OmniPrint: A Configurable Generator for Printed Characters

Abstract

We introduce OmniPrint, a synthetic data generator for isolated printed characters designed to support machine learning research. While being inspired by popular datasets, such as MNIST, SVHN, and Omniglot, OmniPrint provides the unique ability to produce a wide range of printed characters from various languages, fonts, and styles, with custom distortions. OmniPrint includes 935 fonts from 27 scripts, and supports many types of distortions. As a demonstration of its functionality, we present several use cases, including an example of a meta-learning dataset designed for a machine learning competition. OmniPrint is publicly available at a specified github link.

1 Introduction and Motivation

Benchmarks and shared datasets have helped propel progress in machine learning. One popular benchmark is MNIST, used worldwide in tutorials, textbooks, and classes. Many variants of MNIST exist, including Omniglot, which includes characters from several different scripts. Since Deep Learning techniques rely heavily on data, as there is an increasing number of datasets, more, larger datasets are required. Since collecting and labeling data can be time-consuming and expensive, artificial data generation can be used to drive ML research. This motivates the creation of OmniPrint, an extension of Omniglot, specifically designed for the generation of printed characters.

Our focus is on classification and regression problems, where a vector y, which is composed of either discrete or continuous labels, is to be predicted using an input vector x of observations, which in the case of OmniPrint, is an image of a printed character. Additionally, data are often affected by nuisance variables z, which are discrete or continuous labels that represent metadata or covariates. For our work, z may include character distortions such as shear, rotation, line width variations, or background changes. Thus, a data generation process with OmniPrint contains the following steps:

$$Z \sim P(Z), \quad Y \sim P(Y|Z), \quad X \sim P(X|Z,Y).$$

In many domains such as image, video, sound, and text applications, where objects or concepts are target values to be predicted from percepts, Z and Y are independent and hence $P(Y|Z) = P(Y)$. This type of data generation is also encountered in medical diagnoses of genetic disease, for which x would be a phenotype and y a genotype, and also analytical chemistry where x might be chromatograms and y would be compounds to be identified. We expect that progress made using OmniPrint to benchmark machine learning systems should foster progress in these domains.

Character images represent excellent benchmarks for machine learning, given their simplicity, and visual nature, and for enabling the development of real-world applications. However, our exploration of available resources revealed that there is no synthesizer that fulfills all of our needs. No available synthesizer allows for the generation of realistic small-sized images, supports a wide variety of character sets, and offers control over the variation of realistic conditions through parameters. The synthesizer must support pre-rasterization manipulation of anchor points, post-rasterization distortions, seamless background blending, foreground filling, anti-aliasing rendering, and be easily extensible with new fonts and styles.

.

2 The OmniPrint Data Synthesizer

2.1 Overview

OmniPrint builds on the open-source software TextRecognitionDataGenerator, adapting it to our specifications. The software is designed to allow researchers to generate data in a form that makes it easier to train machine learning models. To obtain a large number of classes (Y labels), we manually selected and filtered characters from the Unicode standard, forming alphabets for over 20 languages. These alphabets are divided into partitions (e.g., Oriya consonants). Nuisance parameters (Z) are divided into Font, Style, Background, and Noise. The fonts are selected by an automatic font collection module. We added a feature using the FreeType rasterization engine which enables vector-based pre-rasterization transformations. Additionally, we enriched background generation with seamless blending, and enabled custom post-rasterization transformations. We also implemented utility code including dataset formatters, and a data loader which generates episodes for meta-learning applications. To our knowledge, OmniPrint is the first text image synthesizer geared toward ML research to support pre-rasterization transforms.

2.2 Technical Aspects of the Design

The OmniPrint's design has extensibility as a key feature. Users can add new alphabets, fonts, and transformations to the generation pipeline.

The design can be summarized as follows:

- **Parameter configuration file:** Support for both TrueType and OpenType font files is included. Style parameters include rotation angle, shear, stroke width, foreground, text outline, and other transformations.
- **FreeType vector representation:** Text, font, and style parameters are used by the FreeType rasterization engine.
- **Pre-rasterization transformed character:** FreeType performs all the pre-rasterization (vector-based) transformations. Pre-rasterization manipulations include linear transforms, stroke width variation, random elastic transformation, and variation of character proportion. The RGB bitmaps output by FreeType are called the foreground layer.
- **Pixel character on white background:** Post-rasterization transformations are applied to the foreground layer. The layer is kept at a high resolution, using ReLU activations, to avoid artifacts. The RGB image is then resized using a three step process; applying a Gaussian filter to smooth the image, reducing the image by an integer factor, and resizing using Lanczos resampling.
- **Pixel character on textured background:** The foreground is then pasted onto the background.
- **Logging and Visualization:** The library utilizes a Weights Biases tool to log the training process and the visualizations. It visualizes the condition's traversals, latent factor traversals, and output reconstructions as static images and animated GIFs.