Supplement: Hangul Fonts Dataset: a Hierarchical and Compositional Dataset for Investigating Learned Representations

Jesse A. Livezey[1,2] **Ahyeon Hwang**[3] **Jacob Yeung**[1]

Kristofer E. Bouchard[1,2,4,5]
[1]Biological Sciences and Engineering Division, Lawrence Berkeley National Laboratory, Berkeley, CA
[2]Redwood Center for Theoretical Neuroscience, University of California, Berkeley, CA
[3]Mathematical, Computational and Systems Biology, University of California, Irvine, CA
[4]Helen Wills Neuroscience Institute, University of California, Berkeley, CA
[5]Computational Research Division, Lawrence Berkeley National Laboratory
Berkeley, CA
{jlivezey, kebouchard}@lbl.gov, ahyeon.hwang@uci.edu, jacobyeung@berkeley.edu

A Generating the Hangul Fonts Dataset

To increase the reproducibility and potential for future development of the Hangul Fonts Dataset (HFD), we are releasing the code used to generate and normalize the HFD. The code is released under a Lawrence Berkeley National Labs BSD variant license. The code and instructions for reproducing the HFD, including a docker image and Dockerfile which specifies the environment, can be found at `https://github.com/BouchardLab/HangulFontsDatasetGenerator`. The `hangul_analysis` folder contains a separate repository that contains the analysis and plotting code and notebooks. The software and curated open font files are available here.

This code (with default settings) generates the following

- `pngs/*`: folders per font and fontsize (default 24) containing png files of each block
- `pdfs/*`: folder per font containing pdfs which show all possible blocks
- `h5s/*`: folder per font containing HDF5 files which contain the following (key: variable) pairs
  - `images`: block images at max size across blocks within the font (not used directly)
  - `images_median_shape`: block images at median size across fonts (used for training/analysis)
  - `labels`: concatenated IMF class labels
  - `{initial, medial, final}_geometry`: IMF geometry hierarchy class labels
  - `all_geometry`: class labels based on the product of the IMF geometry class labels
  - `atom_bof`: IMF atoms features
  - `atom_mod_rotations_bof`: IMF atoms mod rotations features

Submitted to the 35th Conference on Neural Information Processing Systems (NeurIPS 2021) Track on Datasets and Benchmarks. Do not distribute.

input : Hangul Unicode glyph `text_files`, Hangul `font_files`, `fontsize`
output : Hangul Images
for *font in font_files* **do**
  for *block in text_files* **do**
    Convert `block` to `block_image` using `convert` [1] with `fontsize`;
  end
  Resize `block_image` to max image size across `block_images`;
  if *initial, medial, final glyphs available* **then**
    Convert `glyph` to `glyph_image` using `convert` [1] with `fontsize`;
  end
end
Resize all `block_images` to median image size across fonts

Algorithm 1: Pseudocode for Creating Hangul Images. The outer loops are over the fonts. The inner loop is over the blocks. Since the image sizes of blocks were originally different across fonts, the blocks were resized to the max image size across blocks in the font. After the outer loop, the image sizes of blocks across fonts were different, so the blocks were resized to the median size across fonts. For our analysis, we used font size 24.

B The Hangul fonts dataset: summary statistics and visualization

B.1 Font families

For certain fonts, bold and light versions of the same font are included, a "natural" but fairly regular source of variation across fonts (Fig 1).

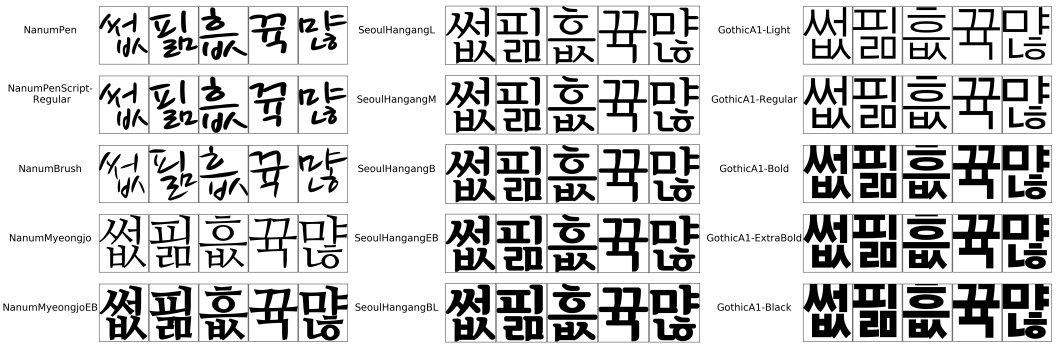

Figure 1: **Variation across different font families: Nanum, SeoulHangang, GothicA1.** The same five randomly chosen blocks are displayed for each font. SeoulHangang and GothicA1 fonts are ordered thin to bold from top to bottom.

We also performed correlations across fonts to analyze how fonts differ from one another. Fig 2 shows the dendrogram that results from hierarchical clustering of the correlation matrix using Ward's method. GothicA1-Bold, GothicA1-SemiBold, GothicA1-Regular, GothicA1-Light, GothicA1-Thin, GothicA1-Black, GothicA1-Medium, and GothicA1-ExtraBold all fall under the same GothicA1 category.

B.2 Summary statistics of dataset

Various statistics were calculated on fontsize 24 images within a single and across all fonts. The mean, median, and standard deviation of the images were taken in Fig 3. This was done for all blocks in all fonts, all blocks in a single font, and a single block in all fonts. All three statistics for all fonts all blocks and one font all blocks preserve the block structure of the images, whereas '가' is clearly shown for all fonts single block across the statistics. Fig 4A shows a histogram of the pixels of all the images within a font for all 35 fonts. The histograms are very similar across the four fonts highlighted in the legend. Fig 4B shows a histogram of pixels within a font across all blocks for all 35 fonts. The Frobenius norm is taken for all blocks to study how blocks differ within a font. NanumMyeongjo

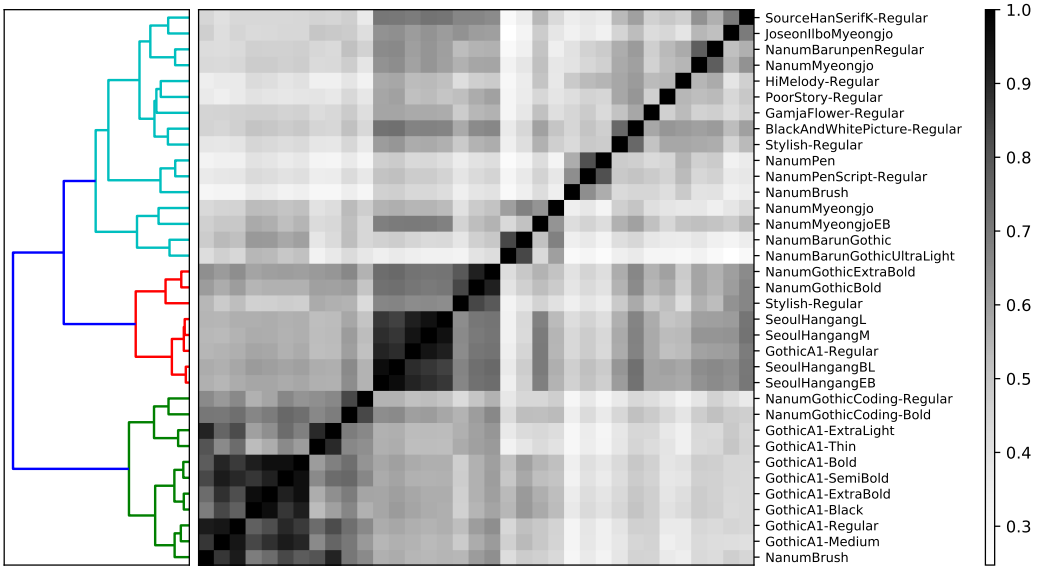

Figure 2: **Distance matrix and dendrogram showing clustering of similar fonts.** Distinct groups such as Nanum, SeoulHangang, and GothicA1 cluster together.

and NanumBrush are similar fonts as they have overlapping block norms. GothicA1-Regular, which resembles computer-type fonts, has the thinnest distribution as its blocks do not differ greatly.

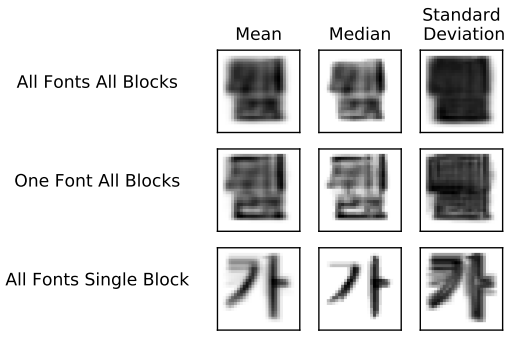

Figure 3: **Summary statistics of images in Hangul dataset.** The mean, median, and standard deviation is taken for all blocks in all fonts, all blocks in a single font, and a single block '가' in all fonts.

B.3 Dimensionality Reduction: PCA, ICA, NMF

Linear models were trained on individual fonts and the learned dictionaries are shown in Fig 5.

B.4 Classification accuracy

The supervised deep networks were trained on 3 tasks: initial, medial, and final (IMF) classification. Here we report the test-set accuracy for logistic regression as well as the best deep networks (selected by the validation accuracy). Logistic regression had accuracies of $57.6 \pm 2.7\%$, $70.8 \pm 2.6\%$, and $73.0 \pm 3.1\%$ respectively for the IMF tasks. Deep networks had accuracies of $98.5 \pm 2.0\%$, $98.0 \pm 2.6\%$, and $97.4 \pm 3.3\%$ respectively for the IMF tasks. Chance accuracy is 5.3%, 2.6%, and 3.1% respectively for the IMF tasks.

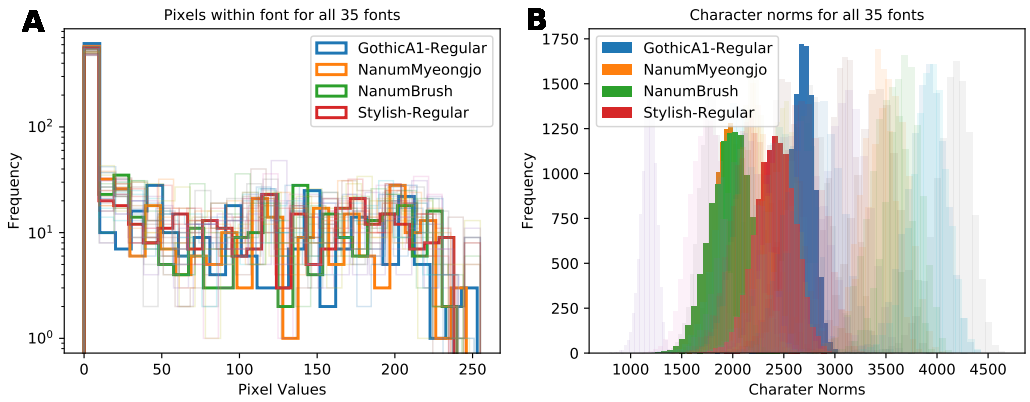

Figure 4: **Characteristics of pixels and blocks for each font.** Four fonts are shown clearly while the rest are shown in a lighter shade. **A** Outlines of the histograms of pixels within a font. **B** Filled-in histograms of pixels within font across all blocks.

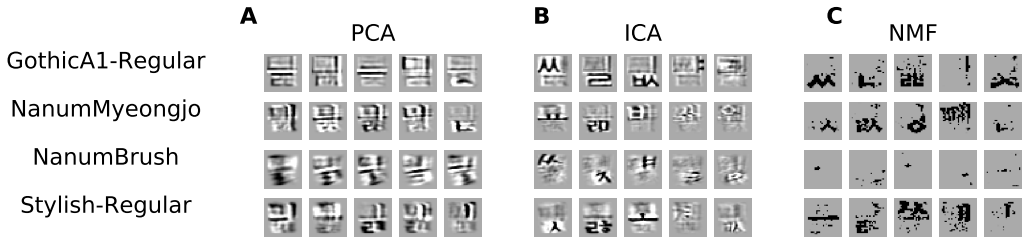

Figure 5: **Dictionary elements learned from four fonts: GothicA1-Regular, NanumMyeongjo, NanumBrush, Stylish-Regular. A** PCA elements: first five PCs that explain the most variance are plotted from left to right. **B** ICA elements: first 3 ICs show distinct glyphs, last 2 are randomly chosen ICs **C** NMF elements: first three best factors are chosen, last 2 are randomly chosen. NanumBrush had no distinct components.

B.5 Disentangled reconstructions from β-VAE

β-VAEs with $\beta < 1$ had disentangled reconstructions (shown in Fig 6) that were qualitatively similar to disentangled reconstructions from β-VAEs with $\beta > 1$.

B.6 UMAP visualization

Fig 7A-C shows the result of applying UMAP to a single font's images, GothicA1-Regular, with initial, medial, and final labels. Individual glyphs are plotted with a red kernel density estimate in the background. Fig 7B shows the best clustering with regards to the geometry of the glyphs. The more verticially-oriented glyphs (ㅏ, ㅑ, ㅕ, ㅓ) cluster together in the right side while the more horizontally-oriented glyphs (ㅗ,ㅡ,ㅜ,ㅠ) cluster in the left. In Fig 7C several of the duplet glyphs embed in the same location, suggesting similarity in the duplet structure.

Fig 8A-C shows the result of applying UMAP to the images with initial, medial, and final geometry labels. Actual points are plotted rather than glyphs. Fig 8B shows the best separation among the 5 geometric types as they are each distinct, and hence affect the overall structure of the block. For example, right-single and right-double medial geometric types are always placed on the left region of the blocks. In contrast, initial and final geometry types which include none, single, or double do not drastically influence the greater structure of the block. Single and double geometric types have very similar embeddings in both the initial and final geometry plots.

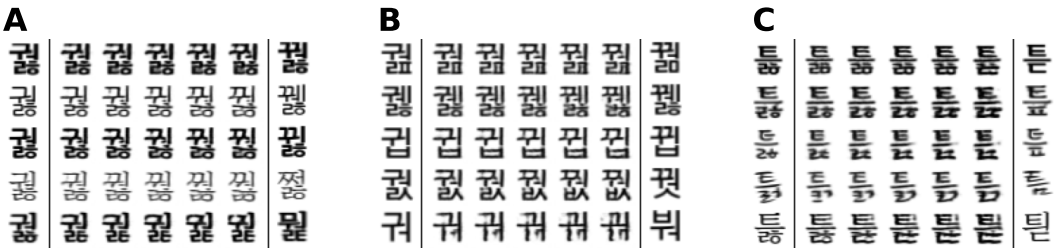

Figure 6: **Disentangled reconstructions from $\beta-$VAE.** Latent traversals of a single latent variable. The right column is the input image, middle columns are the traversals, and final column is the block the traversals appear to morph into. **A, Initial Across Fonts:** First three rows are similar traversals of one block across fonts. Fourth row shows an entangled traversal changing initial and medial glyphs. Final row shows an entangled traversal changing initial and final glyphs. **B, Initial Across Blocks:** First three rows are similar traversals of blocks (with the same hierarchy across medial and final glyphs) in same font. Fourth row is an entangled traversal changing initial and final glyphs. Final row shows a traversal of a block without a final glyph (different hierarchy) - it is not very interpretable. **C, Final Across Fonts:** First four rows are similar traversals of one block across increasingly naturalistic fonts. Final row is an entangled traversal between medial and final glyphs.

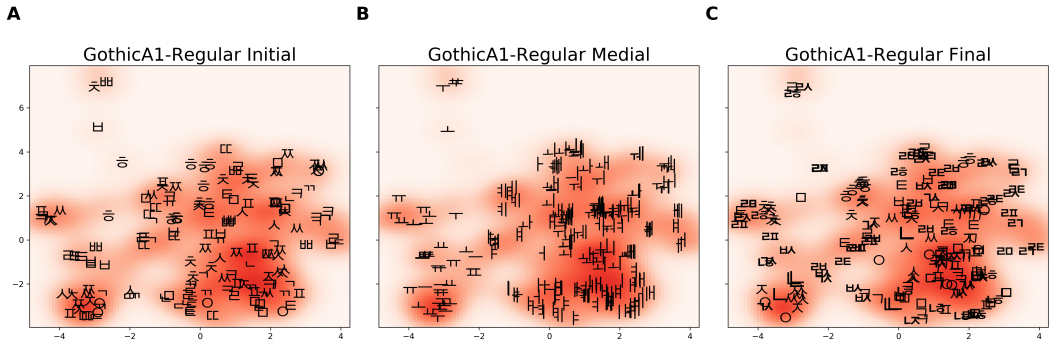

Figure 7: **GothicA1-Regular UMAP embedding of Initial, Medial, and Final Labels.** The kernel density plot indicated by the red coloration is the same for all three figures. Darker shade means higher density of blocks. **A** 19 initial glyphs, 10 of each plotted **B** 21 medial glyphs, 10 of each plotted **C** 28 final glyphs, 8 of each plotted

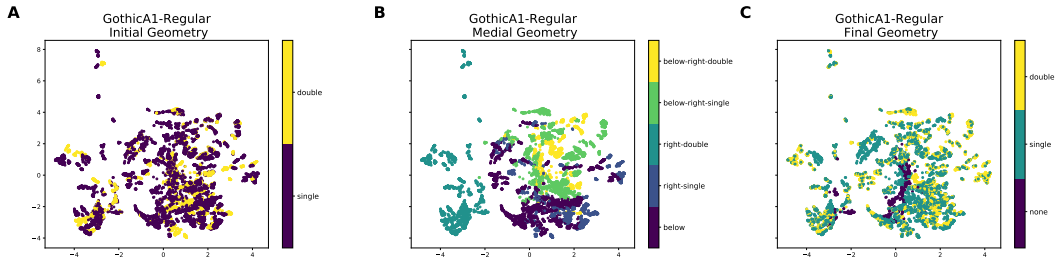

Figure 8: **GothicA1-Regular UMAP embedding of Initial, Medial, and Final Geometry Labels.** **A** 2 initial geometric types **B** 5 medial geometric types **C** 3 final geometric types

C Hangul definitions and linguistic meanings

Block: the visual unit in Hangul writing which combines initial, medial, and final (IMF) glyphs into a roughly square area. Each block corresponds to a spoken syllable. In written Korean, blocks are concatenated to form words. Each block can be annotationed with IMF class labels, geometric hierarchy class labels, and compositionality binary features.

Glyph: the smallest visual unit being considered in this work. Each glyph represents a consonant or vowel phoneme. Glyphs are composed into blocks to form syllables. Several glyphs are shared across initial, medial, or final positions.

Atomic glyph or **atom**: The unique set of glyphs that are used in initial, medial, and final glyphs.

D Methods continued

D.1 KMeans clustering accuracy

For a dataset of sample size n and a representation $h \in \mathbb{R}^{n \times d}$ we want to compare a clustering of h and a categorical generative variable $y \in \{1, \ldots, M\}$. First, h is clustered into M clusters with k-means (with k=M) or from cutting the dendrogram to give M clusters. Given this clustering, each sample is assigned a class label $\hat{p}$. We then have to find the best alignment between the M cluster labels and the M generative variable labels. To do this we form a similarity matrix $S \in \mathbb{R}^{M \times M}$. To calculate S_{ij}, we first form the set A which contains the samples labeled $\hat{p}_i$ and B which contains the samples labeled y_j. Then the similarity is the cardinality of the intersection of the sets divided by the cardinality of the union of the sets: $S_{ij} = \frac{|A \cap B|}{|A \cup B|}$, that is, given all samples labeled with either label, what fraction of them are labelled as both. We then use the Hungarian method [2] to optimally pair the generative labels y with a permutation of the cluster labels $\hat{y}$ using this similarity matrix. If the cluster labelling is an exact permutation, the clustering accuracy will be 100%, and chance for a random relabelling.

E Hyperparameters

All supervised networks were trained with 3 hidden layers of the same dimensionality and ReLU nonlinearities. Table 1 lists the hyperparameters that were randomly sampled and their ranges. $\dim_i$ and $\dim_0$ indicate the input and output dimensionality of the data and task.

Unsuperised $\beta-$VAE encoders were trained with 7 $2-$d convolutional layers of the same channel depth and kernel size, ReLU nonlinearities, and one hidden layer of the same dimensionality. The decoders were trained with one hidden layer of the same dimensionality, transposed $2-$d convolutional layers of the same channel depth and kernel size, and ReLU nonlinearities. The $\gamma < 1$ networks were trained with the training regime from [3]. The $\gamma > 1$ networks were trained with the same training regime as $\gamma < 1$ for 125 epochs (using $\gamma < 1$), after which the γ term was increased exponentially, similar to the training regime proposed by [4].

Table 1: Hyperparameters for dense networks

Name	Type	Range/Options
Init. momentum	Float	.5
Learning rate reduction on plateau	float	.5
Epochs of patience for learning rate reduction	int	10
Epochs of patience for early stopping	int	10
Dense layer size	int	$\min(\dim_i - 1,\ \dim_o \times 10) : 2 \times\ \dim_i$
$\log_{10}$ learning rate	float	-6 : 1
$\log_{10}$(1-momentum)	float	-2 : -.00436 (momentum=.01)
$\log_{10}$ L_2 weight decay	float	-6 : 1
Batch size	int	32 : 512
Input dropout rate	float	.1 : .99
Input dropout rescale	float	.1 : 10
Hidden dropout rate	float	.1 : .99
Hidden dropout rescale	float	.1 : 10

Table 2: Hyperparameters for β-VAE networks

Name	Type	Range/Options
Init. momentum	Float	.5
Learning rate reduction on plateau	float	.5
Epochs of patience for learning rate reduction	int	5
Epochs of patience for early stopping	int	125
$\log_{10}$ learning rate	float	-5 : -2
$\log_{10}$(1-momentum)	float	-2 : -.00436 (momentum=.01)
$\log_{10}$ L_2 weight decay	float	-6 : 1
$\log_{10}$ $\gamma(< 1)$	float	-4 : -1
$\log_{10}$ $\gamma(> 1)$	float	1 : 3
log KL-divergence target (C) max value	float	20 : 50

F Datasheet for the Hangul Fonts Dataset

We follow the Datasheets for Datasets [5] recommendations for documenting datasets.

F.1 Motivation

For what purpose was the dataset created?
Who created the dataset and on behalf of which entity?
Who funded the creation of the dataset?

The Hangul Fonts Dataset was created as a benchmark machine learning dataset for investigating whether and how hierarchy and compositionality are found representation learning methods with a focus on deep networks. It was created by Jesse Livezey, Ahyeon Hwang, and Kristofer Bouchard at Lawrence Berkeley National Laboratory (LBNL) and was funded through the Laboratory Directed Research and Development Program at LBNL.

F.2 Composition

What do the instances that comprise the dataset represent? Does the dataset contain all possible

instances or is it a sample of instances from a larger set? What data does each instance consist of?
How many instances are there in total?
Is there a label or target associated with each instance?
Are there recommended data splits?
Are there any errors, sources of noise, or redundancies in the dataset?
Is the dataset self-contained, or does it link to or otherwise rely on external resources?
Does the dataset contain data that might be considered confidential? Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? Does the dataset relate to people?

The Hangul Fonts Datasets consists of all possible blocks for each font along with a set of labels and features for each block which are common across all fonts. We provide scripts to render each block and font into an image and store them together in an HDF5 file for each font and fontsize specified. In addition, we have curated a set of open Hangul fonts, most of which are licensed under a Open Font License (see http://scripts.sil.org/OFL). Licensing information for each font has been included with the font files.

Each font has 11,172 possible blocks and we have curated 35 fonts for a total of 391,020 possible images. The spoken consonants and vowel correspond to a set of "initial", "medial", and "final" labels for each block with 19, 21, and 28 classes respectively. In addition, each block can also be assigned a hierarchy label based on the block geometry. Finally, the individual glyphs (atoms) are composed across the initial, medial, and final locations along with potential scalings, translations, or rotations. We use these relationship to define a bag-of-atoms set of features.

Rendering the fonts to images transforms them from a vector-based representation to a pixel-based representation which will introduce some smoothing or loss of information depending on the font size used. We find that there is not much gain in classification accuracy beyond font size 24 which is the font size we recommend and generate by default.

The dataset relies on a set of open font files, which we have curated. Otherwise, the dataset is self contained and we provide scripts for users to generate their own copy of the dataset. Using the scripts, users are also able to generate datasets with varying fontsizes or new open or closed fonts.

The dataset does not contain any data that might be considered confidential or harmful. Although by their nature fonts and the Hangul writing system relate to people in their creation and use, the instances in this dataset are not related to data from any people.

F.3 Collection Process

How was the data associated with each instance acquired? What mechanisms or procedures were used to collect the data?
If the dataset is a sample from a larger set, what was the sampling strategy?
Who was involved in the data collection process and how were they compensated?
Over what timeframe was the data collected?
Were any ethical review processes conducted?

The open font files which are used in conjunction with the library to generate the images were curated from sever sources of Hangul fonts (see README for details). All available, open, de-duplicated fonts were used in the final dataset. Jesse Livezey and Ahyeon Hwang curated the font files and wrote the library and were paid as part of a postdoc and a research assistant position, respectively, at LBNL. The fonts were curated and the library was written between October 2017 and February 2019. No ethical review process was conducted.

F.4 Preprocessing/cleaning/labeling

Was any preprocessing/cleaning/labeling of the data done?
Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data?
Is the software used to preprocess/clean/label the instances available?

The vector font files are converted into a set of images. There are choices in terms of font size and final image standardization made and documented in the library. The initial, medial, and final labels, the geometry labels, and the atom labels are based on the structure of the Hangul writing system. The identification of different atoms across rotations was done by hand. In lieu of releasing the binary image files, we provide the library used to create the images along with a Docker image which can be used to exactly replicate the images.

F.5 Uses

> Has the dataset been used for any tasks already?
> Is there a repository that links to any or all papers or systems that use the dataset?
> What (other) tasks could the dataset be used for?
> Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?
> Are there tasks for which the dataset should not be used?

The dataset has not been used outside of its original preprint/publication. The dataset could be used for other types of generative and supervised machine learning analysis of Hangul fonts. There were preprocessing choices made which may impact future use, but the library used to generate the images is being released. We cannot think of potential tasks for which the dataset should not be used.

F.6 Distribution

> Will the dataset be distributed to third parties outside of the entity on behalf of which the dataset was created?
> How will the dataset will be distributed (e.g., tarball on website,API, GitHub)?
> When will the dataset be distributed?
> Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use(ToU)?
> Have any third parties imposed IP-based or other restrictions on the data associated with the instances?
> Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?

The library used to generate the images from the font files is distributed on Github under an open source license. We have also curated a set of open font files. The library is distributed under a Lawrence Berkeley National Labs BSD variant license license. There are no restrictions or controls on the dataset or instances.

F.7 Maintenance

> Who is supporting/hosting/maintaining the dataset?
> How can the owner/curator/manager of the dataset be contacted?
> Is there an erratum?
> Will the dataset be updated?
> If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances?
> Will older versions of the dataset continue to be supported/hosted/maintained?
> If others want to extend/augment/build on/contribute to thedataset, is there a mechanism for them to do so?

The dataset will be hosted on Github (with an archive or the publication version on Zenodo) and maintained by authors who can be contacted through Github and/or their institutional email addresses. The dataset will be updated to correct any errors which will be noted in a erratum. Future versions of the dataset will be extended to work with additional open fonts as they become available and contributions/corrections from others. Contributions/corrections can be made through Github. Old versions will be tagged and maintained as applicable. The dataset does not relate to people.

References

[1] LLC ImageMagick Studio. Imagemagick, 2008.

[2] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2 (1-2):83–97, 1955.

[3] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

[4] Alexander Amir Alemi, Ben Poole, Ian S. Fischer, Joshua V. Dillon, R. Saurous, and K. Murphy. Fixing a broken elbo. In *ICML*, 2018.

[5] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé III, and Kate Crawford. Datasheets for datasets. *arXiv preprint arXiv:1803.09010*, 2018.