
Precision	**3.165**	0.0015
Recall	**40.29**	10.23
F1-Score	**5.869**	0.0031
Buildings	Grayscale	RGB
Precision	88.28	**82.92**
Recall	2.33	**6.69**
F1-Score	4.54	**12.38**

Table 5: Gaussian Pyramid Results

Settings	6 Scales	3 Scales	1 Scale
Lesions			
Precision	0.9387	**3.165**	0.1928
Recall	5.621	**40.29**	37.17
F1-Score	1.609	**5.869**	0.3837
Buildings			
Precision	87.24	**82.56**	66.82
Recall	10.52	**26.21**	21.61
F1-Score	18.78	**39.79**	32.66

Another parameter to study is the number of scales used in the Gaussian pyramid which is used to have scale independent features. Table 5 shows the results for using 1,3,6 scales on both image segmentation tasks. In the results it is noticeable that using 1 scale yielded worse recall since it won't capture varying sizes. When we use six scales it leads to bad result too since in buildings data some get merged together in the downscaled image and blurred in coarser resolution. So 3 scale pyramid gave the best results.

Finally, the patch size is another aspect to consider, since this will affect the dictionary learning and we had to make sure that the each patches would contain enough information for learning a representative dictionary. In table 6 it is shown that a moderate patch size yields best result. The patch needs to be bigger than the smallest object of interest (smallest lesion is around 5x5). But a big patch tends to introduce too much noise so it's a trade-off one has to make.

3.4.2 Main Parameters

The second set of experiments is the analysis of different dictionary learning and encoding techniques. Classification is studied as well.

Table 6: Patch Size Results

Patch Size	Small	Medium	Large
Lesions	3x3	5x5	9x9
Precision	1.385	**3.165**	2.516
Recall	29.95	**40.29**	23.82
F1-Score	2.647	**5.869**	4.552
Buildings	5x5	9x9	15x15
Precision	81.43	**87.24**	79.71
Recall	7.71	**10.52**	5.10
F1-Score	14.09	**18.78**	9.59

Table 7: Balanced Data Results

	Imbalanced	Balanced
Lesions		
Precision	9.412	**3.893**
Recall	0.1777	**15.27**
F1-Score	0.3489	**6.204**
Buildings		
Precision	82.56	**61.08**
Recall	26.21	**57.86**
F1-Score	39.79	**59.43**

(1) Classification method: Experiment in this part is about choosing the suitable classification method for the task at hand. The classifiers used are Logistic Regression, Random Forests and Adaboost/RUSboost. RUSboost is specifically used for the lesions data since it's highly skewed. The Random Forests is experimented with a varying cost matrix. Table 8 shows that Random Forests achieves the best F1-score. Note that the number of trees is set to 50, which was selected according to the out of bag error, and the number of features to sample is 50. There are two main reasons for the superior performance of RF. First, RF is a non linear classification method that is more appropriate to the segmentation tasks. The second reason is that random forest is able to handle mis-labeled data that is abundant in the aerial imagery datasets. A cost matrix is often used for practical applications where the misclassification cost is clearly different for different classes. The main benefit of using it in our case is to balance the precision and recall.

Table 8: Results using different classifiers
cost: misclassification cost of foreground class for building; of class background for lesion

Classifier	Logistic Regression	Boosting	Random Forest		
			cost= 1	cost= 5	cost= 10
Lesions		RUSBoost			
Precision	3.165	0.080	3.893	**5.662**	8.036
Recall	40.29	98.34	15.27	**7.39**	3.355
F1-Score	5.869	0.160	6.204	**6.410**	4.734
Buildings		Adaboost			
Precision	82.92	89.00	89.09	87.23	**82.56**
Recall	6.69	11.00	9.57	22.02	**26.21**
F1-Score	12.38	19.58	17.29	35.17	**39.79**

(2) Imbalanced data: Experiments in this part is about dealing with imbalanced data. Table7 shows the results obtained on buildings and lesions. Both show better F1-score with enforcing balanced data with 1:2 ratio (a common practice) between foreground and background respectively. Note that for buildings data, we used $cost = 10$ for the imbalanced data and $cost = 2$ for balanced one.

(3) Dictionary learning: Experiment in this part is varying dictionary learning algorithms and was done on balanced data. Three different learning techniques were utilized to compare the results on both segmentation tasks. The algorithms compared are K-SVD, OMP-1, and Sparse Coding. Table 9 shows the results. The results obtained in both applications demonstrate no significant difference between the different learning techniques. But since OMP-1 is more computationally efficient than others, it is recommended to use. This conforms with the conclusions presented in [6], that mentions that the dictionary learning part does not significantly affect the final classification results. Adding to that, we tested with swapped dictionary between the two applications: segmenting lesions using the dictionary learned from buildings and vice versa. Very interestingly, the experiment shows no significant difference than other learned dictionaries, again conforming with the small impact of the dictionary learned.

(4) Encoding: Also different encoding techniques are compared against each other. Four encoding techniques that are tested are $D^T x$, Sparse Coding, OMP-K, and Thresholding. Table 11 shows the results of this experiment. Among the techniques, $D^T x$ gave the best results. It was the most computationally efficient one as well. Finally for different dictionary size Table 10 shows different dictionary sizes that was used. A bigger dictionary is considered to be more powerful as it offers more basis. But we found that a larger dictionary does not necessarily give better result. For

Table 9: Dictionary Results

Learning	K-SVD	OMP-1	SC	Swapping
Lesions				
Precision	2.830	**3.165**	2.387	2.971
Recall	44.02	**40.29**	38.62	34.91
F1-Score	5.317	**5.869**	4.496	5.476
Buildings				
Precision	**62.35**	61.08	59.62	58.23
Recall	**61.18**	57.86	55.46	52.76
F1-Score	**61.76**	59.43	57.47	55.36

Table 10: Dictionary Size Results

Size	32	100	> 400
Lesions			
Precision	0.92	**3.165**	3.098
Recall	8.643	**40.29**	0.5669
F1-Score	1.663	**5.869**	5.5022
Buildings			
Precision	**61.08**	58.25	66
Recall	**57.86**	56.96	15
F1-Score	**59.43**	57.59	24.44

Table 11: Encoding Results, T: Thresholding, SC: Sparse Coding

Encoding	$D^T x$	T	SC	OMP-4
Lesions				
Precision	0.4658	**3.165**	1.138	0.8887
Recall	37.38	**40.29**	23.41	26.32
F1-Score	0.9202	**5.869**	2.171	1.719
Buildings				
Precision	**61.08**	56.01	82.51	81.05
Recall	**57.86**	56.48	15.34	15.10
F1-Score	**59.43**	56.25	25.85	25.46

buildings the 32 dictionary size is the best setting for dictionary learning. For lesions, 100 yields better result. Keep increasing it did not improve the performance. Although [4] mentioned that increasing the dictionary gets better results, but that work was based on the analysis of image classification. And this is explained by the fact that the classification task is using larger patch size and is used later to encode a region. But in our case we're handling pixel wise classification task, and smaller patch size. So with larger dictionary size the learned dictionary will tend to overfit the training data and thus will not be representative for other data.

The best final results obtained on buildings segmentation is F1-score of 61.7%; with 6.4% for MS lesion segmentation.

4 Conclusion and Future Work

In this research multiple components were studied and general guideline for image segmentation using feature learning were derived. The empirical results show that preprocessing is substantially important for the pipeline and the performance can be bounded with improper preprocessing. Specific conclusions are listed below.

- Patch size and pyramid should be selected based on the data. Patch size in each pyramid scale should not be smaller than the smallest region and also should not be much larger than the biggest object of interest.
- In dictionary learning, we see the dictionary learning method barely affect the performance. As even swapping the learned dictionaries on these two different domain does not affect the performance. This is similar to the findings in image classification literature. However, in contrast to image segmentation task, best performance is not limited by dictionary size, larger dictionary will cause overfitting.
- In encoding, we found extremely good results can be achieved by simple encoders. This conforms with analysis in image classification. Therefore, we recommend to first try with simplest encoders and focus the tuning on other components.
- In classification, we state that making the data balanced by under/oversampling or, equally, using classifiers that handle the skewed data has the most significant effect.

For the future work, we will study multiple layers to see if having more layers negates any of our conclusion or not. In case of these specific tasks, the main priority is to use the whole dataset not just a tiny portion of it.

References

[1] M. Aharon, M. Elad, and A. Bruckstein. k -svd: An algorithm for designing overcomplete dictionaries for sparse representation. *Signal Processing, IEEE Transactions on*, 54(11):4311–4322, Nov 2006.

[2] Yoshua Bengio, Pascal Lamblin, Dan Popovici, Hugo Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

[3] Y-Lan Boureau, Francis Bach, Yann LeCun, and Jean Ponce. Learning mid-level features for recognition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2559–2566. IEEE, 2010.

[4] Adam Coates and Andrew Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 921–928, 2011.

[5] Adam Coates and Andrew Y Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, pages 561–580. Springer, 2012.

[6] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International conference on artificial intelligence and statistics*, pages 215–223, 2011.

[7] Hrishikesh Deshpande, Pierre Maurel, and Christian Barillot. Detection of multiple sclerosis lesions using sparse representations and dictionary learning. In *2nd International Workshop on Sparsity Techniques in Medical Imaging (STMI), MICCAI 2014*, number 71-79, 2014.

[8] Alejandro F Frangi, Wiro J Niessen, Koen L Vincken, and Max A Viergever. Multiscale vessel enhancement filtering. In *Medical Image Computing and Computer-Assisted InterventationMICCAI98*, pages 130–137. Springer, 1998.

[9] David A Freedman. *Statistical models: theory and practice*. cambridge university press, 2009.

[10] Yoav Freund and Robert Schapire. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[11] Ezequiel Geremia, Olivier Clatz, Bjoern H Menze, Ender Konukoglu, Antonio Criminisi, and Nicholas Ayache. Spatial decision forests for ms lesion segmentation in multi-channel magnetic resonance images. *NeuroImage*, 57(2):378–390, 2011.

[12] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[13] Tin Kam Ho. Random decision forests. In *Document Analysis and Recognition, 1995., Proceedings of the Third International Conference on*, volume 1, pages 278–282. IEEE, 1995.

[14] Kevin Jarrett, Koray Kavukcuoglu, Marc'Aurelio Ranzato, and Yann LeCun. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.

[15] Ryan Kiros, Karteek Popuri, Dana Cobzas, and Martin Jagersand. Stacked multiscale feature learning for domain independent medical image segmentation. In *Machine Learning in Medical Imaging*, pages 25–32. Springer, 2014.

[16] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

[17] Honglak Lee, Alexis Battle, Rajat Raina, and Andrew Y. Ng. Efficient sparse coding algorithms. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 801–808, 2006.

[18] Volodymyr Mnih. *Machine Learning for Aerial Image Labeling*. PhD thesis, University of Toronto, 2013.

[19] Eric Nowak, Frédéric Jurie, and Bill Triggs. Sampling strategies for bag-of-features image classification. In *Computer Vision–ECCV 2006*, pages 490–503. Springer, 2006.

[20] Roberto Rigamonti, Engin Türetken, Germán González Serrano, Pascal Fua, and Vincent Lepetit. Filter learning for linear structure segmentation. Technical report, 2011.

[21] Chris Seiffert, Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Rusboost: Improving classification performance when training data is skewed. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4. IEEE, 2008.