TYPO
TYPOGRAPHY

Topography Typography: Detecting Letterforms from Satellite Imagery

Kara Lynn Bressler*
Department of Computer Science, Princeton University

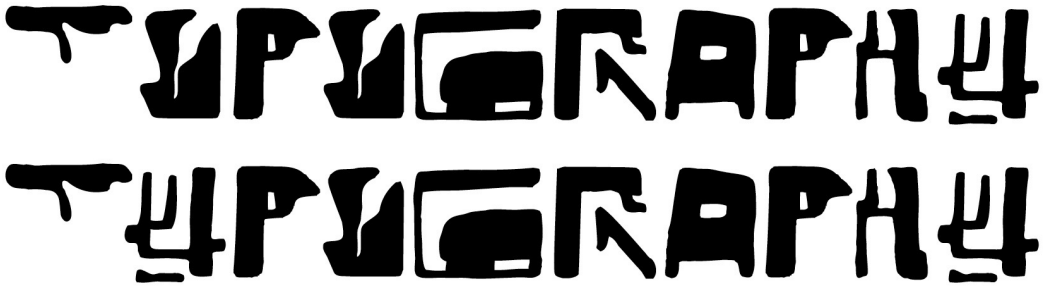


Figure 1: *Title of the paper in the font PRINCETON, New Jersey.*

Abstract

From screens to stop signs, we see typographic language everywhere. The pattern of written language guides us through the world. In this paper, I propose and implement a method for seeing the typographic forms in the topography of our physical worlds. I use the Extended MNIST dataset containing letters and numbers to train a recurrent convolutional neural network which detects the character forms in a satellite image. This project results in the generation of over one hundred vectorized fonts as well as a system for creating an infinite number of fonts from satellite imagery.

Keywords: typography, satellite imagery, object detection

Introduction

This project serves to be at the fringe of computer science research as you will see mimicked in the writing style and structure of this paper. Please keep this in mind as you consume this paper and the contents held within.

Origin Story

Let me set the scene: In November 2017, I was flying home to Naples, Florida for Thanksgiving break, peering out the airplane window at the landscapes below. All of a sudden, I settled my eyes on a large body of water in the shape of a cowboy boot with spurs. The resemblance was uncanny. Slowly, we flew past the lake, but I sat transfixed in my seat thinking about this sight.

During that semester at Princeton, I was taking a Visual Arts class called *Typography* taught by David Reinfurt. During my coursework, I was thinking about and working with the deconstruction of letterforms. Having just seen this extraordinary boot-shaped lake, I racked by brain to figure out how I might fit this visual image into my artistic practice. It occurred to me before we landed that I wanted the ability to find aerial letterforms for any given location, similar to my experience with the image of the boot.

I wrote the idea in my journal and stored it away for a couple months, returning to it for my junior independent work in this paper.

*e-mail:karab@princeton.edu

Goal

The goal of this project is to create a system for generating vectorized letterforms from a location's topography. Given satellite imagery, I want to find the alphanumeric characters of the image algorithmically. As a final product, I want the ability to type with these forms in a somewhat-legible, complete typeface.

Related Work

I would highly recommend for you to spend ten minutes on the search engine of your choice and search **pareidolia**. You should find an incredible visual collection, and the concept of pareidolia has reshaped my everyday visual experiences. This psychological phenomenon describes the ability to observe patterns in random data. Today we see pareidolia manifesting itself as sightings of faces or common objects in impossible environments with experiences ranging from Hermann Rorschach's inkblot test to the Viking missions' 1976 satellite image "Face on Mars." [Phillips 2001]

My favorite example of pareidolia comes from my home state of Florida where in 2004 Diana Duyser sold on eBay for $28,000 a 10-year-old grilled-cheese sandwich with the the image of the Virgin Mary seared into the bread:

> *Diana Duyser made the sandwich with white bread from Publix and Land 'O Lakes American cheese in 1994. She took a bite, then looked down and saw the Virgin Mary staring up at her. Indeed, the image of a woman could be perceived in the charred bread, but it could have been a barroom portrait as much as the mother of G-d.* [Nolin 2015]

People today find entertainment in this phenomenon, spotting images of rabbits in the sky while cloud gazing or finding dogs in a plate of spaghetti while experimenting with Google's DeepDream algorithm which incorrectly detects and then exaggerates zoomorphic features in the input image. [Mordvintsev et al. 2015; Russon 2015]

Paul Elliman

Pareidolia is imperative to the life work of renowned typographer, graphic designer, and artist **Paul Elliman** who situates himself within the cross-section of typography and found objects. In 1995, Elliman began curating his typeface *Found Font* (also known as

Bits), a system of letters created by objects in the world where the letterforms can only be used once in any given publication of the script. As displayed in MoMA's *Ecstatic Alphabets / Heaps of Language* exhibit, the found letterforms comprise of zip ties, scissor handles, hoop earrings, and other various small objects. [Reinfurt et al. 2012; Wooldrage 2017]

The Aerial Bold Project

This scope of this project is admittedly quite niche. However, there has been some related research in the field of letterform detection from satellite imagery with Benedikt Groß and Joey Lee's **Aerial Bold Project**. This duo's approach comes from a major collaborative effort involving 140 people around the world (mostly living in North America and Europe). These participants spent over two years using an app to manually detect and label over 11,000 letterforms in cities. Groß and Lee proceeded to parse through the labeled images to find what they deemed to be the most believable and legible forms to create their alphabets for their predetermined locations. While they commissioned machine-learning research for character recognition in satellite imagery, their final product did not implement the proposed algorithm. [Groß and Lee 2016; Arawal 2015].

Approach

While visiting my eldest brother Liam in Austin, Texas in January 2018, I began developing my approach for this project and forming a blueprint for the resources I would need. We debated how to find the "best" alphabetic forms of a landscape, discussing using human intelligence to manually pan Google Earth and create a test set of obvious letterforms. We would then test whether the proposed algorithm could detect and correctly label the same obvious letterforms. I quickly decided against this approach.

As I later researched in the Aerial Bold Project, Groß and Lee entirely crowd-sourced their repository of over eleven thousand letterforms. This use of crowd-sourcing brings to light questions about global labor exploitation. Mechanisms in place today like Amazon Mechanical Turk outsource small "microtasks" to large populations of people on the Internet, making the process streamlined to create a dataset from scratch. However, this process does not take into consideration editing the collected datapoints or ethics informing the practice of outsourcing the completion of these "microtasks."

In order to curate a new, properly-sized dataset for this project, I would need to crowd-source the collection of the training and test images to create any valid model. I instead chose to use a pre-exisiting, relatively clean, and well-curated dataset – EMNIST. Below we will look at the history as well as the overall structure of this dataset.

Before EMNIST

Before we look at the EMNIST dataset, let us take a look at what came before.

In 1995, the United States' National Institute of Standards and Technology (NIST) created a novel database of handwritten characters called *NIST Special Database 19*. The characters were curated from nearly 3700 instances of the *Handwriting Sample Form* filled out by both employees of the American Census Bureau and American high school students. See *Figure A1* for an example of a completed form. Each of the over 800,000 characters in *Database 19* is represented by a binary black-and-white 128×128 pixel image fulfilling membership in one of the sixty-two following classes:

- Uppercase letters: $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \ldots, \mathbf{Z}\}$
- Lowercase letters: $\{\mathbf{a}, \mathbf{b}, \mathbf{c}, \ldots, \mathbf{z}\}$
- Digits: $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots, \mathbf{9}\}$

Database 19 splits its training data into characters written by American Census Bureau workers and testing data into characters written by the high school students. [Grother 1995] In an effort to create a representative testing set accommodating for the potential biases in between NIST's training and testing sets, *LeCun et al.* curated the MNIST dataset from *NIST Special Database 19*.

MNIST (the Modified National Institute of Standards and Technology database) is the canonical machine learning (ML) dataset. In fact in my junior fall, I encountered MNIST in two different computer science assignments introducing ML topics. Each of the 70,000 characters in MNIST are represented as smaller versions of those in the NIST database, depicted by an anti-aliased grayscale 28×28 pixel image fulfilling membership in one of the ten following classes: $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots, \mathbf{9}\}$. MNIST is split into training and testing sets, sized 60,000 and 10,000 respectively, which scramble together *NIST Special Database 19*'s training and testing data. [LeCun et al. 2013]

EMNIST

In 2017, *Cohen et al.* published an extended version of the MNIST dataset known as **EMNIST** (Extended MNIST). The two are, or course, pronounced the same. Samples in EMNIST are based in the same anti-aliased grayscale 28×28 pixel format as MNIST, all with a white character on a black background. This extension functions as another variant on the *NIST Special Database 19*, now containing all of the original sixty-two classes cleaned and arranged in multiple datasets including: ByClass, ByMerge, Balanced, Letters, Digits, and the original MNIST. The later four of these classes are balanced, meaning there is an equal number of samples in each class. For example in the Letters class, there are 5600 instances of the class *A* and there are 5600 instances of *B* and so forth. For this project I chose to use two of these clean balanced datasets:

- Letters: $\{\mathbf{A}$ and $\mathbf{a}, \mathbf{B}$ and $\mathbf{b}, \mathbf{C}$ and $\mathbf{c}, \ldots, \mathbf{Z}$ and $\mathbf{z}\}$
- Digits: $\{\mathbf{0}, \mathbf{1}, \mathbf{2}, \ldots, \mathbf{9}\}$

The Letters dataset contains 5600 instances for each of its twenty-six classes, and the Digits dataset contains 28,000 instances for each of its ten classes. The Digits class represents a smaller set of the original MNIST dataset, and we use this more manageable size without a statistically significant difference in testing error. [Cohen et al. 2017]

Implementation

We will now look at the process of the letterform detection, starting from curating the source material to running the material through a neural net to creating the final vectorized letterforms.

Location Curation

Given a specific place like a town, neighborhood, city, I first curate satellite imagery from Google Earth. By using the desktop application Google Earth Pro, I save a single image with the rough dimensions of 1200×700 pixels.

I then make slight adjustments to the image in order to emphasize the difference between "foreground" and "background." For all images, I crop out the Google Earth watermark. In images of mostly uniform color density and tone, I increase contrast and slightly

adjust the highlights and shadow properties in a basic photo processing application like Mac's default Photo. See an example of this transformation in *Figure A2* and *Figure A3*. For other images with relatively good contrast throughout the topography, I choose to leave the images in their raw state.

Setup

I write the scripts for this project in a set of Jupyter Notebooks (`jupyter.org`), an open-source project which easily runs my Python3 scripts. I host these notebooks directly on my computer. I primarily run two notebooks, one written to detect letters and the other to detect numerical forms in a given image.

Using a pre-existing dataset allows for guidance from past work. Especially using a dataset in the family of MNIST, I found plenty of online interest and experimentation with my dataset on the code-sharing website Github (`github.com`), used commonly by developers for version control and code distribution. I found *coopss*' off-the-shelf solution for the object detection of characters in EMNIST using a neural net, and I use this code as a template for my algorithm. [coopss 2016]

Image Processing

Before I give the neural net the input samples for class prediction (i.e. **A**, **B**, **C**, ...), I must preprocess the image and curate an appropriate set of grayscale, 28×28 pixel samples from the image.

Below, I develop an algorithm for processing the satellite imagery to create the set of samples to feed into the neural net. Here we will look at the steps of processing the input sample, including the curation of the sample, the transformations applied upon the sample, and the image pyramid structure used for evaluating the original image at different resolutions.

Sample Curation

Before I give the neural net the input image, I must preprocess the image and curate a grayscale, 28×28 pixel sample from the image.

In the script, I first convert the image to grayscale from RGB. This allows the input to the neural net to be in the same form as the EMNIST samples.

After preprocessing, I curate samples from the image by **stepping through** the image. Stepping through an image means something different than splitting an image into a grid and iterating through each element. Instead imagine having a 28×28 pixel window which we slide across the rows and columns of the image to take our sample. Let our reference location be the top left of the 28×28 window – we'll start the sample curation at the top left of the image, and thus our reference location starts at (0,0). With a step size of 4, we move our window across the image row skipping 4 pixel in between. Our reference point moves from $(0, 0) \rightarrow (4, 0) \rightarrow (8, 0) \rightarrow$ etc. Once our window reaches the end of the row, we restart at the beginning of the row and shift down by our step size of 4. Our reference point proceeds to move across the row from $(4, 4) \rightarrow (4, 4) \rightarrow (8, 4) \rightarrow$ etc. In this manner, we step through all of the pixels in the image. Traversing multiple images, I have found a step size of 4 to be most effective in terms of trading off between speed and content covered.

Sample Transformations

Having generated a 28×28 pixel sample from the image, I then transform the image in order to provide the neural net with as much fodder as possible. There are three methods for transforming the curated sample: **rotation**, **reflection**, and grayscale **inversion**. I perform the sixteen possible combinations of these transformations in order to create the best possible representation of the set. See *Figure 2* for a visual example of these sixteen combinations of transformations.

Grayscale inversion is important in the case of prediction against EMNIST. Because all of EMNIST's images present grayscale depictions of white characters on a black background, the neural net learns to predict a lighter form on a darker background. Therefore if the letterform happens to be of a darker form on a lighter background, we want to feed the neural net an image of the letterform as a lighter object on a darker background. Inverting the grayscale values of all the sample's pixels efficiently performed this desired transformation.

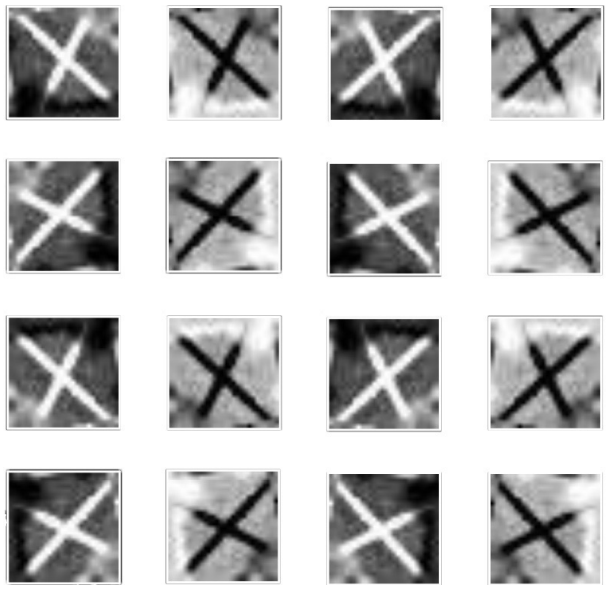


Figure 2: *Sixteen possible combinations of the three transformations – rotating, flipping, and inverting – performed on a grayscale,* 28×28 *pixel sample. This sample is an image of the sidewalk in front of Scully Hall in Princeton, New Jersey.*

Image Pyramids

In order to not only find letterforms in the 28×28 pixel form of the original image, I create an **image pyramid** of the 1200×700 pixel original image. This means *downsampling* the original image to multiple smaller pictures including images of the following dimensions:

$$\overset{(1)}{\downarrow}$$

$$\mathbf{1200 \times 700}$$

$$\overset{(2)}{\downarrow}$$

$$\mathbf{600 \times 350}$$

$$\overset{(3)}{\downarrow}$$

$$\mathbf{300 \times 175}$$

$$\overset{(4)}{\downarrow}$$

$$\mathbf{150 \times 87}$$

$$\overset{(5)}{\downarrow}$$

$$\mathbf{75 \times 43}$$

We see that for an original image of size 1200×700 pixels requires an image pyramid with five tiers of depth.

For this algorithm, I step through each of these differently-sized images, looking at 28×28 pixel samples in the method detailed before. I save the most confident detection per character class, so by the end of the script, one letter may be at a different resolution than another. For example in the computer-generated output, one letter may be 64×64 pixels in the original image while another letter could be 28×28. See an example of multi-scale letterform detection in *Figure A4*.

Applying a Neural Network

Now, I feed this set of curated 28×28 pixel samples harvested from the satellite imagery to the neural network. From the network, I receive a prediction of the label **A**, **B**, **C**, *etc.* as well as the network's percent confidence in the prediction.

One second – what exactly is a neural network, and how does it work? The computational power of modern computers is astounding. However, there is no program that can independently think and make associations without a human programmer initiating these decisions.

Let's look back at the idea of pareidolia and the way that the human brain can detect patterns in random data. Psychologists hypothesize that pareidolia stems from an infant's necessity to identify both its parents and potential predators. [Coolidge and Coolidge 2016] We compare this ability to that of the modern computer and see that unlike the human brain, the computer can only do what it is told. In an effort to create more computational complexity, we create artificial neural networks to simulate connected neurons in the human brain, attempting to mimic the brain's ability to learn and make decisions. [Woodford 2018]

In this project, I specifically use a recurrent convolutional neural network (**RCNN**). I decided to run an RCNN due its limited parameters compared to a basic convolutional neural network and the RCNN's recent success and speed in object detection. [Liang and Hu 2015; coopss 2016]

First, I **train** the RCNN on the thousands of character instances in an EMNIST dataset. To do this, we pass the pre-labeled EMNIST character images through the network one-by-one. As more and more images pass through the network, the quality of predictions of the subsequent characters improves. In order to test the efficiency of the trained RCNN, we take pre-labeled characters which we did not use to train the network and see how well they perform on the trained net. With EMNIST's Digits dataset, I train the network to a testing accuracy of over 99 percent; for EMNIST's Letters dataset, I achieve over 93 percent.

After training the network, I then **test** the network on the curated 28×28 pixel samples in an attempt to simulate typographic pareidolia. I run each 28×28 pixel sample through the trained artificial neural network in order to "predict" the letterform seen in that sample.

Regarding timing, training the RCNN with ten epochs on EMNIST's Letters dataset takes about fifty minutes, comparable to using six epochs on Digits. Testing a 1200×700 original image takes a shorter period of time, averaging at about twenty minutes to complete the letterform detection. In order to test my script on multiple locations, I would queue around ten locations and run my script overnight.

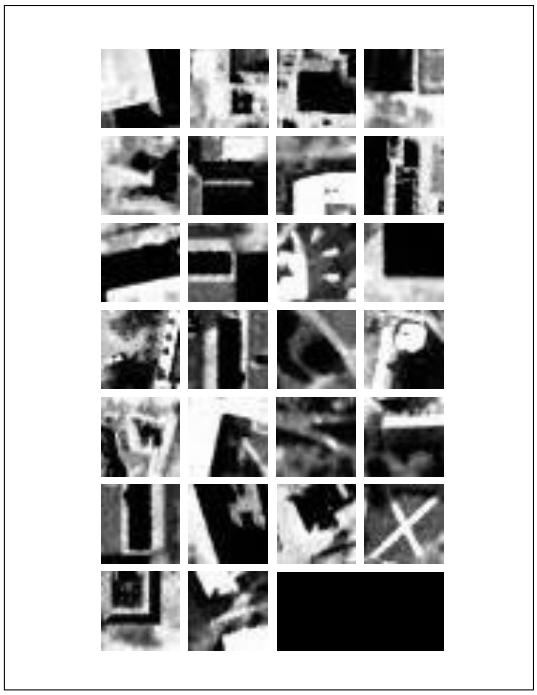


Figure 3: *Computer-detected letterforms for Princeton, New Jersey. The output grid should be read in alphabetic order from the top left, across and then down.*

Tracing and Vectorizing Letterforms

From the 28×28 output of the RCNN algorithm, I hand-trace the forms of the letters. I overlay a roll of tracing paper atop the grid of outputted images and with an ink pen, I trace the outline of the letterforms. See an output of Princeton, New Jersey in *Figure 4*.

Having finished the traced letterforms, I copy the sheet of tracing paper in a copy machine and fill in the forms with black Sharpie™. I then take a high contrast picture of this sheet with the filled-in letterforms and import the photo into Adobe Illustrator. There I make a Live Trace of the image (accessed under Object $\rightarrow$ Live Trace $\rightarrow$ Make). This function turns the black bitmap into vectorized outlines of the letterforms.

These outlines from Illustrator copy and paste directly into the **Glyphs App** (`glyphsapp.com`), where each of the forms are entered in as individual characters in the alphabet. The vectorized font is then exported directly from the app as a TrueType (*.ttf*) or OpenType (*.otf*) file. These files open up directly on a computer via an application like FontBook to be instantly downloaded and used on any computer.

Throughout the process, I attempted different ways of approaching the tracing and vectorizing procedure. I tried out tracing the letterforms digitally in Illustrator with Bèzier curves as well as exporting high-resolution computer-generated outputs as bitmaps for vectorization directly to to Illustrator. However, the hand-drawn outlines on physical tracing paper felt most right for this project; this technique gave me the best result in terms of smoothness of a character's form and control over the font's overall weight.

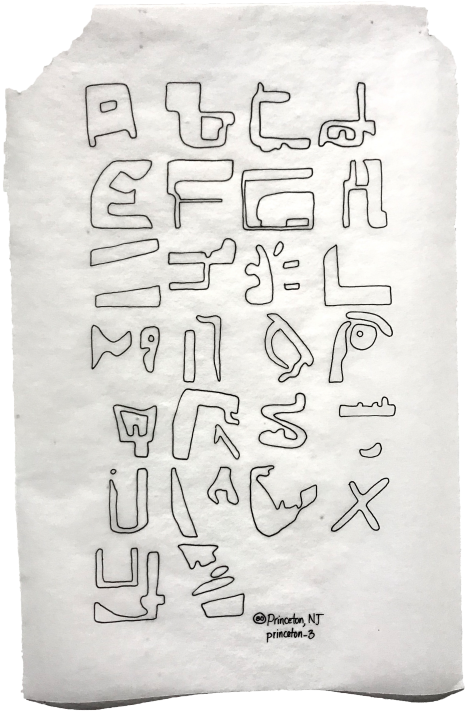


Figure 4: *Hand-traced letterforms for Princeton, New Jersey.*

Evaluation

In its conception, I meant this project to lie at the fringe of computer science research. Unlike Groß and Lee with their Aerial Bold Project, I was not searching for the most legible and clean letterforms. I instead wanted to keep some trace of the topography in the final vectorized version of the typefaces.

I fully understand that this measurement metric is subjective, but I feel that the evaluation techniques I detail below stay true to the intended outcome of this project, and I am delighted to present three months of work evaluated in the form below.

Initial Testing on Naples

I am often annoyed at computer graphics research today in seeing researchers using a test set they curated haphazardly from the web or did not curate themselves at all. Using a preexisting dataset of test images like ImageNet or COCO (Common Objects in Context) from past work is understandably helpful in the search for comparison and reproducibility. However, without constraints of a large curated dataset or randomized content, I feel novel research should be initially tested on images hand-selected by the researcher, not Internet stock images. [Deng et al. 2009; Lin et al. 2015]

For this project, I intentionally first tested my algorithm on two specific locations: Naples, Florida and Naples, Italy.

Naples, Florida represents my hometown, the town in which I lived for the first eighteen years of my life. In aerial imagery of Southwest Florida, there is stark contrast in the dark bodies of water weaving around the light-colored, human-inhabited structures like roads and neighborhoods.

Naples, Italy represents the city counterpart to Naples, Florida both in name and in aerial view. While both Naples, FL and Naples, Italy are on ocean-facing coastlines, aerial imagery of the Italian city has comparatively very little visual contrast separating its building structures. This serves as different fodder for the letterform detection, allowing for experimentation for contrast parameters in order to optimize the legibility of the given satellite footage.

Over the course of this project, I proceeded to create 103 fonts.

Las Piedras River and Time

During the Spring 2018 semester while creating over one hundred of these fonts, I took another Visual Arts class with David Reinfurt called *I-n-t-e-r-f-a-c-e*. The curriculum of this course broadly focused on the concept of time, especially non-conventional perceptions of the interface of time. Throughout the semester, I used the river in my work as metaphor for the passage of time; I created a couple fonts from sections of Peru's Las Piedras, a river I'd been focusing on throughout the semester for the handwritten quality of its curves. Iterating through multiple designs for the Apple Watch clock interface, I settled on a version using digits created by the letterform detection algorithm and connecting the characters from one to the next, ultimately shaping a river of time. See *Figure A5* and *Figure A6* for visual depictions.

The VIS Junior Show posters

While developing the algorithm for the letterform detection, I was in the process of creating the printed matter for the Princeton Visual Art Departments Junior Independent Work Show, known around campus as *The VIS Junior Show*.

I decided to incorporate the use of my font system in a poster campaign for the show, asking each of the twenty artists displaying work for the one physical place in the world which has most inspired his or her artistic practice. This question results in a wide array of answers, from Midtown Sacramento to I-95 to Southern Norway. See the final result of the posters series in *Figure A7* and *Figure 5*. My two favorite posters are Kyra's in the font *I-95* and mine (Kara's) in the font *NAPLES*, Florida.

Discussion

Public Opinion

Since publically publishing the *VIS Junior Show* posters, I have received both direct and indirect commentary on my fonts, all of which I find informative if not entertaining.

In advertising the opening reception for the *VIS Show* to members of Terrace, a student organization on campus, a group leader wrote an email referencing the posters:

> *Tonight @ 5:30 – 6:30pm: Junior Vis Show opening reception featuring some [Terrace members] whose names I can't read cause the font is too difficult* [Anonymous 2018]

As well, in April 2018 one of my fonts become a meme. A peer in the Princeton Computer Science Department posted one of my posters in the *graphic design is my passion* Facebook meme page hosting over fifty-six thousand followers. The post's caption read:

> *you'd think a visual arts department would understand how to design legible letters but I guess thats too much to ask*

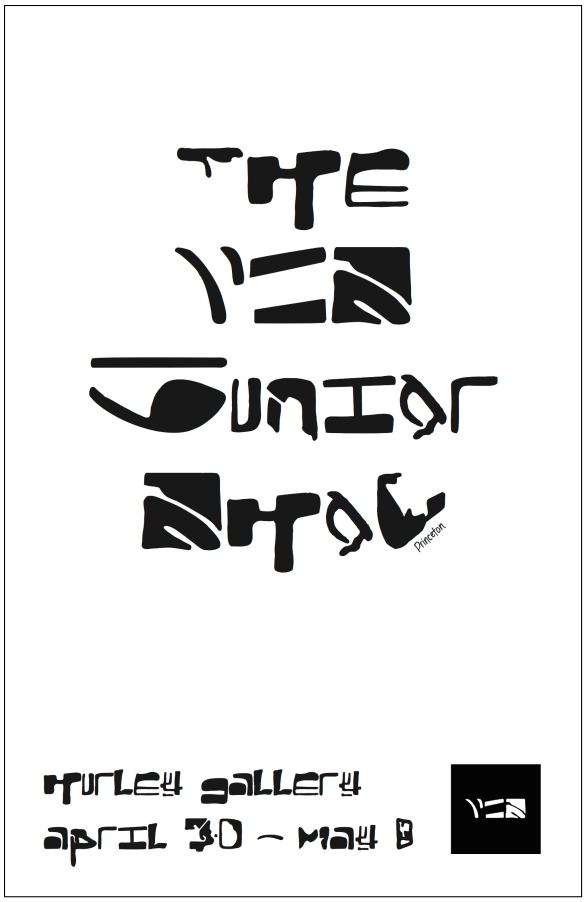


Figure 5: *Circulating poster for The VIS Junior Show, April 2018. The entire poster is in the font PRINCETON, New Jersey. This poster was posted on screens and signs at Princeton University. It also appeared on the graphic design is my passion Facebook meme page in April 2018.*

Within the course of the night, several people responded to the post. A couple Visual Arts peers spoke about the conception of the work and the poster campaign. Some members of the meme community proceeded to guess the words formed on the poster ("the viz junior shoc") while others noted the phallic qualities of the *S* in *VIS*. Generally, I received positive feedback, including:

> *Actually kinda interesting*
>
> *This is graphic design at its finest*

Less than twelve hours after the post was initially written, the meme was deleted from the group. [Rivitz 2018]

Legibility Constraints

Without context, people often comment on the "hieroglyphic" and "blocky" qualities of the letterforms. As we just read, many people also comment on the fact that the forms are difficult to read.

In the process of hand-tracing these letterforms, I understand the illegibility of these fonts to be primarily due to the absence of a stem in particular letterforms, particularly in the forms of *I*, *J*, *L*, and *T*. In typographic anatomy, the **stem** of the letterform acts as the main vertical stroke. As we see in *Figure A8* with the character "I," the algorithm detects different combinations of horizontal strokes for these four problematic letterforms. The *I* is often represented by two horizontal strokes of equal length on the top and bottom, the *J* by two horizontal strokes on the top and bottom with the top longer than the bottom, the *L* by one horizontal strokes on the bottom, and the *T* by one horizontal strokes on the top.

Conclusion

The process of creating over one hundred of these fonts has allowed me to spend time analyzing the forms of both topography and typography is ways I had never explored. By collaborating with the ideas of others – be that my Visual Arts peers or faceless developers on Github – I have been able to grow this project into a form which definitely satisfies and surpasses my initial goals. With results like I-95 stemming from my unorthodox "evaluation" using the *VIS Junior Show* posters, I have created a field for further investigation, perhaps outside the realm of pure satellite imagery. I have honestly had a blast working on this project, and I will keep this algorithm as a tool for my future artistic and academic work as I continue to enter this new territory.

Acknowledgements

Thank you to Professor David Dobkin, David Reinfurt, and the students in Spring 2018 *COS IW02: Using Publicly Available Data to Learn, Explain, Evaluate and Improve* for all your feedback, patience, and encouragement.

References

ANONYMOUS, 2018. Terrace email. Gmail, personal communication, 4.

ARAWAL, A. 2015. *Character Recognition in Satellite Images using Machine Learning*. Master's thesis, Ravensburg-Weingarten University of Applied Sciences.

COHEN, G., AFSHAR, S., TAPSON, J., AND VAN SCHAIK, A. 2017. EMNIST: an extension of MNIST to handwritten letters.

COOLIDGE, F. L., AND COOLIDGE, M. L. 2016. Why People See Faces When There Are None: Pareidolia. *Psychology Today*.

COOPSS, 2016. EMNIST. `https://github.com/Coopss/EMNIST`.

DENG, J., DONG, W., SOCHER, R., LI, L.-J., LI, K., AND FEI-FEI, L. 2009. ImageNet: A Large- Scale Hierarchical Image Database. In *CVPR09*.

GROß, B., AND LEE, J., 2016. The Aerial Bold Project. `http://type.aerial-bold.com/process/`.

GROTHER, P. J. 1995. NIST Special Database 19 Handprinted Forms and Characters Database. *National Institute of Standards and Technology*.

LECUN, Y., CORTES, C., AND BURGES, C. J., 2013. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

LIANG, M., AND HU, X. 2015. Recurrent Convolutional Neural Network for Object Recognition.

LIN, T.-Y., MAIRE, M., BELONGIE, S., BOURDEV, L., GIRSHICK, R., HAYS, J., PERONA, P., RAMANAN, D., ZITNICK, C. L., AND DOLLR, P. 2015. Microsoft COCO: Common Objects in Context.

MORDVINTSEV, A., OLAH, C., AND TYKA, M. 2015. Inceptionism: Going Deeper into Neural Networks. *Google Research Blog*.

NOLIN, R. 2015. Woman on a roll, gets big bread for cheesy sandwich. *Sun Sentinel*.

PHILLIPS, T., 2001. Unmasking the Face on Mars. https://science.nasa.gov/science-news/science-at-nasa/2001/ast24may_1.

REINFURT, D., BAILEY, S., AND KEEFER, A. 2012. *Ecstatic Alphabets/Heaps of Language*. The Serving Library.

RIVITZ, W., 2018. graphic design is my passion. Facebook, personal communication, 4.

RUSSON, M. A. 2015. Google DeepDream robot: 10 weirdest images produced by AI 'inceptionism' and users online. *International Business Times*.

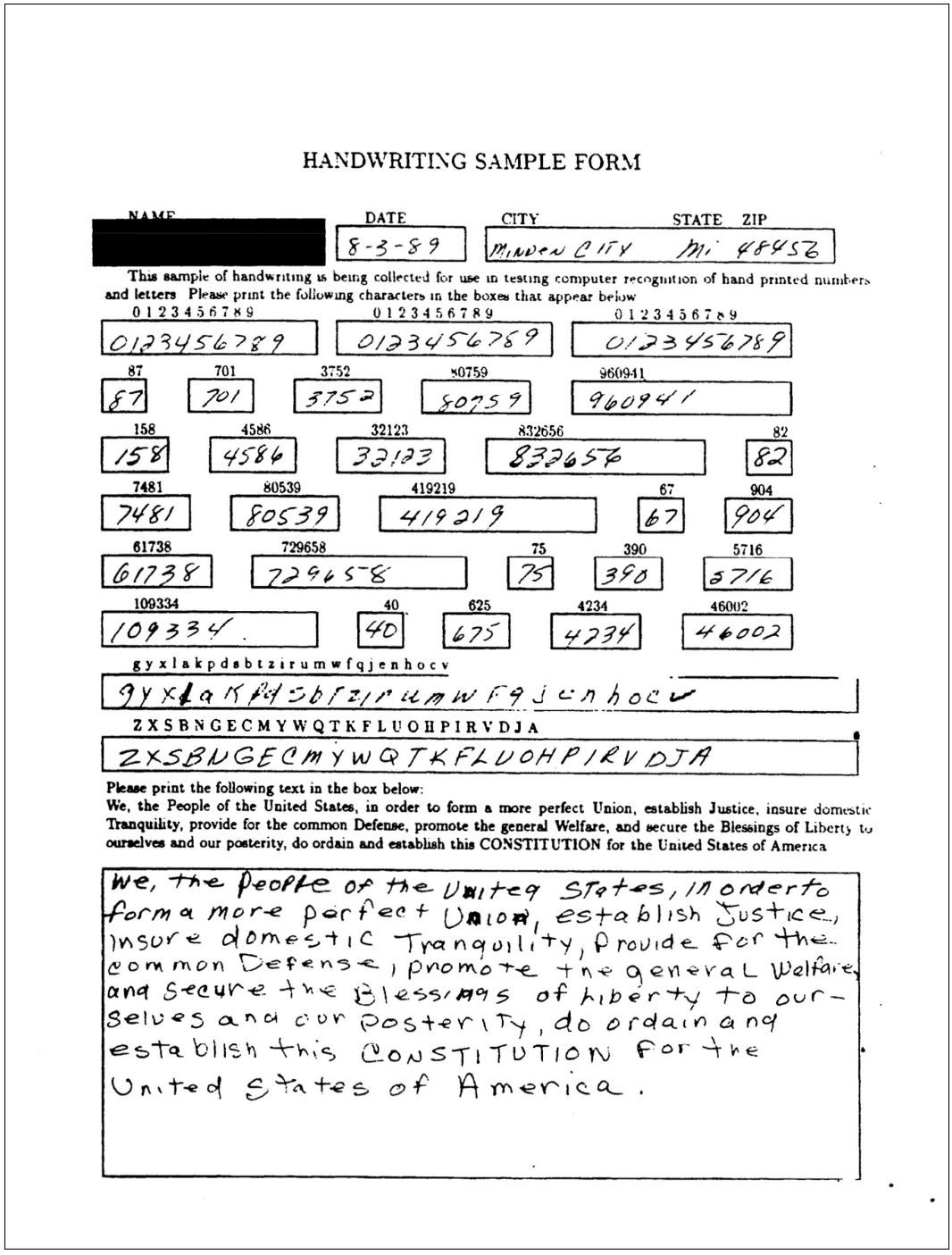
WOODFORD, C., 2018. Neural networks. *Explain that Stuff*.

WOOLDRAGE, C. 2017. Found Font (1995 – present). *Medium*.

Honor Code

I pledge my honor that this report represents my work in accordance with University regulations.

– Kara Lynn Bressler



HANDWRITING SAMPLE FORM

NAME [redacted] DATE 8-3-89 CITY Minden City STATE MI ZIP 48456

This sample of handwriting is being collected for use in testing computer recognition of hand printed numbers and letters Please print the following characters in the boxes that appear below

0 1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9

0123456789 | 0123456789 | 0123456789

87 | 701 | 3752 | 80759 | 960941

87 | 701 | 3752 | 80759 | 960941

158 | 4586 | 32123 | 832656 | 82

158 | 4586 | 32123 | 832656 | 82

7481 | 80539 | 419219 | 67 | 904

7481 | 80539 | 419219 | 67 | 904

61738 | 729658 | 75 | 390 | 5716

61738 | 729658 | 75 | 390 | 5716

109334 | 40 | 625 | 4234 | 46002

109334 | 40 | 675 | 4234 | 46002

g y x l a k p d s b t z i r u m w f q j e n h o c v

9 y x l a k p d s b t z i r u m w f q j e n h o c v

Z X S B N G E C M Y W Q T K F L U O H P I R V D J A

Z X S B N G E C M Y W Q T K F L U O H P I R V D J A

Please print the following text in the box below:
We, the People of the United States, in order to form a more perfect Union, establish Justice, insure domestic Tranquility, provide for the common Defense, promote the general Welfare, and secure the Blessings of Liberty to ourselves and our posterity, do ordain and establish this CONSTITUTION for the United States of America

We, the People of the United States, in order to form a more perfect Union, establish Justice, insure domestic Tranquility, provide for the common Defense, promote the general Welfare, and secure the Blessings of liberty to ourselves and our Posterity, do ordain and establish this CONSTITUTION for the United States of America.

Figure A1: *NIST Handwritten Sample Form, August 1989. Image courtesy of Patrick Grother. [Grother 1995]*

Figure A2: *Raw satellite imagery of Princeton, New Jersey from Google Earth Pro, February 2018.*

Figure A3: *Cropped satellite image of Princeton, New Jersey adjusted for contrast, highlights, and shadows.*

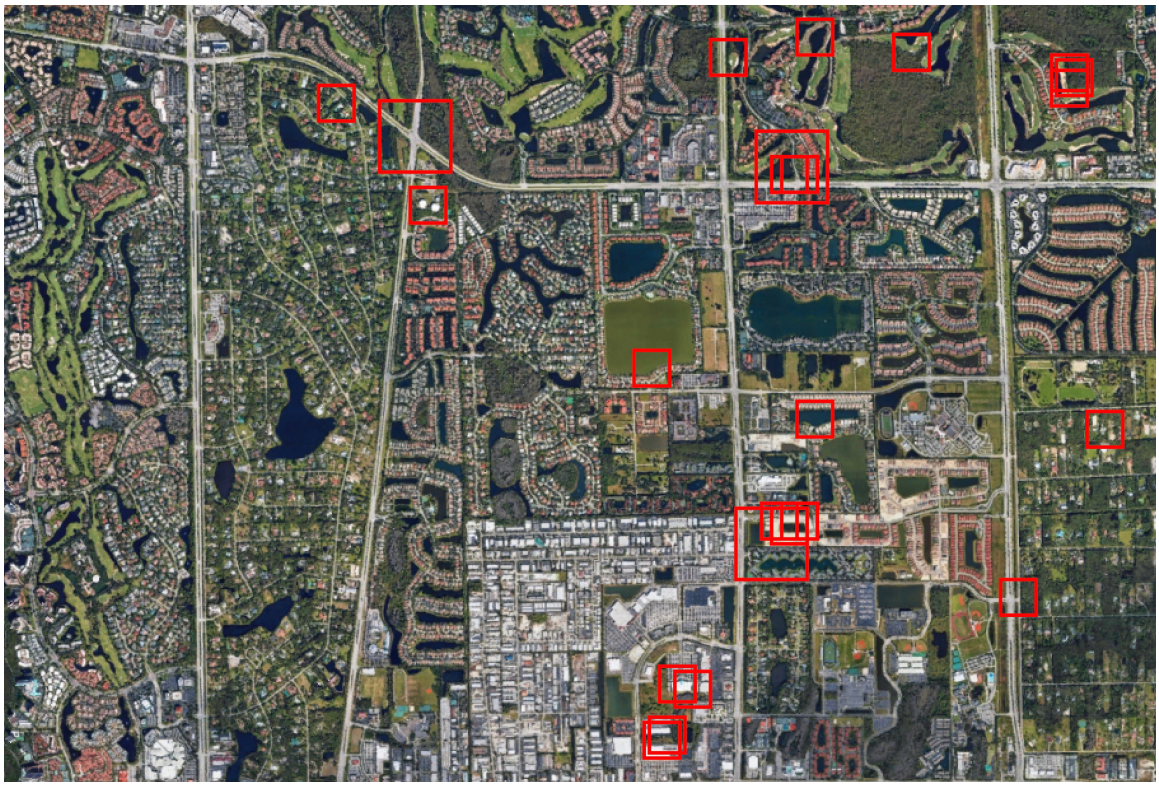


Figure A4: *Letterform detection algorithm applied to satellite image of Naples, Florida, March 2018. This output shows character detection of both* 28×28 *and* 64×64 *pixel samples. For this image, the algorithm produced a five-tier image pyramid.*

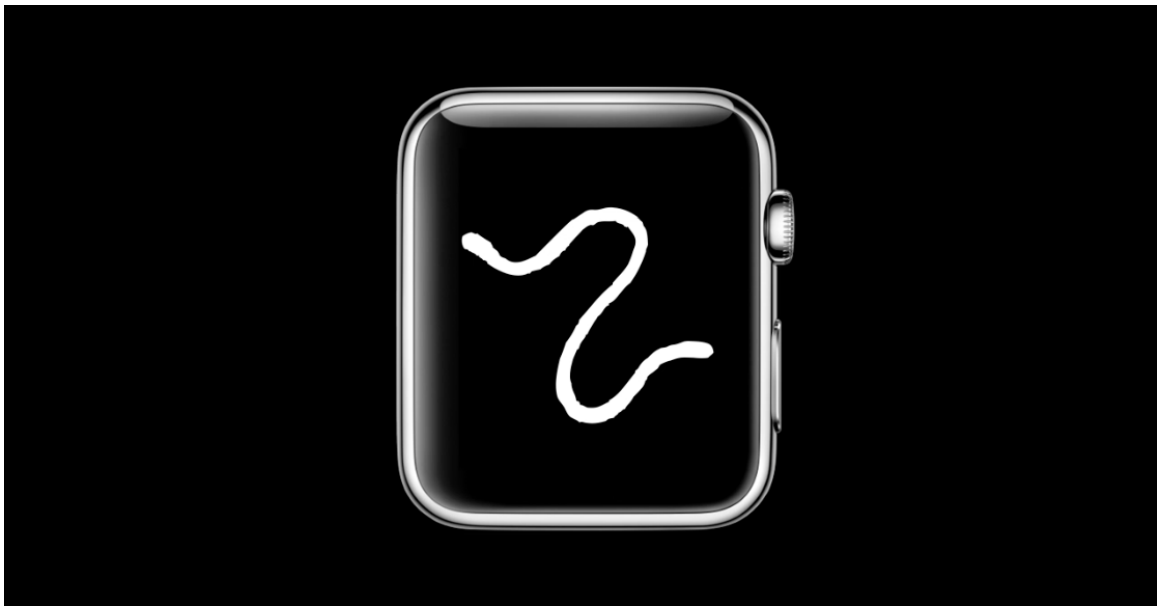


Figure A5: *Design for an Apple Watch clock interface using the font LAS PIEDRAS, Peru. The watch currently presents the number "2."*

Figure A6: *Continuous river of numbers using the font LAS PIEDRAS, Peru.*

Figure A7: *Poster series with individual names for The VIS Junior Show, April 2018.*

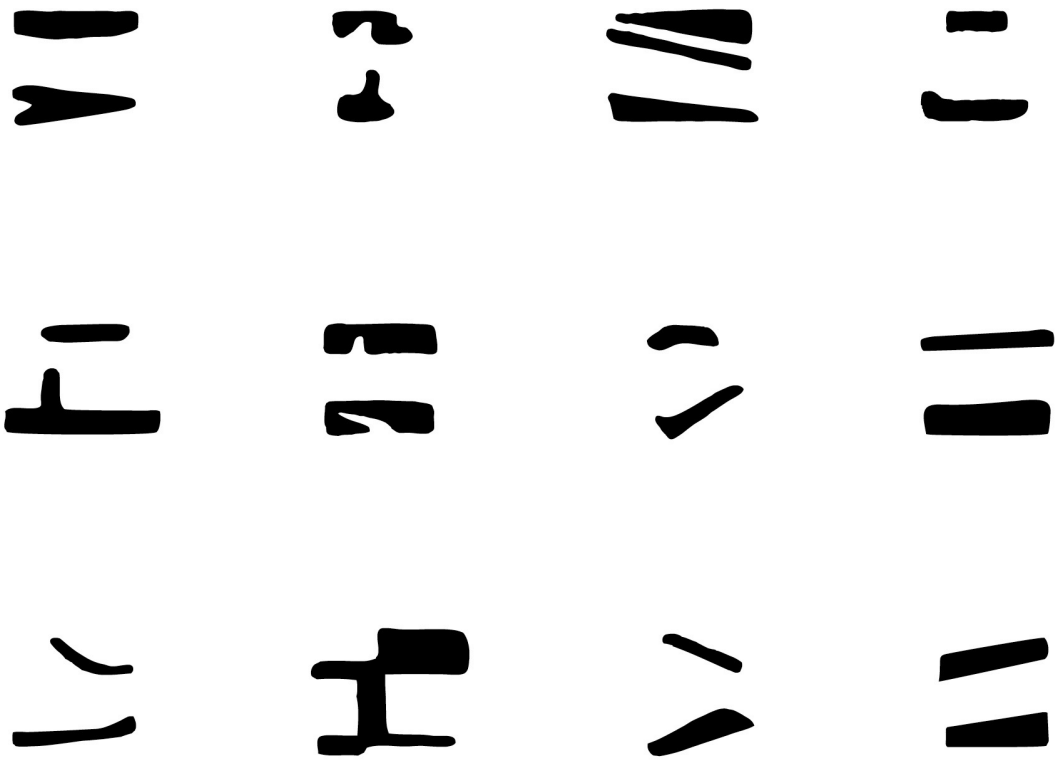


Figure A8: *Collection of vectorized "I" characters, many of which lack stems.*