Learning a Legibility Classifier for Deformed Letters

Warunika Ranaweera
wranawee@sfu.ca

Zeinab Sadeghipour
zsadeghi@sfu.ca

Jaime Vargas-Trujillo
jvargast@sfu.ca

Abstract

We introduce an approach for classifying a deformed letter from the English alphabet based on its *legibility*. Given an image of a deformed letter d, our method proposes the use of the filtered medial axis (i.e. strokes) as its simplified representation, from which we can compute a feature vector. Based on the anatomy of the letter d we identify the strokes as: ascender, descender and counter. Feature correspondence across deformations is ambiguous; thus we propose using a kernel-based multi-instance classifier model. Our classifier is trained using legibility score labels collected through a user-study. Our approach obtains an accuracy of 71.1% and outperforms a classic boundary-based method. Our findings suggest that, although our features are meaningful descriptors of the letter forms, a robust methodology for acquisition of legibility scores is required to ensure accuracy.

1 Introduction

A calligram is a poem, phrase, or word in which the typeface, calligraphy, or handwriting is arranged in a way that creates a visual image (figure 1). If we want to generate calligrams automatically, we need to embed an input word into a visual image which inevitably involves deforming the letters of the word. Crucial to this problem is the preservation of the *legibility* of the letter shape even after drastic deformations. In order to achieve this goal, we need to develop a legibility metric that will drive our deformation process. Legibility is defined as the degree to which individual characters in text are understandable or recognizable based on appearance. It is based on human perception, and thus, developing a metric requires input from human participants. Recognizing the subjective nature of legibility, we rely on a user study to learn separate scoring functions for the legibility of letter d. Users are provided with pairwise comparison queries on legibility, i.e."Is this letter d more legible than the other?", and the collected results are used to train a *Multiple Instance Classification* model for identifying the legibility of new letter shapes.

Figure 1: A few calligrams in different languages.

2 Related Work

The existing body of work on legibility can be broadly classified into two categories. The first category focuses on examining the factors that contribute to legibility under specific cases and circumstances, such as the effect of different font types of online text on elderly users [1] and improving the legibility of highway guide signs using novel fonts [2]. However, this category does not address *deformation* as a factor in legibility.

The second category encompasses work related to text-based human/computer distinguisher tests (i.e. CAPTCHAS), where text undergoes distortion and obfuscation techniques, while maintaining a balance between a positive user experience [3] and robustness against attacks [4]. This category encompasses work that has a closer resemblance to our problem. However, insofar as we were able to determine, the particular geometric deformations used for the generation of CAPTCHAs differ from our own approach in a manner which makes their findings incompatible with our goal.

3 Methodology

To accomplish the goal of classifying different forms of the letter *d* regarding their legibility, some training data with features and labels was required. Since there was no other similar study, we were obliged to collect some data, extract their features and assign a label to each letter. To obtain such data set, first we generated various forms of the letter *d* using a *deformation model* explained in section 3.1. Then we conducted a user study to acquire labels for these forms. To extract features, we examined the *medial axis* of the standard form of the letter *d* and attempted to define some properties that appear to make a letter shape a *legibile d*. At last, we trained a model using *Multiple-instance Learning* to recognize legibile forms than illegibile ones.

3.1 Deformations

The particular setting of our problem required that we obtain or produce a set of deformations. This set should be representative of the types of deformations which we anticipate our algorithm will produce. To the best of our knowledge, no such set exists. We initially considered utilizing the UJI Pen Characters Data Set, with some modifications, but we abandoned the idea because it wasn't clear how we could relate the data contained in this repository to our own setting.

We decided on implementing an interactive deformation sandbox on MATLAB, which we accomplished by making some adjustments to the demo code provided in Bounded Biharmonic Weights for Real-Time Deformation [5]. We utilized this deformation tool to manipulate letters and create a variety of deformations. Control points were defined along the boundary as well as the skeleton of each letter, in order to produce interesting variations in width and shape.

3.2 Labeling the Data

We conducted an online user study to approximate a legibility score for each deformed letter d_i to be considered as ground truth. However, given a single image, it is a difficult task for a human to decide an absolute scoring for the legibility. A frequently used approach that addresses this difficulty is presenting the user with pair-wise comparisons to find a subjective scoring instead of an absolute score [6]. Following a similar approach, we presented two forms d_i and d_j of the same alphabetical letter "d" to the users, asking them to select the most legible form out of the two.

Users underwent the following steps during the study.

1. Control question: identify the given letter
2. Pair-wise comparisons: select the most legible form out of the two forms d_i and d_j

A total of 88 users participated in the study, and each participant was given 20 pairs for comparison. We collected 1780 ($> \binom{60}{2}$) judgements in total; hence at least 55 votes per each d_i (i=1..60).

Subsequent to the collection of votes from users, we calculated a legibility score (s_i) for each d_i by taking the ratio of the number of votes per each d_i (i.e. $votes_i$) and the number of users who were

presented with d_i (i.e. $voters_i$). In the legibility scores collected, $|d_i; s_i < 0.5|$ = $|d_i; s_i \geq 0.5|$. Hence, to get a better coverage of legible and illegible letters, we selected a threshold of 0.5 and labeled each d_i as illegible (0) if $s_i < 0.5$ and 1 otherwise.

3.3 Feature Extraction

As was previously stated, our motivation is to incorporate legibility into a deformation model for letters in the English alphabet. Deformation models are generally driven by paradigms such as deformation cages, skeletons, or independent control points. Therefore, we require a set of simple, geometric features that could provide us with the potential to understand how the manipulation of said paradigms correlates to the legibility of the letter.

For this purpose, we settled on using the *medial axis* as a simplified representation for each of the instances we will be examining in this project. We believe this representation makes sense, because the *skeleton* obtained via the *medial axis* can be thought of as *strokes*, i.e. the path taken by an imaginary marker with a nib that can be continuously adjusted for size. In the following sections, we explain in detail how we used these *strokes* to extract feature vectors.

3.3.1 Idealized Representation of a Letter

To begin our study, we constructed an idealized representation of the letter *d* in a vector graphics editor. This representation removes all ornaments and provides us with what can be considered the *essence* of the letter (which was confirmed in our user study, this idealized letter received a perfect score). Analysis of the strokes of this idealized representation is straightforward, provide us with an accurate segmentation of the letter into the parts of its anatomy (as defined in typography) and illustrated in Figure 2

- The stroke that describes the *counter*, i.e. the *hole* of the letter *d*
- The long stroke connected to the counter, named the *ascender*
- The short stroke connected to the counter, called the *descender*

3.3.2 Vector representation of features

We require a feature representation that captures the spatial and geometric characteristics of the letter's *ascender,descender and counter*. Samples were not *aligned* in our experiments in order to preserve information about orientation, but they were *centered* with respect to the counter's centroid, and *scaled* to uniform height while maintaining aspect ratio.

The stroke that describes the *counter* is a closed curve which we chose to represent with *unnormalized* Elliptic Fourier Shape Descriptors. We use the first 15 harmonic values, which seems to be best practice for object recognition. We opted not to normalize these values, because normalization discards scaling and orientation which we believe are significant contributors to legibility. However, we take steps to ensure the starting point remains consistent.

The strokes that describe the *ascender* and *descender* are represented explicitly by 20 points uniformly sampled along these strokes, with coordinates *relative* to the *starting* point of the stroke (i.e. the point at which the stroke connects to the counter). We also capture the relative position of these strokes by including the coordinates of their starting points with respect to the *counter's centroid*.

To summarize, a letter's representation would consist of:

- The *Counter,* represented by 15 harmonics from its Elliptic Fourier Shape Descriptor, Each harmonic is encoded by 4 values, which gives us a representation containing 60 dimensions.
- The *ascender and descender,* each represented by 21 sample points, such that each sample is encoded by two dimensions which gives us a representation containing 42 dimensions for each, 84 dimensions in total.

Therefore our total representation is described in a feature space in $\mathbb{R}^{144}$.

3.4 Learning Model

After obtaining features and labels for a variety of forms of the letter *d*, we employed a machine learning algorithm to classify data as legibile or illegibile.The model that seemed to conform to our problem the most is *Multiple-instance learning (MIL)*. MIL is a generalization of supervised classification in which we can not describe each data sample, by a single characteristic feature vector. Instead each data point is represented as a *bag* (or set) of *instances* (or patterns) [7]. MIL aims to predict the bag labels by integrating the information in the instances.

Multi-instance learning can be performed based on different assumptions. The standard MIL assumption for classification states that a bag is labeled as a particular class if at least one instance in the bag belongs to that class. For example in a binary classification task, the algorithm assigns the *positive* label to a bag if at least one of the patterns related to that bag is labeled as *positive*; otherwise, the whole bag is a *negative* example [7].

In formal words, the input to MIL algorithms is a set of bags $X_1, X_2, ..., X_m$, where each bag is a set of instances $X_I = \{x_i : i \in I\}$. The classifier assigns each bag X_l a label Y_I based on the instances in the bag. Hence, the goal of a MIL algorithm is to learn a discriminant function $f : X \to -1, 1$, which maps each bag to the label.
Multi-Instance Kernel (MI-kernel) [8] is a kernel-based algorithm for multi-instance classification. In this method, a kernel is defined directly on the bags by aggregation of the base kernels over all the instance pairs. For example, given two bags X and X', and a base kernel k_x, the bag kernel is defined as:

$$K(X, X') := \sum_{x \in X, x' \in X'} k_x(x, x')$$

Finally, this kernel is normalized and plugged into an SVM classifier.

Our Problem: The principal struggle in our problem is that when we want to verify the legibility of a new form of a letter, we can not assume that we know the deformation model. In other words, we can not be sure about a correspondence between strokes in the original letter and the deformed one. As a result, it may not be possible to detect the legibility of the letter shape by describing a *single* feature vector. However, we know the *set* of feature vectors than can describe the letter.

This is how we formulated our problem: as you can see in the Figure 3, we considered each form of the letter *d* as a bag and each combination of isolated segments of medial axis as an instance of that bag. Then we used a kernel-based MIL [8] on our data to train the model for legibility classification.

4 Experiments and Results

We performed our experiments were based on MI-Kernel algorithm described in Section 3.4. Our choice of kernel was RBF, and thus we required to find optimal values for both the *C* and γ pa-

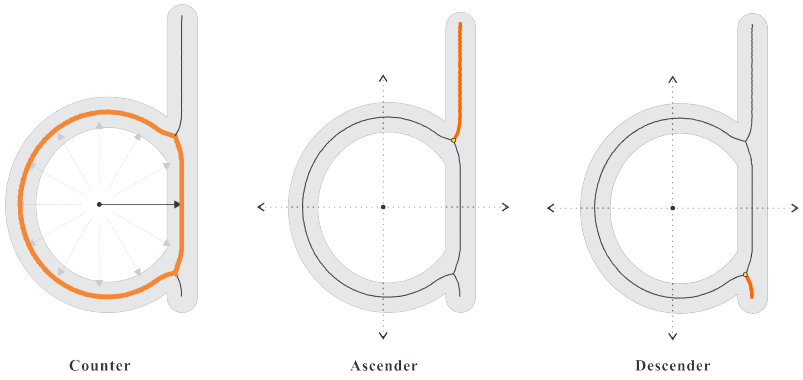

Figure 2: Components of an idealized representation of the letter *d*

rameters. Thus, we built an implementation of the grid method for parameter search on top of the MI-Kernel implementation.

4.1 Experiment 1: Parameter tuning, 5-fold cross validation

In this experiment, we conducted 5-fold cross validation using grid parameter search with the following parameters:

- $Log_2\gamma$ range: -5 to 5, step size = 0.25
- $Log_{10}C$ range: -10 to 10, step size = 1

The best accuracy yielded by this method was 63.3%, for $C = 10^2$ and $\gamma = 2^2$.

This accuracy is low, so two questions are raised:

- Is there a *relationship* between features and class?
- Can we *trust* our labels?

We then proceed to conduct a series of experiments to attempt to discover the root of our low accuracy results.

4.2 Experiment 2: Exclusion of ambiguous data

For the purpose of this experiment, we *excluded* data samples from our data set according to the criteria $0.4 < s_i < 0.6$ in an attempt to remove ambiguity in our labels. We hypothesize that, if there is a relationship between features and class, exclusion of ambiguous labels will yield a higher accuracy.

The method of our experiment is, as before, 5-fold cross-validation using grid parameter search. The best accuracy yielded by this method was 71.1%, for $C = 10^2$ and $\gamma = 2^{-1.75}$.

This is a noticeable increase in accuracy, which supports our notion that our labels are noisy. However, we still need to compare our results to a baseline method, which we will describe in the following sections.

4.3 Baseline Experiments

For these experiments, we replicate the conditions of experiments 1 and 2, but we utilize classic C-SVM with a feature set consisting of the *normalized fourier descriptors* of the shapes' boundaries. We chose this method as a baseline due to the fact that it is well studied in the field of shape recognition.

Baseline experiment 1 conducts 5-fold cross validation across the *entire* data set, and with parameter tuning the best result it yields is 51.6%, for $C = 10^{-7.4}$ and $\gamma = 2^{7.4}$. Baseline experiment 2

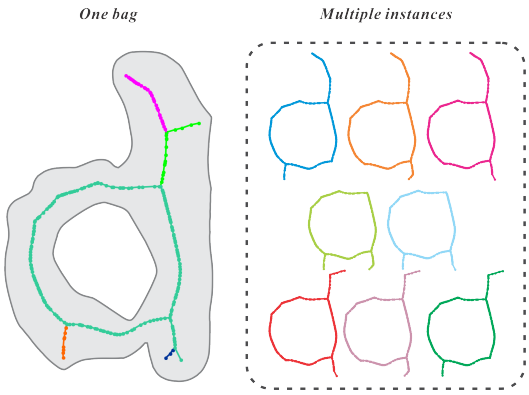

Figure 3: A bag and its multiple instances

conducts 5-fold cross validation across the *unambiguous* data set, and with parameter tuning the best result we were able to obtain is 66.67% for $C = 10^{9.3}$ and $\gamma = 2^{3.9}$.

This suggests that our features may contain more meaningful information about legibility than the standard, boundary based approach. A summary of our results is shown in Figure 4.3

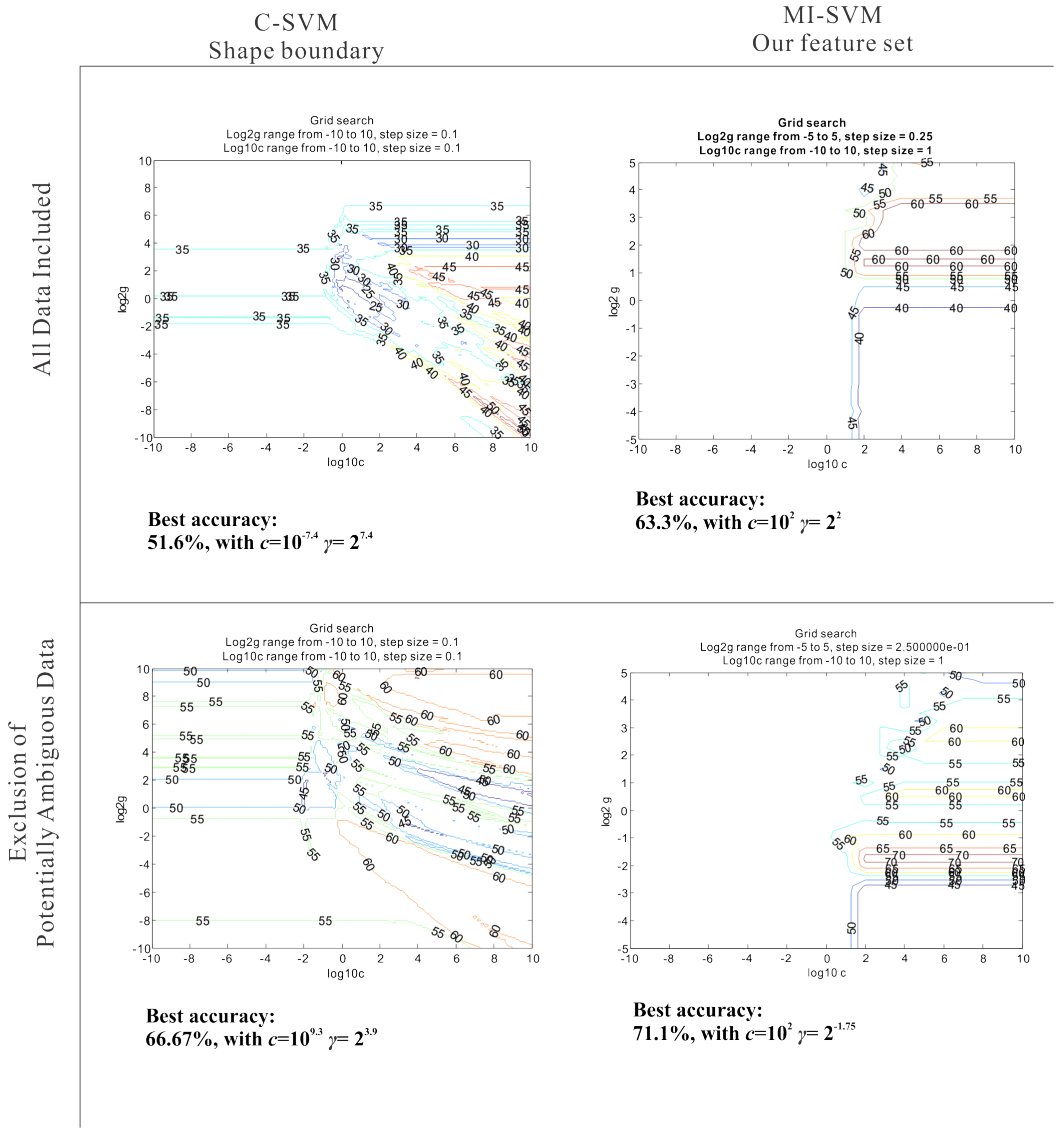

Figure 4: Comparison of experimental results using our approach and a classic, boundary based C-SVM approach.

5 Conclusion and Future Work

We present a learning model to classify a deformed letter as legible or illegible. Given the features of a deformed letter (d_i), our final classifier provides an accuracy of 71.1% using an MI SVM classification model with an RBF kernel. Our experiments suggest that this method yields a better performance than the boundary-based baseline method. Hence, our features can be considered as meaningful descriptors of the letter forms. Our experiments further suggest that this classification problem is hard, and further research is required to ensure a robust labeling method. Some areas to explore include:

- Active learning methods for improved selection of samples.
- Global pairwise ranking aggregation methods, such as the Bradley-Terry model.
- Using crowdsourcing platforms for a large scale labeling effort, since our final solution will require labeling for uppercase and lowercase forms of all 26 letters of the alphabet.

Results also suggest that the formulation of our problem may be more suited to other machine learning methods, such as Rank SVM. Further experiments will also be conducted in feature selection to reduce dimensionality. Letters of the alphabet possess differing topologies, therefore a different formulation for distinct topologies will be required. Once a proper legibility classifier is defined for a single letter, a generalized model can be built to measure the legibility of all the 26 alphabetical letters (both upper case and lower case).

Acknowledgments

We thank Richard Zhang, Ping Tan, Ibraheem Alhaseem and Ruizen Hu for all the ideas and insightful discussions. We also thank Arash Vahdat for the helpful suggestions; Hossein Hajimirsadeghi for providing the code for the novel MI-Kernel model; Alec Jacobson for providing the code on the deformation model; and to all the participants of the user study.

Contributions

JVT, ZS, WR developed the methodology, constructed the deformed letters, and conducted the experiments. JVT and ZS created the feature vectors. WR collected the ground truth labels.

References

[1] Philip Garvey, Martin Pietrucha, and Donald Meeker. Effects of font and capitalization on legibility of guide signs. *Transportation Research Record: Journal of the Transportation Research Board*, 1605(1):73–79, 1997.

[2] Ashley Colley, Piiastiina Tikka, Jussi Huhtala, and Jonna Häkkilä. Investigating text legibility in mobile ui: A case study comparing automated vs. user study based evaluation. In *Proceedings of International Conference on Making Sense of Converging Media*, AcademicMindTrek '13, pages 304:304–304:306, New York, NY, USA, 2013. ACM.

[3] Christos A. Fidas, Artemios G. Voyiatzis, and Nikolaos M. Avouris. On the necessity of user-friendly captcha. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2623–2626, New York, NY, USA, 2011. ACM.

[4] Elie Bursztein, Matthieu Martin, and John Mitchell. Text-based captcha strengths and weaknesses. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 125–138, New York, NY, USA, 2011. ACM.

[5] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH)*, 30(4):78:1–78:8, 2011.

[6] Jun-Yan Zhu, Aseem Agarwala, Alexei A Efros, Eli Shechtman, and Jue Wang. Mirror mirror: Crowdsourcing better portraits. *ACM Transactions on Graphics (SIGGRAPH Asia 2014)*, 33(6), 2014.

[7] Stuart Andrews, Ioannis Tsochantaridis, and Thomas Hofmann. Support vector machines for multiple-instance learning. In *Advances in neural information processing systems*, pages 561–568, 2002.

[8] Thomas Gärtner, Peter A Flach, Adam Kowalczyk, and Alex J Smola. Multi-instance kernels. In *ICML*, volume 2, pages 179–186, 2002.