Mkhanyisi Gamedze
COSI 165B Deep Learning
Final Report

World Event Recognition using Deep Convolutional Neural Network Architectures

Abstract

Human scene context analysis and classification into event categories is a very challenging problem within Computer Vision. The ChaLearn challenge in 2015 curated a world cultural event dataset consisting of 99 famous world cultural events and a 1 none class, for the ChaLearn Looking at People challenge. Most of the top performing submissions on this challenge utilized deep convolutional neural network architectures. This paper presents an exploration of deep learning models applied to this recognition task and tries to reimplement some of the results from the winning submissions and publication models from the competition. The dataset is trimmed down to only 11 event categories from the challenge, including the none-class as images as well. Pretraining models on the famous deep models ImageNet dataset improved the object recognition accuracy of the models explored. The 8-layer pre-trained AlexNet had the best performance with 61% test accuracy, compared to the other deeper models, as the ResNet34 was the second closes. Perhaps the deeper model did not perform as well in this task given the relatively small data size more fine-tuning could help improve accuracy.

Introduction and Motivations

Using computers to understand the context within images and intelligently identifying or classifying the categories which best describe the event is a problem that researchers have explored for a long time. In particular, event recognition from still images with people on the scene through automatic analysis of the human body is also named Looking at People. This is a domain that has been widely explored and has multiple subareas. To the evolution and biologically trained human visual system, event recognition is an almost effortless task, but this is a challenge that still remains hard to train a computer to intelligently perform well at in high accuracy. Even with the most recent advances in many Machine Learning or Deep Learning algorithms, there are still lots of areas left unexplored.

The recognition of continuous, natural human signals and association to activities or events is very challenging due to the multimodal nature of the visual cues (e.g., movements of fingers and lips, facial expressions, body pose), as well as technical limitations such as spatial and temporal resolution, especially within events of a cultural context. Identifying the activity at an event from still images presents a very challenging recognition problem due to the high variability of clothing choice, objects, human poses, illumination, and scene context to name a few. The world is so diverse and events are so complex, as there are many conclusions, classifications, or interpretations that can be drawn from a still image.

Classification problems from still images have been explored extensively within the field of Computer Vision, and developments have been rapid within the past decade, especially with deep learning architectures performing margins better compared to other traditional machine learning methods. Deep learning architectures grew in dominance within the field of computer vision in the last decade largely due to innovation challenges

and state-of-the-art improvements in computing resources plus data. Challenge competitions always spark innovative solutions to challenging unexplored research problems which sustain the effort in advances in knowledge, and so has been the case within deep learning.

The ChaLearn Looking at people challenge - Dataset Source

The ChaLearn Looking at People Challenge has continually run a series of competitions to explore the performance of recognition methods in identifying actions or gestures from still images. The 2015 Cultural Event Recognition track within the competition curated a dataset collected from two major search engines, Google and Bing. The dataset represented a total of 99 categories corresponding to different worldwide cultural events, and 1 none class to be considered as well. The total dataset contained a total of more than 28705 images mapped to event categories. It is important to note that this competition presented a unique dataset and problem that was not well explored in literature at the time. 70% of the dataset which constituted 20036 images was released for training and validation during the competition and the remaining 30% was used for testing.

Figure 2. Samples of cultural event recognition dataset at the ChaLearn LAP challenge 2015. The cultural event recognition dataset has 50 important cultural events in the world. It includes: Annual Buffalo Roundup (USA), Battle of the Oranges (Italy), Chinese New Year (China), Notting Hill Carnival (UK), Obon (Japan) and so on. All the images are collected from the Internet by using Google and Bing search engines. These images exhibit large intra-class variations and are very challenging for event recognition.

World events within consideration in the dataset included Carnivals (Brasil), Celebrations (Holi Festival, Chinese New Year), and costumes (Frozen Dead Guy Days, Halloween). Within the image categories, garments, human poses, objects, and illumination constituted the cues that could or can be exploited in recognizing these events as humans do in associating classifications.
From Escarela's ICCV conference paper highlighting the outcomes from the 50 submissions in the competition, it is noted that most of the participants based their solutions on deep learning architectures. All of the top-performing submissions used variations of deep Convolutional Neural Network Architectures including VGGNet, GoogLeNet, VGG16, CaffeNet, AlexNet, and others.

Table 7. Table of Cultural Methods.

Team	Proposed Method
VIPL-ICT-CAS	Model used: VGGNet, GoogLeNet. Prediction using Logistic Regression and LDA for final classification.
FV	Model used: VGG16, VGG19, Place-CNN. Prediction using 5 CNNs, fusion of 5 feature vectors and final logistic regression classification [20].
MMLAB	Models used: GoogLeNet, VGGNet. Prediction using Object and scenes activations [44], CNN features and Fisher Vectors [43], final SVM classification.
NU&C	Model: CaffeNet based on ImageNet and Places205. Combination of Object CNN stream and Scene CNN stream for prediction.

Important to note is that the winning submissions used adaptations of off-the-shelf VGGNet and GoogLeNet which were fresh off the shelf at the time, and other model submissions used pre-train and fine-tune or adapted their models to this task. This is widely known as transfer learning.

Project Focus

The well-documented challenge and top-performing submissions published papers as well presented the opportunity to explore my ability to work with the Cultural Event Recognition dataset and explore deep learning method implementations on par with some of the winning submissions or even better. Reimplementing some of the winning submissions with my computing resources and comparing results is the objective of my project and then reporting observations. In particular focus, the goal is to re-implement Limin Wang's Object-Scene Convolutional Neural Network architecture which was a winner of the previous year's iteration of the same competition and got outperformed by newer and deeper VVGNet and GoogLeNet architectures.
My analysis involves cleaning, formatting, and loading the challenge dataset for the multiple models and then implementing, training, optimizing, and testing recognition performance on the competition evaluation dataset.

Experiments

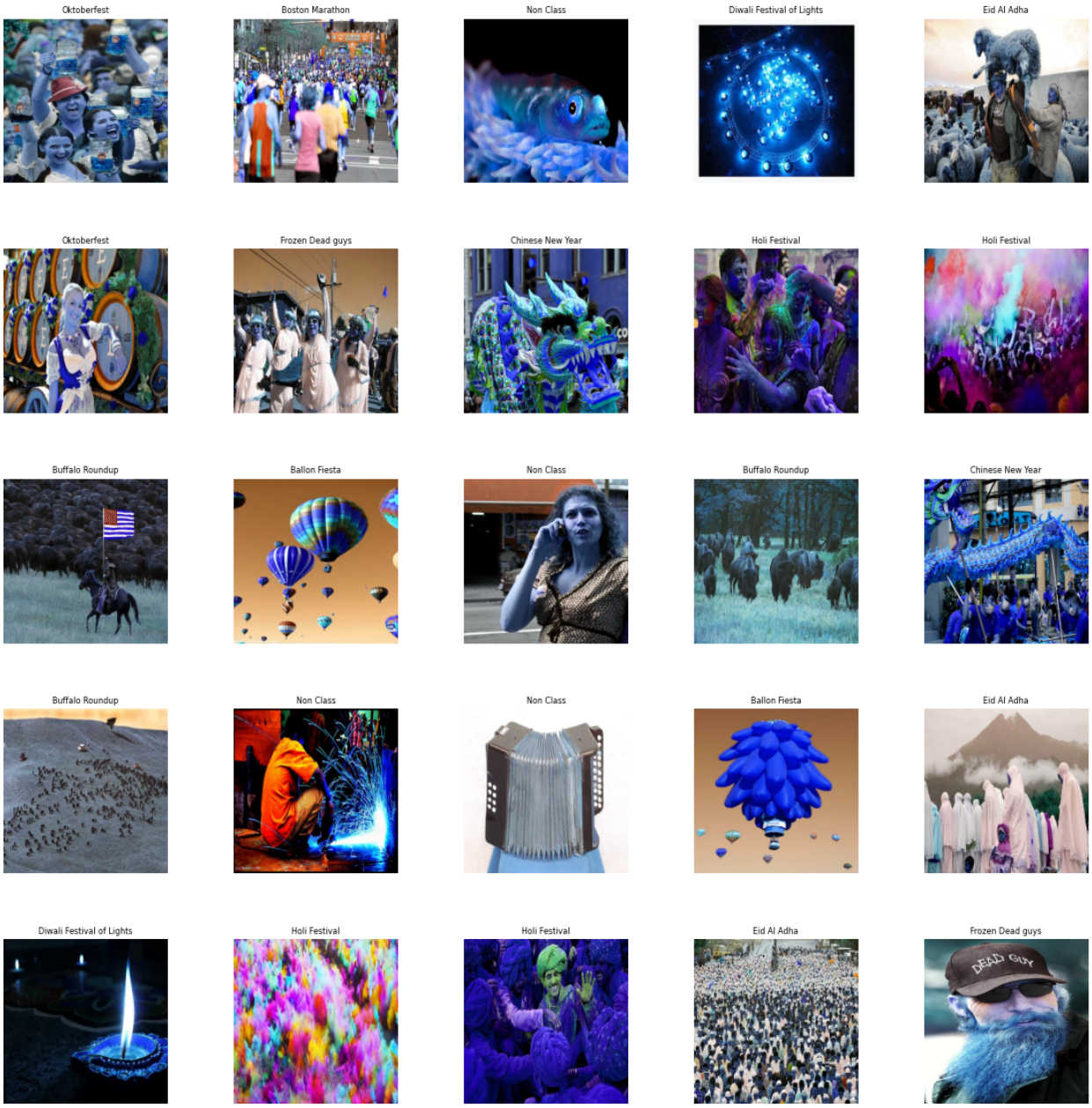
Implementation Details

The relatively large image dataset was downloaded from the CodLab competition website and was then formatted locally for training. All development was done in Python and mainly using the PyTorch deep learning library for model implementation. The competition images dataset was not provided in a clean format, but rather in multiple smaller files for individual images and text files indicating which images were within a label. Important to note as well was that the images were presented in raw format of various types and dimensions, and thus needed to be standardized after being mapped to labels. The train and validation datasets had a combination of 20036 images in total, with approximately 200 images per event classification category. This was all done within the load and exploration notebook, and the results were saved in the *imageLabelEncodings.xlsx* file. All the images were resized to 256 x 256 width-height with 3 RGB channels for the images and then shaped into

[256,256,3] NumPy arrays or PTorch tensors. These were then concatenated with corresponding encoded categories from 0 to 99 for loading into our deep learning models.
The train dataset images of size 20036 x 256 x 256 x 3 proved not only too large for training locally within my personal computer but using Google Cloud Platform GPU resources as well. To rescale and downsize the problem, I downsampled the event categories to 10 selected events instead and the non-class label images as well, for a total of 11 event classification categories.

Category	Encoding	Num images in Category
Ballon_Fiesta	0	189
Diwali_Festival_of_Lights	1	151
Holi_Festival	2	178
Frozen_Dead_Guy_Days	3	161
Annual_Buffalo_Roundup	4	161
Battle_of_the_Oranges	5	144
Oktoberfest	6	229
Eid_al-Adha	7	175
Chinese_New_Year	8	201
Boston_Marathon	9	165
Non-Class	10	500

My smaller dataset had 2254 total images and reencoded the 11 final categories with values from 0 to 11.



A grid example of some of the images from 11 categories used for training in the project

The small and final datasets used for training were saved in NumPy files, *smallerdataX.npy (Images),* and *reencoded_smallerdataY.npy* (Labels) for faster loading.
For training

Experimental or Training Results

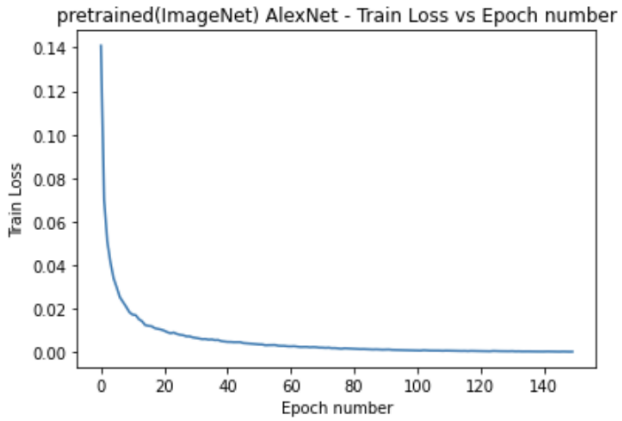
The size of the dataset and computing resources available to complete this project proved to be a challenge in the exploration of the results of this challenge. The deep CNN models explored within this dataset included two convoluted vanilla CNN models, vanilla AlexNet, and ImageNet pre-trained AlexNet, ResNet18, ResNet34, GoogLeNet, Inception, and ResNext50 architectures. All the models were trained using the Adam optimizer for backpropagation. The batch size and number of train epochs varied depending on the GPU limitations, as for computation heavy models in operations like the GoogleNet or VGG16, the GPU ran out of memory during training. The VGG16 model was never explored for this reason.
Inspired by the Object-Scene Convolutional Neural Network paper submission by Liming Wang, pretraining the models on the Imagenet dataset provided better object context for the models. Training the model on the competition data thus adapted the models to the recognition task of the Looking at People challenge, a common method among other submissions as well. The convolution models and weights of the convolution or feature detection layers were unmodified during fine-tuning, but the classification fully connected layers were modified so that the final layer had 11 classification output features with softmax.
A summary of the models and test performance is highlighted in the table below:

Model	PreTrained	Epochs	Learning Rate	Optimizer	Batch Size	Test performance (%)
Vanilla CNN	No	100	0.00001	Adam	250	***35.70%***
Vanilla CNN	No	150	0.00001	Adam	100	***40.80%***
AlexNet	No	100	0.00001	Adam	100	***33.70%***
AlexNet	ImageNet	100	0.00001	Adam	200	***58.54%***
AlexNet	ImageNet	150	0.00001	Adam	200	***61.197***
ResNet18	ImageNet	100	0.00001	Adam	100	***38.51%***
ResNet18	ImageNet	150	0.00001	Adam	100	***39.02%***
ResNet34	ImageNet	80	0.00001	Adam	100	***48.56%***
ResNet34	ImageNet	150	0.00001	Adam	100	***51.40%***
ResNext50	ImageNet	50	0.00001	Adam	5	***1.33%***
ResNext50	ImageNet	100	0.00001	Adam	5	***0.22%***
GoogLeNet	ImageNet	140	0.00001	Adam	2	***22.20%***
GoogLeNet	ImageNet	65	0.00001	Adam	2	***22.20%***

Summary of the training results of the deep learning models on ChaLearn Dataset

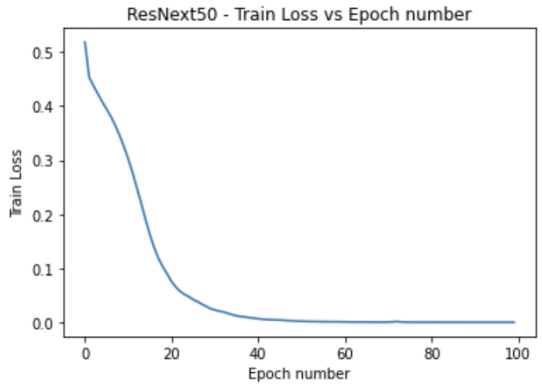
Analysis

The best performing model from the models explored was the AlexNet model pre-trained on the ImageNet dataset, as the Vanilla AlexNet model almost double its test accuracy from 33.7% to 58.54% on 100 epochs. The test accuracy of the pre-trained AlexNet model on the ChaLearn dataset is on par with the ImageNet accuracy of 56.6%. This does validate some of the initial assumptions from Limin Wang's Object CNN method, as the ImageNet dataset has a large collection of objects which always part of a scene context and helps improve accuracy. Like most of the other deep models' train loss vs epoch, once the training loss converged, the improvement in test accuracy was marginal on testing.



ImageNet pre-trained AlexNet model train loss converged after 100 epochs

The 4 convolution layer plain AlexNet and pre-trained models still performed better than the deeper ResNet and deep convolution GoogLeNet architectures to my surprise as the structure of the pre-trained object net for scene recognition I thought would be similar to the Imagenet classification challenge. Pretraining did prove to be effective, but the nature of the convolution layers and architecture of the models remains an area that needs further literature review on this problem. The 50-layer ResNext wide convolution architecture performed the poorest of all my explored models with a 1.33% accuracy at 50 epochs and 0.22% test accuracy at 100 epochs. Instead of improving performance as a model train loss converged, the overfitting through additional training deteriorated the test accuracy. The highly modularized network architecture repeats building blocks that aggregate a set of transformations that have the same topology.



Improvements and Development Areas

A major initial limitation in exploring this problem was the availability of computing resources to replicate the results of this challenge, as my Personal Computer was weak and most of the training of deep models had to be done on Google Cloud Services GPUs. Dense operation models like VGG deep nets could not be explored even within the high-end GCP GPUs, as the computations were memory intensive.
The ChaLearn challenge dataset was relatively large at 20036 images used for this project. For 100 category labels in the train validation dataset used, however, this meant approximately 200 images per event class in this challenge. My models were trained with a dataset of size 2254 in total images, and this is very small sample size for models to converge and give good classification results. The quick convergence of the training loss highlighted this challenge with most of the models. Pretraining on larger datasets or related problems helped improve accuracy significantly and thus I can conclude as well that having more samples per category would be beneficial as well. An additional solution would be to download and add the test images used.
I was unable to implement the Object Scene Convolution Neural Network paper, largely due to a lack of technical depth in implementing building the model given the AlexNet Places dataset pre-trained weights. Ensembling the pre-trained models on ImageNet(Objects) and Places(Scene) datasets would have provided the deep learning cues fundamental to this recognition problem and improved test performance. With more time, this would have been interesting to explore how the score fusion of two pre-trained models within an ensemble would improve accuracy.
From the literature review papers on winning models, not all other classification methods were implemented t improve the performance of deep learning models. The quick to converge Adam optimizer was used instead of Given the variety of objects within this dataset and scene contexts within different event categories, more advanced deep learning methods like detection and segmentation, or attention applied to this problem could be used to fine-tune the prediction results and improve the overall accuracy of models.

Conclusion

This paper presented an exploration of deep convolutional neural network architectures for the task of cultural event recognition from still images. My project goal of exploring deep learning methods applied to a real-world problem and subdomain called Looking at People or Human body or scene context analysis within Computer Vision was a challenging task. Transfer learning methods on this classification task using pre-trained Object networks improved prediction results from baseline models, especially given the relatively small dataset size for each category. The comparative study between deep earlier CNN models and very deep more recent models presented limitations in fine-tuning and adapting models to this event recognition task.

References

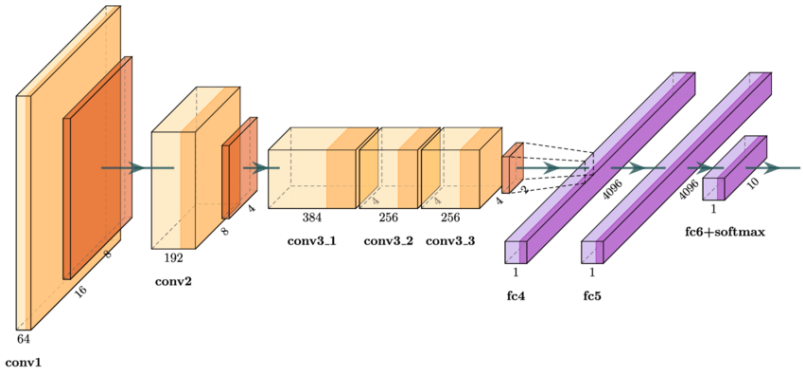
1. https://www.cv-foundation.org/openaccess/content_iccv_2015_workshops/w11/html/Escalera_ChaLearn_Looking_at_ICCV_2015_paper.html
2. https://www.cv-foundation.org/openaccess/content_cvpr_workshops_2015/W09/papers/Wang_Object-Scene_Convolutional_Neural_2015_CVPR_paper.pdf
3. https://competitions.codalab.org/competitions/4081#participate-get-data
4. https://www.kaggle.com/code/faressayah/cifar-10-images-classification-using-cnns-88/notebook
5.

Models Used

Vanilla CNN

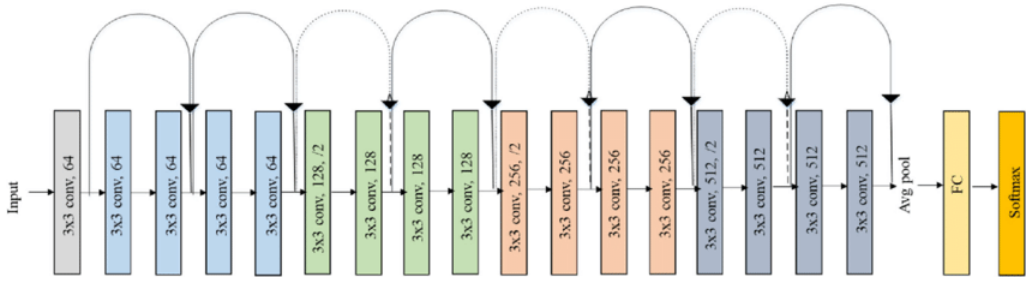
```
Vanilla CNN:
 Net(
  (conv1): Conv2d(3, 6, kernel_size=(5, 5), stride=(1, 1))
  (conv2): Conv2d(6, 16, kernel_size=(5, 5), stride=(1, 1))
  (fc1): Linear(in_features=59536, out_features=500, bias=True)
  (fc2): Linear(in_features=500, out_features=250, bias=True)
  (fc3): Linear(in_features=250, out_features=11, bias=True)
  (softmax): Softmax(dim=1)
)
```

AlexNet



```
AlexNet(
  (features): Sequential(
    (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(2, 2))
    (1): ReLU(inplace=True)
    (2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (3): Conv2d(64, 192, kernel_size=(3, 3), stride=(1, 1))
    (4): ReLU(inplace=True)
    (5): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (6): Conv2d(192, 384, kernel_size=(3, 3), stride=(1, 1))
    (7): ReLU(inplace=True)
    (8): Conv2d(384, 256, kernel_size=(3, 3), stride=(1, 1))
    (9): ReLU(inplace=True)
    (10): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1))
    (11): ReLU(inplace=True)
    (12): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  )
  (classifier): Sequential(
    (0): Dropout(p=0.5, inplace=False)
    (1): Linear(in_features=36864, out_features=4096, bias=True)
    (2): ReLU(inplace=True)
    (3): Dropout(p=0.5, inplace=False)
    (4): Linear(in_features=4096, out_features=4096, bias=True)
    (5): ReLU(inplace=True)
    (6): Linear(in_features=4096, out_features=11, bias=True)
  )
)
```

ResNet18



GoogLeNet