A Comparison of Random Forests and Dropout Nets for Sign Language Recognition with the Kinect

Anonymous Author(s)
Affiliation
Address
email

Abstract

Random Forests (RF) and Dropout networks are currently two of the most effective machine learning algorithms available. However, so far a study directly comparing the accuracy of both on the same dataset has not been performed. We hope to fill this gap by testing the classification accuracy of both of these ensemble methods on a novel dataset of American Sign Language (ASL) hand signs collected using the Microsoft Kinect. Results show that dropout nets achieve a higher gesture classification accuracy, particularly as the number of classification labels increases. Further, a neural network trained with dropout outperforms the same net without dropout, demonstrating the effectiveness of the technique. Individual gesture recognition accuracy as well as computation times for both algorithms will be presented.

1 Introduction

Before last year, Random Forests (RF) may have been the most effective machine learning technique. A wide variety of problems have benefitted from the application of RFs, including identification of DNA-binding proteins, classification of aerial images, predicting the population distributions of bird, mammal, and vegetation species, language modeling, diagnosis of Alzheimer's disease, and recognition of handwritten digits [1]. They have also been employed commercially with great success, allowing the Microsoft Kinect to recognize the spatial location of joints of a human body [2]. Random forests have been praised for their computational speed, simplicity, and their ability to handle large datasets and large feature spaces [3]. In 2006, a large-scale study compared the performance of several of the most popular algorithms on binary classification with a variety of datasets, using empirically chosen optimal parameter settings [4]. The results indicated that Random Forests outperformed other methods across measures such as accuracy, precision, recall, squared error, and area under the ROC curve. While Artificial Neural Networks (ANN) also obtained impressive performance, the authour referred to the results of the study as "a clean sweep for ensembles of trees" [4].

It has been hypothesized that the strength of random forests comes from aggregating the decisions of many classifiers [5]. Random forests use both *bagging* (bootstrap aggregating) and random feature selection to create many diverse hypotheses about how to classify the training data correctly. This *ensemble* of trees is able to overcome the shortcomings which plague many learning techniques. For example, in situations where the hypothesis space is large and there is too little training data, several hypotheses may be consistent with the set of training examples, but it is impossible to determine which will be better at predicting future data. Allowing several classifiers to cover this set of consistent hypotheses and vote on the classification of future data makes ensemble methods robust [5]. Similarly, an individual classifier which uses heuristics may end up with a poor estimation of the true function it is attempting to learn. For example, gradient descent (a learning method used in training neural networks) can get stuck in local minima [5]; an ensemble, however, could solve this problem.

Using this line of reasoning, in 2012 Geoffrey Hinton proposed a technique called *dropout* that allows a feed-forward artificial neural network to behave like an ensemble of networks [6], enhancing the already powerful, biologically inspired, neural network methodology. Hinton's dropout technique is important because it can reliably improve the performance of feed-forward ANNs [7], which have already been employed in a wide variety of applications such as bankruptcy prediction, medical diagnosis, speech recognition, and the recognition of handwritten digits [8]. Since it has been shown that a neural network can approximate any function [8], and many advanced applications currently make use of neural networks [9], increasing the accuracy of this method advances the state of the art in Machine Learning.

The idea behind dropout is simple: as each training example is presented, hidden neurons in the net randomly "drop out" (and are not trained) with probability 0.5 [6]. Rather than training a single neural net with N hidden neurons, dropout essentially trains 2^N different networks, each on a subset of the data. It is essentially an extreme form of bagging, with the additional constraint that the networks must share the same parameters [7]. Dropout prevents the hidden neurons from learning features that only function in the context of other working neurons, prevents overfitting, and allows the net to model multiple different hypotheses about how to classify the data [6]. It has been described as a "model averaging technique", because all neurons are used during testing, and dividing the result by two gives the exact equivalent of the geometric mean of the predictions of all 2^N networks [7]. In keeping with the methodology of ensemble techniques like random forests, dropout can be further improved by randomly omitting 20% of the input features at each training step. Using dropout, Hinton was able to achieve groundbreaking results on such popular datasets as MNIST, which contains 28x28 pixel images of handwritten digits [6]. Other researchers have since employed dropout with deep convolution networks to set the state-of-the-art in image classification [10].

The intent of this research is to directly compare the performance of random forests against dropout nets, on a novel dataset collected using the Microsoft Kinect. The Kinect is a peripheral component used to obtain 3D depth data about the user. By projecting a pattern of infra-red light and using a corresponding infra-red sensor to detect distortions in the pattern, the Kinect can estimate the distance of nearby objects [11]. Not only does it hold the Guinness World Record for the fastest selling consumer electronic device, but the Kinect has been used extensively in computer science research [11]. Because it allows the user to interact with a computer without the need to hold or carry a peripheral device, it has attracted Human Computer Interaction (HCI) researchers who hope to create the elusive Natural User Interface (NUI) [11]. These researchers are pursuing methods of manipulating interface objects using only motions or hand signs, which leads naturally to research on gesture recognition with the Kinect. Computer vision research has also made use of the Kinect, because of its ability to easily perform background subtraction, and the invariance of the collected depth images to changes in illumination [3]. This makes it an excellent tool for performing hand gesture recognition, where the difficulty of recognizing hands of varying skin tones and distinguishing them from the background has made camera images problematic [12].

This research will address recognizing the hand signs of the American Sign Language (ASL) alphabet from depth images collected with the Kinect. Sign language recognition can be a difficult problem, because there is a great deal of individual difference in hand shape, size, dexterity, and signing style [3]. However, both neural nets and random forests have previously been applied successfully to ASL recognition. A two layer feed-forward neural network implementation achieved over 99% accuracy recognizing ASL from camera images, but used a significant amount of image preprocessing, including wavelet decomposition, application of the Sobel operator, and a genetic algorithm to select the best transformation technique from gamma correction, Laplacian enhancement, and a variety of filters [13]. Therefore these results cannot be used as an indication of the ASL recognition accuracy of neural networks alone, nor the accuracy expected with Kinect depth images [3]. Previous research on recognizing ASL from Kinect depth images achieved an accuracy of 69% using random forests, with a dataset of 48,000 images collected from 5 subjects [3]. Since random forests and neural networks have been usefully applied to the problem of sign language recognition, we felt that this problem would make an ideal testing ground to compare the accuracy of both.

2 Method

2.1 Data collection and Preprocessing

We collected data from a total of 17 participants (8 female) taking a graduate level course at the University of British Columbia. Subjects were asked to sit in front of the Kinect and form each ASL symbol with their right hand for one minute, as the Kinect captured and saved depth images of the hand at a rate of ten frames per second. We also asked that subjects rotate and tilt their hand while maintaining the pose, to obtain a good variety of viewing angles (similar to the methodology discussed in [3]). It should be noted that there were many partial occlusions of signs during collection, and that the dataset should be considered noisy. Since data collection is time consuming, we limited the study to ten ASL signs: '2', '7', '5', 'A', 'C', 'H', 'I', 'L', 'Q', and 'Y' (pictured in Figure 1). We collected a total of 6000 images of each sign, resulting in a final dataset of 60,000 images of 10 signs. We chose this number based on the popular MNIST dataset of the same size, which has been used to demonstrate the effectiveness of random forests, neural nets, and dropout nets [6].

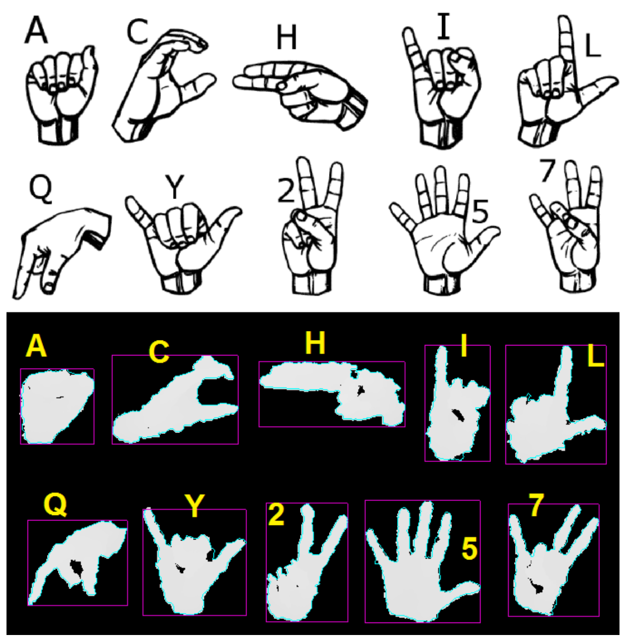

Figure 1: ASL signs used in the study (above) and the corresponding Kinect depth images (below)

In order to obtain the depth images from the Kinect, we used an open source library called OFxKinect, part of a larger package of C++ libraries known as Open Frameworks [14]. The depth information is extracted as a 256x256 grayscale image called a *depth map*, in which pixel values range from 255 (the closest possible depth) to 0 (farther away than the Kinect can sense). Performing background subtraction using the Kinect is trivial; we simply classify any points more distant than a given threshold as background. The threshold was chosen to be approximately arm's length away from the Kinect so that only hands would be part of the foreground, but it could also be set dynamically during data collection using a keyboard command. Background subtraction results in images of a hand silhouetted in white against a black background (see figure 1). These images were shown to the user during collection to allow adjustment of the hand pose in real time.

Since image preprocessing has been shown to be critically important for this type of problem [13], we took steps to standardize the depth images. An obvious requirement was extracting only that portion of the depth map which pertained to the hand sign, since when a person is standing three metres away from the Kinect, the hand occupies a region that corresponds to less than 64x64 depth pixels [15]. Open Frameworks provides the ability to extract the contours of an image, and the *bounding box* which completely contains those contours. We used the OFxCVContour library to find the precise location of the hand, and extract the bounding box (the pink lines shown in Figure 1). We added padding around this bounding box to form a region with a square aspect ratio, which we then scaled down to 32x32 pixels and saved to a file. Thus, we obtained a final dataset of 60,000 32x32 greyscale images of silhouetted hands.

During training and testing, we were careful to ensure that gestures from the same individuals did not occur in both the train and test set. In initial tests, we found that when the data was randomly shuffled in such a way, we achieved approximately 99% accuracy. We felt this was an unrealistic result, since in practice a gesture recognition system would likely be used to recognize the gestures of novel users, rather than those who had provided training data. Therefore when creating the train and test sets, we ensured that all types of gesture were split evenly across both, but the images within one gesture were kept in the original order obtained when collecting the data.

2.2 Random forests

Our implementation of random forests for classifying depth images closely followed that used by Microsoft to perform skeletal joint recognition with the Kinect [2]. We treated each pixel in the image as a feature, and randomly selected the features that could potentially be used for classification in each node of each tree. As in [2], we found that a forest of only three deep trees performed best, although we did test the effects of varying this parameter (see Section 3).

Our random forest implementation utilized a python library of machine learning scripts called Milk [16]. All accuracy results were obtained using two fold cross validation. Tests were run using an Amazon Web Services (AWS) Elastic Cloud Compute (EC2) High-CPU-Medium (c1.medium) instance, which has 1.7GB memory, and two virtual cores which each have 2.5 EC2 compute units. An EC2 compute unit is equivalent to a 1.0-1.2 GHz 2007 Opteron or 2007 Xeon processor [17].

2.3 Dropout nets

Our implementation of dropout nets is based on the code generously provided by Misha Denil at [18], which utilizes a python library designed for large scale scientific computing called Theano [19]. The code has been modified to work with our dataset, and the data was also modified to be within the range $[0, 1]$ in order to train the net. The neural net architecture used for the majority of the tests consists of a 32x32 input layer, with one input neuron for each pixel in the depth image. There are two hidden layers, each with 1200 hidden neurons, and one output layer with an output neuron for each of the 10 classification labels. This 1024-1200-1200-10 architecture was based on that described by Hinton for applying dropout to the similar MNIST dataset [6]. However, the performance of neural networks can be sensitive to this type of architectural design choice [4], so we also tested the results of two additional architectures proposed in [6]: 1024-800-800-10, and 1024-2000-2000-10. Greedy layer-wise training was used to deal with the multiple hidden layers, as in [20]. All networks were trained using the backpropagation algorithm for 500 epochs.

Tests for dropout nets were initially run using the same type of AWS instance (c1.medium), but tests with more than 6 gestures exhausted the available memory on this server. Therefore we were forced to use an alternative high-memory AWS instance (m1.medium), which has 3.75GB of memory, but only one virtual core with 2 EC2 compute units. For this reason, the times obtained for tests with greater than 6 gestures are not comparable to those of random forests, although they took nearly twice as long and clearly had a much higher memory footprint.

3 Results

3.1 Random Forests vs. Dropout

Our results indicate that dropout nets achieve a higher accuracy on this dataset. Figure 2 shows the accuracy of both methods as the number of gestures being classified increases from two to ten. When all ten gestures were included, our random forest implementation achieved an accuracy of 67.27%, comparable to the 69% reported in another study of ASL recognition using random forests applied to Kinect data [3]. Therefore we have reason to believe that the accuracy reported for random forests is a realistic result for this problem. Dropout nets were able to surpass this by a significant margin, achieving an accuracy of 81.34% when classifying ten gestures. Over all nine trials where the number of gestures classified ranged from two to ten, random forests achieved an average accuracy of 76.89% ($SD = .075$), while dropout nets achieved an average accuracy of 84.23% ($SD = .028$). Our results indicate that for this dataset, random forests are more sensitive than dropout nets to increasing the number of classification labels, but actually slightly outperform dropout nets for binary classification. This mirrors previous findings about the superiority of random forests for binary classification, discussed in [4].

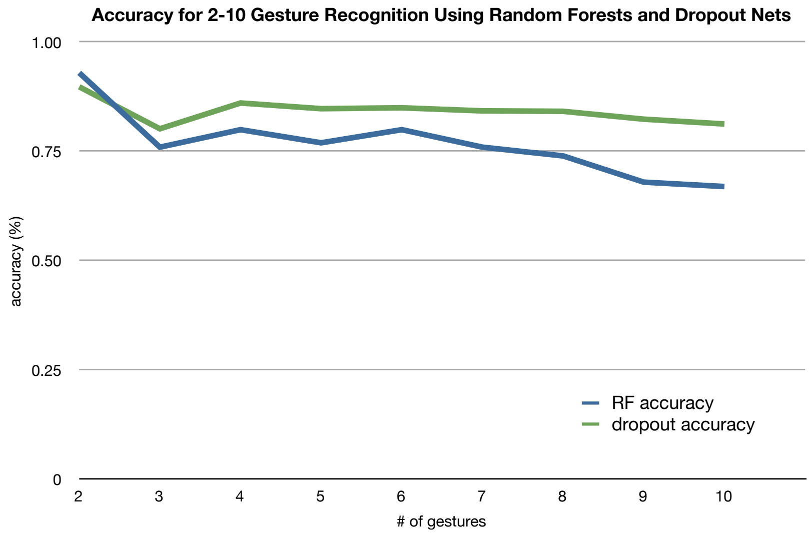

Figure 2: The accuracy of dropout nets surpasses that of random forests, especially as the number of gestures increases.

In order to ensure the validity of our design choices, we also varied the parameters of both dropout nets and random forests and tested the resulting accuracy (shown in tables 1 and 2). The accuracies reported in both tables refer to the problem of classifying ten gestures simultaneously. The results of varying the number of trees in the random forest are shown in table 1. Accuracy does not appear to vary with the number of trees, so we opted to use three trees for the rest of the tests; firstly because it is suggested in the literature [2], and secondly because increasing the number of trees increases the required computation time. It should be noted that if much larger numbers of trees were tested, a pattern may have emerged. However, we found the computation time required to test a large number of trees to be prohibitive. Table 2 shows how the accuracy of the dropout network was affected by the architecture; that is, the number of neurons in the two hidden layers. The three architectures shown here are those used by Hinton to test dropout on the MNIST dataset that is closest to our own [6]. The choice of architecture did not significantly affect the resulting accuracy.

Table 1: Random Forests

Number of Trees	Accuracy
3	67.27%
4	65.68%
5	67.24%
6	65.96%
7	67.91%

Table 2: Dropout Nets

Architecture	Accuracy
1024-800-800-10	81.57%
1024-1200-1200-10	81.34%
1024-2000-2000-10	81.74%

3.2 Dropout Technique

In order to establish the usefulness of the dropout technique, we also tested the same neural net architecture without dropout on the problem of classifying ten gestures. Without dropout, the neural net produced a classification accuracy of 76.88%, compared to the 81.34% obtained with dropout. According to these results, the dropout technique provided a significant improvement in accuracy over the unmodified neural net, confirming the effectiveness of dropout as demonstrated in [6], [7], and [10]. It is interesting to note that even without dropout, neural nets had higher accuracy on this dataset than random forests (67.27%). The error rate for both the unmodified neural net and the dropout net is shown in figure 3. The unmodified net achieves stability sooner than the dropout net, since the unmodified version is training a single net with N nodes, while dropout is essentially training 2^N different nets.

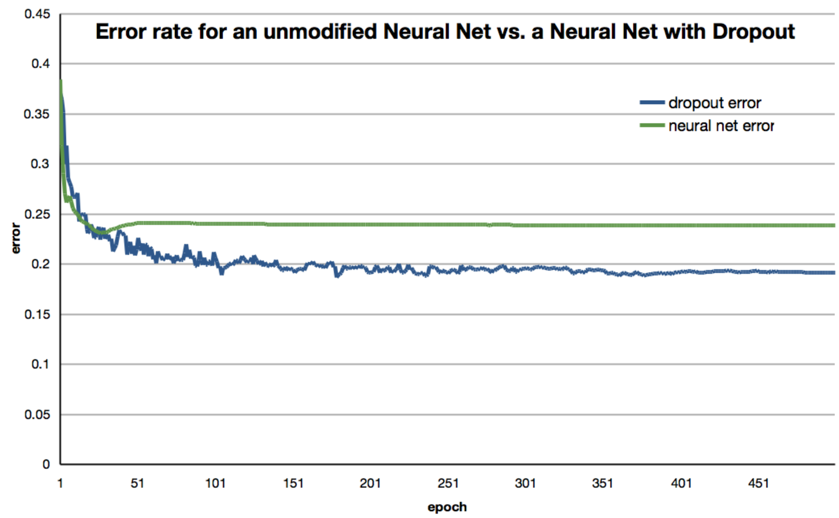

Figure 3: This chart shows the error rate of the same neural net architecture trained with and without dropout over 500 epochs. Adding dropout reduces the error rate by a significant margin.

3.3 Gesture Accuracy

The results from another study on random forest ASL recognition with the Kinect afford us an opportunity to compare the results obtained with our dataset directly to those in the literature [3]. Figure 4 shows the accuracy random forests achieved on each of the ten gestures. While some gestures (such as '5') appear to be distinctive, and result in a high accuracy, other gestures (such as 'L') are not easy to recognize. Both our dataset and the one described in [3] obtained nearly identical accuracies on the letters 'A', 'C', 'H', and 'I', but our results for the letters 'L', 'Q', and 'Y' were significantly worse. Interestingly, [3] achieved a recognition accuracy of 87% for the letter 'L', whereas our random forest could only distinguish the letter 'L' with a meager 45% accuracy. This could be due to the fact that the previous study didn't include number signs [3], and 'L' became confused with one of the numbers. Or, it could be due to the noisiness and idiosyncrasies of our dataset; for example, the low accuracy obtained for the letter 'Q' is no surprise given the amount of difficulty many of the participants had with forming this hand sign.

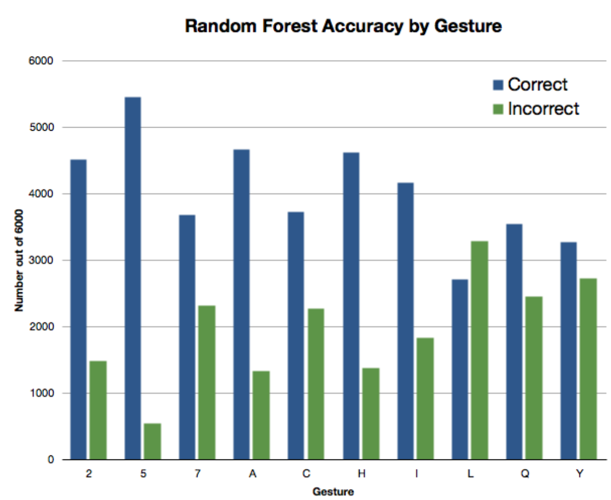

Figure 4: The accuracy obtained by random forests on each of the ten gestures

3.4 Computation Time

Although dropout nets did achieve far better results, obtaining these results also took much longer. Figure 5 shows the computation time in minutes for random forests and dropout nets over each of the ten trials. The greater accuracy of neural networks is paid for by a much longer computation time, as well as a heavier memory footprint. This increased computation time could become a concern for a system that has to learn to classify gestures in real time [2].

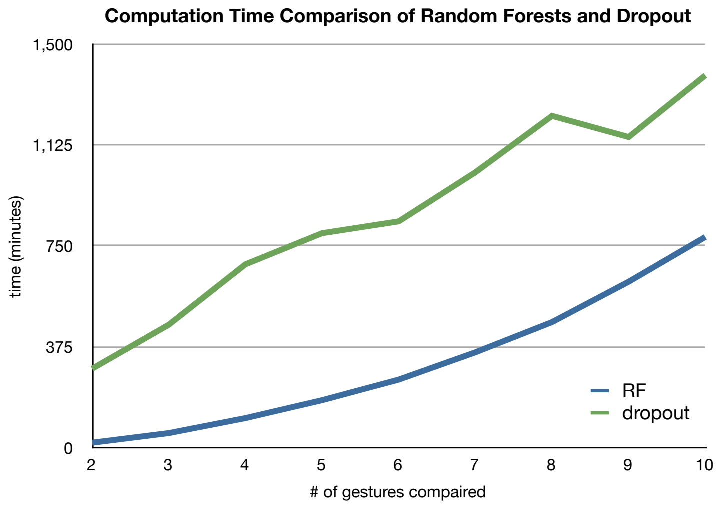

Figure 5: The computation time taken by each algorithm. Note that for 6-10 gestures dropout nets were run using a different server, so these times should not be directly compared.

4 Future Work

Modifications to dropout networks have been proposed that could potentially make them even more effective. One example is increasing the step sized used in stochastic gradient descent [7]. The reasoning behind this proposal is that dropout actually trains many different networks, and each one of these 2^N configurations may only ever see one training example. Therefore, each update must have a larger effect [7]. A further modification to dropout called *maxout* uses hidden neurons that output the maximum of the inputs, and it has been shown that this technique further improves the state of the art on several datasets, including MNIST [7]. Another way to increase the effectiveness of dropout is to apply it to a deep Convolution Neural Network (CNN), a type of network adapted to computer vision problems which contains layers specifically designed to perform filtering operations [10], [20]. Deep networks such as these can model complex dependencies between features and classification labels, and have been shown to outperform random forests in some computer vision applications [20]. In terms of gesture recognition specifically, performing transformations to the images to reduce the number of features from 32x32 pixels to something more compact could be extremely helpful [13]. Using any of these techniques could potentially improve our classification accuracy.

5 Conclusions

Our research tested the effectiveness of random forests against that of dropout nets on a gesture classification task. Silhouette images of ASL hand signs were obtained using the Microsoft Kinect, useful for gesture recognition because of its robustness to variance in lighting and skin tone. Although random forests have been employed commercially by Microsoft for classifying Kinect images [2], the accuracy of random forests on our dataset was over 10% worse than that of dropout nets when classifying ten gestures. Further, the accuracy of random forests showed signs of decreasing further had more gestures been included, whereas the performance of dropout nets was more stable. The addition of the dropout technique also significantly improved the performance of the baseline neural net, demonstrating the effectiveness of dropout. However, the increased accuracy of dropout nets is balanced by increased computation time required for training.

References

[1] Antanas Verikas, Adas Gelzinis, and Marija Bacauskiene. Mining data with random forests: A survey and results of new tests. *Pattern Recognition*, 44(2):330–349, 2011.

[2] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

[3] Nicolas Pugeault and Richard Bowden. Spelling it out: Real-time asl fingerspelling recognition. In *Computer Vision Workshops (ICCV Workshops), 2011 IEEE International Conference on*, pages 1114–1119. IEEE, 2011.

[4] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM, 2006.

[5] Thomas G Dietterichl. Ensemble learning. *The handbook of brain theory and neural networks*, pages 405–408, 2002.

[6] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[7] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. Maxout networks. *arXiv preprint arXiv:1302.4389*, 2013.

[8] Guoqiang Peter Zhang. Neural networks for classification: a survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 30(4):451–462, 2000.

[9] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[10] Alex Krizhevsky, Ilya Sutskever, and Geoff Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.

[11] Maged N Kamel Boulos, Bryan J Blanchard, Cory Walker, Julio Montero, Aalap Tripathy, Ricardo Gutierrez-Osuna, et al. Web gis in practice x: a microsoft kinect natural user interface for google earth navigation. *International journal of health geographics*, 10(1):45, 2011.

[12] Shuangqing Wu, Yin Zhang, Sanyuan Zhang, Xiuzi Ye, Yiyu Cai, Jianmin Zheng, Soumita Ghosh, Wenyu Chen, and Jane Zhang. 2d motion detection bounded hand 3d trajectory tracking and gesture recognition under complex background. In *Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and its Applications in Industry*, pages 311–318. ACM, 2010.

[13] Jason Isaacs and Simon Foo. Optimized wavelet hand pose estimation for american sign language recognition. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 1, pages 797–802. IEEE, 2004.

[14] J Noble. Programming interactivity: A designer's guide to processing, arduino, and openframeworks (2009).

[15] Matthew Tang. Recognizing hand gestures with microsoft's kinect. *Palo Alto: Department of Electrical Engineering of Stanford University:[sn]*, 2011.

[16] Luis Pedro Coelho. Milk: Machine Learning Toolkit for Python. `http://luispedro.org/software/milk`, 2010.

[17] Amazon Elastic Compute Cloud (Amazon EC2). `http://aws.amazon.com/ec2/`, 2013.

[18] Misha Denil. Dropout code - github. `https://github.com/mdenil/dropout`, 2012.

[19] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.

[20] David Grangier, Léon Bottou, and Ronan Collobert. Deep convolutional networks for scene parsing. In *ICML 2009 Deep Learning Workshop*, volume 3. Citeseer, 2009.