Recognizing Handwritten Japanese Characters Using Deep Convolutional Neural Networks

Charlie Tsai
Department of Chemical Engineering
Stanford University
ctsai89@stanford.edu

Abstract

In this work, deep convolutional neural networks are used for recognizing handwritten Japanese, which consists of three different types of scripts: hiragana, katakana, and kanji. This work focuses on the classification of the type of script, character recognition within each type of script, and character recognition across all three types of scripts. Experiments were ran on the Electrotechnical Laboratory (ETL) Character Database from the National Institute of Advanced Industrial Science and Technology (AIST). In all classification tasks, convolutional neural networks were able to achieve high recognition rates. For character classification, the models presented herein outperform the human equivalent recognition rate of 96.1%.

1. Introduction

Written Japanese consists of three types of scripts: logographic *kanji* (Chinese characters) and syllabic *hiragana* and *katakana* (together known as *kana*). The scripts differ in both appearance and usage. Hiragana consists of round and continuous strokes that are more similar to cursive writing seen with latin alphabets. Both katana and kanji consist of straight and rigid strokes. However, kanji are further distinguished from the other two systems in consistently of building blocks. Examples of each script are shown in Fig. 1. As such, the challenges in recognizing each type of writing system are different. Furthermore, there are distinct writing styles for each system that can vary widely from person to person. The differences are more pronounced for written hiragana, which is a more cursive script. All these differences will need to be accounted for in order to successfully recognize handwritten Japanese characters. Either the type of script must be identified and then classified accordingly, or the recognition must be simultaneously accurate for all three scripts. Convolutional neural networks (CNN) have emerged as a powerful class of models for recognizing handwritten text, especially for handwritten digits and Chinese characters. In this work, CNN's are used for recognizing handwritten Japanese characters. For both discriminating between the scripts and classifying the characters within each script, CNN's were able to achieve high accuracies, surpassing those of previously reported results.

2. Related work

In the area of Japanese character classification, previous works have mainly focused on kanji, which are the most complicated out of the three scripts and contains the largest number of character classes. Japanese kanji are also roughly equivalent to Chinese characters, differing mainly in the number of commonly used characters (~6000 for Chinese and ~2000 for Japanese) and the writing styles. This allows for comparisons between models that have been developed for either of the two languages. A human equivalent recognition rate of 96.1% has also been reported [18]. Previous works employing feature extraction and a Bayes classifier yielded accuracies of 99.15% for training examples [10], and 92.8% for test examples [19]. The state-of-the-art recognition rates for Chinese characters are 94.77% for pre-segmented characters using a convolutional neural network (Fujitsu) and only 88.76% for continuous text recognition using a hidden Markov model (Harbin Institute of Technology) [18]. Recent results from Fujitsu [2] report a further

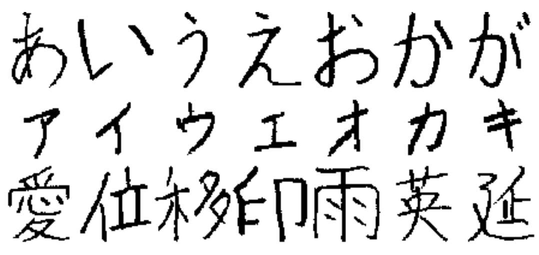
Figure 1. Extracted data taken from the ETL character database. Images have been inverted and enhanced. Top row: hiragana; second row: katakana; third row: kanji.

increase in the classification accuracy of Chinese characters to 96.7%, which surpasses the human equivalent recognition rate of 96.1% [18].

For hiragana, a rate of 95.12% was achieved using a simple three-layer neural network [17], and a rate of 94.02% was achieved using a support vector machine classifier [13]. For katakana, a three-layer neural network achieved a maximum recognition rate of 96.4% for training data [6]. For handwritten Korean, which shares many similar features to Japanese, an accuracy of 95.96% was achieved using a CNN [9].

The majority of the published neural networks employed only fully-connected layers, while most of the efforts were focused on feature extraction. Based on the recent advances in deep convolutional neural networks, there is still ample room to further improve upon these results.

3. Methods

3.1. Convolutional neural networks

A convolutional neural network (CNN) consists of layers that transform an input 3-dimensional volume into another output 3-dimensional volume through a differentiable function. The CNN transforms the original image through each layer in the architectures to produce a class score. Since the input consists of images, the neurons in a convolutional layer of a CNN have an activation "volume" with width, height, and depth dimensions. Activation layers introduce element-wise operations to introduce non-linearities and hence increase the representational power of the model. The rectification non-linearity (ReLU) uses the $\max(0, x)$ activation function, which provides the non-linearity without introducing a problem with vanishing gradients [12]. To control over-fitting, pooling layers is used to reduce the spatial size of the representation and hence reduce the amount of parameters. Drop-out [16] layers provide additional regularization, by only keeping each neuron active with some probability.

The final layer of the CNN is a softmax classifier, where the function mapping $f(x_i; W) = Wx_i$ produces scores that are interpreted as unnormalized log probabilities for each class. The class scores are computed using the softmax function:

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \tag{1}$$

where z is a vector of scores obtained from the last fully-connected layer and f_j is a vector of values between 0 and 1, and sums to 1. The corresponding loss function is then computed as,

$$L_i = -\log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right) = -f_{y_i} + \log\sum_j e^{f_j} \tag{2}$$

where f_j refers to the j-th element of the f, the vector of class scores. The

3.2. Network architectures

Following the network architectures described by Simonyan *et. al.* in their VGGNet work [15], 11 different convolutional neural network architectures were explored. The general architecture consists of a relatively small convolutional layer followed by an activation layer and a max-pooling layer. This is repeated, where the convoluational layer is increased in size at each depth. Finally, there are up to three fully-connected layers before the scores are computed using the softmax classifier.

Following [15], a small receptive field of 3×3 with a stride of 1 was used. This preserves the image size throughout the neural network. All max-pooling are performed over a window of 2×2 pixels with a stride of 2. All convolutional layers and fully-connected layers are followed by a ReLU non-linearity layer [12] and then a dropout of 0.5. The exception is the final fully-connected layer, which is followed directly by the softmax classifier, which is the final layer. Several different sizes for the fully-connected (FC) layers are used, with the final FC layer having the same number of channels as the number of classes (Table. 2).

The different network configurations are summarized in Table 1. The models (M) are named by their number of weight layers. An additional index is used to differentiates between models with the same number of weight layers but different layer sizes. A network with only fully-connected layers (M3) was used to provide a benchmark comparison.

3.3. Classification tasks

Four different classification tasks are investigated in this work: (1) script discrimination between hiragana, katakana, and kanji; (2) hiragana character classification; (3) katakana character classification; and (4) kanji character classification. In practice, either a single CNN model could be used for classifying all scripts, or separate CNN models could be used to identify the type of script and then classify the character. There are far more character classes for the kanji script, so the individual classification tasks will also help illustrate any script-specific challenges.

4. Data

Images of handwritten Japanese characters were obtained from the Electrotechnical Laboratory (ETL) Character Database[3], which contains several million handwritten Japanese characters. Handwritten katakana samples were

Table 1. Convolutional neural network configurations

ConvNet Configurations										
M3	M6-1	M6-2	M7-1	M7-2	M8	M9	M11	M12	M13	M16
3 weight layers	6 weight layers	6 weight layers	7 weight layers	7 weight layers	8 weight layers	9 weight layers	11 weight layers	12 weight layers	13 weight layers	16 weight layers
input (64×64 gray-scale image)										
	conv3-32 conv3-32	conv3-64	conv3-64	conv3-64	conv3-32 conv3-32	conv3-64	conv3-64 conv3-64	conv3-64 conv3-64	conv3-32 conv3-32	conv3-64 conv3-64
	maxpool									
	conv3-64 conv3-64	conv3-128	conv3-128	conv3-128	conv3-64 conv3-64	conv3-128	conv3-128 conv3-128	conv3-128 conv3-128	conv3-64 conv3-64	conv3-128 conv3-128
	maxpool									
		conv3-512	conv3-512 conv3-512	conv3-192	conv3-128 conv3-128	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-128 conv3-128	conv3-256 conv3-256 conv3-256
		maxpool								
				conv3-256		conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 conv3-512	conv3-256 conv3-256	conv3-512 conv3-512 conv3-512
				maxpool		maxpool				
									conv3-512 conv3-512	conv3-512 conv3-512 conv3-512
									maxpool	
FC-5000 FC-5000 FC-5000	FC-256	FC-4096 FC-4096	FC-4096 FC-4096	FC-1024 FC-1024	FC-1024	FC-4096 FC-4096	FC-1024 FC-1024	FC-4096 FC-4096	FC-1024 FC-1024	FC-4096 FC-4096
FC-n_{classes}										
softmax										

taken from the ETL-1 dataset while hiragana and kanji samples were taken from the ETL-8 dataset. Each Japanese character class contains handwriting samples from multiple writers (Fig. 2a): 1,411 different writers for the ETL-1 dataset and 160 different writers for the ETL-8 dataset. The number of character classes and the number of writers per character class are summarized in Table 2. All characters are labeled by their unique Shift Japanese Industrial Standards (Shift JIS) codes. As provided, the images are isolated gray-scale characters that are 64×64 in size. All writing samples have been pre-segmented and centered. In order to maximize the contrast, the images were binarized into black and white using Otsu's method [14]. An example is shown in Fig. 2b. The effect of data augmentation has not been explored in this work.

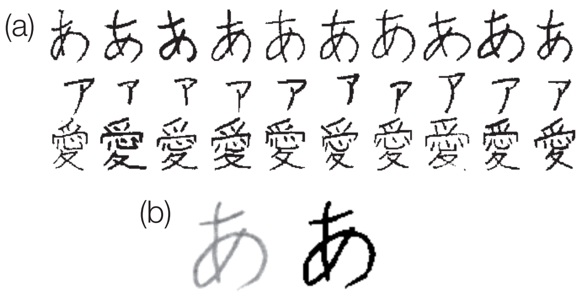

Figure 2. (a): Examples of different handwriting styles for each type of script. Top row: hiragana; second row: katakana; third row: kanji. (b) Grayscale and binarized image.

All regular hiragana and katakana characters are included. The dataset for hiragana characters further includes samples with diacritics, known as *dakuten* and *handakuten*. These are shown in Fig. 3. In the hiragana character set, there are also small versions of the *ya*, *yu*, *yo* characters (together known as *yōon*) and a small version of the *tsu* character (known as *sokuon*). These are functionally distinct but differ only in size. An example is also shown in Fig. 3. While there are 2,136 regularly used kanji (*jōyō kanji*) [1], only 878 different characters are provided in the ETL-8 dataset. The classification accuracy for kanji characters will likely be higher compared to that of models trained on a more complete dataset. Regardless, the results will still reflect the overall ability of the CNN's to discriminate between and recognize the different scripts.

Table 2. ETL dataset information. The number of writers per character and the total number of examples are shown.

Dataset	Script	Writers	Classes	Examples
ETL-1	Katakana	1411	51	71,961
ETL-8	Hiragana	160	75	12,000
ETL-8	Kanji	160	878	140,480
–	All	–	1,004	224,441

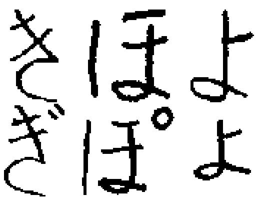
Figure 3. Left: *ki* and *gi* (*dakuten* modification of *ki*); middle: *ho* and *po* (*handakuten* modification of *ho*); right: a normal sized *yo* and a smaller *yōon* version.

5. Classification framework

For all classification tasks, the primary metric used was the classification accuracy, which is the fraction of correctly classified examples.

5.1. Training

Training was performed by optimizing the softmax loss using the Adam optimizer [11] with the default parameters as provided in the original paper: $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. Appropriate initializations are important for preventing unstable gradients, especially in the deeper networks. Weight initializations were performed following He *et. al.* [7], with batch normalization [8] after each weight layer and before each activation layer. Due to time constraints, training was carried out for at least 40 epochs for all models and longer for some models. This may not be sufficient for some of the deeper networks. To determine an initial learning rate and mini-batch size, a grid search was performed and a learning rate of 10^{-4} and a mini-batch size of 16 were found to lead to the largest decrease in training loss in 5 epochs. The learning rate was annealed using a step decay factor of 0.1 every 20 epochs. An example of the loss and training classification accuracy at each epoch is shown in Fig. 4. For training, 80% of all available examples in the corresponding dataset were used. A further 20% of these examples were used for cross-validation at each epoch. Training was stopped when the validation loss and accuracy began to plateau.

5.2. Testing

For testing, the remaining examples not used for training (20% of the total dataset) were used as the test set. The images used at test time are preprocessed in the same way as the training images. As all handwriting images were taken from the ETL database, the test images are also 64×64 in size and are pre-segmented and cropped.

5.3. Implementation details

The models presented herein were implemented in TensorFlow [5] using the Keras interface [4]. Computing resources were provided by Stanford Research Computing.

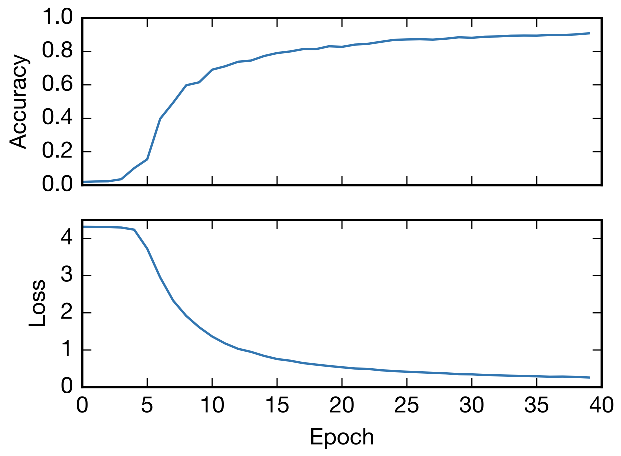

Figure 4. Training accuracy and loss for the M11 model using the hiragana character dataset.

6. Results and discussion

As mentioned previously, the classification accuracy was used as the primary metric for all tasks. All classification results are summarized in Table 3. The best classification accuracies are highlighted in bold. Generally, CNN's with greater depth lead to better classification accuracies on the validation set. This is due to the increased number of parameters that are being introduced to the model with each additional weight layer. The following sections provide detailed comparisons between the models for each classification task.

6.1. Script discrimination

In the script discrimination task, the classifier must label the character as either hiragana, katakana, or kanji. All examples are used (Table 2). Starting with the M3 model, which consists only of fully-connected layers, there is clear over-fitting as the validation accuracy is much higher than the test accuracy. However, the training error could still be improved by increasing the number of parameters. Increasing the depth (M6-1) and the size of the weight layers (M6-2) improves both the validation and the test accuracies. The accuracies remain roughly constant beyond 6 weight layers. Further increases in either the depth or the size of the weight layers don't significantly change the classification rates. The best performing model was the M11 model with 11 weight layers, which resulted in a test accuracy of 99.30%. With a relatively large number of training examples and only three class labels, over-fitting does not become a problem for the deeper networks.

6.2. Hiragana classification

For hiragana classification, there are 75 different classes and 12,000 examples. This is the smallest dataset amongst

Table 3. Classification accuracies for all neural networks. The highest classification accuracies are highlight in bold.

	Script discrimination		Hiragana		Katakana		Kanji		All	
	Val (%)	Test (%)	Val (%)	Test (%)	Val (%)	Test (%)	Val (%)	Test (%)	Val (%)	Test (%)
M3	96.61	91.63	**100.0**	90.17	**100.0**	83.21	98.84	85.78	98.99	89.91
M6-1	93.55	96.61	99.39	96.13	94.61	96.98	92.26	98.33	92.52	97.47
M6-2	99.07	99.27	98.35	95.88	99.04	97.96	99.02	99.28	99.09	98.95
M7-1	99.51	99.08	98.05	96.25	98.81	96.69	**99.55**	99.55	**99.39**	**99.53**
M7-2	99.33	99.22	99.57	**96.50**	99.91	96.92	98.93	**99.64**	98.63	99.20
M8	97.31	99.01	98.98	95.58	99.36	96.02	97.89	99.32	99.07	99.02
M9	99.24	98.60	98.18	94.71	98.92	97.88	99.20	99.03	99.30	99.10
M11	**99.66**	**99.30**	99.47	96.33	99.67	97.30	99.07	99.25	99.12	99.31
M12	99.49	99.18	98.04	96.25	99.03	**98.19**	99.35	99.17	99.35	99.17
M13	93.56*	95.76*	98.70*	95.13*	99.92	96.14	77.68*	93.24*	89.45*	96.69*
M16	99.59	99.29	99.83	96.17	**100.0**	97.79	99.34	98.88	99.07	98.39

*Due to time constraints, these models were only trained for 20 epochs. The accuracies are thus lower than expected.

the three scripts. Once again, the M3 model over-fits the training data, resulting in a perfect validation accuracy but only a 90.15% test accuracy. Increasing the depth of the network and size of layers improves the classification accuracy on the validation set, even up to 16 layers with the M16 model. The test accuracy however, reaches a maximum for 7 layers of depth and remains constant or slightly decreases afterwards. This could indicate over-fitting in the deeper models, due to the relatively small number of examples. Further regularization schemes, such as increasing the dropout rate, may be needed. Another reason for the lower recognition rate is due to the fact that hiragana tends to be a cursive script and is thus more dependent on personal writing styles. To illustrate this, some misclassified characters are shown in Fig. 5. Another perhaps more obvious reason for the lower accuracy is that there exist small versions of *ya*, *yu*, *yo* (together known as *yōon*) and a small version of the *tsu* character (known as *sokuon*). These are functionally distinct from their larger counterparts but differ only in size, which makes it extremely difficult to differentiate when they are pre-segmented (Fig. 3). If the smaller and larger characters are treated as identical, then the classification rates will increase. For the M11 model, the test accuracy increases to 98.83% from 96.33%. Using the JIS labels as provided in the dataset, the best test accuracy was 96.50% from using the M7-2 model.

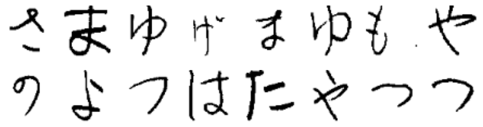
Figure 5. Examples of misclassified hiragana characters. These are either highly cursive writing, or ambiguous *yōon* or *sokuon* characters.

6.3. Katakana classification

For katakana, there are only 51 character classes. Modifications such as diacritics, *yōon*, and *sokuon* are also not included in the ETL-1 dataset. The M3 model with only fully-connected layer over-fits the training data once again. The test accuracy is only 83.12%. Once convolutional layers, with ReLU layers, max-pooling, and dropout are added, the validation accuracy increases to 94.61% for M6 and continues to increase up to 100% again for M16. Again, the results do not seem to be affected by the size of the fully-connected layers, so long as the network is deep enough. The test accuracy continues to improve with layer depth and becomes roughly constant after 12 layers (M12). Due to the large number of examples (1,411 writers per character), the smaller number of classes, and the lack of problematic cases that were present for hiragana characters, a high test accuracy can be achieved. The best result was a 98.19% accuracy using the M12 model. This is a significant improvement over the previously published recognition rate.

6.4. Kanji classification

There are 878 character classes in the kanji dataset. Compared to hiragana and katakana, kanji characters are more complicated and much less ambiguous. Once again, the M3 model without convolutional layers over-fits the training data. With just 6 layers, the test accuracy increased to 98.33%. Both the validation accuracy and the test accuracy plateau at 7 layers of depth (M7-2). Although there is no further improvement with deeper networks, the relatively unchanging test accuracy also indicates that the data is not being over-fitted. The M7-2 model produced the highest classification rate of 99.64%, which far surpasses the previously reported rates for Chinese character classification. However, since the ETL-8 dataset only contains a reduced set of character-classes, this is likely an upper-bound for this neural network architecture.

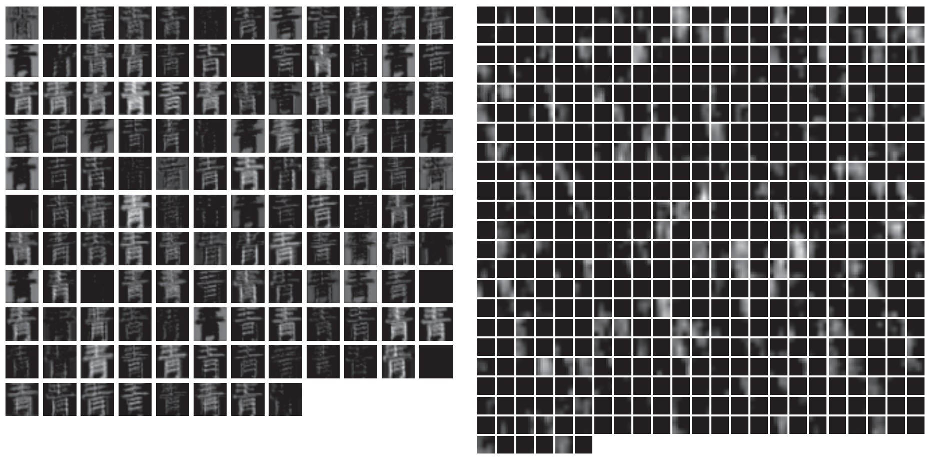

Figure 6. Visualization of the output from the fourth convolution (top) and the final convolution (bottom) layer of the M11 model. Each box corresponds to an activation map of some filter.

6.5. Combined classification

For character classification across all scripts, all examples from the datasets listed in Table 2 were used. In total, there were 1,004 character classes. As the majority of the characters are from the kanji dataset, the trends in the validation and test accuracies are also similar. The best classification rate was 99.53% with 7 weight layers (M7-1).

6.6. Visualization

Additional insights can be obtained by visualizing the output images from each layer in the neural network or by visualizing the weights. The output images after each activation layer (ReLU non-linearity) will indicate the regions of the input image where neurons are being activated. Example outputs from the M11 model are shown in Fig. 6. Even at the fourth convolutional layer, the activations are still dense and follow the outline of the input character. At the 8th convolutional layer, the activations have become sparse and localized.

The weights from the convolutional or fully-connected layers can also be visualized to assess how well the model has been trained. A well-trained model usually has smooth and continuous filters whereas an over-fitted model will display noisy patterns [12]. The 3×3 filters have a small receptive field, which makes it difficult to visualize and compare weights at different depths. In Fig. 7, weights from the first convolutional layer and the final convolutional layer in the M11 model are shown. Due to the low resolution, the weights for all filters are noisy and no evaluation of the smoothness can be made. A modified M7 model was thus trained using a larger 7×7 receptive field (Fig. 7) in order to produce more interpretable images. From the weights from the first and second layer, it is clear that the weights are smooth and continuous. Although this CNN was trained specifically for the purpose of visualization, the architecture is consistent with the other models presented in this work, and could approximate the general behavior in those models.

7. Conclusions

In this work we have investigated different architectures of convolutional neural networks for various classification tasks in handwritten Japanese character recognition. Following the framework of VGGNet's [15], CNN's with vari-

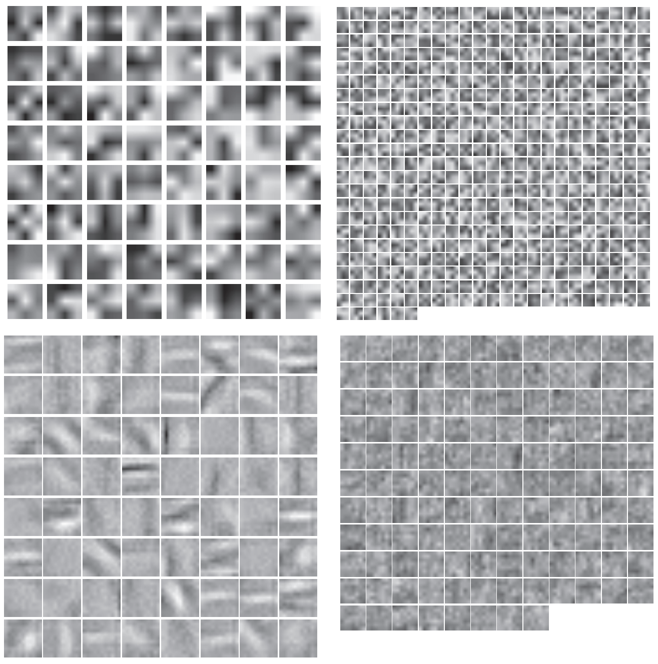

Figure 7. Visualization of the weights from the first (top left) and the final (top right) convolutional layer of the M11 model. The first (bottom left) and second (bottom right) convolutional layers of the modified M7 model with 7×7 filters.

ous depths and layer sizes were trained. CNN's with depths between 7 and 11 were found to perform the best in all tasks. The size of the fully-connected layers was not found to significantly influence the results, as long as the network is deep enough. The best classification accuracies were 99.30% for script discrimination, 96.50% for hiragana classification, 98.19% for katakana classification, 99.64% for kanji classification, and 99.53% for overall classification. In all cases, the recognition rates exceed the reported human equivalent rate. The recognition rate for kanji is likely an optimistic upper-bound, since the dataset only included 878 character classes. However, the high classification rates from the CNN models with depths greater than 7 also indicate that deep CNN's are generally a viable class of models for Japanese handwriting classification.

To further improve the classification of hiragana characters, more pre-processing steps may be needed to disambiguate the regular characters and their *yōon* or *sokuon* counterparts. For the datasets with fewer examples, either more writing samples need to be collected, or further regularization schemes need to be applied. Further fine-tuning of the hyperparameters such as the dropout rate could also further improve the models' performances. CNN's have the advantage of being able to combine individual character recognition tasks with other processes such as such as character segmentation. A natural extension would thus be to classify unsegmented data.

References

[1] The cabinet of japan. jōyō kanji table. `http://kokugo.bunka.go.jp/kokugo_nihongo/joho/kijun/naikaku/kanji/`.

[2] Fujitsu achieves 96.7% recognition rate for handwritten chinese characters using ai that mimics the human brain. `http://www.fujitsu.com/global/about/resources/news/press-releases/2015/0917-01.html`, 2015.

[3] Electrotechnical laboratory character database. `http://etlcdb.db.aist.go.jp/`, 2016.

[4] Keras. `https://github.com/fchollet/keras`, 2016.

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[6] T. Agui, H. Takahashi, M. Nakajima, and H. Nagahashi. Recognition of handwritten katakana in a frame using moment invariants based on neural network, 1991.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CoRR*, abs/1502.01852, 2015.

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[9] I.-J. Kim and X. Xie. Handwritten Hangul recognition using deep convolutional neural networks. *IJDAR*, 18(1):1–13, Sept. 2014.

[10] F. Kimura, T. Wakabayashi, S. Tsuruoka, and Y. Miyake. Improvement of handwritten Japanese character recognition using weighted direction code histogram. *Pattern Recognition*, 30(8):1329–1337, Aug. 1997.

[11] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. *NIPS*, 25:1097–1105, 2012.

[13] K. I. Maruyama, M. Maruyama, H. Miyao, and Y. Nakano. Handprinted hiragana recognition using support vector machines. In *Frontiers in Handwriting Recognition, 2002. Proceedings. Eighth International Workshop on*, pages 55–60, 2002.

[14] N. Otsu. A Threshold Selection Method from Gray-Level Histograms. *SMC*, 9(1):62–66, 1978.

[15] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv*, Sept. 2014.

[16] 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*, 15:1929–1958, 2014.

[17] N. Tsuchiya, S. Ozawa, and S. Abe. Training three-layer neural network classifiers by solving inequalities. In *Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on*, volume 3, pages 555–560 vol.3, 2000.

[18] F. Yin, Q.-F. Wang, X.-Y. Zhang, and C.-L. Liu. ICDAR 2013 Chinese Handwriting Recognition Competition. *2013 12th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1464–1470, 2013.

[19] B. Zhu, X.-D. Zhou, C.-L. Liu, and M. Nakagawa. A robust model for on-line handwritten japanese text recognition. *IJDAR*, 13(2):121–131, Jan. 2010.