Final Report: The Sorting Hat

An Intelligent Waste Item Classification Software

Word Count: 1969

Bob Cui 1003323281, Marko Huang 1003163140

December 2, 2018

1 Introduction

While most people want to contribute to the local recycling program for better waste management, few are willing to take that extra step and carefully read the labels on the waste bins. As a result, many people rely on intuition, and waste items are disposed in the wrong bin due to negligence. Even when one is willing to make an effort to dispose the waste correctly, they may find the labels confusing or unintelligible, and hence not able to make a correct decision in a short amount of time.

The Sorting Hat is an intelligent waste item classification program which advises the user the correct bin a waste item should go in. The program takes in an image of the waste item and a short description, and outputs one of the four categories - landfill, containers, paper, or coffee cups. A device consisting of a camera, microphone, and the proposed software would be placed next to a waste bin. Upon encountering a waste bin equipped with *The Sorting Hat*, the user will present the waste item they wish to dispose, and the program will take a picture, ask for an optional verbal description of the waste item in question, and then suggest a bin for the individual to throw in. This greatly reduces the complexity of the task and eliminates any confusion therein.

2 Overall Structure of Software

A block diagram indicating the logic flow of the software is shown in Figure 1. The software is by default in its standby mode, waiting for a user to present a waste item in front of the designated area. When a sensor detects an object in front of the camera, two parallel branches (orange and blue) are executed.

In the orange path, a picture of the waste item is taken by the camera and fed into a Convolutional Neural Network (CNN). The model predicts a distribution of probabilities of the image belonging in each of the four classes. In the blue path, the program accepts a phrase as input, and also outputs a probability distribution based on the vector representation of the phrase.

The results from the image model and language model are combined to obtain an overall prediction of the most likely bin for disposal. With the ensemble of two independent models, we believe that the program will be able to make more reliable predictions than a single model would.

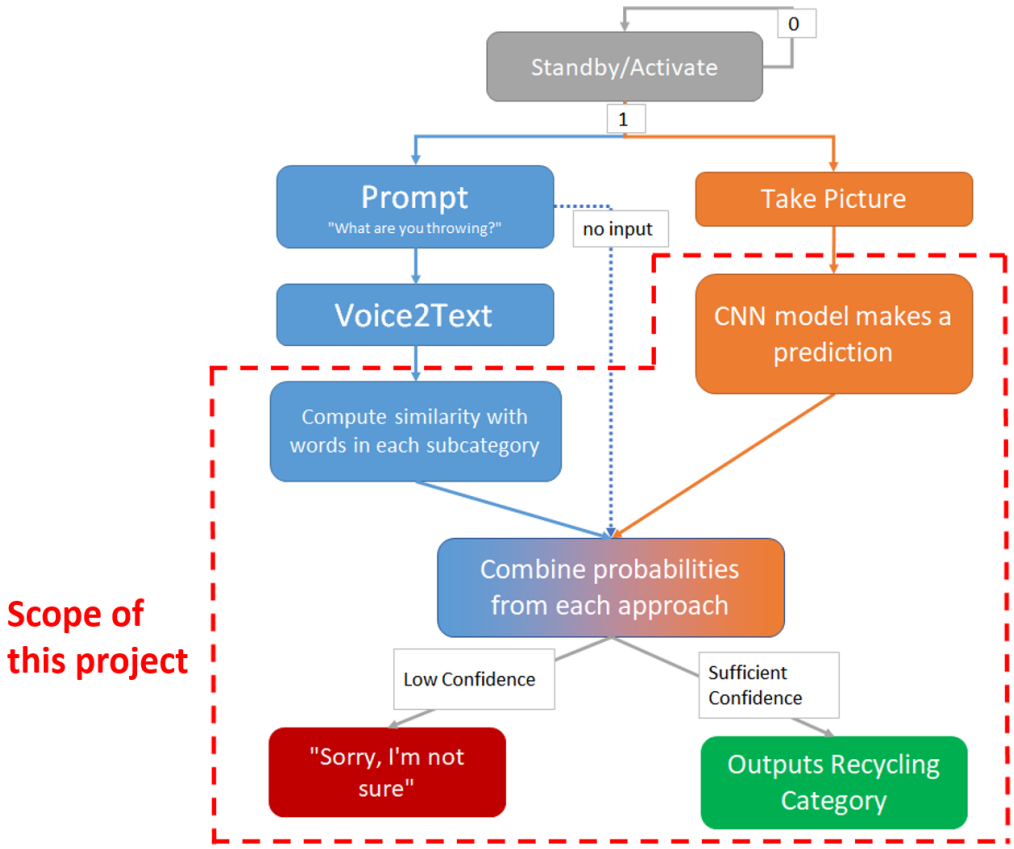

Figure 1: Block Diagram of Overall Structure

3 Source of Data

3.1 Image Classifier

The pictures used to train and validate the CNN were obtained through (a) the online visual database ImageNet (IN) [1], with labels indicating the object in the image, and (b) Google Images (GI), to supplement the training data in cases where a common waste item does not appear on ImageNet.

Our training dataset consisted of 11 subcategories of waste items belonging to four waste bins (classes): landfill, containers/recyclables, paper, and coffee cups. The source of images in each subcategory is summarized in Table 1. After discarding the corrupted and incorrectly labelled images, the remaining images were augmented using techniques shown in Figure 2 to at least 8000 samples per category, in order to improve the model's ability to classify items under different angles, sizes, and qualities. The images were also scaled to 128 by 128 pixels during this process, because this resolution balanced sufficient visual information and reasonable computation speed.

Table 1: Training and Validation Data

Class	Subcategory	Source	# images
Landfill	Food Waste	IN	1073
	Plastic Bags	IN	527
	Snack Packages	GI	246
	Fast Food Wrappers	GI	254
Containers	Plastic Bottles	IN	610
	Pop Cans	IN	1043
	Glass Bottles	IN	1016
	To-go Boxes	GI	281
	Juice Boxes	GI	102
Paper	Paper/Newspaper	IN	1014
Coffee Cups	Coffee Cups	GI	432

Figure 2: Data Augmentation Techniques Used

3.2 Natural Language Processing (NLP) Unit

For the NLP unit of the software, the word vectors were Word2Vec embeddings pre-trained on the Wikipedia corpus. The words and phrases used to train the neural network came from the City of Toronto Waste Wizard [1], which contained words to describe common

[1] https://www.toronto.ca/services-payments/recycling-organics-garbage/waste-wizard/

household items and the bin they should be disposed in. The list was filtered to exclude items that were oversized or otherwise unlikely to be disposed on campus. The list was then augmented manually to include synonyms of some words and other ways users may describe certain items, such as including the specific brand name ("Starbucks cup"), or prepending an irrelevant word ("bag of chips", "small bottle"). Due to the subtlety of text labels, we were unable to automate the augmentation process and the train/validation dataset consisted of 154 samples in total.

4 Machine Learning Models

The Sorting Hat consists of two models: an image classifier and a natural language processing unit. For the image component we constructed one CNN model from scratch, and one model applying transfer learning with the Inception-v3 [2]. Originally, the output was one of 11 classes of images which we identified as being the most common in campus waste disposal. However, we found later that if we instead allowed the neural net to learn the final mapping and output directly 4 classes, the performance was better. This suggested that the model was able to capture additional information in the process of mapping the images directly into 4 categories.

4.1 Training the Image Classifier

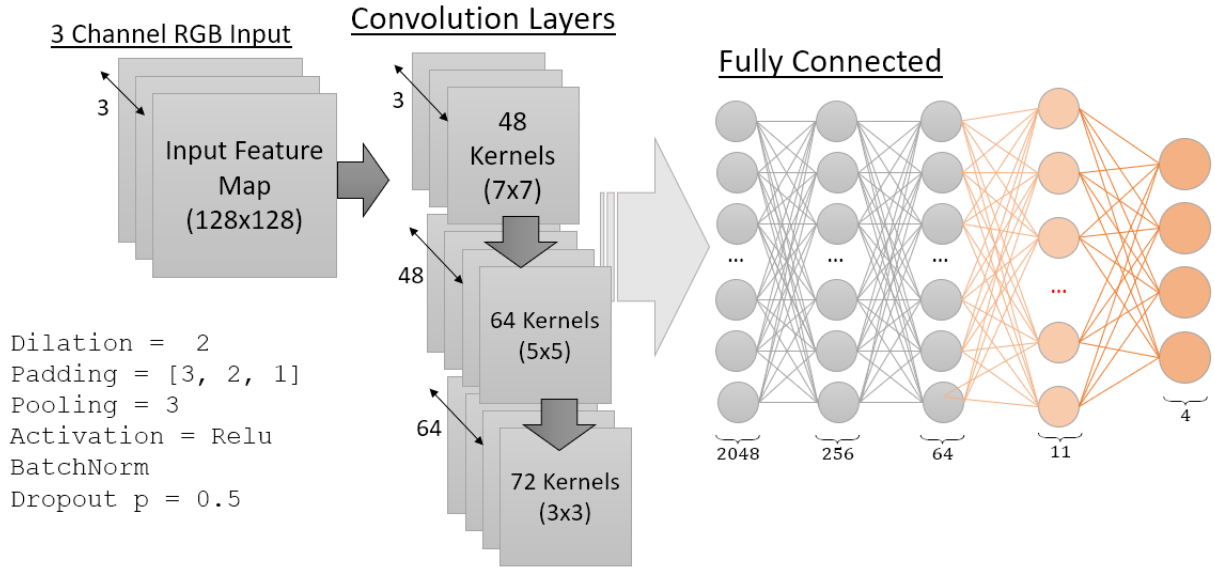

Figure 3: CNN Model Structure and Parameters

Baseline CNN

The model we trained from scratch consisted of 3 convolution layers and 5 fully connected layers as shown in Figure 3. The convolution layers were a scaled-down version of AlexNet [3], with increasing kernel numbers (48, 64, 72) and decreasing kernel sizes (7x7, 5x5, 3x3) on each succeeding layer. The intuition was that the larger receptive field in the second and third layers should be compensated by a smaller kernel size, and the increasing number of kernels allowed the neural net to target more complex features that were characteristic of each waste item. All layers were passed into a pooling function with a pool size of 3x3 for down-sampling. This shrank the sizes of the output feature maps and mitigated the influence of the noise and background of the images. To combat overfitting, dropout layers with $p = 0.5$ were added after every fully connected layer, and the final model plateaued at around 78% accuracy.

Figure 4: Training/Validation Accuracies

Transfer Learning - Inception-v3

In attempt to further improve the classification accuracy, we trained an Inception-v3 model that was pre-trained on the ImageNet database. Freezing all its parameters, we replaced the last fully connected layer with four output neurons, and trained this specific layer using our training data.

The Inception-v3 model consists of inception layers which apply a combination of kernel sizes at once and concatenate the results together into a large activation layer before feeding onto the next layer. In addition, the Inception-v3 model also uses auxiliary classifiers to help with convergence and combat overfitting. These are selected hidden layers in the model

that have independent fully connected layers used to predict the labels. These auxiliary networks are disregarded in inference mode, and only the final fully connected layer outputs the prediction.

Table 2: Model Validation Accuracies

Model Type	**Highest Validation Accuracy**
Baseline CNN (11 class)	78.2%
Baseline CNN (4 class)	79.1%
Inception-v3 (11 class)	87.1%
Inception-v3 (4 class)	88.3%

As shown in Table 2, transfer learning was able to outperform our baseline CNN model, which was expected because it was a more sophisticated network. In addition, Inception-v3 was pre-trained on the entire ImageNet database, making it more capable of generalization.

4.2 Natural Language Processing

For the NLP unit, the limited amount of data presented a significant challenge in training the neural network. In fact, even with a fully connected network of no hidden layers, the number of parameters would still exceed the number of training data. Therefore, to minimize the impact of overfitting, we restricted the model to a small multilayer perceptron with one hidden layer and one Dropout layer. In the end, the training accuracy was 92% and the highest validation accuracy was 75%. In order to examine whether a neural net was the best choice, we also implemented a baseline classifier which simply extracted the maximum cosine similarity between the target word and a list of predefined keywords. This model only achieved an accuracy of 58%. This difference showed that the neural network was able to warp the word vector space and capture the important relationships beyond vector similarity.

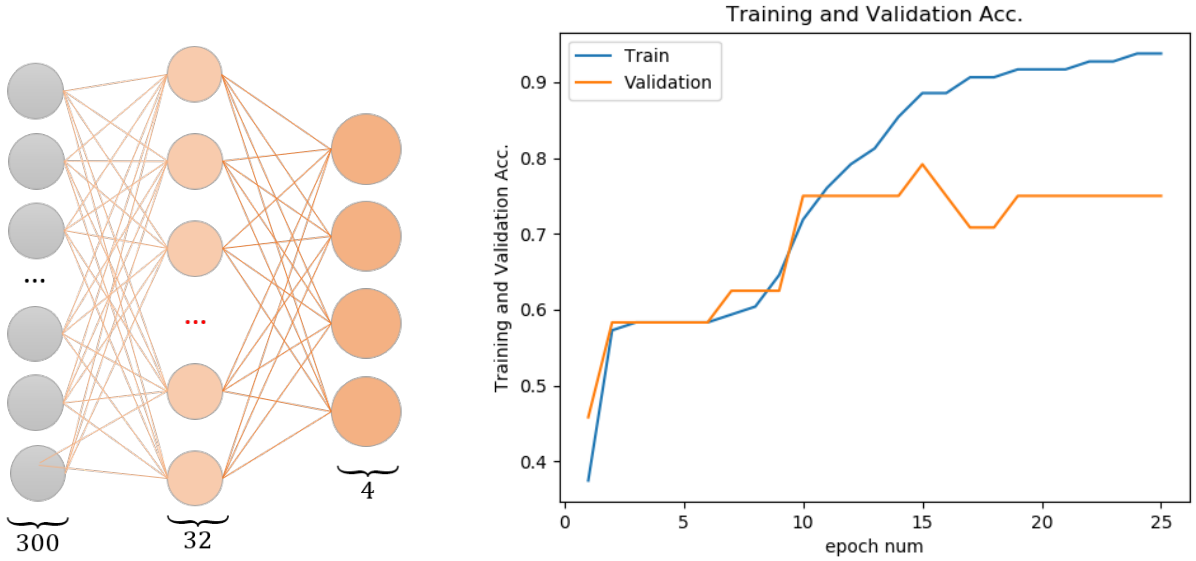

Figure 5: **left:** NLP Model Structure **right:** Training/Validation Accuracies

5 Testing

To test the two models we have trained, we took pictures of actual waste items to simulate the image inputs *The Sorting Hat* would actually receive in its implementation. Each picture was accompanied by a short description of the item to simulate the text input. A total of 89 pictures were taken and labelled manually.

To find the optimal weighting of the image classifier and NLP unit, the probability outputs from the two models were combined with a weight of α and $(1-\alpha)$ respectively to give a final prediction, where α was between 0 and 1. As seen in Table 3, the CNN models performed poorly on the test data – achieving 38.2% test accuracy for the baseline CNN and only 57.3% for the Inception-v3 model. The highest test accuracy occurred when the combined model put 90% weight on the NLP unit and 10% on the CNN.

Table 3: Combined Results with Inception-v3 CNN and Neural Net NLP

CNN	**NLP**	**Combined Accuracy**	**CNN**	**NLP**	**Combined Accuracy**
0.0	1.0	77.53%	0.6	0.4	57.30%
0.1	0.9	78.65%	0.7	0.3	57.30%
0.2	0.8	69.66%	0.8	0.2	57.30%
0.3	0.7	64.04%	0.9	0.1	57.30%
0.4	0.6	59.55%	1.0	0.0	57.30%
0.5	0.5	57.30%			

To investigate the reasons why the test accuracy of the image classifier was significantly lower

than the validation accuracy, we looked into the test and validation dataset and hypothesized several reasons:

Background differences: The training data for coffee cups consisted almost exclusively of white backgrounds, while all other 10 categories did not have solid backgrounds. This may have cued the model to predict coffee cups whenever it sees an image with white background. Since there were no photos with white background in the test set, all the coffee cups pictures had poor test accuracy.

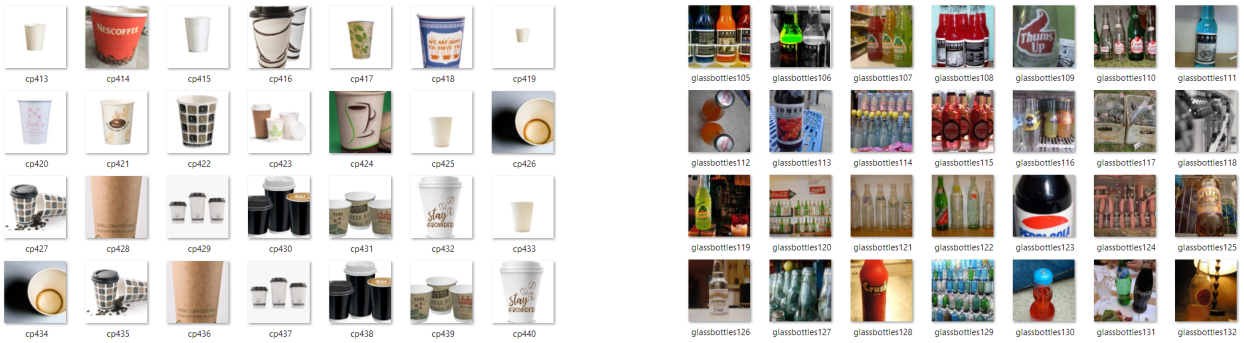

Figure 6: **left:** scraped coffee cup images **right:** scraped bottle images

Biases within dataset: Images scraped from Google Images tended to be more like stock photos – with vibrant colors and clean focus on the object. Images from ImageNet were more natural looking, but most were closeups of the object (i.e. taking up the entire frame). The neural net may have caught on these subtleties and wrongfully recognized these as features for a particular waste item.

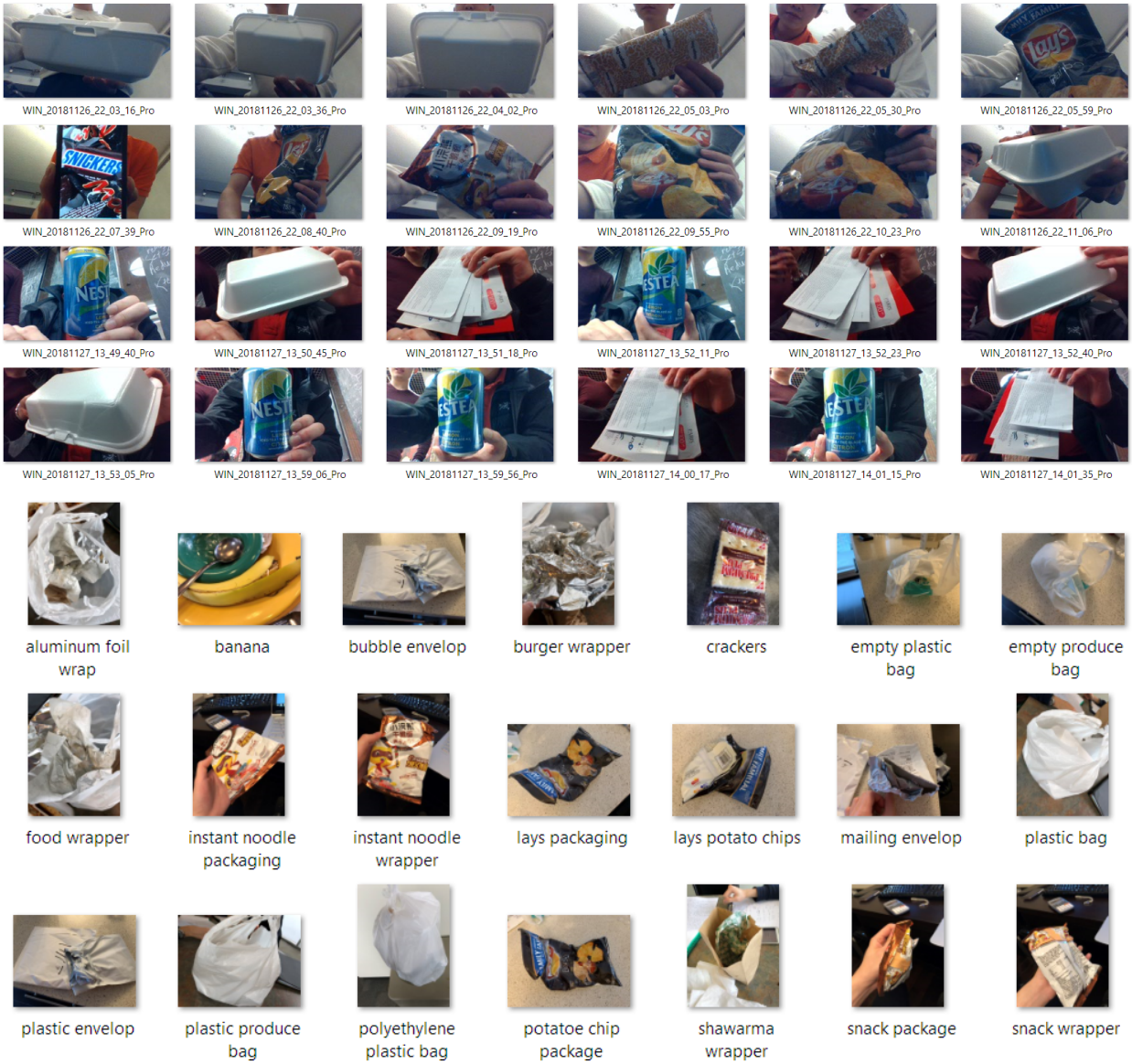

Figure 7: Example test pictures

Non-waste-looking images: Pictures in the test set were actual waste items, taken casually with no consistency in lighting and size as shown in Figure 7. This was done to simulate actual usage of the software as accurately as possible. The training dataset, however, consisted of brand new items before they became waste. This may have limited the model's ability to classify actual waste items present in the test set.

6 Key Learnings

We gained an appreciation for the importance of data acquisition and processing to the training of the model. Through this project we've discovered that the model's ability to successfully generalize largely depended on both the quality and quantity of waste item images. For example, since most coffee cup images were unbranded stock photos, the model had problems generalizing to branded coffee cups that were present in the test data. To improve the test accuracy, more intra-class variance (different looking cups for example) could have been introduced.

If we had more time to do the project again, we would have collected more photos similar to the test data and used them as the training data for the image classifier. This is so that the classifier generalizes better on actual waste items instead of nice-looking stock photos.

We would have leveraged transfer learning in a much earlier stage, since its result was far more convincing than our own CNN model and the training time was also shorter. We would have experimented with different state-of-the-art models, such as AlexNet, ResNet and VGG to see which one works the best for our application.

Coding practices are also part of some key takeaways, since this has been our first experience working on a multi-person programming project. Structuring code early on definitely makes up for the time it takes later on during the integrating stage. Consistent variable names and data loading scripts would have made the model much easier to build onto. Unit testing should take place on every sub-module before everything is put together which would greatly increase the efficiency of debugging.

7 Ethical Issues

Our proposed recycling classification software is not directly intrusive to any industry or job markets. However, we acknowledge potential liability issues when an erroneous output results in a human disposing a recyclable item in the landfill bin, or vice versa. Is this the human's fault or the programs fault? The choice of which bin to throw the item in is ultimately one of the human's, but false predictions will inevitably happen due to errors on the classifier's side.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

[2] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CoRR*, vol. abs/1512.00567, 2015. [Online]. Available: http://arxiv.org/abs/1512.00567

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf