Comparing STNET using different CNN approaches.

Jose Manuel Paredes del Rio

Department of Computer Science

Aalto University

FI-00076 AALTO, FINLAND
jose.paredesdelrio@aalto.fi

Abstract

The new trends in Object Recognition using Neuronal Networks are related to visual attention. It inspired the creation of Selective Tuning of Convolutional Networks (STNet). Bottom-Up and Top-Down information processing are combined to tune the visual representation of Convolutional Neuronal Networks (CNN). STNet has been evaluated previously over the ImageNet benchmark dataset for the task of weakly-supervised.

This evaluation was performed using AlexNet approach, it shows a big improvement of the state-of-the-art. The paper purpose is to evaluate this idea over more CNN architectures which are already developed such as VGG.

Introduction

Since the late 1960s, computer vision techniques have pioneered in the field of artificial intelligence. These methods were meant to emulate the human visual system; it was going to be an immeasurable improvement to endow robots with intelligent behaviour.

After some research in the field, it was discovered that this task was not as easy as expected. Despite these situation difficulties and thanks to some software and hardware improvements, the community have never stopped to develop more methods to solve this problem.

Currently, the best methods to perform this task are focused on convolutional neural networks. One example of their achievements can be reviewed in a challenge called: ImageNet Large Scale Visual Recognition Challenge. This dataset is the most famous benchmark in the area of object detection and classification. The number of pictures included reaches millions and there are hundreds of object classes. One fact to keep in mind is that the performance of this kind of network is similar to human beings on the ImageNet test.

Recently, the community has developed new algorithms and models focused on various tasks in the visual field such as object detection and classification, semantic segmentation, scene understanding and action recognition. The main improvement of these models is the obtaining of a visual representation that is much more precise. Therefore, the amount of data and information given by the picture or video is bigger and easier to analyse.

Most of these new ways to manage information have been developed in the area of Deep Learning.

Deep learning is one of the methods of machine learning inside the area of neuronal networks (NN). NN tries to replicate or emulate the human brain. Therefore, the information is analysed and processed by some layers of nodes, also known as neurons. One particular type of NN is Convolutional Neural Networks (CNN). CNNs are widely used to process images to improve the results obtained by NN. CNNs can preserve the main information of the picture performing some transformations in the input data; this feature makes CNN the best option to analyse pictures.

Similar to the NN, the biological processes were the inspiration of CNN since it emulates the connectivity pattern among neurons. It resembles the way the visual cortex is organized. Thanks to this method, a receptive field is created; it means that if there are stimuli, responses from each cortical neurons are restricted in a visual field region. These different neurons receptive fields are overlapped partially; therefore, they cover the entire visual field.

In comparison to other types of NN, CNN uses relatively little pre-processing in the task of classifying images. This feature means that the network can learn the filters which have to be hand-engineered in the traditional methodologies. Therefore, its major advantage is the ability to perform independently from previous knowledge and human effort in feature design.

Regarding the methods already explained, most of them involve the management of CNN to improve the visual analysis of pictures. In this article two of them are explained: Bottom-Up (BU) and Top-Down (TD)

The data-driven BU processing stream applies a convolution to the input data by a transformation of the information. It means that the BU processing stream, throwing hierarchical cascading stages to the information processing, forms the visual representation of the input data.

The task-driven TD processing stream modulates the visual representation in order to fulfil the task requirements. Therefore, the TD processing stream is the protector of the task knowledge on top of the formed visual representation to obtain task requirements.

As you can understand, both of them focus on a different approach to solve the same problem; visual learning. However, scientists have not developed both at the same time.

Unfortunately, the community attempts have benefitted the BU processing paradigm while TD processing has seldom gone unnoticed from the community of computer vision. The amazing BU results have caused this situation. These results have been quantitatively very successful by popular benchmark datasets. One example of these results is in references 3 and 4

STNet paper uses both paradigms in order to improve the results as well as introducing some new ideas in the TD area. The paper argues that the use of visual attention in TD can improve the visual representation of the images significantly.

The visual attention has two methods to choose from:

- Overt attention. The movement of the eye preserves the visual acuity at the fixation time while the formed visual representation is maintained intact. In more technical terms: as there is a lack of visual acuity throughout the entire field of view (perception, cognition and action cycle) it tries to compensate it by an eye-fixation controller.
- Covert attention maintains the fixation point unchanged and modulates the shaped visual representation

Regarding these methodologies and the previous two explained above, the paper purposes to assemble the BU and TD processing in a unique infrastructure which integrates the processes of attentive selection into the hierarchical representation. The name of this new network is STNet.

It is important to evaluate new approaches on benchmark datasets to analyse the strengths and weakness and to realize the repercussion over the community.

In order to evaluate STNet, it is needed to analyse the performance with more CNN architectures to understand how it works and come to conclusions. In this paper, STNet performance using VGG approach will be analysed over the famous benchmark ImageNet dataset.

Methodology

STNeT

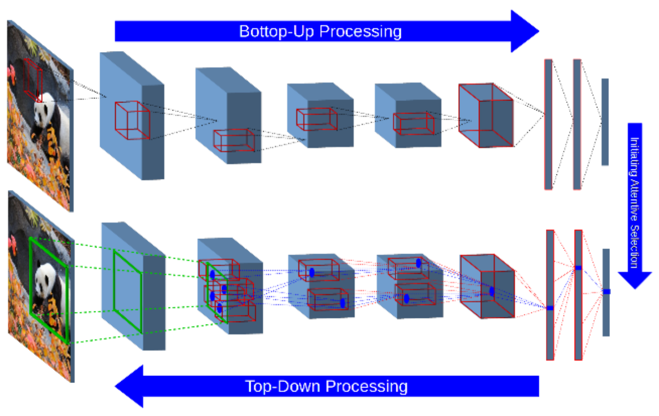
STNet is the integration of the conventional bottom-up processing by convolutional networks with the biologically-plausible attentive top-down processing, in other words, STNet consists of two interactive processing streams.

BU stream forms the representation throughout the entire visual hierarchy. It means that the information is processed layer by layer in a strict parallel paradigm. This pathway processes information at each layer using a combination of basic operations such as convolution, pooling, activation, and normalization functions.

The task-driven TD processing stream modulates the visual representation in order to fulfil the task requirements. Therefore, the TD processing stream is the protector of the task knowledge on top of the formed visual representation to obtain task requirements.

The information flow over both streams layer by layer, in the way that according to the hierarchical structure the layer output feed the following one.

The BU structure can be any stream which can perform the primary visual task. In this project, AlexNet and VGG have been chosen for the BU stream.



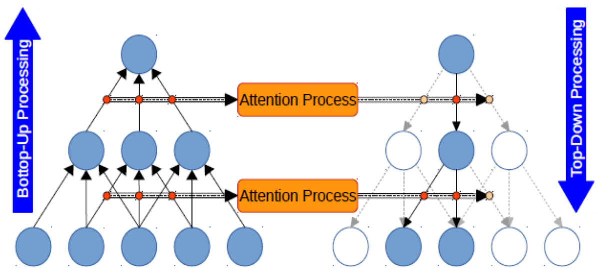
Structure of the Top-Down Processing

The TD structure is defined based on BU processing. The TD processing is initiated and pierced layer by layer looking for the layer which satisfies task requirements.

There are some new nodes to interact with the BU processing structure. These nodes determine the TD information flow and they are rarely activated as the TD processing is adjusted to activate just relevant parts of the representation

To process the attentive visualization, it just returns the set of all the nodes in the layer below that falls inside the receptive field of the top node according to the BU processing connectivity structure.

More precisely, it locally processes information to determine connection weights of the TD processing structure and consequently, the gating node activities at each layer. Once the information flow in the BU processing stream reaches the top of the hierarchy at layer L, the TD processing is initiated by setting the gating node activities of the top layer as illustrated in the following figure.

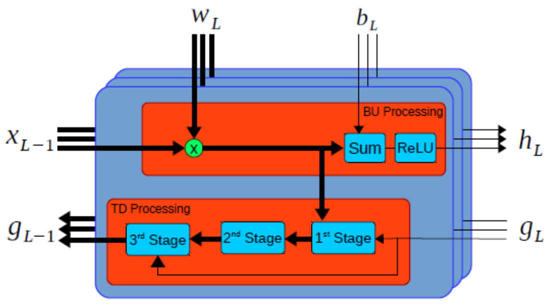


Connection weights between the top gating node and all the gating node in the layer below within the are computed using the attentive
selection process. Finally, the gating node activities of layer L - 1 are determined according to the connection weights. This attention procedure is consecutively executed layer by layer downwardly to a layer the task requirements.

Stages of Attentive Selection

During the training phase at the backpropagation weights of the BU structure are learned. On the contrary, the TD structure computed weights using a deterministic and procedural selection process from the Post-Synaptic (PS) activities.

The selection process has three stages of computation. Each stage processes the input PS activities and then feed the selected activities to the next stage.



In the first stage, the purpose is to distinguish relevant regions from irrelevant ones. Winner-Take-All (WTA) is a biologically plausible mechanism that implements a competition between input activities. At the end of the competition, the winner retains its activity, while the rest become inactive.

Algorithm 1 Parametric WTA Optimization

```
1: $NEG(PS) = \{s \mid s \in PS(g), s \leq 0\}$
2: $POS(PS) = \{s \mid s \in PS(g), s > 0\}$
3: $SUM(NEG) = \sum_{n_i \in NEG(PS)} n_i$
4: $buffer = SUM(NEG)$
5: $i = 0$
6: while $i \leq |POS(PS)|$, $buffer < \epsilon$ do
7:     $buffer += SORT(POS(PS))[i]$
8:     $i += 1$
9: end while
10: return $SORT(POS(PS))[i-1]$
```

The goal of the second stage is to make a more restrictive selection procedure.

Depending on the spatial dimension of the current layer it is applied one of the following selection modes: Spatially-contiguous (SC) and Statistically-Important (SI) selection.

As in the fully connected layers, there is no order in the information among nodes, SI selection mode is proposed to find the statistically important activities. Based on an underlying assumption that the node activities have a Normal distribution.

On the other hand, convolutional layers preserve spatial ordering. SC determines the most spatial region of the winners based on their PS activities.

In the beginning, SC selection mode partitions the set of winners into groups of connected regions in the way that a node has eight immediate adjacent neighbours and a connected region is defined: the set of all nodes that are recursively in the neighbourhood of each other.

The result of the SC selection mode is the set of nodes that fall inside the winner connected region. Thanks to this set, it is straightforward to compute the weights connections in the TD structure.

In the last stage, the winner activities are normalized. Once multiplicatively biased by the top gating node activity, the activity of the bottom gating node is updated consequently.

Experiments and Results

STNet uses two stream processes: Top-down and Bottom-Up. Therefore, BU processing is not enough and visual attention TD complete the object localization task.

STNet is developed in Pytorch, it uses AlexNet as a base architecture of the BU structure (ST-AlexNet) and the model weights are retrieved from a Pytorch Model Zoo repository. They have been calculated over the ImageNet benchmark dataset.

The ImageNet project is a large visual database designed for use in visual object recognition software research.

The ImageNet dataset contains more than 14 million images which belong to more than 20,000 classes. They also provide bounding box annotations for around 1 million images, which can be used in Object Localization tasks.

The purpose of this section is to develop a new STNet using VGG (ST-VGG) approach and analyse the results comparing them with ST-AlexNet.

The localization prediction accuracy is measured using Intersection-over-Union (IoU) metric. The label prediction accuracy is measured using Top1 metric.

ST-AlexNet Results:

Parameters	**Values**
IOU	0.5
METERS	Label accuracy
NUM CLASSES	1000
BATCH SIZE	128

INPUT SIZE	[3, 224, 224]
Dataset length	50000
Number of mini-batches	390

Time running: 20 minutes
Label Accuracy: 55,64%

ST-VGG Results:

Parameters	**Values**
IOU	0.5
METERS	Label accuracy
NUM CLASSES	1000
BATCH SIZE	50
INPUT SIZE	[3, 224, 224]
Dataset length	50000
Number of mini-batches	1000

Time running: 5 hours 20 minutes
Label Accuracy: 68,77%

Conclusions

The STNet approach has been tested on the task of object recognition over one of the biggest benchmark datasets. The results show the potential of this new approach improving the performance when using the combination of the two processing streams over the common BU approach.

To make a proper comparison, it was needed to perform the test of ST-AlexNet and ST-VGG with the same parameter but unfortunately, due to the resources available the batch size had to be decreased (from 128 to 50) at performing the ST-VGG test, it can affect the final result of the test. The parameters used to perform these tests can be seen in the previous section of this paper.

Both of them improved the performance of the base neuronal network without the visual attention approach.

One interesting feature is the running time, it is known that VGG uses a large number of resources but at using ST-VGG the consumption of memory and the computational cost increased hardly, therefore the running time is more than five hours. On the other hand, ST-AlexNet running time is about 20 minutes. The final accuracy of each model is the following:

	ST-AlexNet	**ST-VGG**
Label Accuracy	55,64%	68,77%

There is a big improvement of the accuracy in the ST-VGG approach but using that model imply more time and the need for a powerful computer.

Project Difficulties

In this section, I collect the main issues I faced to develop this project as well as what I have learnt.

In my opinion, the high barriers have been the working environment and the already developed code of STNet.

I used a shared environment provided by the university, Triton. In the beginning, it was a little bit confused about how it works, how I could install the software I need. Actually, the online documentation is really explanatory and the Triton staff has helped me satisfactory in garages or by email.

Regarding the STNet code, it was not simple to understand how it works and it was needed to go through all the files to know it. I am really grateful for the code developer as he shared it with me and I did not have to develop from scratch but it implied some restrictions of software versions and some version mismatching appeared.

As it is a deep learning research project which goes deep into neuronal networks, therefore, some new layers have to be defined, compiled and added to the project, it caused huge research of weeks to compile custom layers for different architectures. Hard coding has been another problem in the code, for example, the place where the custom layer where shaved and the maximum memory of the device was hard coded.

I would like to mention also that getting the dataset was more difficult as I had expected. It was required to use ImageNet dataset. I sent the request to obtain it by the official webpage and I got no reply. Fortunately, I could obtain it by Kaggle competition although I had to ensure that the structure of the dataset fit the required structure for STNet code.

Time has been another issue to declare, I mainly started this project in December but I had some university deadlines and exams, therefore, I had to postpone it until February.

Thanks to these issues I had reinforced my self-learning characteristic and I learned new knowledge.

Understanding acquired

I mainly reinforced my self-learning feature. I could study a trending topic in the computer vision area, Visual Attention, and the main tools in the field. Besides, I learnt how to develop a project in a shared resources context. In this case, I used Python and Pytorch as well as Cuda library to compile custom layers. It gave me the vision that everything is not developed and if there is an idea it can be done. I am always surprised by how different technologies can work together.

I have learnt about the shared working environment, I am sure that I will use this in the future as some companies would use a similar approach in relation with it, I used Python environments and I found it really useful in this context, I already knew about them but I have not used them.

As a conclusion of this project, I am satisfied with my work since I worked in a totally new environment about one trending topic and I managed to make it running in spite of the difficulties. For all these reasons I can claim that my results are satisfactory.

Running this project in Triton

As I mentioned in the previous section of this report I worked on Triton and I find interesting to give some instructions for future researchers in case they want to replicate my results in Triton.

First of all is to have a look of the STNet creator GitHub webpage: https://github.com/mbiparva/stnet-object-localization.

The best way to gather the ImageNet dataset is by using kaggle. Make sure that the structure of the folder is the same that is mentioned in the documentation. I also tried to get the dataset from the official webpage but I had no replies.

The prerequisites section mentions that it is needed: CUDA 8.0 or higher, Python 3.6 and PyTorch 0.4.1. I managed to run with CUDA 9.

Therefore, the modules which are needed to load are CUDA and cuDNN. To develop the project, I used a CONDA environment with Python 3.6. so anaconda module should be loaded as well. Ensure that you install the correct version of PyTorch, the way that custom layers are developed changed in newer versions therefore that code will not work for recent versions.

Once the conda environment is ready it is time to compile the custom layers. To do that I recommend to keep in mind your GPU architecture and modify the Makefile to suit it. In addition, line 16 of the build_fii.py file should be changed to /share/apps/easybuild/software/CUDA/9.0.176/lib64/*.a. If it does not work maybe the library path is different in your case, check it: echo $LD_LIBRARY_PATH

When the custom layers are compiled the project is ready to run, good luck with your research.

References

1. STNet: Selective Tuning of Convolutional Networks for Object Localization: https://arxiv.org/abs/1708.06418
2. An Intuitive Explanation of Convolutional Neural Networks: https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/
3. Deep residual learning<for image recognition https://arxiv.org/abs/1512.03385

4. ImageNet classification with deep convolutional neural networks. https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf