Fonts display a large amount of stylistic variance; as a result, the task of creating new ones is a labor-intensive process usually reserved for skilled artists and designers. In this paper, we examine methods of generating characters for a language set from fonts which lack them—methods which, with refinement, could serve as useful tools for designers looking to “internationalize” existing and in-progress fonts quickly and easily.

We present neural networks for three tasks: discrimination between in-font and not-in-font using a set of four basis characters (AHQU), generation of all 26 Latin uppercase characters given basis characters, and generation of 46 Japanese hiragana given basis characters.

Data

We adapted a script [1] to produce 64x64 resolution, 8-bit grayscale images of individual characters while preserving relative vertical alignments. The fonts were taken from the Google Fonts repo, which offers 2908 free and open-source fonts in various styles and weights. Of these, there are only eight font families that support Japanese.

Model

For the first task, we tried two “tower” architectures [2] one with densely connected layers in the towers, as well as one with convolutional layers. We also developed one “shared” architecture with convolution, to see if individual filters extracting similar features from all input characters could be useful to the network.

For the second, we took a multitask approach, where the most descriptive encoding from the first task is fed through shared layers generating all characters. We experimented with numbers of neurons, layers, and convolutional layers.

In the third, we attempted transfer learning by using the weights and network developed in the second task, and training it on the smaller set of Japanese fonts.

Results & Discussion

In the discrimination task, the tower architecture achieved 89.76% accuracy on a test set of 332 examples, with erroneous classifications leaning towards false negatives.

In multitask learning, adding convolutional layers gets rid of grainy effects in generated letters. Adding more neurons results in an improvement in MSE loss, but adding more layers has less of a beneficial effect.

Transfer learning works well in the third task, with an improvement in training loss and validation set loss.

Predicting fonts’ Japanese sets worked well when the fonts were similar to ones we trained on. Common features can be seen between basis and output.

Conclusions

Limitations: similar fonts can have distinguishing marks, some fonts are not represented in Japanese (e.g., the monospaced-serif font Courier New or the more hand-drawn Comic Sans).

Interesting avenues for future work:
- Generating more characters, modified/larger networks, more fonts from more sources
- VAEs or GANs, SVG decoders for translating encoding into scale-invariant vector

Footnotes


Link to code: github.com/garrickf/font-gen

Special thanks to the CS230 course staff for inspiration and support!