# Kuzushiji Character Recognition

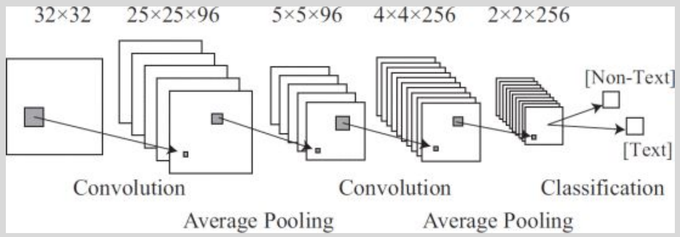
Anthony Le & Nathan Traxle; {antle1, nathant4} @stanford.edu
Stanford University - CS 229: Machine Learning

## Motivation

Kuzushiji is an ancient Japanese style of cursive handwriting that has fallen out of use in modern society. The surviving texts written in kuzushiji can only be read by about .01% of Japanese natives. We plan to take thousands of these texts, locate kuzushiji characters on the page, and correctly identify the corresponding modern japanese character. We are interested in this project as it could unlock thousands of stories, records, and texts that were previously hidden from a vast majority of society.

## Goal

Our goal was to build a model that could accurately transcribe ancient Kuzushiji into contemporary Japanese characters. This is a problem of object detection and classification. The model trained on several thousand pre-labeled images of ancient Kuzushiji scripture. In this project we attempt to use deep neural nets to directly estimate the contemporary japanese character that corresponds with each Kuzushiji character on the testing images.

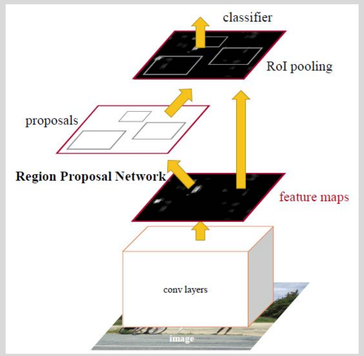



## Data

We used the Kuzushiji Recognition competition on Kaggle for our training and testing dataset. This dataset contained 3881 labeled images of Kuzushiji scripture. These labels gave the label of the selected Kuzushiji character's translation as well as the coordinates and dimensions of the bounding box containing the single character. The combined raw data totals was about 3.28 GB. Some of the pictures did not have any Kuzushiji characters on them, so we deleted these training examples. The original pictures were had resolutions about the suggested 720 x 1280 pixels, so we scaled down the resolution of our training pics in order to make the model train faster.

## Models

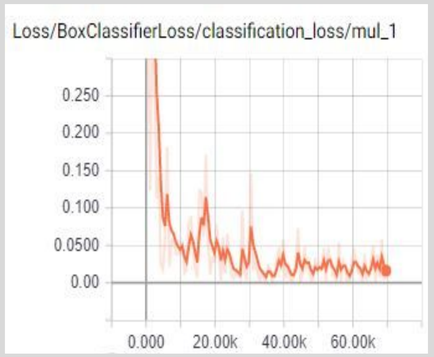
We used two different object detection models supplied by the Tensorflow Object Detection Model Zoo API Github repository [2]. We chose to use a deep neural net because text recognition was our main concern and there are published results about successful modeling techniques utilizing the power of CNN's [1]. Our first model the SSD MobileNet trained quickly but performed much worse than the Faster R-CNN Inception V2 model. The R-CNN model took significantly longer to train however the performance was much better.




- SSD MobileNet
- Faster R-CNN Inception V2

## Results

We found that the MobileNet was not robust enough for our use case, however the Faster R-CNN did a much better job of detecting and translating the characters. Our current implementation still is not a good enough model to effectively translate the text. A future addition to this project could be to use a separate neural network to rework the classifications based off of the Japanese Grammar.




| Model | SSD MobileNet | Faster R-CNN Inception V2 |
|---|---|---|
| Training $F_1$ Score | .58 | .71 |
| Test $F_1$ Score | .55 | .69 |

$$F_1 = \left( \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}.$$

## References

[1] T. Wang, D. J. Wu, A. Coates, A. Y. Ng, End-to-end text recognition with convolutional neural networks. *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012).*, 3304-3308 (2012).

[2] Tensorflow. "Tensorflow Detection Model Zoo." GitHub, Google, 2019,