Investigating Co-adaptation in a Handwriting Recognition System

1st Author Name
Affiliation
Address
e-mail address
Optional phone number

2nd Author Name
Affiliation
Address
e-mail address
Optional phone number

3rd Author Name
Affiliation
Address
e-mail address
Optional phone number

ABSTRACT

Handwriting is a natural and versatile method for human-computer interaction, especially on small mobile devices such as smart phones. However, as handwriting varies significantly from person to person, it is difficult to design handwriting recognizers that perform well for all users. A natural solution is to use machine learning to adapt the recognizer to the user. One complicating factor is that, as the computer adapts to the user, the user also adapts to the computer and probably changes their handwriting. This paper investigates the dynamics of co-adaptation, a process in which both the computer and the the user are adapting their behaviors in order to improve the speed and accuracy of the communication through handwriting.

The contributions of this paper are in three complementary directions. First, we devise an information-theoretic framework for quantifying the efficiency of a handwriting system where the system includes both the user and the computer. From this framework, we derive two performance measures. Second, we combine these performance measures with an adaptive handwriting recognition system to create a collaborative game for improving the communication through handwriting over time. Finally, we perform a statistical analysis of the systems performance on the data collected from 15 different users over multiple sessions and characterize the impact of machine adaptation and of human adaptation.

Author Keywords

Co-adaptation; handwriting recognition; communication channel;

ACM Classification Keywords

H.5.2 Information Interfaces and Presentation (e.g. HCI): User Interfaces

General Terms

Algorithms; Measurement; Human Factors

Submitted to IUI'14.
Do not cite, do not circulate.

INTRODUCTION

One of the challenges brought on by the miniaturization of mobile computing devices such as smart phones and tablets is the difficulty of entering information into the device. The communication bandwidth from the device to the human, utilizing high-resolution screens and high fidelity sound, is very high. However, the bandwidth from the human to the device is severely constrained by the size of our fingers and by the difficulty of performing voice recognition in noisy environments.

As the screen real estate becomes scarce, the standard soft-keyboard can only fit up to about 40 fingertip-sized keys on one screen. While 40 keys are sufficient for languages with a small set of characters such as English, it is not suitable for languages with more characters such as Thai or Chinese which contains more than 60 and thousands of different characters respectively. For such languages, as well as in the multilingual settings, the users are required to either repeatedly switch between different keyboard layouts in order to find the desired character.

Many alternatives to the standard soft-keyboard exist. For example, Swype® provides a method for tracing a path between keyboard keys and lifting the finger from the screen at the end of each word. Dasher [4] is a particularly innovative method where typing is replaced by using a joystick-like pointer to fly through clouds of characters. Finally, there are handwriting recognition software that allow the user to enter information using natural handwriting.

A user of any one of these methods typically improves significantly with practice. There are many competitions between different data entry methods. However, these comparisons are inherently flawed in that the contender is always a person who can enter information faster than the current record holder, most likely as a result of extensive training. In other words, *user adaptation* cannot be ignored.

To evaluate the efficiency of a particular data entry method, it is therefore necessary to measure the performance of each user over a sufficiently long period of time so that the performance of the user stabilizes. As we are interested in machine learning methods, we arrive at the interesting situation in which both the computer and the human adapt over time in an effort to maximize the input rate of the data entry method. We refer to this situation as *co-adaptation*.

Designing an intelligent system that co-adapts with the users is a challenging problem on its own [5, 8, 7]. Our goal in this

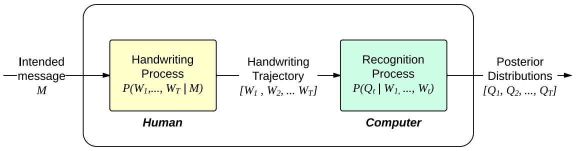


Figure 1: A summary of the handwriting recognition channel.

paper is not to address those challenges, but rather to focus on characterizing the impact of machine adaptation and of human adaptation in the context of handwriting recognition.

The paper is organized as follows. First, we propose an information-theoretic framework for quantifying the efficiency of a handwriting system where the system includes both the user and the computer. Next, we describe our adaptive handwriting recognition algorithm that we developed for our experiment. Then, we describe the experiment and present the results in terms of the performance measures derived from the proposed framework Finally, we draw some conclusions.

HANDWRITING RECOGNITION AS A COMMUNICATION CHANNEL

Unlike typing, which transmits information to the computer at discrete time points, handwriting continuously transmits information as the writer creates the trajectory. Traditionally, handwriting data is analyzed one "unit" at a time where "unit" can be a stroke, a character, a word or even a sentence. In this work, we propose an alternative analysis where the data is analyzed in fixed intervals of time. We consider the process of writing as a process through which the intended letter is disambiguated from the other possible letters.

We formalize this process using the concept of a communication channel [12]. Let $\mathcal{E}$ denote the set of all possible input. Technically, $\mathcal{E}$ can be a set of sentences, a set of words, or a set of characters. Without loss of generality, in this work, we assume that $\mathcal{E}$ is a set of 26 English characters. We also ignore dependencies between characters due to the language model and due to the co-articulation effects between neighboring handwritten characters.

As shown in Figure 1, the channel is comprised of two separate processes. First, the handwriting process is the process of which the user translates an intent $M \in \mathcal{E}$ into a series of hand movements which is sampled at some rate to create a discrete time trajectory: $W_{1:T} = [(x_1, y_1), \ldots, (x_T, y_T)]$. In other words, this process *encodes* the intent M into a trajectory $W_{1:T}$. The distribution $P(\overline{W}|M)$ denotes the variability of the encoding process. The second process is the recognition process that decodes the handwriting trajectory into the original intent. For each time step $1 \le t \le T$, the process maps a trajectory $W_{1:t}$ to a distribution over $\mathcal{E}$, denoted by $\mathcal{Q}_t$.

Let T and $\mathcal{Q}_{\text{final}}$ denote the time and the posterior distribution when the user finishes writing the trajectory respectively. According to the theory of channel capacity, the information transmitted through the channel is quantified by the mutual information between the input M and the decoding posterior $\mathcal{Q}_{\text{final}}$, denoted by $I(M; \mathcal{Q}_{\text{final}})$. We define the mean posterior of $\mathcal{Q}_{\text{final}}$ conditioned on M and the average posterior distribution as follows.

$$P(\mathcal{Q}_{\text{final}}|M) = \int\limits_{\overline{W} \sim P(\overline{W}|M)} P(\mathcal{Q}_{\text{final}}|\overline{W}) P(\overline{W}|M)$$

$$P(\mathcal{Q}_{\text{final}}) = \sum_{m \in \mathcal{E}} P(M = m) P(\mathcal{Q}_{\text{final}}|M = m)$$

Given these two expressions, we can define the mutual information between the character M and the decoding $\mathcal{Q}_{\text{final}}$ to be

$$I(M; \mathcal{Q}_{\text{final}}) = H(\mathcal{Q}_{\text{final}}) - \sum_{m \in \mathcal{E}} P(M = m) H(\mathcal{Q}_{\text{final}}|M = m)$$

where the entropy of $\mathcal{Q}_{\text{final}}$ is defined as

$$H(\mathcal{Q}_{\text{final}}) = -\sum_{m \in \mathcal{E}} P(\mathcal{Q}_{\text{final}} = m) \log_2 P(\mathcal{Q}_{\text{final}} = m)$$

Now we define the channel rate to be

$$R_{\text{MI}} = \frac{I(M; \mathcal{Q}_{\text{final}})}{\mathbb{E}[T]} \tag{1}$$

The channel rate R_{MI} is not suitable for practical implementation for two reasons. First, the estimates of R_{MI} relies on the estimates of $H(\mathcal{Q}_{\text{final}}|M)$ which typically require an extensive amount of data. Second, suppose the original intent is m, R_{MI} yields a high value as long as $P(\mathcal{Q}_{\text{final}}|M = m)$ concentrates on some intent n regardless of whether $n = m$. We propose an alternative to the R_{MI}, called R_{LL}, based on the idea of log loss. We define R_{LL} to be

$$R_{\text{LL}} = \frac{H(\mathcal{Q}_{\text{final}}) - \sum_{m \in \mathcal{E}} P(M = m)(-\log_2 P(\mathcal{Q}_{\text{final}} = m|M = m))}{\mathbb{E}[T]} \tag{2}$$

LEMMA 1. *Let $P_{i,j}$ denote $P(\mathcal{Q}_{\text{final}} = i|M = j)$. For any intent $m \in \mathcal{E}$ and $a = -\log_2 P_{m,m}$,*

$$H(\mathcal{Q}_{\text{final}}|M = m) \le a + a^{-1} + \log_2(|\mathcal{E}| - 1)$$

PROOF. By definition:

$$H(\mathcal{Q}_{\text{final}}|M = m) = -\sum_{i \in \mathcal{E}} P_{i,m} \log P_{i,m}$$

Given the value of $P_{i,m}$, the entropy is maximized if the remaining probability is divided uniformly across the remaining $|\mathcal{E}| - 1$ values. Let p denote $P_{m,m}$.

$$H(\mathcal{Q}_{\text{final}}|M = m) \le -p \log_2 p - (1 - p) \log_2 \left(\frac{1 - p}{|\mathcal{E}| - 1} \right)$$

We can rewrite this expression as

$$H(\mathcal{Q}_{\text{final}}|M = m) \le -p \log_2 p - (1-p) \log_2 (1-p) + (1-p) \log_2 (|\mathcal{E}| - 1)$$

Using the definition $a = -\log_2 p$, we have $p = 2^{-a}$. We can bound each term on the right hand side as follows.

$$H(\mathcal{Q}_{\text{final}}|M = m) \leq a + a^{-1} + \log_2(|\mathcal{E}| - 1)$$

□

From Lemma 1, it follows that when $(-\log_2 P(\mathcal{Q}_{\text{final}} = m|M = m))$ is small then the conditional entropy $H(\mathcal{Q}_{\text{final}}|M)$ is also small. As a result, the mutual information $I(M; \mathcal{Q}_{\text{final}})$ will be close to its maximal possible value of $H(\mathcal{Q}_{\text{final}})$. In other words, the log loss $(-\log_2 P(\mathcal{Q}_{\text{final}} = m|M = m))$ provides an upper bound for the conditional entropy $H(\mathcal{Q}_{\text{final}}|M)$. For the remaining of this paper, we refer to R_{LL} as *the channel rate*.

Intuitively, the channel rate is a measure that quantifies both accuracy and speed of a handwriting recognition channel at the same time. Handwriting, as well as many other motor control tasks, obeys the speed-accuracy tradeoff [3]. It is not sufficient to quantify the efficiency of a handwriting recognition system by its recognition accuracy alone. For example, a system that requires the user to write each character in a specialized form may attain a very high recognition accuracy, but it would require the user more time and effort to use. Such system might not be as efficient as a system that makes more errors but allows the user to write freely. In a sense, maximizing the channel rate is equivalent to finding a balance between maximizing the recognition accuracy and minimizing the writing time and effort of the user.

Based on this framework, we suggest that the channel rate can be improved by a combination of human learning and machine learning, which corresponds to improving the handwriting process and the recognition process respectively. Ideally, $\mathcal{Q}_{\text{final}}$ is always concentrated on the original intent M. This would mean that the channel is perfect and works without error. However, in real-world scenarios, errors will occur. One source of errors comes from misrecognition in the recognition process. These recognition errors can be reduced using training data and machine learning. The harder problem is when there is a significant overlap between $P(\overline{W}|M)$ for different intents. In this situation, we need to rely on the user to make their handwriting less ambiguous. Although the effect of human learning is always present, we believe that it can be enhanced by giving useful feedback to the user in the form of guidance or lessons.

ADAPTIVE RECOGNITION ALGORITHM

We developed an adaptive handwriting recognition algorithm that maps a handwriting trajectory $W_{1:t}$ to a posterior distribution over $\mathcal{E}$, denoted by $\mathcal{Q}_{\text{final}}$. By realizing that the effect of user adaptation is likely to be present, we designed our recognition algorithm so that it can adapt not only to each individual user, but also to the changes of the handwriting trajectory distribution $P(\overline{W}|M)$ unique to each user over time. The idea of specializing and adapting the recognizer for each user has been studied and shown to be effective in reducing the error rate [2, 9, 6].

At a high-level, our adaptive recognition system can be outlined as follows. For each user, the system creates and maintains one or more Markov-based models [13] for each element in $\mathcal{E}$. We refer to each of such models as a *prototype*. Each prototype is a left-to-right HMM with Gaussian observations with each of the hidden states corresponds to a sample point in the trajectory. Let $\mathcal{P}_u$ denote a set of prototypes for a user u. The adaptivity of our system comes from the retraining of $\mathcal{P}_u$. In decoding, given a handwriting trajectory and a set of prototypes $\mathcal{P}_u$, the system computes a posterior distribution $\mathcal{Q}_{\text{final}}$ and, when a single prediction is needed, the element with the maximum likelihood is predicted.

Feature vectors and distance function

As a preprocessing step, each instance of the handwriting trajectory is appended with the supplemental information. After preprocessing, a handwriting instance is described by a sequence of feature vectors $\langle f_1, \ldots, f_T \rangle$ where $f_i = (x_i, y_i, dx_i, dy_i)$. (x_i, y_i) denotes the normalized touchscreen coordinate and $(dx_i, dy_i) = (\frac{x_i - x_{i-1}}{z}, \frac{y_i - y_{i-1}}{z}), z = \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}$ denotes the writing direction.

For measuring the similarity between two handwriting instances or a handwriting instance and a prototype, we use *dynamic time warping* (DTW) distance [11] as the distance function in our algorithm. The DTW distance is commonly used for variable-length data such as handwriting and speech.

Initial adaptation

The initial adaptation is critical for any intelligent system. It is unquestionable that the performance of any well-behaved intelligent system increases as the system learns more about the user. Many times, the users can get frustrated with the system and stop using it before it can fully adapt to the user.

We address the problem of initial adaptation by leveraging the power of *the crowd*. Typically, people do have similar handwriting particularly when they share the same educational culture. In the very first interaction with the user u, our system has no information about the user and, therefore, assign a set of typical prototypes which has been trained using data from multiple users in the past. We refer to this set of prototypes as $\mathcal{P}_0$. After the first interaction, the system selects a set of prototypes $\mathcal{P}_{(u,1)}$ from the database of all prototypes across different users such that each prototype in the set minimizes the distance to the corresponding handwriting instances.

Adapting the prototypes over time

After the initial adaptation phase and after the user has given a few examples of his/her handwriting, the system performs a weighted cluster analysis using K-means on the data and the current prototypes in $\mathcal{P}_{(u,i)}$ to generate a new set of prototypes $\mathcal{P}_{(u,i+1)}$ such that the new set of prototypes minimizes the average inter-example distances of the same label.

Sometimes, a trained prototype has too many hidden states than necessary. The system performs an additional step to reduce the number of hidden states by a series of removing and merging hidden states while maintaining the same recognition power, using a variant of forward-backward algorithm. Figure 2 shows the hidden states before and after the reduction step.

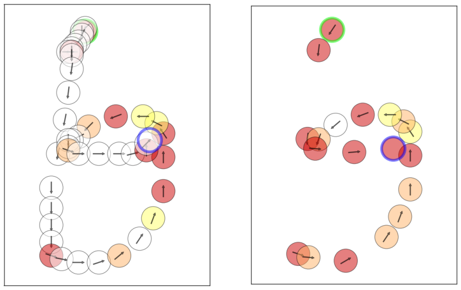


Figure 2: The hidden state reduction process is applied to each prototype to remove rarely visited states with respect to the training set. The originally trained prototype is shown on the left and the reduced prototype is shown on the right. The intensity of the colors corresponds to the expected number of times the state being mapped to.

Decoding

Our decoding algorithm is based on the standard Bayesian inference. Given a handwriting trajectory $W_{1:T}$ and the current set of prototypes $\mathcal{P}_u$, at every time step $1 \leq t \leq T$, the algorithm computes the distance from $W_{1:t}$ to each of the prototypes in $\mathcal{P}_u$. The distances are transformed into a probability distribution $\mathcal{Q}_t$. This decoding process is implemented using a dynamic programming technique which is similar in spirit to the forward algorithm [1]. When a single prediction is expected, the algorithm returns the prediction with the maximum likelihood.

EXPERIMENT

The main objective of our experiment is to determine and quantify the effect of machine adaptation and of human adaptation when the users interact with the system over some period of time. We conducted an experiment in the format of a game where the participants were asked to compete in a writing game. In each session, each participant was presented with a random permutation of the 26 lowercase English alphabets i.e. $\mathcal{E} = [a \ldots z]$ and $P(M)$ is uniform. The objective of the game was to write the presented characters as quickly as possible and, more importantly, the handwritten characters should be recognizable by the system. A score, which is the average *channel rate* of the session, was given to the user right after each session to reflect the performance of the session. There were 15 participants in this experiment. We did not control past experience of the participants. Some of them had more experience with touch screens than others. Each participant was asked to play the writing game, which is an implementation our adaptive recognition algorithm on the Apple iPads and iPhones, for at least 20 sessions over multiple days in his/her own pace.

The experiment was set up to demonstrate a condition called *co-adaptation* where both the user and the computer were allowed to adapt together. We denote this condition R_{adapt}.

To investigate the effect of co-adaptation, we create a controlled condition called R_{fixed} where the computer was not allowed to adapt with the user. In other words, we ran a simulation to figure out what the channel rates would have been if the prototype sets were never changed from $\mathcal{P}_0$. Ideally, it would be more preferable to have R_{fixed} determined by another control group where the prototypes were kept fixed and never changed. Unfortunately, we found that hard to do since the experiment was done on volunteers. However, the results from the simulated condition can be seen as a lower bound on the amount of the improvement attributable to human learning and, therefore, it is sufficient to demonstrate our point.

RESULTS AND DISCUSSION

The average channel rates per session of the two conditions R_{adapt} and R_{fixed} are shown in Figure 3a and Figure 3b respectively.

Although the prototype set was not changing in R_{fixed}, we observe that channel rate increases over the sessions. The paired t-test indicates a significant difference between the average channel rate in the first 5 sessions and the average channel rate in the last 5 sessions ($p < 0.0011$). This suggests that the users improve the handwriting on their own. We call this effect *user adaptation*.

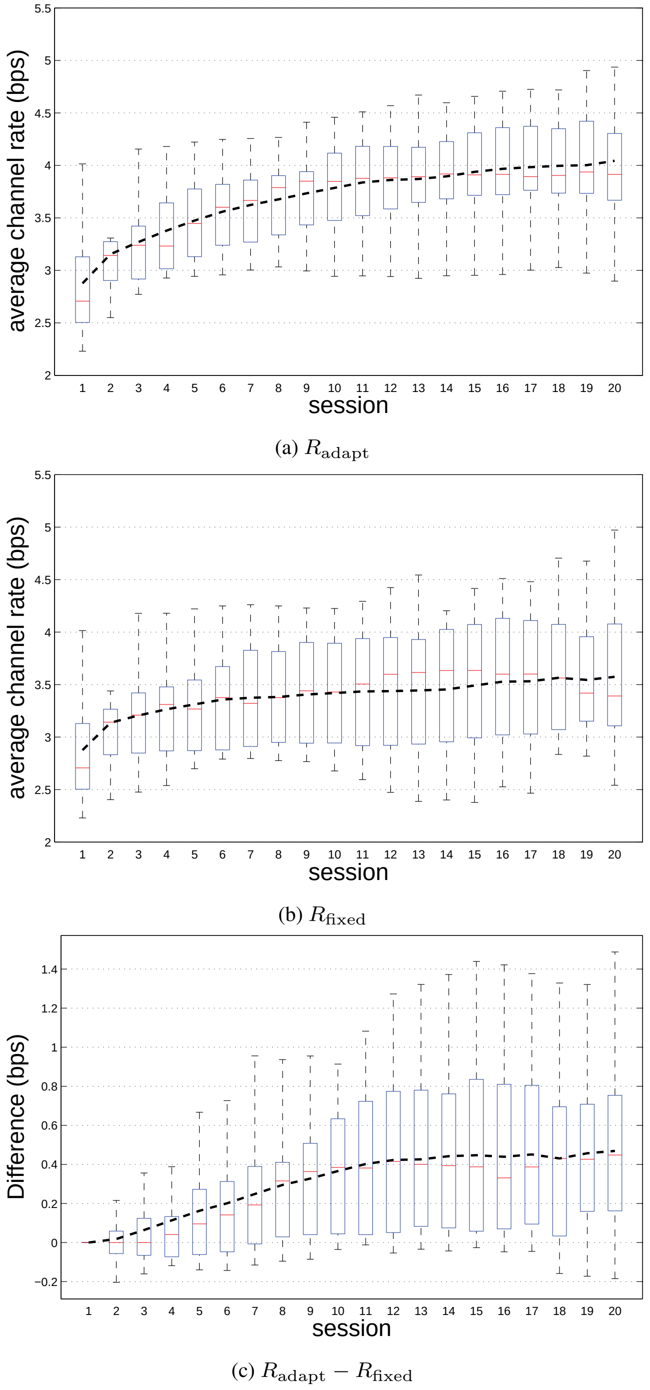
Figure 4a and Figure 4b reveal that the major contribution of *user adaptation* comes from the fact that the users write faster in the last 5 sessions compared to the first 5 sessions ($p < 0.0001$), and not because of the system received more information from the user ($p = 0.9723$). This result is as expected according to the law of practice [10].

In Figure 5, we compare R_{adapt} and R_{fixed} for each user. We found that the channel rate of R_{adapt} is significantly higher than that of R_{fixed} with $p < 0.0006$. This result confirms that the computer adaptation helps improving the overall channel rate. In addition, we calculate the theoretical maximum of the channel rate under the assumption of the perfect recognition, denoted by R_{ideal}. The maximum rates are given by $H(\mathcal{Q}_{\text{final}})/\mathbb{E}[T]$ and we approximated $H(\mathcal{Q}_{\text{final}}) = \log_2(26)$.

In the case of perfect recognition, a simple way to increase the channel rate is to increase the size of the character set $\mathcal{E}$. However, in reality, doing so can lead to a recognition error rate which impairs the channel rate. An interesting future direction is to find a character set that would maximize the channel rate. Figure 6 reveals the efficiency of each letter for our handwriting channel. Characters with complex stokes, such as 'q', 'g','k', are not as efficient as characters with simple strokes such as 'c' ,'o', 'l'.

Confusion and the conditional entropy

In addition to the experiment, we performed a detailed analysis on the recognition errors made by the system to visualize and understand about the mistakes. Specifically, we computed a confusion matrix based on the data from the experiment. The confusion matrix indicated that 99% of the mistakes concentrate among 33 pairs of prototypes out of the total of 2278 pairs. This suggests that the confusions only happen



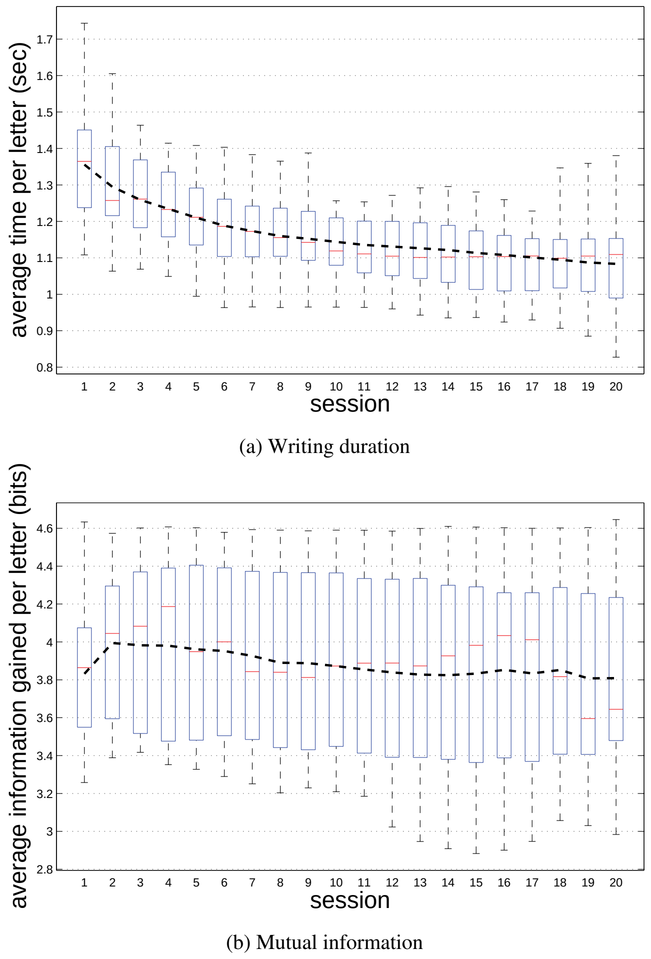
(a) R_{adapt}

(b) R_{fixed}

(c) $R_{\text{adapt}} - R_{\text{fixed}}$

Figure 3: Channel rate per session of each user with (3a) and without (3b) presence of machine learning.

between a few pairs of prototypes. Figure 7 shows some of the confusion pairs and the handwritten examples that were misrecognized. By inspection, we found that the confused handwritten characters were very similar for some letter pairs such as ’n’-’u’, ’n’-’h’ or ’r’-’v’.



(a) Writing duration

(b) Mutual information

Figure 4: The average writing time per session and the average mutual information per session under the condition R_{fixed}.

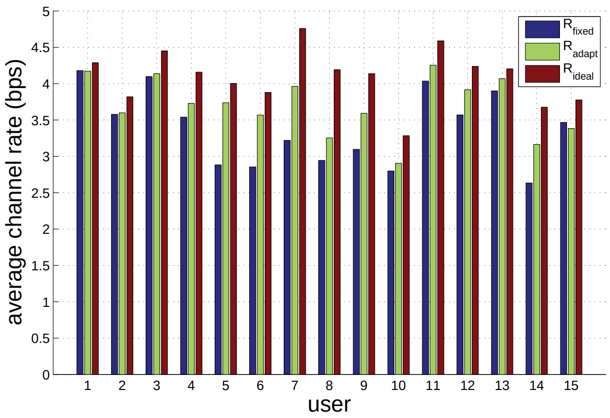


Figure 5: The average channel rate of each user in R_{adapt} and R_{fixed}. R_{ideal} shows the maximum channel rate possible given the average writing speed of each user.

The confusion is closely related to the conditional entropy $H(\mathcal{Q}_{\text{final}}|M)$. When this is no confusion, the entropy quickly converges to zero as demonstrated in Figure 8a. This suggests

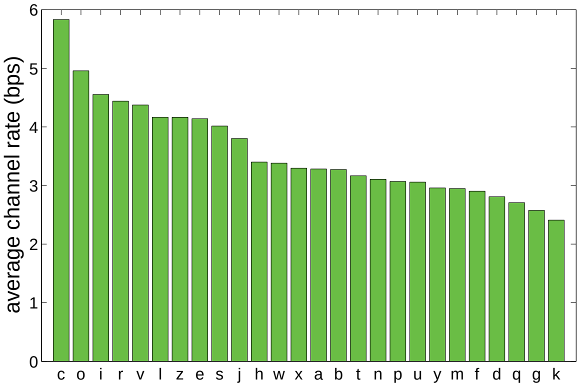


Figure 6: Average channel rate of each character under the condition R_{adapt}.

that early termination of the writing is viable. The system could have notified the user to stop writing at $\boxed{2}$ and it can still recognize the partial handwriting as a 'z'. On the other hand, when there is a confusion, the entropy does not necessarily converge to zero when at the end of the writing e.g. the entropy of 'y' in Figure 9c.

In Figure 9, we look closely at the evolution of $\mathcal{Q}_t$ of a confusable triplet: 'g', 'y' and 'q'. In Figure 9a, the probability of 'g' starts to dominate other contenders e.g. 's' and 'a' after $\boxed{3}$. Similarly, in Figure 9b, the posterior distribution evolves similarly to what we observe in Figure 9a then the probability of 'q' increases towards the end of the handwriting. This indicates that the crucial information that distinguishes between 'g' and 'q' is concentrated towards the end of the trajectory. Based on 7, the system sometimes confuses 'y' with 'g'. We suspect that such confusion happens when the probability of 'y' between $\boxed{1}$ and $\boxed{2}$ is too small relative to the probabilities of the contenders. The posterior distribution of a correctly recognized 'y' is shown in Figure 9c.

In Figure 8b, we show the posterior distributions over time of 3 examples selected from a single user: a correctly recognized 'n', a correctly recognized 'h' and an 'n' that was recognized as an 'h'. We notice that, when the system correctly recognized an 'n', the probability of 'n' increases significantly between $\boxed{2}$ and $\boxed{4}$, which corresponds to the upward movement of the hand when writing both 'n' and 'h'. This information can be delivered to the user in a form of the instructional feedback to encourage the user to pay more attention to the upward movement part when writing the pair.

CONCLUSIONS

In this paper, we presented a theoretical framework for quantifying the data transfer rate of a system that combines a human writer with a handwriting recognition system. We developed an adaptive handwriting recognition algorithm and showed the results from a small deployment of the system. From the results, we concluded that both the adaptation of the computer (machine learning) and of the human (learning to write) need to be considered together in order to design a system that maximizes the information rate. Finally, we performed a detailed analysis of the information transmission during the time that a single letter is written. Based on this analysis, we can pinpoint the location where the writer is failing to clearly disambiguate between two or more letters. On the other hand, we found other letters which are recognized shortly after the writing begins. These identify inefficiencies in the coding process and suggest ways we can guide the user to write in a way that would increase the overall channel rate.

REFERENCES

1. Bilmes, J. A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Tech. rep., ICSI, 1997.
2. Connell, S. D., and Jain, A. K. Writer adaptation for online handwriting recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on 24*, 3 (2002), 329–346.
3. Fitts, P. M. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology 47*, 6 (1954), 381–391.
4. Garrett, M., Ward, D., Murray, I., Cowans, P., and Mackay, D. Implementation of Dasher, an information efficient input mechanism. *Nature* (2003), 1–6.
5. Höök, K. Steps to take before intelligent user interfaces become real. *Interacting with computers 12* (2000), 409–426.
6. Kienzle, W., and Chellapilla, K. Personalized handwriting recognition via biased regularization. In *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, no. Section 6 (Pittsburgh, Pennsylvania, 2006), 457–464.
7. Lim, B. Y., and Dey, A. K. Assessing demand for intelligibility in context-aware applications. *Proceedings of the 11th international conference on Ubiquitous computing - Ubicomp '09* (2009), 195.
8. Maes, P. Agents that Reduce Work and Information Overload. *Communications of the ACM* (1994).
9. Matic, N., Guyon, I., Denker, J., and Vapnik, V. Writer adaptation for on-line handwritten character recognition. In *Proceedings of the Second International Conference on Document Analysis and Recognition (ICDAR '93)*, IEEE (1993), 187–191.
10. Newell, A., and Rosenbloom, P. S. Mechanisms of skill acquisition and the law of practice. In *Cognitive skills and their acquisition*, J. R. Anderson, Ed., vol. 6 of *Cognitive skills and their acquisition*. Erlbaum, 1981, ch. 1, 1–55.
11. Rabiner, L., and Juang, B.-H. *Fundamentals of Speech Recognition*, vol. 103 of *Prentice Hall signal processing series*. Prentice Hall, 1993.
12. Shannon, C. E. A Mathematical Theory of Communication. *Bell System Technical Journal 27*, July 1928 (1948), 379–423.

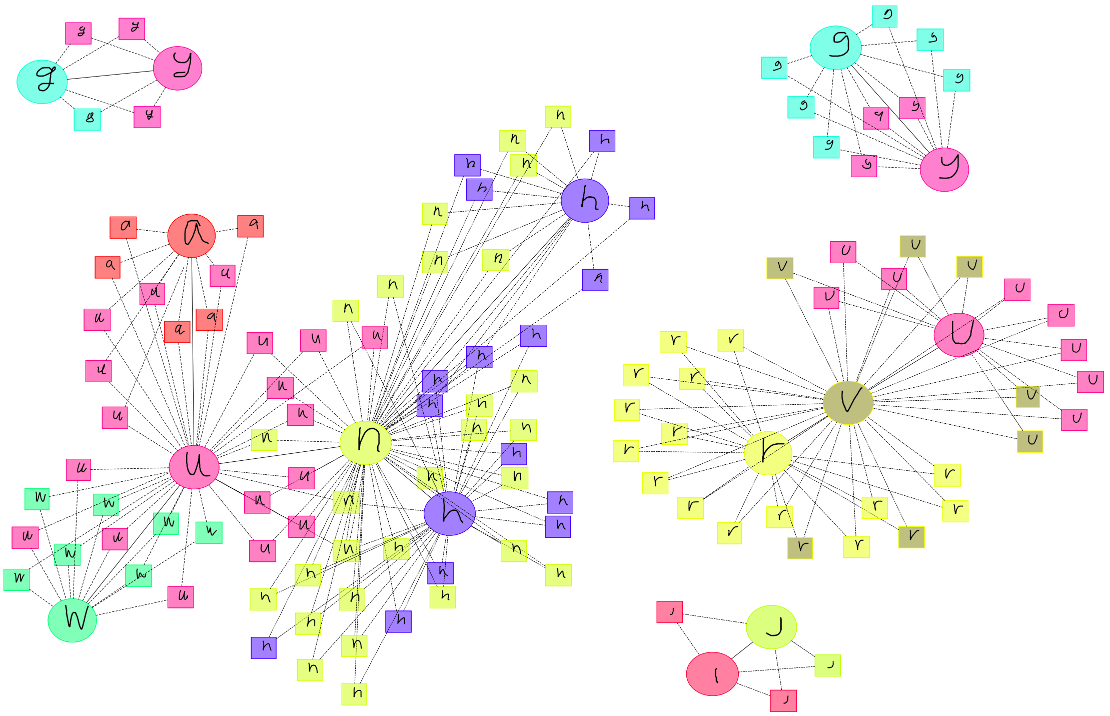
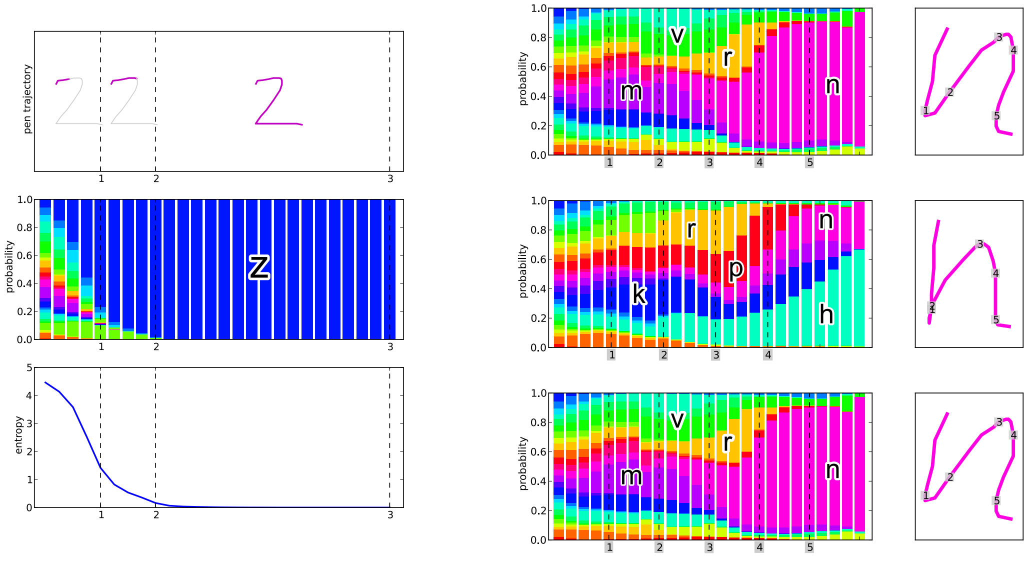


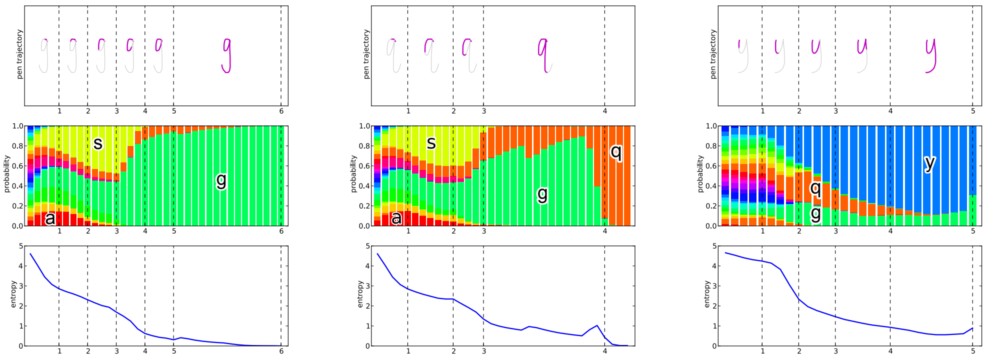
Figure 7: The confusion between two prototypes (circle nodes) is represented by an edge. Some of the confusing examples are shown in square nodes. Only confusable prototypes are shown.



(a)

(b)

Figure 8: The conditional entropy $H(\mathcal{Q}_{\text{final}}|M)$ quickly reduces to 0 when there is no confusion with other prototypes. — Three handwritten examples from a single user. The top example and the bottom example are recognized correctly as an ’n’ and an ’h’ respectively. The middle example is recognized as an ’h’ instead of the true label ’n’.



(a)

(b)

(c)

Figure 9: The posterior distributions as a function of time. The top row is the partial handwriting trajectories up to each dotted line. The middle row is the posterior distributions over time where each color corresponds to each label. The bottom row is the entropy over time.

13. Thomas Ploetz, and Gernot A Fink. *Markov Models for Handwriting Recognition*. SpringerBriefs in Computer Science. Springer, 2011.