JKanji: Wavelet-based Interactive Kanji Completion

Robert Stockton[1*]
rgs@justresearch.com

[1]Just Research
4616 Henry Street
Pittsburgh, PA 15213
U.S.A.

Rahul Sukthankar[1,2 †]
rahuls@cs.cmu.edu

[2]The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213
U.S.A.

Abstract

JKanji is an interactive character completion system that provides stroke-order-independent recognition of complex hand-written glyphs such as Japanese kanji or Chinese hanzi. As the user enters each stroke, JKanji offers a menu of likely completions, generated from a robust multi-scale matching algorithm augmented with a statistical language model. Drawbacks of traditional wavelet-based approaches are addressed by a redundant, phase-shifted basis that is insensitive to variations of the input character across quadrant boundaries. Unlike many existing systems, JKanji can incrementally incorporate new training examples, either to adapt to the idiosyncrasies of a particular user, or to increase its vocabulary. On a kanji input task with a vocabulary of 6369 kanji and English characters, JKanji has demonstrated 93%–96% recognition accuracy and up to 80% reduction in the number of input strokes. JKanji is computationally efficient, processing images at 5–10Hz on an inexpensive portable computer, and is well-suited for integration into personal digital assistants (PDAs) as an input method. JKanji's recognition system also processes low-quality digital camera images and has been integrated into a prototype tourist's guide *that interprets unfamiliar kanji in the environment.*

1. Introduction

Traditional computer input devices, such as keyboards, are ill-suited for languages with large vocabularies of complex glyphs (e.g., Japanese kanji or Chinese hanzi). Research has therefore focused on alternative input methods such as optical character recognition [6] or stroke-based handwriting recognition [3]. The former has proved successful in processing scanned documents, while the latter has become increasingly common in interactive applications.

Kanji completion is an interactive task that combines certain aspects of both character recognition and handwriting recognition. However, kanji completion differs from these tasks in two important respects: (1) a kanji completion system must endeavor to match a user's partially-drawn glyphs against the (complete) glyphs in the training set as early as possible; (2) a kanji completion system can rely on the user to select the correct glyph from a small menu of likely completion candidates at each stage of the process. We present JKanji, an approach to the task of interactive glyph input that addresses common problems with existing input methods by employing techniques inspired by work in sketch-based image retrieval [2]. Our system has three main goals: (1) increasing the speed of stroke-independent[1] glyph input; (2) incrementally incorporating additional training data (either new characters in the vocabulary or idiosyncratic examples of known glyphs); (3) providing a solution suitable for handheld devices (low requirements on computing power and memory usage).

JKanji has been integrated into a text-editor targeted for non-native kanji users (see Figure 1) and is also a component in a prototype *tourist's guide*. The latter is a device equipped with a digital camera, capable of interpreting common kanji signs from images (see Section 3).

*Robert Stockton **(rstock@whizbang.com)** is now with WhizBang! Labs–Research, 4616 Henry Street, Pittsburgh, PA 15213, U.S.A.

†Rahul Sukthankar **(rahuls@cs.cmu.edu)** is now with Carnegie Mellon University and Compaq Computer Corporation (Cambridge Research Lab), One Cambridge Center, Cambridge, MA 02142, U.S.A.

[1]While native speakers are taught to use a canonical stroke order when entering characters, non-native speakers (who may benefit most from a character completion system) typically do not. JKanji can also be used to enter user-defined characters that have no canonical stroke order.

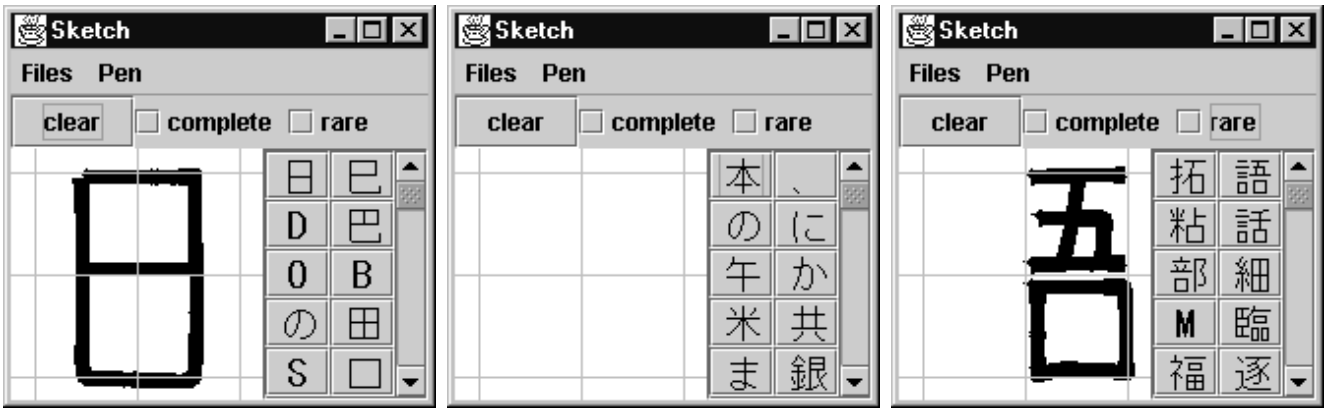

Figure 1. JKanji significantly facilitates interactive kanji input. Here, the user is entering the phrase "Japanese language". Upon confirmation that the first glyph is "sun", JKanji's language model immediately suggests "book" (the combination means "Japan"). The third glyph, recognized after the first few strokes, demonstrates partial matching. JKanji is not stroke-order dependent.

2. System Architecture

JKanji consists of several components, shown in Figure 2. The input to the system consists of a sequence of images: snapshots of a sketchpad window[2], captured at the conclusion of each glyph stroke (up to 10Hz). These images are first preprocessed to ensure consistency in line thickness, and scaled appropriately. Next, a feature vector is extracted from each pre-processed image using a set of redundant wavelet decompositions. These feature vectors are matched against stored feature vectors corresponding to the training images, and the best matches (weighted by several language models) are presented to the user. This asymmetric matching process is designed so that a partial glyph in the input image correctly matches the respective complete glyph in the training data. As the user adds strokes to the glyph in the input area, JKanji recomputes the wavelet features and updates the candidate list; this process repeats until the desired glyph appears and is selected by the user. At this point, dynamic language model parameters are updated and JKanji clears the input area in preparation for a new character. JKanji's language model enables it to frequently predict likely glyphs before the user enters *any* strokes (see Figure 1), thus drastically reducing the input effort. The major components are detailed below.

2.1. Training set preparation

JKanji can incorporate training data from a variety of sources. In the experiments described in this paper, one image for each of 6369 kanji was synthesized from the *MS Gothic* font and processed as described below to create a single training example per glyph in the vocabulary[3]. The user may easily augment the training set, either when difficulties in matching a known glyph are experienced (adapting to user idiosyncrasies), or when he/she wishes to add a new glyph; the wavelet features for the current input image (assumed to be a complete glyph) are simply appended to the vocabulary of known kanji characters. The language models (described in Section 2.5) are initialized from a corpus of Japanese natural language text.

2.2. Image pre-processing

Unlike traditional handwriting input systems, JKanji does not extract features from a time-series of the user's gestures. Rather, the snapshot of each successive version of the sketched glyph is processed independently. This lack of temporal information results in two important benefits: (1) JKanji is insensitive to stroke-order; (2) JKanji can easily be applied to noisy input from scanned text or photographs (see Section 3).

Each image is first processed to ensure that strokes are of uniform width using a heuristic algorithm adapted from [6]: *shrink* or *grow* filters are successively applied until the *thickness metric* for the input image matches that of the images used during training.

JKanji creates two copies of the input image: one to test the hypothesis that the image is a complete glyph, and the other to attempt partial matches. The former is centered and then robustly scaled so that 75% of the dark pixels lie within a specified "central region"; the latter is scaled with-

[2]Glyphs are entered using a mouse-driven sketchpad with physics-based smoothing (inspired by *DynaDraw* [1]).

[3]Training and input images are treated identically in JKanji except during the asymmetric matching process.

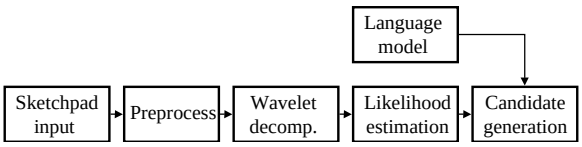

Figure 2. Overview of JKanji system.

out centering (centering would destroy information about subfeature location). Wavelet features for each of these two images are independently computed, and matched against candidate glyphs in the training set as described below.

2.3. Wavelet decomposition

Wavelet-based approaches are becoming increasingly popular in pattern recognition, and have recently been applied to character recognition [4]. However, JKanji's wavelet decomposition is unrelated to prior research in this area; it was inspired by work in sketch-based image retrieval [2]. For background material on wavelets, see [5, 7].

A serious drawback with employing a wavelet-based approach in this domain is that the decomposition is overly sensitive to small variations in the neighborhood of quadrant boundaries. Kanji characters are particularly prone to this problem since they commonly contain centered vertical and horizontal strokes. Slight shifts in such a stroke's position can cause significant changes in the wavelet coefficients. To alleviate this problem, JKanji employs a novel redundant basis: (1) three copies of the input image are made, with the pixels in each "barrel shifted" (toroidal translation) up and to the left by 0, $1/5$, and $1/3$ of the image width; (2) 128×128 wavelet decompositions are independently performed on each of the three images; (3) the most important coefficients (see below) from each of these decompositions are retained. Since the barrel shift multiples share no common factors, a particular stroke in the original input image will lie on a critical region in at most one of the three decompositions (see Figure 3).

The images are filtered (both horizontally and vertically) using a simple variant of the Haar wavelet[4]. Although the Haar basis is rarely the best choice for filtering natural images, our experiments have shown that it outperforms Daubechies and cubic spline bases in the kanji completion domain. We hypothesize that this is because the binary images created by the sketchpad application consist of crisply separated black and white regions (see Figure 1) whose sharp discontinuities are effectively expressed by the Haar basis.

[4]Identical to the Haar wavelet except: (1) scaling constants are omitted; (2) averages are replaced by sums. This integer variant is more efficient than the standard Haar. The scaling constants for coefficients in different frequency bands is partially subsumed in the w_{ij} weights (see Section 2.4).

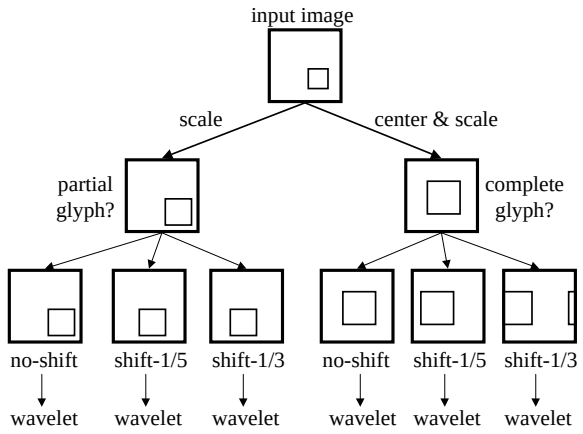

Figure 3. JKanji considers two independent hypotheses: that the input glyph will partially or completely match glyphs in the training set. To overcome quadrant effects present in standard wavelet decompositions, three redundant wavelet coefficients are extracted by barrel shifting each scaled input image.

From the 128×128 wavelet coefficients in each of three images, the signs and coordinates of the 40 coefficients with the greatest magnitude are retained as features. Each glyph can be compactly expressed as a list of 120 signed integers (2 bytes each). Therefore, the 6369 kanji in JKanji's training set consume only $6369 \times 40 \times 3 \times 2 \approx 1.5$ MB of storage space (small enough to comfortably fit into a handheld computer's RAM).

2.4. Likelihood estimation

In this step, the 120 features of the input image (probe) are compared against the stored features of every kanji (target) in the training set. JKanji employs an asymmetric weighted comparison metric in order to reward similarity between probe and target without overly penalizing missing strokes in the probe kanji. Specifically, the score for a target kanji is increased whenever a coefficient in the probe appears in the target (with the same sign). The increment $w_{i,j}$ is a function of the coefficient's coordinates:

$$w_{i,j} = b^{\lfloor \log_2 \max(i,j) \rfloor}$$

where $b = 1.2$ (determined empirically). Since two versions of the wavelet decomposition are performed per kanji (for partial and complete matches), each kanji is assigned a final score that is the maximum of its two scores. This score is used to determine an ordering of the training set kanji (in response to the given input glyph).

The final step of this stage is to convert the ranking into likelihood estimates. The target kanji are assumed to follow Zipf's law [8] and accordingly, each kanji is assigned a likelihood proportional to the inverse of its rank.

2.5. Language models

The likelihood estimates implicitly assume a uniform prior distribution on every kanji in the training set. This is almost certainly violated when the input stream consists of a coherent sequence of well-formed, meaningful text. JKanji therefore considers three additional priors: (1) a unigram model based upon a large corpus of Japanese text; (2) a bigram model based upon the same corpus[5]; (3) a unigram model based solely upon the user's input to the kanji editor. Note that (2) and (3) are dynamically updated as the user enters text.

The user may selectively enable or disable particular language models. For instance, a user using JKanji simply for dictionary lookup could disable the bigram model, or could restrict JKanji to the uniform prior.

2.6. Candidate merging

The goal of this stage is to generate a single list of kanji candidates, to be displayed to the user. JKanji uses an unorthodox scheme that has produced excellent empirical results: the rankings according the each language model is independently computed by multiplying the likelihoods for each kanji by its respective prior generated from the model. These ordered rankings are now interleaved (with duplicates removed) to create a menu of candidates (see Figures 1 and 4).

3. Tourist's Guide

Tourists who are unable to read kanji could benefit substantially from a handheld device that recognized glyphs in their environment (e.g., shop signs or posters). We have developed a prototype application that enables users to snap a picture with a handheld digital camera and select an unfamiliar glyph in the image. JKanji processes this low-quality image and presents a menu of likely matches. The user may move a cursor over each match to obtain a translation. For instance, in Figure 4, JKanji correctly recognizes the glyph for "son" from a blurred and noisy image. This task is substantially more difficult than the text editor application since: (1) creating a good binary input image from the noisy greyscale photograph is non-trivial; (2) JKanji is unable to derive benefits from its bigram language model; (3) the appearance of the glyph may differ substantially from its single prototype in the training set (*MS-Gothic*). Nevertheless, as discussed in Section 4, JKanji's performance is surprisingly promising.

[5]Such a model can give the probability $P(x|y)$ of seeing a particular kanji, x, given that the previous kanji was y.

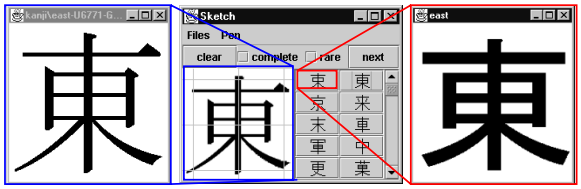

Figure 5. JKanji correctly recognizes 93% of 6369 glyphs from *MS-Mincho* even though it was trained on glyphs from the *MS-Gothic* font.

4. Results

We first present baseline results that demonstrate the competency of JKanji's glyph recognition system. In this experiment, JKanji was trained on *MS-Gothic* glyphs and tested on each of the glyphs in the *MS-Mincho* font. Although the glyphs differ in appearance (see Figure 5), JKanji achieved a 96.2% accuracy on the 2521 common glyphs and 93.0% on the complete set. In this experiment, JKanji received no benefit from its language models since the unigram and bigram frequency statistics of the test glyphs were not consistent with the natural language corpus for initialization.

An instrumented version of the kanji text editor was tested by several users (both native Japanese speakers and novices). Since recognition accuracy is not meaningful in this interactive context (users modify the glyph until JKanji offers the correct completion), we present statistics on kanji completion: JKanji reduced the number of input strokes required to enter Japanese text by 50%-80% (greater benefits were achieved when the input was consistent with the bigram language model). Novice users who were unfamiliar with canonical kanji stroke order reported the greatest benefits.

Finally, the tourist's guide prototype was tested on a set of low-quality images captured using a handheld digital camera. JKanji performed surprisingly well on this challenging task (see Figure 4), correctly recognizing 76% of the glyphs (in the absence of a bigram language model).

5. Conclusion

JKanji demonstrates that an interactive kanji completion system can significantly reduce the effort required to enter complex glyphs. JKanji has been successfully integrated

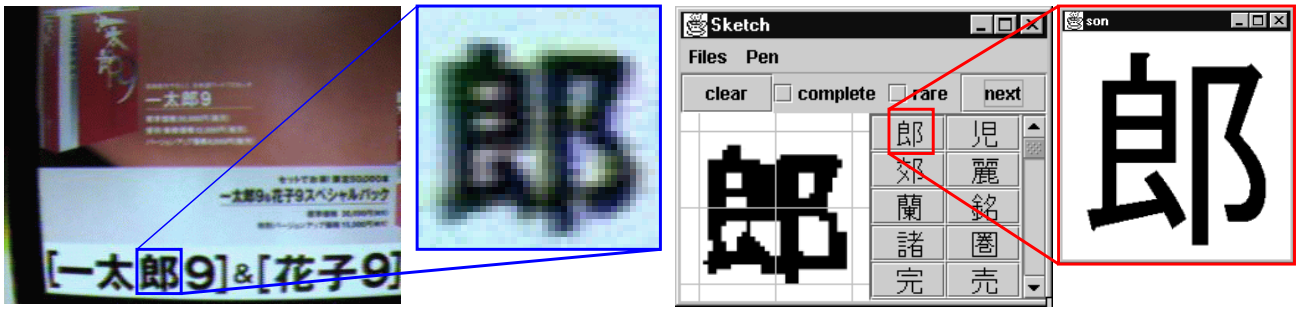

Figure 4. JKanji can also recognize kanji characters in photographs or scanned documents. A glyph extracted from a low-quality image of a poster (left) is shown enlarged (center). The correct identification is also shown (right).

into two prototype applications: a kanji text editor, and a tourist's guide.

6. Acknowledgments

Thanks to Justsystem Corporation for the Japanese document training data, researchers at Just Research for evaluating JKanji text editor, and Gita Sukthankar for valuable feedback on this paper.

References

[1] P. Haeberli. Dynadraw: A dynamic drawing technique, 1989. <http://www-europe.sgi.com/grafica/dyna/>.

[2] C. Jacobs, A. Finkelstein, and D. Salesin. Fast multiresolution image querying. In *Proceedings of SIGGRAPH*, 1995.

[3] H. Kim, J. Jung, and S. Kim. Online chinese character-recognition using art-based stroke classification. *Pattern Recognition Letters*, 17(12), 1996.

[4] T. Shioyama, H. Wu, and T. Nojima. Recognition algorithm based on wavelet transform for handprinted chinese characters. In *Proceedings of ICPR98*, 1998.

[5] G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley Cambridge, 1997.

[6] R. Suchenwirth. *Optical recognition of Chinese characters*. Braunschweig, 1989.

[7] W. Sweldens. The wavelet home page. <http://www.wavelet.org/>.

[8] G. Zipf. *Psycho-Biology of Languages*. Houghton-Mifflin, 1935.