Video-Based Chinese Sign Language Recognition Using Convolutional Neural Network

Su Yang, Qing Zhu
Faculty of Information Technology
Beijing University of Technology
Beijing, China
e-mail: ymiyya@gmail.com, ccgszq@bjut.edu.cn

Abstract—**Convolutional neural network (CNN) has been widely used in computer vision tasks recently and achieved remarkable success. This paper presents a novel video-based recognition approach using CNN for Chinese Sign Language (CSL). The proposed method extracts upper body images directly from videos, and employs a pre-training convolutional network model to recognize the gesture in the image. The new method simplifies the hand-shape segmentation, and avoid information loss in feature extraction. We evaluate the method on our self-built dataset includes 40 daily vocabularies, and show that the proposed approach has good performance on sign language recognition task, with accuracy reaching to 99%.**

Keywords-sign language recognition; convolutional neural network (CNN); deep learning; skin-color model

I. INTRODUCTION

Sign language is the essential tool for the hearing impaired people to communicate with others. But there're just a few of ordinary people learned sign language. So it's difficult for the most normal people to understand the hearing impaired. The mission of sign language recognition system is to provide a convenient platform that converts the sign language movements into text, and give support to daily conversations between the normal and the hearing impaired. Besides, sign language recognition plays critical role in human-computer interaction area and graphic applications [1]. It receives extensive attention in recent years.

In this paper, we present an effective approach for video-based Chinese Sign Language recognition (CSL) to improve the performance of this task. Upper body images centered on hand of the CSL user are extracted directly from videos, as the material of recognition after processing. And convolutional neural network (CNN) model are trained for recognition. Because of the advantage of CNN, the novel method can simplify feature extraction, which has significant impact on establishing a real-time system. The method is being experiment on our self-built dataset, which contains 40 videos of CSL daily vocabularies.

II. RELATED WORK

Starner [2] proposed a method based on Hidden Markov Model (HMM) for American Sign Language recognition in 1995. They required the subject to wear distinct colored gloves on each hand, which made it easy to capture hand shapes, orientation and trajectory. The experiment on 99 test sentences, which consisted of 40 vocabulary words, showed that the method achieve 90.7% precision. 1998, his team collected hand movements with skin-model from two views, and extracted hand blobs as feature. The wearable-based recognition system reached a high accuracy of 96.8% on 100 test sentences, while the desk-based system only 74.5% [3]. HMM has been very popular for dealing with continuous signs and many researchers developed gesture recognition methods based on it [4]-[6].

Meanwhile, many sign language recognition approaches are developed on other learning algorithm. Yang et al. used time-delay neural network (TDNN) to learn the motion patterns from the extracted trajectories. They applied the proposed method to recognize 40 hand gestures of ASL, and proved its availability on sign language recognition [7]. Keskin et al. trained random decision forests (RDF) on depth images retrieved from Kinect, and described support vector machine (SVM) based module for recognition. In the experiment on ASL, this method attained a recognition rate of 99.9% on live depth images in real–time [8]. Tangsuksant et al. extracted 3D coordinate points by using Circle Hough Transform to detect markers. After the feature computation of feasible triangle area patches, a classification using Artificial Neural Network (ANN) was designed. The experimental result showed the high performance of the method [9]. Adithya et al. proposed a recognition method for Indian sign language, which used digital image processing technology and ANN. This approach was vision based and didn't utilize other devices [10].

For CSL recognition, many efforts have been made. Gao et al. developed a system based on ANN-DP combined approach to recognize continuous large vocabulary CSL. Experiments showed that proposed method are efficient for CSL [11]. The accelerometer and surface electromyographic (sEMG) sensors were applied to Li et al.'s sign language recognition system for extracted features. And experiments on a vocabulary of 120 signs and 200 sentences demonstrated the feasibility of the method [12]. Geng et al. used hand shape and spherical coordinate as feature and Extreme Learning Machine (ELM) as recognition method. The superior performance of ELM was proved in experiments on their self-built dataset [13].

III. EXTRACTION OF UPPER BODY

As a video-based approach, our system need to extract the upper body images and preprocess them, in order to provide materials for the subsequent recognition. Since the images are required to focus on hand shapes and movements, we should detect the center of hands and capture the upper body images around the hand. To increase the efficiency of recognition, the background of images should be removed.

A. Detection of Hand Region

Detection of hand region is a crucial step for the extraction of upper body image. As shown in Fig. 1, the hand detection process is as follows:

- Remove the face region. Since the detection of hand region is based on skin-color detection, the face region need to be removed in advance, in order to avoid the interference in hand detection.
- Detect the hand contour. The skin-color based detect method is used to get the contour of both hands.
- Find the hand center. In order to collect enough materials for following recognition, the traditional hand image segmentation is replaced by the location of hand center.
- Extract upper body image around hand center. The upper body images is captured with size 256*256.

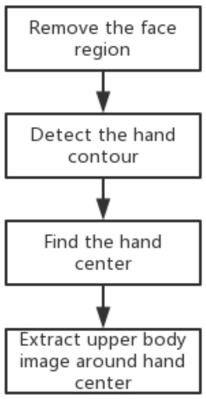


Figure 1. Hand detection process.

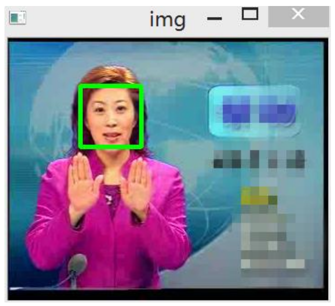


Figure 2. Face region detection result.

For the face region remove, a rapid and effective face detection method shall preferably be employed. So we choose the Haar feature classifier, which has been proved that can satisfy the request above [14], [15]. Opencv provides a convenient cascade classifier supporting Harr feature [16], which used to detect the face region. And it works well in experiments. The results is as shown in Fig. 2.

The hand region detection is based on skin-color detection approach, which has been widely used in hand location and gesture recognition [17]. Considering the people have different skin color, we build a specific skin color model on the face region. In order to make the skin color cluster more compact, the image based on RGB color space is transformed to Ycbcr color space. The transformation matrix is as (1).

$$\begin{bmatrix} Y \\ C_r \\ C_b \end{bmatrix} = \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.5 & -0.419 & -0.081 \\ -0.169 & -0.331 & 0.5 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

where Y is the luminance component and C_r, C_b represent the chrominance factors. To improve calculation speed, the basic range of skin color is set at first, with range of C_r, C_b and Y are [133, 173], [77, 127], [0, 255] respectively[18]. Calculate the gray value of RGB image as follows.

$$\hat{x}_{ij} = 0.2989x^{r}_{ij} + 0.5870x^{g}_{ij} + 0.1140x^{b}_{ij} \tag{2}$$

where $x^{r}_{ij}, x^{g}_{ij}, x^{b}_{ij}$ are the pixels of R, G, B channels. The range of each component is update according to the following rules shown in Fig. 3. These three ranges after updating compose a specific skin-color model for the subject in images.

The output of the color detection is a single-channel image that have the same size with input image. With the skin-color model, the output pixel at point (i, j) is described as follows,

$$P(i,j) = \begin{cases} 255, if [C_r(i,j) \in R_{C_r} \cap C_b(i,j) \in R_{C_b}] \\ 0, otherwise \end{cases} \tag{3}$$

where R_{C_r}, R_{C_b} are the range of C_r, C_b.

```
proc Update_Skin_Model($\hat{x}_{ij}, Y_{max}, Y_{min}, C_{r_{max}}, C_{r_{min}}, C_{b_{max}}, C_{b_{min}}, Y, C_r, C_b$)
/* $[Y_{min}, Y_{max}], [C_{r_{min}}, C_{r_{max}}], [C_{b_{min}}, C_{b_{max}}]$ are the range of three componets*/
begin
  if ($\hat{x}_{ij}$<200 and $\hat{x}_{ij}$>40 and R>G and R>B):
    if $Y > Y_{max}$ then $Y_{max} = Y$
    if $Y < Y_{min}$ then $Y_{min} = Y$
    if $C_r > C_{r_{max}}$ then $C_{r_{max}} = C_r$
    if $C_r < C_{r_{min}}$ then $C_{r_{min}} = C_r$
    if $C_b > C_{b_{max}}$ then $C_{b_{max}} = C_b$
    if $C_b < C_{b_{min}}$ then $C_{b_{min}} = C_b$
end
```

Figure 3. C_r, C_b and Y component range update rules.

After the skin-color detection, we use the tools provided by Opencv to obtain the close contour of each skin color areas. Since the image may have one or two hands, we use the two largest contours to calculate the hand regions. The reason for this is that the face region has been removed, so the hand region is the largest skin-color area. To increase the robust of the system, a threshold is set to filter the minor skin-color area. For our self-built dataset, this threshold is set to 370 , which attained from experiments. We calculate up-right bounding rectangle of hand region, and the center of rectangle is the center of hand region. The upper body images is extracted around the center, with a specific size fit the original image size. The result of extraction is illustrated in Fig. 4.

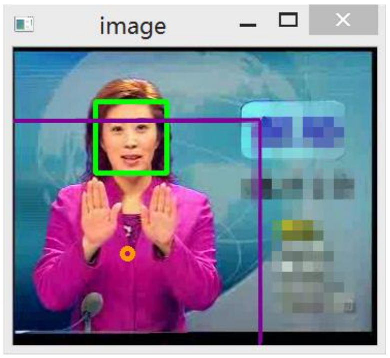


Figure 4. Hand center location and upper body extraction.

B. Background Removal and Color Space Transform

Since the recognition method is based on CNN model and the static background may cause network overfitting, it's necessary to remove the image static background. We compute the mean image over our training and testing datasets and subtract the mean from each input image. For the subtraction may get negative numbers that cannot be stored as pixels and the absolute values of the negative are very small in practice, the negative numbers are replaced by 0.

For the convolution operation takes a lot of time in training, we transform the images to the Hue-Saturation-Value (HSV) color space, which is closely to the human perception of color [19]. The hue component is the most significant feature used to detect uniform color regions. So we choose it to build the single channel images as the recognition method input. The result is as shown in Fig. 5.

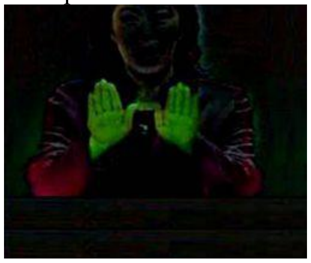


Figure 5. Upper body image after background removal and color-space transform.

IV. Sign Language Recognition Based on CNN

Convolutional neural network is a popular deep learning architecture, which has been widely used in object detection recently and proved its effectiveness on computer vision work [20]-[22]. For images, each pixel of a field can be represented by a neuron and all operations can be modeled by connections between neurons in the network. Compared with traditional image processing algorithm, the advantage of CNN is that it avoids the complex feature extraction and allows direct image input. That's why we choose it as the basic of classification. In this paper, we focus on using CNN to build an efficient sign language recognition method.

A. Convolutional Neural Network

For images, CNN classifies the input by a set of connection operations. CNN contains a serial of sequential convolutional layers and subsampling layers, and these layers are followed by one or more fully connected layers. Several fields of same size representing the results of previous layer comprise the subsequent layers. Each convolutional layer executes a different two-dimensional convolutional operation on its input image with a given kernel. After the convolution, a bias is added to every output map, and the result is passed through a non-linear transfer function for activation. The output of convolutional layers is described as (4).

$$a_j^l = f(\sum_{i\in M_j} a_i^{l-1} * k_{ij}^l + b_j^l) \quad (4)$$

where $*$ is the convolution operation, a_j^l is the *j*-th output map of *l*-th layer, a_i^{l-1} is *i*-th input map in (*l*-1)-th layer, M_j is the set of input maps, k_{ij}^l is the kernel of *l*-th layer in the network between the *i*-th input map and *j*-th output map, b_j^l is the bias of *l*-th for *j*-th output, and *f* is activation function.

The objective of subsampling layers is to lower the dimensions of results calculated by convolutional layers. It can reduce the parameters of network. Each feature map after convolution will be subsampled by mean or max method over a region with specific size, which is the patch size of subsampling layer. So we have that:

$$a_j^l = g(a_i^{(l-1)}), \forall i \in R_j \quad (5)$$

where g is the subsampling operation and R_j is the *j*-th subsampling region in the *i*-th input map.

Furthermore, the fully connected layer can be described as a set of convolutions where each feature map is connected with every field of the successive layer and the filters have the same size as the input image. In order to prevent network from overfitting, the fully connected layer usually followed by dropout on the hidden layer [23]. Dropout means drop units out randomly with a probability p , which can be set zero during feedforward and back-propagation in the network. It can be described as follows.

$$a^l = f(w \cdot (a^{(l-1)} \bullet r) + b) \quad (6)$$

where a^l is the output unit in feedforward propagation, $\bullet$ is the element-wise multiplication operator, and $r \in R^h$ is a masking vector of Bernoulli random variables with probability p of being 0.

B. Architecture Overview

Table I shows the CNN architecture used in our work. This network contains three convolutional layers followed by max-pooling. Considering the amount of memory available on our GPU and the training time we can bear, we use as many subsampling layers as possible to reduce the amount of parameters. Then we flatten the output in the convolutional layer, and the followed fully connected layers are regularised by dropout, which can avoid the network overfitting. In the bottom layer, the recognition is made by softmax function , which is usually used in multiple classification CNN architecture.

The size of input windows is 256×256×1. We choose the hue component of HSV color space as the single channel of input window. The first convolutional layer uses 16 kernels of size 5×5. The max-pooling of patch size 5×5 is performed on the output of convolution, and the result will be the input of second convolutional layer. The second convolutional layer filters it with 32 kernels of size 5×5, and the third one use 64 kernels of size 3×3. This two convolutional layers are followed by max-pooling layer with size 3×3. The forth column in Table I, param#, represents the number of parameters we need to learn for each layer. A rectification activation function(RELU) is used in the hidden weight layers following convolutional layers. Compared with tanh or sigmoid activation functions, RELU need less computation and training time [22]. It makes part of the neuron output 0, which reduces the interdependence relation within parameters and eased the network overfitting.

TABLE I. CNN ARCHITECTURE

Layer type	Kernel size	Output size	Param #
input	-	256×256×1	-
convolution	16@5×5	16@252×252	0.4K
max-pooling	16@5×5	16@50×50	0.4K
convolution	32@5×5	32@46×46	0.8K
max-pooling	32@3×3	32@15×15	0.2K
convolution	64@3×3	64@13×13	0.5K
max-pooling	64@3×3	64@4×4	0.5K
flatten	-	1024	-
dropout(0.2)	-	1024	-
fullyconnected	-	128	13K
dropout(0.5)	-	128	-
softmax	-	40	5k

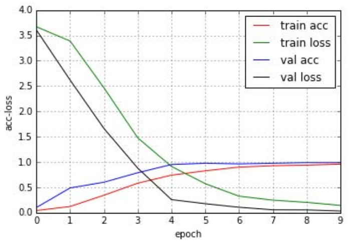
The dropout rates in our architecture are set to 0.2 and 0.5 for the flatten layer and full connected layer. And the loss function used in output layer is log-likelihood function, which corresponds to the softmax function. Adagrad [24] and ADADELTA [25] are chosen as the learning algorithm in our method. Adagrad is based on gradient descent algorithm, but using different learning rate for each parameters. For Adagrad, the initial learning rate is set to 0.01, and there is no need to tune learning rates manually during the training. It has high robust on sparse data. ADADELTA makes further improvements on Adagrad. It can be applied without setting learning rates, and is insensitive on hyper-parameters.

This architecture has been proved its performance on a video sign language dataset in our experiments, which will be presented in Section V.

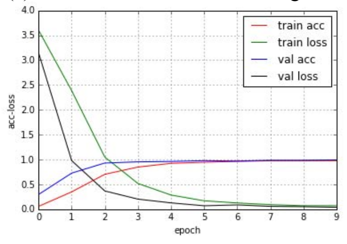
V. EXPERIMENT AND RESULTS

Experiments in our paper are conducted on a Chinese sign language instructional video named *We Learn Sign Language*. It's a widely used instructional materials in China and can be download online. In the video, each word is performed several times, so that we can get more accurate gestures. We choose 40 daily vocabularies in Chinese sign language, and obtain the corresponding gesture videos to build our dataset. Considering the regular speed of Chinese sign language movement, these videos are over 0.1s length.

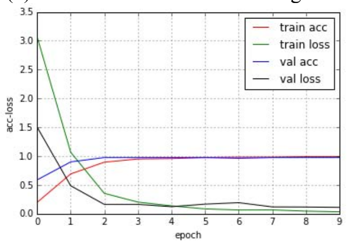
The video sampling rate is set to 1000 frames per second in our work. With the method presented in Section III, 80 upper body images are extracted from video and processed for each word. The full set is split into two disjoint sets with percentage: (1) 50% for training and 50% for testing; (2) 80% for training and 20% for testing. We load the training data in random order, and take 5% to validate the network while training. So for the two proportion above, the data volumes for training, validating and testing are 1520/80/1600 and 2432/128/640. The experiment results are shown in Fig. 6 and Table II.



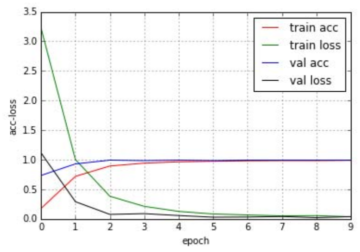
(a) ADADELTA and 50% training data



(b) ADADELTA and 80% training data



(c) Adagard and 50% training data



(d) Adagrad with 80% training data

Figure 6. Lost decreasing and accuracy increasing of training set and validate set with: (a) ADADELTA and 50% training data; (b) ADADELTA and 80% training data; (c)Adagard and 50% training data; (d) Adagrad with 80% training data.

TABLE II. ACCURACY WITH DIFFERENT DATA ALLOCATION

	50% training data		**80% training data**	
	Training	***Test***	***Training***	***Test***
ADADELTA (%)	99.313	98.625	99.805	100
Adagrad (%)	99.875	99.438	99.844	99.688

Fig. 6 demonstrates the loss decreasing and accuracy increasing on training set and validating set with two learning algorithm, ADADELTA and Adagrad. The loss values smoothly decline and gradually stabilizes for ADADELTA. In contrast, the loss decreases faster for Adagrad, but a few shaking in the middle. The accuracy achieves over 90% at the 4th epoch for Adagrad, which rises more slowly for ADADELTA.

Table II shows the recognition accuracy with different data allocation. The CSL recognition method using CNN model achieves high accuracy in each case, which is about 99%. It's proved that the proposed method has remarkable recognition ability on CSL, and the CNN model we built is applicable to sign language recognition. As shown in Table. II, the training accuracy with ADADELTA rises gradually with the data size, and the test accuracies are higher than Adagrad at the same conditions. It demonstrates that ADADELTA is more appropriate for our method.

TABLE III. COMPARSION WITH ANN BASED METHOD ON OUR DATASET

	50% training data		**80% training data**	
	Training accuracy	***Test accuracy***	***Training accuracy***	***Test accuracy***
ANN (%)	59.98	60.214	66.016	61.039
Our method (%)	98.938	98.125	99.805	99.844

The method we described above is not assisted by sensors, data gloves, etc. Hence, we compare the experimental results with the method based on ANN shown in [10], which does not use accessibility tools either. The comparison results are shown in Table III. Obviously, the ANN method is ineffectiveness for the large data task, since the accuracies of training and test are only about 65%. Meanwhile, the recognition rates of our method reach over 98%. Experiments show that the proposed recognition method based on CNN has powerful and stable feature learning ability, which is essential in a practical application.

VI. CONCLUSION AND FUTURE WORK

In this paper, we have presented an effective video-based recognition method for CSL, which using CNN as classifier. It extracts upper body images from videos directly, and recognizes the gesture in images with a CNN model. The method have produced remarkable recognition accuracy on videos without any assistance of motion capture devices. And our work have proved that CNN model is suitable for learning from many video samples to recognize CSL gesture .

In the next step, we plan to collect more data to refine our recognition method. Considering the method's practicability, we will focus on the recognition of phrases and sentences in sign language.

REFERENCES

[1] M. Billinghurst, "Put that where? voice and gesture at the graphics interface," Acm Siggraph Computer Graphics, vol. 32, no. 32, 1998, pp. 60-63.

[2] T. Starner, and A. Pentland, "Visual Recognition of American Sign Language Using Hidden Markov Models," International Workshop on Automatic Face & Gesture Recognition, 1995, pp. 189-194.

[3] T. Starner, J. Weaver, and A. Pentland, "Real-time American sign language recognition using desk and wearable computer based video," IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 20, no. 12, 1998, pp. 1371-1375.

[4] N. Tanibata, N. Shimada, and Y. Shirai, "Extraction of Hand Features for Recognition of Sign Language," International Conference on Vision Interface, 2002, pp. 391-398.

[5] M. Elmezain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis, "A hidden markov model-based continuous gesture recognition system for hand motion trajectory," in 19th International Conference on Pattern Recognition, 2009, pp. 1-4.

[6] C.B. Park and S.W. Lee, "Real-time 3D pointing gesture recognition for mobile robots with cascade HMM and particle filter," Image & Vision Computing, vol. 29, no. 1, 2011, pp. 51-63.

[7] M. H. Yang, N. Ahuja, and M. Tabb, "Extraction of 2D motion trajectories and its application to hand gesture recognition," IEEE Transactions on Pattern Analysis & Machine Intelligence, vol. 24, 2002, pp. 1061-1074.

[8] C. Keskin, F. Kirac, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in IEEE International Conference on Computer Vision Workshops, 2011, pp. 1228-1234.

[9] W. Tangsuksant, S. Adhan, and C. Pintavirooj, "American Sign Language recognition by using 3D geometric invariant feature and ANN classification," in Biomedical Engineering International Conference, 2014, pp. 1-5.

[10] V. Adithya, P.R. Vinod and U. Gopalakrishnan, "Artificial neural network based method for Indian sign language recognition," Proc. Information & Communication Technologies, 2013, pp. 1080-1085.

[11] W. GAO, J. MA, J. WU, and C. WANG, "SIGN LANGUAGE RECOGNITION BASED ON HMM/ANN/DP," International Journal of Pattern Recognition & Artificial Intelligence, vol. 14, 2011, pp. 587-602.

[12] Y. Li, X. Chen, X. Zhang, K. Wang, and Z. J. Wang, "A sign-component-based framework for Chinese sign language recognition using accelerometer and sEMG data," IEEE transactions on bio-medical engineering, vol. 59, 2012, pp. 2695-2704.

[13] L. Geng, X. Ma, H. Wang, and J. Gu, "Chinese sign language recognition with 3D hand motion trajectories and depth images," in Intelligent Control and Automation, 2014, pp. 1457-1461.

[14] R. Lienhart and J. Maydt, “An extended set of Haar-like features for rapid object detection,” Proc. International Conference on Image Processing. 2002. Proceedings, 2002, pp. I-900 - I-903 vol.901.

[15] J. Cho, S. Mirzaei, J. Oberg, and R. Kastner, "Fpga-based face detection system using Haar classifiers," in Acm/sigda International Symposium on Field Programmable Gate Arrays, FPGA 2009, Monterey, California, Usa, February, 2009, pp. 103-112.

[16] OpenCV 2.3.2 documentation

http://www.opencv.org.cn/opencvdoc/2.3.2/html/index.html

[17] S. Akyol and P. Alvarado-Moya, "Finding Relevant Image Content for mobile Sign Language Recognition," in IASTED International Conference Signal Processing, 2012, pp. 48-52.

[18] D. Chai and K. N. Ngan, "Locating facial region of a head-and-shoulders color image," in IEEE International Conference on Automatic Face & Gesture Recognition, 1998, p. 124.

[19] N. Herodotou, K. N. Plataniotis, and A. N. Venetsanopoulos, "A color segmentation scheme for object-based video coding," in Advances in Digital Filtering and Signal Processing, 1998 IEEE Symposium on, 1998, pp. 25-29.

[20] H. Li, Z. Lin, X. Shen, J. Brandt, and G. Hua, "A convolutional neural network cascade for face detection," in Computer Vision and Pattern Recognition, 2015, pp. 5325-5334.

[21] D. Annane, J. C. Chevrolet, S. Chevret, and J. C. Raphaël, "Two-Stream Convolutional Networks for Action Recognition in Videos," Advances in Neural Information Processing Systems, vol. 1, 2014, pp. 568-576.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems, vol. 25, 2012, pp. 2012.

[23] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," Journal of Machine Learning Research, vol. 15, 2014, pp. 1929-1958.

[24] J. Duchi, E. Hazan, and Y. Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," Journal of Machine Learning Research, vol. 12, 2010, pp. 257-269.

[25] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," Computer Science, 2012.