StrokeBank: Automating Personalized Chinese Handwriting Generation

Alfred Zong and **Yuke Zhu**
Department of Computer Science, Stanford University
353 Serra Mall, Stanford, California 94305
{alfredz,yukez}@cs.stanford.edu

Abstract

Machine learning techniques have been successfully applied to Chinese character recognition; nonetheless, automatic generation of stylized Chinese handwriting remains a challenge. In this paper, we propose StrokeBank, a novel approach to automating personalized Chinese handwriting generation. We use a semi-supervised algorithm to construct a dictionary of component mappings from a small seeding set. Unlike previous work, our approach does not require human supervision in stroke extraction or knowledge of the structure of Chinese characters. This dictionary is used to generate handwriting that preserves stylistic variations, including cursiveness and spatial layout of strokes. We demonstrate the effectiveness of our model by a survey-based evaluation. The results show that our generated characters are nearly indistinguishable from ground truth handwritings.

Introduction

In today's digital age, virtually all text-based information is transmitted electronically. As a consequence, the uniform-looking typeset characters have lost a personal touch as compared to their handwritten counterparts. Hence, personalized handwriting generation has an untapped potential in many applications as it incorporates one's unique style into the digital information. In this paper, we propose a novel method to learn personalized Chinese handwritings from a few examples and to automatically generate personalized handwritings.

We focus on Chinese characters due to their rich structure and atomicity. The complex structure of characters yields great variations. In Chinese culture, stylized variation is deemed to reflect a writer's emotion and personality. Besides, in contrast to phonological languages (English, French, etc.), each Chinese character is a visual symbol (logograph) representing a word. This allows us to process each character independently.

Machine learning techniques have been widely applied to recognizing Chinese handwritings over the past few decades (Stallings 1976; Amin, Kim, and Sammut 1997; Ma and Liu 2008). Nonetheless, automatic generation of Chinese characters remains a challenge. Some attempts have been made to generate Chinese handwritings (Xu et al. 2012; 2009). Most works rely on the hierarchical structure of Chinese characters to decompose each character into a set of simple strokes. Though the results are promising, their systems need input of expert knowledge of the structure of Chinese characters, yet they fail to handle cursive characters that are greatly distorted, displaced, and sometimes indistinguishable. In reality, daily handwriting tends to produce more cursive and thus continuous strokes.

In this paper, we propose a novel approach to Chinese handwriting generation by building a collection of components called *StrokeBank*. By means of the hierarchical nature of Chinese characters, our model decomposes Chinese characters into a tree of components, where each component becomes an element in the bank. In this way, the model builds a ground truth StrokeBank using a standard Chinese font. When learning personalized handwriting styles, our model decomposes characters into their components. We utilize a similarity-based method to establish the mapping from standard font components in StrokeBank to their handwritten counterparts. The mapped StrokeBank can be considered as a dictionary of character components used for new character generation. To preserve the stylistic variations in handwriting, we further use a probabilistic approach to capturing inter-component spatial layouts. Our experiment results show that the generated characters resemble the ground truth characters in handwritings of two calligraphists with very different degrees of cursiveness, stroke characteristic and

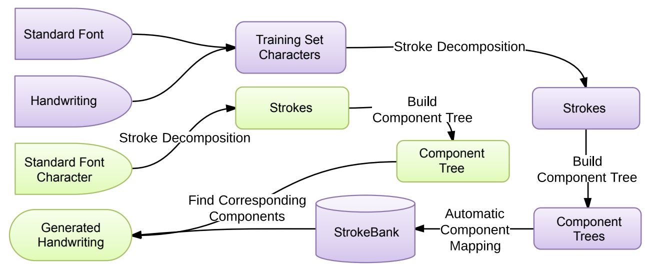


Figure 1: System pipeline. We construct the StrokeBank (purple) that is used to generate handwriting for a test character in standard font (green).

component layout. Fig. 1 shows our system pipeline.

Our major contributions to Chinese handwriting generation are twofold: 1) components instead of simple strokes are used to capture the cursiveness of handwriting characters; 2) a semi-supervised framework is introduced to construct StrokeBank with minimum human supervision.

Related Work

Handwriting generation is an active research field in the last decade. In this section, we briefly introduce recent works in handwriting synthesis in other languages, and discuss works in Chinese handwriting generation in greater detail.

Handwriting Generation

Researchers have made efforts in handwriting generation in various languages. The majority of existing algorithms focus on English (Lin and Wan 2007; Graves 2013), where the cursiveness and spatial layout are relatively simple, while only a few attempts have been made to synthesize languages (e.g. Indian languages, Korean, Arabic) with more complex characters (Jawahar and Balasubramanian 2006; Lee and Cho 1998; Dinges et al. 2011). However, most models deal with characters with relatively simple structure and with limited cursiveness. On the other hand, the complex layouts and variations in Chinese characters demand a more elaborate model to generate handwritings with high fidelity.

Chinese Handwriting Generation

Research in Chinese handwriting generation has largely focused on the automatic synthesis of artistic calligraphy (Xu et al. 2005). The most related work to ours is from (Xu et al. 2009), which uses a shape grammar rule to decompose words into a hierarchical representation. It then generates stylized characters by maximizing a likelihood estimate that captures personalized handwriting characteristics. However, there are three major drawbacks to the model: 1) it makes strong assumptions on the structure of handwriting characters, which do not hold for joined-up writing, where the strokes are largely cursive and connected; 2) expert knowledge is required to design a complete set of shape grammars; 3) human supervision is needed to correctly extract strokes so the shape grammars can be applied.

In contrast, our model aims to construct a StrokeBank with little human supervision. Instead of applying expert knowledge to decompose Chinese characters, we propose an automatic approach to extracting component and building a StrokeBank for handwriting generation.

Stroke Extraction

Stroke extraction is usually the first step in analyzing Chinese characters. A model to utilize degree information and stroke continuation property was proposed to perform stroke segmentation (Cao and Tan 2000). This model addresses two major problems in stroke extraction: identifying primary strokes and resolving ambiguities at intersection points. (Liu, Kim, and Kim 2001) proposed a model-based algorithm to extract strokes and inter-stroke relations by matching them with a reference character described in an attributed relational graph.

The StrokeBank Model

Problem Formulation

We propose a component-based model to tackle the Chinese handwriting generation problem. We first build a StrokeBank containing a collection of components in a standard Chinese font mapped to handwritten components. During generation, our StrokeBank is used as a dictionary to retrieve handwritten components from the standard font.

Stroke Extraction

We use a stroke extraction approach similar to (Cao and Tan 2000). For each character pixel in the bitmap, we calculate the orientation distances, the boundary-to-boundary distance on a line that passes through the pixel. We quantize the orientation space $[0, \pi)$ for each pixel into 40 bins. Given the distribution of orientation distances, we find all crests by taking the mean of all distances as a threshold. Each crest represents a possible orientation of the stroke. For example, for a pixel in the character 一 (Yi), it has one crest; for a pixel at the intersection in character 十 (Shi), it has two crests.

Finally, we merge all the possible orientations into a binary 3D space called ρ-space. Each element (x, y, m) is 1 iff the orientation m is in one of the crests of pixel (x, y). An example of a ρ-space is shown in Fig. 2. By looking for connected components in the ρ-space, we are able to find all the strokes of a Chinese character.

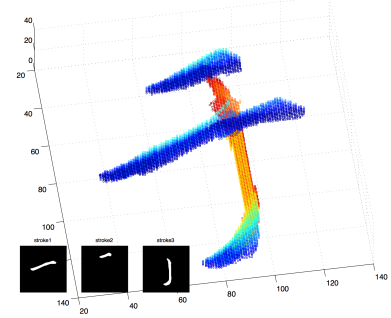


Figure 2: ρ-space and strokes of character 于 (Yu). Color indicates orientation (z-axis): blue for 0 and red for π.

Component Tree Generation

After stroke extraction, we build a tree representing the structure of the character at different decomposition levels. Since it is possible to decompose the character in different ways, for each character we generate a forest consisting of variations of trees. Each tree is constructed bottom-up, with leaves being the most finely decomposed strokes. Strokes are grouped together based on simple heuristics, for example, when they cross each other, share the same joint, or lie in close proximity with one another. Finally, the root of a tree is the whole character itself. Fig. 3 shows a tree learned from decomposing a Chinese character 说 (Shuo).

Feature Extraction

We extract two features from each component in the tree.

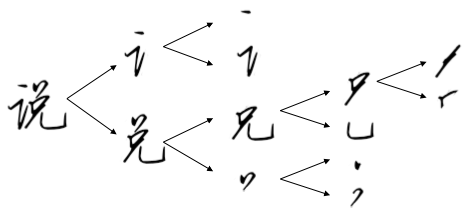


Figure 3: A tree of components obtained from one possible decomposition of a Chinese character 说 (Shuo).

Directional element feature (DEF) The directional element feature vector is well suited to describe stroke-based characters (Kato et al. 1999), such as those in Chinese and Japanese. Similar to the procedure proposed by Kato et al., the contour of a component is computed (Fig. 4). Then for each black pixel centered in a 3×3 mask, we assign a direction or a combination of directions. We only consider four spatial orientations: horizontal, vertical, left- and right diagonals. To improve robustness against spatial dislocation, we perform spatial pooling (Fig. 4) by counting the number of each orientation assigned to the black pixels in a subregion. If we denote the feature vector of each subregion by $x_1, \ldots, x_4$ where x_i is the count of black pixels assigned with direction i, we compute x_i as

$$x_i = x_i^{(A)} + 2x_i^{(B)} + 3x_i^{(C)} + 4x_i^{(D)}, i = 1, \ldots, 4. \quad (1)$$

As we use 49 subregions, the final DEF feature vector has 196 dimensions.

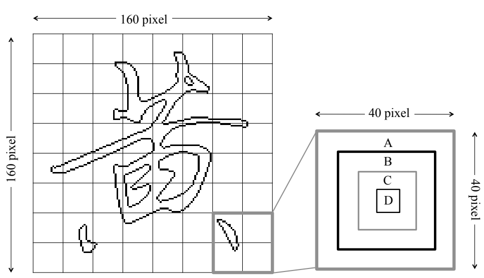


Figure 4: Left: the contour of a character 黄 (Huang) placed in an 8×8 grid. Right: the schema for spatial pooling of direction elements in each subregion (Kato et al. 1999).

Fourier spectra The second feature is constructed from the Fourier spectra of the component bitmap. Inspired by the application of Fourier spectra in face recognition (Spies and Ricketts 2000), we build a $2k$ dimensional vector using both the real and imaginary parts of the top k most variant frequencies from the lower quadrant. The DC component is set to zero in order to minimize the negative effect of spatial displacement. We adopt $k = 20$ in our model.

The final representation is the concatenation of DEF and Fourier spectra vectors. The full feature vector therefore has $196 + 40 = 236$ dimensions.

Automatic Component Mapping

Given two collections of component trees (i.e. two forests) generated for a standard Chinese font and a particular handwriting, we would like to automatically learn a mapping function between components in the standard font and its handwritten counterparts. We solve the mapping problem in a semi-supervised framework (Zhu 2005). A small set of manual mapping is used with a large amount of unlabeled data to learn a robust and reliable mapping function.

Mapping of the DEF vector The DEF vector captures the stroke orientation information of a component. Here, we assume that the DEF vector of handwritings can be approximated by an affine transform of the DEF vector of the standard font. This assumption is reasonable, especially for components consisting of strokes that have little curvature. As a simple example, the character 十 (Shi) usually has a perfectly horizontal and a perfectly vertical stroke in the standard font. However, in most handwritings, the horizontal stroke is tilted but the vertical stroke remains perfectly vertical. An affine transform is capable of capturing this rotational variation. Though for more complicated components, this assumption does not capture all the nuances in the mapping from standard font to a handwriting, in practice our model performs well.

We adopt a multivariate linear regression model to learn the affine transform. Given a standard font component $X^{(i)}$ and its mapped handwritten component $Y^{(i)}$, let $f(X^{(i)})$ and $f(Y^{(i)})$ be the DEF feature vectors. Our hypothesis says $f(Y^{(i)}) = \theta f(X^{(i)})$, and the closed form solution is:

$$\theta = (\mu\mathbb{I} + f(\mathcal{X})^T f(\mathcal{X}))^{-1} f(\mathcal{X})^T f(\mathcal{Y}) \quad (2)$$

where μ is the regularization parameter and $\mathbb{I}$ is the identity matrix. $\mathcal{X}$ and $\mathcal{Y}$ represent the collection of all labeled components in the standard font and a particular handwriting. Thus, the i-th row in $f(\mathcal{X})$ represents $f(X^{(i)})$; the same for $f(\mathcal{Y})$. During training, we use cross-validation to choose the appropriate value of μ. We set $\mu \sim ||f(\mathcal{X})^T f(\mathcal{X})||_F$, where $||\cdot||_F$ stands for the Frobenius norm.

Weighted affinity of the full feature vector Successful automatic mapping heavily relies on a good affinity measure between the *full* feature vectors including both DEF and Fourier spectra.

One naïve way is to compute a distance measure (e.g. Euclidean and χ-square) between the affine transformed feature of standard font component and the feature of handwritten component. However, this approach performs poorly due the varying dimensions and scales of different features in the concatenation. Methods in combining multiple types of features have been recently proposed (McFee and Lanckriet 2011; Song and Zhu 2013; Wang et al. 2014). Here, we propose an optimization approach that learns weights w for each feature dimension.

Let $\phi(X^{(i)})$ and $\phi(Y^{(i)})$ be the *full* feature vectors of $X^{(i)}$ and $Y^{(i)}$. As discussed, the first 196 dimensions of $\phi(\cdot)$ is the DEF vector $f(\cdot)$ and the rest is the Fourier spectra vector. Let Θ be the affine transform operator as discussed in the previous subsection. Note that Θ only operates on the DEF vector while leaving the Fourier spectra vector unchanged. In matrix representation,

$$\Theta = \begin{bmatrix} \theta & 0 \\ 0 & \mathbb{I} \end{bmatrix}. \tag{3}$$

Let the weighted distance metric between *any* pair of components in the standard font X and a handwriting Y be

$$dist(\Theta(X), Y) = \sum_{j=1}^{d} w_j \left|(\Theta\phi(X))_j - \phi(Y)_j\right| \tag{4}$$

where d is the dimension of the full feature vector and $w_j, (\Theta\phi(X))_j$ and $\phi(Y)_j$ are the j-th component in the vector. $dist(\Theta(X), Y)$ can also be written as $w^T D(X, Y)$, where $D(X, Y) = |(\Theta\phi(X))_j - \phi(Y)_j|, j = 1, 2, \dots d$.

If $X^{(i)}$ and $Y^{(i)}$ correspond to the same component, we want $dist(\Theta(X^{(i)}), Y^{(i)})$ to be smaller than $dist(\Theta(X^{(i)}), Y), \forall Y \neq Y^{(i)}$. Therefore, we formulate the learning problem in the following quadratic programming (QP) scheme:

$$w^* = \arg\min_{w,\xi} \frac{\lambda}{2} w^T w + \sum_i \xi_i \tag{5}$$

such that

$$w^T D(X^{(i)}, Y^{(i)}) - w^T D(X^{(i)}, Y) \geq 1 - \xi_i, \ \forall Y \neq Y^{(i)}$$
$$\text{and } \ \xi_i \geq 0, \qquad \forall i$$

where λ is the regularization parameter, ξ_i is the slack variables for handling soft margins, and i goes through all the labeled mappings. The optimization problem in Eq. (5) is equivalent to $w^* = \arg\min_{w,\xi} \frac{\lambda}{2} w^T w + R(w)$, where $R(w)$ defines a hinge loss function:

$$R(w) = \sum_i \max(0, R^{(i)}(w)), \text{ where}$$
$$R^{(i)}(w) = \max_{Y \neq Y^{(i)}} \left[1 + w^T \left(D(X^{(i)}, Y^{(i)}) - D(X^{(i)}, Y)\right)\right]$$

The optimization problem is convex and we solve it with an off-the-shelf solver (Do 2009). Table 1 compares the error rates of our automatic mapping before and after using the weights w^*. The improvement is clear.

Table 1: Mapping Error Rate

without w^*	4.63%
with w^*	0.77%

Interpretation of the weighted affinity A larger weight in w^* corresponds to a more discriminative feature dimension. One interesting observation is that the average weight w^*_{outskirt} for DEF features in the outskirt of a character is much larger than that in the core area w^*_{core}, where core and outskirt regions are labeled in Fig. 5. The larger weight distribution towards the outskirt agrees with the finding of previous work (Kato et al. 1999) and agrees with theories of Hui Gong Grid, which were developed by a famous Chinese calligraphist Weiguo Yang to better visualize the structure of Chinese characters. Specifically, strokes extending to the outskirt define the overall aesthetics of the character, and contribute more to its defining characteristics. On the other hand, strokes in the core region are more compact and subject to deformation, hence adding more noise to the DEF feature and getting less weight.

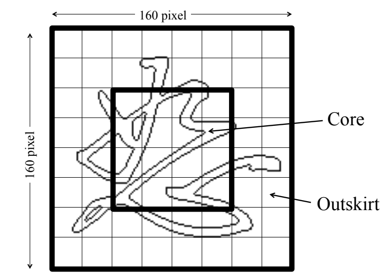


Figure 5: Labeled outskirt and core regions of a character 恕 (Shu). Two bold borders resemble Hui Gong Grid.

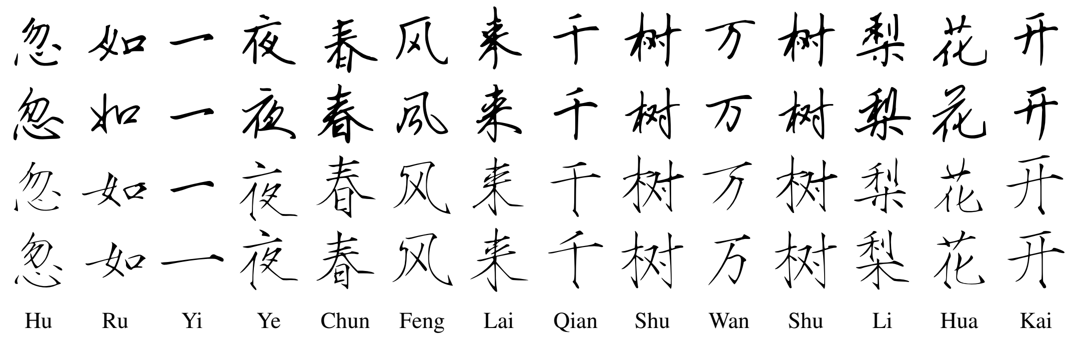
Handwriting Generation

Once we have the mapping dictionary, we are able to generate new personalized handwriting. Given an input character, we first decompose it into a set of components $\{Z^{(1)}, Z^{(2)}, \dots, Z^{(m)}\}$. Since each component consists of a set of basic strokes S, let $S^{(i)}$ be the set of basic strokes $Z^{(i)}$ covers. We want to find a set of components $\{Z^{(a_1)}, Z^{(a_2)}, \dots, Z^{(a_n)}\}$ such that $S = S^{(a_1)} \cup S^{(a_2)} \cup \dots \cup S^{(a_n)}$ and $S^{(a_i)} \cap S^{(a_j)} = \emptyset$ for $i \neq j$. We measure the similarity from $Z^{(k)}$ to an entry in the dictionary $(X^{(i)}, Y^{(i)})$ as:

$$s(Z^{(k)}, X^{(i)}, Y^{(i)})$$
$$= e^{-||f(Z^{(k)}) - f(X^{(i)})||_2} + e^{-dist(\Theta(X^{(i)}), Y^{(i)})} + \nu|S^{(k)}|$$

where $f(\cdot)$ is the DEF vector, $|S^{(k)}|$ is the cardinality of the set $S^{(k)}$ and ν is a positive constant. The $\nu|S^{(k)}|$ term models the desired behavior that if we manage to find a well matched component that consists of many strokes in the dictionary, we bias towards using this larger component. This biased selection will be robust against cursive handwriting for which substituting individual strokes would result in a disconnected, unnatural character as a whole. Formally speaking, the bias preserves more correlation such as relative spatial position, size, and orientation between individual strokes. We adopt $\nu = 0.2$ in our experiment.

The generation problem is reduced to finding a valid set $A = \{a_1, a_2, \dots, a_n\}$ such that $\sum_{j=1}^{n} s(Z^{(a_j)}, X^{(i)}, Y^{(i)})$ is maximized. Solving the exact inference problem is NP-hard; we adopt an efficient greedy forward search approach. Once we retrieve these handwritten components, we align them based on the centroid of the original component.



(a) An excerpt from a Tang poem "忽如一夜春风来千树万树梨花开".

(b) An excerpt from an famous essay *Lou Shi Ming*: "山不在高有仙则名水不在深有龙则灵".

Figure 6: Generated handwritten characters using our StrokeBank approach, as compared to the ground truth. In both (a) and (b), the 1st and 2nd rows are written in Xing, and 3rd and 4th rows in Shou Jin. For each font, the first line is generated while the second line is the ground truth. Phonetic labels are shown at the bottom of each column.

Modeling stylistic variations The same character written by the same person can vary in shape from time to time due to intrinsic uncertainties in human handwriting. We capture this slight variation in styles by learning the probability distribution of the position of a component in the character. For a particular component, let $\mathcal{P}(x, y|x_o, y_o, \mathcal{F})$ be the probability of it being written at position (x, y), given the standard position (x_o, y_o) and some generic features of the component $\mathcal{F}$, such as the combined DEF and Fourier spectra vector. All positions refer to the coordinates of the centroid. For almost all components, (x, y) is clustered around (x_o, y_o) in very close proximity; we thus model $\mathcal{P}$ as a multivariate Gaussian independent from $\mathcal{F}$.

We train $\mathcal{P}$ by observing the positions of the same radical that appears in different characters of the same handwriting. Radicals are chosen because they are generally located in the same position despite drastic variations in other parts of the character. A standard deviation of roughly 2 pixels in either x- or y-direction is obtained, which is applied to aligning the centroid of a retrieved component away from its standard position according to $\mathcal{P}$.

Evaluations

Datasets

We use Kai, a publicly available font, as our standard font because it is widely seen in presswork and official documents. Two handwriting styles, Xing and Shou Jin, are used for evaluation. Xing is cursive, while Shou Jin is notable for the unique endpoints in its strokes. The characters we used in Xing and Shou Jin are based on handwritings of Jinsheng Qiu and Yingzhang Tian respectively; both are well-known calligraphists. Xing and Shou Jin are chosen because they represent very different handwriting styles in terms of cursiveness, stroke characteristics, and component layouts.

Metric

Since the exact quality of computer-generated characters is hard to measure, we conduct a survey-based method to quantitively evaluate the effectiveness of our model. We invited 20 well-educated Chinese scholars to distinguish generated characters from ground truth handwritten characters. The respondents are first given 20 sample ground truth handwriting characters of each style as reference. They are then shown a sheet of 100 characters, mixing ground truth and generated ones. They are asked to judge whether a character is generated. We measure the identification accuracy of the k-th person as:

$$A_k = \frac{\text{\# correct decisions made by } k}{\text{\# total decisions made by } k}, k = 1, 2, .., 20 \quad (6)$$

We then calculate the mean identification accuracy $\langle A \rangle$. We expect $\langle A \rangle$ to be close to 50%, which is the expected value if the respondent performs uniform random guessing.

Results

We present a few generated samples here. Fig. 6 shows two famous excerpts written in generated handwriting based on the StrokeBank approach, as compared to the ground truth. Based on visual inspection, our generated characters share similar structure and cursiveness with the ground truth. Since there is no cursiveness in the standard font, the generated cursive character in Xing is a result of our component-based model that captures cursiveness in the dictionary mapping. Examples of generated cursive characters include 来 (Lai) in Fig. 6a (1st row) and 深 (Shen) in Fig. 6b (1st row).

Our survey results are shown in Table 2. For both handwriting styles, the mean identification accuracy $\langle A \rangle \sim 50\%$, which is the mean accuracy of random guess. Despite very different styles in cursiveness, stroke characteristics, and component layouts between Xing and Shou Jin, our model performs well in both, indicating its robustness in large variations in handwritings.

For Shou Jin, $\langle A \rangle = 49.6\%$, which is remarkably close to 50%, indicating that our generated characters very closely resemble the actual handwriting. For certain generated characters in Xing, such as 深 (Shen), 千 (Qian), 无 (Wu) and 梨 (Li), more than 90% of the respondents identified them as the ground truth.

Table 2: Survey results of 20 Chinese scholars

	$\langle A \rangle$	$A_{\min}$	$A_{\max}$
Xing	0.585	0.380	0.870
Shou Jin	0.496	0.360	0.620

For $\langle A \rangle$ and $A_{\max}$, Shou Jin has a much better performance than Xing. This is because for Xing, we cannot reliably make two disconnected strokes in the standard characters cursively connected in the generated ones. One example is 刂 in character 则 (Ze) in Fig. 6b (1st and 2nd rows). This is because all nodes in our component tree are connected bitmaps. During generation, we search for the two individual strokes in 刂 instead of 刂 itself in the dictionary. This reduces the degree of cursiveness in generated Xing characters in certain cases, but can be remedied in future work if we use proximity information to include nodes with disconnected bitmaps in the component tree.

Conclusion

We have presented a novel approach to Chinese handwriting generation based on StrokeBank, a dictionary that maps components in standard font to a particular handwriting. The model does not need expert input about the structure of Chinese characters or human supervision in stroke extraction, yet it is able to capture cursiveness, spatial correlation between strokes, and other font characteristics through component mapping based on weighted features. The generated characters resemble the actual handwritings in both structure and cursiveness. Our survey results also suggest that our generated characters are almost indistinguishable from ground truth handwritings. Future work would include extending this pipeline to other languages and deploying it to applications such as artistic font design and handwritten instant messaging.

References

Amin, A.; Kim, S.-G.; and Sammut, C. 1997. Hand printed chinese character recognition via machine learning. In *ICDAR*, 190–194. IEEE Computer Society.

Cao, R., and Tan, C. L. 2000. A model of stroke extraction from chinese character images. In *ICPR*, 4368–4371.

Dinges, L.; Elzobi, M.; Al-Hamadi, A.; and Aghbari, Z. 2011. Synthizing handwritten arabic text using active shape models. Advances in Intelligent and Soft Computing.

Do, T. A. T.-M.-T. 2009. Large margin training for hidden markov models with partially observed states.

Graves, A. 2013. Generating sequences with recurrent neural networks. *CoRR* abs/1308.0850, abs/1308.0850.

Jawahar, C. V., and Balasubramanian, A. 2006. Synthesis of online handwriting in indian languages. *International Workshop on Frontiers in Handwriting Recognition* 108–113.

Kato, N.; Suzuki, M.; Omachi, S.; Aso, H.; and Nemoto, Y. 1999. A handwritten character recognition system using directional element feature and asymmetric mahalanobis distance. *PAMI* 21.

Lee, D.-H., and Cho, H.-G. 1998. A new synthesizing method for handwriting korean scripts. *IJPRAI* 46–61.

Lin, Z., and Wan, L. 2007. Style-preserving english handwriting synthesis. *Pattern Recognition* 40(7):2097–2109.

Liu, C.-L.; Kim, I.-J.; and Kim, J. H. 2001. Model-based stroke extraction and matching for handwritten chinese character recognition. *Pattern Recognition* 34(12):2339–2352.

Ma, L.-L., and Liu, C.-L. 2008. A new radical-based approach to online handwritten chinese character recognition. In *ICPR*, 1–4.

McFee, B., and Lanckriet, G. R. G. 2011. Learning multi-modal similarity. *Journal of Machine Learning Research* 12.

Song, Z., and Zhu, Y. 2013. Graphical model-based learning in high dimensional feature spaces. In *AAAI*.

Spies, H., and Ricketts, I. 2000. Face recognition in fourier space. *Proc. Vision Interface*.

Stallings, W. 1976. Approaches to chinese character recognition. *Pattern Recognition* 8(2):87–98.

Wang, B.; Mezlini, A. M.; Demir, F.; Fiume, M.; Tu, Z.; Brudno, M.; Haibe-Kains, B.; and Goldenberg, A. 2014. Similarity network fusion for aggregating data types on a genomic scale. *Nature methods*.

Xu, S.; Lau, F. C. M.; Cheung, W. K.; and Pan, Y. 2005. Automatic generation of artistic chinese calligraphy. *IEEE Intelligent Systems* 20(3):32–39.

Xu, S.; Jin, T.; Jiang, H.; and Lau, F. C. M. 2009. Automatic generation of personal chinese handwriting by capturing the characteristics of personal handwriting. In *IAAI*.

Xu, S.; Jiang, H.; Lau, F. C. M.; and Pan, Y. 2012. Computationally evaluating and reproducing the beauty of chinese calligraphy. *IEEE Intelligent Systems* 27(3):63–72.

Zhu, X. 2005. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison.