Multi Label Classification of Yelp Dataset

Joseph Vele 000539668
(vele.j@husky.neu.edu)

Ruoxi Pan 001490605
(pan.ru@husky.neu.edu)

1 Introduction

In the digital age, with social media platforms such as Instagram, Facebook, and Twitter, consumers take photos for nearly every occasion, especially their dinning experience. Yelp is looking to capitalize on this trend and strengthen their platform by using the images provided by consumers. The goal of this project is to construct a model that can accurately classify the multi-classes and exploit the interclass dependencies. The problem is challenging such that it differs from existing popular image classification challenges like ImageNet, which are all single-class problems.

The traditional approach to solve this problem is to use a convolutional neural network (CNN) with a binary loss function; however, this method assumes independence amongst the classes and assigns classes using a rank method based on loss. This would be erroneous considering the labels and our knowledge of the dataset. Some classes such as "ambience is classy" and "restaurant is expensive" have a clear interdependence. Current research [1] exploits these dependencies using a unified CNN-RNN (recurrent neural network) framework. The model takes an approach similar to an image-captioning problem using the CNN to project the image features onto a image embedding space and then feeding them into the RNN in order to generate a sequence via beam search. The model amplifies the performance of state of the art Alexnet on datasets such as Microsoft Coco and NUS-wide.

Albeit, an improvement over the traditional approach, this method has still fallen short at capturing small objects due to the limited discriminability of the visual features. In order to remedy this issue, we look to implement spatial pyramid pooling (SPP). Currently, SPP has only been implemented with a variety of CNN architectures with each model improving by 3%, and more importantly, it is able to improve small feature detection [2]. SPP pools the last convolutional layer into N different flattened vectors and concatenates them. By doing so, the model has N different views of the last convolutional layer and a fixed representation that can be fed into the dense layer.

Overall, we aim to prove that SPP not only improves traditional CNN image based classification tasks, but unified CNN-RNNs as well. In doing so, the following tasks have been completed.

- Implement a comparative study between a CNN and CNN-RNN framework on a less widely used dataset.
- Incorporate an SPP layer in both models, monitoring the effects on accuracy, precision, and F1 Score.

2 Related Work

The core problem of this challenge is essentially an image classification problem where photos of restaurants should indicate their labels. Previous works showed that CNN exhibited its great success in single-label image classification. In 2012, AlexNet of CNN was able to achieve a top-5 error of 15.3% [7] in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). Later on in 2014, the evolution of CNN became even more evident in ILSVRC with GoogLeNet[8] achieving a top-5 error of 6.7%. Then ResNet[9] achieved a top-5 error of 3.6% in 2015. These promising and impressive results proved that CNN performed stunningly in learning image problems.

Preprint. Work in progress.

However, being able to distinguish single label is not enough for the real world. In reality, images generally contain multiple labels that might correspond to different attributes and scenes in an image. So, it is important to model the rich semantic information and their dependencies for image classification problems. The idea is to be able to implicitly adapt the attention area in the images such that when predicting different labels, the CNN could focus its attention on different parts of the images. People have tried designing the RNN framework by adapting the image features based on the previous prediction results, by encoding the attention models implicitly in the CNN-RNN structure. As it is proven, modeling the label dependencies using a CNN-RNN framework would significantly improve the multi-label image classification accuracy. Based on this idea, we make some further modification in the CNN part by employing the spatial pyramid pooling (SPP) layer to solve the varying image size problem.

Our CNN-RNN-SPP network is inspired and relates strongly to the spatial pyramid pooling layer of CNN [1] used in image classification. Usually CNNs require a fixed input size dataset for the dense layer and traditionally this has been fixed by resizing, cropping, or scaling the image; however, the modeler risks losing important information. Therefore, for the CNN portion of our model, we look to implement spatial pyramid pooling (SPP) layer. Research has suggested these layers can amplify state-of-the-art-models such as Alexnet [3] and that multiple SPP layers can improve classification accuracy.

In addition, we have actually discovered two papers that try to tackle the same problem with this Yelp data set. The first paper introduced two approaches for the yelp photo classification problem [2,3]. It presents a design using three models: 1.PCA+ classification. 2. Transfer learning. 3. Feature extraction with ResNet. Our work differs from theirs greatly and completely. How they unified the photo sizes was by simply resizing all the photos in the training set to the same dimension of 224*224. Then by using principle component analysis (PCA), they chose only the first 600 PCA and then repeated the same process on the test set to make predictions. Next, they performed transfer learning on pretrained ResNet model, specifically on ResNet-50 and ResNet-100. Lastly, in order to solve the big gap between photo level result and restaurant level result, they developed the strategy of using only pretrained CNN as feature extractor without training. And then for each restaurant, they grouped the features of all the photos that belong to the restaurant and trained directly on the restaurant. Regarding method differences between our works, we included spatial pyramid pooling (SPP) to deal with the photo size problem while they used the PCA method. We employed AlexNet architecture while they used ResNet architecture. Although how we trained the model is completely divergent, it is interesting to see how to differently approach the same problem. It is worth mentioning that their optimal model only achieved 73%.

Next, the second paper did not attempt to directly build a restaurant label classifier. Instead, they developed a photo label classifier that would predict label scores for photos, which would then be combined and used in multiple ways to predict labels for a restaurant associated with those photos. Their photo label classifier was a model that has 9 output labels and 2 classes. In general, they would have 18 scores for a given input image. They used the CaffeNet architecture, but they made two modifications based on the yelp problem. CaffeNet had the same layers as AlexNet except for the FC8 layer because CaffeNet was designed to output for 20 classes. Hence the output layer of the CaffeNet was modified to suit the multi-output binary classification. The approach was to have 9 separate binary output nets and a 2-class loss function for each net to minimize for training. Another modification was to replace the ImageData input layer of CaffeNet with Data input layer with lmdb backend to speed up training. To sum up, approach of this paper was the contrary to ours. We used the AlexNet-SPP architecture combined with RNN-LSTM, with nothing concerning the 5. Their optimal validation accuracy was 82.7% specifically for the *has_table_service* label, and lowest validation accuracy of 55.2% specifically for the *outdoor_seating* label.

3 Method/Model

3.1 CNN-RNN

The primary models considered for this experiment will be the CNN-RNN models with and without SPP, both of which are composed of an encoder and decoder. The principal objective of the CNN-RNN

is to maximize the probability a set of labels is generated given an image:

$$\theta^* = \underset{\theta}{\operatorname{argmax}} \sum_{I,L} log(p(L|I,\theta)$$

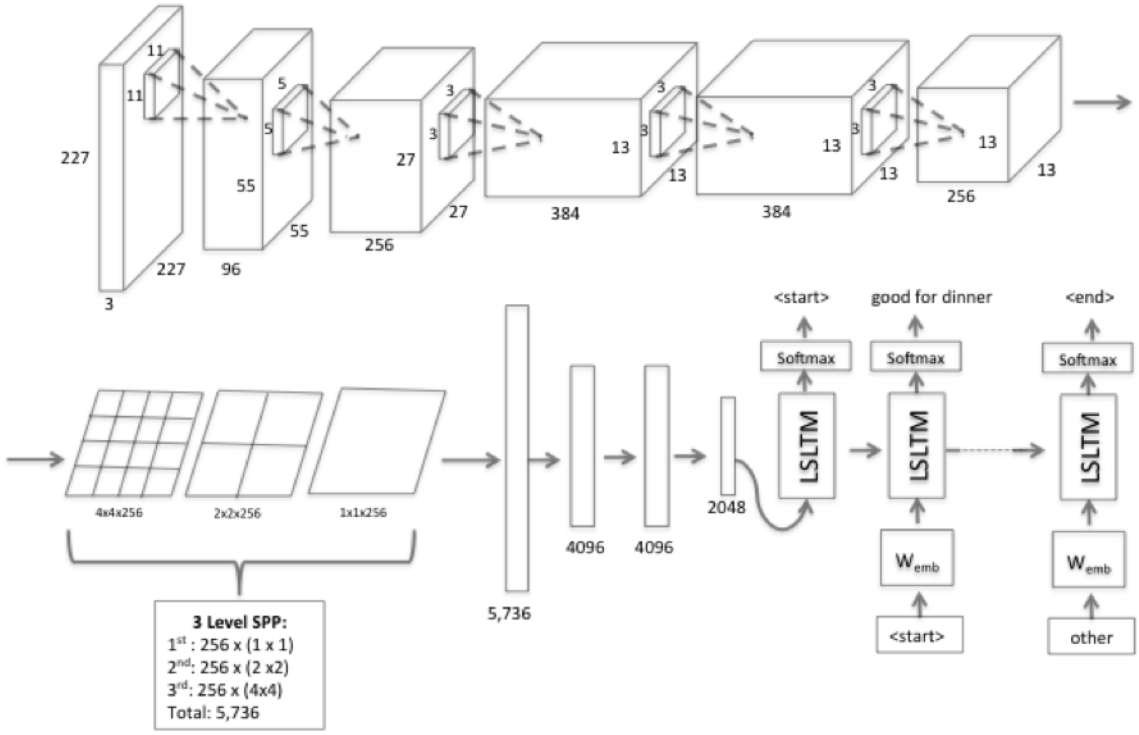
where I is the image, L is the labels, and θ is the parameters of the model. The probability of L is the joint probability over L_1, ..., L_N:

$$log(p(L|I) = \sum_{t=0}^{N} log(p(L_t|I, L_1, ..., L_{t-1})$$

where N is the number of labels assigned to a given image, and L_t is the label assigned at step t.

3.1.1 Encoder

The encoder is a CNN that leverages the AlexNet architecture; however, in order to incorporate the SPP layer, the dense layer input has been modified from 9,216 to 5,736. The SPP layer bins the input and performs a variation of max pool to produce a fixed output, resulting in n k M dimensional vectors where n is the number of levels, k is the number of filters, and M is the number of bins. For our SPP layer, we have considered 3 levels where each level pools the last convolution into 16, 4, and 1 bin(s), eventually concatenating the flattened vectors, resulting in 5,736 features. Once complete, the fixed length output feeds the dense layer, which generates an image embedding vector,v, fed as input to the Decoder/LSTM model.



3.1.2 Decoder

The decoder is composed of an LSTM. The LSTM has three gates that help extend the RNN neuron: a forget gate, an input gate and an output gate. These gates enable the LSTM to learn long-term dependence and make the optimization easier. This RNN framework has proven to be a more compact and powerful model compared with others, effectively improving the multi-label classification by employing an end-to-end model to learn the semantic redundancy and the co-occurrence dependency.[5].

The inputs of our LSTM model at time t are the image embedding vector, v, a label embedding vector, x_t, and the hidden state, h_t such that

$$l_{t+1} = LSTM(v, x_t, h_t), t \in \{0...N-1\}$$

where l_{t+1} is the predicted label. The label embedding x_t is constructed in order to project the label onto same subspace as v:

$$x_t = W_L L_t, t \in \{0...N-1\}$$

where L_t is a label encoded as a one hot vector with a size equal to the number of labels and W_L are the parameters from the label embedding matrix.

Within our LSTM model, the recurrent layer takes the label embedding of the previously predicted label, and models the co-occurrence dependencies in its hidden recurrent states by learning nonlinear functions:

$$o_t = h_o(r_{t-1}, x_t), r_t = h_r(r_{t-1}, x_t)$$

where r_t and o_t are the hidden states and outputs of the recurrent layer at the time step t, respectively, x_t is the label embedding of the t-th label in the prediction path, and $h_o(.)$, $h_r(.)$ are the non-linear RNN functions.

3.1.3 Predicted Path Sequence

The predicted path sequence can be generated in two ways: greedy or beam search. The greedy search algorithm takes the predicted label with the highest probability at each time step until it achieves a max sequence length of 11 or until <end> is returned. This method is considerably faster; however, it risks the chances of choosing a non-optimal path. For example, if the model returns the first label incorrectly, then this will continuously inform the rest of the model, eventually returning an entirely incorrect path. Beam search considers the top k most probable labels per time step for each given path. For the purpose of this experiment k is defined as 2. We monitor the labels and probabilities in separate vectors. After selecting 2 labels, the sequences will be generated recursively, continually creating paths until a sequence length of 11 is achieved or <end> is returned. By the end of the search, a multitude of prediction path sequences are generated; however, ultimately the one assigned, has the highest probability of occurring.

3.2 CNN

In order to observe the efficacy of the SPP layer on the CNN-RNN model, the two baseline models, CNN and CNN with SPP, will both maintain the same CNN architecture from the former model. The CNN will maintain the Alexnet architecture, but with an output of 9 (9 represents the number of the labels). The labels will be assigned using a sigmoid function, treating each label assignment independently. The incorporation of the SPP layer will remain the same as the CNN-RNN with the last max pool layer being replaced with a 3 level spp layer, resulting in 5,376 features.

4 Experiment

4.1 Dataset and Preprocessing

The data set used was leveraged from Kaggle and contains a total of 234, 842 images. These images were partitioned into training: 164,382 (70%), validation: 23,483 (10%), and test data: 46,977 (20%). The purpose of the validation data is for hyper parameter tuning such that it dictates which learning rates, weight decay, embedding sizes, number of hidden units, and number of layers are optimal. The unique labels for the images are: *good_for_lunch*, *good_for_dinner*, *takes_reservations*, *outdoor_seating*, *restaurant_is_expensive*, *has_alcohol*, *has_table_service*, *ambience_is_classy*, and *good_for_kids*. For the CNN-RNN models <start>, <padding>, and <end> have also been included. Given the pretrained parameters, all images have been normalized considering the mean and standard deviation of the ImageNet dataset. Also, all images were resized to 227 x 227 for training, including the SPP models. This approach has been shown to maintain the benefit of SPP, while exploiting the benefits of training with fixed size images[3]. For testing purposes the images remained the same size for the SPP models using a batch size of 1.

4.2 Hyper parameter Tuning

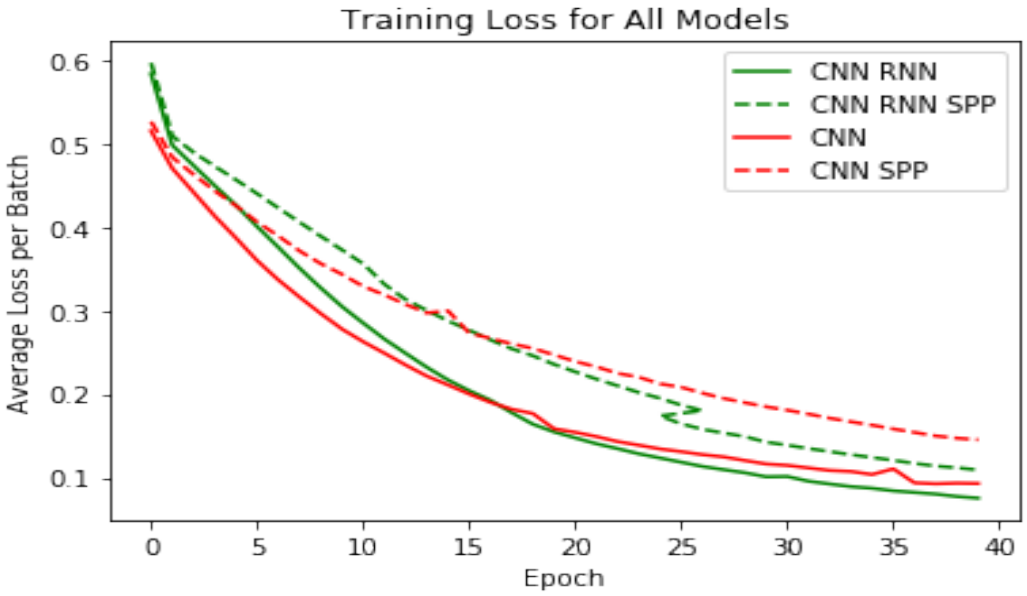
Previous studies have concluded that hyper parameter tuning for the SPP models is not required and that the benefits should be prevalent with the same parameters from the base models. Because of this, we consider only tuning the CNN and CNN-RNN. Both were trained employing random search with the models trained for one epoch and then tested against the validation data set. All models used an Adam optimization algorithm with the CNN-RNN models considering cross-entropy loss and the CNN models considering a binary logit loss. The model that minimized the validation loss was chosen as the optimal model.

For the CNN- RNN, 64 random trials were considered. The number of layers was drawn from a set of [1,2,3] with uniform probability. The image and label embedding sizes were chosen from a set of [256,512,1024,2048] with a uniform probability. The learning rate and weight decay were both drawn geometrically with the former choosing from .0001 to .01 and the latter choosing from 3.1e-7 and 3.1e-5. Drawing geometrically from a set A and B entails drawing uniformly in the log domain between log(A) and log(B), exponentiating to get a number between A and B. The optimal model had the following hyper parameters: Image Embedding Size = 2048, Label Emedding Size = 2048, Layers = 1, learning rate = 0.00025, and weight decay = 1.71e-06.

In regards to the CNN model, only 8 random trials were considered, since the learning rate and weight decay were the only parameters. Both hyper parameters were drawn geometrically from .0001 to .01 and 3.1e-7 and 3.1e-5, with the optimal values being .0002 and 1.73e-5.

4.3 Results

All models were trained for a total of 40 epochs over the course of a few days using GPU. The losses are summarized below:



Interestingly, the CNN-RNN model's initial loss is higher than that of the CNN model; however, the former learns faster, eventually achieving a smaller average loss per batch.The most evident and surprising information from the plot is that the SPP models learn significantly slower, potentially implying that SPP is a form of regularization. Unfortunately, per the results on the test data, SPP did not perform better in terms of accuracy, precision or the f1-score; however, it seems the layer had similar effects on both the CNN and the CNN-RNN.

Model Performances				
Model	Accuracy	Precision	Recall	F1-Score
Alexnet	0.78	0.80	0.79	0.79
Alexnet-SPP	0.75	0.75	0.79	0.77
CNN-RNN	0.77	0.78	0.79	0.78
CNN-RNN-SPP	0.71	0.70	0.80	0.74

These results would suggest that there may be some limitation to incorporating an SPP layer and that it potentially should be viewed more as an image processing technique or hyper parameter. The idea from previous papers that suggest it improves all CNNs, may require further research. Below we explore further to see a sample image captioned by all the models.

Label: Good for dinner, takes reservation, expensive, alcohol, table service, classy
CNN : Good For Lunch, out door seating, good for kids
CNN SPP: Good for dinner, takes reservation, expensive, alcohol, table service, classy
CNN-RNN: Good For Lunch, Outdoor seating, good for kids
CNN-RNN SPP: Good for dinner, takes reservation, outdoor seating, alcohol, table service, good for kids

Interestingly, both base models under-perform here achieving 0% accuracy . This would imply that SPP does aid in some cases; however, at the cost of more obvious paths. Another aspect of the data we inspected is the accuracy in terms of the complete paths. We define a complete path as a predicted label assignment that exactly matches the true labels. As expected, the CNN-RNN is able to identify complete path sequences better than the traditional CNN. However, we don't see much influence from the SPP layer.

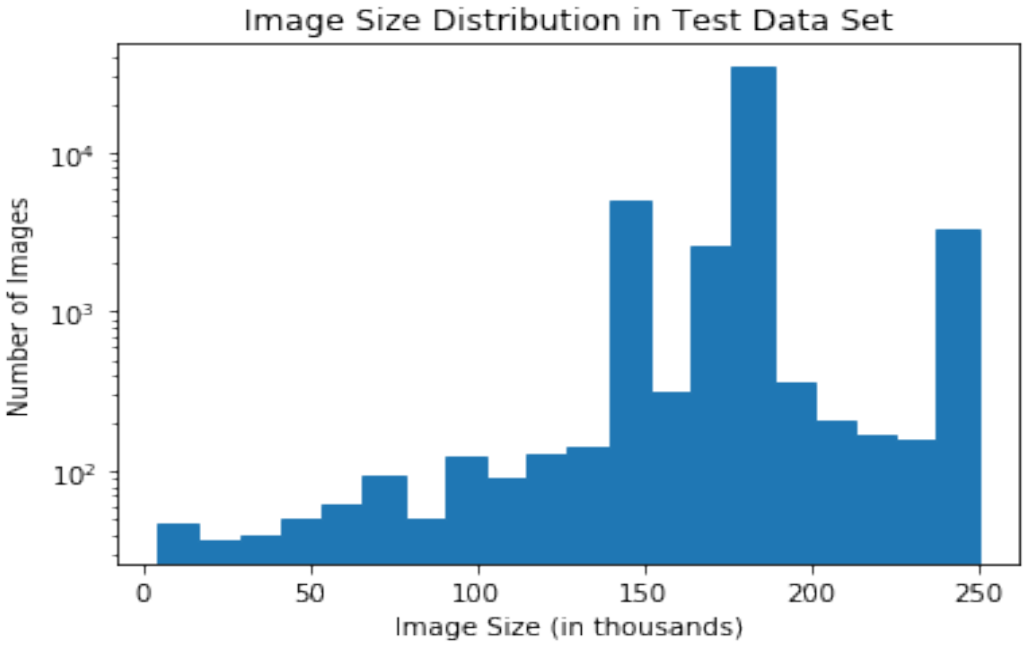
Complete Path Performances	
Model	%
Alexnet	24%
Alexnet-SPP	18%
CNN-RNN	26%
CNN-RNN-SPP	22%

5 Conclusion

In this experiment, we have considered two models, a CNN and a CNN-RNN, with the former having a history of better performance when coupled with a spatial pyramid pooling layer. We observed that in the case of our experiment, that the CNN-RNN with an SPP layer behaves similarly, albeit both models achieved less accuracy than their base counterparts. This implies that maybe further research is required regarding SPP.

The areas for further investigation include the number of bins, number of levels, and the limitation in image size variability. Regarding the latter, it was apparent that some images may be too small for an SPP layer such that some of our images would produce an error because by the final convolution,

they were too small and unable to be partitioned into bins. It is also possible the number of bins and levels are a function of the variability of the sizes of the images such that an optimal bin number will properly reflect the max and min size of the image data set. Below is a distribution regarding our data:



Overall, numerous things have been learnt over the course of this experiment. The most obvious point being that not all existing research is reproducible. Although our data set was different, we expected our theory to show some improvement to the baseline models, at least pertaining to the CNN, since there are many published research papers on it. Secondly, all data sets behave differently and it should not be expected to follow similar patterns as other popular data sets such as MNIST, hyper parameters of a model and image processing differs across data. Lastly, research is never complete. Experiments constantly lead to more questions and what ifs. Its a key part of the machine learning field that makes us further develop solutions to complex problems. By failing, we are able to generate better problems.

References

1. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, pp. 346–361, 2014 : https://cmt.research.microsoft.com/NeurIPS2018/

2. Cyril Goutte and Eric Gaussier. A Probabilistic Interpretation of Precision, Recall and F-score, with Implication for Evaluation. In Proceedings of the European Colloquium on IR Resarch (ECIR'05), LLNCS 3408 (Springer), pp. 345–359: https://link.springer.com/chapter/10.1007/978-3-540-31865-1_25

3. Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In Advances in Neural Information Processing Systems 25 (NIPS 2012) https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks

4. J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu. CNN-RNN: A Unified Framework for Multi-label Image Classification, 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016. https://arxiv.org/abs/1604.04573

5. R. Roy. Yelp Restaurant Photo Classification. http://cs231n.stanford.edu/reports/2016/pdfs/014_Report.pdf

6.H. Peng, Y. Ding, and B. Huang. Yelp Restaurant Photo Classification. http://cs229.stanford.edu/proj2017/final-reports/5244133.pdf

7.Alex Krizhevsky, Ilya Sutskever, and Geoffrey E.Hinton. 2012. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12), F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Vol. 1. Curran Associates. https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

8.Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9). https://arxiv.org/pdf/1409.4842.pdf

9.He, K., Zhang, X., Ren, S., Sun, J. (2016). Deep residual learning for image recognition. In Proceed- ings of the IEEE conference on computer vision and pattern recognition (pp. 770-778). https://www.cv-foundation.org/openaccess/content_cvpr_2016/papers/He_Deep_Residual_Learning_CVPR_2016_paper.pdf

Github Repository: https://github.com/velej/Yelp_Image_Classification