COMP4801 Final Year Project

Object Recognition by Deep Learning Neural Networks

Final Report

Wang Shunqi (3035085736)
Supervised by Dr. KP Chan

April 4th, 2018
Abstract

Among the most discussed topics in the computer vision field, object recognition is one of the most widely-used and commercialized one in the modern world. In 2014, Girshirk, Donahue, Darrell and Malik first introduced R-CNN algorithm in their paper, which was a major breakthrough in the object recognition field. They were the first to apply deep learning neural networks technologies into the region of computer vision. Their attempt had already been proven to be successful, as R-CNN algorithm increased the best mean average precision (mAP) from approximately 20% to 53.3%, raising the threshold of the development of other solutions in this field. The performance of object recognition algorithms has been raised to a new and extremely high level on the basis of R-CNN. Recent algorithms improved from R-CNN, specifically a variation of the Faster R-CNN algorithm published in November 2016, had increased the mAP to 88.6%[1], making a big leap on the road of human beings to completely solve the problem of object recognition. This project is also inspired by the R-CNN algorithm. This report will discuss our implementation and improvement of the R-CNN algorithm. In the first part, we will introduce our reimplementation of the R-CNN algorithm using Python, and the second part addressed our replacement of the R-CNN classifier with Latent Dirichlet Allocation, an implementation of the topic
model.
Acknowledgment

We would like to express our thanks to our supervisor Dr. KP Chan for his instruction and insightful thoughts he provided. We also want to show our gratitude to him for providing us with a high-performance workstation and try his best to fulfill our software and hardware requirements.
# Table of Contents

Abstract .........................................................................................................................2
Acknowledgment ..........................................................................................................4
Table of Contents ..........................................................................................................5
List of Figures .................................................................................................................7
List of Tables .................................................................................................................8
Glossary and Abbreviations ............................................................................................9

1. **Introduction** ..............................................................................................................10
   a. Object recognition and R-CNN ................................................................................10
   b. Topic model and Latent Dirichlet Allocation .........................................................11

2. **Outline** ...................................................................................................................12

3. **Motivation** .............................................................................................................12

4. **Objectives** .............................................................................................................17

5. **Scope** ...................................................................................................................19

6. **Deliverables** .........................................................................................................20

7. **Related works** .......................................................................................................21

8. **Prerequisites** .......................................................................................................22

9. **Methodology** .......................................................................................................23
   a. Datasets preparation ..............................................................................................23
   b. Python version implementation ..........................................................................24
      i. Selective Search [12] .......................................................................................25
ii. Convolutional neural network ................................................................. 26

iii. Latent Dirichlet Allocation .............................................................. 27
c. Training .......................................................................................... 28

10. Progress .......................................................................................... 29
    a. Environment setup ......................................................................... 29
    b. Project webpage ............................................................................ 29
    c. R-CNN reimplementation ........................................................... 30
    d. Timeline ....................................................................................... 33

11. Results .............................................................................................. 34

12. Obstacles ........................................................................................... 35

13. Recommendations and Future works .............................................. 37
    a. Reduce cache image size ............................................................. 37
    b. Apply supervised LDA classifier .................................................. 38
    c. Simple UI to assist presentation .................................................. 38

14. Conclusion ......................................................................................... 39

References .............................................................................................. 41
List of Figures

Figure 1 – Structural similarity between documents and images ..........17
Figure 2 – General flow of R-CNN [4] ..............................................24
Figure 3 – Initial, intermediate and final status of the segmentation and the
selection procedure of the algorithm [12].........................................25
Figure 5 – Screenshot of the project website.......................................30
Figure 6 – Reimplementation progress.............................................31
Figure 7 – Image input method and the results generated by selective search
........................................................................................................32
Figure 8 – Sample result after conversion...........................................32
Figure 4 – Different ways to resize a segmentation. A is the original
segmentation.....................................................................................36
Figure 9 – Sample result.....................................................................39
List of Tables

Table 1 – Sample distribution of words..................................................16

Table 2 – Comparisons between Zhiling Pan’s team model and other models [9].................................................................................................21

Table 3 - Detail timeline ...........................................................................33
Glossary and Abbreviations

CNN - Convolutional neural network

Caffe [1] – A open-source deep learning framework

LDA - Latent Dirichlet allocation

R-CNN - Regions with Convolutional Neural Network Features

SVM - Support vector machine, a linear supervised learning model

Topic model – a type of statistical model. Mainly used in machine learning and natural language processing to find the abstract topics among documents.

mAP – Mean average precision, the assessment unit for object recognition
1. Introduction

This section introduces some basic background of this research project. This section first introduces what is object recognition and some related algorithms. Then it shows the significance and importance of one of the algorithms, R-CNN. It also briefly presents some basic information of topic model and Latent Dirichlet Allocation.

a. Object recognition and R-CNN

Object recognition is a combination of computer vision and image processing technology. It is widely used in various regions. In security aspects, object recognition technologies are applied on face detecting and people counting. The manufacturing industry uses it to check products to improve efficiency and decrease defect rate. Every Other high-tech products, such as autonomous vehicles and augmented reality devices cannot run properly without an efficient and high-accuracy object recognition algorithm.

However, in 2014, the traditional object recognition algorithms based on SIFT [2] and HOG [3] had been facing their bottlenecks on improving their performances [4] for years. The introduction of deep learning and convolutional neural networks into this region by researchers explored a new direction of the study and broke the
predicament. R-CNN was raised and received great success under this circumstance. Traditional object recognition algorithms can achieve a mean average precision (mAP) around 25% and R-CNN successfully improved this measurement by more than 30% [4].

b. Topic model and Latent Dirichlet Allocation

Topic model is a machine learning model originally used in natural language processing. It is a statistical model to identify abstract topics and calculate their proportion respectively in a given document. The currently existing topic models vary in the statistical distribution they use. One of the most popular topic models now is Latent Dirichlet Allocation (LDA) [5], which was developed by David Blei, Andrew Ng, and Michael I. Jordan in 2002. This model uses Dirichlet distribution. The topic model was named after this distribution.

Topic model technology is broadly used in document classification and reading recommendation. The application of topic model technology in reading recommendation has two stages. First of all, topic modeling is applied to a reader’s reading history, and a list of topics that appear the most in the reading history will be submitted to the content providers, implying which topics the reader is most
interested in. Then the content provider applies topic modeling to the pool of reading materials the provider has to offer (this process is usually completed when the reading material database is first established), identifies documents that contain similar topics, and offers them to this reader.

Our project uses LDA instead of SVM as the classifier of the R-CNN algorithm. We will explain this choice in detail in section 3.

2. Outline

This report will first explain the objectives, motivations, scope, and deliverables in section 3 – 6. Choices and justifications will also be discussed in this part. Later in section 7, we will introduce some related works. Section 8-9 are used to illustrate the prerequisites, methodology and expected results of the project, followed by the progress and results in section 10 and 11. The obstacles and the recommendations of the project will be presented at last in section 12-13.

3. Motivation

This section will explain the reasons why R-CNN and topic model (LDA) are chosen as the research topic and why Python, specifically Python 3, is chosen as the language to implement this project.
a. R-CNN

As mentioned earlier, R-CNN is a milestone in the study of object recognition solutions and inspires the development of subsequent algorithms. Thus, studying R-CNN not only helps understand the newer algorithms inspired by R-CNN algorithm, but also provides potential opportunity to further improve the algorithm. There are many object recognition algorithms using deep learning including R-CNN, Fast R-CNN, Faster R-CNN, YOLO and YOLO v2, yet R-CNN is still the most preferred one as a research target. There are three reasons why R-CNN is so unique.

First of all, R-CNN is the pioneering algorithm with satisfying performance [6] [7] which applies deep learning technologies into the area of object recognition. Many of the later algorithms like Fast R-CNN, Faster R-CNN are all based on R-CNN. Conducting research on the original version of the algorithms is beneficial to the understanding of the topic.

Secondly, R-CNN algorithm has a modular structure. The modules use interfaces to communicate with each other. This “hierarchical, multi-stage” [4] design of R-CNN is easy to make replacement and
improvement on, as the different modules are relatively isolated with each other. The modification of any part won’t break the general structure of the algorithm, just like replacing a piece of brick from the wall. The flexibility of the R-CNN algorithm leaves great possibility for researchers to make various attempts in order to improve the performance of the algorithm. This project is one of those attempts. This project is going to make a comparison between two classifiers, SVM and topic model (LDA). A stretch of this project will be to actually implement the algorithm with LDA as its classifier and compare the different behavior of the algorithm with the two different classifiers respectively.

Thirdly, R-CNN algorithm is easy to implement. A lot of open source libraries can be exploited to implement algorithms used in R-CNN. Take CNN as an example, potential choices include Caffe, Keres, PyTorch and TensorFlow, which all satisfy this project’s requirement.

b. Python 3

The original R-CNN is written in MATLAB, which is a restricted commercial software that sells its license at a high price of $55 for student version. What’s more, MATLAB is a rather isolated
language, as it doesn’t provide strong support on the connection with other languages. It’s not so friendly for developers either, for it contains much less libraries and packages than Python does. In contrast, Python is a popular and strong programming language which is widely used in machine learning. It is free and open source; and consequently, it has more users and active communities, which in result produces numerous third-party libraries and add-ons for developers to help conduct their research.

Currently there are two versions of Python that are widely used by developers, Python 2 and Python 3. The latter one is chosen as the main language used to implement this project. The reason why Python 3 is preferred is that although still widely used, Python 2 is the outdated legacy and will stop being supported on 2020. Besides, all the tools and packages this project need all have the Python 3 version. Taking future improvement and maintenance into consideration, Python 3 is a better choice.

c. Latent Dirichlet allocation

As introduced before, LDA aims to identify and calculate the distribution of hidden “topics” in the documents. According to the theory of topic model [5], a document is a mixture of several topics,
and each word in a document belongs to one or more topics. As shown in Table 1, a word may belong to different topics and a distribution of possibility will be assigned to each topic.

By applying the Dirichlet distribution formula, the document will generate a report showing the proportion of each topic and this report can be used for classification.

<table>
<thead>
<tr>
<th>Words</th>
<th>Food</th>
<th>Hardware</th>
<th>Social</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Company</td>
<td>Network</td>
</tr>
<tr>
<td>Apple</td>
<td>50%</td>
<td>50%</td>
<td>0%</td>
</tr>
<tr>
<td>Microsoft</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
</tr>
<tr>
<td>Facebook</td>
<td>0%</td>
<td>20%</td>
<td>80%</td>
</tr>
</tbody>
</table>

The reason why LDA can be applied to object recognition is that documents and images have similar structures so that the theory of topic model also take effect. As illustrated in Figure 1, an image is a combination of objects and each object have a lot of characteristics that related to it.
Apart from the similarity between image and document which shows the rationality of applying a document-based topic model as the identifier for image-related problem, the feasibility of such approach has been proved by Han Bing and Yang Chen. They modified the LDA model and developed an algorithm based on the modified version of the model to classify and analyze images.

Furthermore, this project is not the first attempt to apply the LDA model in R-CNN algorithm. Z. Pan, Y. Liu, G. Liu, M. Guo and Y. Li used the LDA model as the pooling layer of CNN and modified the R-CNN algorithm accordingly.

4. Objectives
   a. Task setting
Two major tasks will be accomplished in this project. The first is to reimplement R-CNN with python. The objective of this task is to reduce the study cost of R-CNN and strengthen its extendibility. R-CNN is an open-source project written in MATLAB. As explained in the last section, using an open-source language to reimplement R-CNN can reduce the cost and may attract more people to join the research. In addition, Python has more third-party libraries and resources than MATLAB. These libraries can be applied to improve R-CNN in the future with little modification, which can strengthen R-CNN’s extendibility.

The second task is to replace the original SVM classifier with LDA topic model. R-CNN is designed as a multi-stage pipeline and can be divided into four main parts: 1. Image Input 2. Selective search [8]. 3. Convolutional Neural Network. 4. SVM Classifier. Details of those parts and how they proceed data to the next stage will be illustrated later in section 9. This project aims to replace the SVM modules with LDA. The intention of this operation is to test the performance of LDA in object recognition and positive results are expected.

b. Expected Results
Better performance is expected. The original R-CNN achieve 49.6% mAP [4] when using dataset PASCAL VOC 2012 for evaluation and our model is expected to outperform this record.

If fair results are achieved, this partially confirms the assumption that LDA can be used for object recognition and worth further development. Otherwise, further research should be made, and deductions should be revised.

5. Scope

The project will focus on the reimplementation and improvement of the original version of R-CNN. Later algorithms such as Fast R-CNN, Faster R-CNN and YOLO will not be considered and analyzed in detail in this project.

This project will only focus on the research of the algorithm itself. No web nor mobile UI for external users will be implemented and no application and service using this algorithm will be created. We are considering implementing a simple UI in the future for presentation purpose. This will provide the audience with a more intuitive understanding of the algorithms and the test results.
In order to retrieve an accurate and reasonable result, all the possible interference should be eliminated when we are comparing the two classifiers (LDA and SVM). Thus, all parts of the algorithm except for the classifier part will remain unchanged during implementation.

The project will only focus on three datasets: ImageNet ILSVC 2012 and PASCAL VOC 2007 for training and PASCAL VOC 2012 for evaluation. No other datasets will be used to train or evaluate the model. The reason for this decision is that the original R-CNN is using those three datasets and to make better and fairer comparison, all factors should be kept as same. This means that the model may not have a good performance when applied to real-life scenarios because there are more objects and more categories of objects in the real world.

6. Deliverables
The project aims to deliver a python implementation of the R-CNN algorithm with a topic model named Latent Dirichlet allocation as the classifier. The original classifier SVM will be replaced by LDA and an improvement on the algorithm’s performance is expected. A detailed performance report will also be delivered to show the results of the new model, its comparison with the original R-CNN and some analysis.

A webpage is built for the project to record the project progress and it
should be updated regularly.

7. Related works

Some related works will be introduced and explained in this section. Some of them are about applying LDA into the region of computer vision and object recognition, others are trying to make an improvement on the R-CNN algorithms by various means.

Plenty of the new algorithms are created to make an improvement on R-CNN and many of them achieve satisfying results. Fast R-CNN + YOLO yields 70.7% mAP and Faster RCNN, ResNet (VOC+COCO) yields 83.8% mAP, both exceeds the R-CNN’s record of 53.3% mAP greatly [7] [8].

<table>
<thead>
<tr>
<th>Models</th>
<th>Caltech101</th>
<th>VOC2007</th>
</tr>
</thead>
<tbody>
<tr>
<td>VQ [14]</td>
<td>74.41</td>
<td>56.07</td>
</tr>
<tr>
<td>LCC [16]</td>
<td>76.95</td>
<td>57.66</td>
</tr>
<tr>
<td>FK[17]</td>
<td>77.78</td>
<td>61.69</td>
</tr>
<tr>
<td>SPP net [2]</td>
<td>91.44</td>
<td>70.82</td>
</tr>
<tr>
<td>Zhiling Pan’s team</td>
<td>79.98</td>
<td>65.13</td>
</tr>
</tbody>
</table>

Zhiling Pan and his/her colleagues have modified R-CNN by replacing the pooling layer of CNN with LDA topic model [9]. Table 2 indicates that their version outperforms the original R-CNN also. Han Bing and Yang Chen have applied an LDA based algorithm to classify and
analysis aurora images by treating the images as documents [10]. There are also researches making comparisons to LDA and SVM showing that LDA based algorithms are more robust and outperform SVM when trained by either large-scale or small-scale datasets [11].

8. Prerequisites
This section will introduce the prerequisites for this project. This project is developed and will be tested and evaluated under this specific environment. Any modification of environment may affect the final results. Some of the prerequisites are only compatible with Linux operating system, so this project may not be able to run properly on Windows. This project has a high and strict requirement for the hardware, a lower configuration may lead to longer training time and unsatisfying results.

a. Hardware requirement
   i. Intel i7 CPU or above.
   ii. 32G RAM or above.
   iii. NVIDIA GeForce GTX1080 GPU or above.

b. Software prerequisites
   i. Caffe [1] and Caffe's prerequisites with Python layer and
pycaffe framework.

ii. **Matlab** to build R-CNN source code.

iii. **Python 3** to reimplement R-CNN.

iv. **R-CNN** source code which based on Matlab.

v. **Linux** operating system.

vi. **SSH enabled** for remote access.

9. **Methodology**

In this section, development and training methodologies will be introduced. The usage of three datasets will first be explained, followed by the development methodologies. Finally, the training methods will be clarified. Since R-CNN is following a modular design pattern, each module of R-CNN will be introduced respectively.

a. **Datasets preparation**

Because this project aims to make a comparison with the performance of the original MATLAB version and that of the Python version, it is fairer to train and test those implementations with the same datasets. In this project, three datasets mentioned in the R-CNN paper [4] are leveraged. ImageNet ILSVC 2012 are used to pre-train the CNN network, PASCAL VOC 2007 are used to train the classifier and optimize the parameters and PASCAL
VOC 2012 are used to evaluate the performance. All the datasets are opensouce and free.

b. Python version implementation

![Diagram of R-CNN](image)

Figure 2 – General flow of R-CNN [4]

R-CNN is designed as a pipeline and Figure 2 shows the general flow of it. The first step is to input the image to do object recognition. Then R-CNN applies selective search algorithm on the image and 2000 regions that are highly likely contains some single objects are extracted. Those segmentations are all rectangles but are of different sizes. Because the CNN algorithm that the project uses in step 3 requires the input segmentations to be equal size, R-CNN resizes the segmentations to make them all equally big. R-CNN then perform the calculation of features by using CNN and yields a feature vector for every input segmentations. Finally, R-CNN does the classification using SVM or topic model. More details of each part of R-CNN are discussed below.
Selective Search is the algorithm that are leveraged in this project. It generates a certain number of image segments which may contain an object or a group of objects. As shown in the left part of Figure 3, the algorithm is using a bottom-up approach to segment the target image [13] [14]. The algorithm is based on the view that pixels deliver less information [12] that regions when identifying a general object so regions are treated as the basic unit in the algorithm. It starts with some small regions generated by Felzenszwalb and Huttenlocher’s method [14] to avoid multiple objects appearing in the same region. Then it combines those regions of different colors based on their similarities. Regions with higher similarities are merged into a bigger region. Finally, only one or two big segmentations remain on the image. Figure 3 illustrates this procedure well, the left-bottom image is quite colorful and after
some combinations, the left-top image only has two segments colored. During the combination process, the algorithm ranks every region regardless of their size and shape. And after the combination completes, a certain number of regions with the highest ranking are picked out as candidates, like the right part of Figure 3. In this project, the number of candidates selected is 2000.

ii. Convolutional neural network

Convolutional neural network is a popular and tested artificial neural network and it is widely used in the region of image analyzing and processing. Many successful algorithms based on ImageNet datasets, such as R-CNN and T-CNN [15], utilize CNN to do the image analyzing work. CNN does not need a lot of pre-processing which means it does not require much manpower to manually input features into it. CNN is inspired by and quite similar to the natural neural networks of human beings. CNN consists of an input layer, an output layer, convolutional layer and pooling layer, each can be matched to something in the real neural. Input and output layers are the eyes, nose, mouth and any other sensory organs that are interacting with the outside. The way convolutional layer
working is like how people learn or study, the more you read, think and summarize, the more impression and knowledge you will get. And the pooling layer kicks out less important features under a certain category to save space, which has a little resemblance to our memory management system.

There are several open-source implementations of CNN available, e.g. TensorFlow and Caffe [1]. This project uses the Caffe to implement our convolutional neural network. Caffe is very user-friendly and easy for beginners to use. It also provides a simple but well-documented tutorial of object recognition using a dialect of R-CNN for newcomers to catch up. Although Caffe is written by C, it has a python interface PyCaffe for us to leverage.

iii. Latent Dirichlet Allocation

As introduced before, Latent Dirichlet Allocation is an example of topic model and is originally used in text modeling, analysis, and classification [5]. LDA is a model that treats documents as a collection of words and there is no sequence order between words. It claims that a document is consist of several topics and every word is generated by one of the topics. This concept also
applies to object recognition, because images can also be treated as a combination of topics. The popular concept of tagging images is quite like the idea of topic in LDA.

Additionally, compared to SVM, LDA is an unsupervised learning algorithm and does not require manual work to prepare pre-training data. This will save memory and speed the system up.

Since R-CNN is using a modular design, a replacement in this section won’t have great effects on other parts of the model. Only the input datatype format should be slightly modified since the model is using a new classifier.

c. Training

The python version of R-CNN are trained with the same datasets used by original R-CNN. More datasets may be introduced if the system yields a promising performance and time is allowed. Some pre-trained model checkpoints should also be settled. If later training does not yield a good result, the training process can be rolled back to one checkpoint.
10. Progress

Current status and future planning will be introduced detailly in this section. Tasks accomplished will be explained and illustrated separately.

a. Environment setup

The project will be running on the workstation provided by Professor K. P. Chan. The workstation is equipped with Intel i7 CPU, 32G RAM, and NVIDIA GeForce GTX1080 GPU. It has 157G available disk space now which is not enough for this project because the training procedure of R-CNN requires about 200GB of free disk space to store the feature cache [16]. More disk space will be added by Professor K. P. Chan.

Access to the workstation has been granted and Python 3, Caffe [1], Caffe’s prerequisites and Pycaffe, a python interface for Caffe, are installed. MATLAB will be installed later.

b. Project webpage

A webpage has been set up for the project to give the audience a brief introduction of the project. Documents, project progress and contacts of project members are posted on the webpage and it will
be updated regularly. The URL of the webpage is: http://i.cs.hku.hk/fyp/2017/fyp17015/ and below is a screenshot of the project website.

![Screenshot of the project website](image)

Figure 4 – Screenshot of the project website

c. R-CNN reimplementation

As introduced before, R-CNN is following a modular design and can be divided into 4 major modules, so the reimplementation process is also based on the modules and the reimplementation for the first tow modules have been completed (as shown in Figure 6).
When doing dataset preparation, we noticed that another student has already downloaded PASCAL VOC 2012 dataset, so we didn’t download it again to save disk space. A model pre-trained by ImageNet ILSVR2013 are exploited and this avoids the download of ImageNet ILSVR2013.

The image input module is implemented as a command line interface which means that there is no UI for this module. After an image is inputted into the system, it will apply selective search to it and identify areas that have higher possibilities to contain objects. Those areas are represented and located by the coordinates of vertex points along with the size of the area.
In Figure 7, the first red rectangle is the code to input an image and the second red rectangle is the results generated by selective search. This implementation is not easy to understand so a temporary converter is implemented for easy illustration and testing. Figure 8 illustrated the converted sample results generated in Figure 7.

When implementing this part, we learned and took advantage of
some opensource GitHub projects, e.g. AlpacaDB’s selectivesearch project [17] and saisrivatsan’s selective-search project [18]. Because their input and output type cannot satisfy the project requirement, we implemented our own version of selective search.

We make use of scikit-learn package to implement the SVM classifier.

d. Timeline

<table>
<thead>
<tr>
<th>Time Frame</th>
<th>Task</th>
<th>Status</th>
</tr>
</thead>
</table>
| Nov. 2017 – Oct. 2017 | ● Study basic knowledge  
                      ● Environment setup  
                      ● Build project website | Completed |
                      ● Train and evaluate Python version of R-CNN | Completed |
<table>
<thead>
<tr>
<th>Date</th>
<th>Task</th>
<th>Completion Status</th>
</tr>
</thead>
</table>
| Jan 2018 – Feb 2018 | - Replace the SVM classifier with the LDA topic model without changing other parts  
                    - Train and evaluate the new version | Completed          |
| Mar 2018   | - Make a comparison with the results and performance of the original R-CNN, the python version with SVM classifier and the python version with LDA classifier  
                    - Optimize parameters for CNN network and classifier models | Completed          |
| Apr 2018   | - Final report submission  
                    - Final project presentation  
                    - Final presentation poster design | Completed          |
| Apr 2018   | - Final project exhibition | Completed          |

### 11. Results
We first test our model using the python implementation of R-CNN. The classifier we were using is SVM. The best mAP we got among all the experiments is 45.2%. This score failed to exceed original R-CNN’s 49.6% mAP.

12. **Obstacles**

a. Segmentation size

We encountered an obstacle in this project. In our project, the selective search algorithm generates 2000 segmentations with different sizes and shapes. Nonetheless, the CNN requires all the input segmentations to have exactly same size. To resolve this conflict, three possible measures can be taken.

The first one is to fix the images to the same size during the selective search. The problem with this method is that it cannot position an object correctly. It may miss a lot of important objects since many objects may be bigger than the size chosen. It may also include multiple objects in one segmentations which will confuse the neural network. Column B in Figure 4 is the result of after applying this method, there appear two horses in one segmentation.
Figure 8 – Different ways to resize a segmentation. A is the original segmentation.

The second one is to wrap all the images to a certain size after generating the images. However, this may cause some misunderstanding for CNN. Because the size of the object has been modified, a small little tree may be recognized and treated as a big wooden door after resizing. Column D in Figure 4 is the consequence if this method is applied.

The third method is to pad the picture with some background color. This may also cause confusion since the neural network is taking the whole picture into consideration. Column C in Figure 4 is the example of after using this method.

All the three methods may reduce accuracy and affect the
performance. In order to reduce the impact, a combination of the second and third method are conducted: pad the image with a background color similar to the original background color when the image is small and wrap the image when it is relatively big.

b. Hardware issue:

R-CNN requires high-performance hardware support. High-performance GPU, large disk spaces (SSD preferred) are needed to support the calculation and training. After switching the language to python, hardware requirement are even higher, and the original project schedule suffers from some delays.

Mitigations:

i. Adopt a more flexible project schedule.

ii. Upgrade hardware beforehand to prevent accidents due to the hardware problem.

13. **Recommendations and Future works**

a. Reduce cache image size

The 2000 segmentations generated by the selective search module are stored on the hard disk and it occupies a lot of storage (~200G) [8]. In addition, disk read and write operation consumes a lot of time which slows down the whole system. Further research can be
made to solve this problem. Possible solutions include directly pass the coordinates to next module instead of using the coordinates to crop the image.

b. Apply supervised LDA classifier

The original R-CNN is using a supervised classifier SVM and our model is using an unsupervised classifier LDA. In order to make a better comparison, supervised LDA is worth trying. This can yield a more comprehensive understanding of the two classifiers. Some LDA dialects, e.g. sLDA [17], may be a potential candidate for this research direction.

c. Simple UI to assist presentation

We hope to implement a simple UI to upload image easily and show the plotted result intuitively. A sample result is Figure 9. Objects will be highlighted with rectangles and the categories of the object will be marked.
14. Conclusion

This project aims to reproduce the R-CNN with python and replace SVM classifier with LDA topic model. We make use of a lot of open source packages including scikit-learn and Caffe to implement and train our program. We also leveraged some pre-trained model to reduce time and increase efficiency. Although we don’t have a better performance, we still managed to prove that LDA has the potential to be used as a image classification algorithm.

Because of the limited time, we didn’t get satisfying results. The best precision we have didn’t beat the original R-CNN. We will continue to improve our program as well as the algorithm and conduct more
experiments. We also plan to build a simple UI for the project to test and display intuitively and easily.
References


