FontGen

Generating fonts using a neural net

Matt Gaikema

In[1]:=
```
ClearAll["Global`*"]
SetDirectory[NotebookDirectory[]];
```

I spend a lot of time trying to identify and download fonts, so a program that could do it for me would be wonderful. There are many academic papers on the subject, but none of them have published their code, so I decided to implement it myself and learn more about deep learning along the way.

Data

The data is a whopping 13+ gigabytes and was scraped from the internet by someone else. It can be downloaded here.

```
Import["fonts.hdf5", {"Dimensions"}]
```

```
⟨|/fonts → {56443, 62, 64, 64}|⟩
```

I debugged the neural net using a much smaller dataset, consisting of my personal collection of fonts. The smaller dataset was created by running a script written by the same guy who got the original data.

```
Import["fonts.small.hdf5", {"Dimensions"}]
f = Import["fonts.small.hdf5", {"Datasets", "fonts"}];
Image[f[[1]][[1]]]
```

```
⟨|/fonts → {21, 62, 64, 64}|⟩
```

Network

Each letter is 64x64 pixels.

```
(* We use 4 letters as input... *)
4 * 64^2
(* ... and get 62 letters as output. *)
62 * 64^2
```

```
16 384
```

```
253 952
```

There are $4*64^2 = 16,384$ input neurons, and $62*64^2 = 253,952$ output neurons. The letters chosen for the inputs are “B”, “A”, “S”, and “Q”.

```
(* http://bit.ly/2oeHyZX *)
Range[0, 25];
letters = AssociationThread[ToUpperCase[#] & /@ Alphabet[], %];
Lookup[letters, {"B", "A", "S", "Q"}]
Image[f[[1]][[# + 1]] & /@ %
```

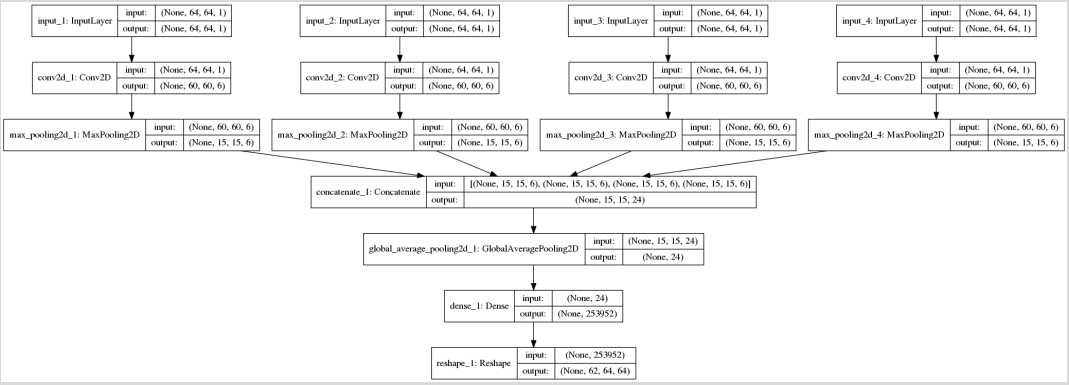
```
{1, 0, 18, 16}
```

Model

In[9]:=

```
Import["img/model.png"]
```

Out[9]=



Running `model.save_weights()` saves the model and its weights to an HDF5 file, which can be loaded to recreate the model (see the documentation).

In[10]:=
```
Import["model.hdf5", {"Dimensions"}]
```

Out[10]=
```
⟨| /conv2d_1/conv2d_1/bias:0 → {6}, /conv2d_1/conv2d_1/kernel:0 → {5, 5, 1, 6},
 /conv2d_2/conv2d_2/bias:0 → {6}, /conv2d_2/conv2d_2/kernel:0 → {5, 5, 1, 6},
 /conv2d_3/conv2d_3/bias:0 → {6}, /conv2d_3/conv2d_3/kernel:0 → {5, 5, 1, 6},
 /conv2d_4/conv2d_4/bias:0 → {6}, /conv2d_4/conv2d_4/kernel:0 → {5, 5, 1, 6},
 /dense_1/dense_1/bias:0 → {253 952}, /dense_1/dense_1/kernel:0 → {24, 253 952} |⟩
```

Evaluation

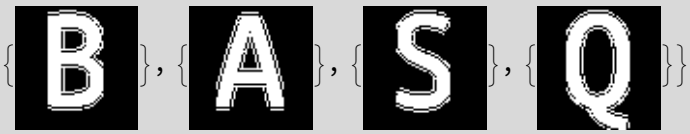
In[3]:=
```
(* Transforms a list of image arrays to a list of images. *)
displayFont[f_]:=Map[Map[Image,#]&,f];
```

I tested the model on the font Ubuntu Mono, which is available for download here.

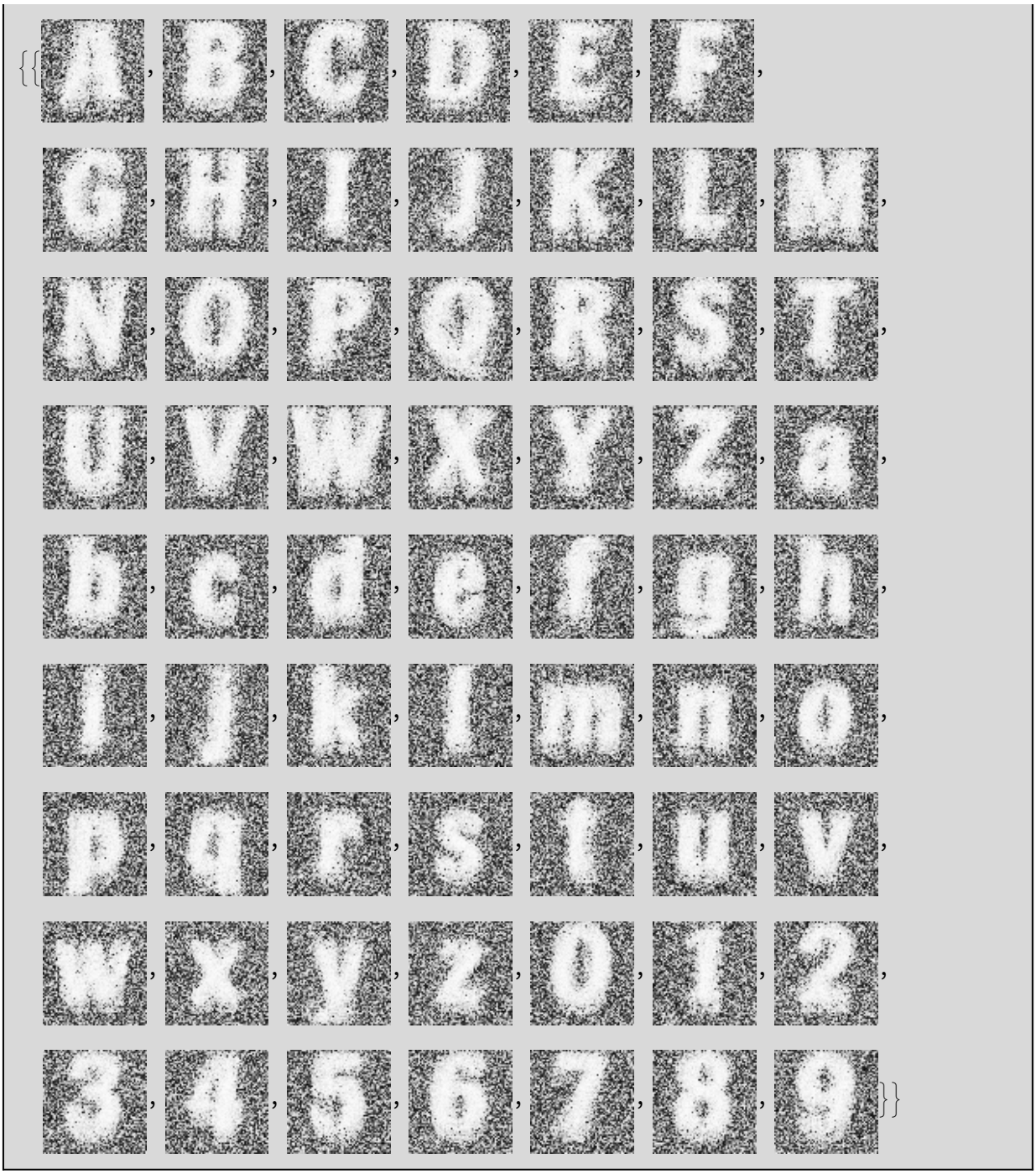
In[11]:=
```
Import["test.hdf5", {"Dimensions"}]
testInput = Import["test.hdf5", {"input"}];
displayFont[testInput]
output = Import["test.hdf5", {"output"}];
displayFont[output]
```

Out[11]=
```
⟨| /input → {4, 1, 64, 64, 1}, /output → {1, 62, 64, 64} |⟩
```

Out[13]=



Out[15]=



You can make out the individual letters, but clearly this isn't an exact recreation. Since I'm still relatively new to deep learning, I'm fairly pleased with these results.

Acknowledgements

- https://erikbern.com/2016/01/21/analyzing-50k-fonts-using-deep-neural-networks.html
- https://research.google.com/pubs/pub45369.html

- Fonsi Bonilla, for answering all of my questions.