A Novel HMM Decoding Algorithm Permitting Long-Term Dependencies and its Application to Handwritten Word Recognition

Volkmar Frinken, Ryosuke Kakisako, Seiichi Uchida
Faculty of Information Science and Electrical Engineering
Kyushu University, Japan
{vfrinken, uchida}@ait.kyushu-u.ac.jp, kakisako@human.ait.kyushu-u.ac.jp

***Abstract*—A new decoding for hidden Markov models is presented. As opposed to the commonly used Viterbi algorithm, it is based on the Min-Cut/Max-Flow algorithm instead of dynamic programming. Therefore non-Markovian long-term dependencies can easily be added to influence the decoding path while still finding the optimal decoding in polynomial time. We demonstrate through an experimental evaluation how these constraints can be used to improve an HMM-based handwritten word recognition system that model words via linear character-HMM by restricting the length of each character.**

I. Introduction

The automatic recognition of off-line handwritten words is, after years of intense research, still a hard problem [11], [17]. The large variety of different writing styles encountered in writer independent recognition tasks require sophisticated recognition systems trained on a large training set.

A very successful state-of-the-art approach is based of hidden Markov models (HMM) [4], [5], [8], [13]. HMM have the advantage of being well-understood statistical models with a clear mathematical background. The Markov property, which states (among other conditions) that current state of the model depends only on the current observation and the previous state, simplifies the modeling of complex pattern. In addition, computationally efficient algorithms for training and decoding exist.

However, the Markov property is also one of the biggest drawbacks of HMM-based handwriting recognition. As a matter of fact, handwriting is not a Markovian process as the writing path is influenced frequently by events in the past. For example, when drawing the character "0", it is important that the pen returns to a position close to the beginning of the stroke, to prevent any confusion with the digit '6'. Hence, the position of the final part of the stroke depends upon the initial position, clearly violating the Markov property. It is in principle possible to model long-term dependencies with suitable HMM topologies, but this comes at the cost of an increased model complexity.

In this work, we present an different way of adding non-Markovian constraints to HMM decoding. Instead of using the Viterbi algorithm for decoding, we propose to represent the decoding problem as a graph and employ the Min-Cut/Max-Flow algorithm to find an optimal path. This representation allows to easily constrain the decoding path through the addition of adequate edges into the graph. As a result, we derive a polynomial run-time algorithm for decoding linear HMM which respects long-term constraints. To the knowledge of the authors, this is the first time such an approach has been proposed. The presented algorithm is an extension of a previous publication in which the Min-Cut/Max-Flow algorithm is applied for dynamic time warping (DTW) [16]. The main contribution of the paper is the representation of the HMM decoding problem as an instance of a different problem class, viz. graph cut. Existing solutions for the new representation allow new constraints on the solutions space in polynomial time, something which was not possible before.

The rest of the article is outlined as follows. In the next section (Section II), underlying technologies are reviewed, in particular HMM and Min-Cut/Max-Flow algorithms. The proposed method of using Min-Cut/Max-Flow as a substitution of the Viterbi-decoding is explained in Section III. In Section IV, Non-Markovian constraints are introduced. An experimental evaluation is presented in Section V and conclusions are drawn in Section VI.

II. State of the Art

In this section, HMM-based word recognition is reviewed followed by an introduction of how the Min-Cut/Max-Flow algorithm can be used for DTW. These are the underlying methodologies upon which the proposed handwritten word recognition system is based.

An HMM is a model to explain observation sequences through a set of unobservable hidden states. At each time step, the state of the model is updated and the model emits an element in a stochastic process.

To fix the notation, we describe a HMM λ as a tuple $\lambda = (S, A, B, \Pi, F)$ containing a set of N states S, a matrix of state transition probabilities $A = \{a_{ij}\}$, a set of observation probability distribution functions B, a set of starting probabilities Π, and a set of final states $F \subset S$. Following [13], we describe the actual state at time t as q_t. Then $a_{ij} = P(q_t = s_j | q_{t-1} = s_i)$ is the state transition probability that the model changes from state s_i to s_j.

In order to find $\mathbf{S} = s_1 s_2 \cdots s_t$, the most likely sequence of hidden states, given the observation sequence $\mathbf{O} = o_1 o_2 \cdots o_T$

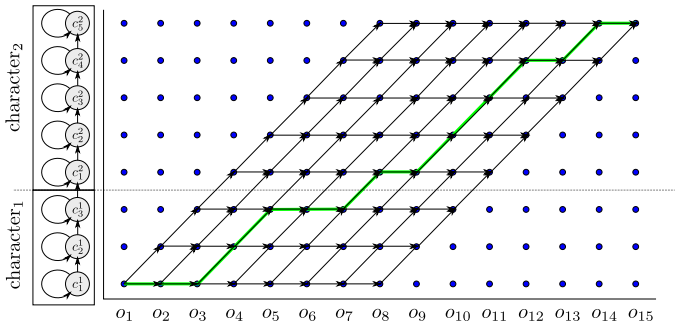


Fig. 1. The Viterbi algorithm matches an input sequence $\mathbf{o}$ to an HMM, while taking the state transition probabilities into account. In this example, the input is matched to a linear word-HMM as a composition of character-HMM. Allowed state transitions are indicated by arrows, a possible Viterbi path is highlighted in green.

of length T, we can use the Viterbi algorithm. It makes use of the Markov property, which states that the probability $P(q_t = s_j)$ of being in a specific state s_j at time t depends only on the observation probability $b_j(o_t)$ and the state q_{t-1}. Hence, it can be formulated as a dynamic programming (DP) algorithm:

$$\text{Initialization}: \quad (t = 1) \tag{1}$$

$$L(1, j) = \max_j \pi_j \cdot b_j(o_1) \tag{2}$$

$$\text{Iteration}: \quad (t = 2, \ldots, T) \tag{3}$$

$$L(t, j) = \max_i a_{ij} \cdot L(t-1, j) \cdot b_j(o_t) \tag{4}$$

where $L(t, j)$ is the likelihood to be in state j at time t. The final step of the algorithm is to find the highest value among the final states $\max_{j \in F} L(T, j)$.

To recognize a handwritten word using HMMs, we transform the word into a sequence of feature vectors and compare the likelihood given by the Viterbi algorithm for different HMMs, one for each possible word. The word whose HMM results in the highest likelihood is returned. However, training a distinct HMM for each word is usually unfeasible for large dictionaries. Thus a common approach is to explicitly model only single characters and to concatenate those character models in order to build words.

A common model topology is the the linear HMM approach in which states are aligned in a total ordering and only transition to the same state and the next state have non-zero probabilities. The advantage of this approach is a simple, and effectively trainable model, which still leads to a powerful recognition system. A word composed as a concatenation of linear character-HMM and the relation to DP is shown in Fig. 1. The input sequence $\mathbf{o}$ has to be mapped to the set of states, while the only allowed paths are shown by the arrows.

Min-Cut/Max-Flow algorithms find solutions to the maximum flow problems in a flow network containing a dedicated *source* and a *sink* node [6]. The solution to the maximum flow problem is simultaneously also the solution to the Min-Cut problem of splitting a graph with minimal costs into two parts leaving the sink node in one of the two parts and the source node in the other. The cost to be minimized is the sum of the weights of all edges that have to be removed. Min-Cut/Max-Flow can obtain a globally optimal solution efficiently and thus has been applied to various optimization problems [12] including matching tasks between 1D patterns [14], [15], 2D patterns [3] and 3D patterns [2].

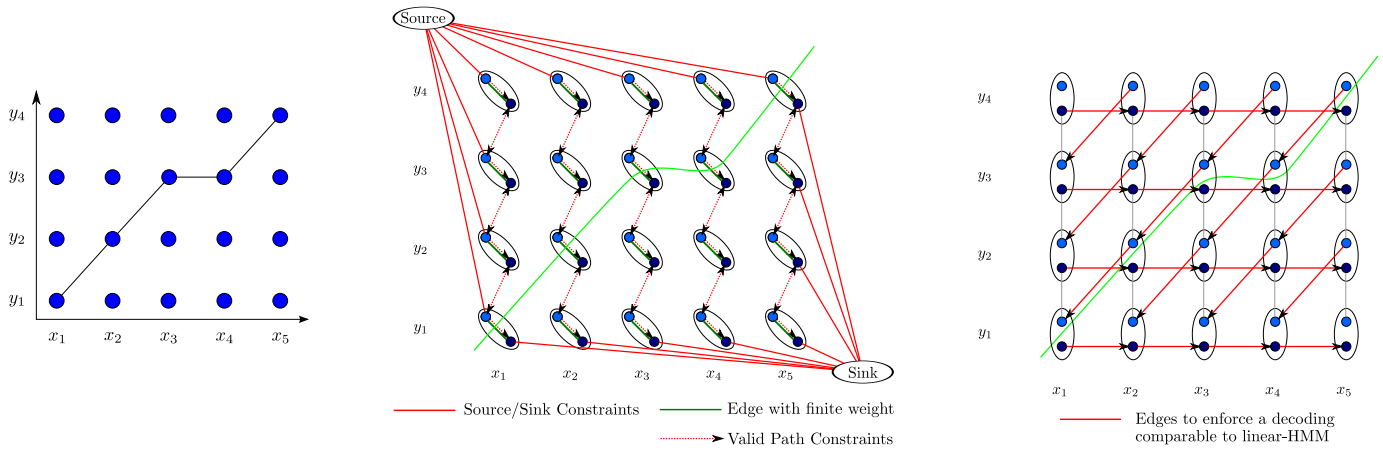
In [16], the use of Min-Cut/Max-Flow to perform DTW to align the two sequences. This is done by transforming the DP plane, spanned by $\mathbf{x} = x_1 x_2 \ldots$ and $\mathbf{y} = y_1 y_2 \ldots$, into a graph such that the minimal costs of a Min-Cut represents the best matching path. In short, this is done by creating two nodes (a top node and a bottom node) for each position (x_i, y_j), connected by an edge with weight $w = c(x_i, y_j)$, where $c(x_i, y_j)$ is the cost of matching x_i to y_j. Further edges with a weight $w = \infty$ are set to limit the path and implement the constraints, as shown in Fig. 2. The DP plane between the two sequences with a possible warping path is shown in Fig. 2(a). The goal of the representation of the DP plan as a graph (Fig. 2(b)) is to find splitting that separates the lower right half from the upper left half of the plane. The optimal solution corresponds to the DP path. In Fig. 2(c), a simplified version of the graph is shown that permits any path in which each x_i is assigned to exactly one y_j value. Furthermore, the path is restricted to be monotonous increasing with a slope of either 0 or 1. Note, that cuts respect edge directions, i.e., a directed edge with an infinite weight can be crossed from one direction but not the other.

III. Viterbi-Path Estimation using Min-Cut/Max-Flow

The first contribution of this paper is the representation of the HMM-decoding problem as a Max-Flow/Min-Cut problem. This is achieved my extending the above mentioned approach [16] to HMM decoding, i.e. aligning the observation sequence $\mathbf{o}$ to the sequence $\mathbf{s}$ of hidden states. However, not only the observation probabilities $b_j(o_i)$ (that o_i was emitted in state s_j) but also the state transition probabilities $a_{j,j'}$ of going from state s_j to state $s_{j'}$ have to be considered (cf. Eqn. 4). Consequently, for the Max-Flow/Min-Cut representation, the observation and transition costs have to be modeled through appropriate edges in the graph. Observation costs are included in the graph in a straight-forward way by setting the weight of the edge between the top and the bottom node at position (o_i, s_j) accordingly.

How dedicated edges in the graph can be used to account for state transition probabilities is best explained with the help of Fig. 3. The cut, after passing through position (o_i, s_j) can only continue through positions (o_{i+1}, s_j) or (o_{i+1}, s_{j+1}), reflecting the property of linear HMM where state the only permitted transitions from one state are back to the same state or the next state, as shown in Fig. 1.

One edge corresponding to a_{jj} is placed between the bottom node of position (o_i, s_j) and the top node of position (o_{i+1}, s_j) for every $i = 1, 2, \ldots, |\mathbf{o}| - 1$ (shown as the blue edge in Fig. 3). Similarly, an edge corresponding to $a_{j,j+1}$ is placed between the top node of position (o_i, s_j) and the top node



(a) The DP plane and the warping path on it.

(b) The warping path as a Min-Cut/Max-Flow problem.

(c) Simplified view and with added restrictions to the admissible DP path.

Fig. 2. Representing DP as a Max-Flow/Mic-Cut problem in which the two sequences $\mathbf{x}$ and $\mathbf{y}$ are to be aligned.

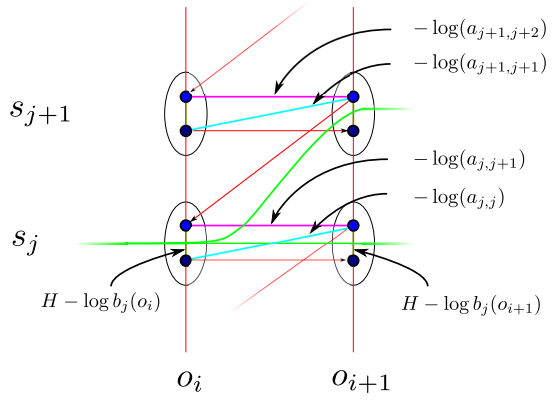


Fig. 3. A cut (green) going through the point (o_i, s_j) can only continue through the points (o_{i+1}, s_j) and adding $-log(a_{jj})$ to the cut costs (blue edge) or through (o_{i+1}, s_{j+1}) and adding $-log(a_{j,j+1})$ (purple edge).

of position (o_{i+1}, s_j) (shown as the purple edge in Fig. 3). In this setup, a cut going from (o_i, s_j) to (o_{i+1}, s_j) cuts the edge corresponding to a_{jj}, and a cut from (o_i, s_j) to (o_{i+1}, s_{j+1}) cuts the edge corresponding to $a_{j,j+1}$.

Edge costs are chosen to be the negative logarithm of the state transition probabilities and observation probabilities to transform the solution of the Viterbi algorithm – a path with the maximum product of probabilities – into a solution Min-Cut problem – a path with a minimum sum of edge costs.

Note, however, that the direct application of this rule can lead to negative edge costs, which poses a severe problem to the for the Min-Cut algorithm[1] [1]. This is because in continuous HMM, observation probabilities are represented as probability density functions $p : \mathbb{R}^n \to [0 : \infty)$ and $-\log(p)$ can not be guaranteed to be positive.

Therefore, we add a large constant H to the weights of all observation edges with

$$H = \log\left(\max \sum_{o \in \mathbf{o}, s \in S} p(o|s)\right)$$

to make all weights non-negative. The difference between the cost of the minimal cut and the Viterbi path is then $H \cdot T$, since the constant H is added once for each element of the input sequence $\mathbf{o} = o_1 o_2 \cdots o_T$. Finally, we subtract this constant from the cost of the cut to get the same result.

IV. Non-Markovian Constraints

The second contribution of this paper is the introduction of long-term constraints in the decoding of HMM. A big advantage of representing the problem as a Max-Flow/Min-Cut problem is the fact that we can add arbitrary edges into the graph and still be able to find an optimal path in polynomial run-time. This is used to restrict the space of possible paths by adding well-placed edges with infinite weight. As a consequence, we can add non-Markovian long-distance constraints to the HMM decoding, something that is not possible with the existing Viterbi decoding.

A. Basic Constraints

Two path limitations, the upper and lower path limit between two positions, can be considered as the fundamental constraints, which can be implemented with a single edge. In particular, given the constraint starting position (o_i, s_j) and an observation length t', a constraining edge between the bottom node of (o_i, s_j) and the bottom node of $(o_{i+t'}, s_{j+m})$ has the effect that any path which is in a state $s_{j'}$ with $j' \geq j$ at time $t = i$ has to be in a state s_k with $k \geq j + m$ after t' further time steps. Hence we can force a lower limit on the state index at time $t = i + t'$ conditioned upon q_i, the state

[1] In fact, allowing negative edge costs could be used to solve the Max-Cut problem, which is NP complete.

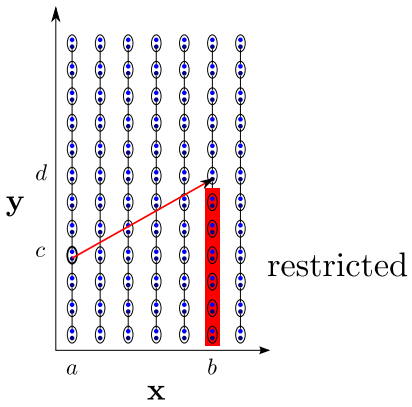


Fig. 4. A basic constraints on the path can be imposed with single edges, in this case a lower constraint. If the path goes through the indicated node it can not go through the nodes marked in red.

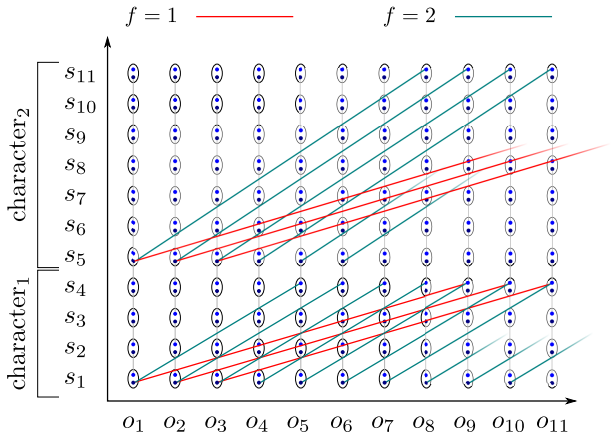


Fig. 5. Inserting appropriate edges can act as lower (or upper) limit of how many features are assigned to one character. Here, two different lower limits are shown. The lower limits for the character length limitation factor $f = 1$ in red and $f = 2$ in green.

at $t = i$. Similarly, we can impose upper limits by placing an edge between the top nodes of two positions.

B. Non-Markovian Constraints for HMM Word Recognition

The constraints implemented in the experimental evaluation are character duration limits in the context of handwritten word recognition. As opposed to a maximum state duration, which can be modeled fairly easily with an appropriate HMM topology, a limit on the time spent in one character HMM, without imposing any further constraints on how much time is spent in each state of this HMM, is far more complex. As a Max-Flow/Min-Cut problem, however, it is sufficient to insert edges between the first state and the last state of each character and the maximum permitted time, as shown in Fig. 5.

Now consider the lower limits defined by edges between nodes (o_i, s_j) and $(o_{i+f \cdot n_c}, s_{j+n_c})$ where f is a character length limitation factor, c is a character of the word to be recognized, j is the position of the first state in the word-HMM and n_c is the number of states in the character-HMM of c. Imposing these restriction for every $i = 1, 2, \ldots, T - n_c$ ensures that in the decoding, not more than $f \cdot n_c$ observations are assigned to character c, which effectively limits the maximum length of character c to $f \cdot n_c$.

HMM state duration control is only one effect that can be realized easily with this approach. Any smoothness-control of the path, i.e. a maximum and minimum derivation, not only within characters but in over the entire sequence, can as as easily be implemented as well.

V. Experimental Evaluation

By restricting the decoding path through non-Markovian constraints, we can alter the words recognized by the HMM. To show how this positively affects the recognition rate, we devised the following experiment. In short, a word in the test set is recognized several times with different constraints f. Each time a recognition hypothesis for a word w is returned, resulting in a set of hypotheses $\{(f_1, w_1), (f_2, w_2), \ldots, (f_n, w_n)\}$. From the validation set we can estimate for each constraint-word tuple (f_k, w_k) the recognition accuracy, which we consider as the confidence $c(f_k, w_k)$. Finally we chose the output with the highest confidence.

The imposed long-term constraints are defined by the parameter f, which indicates the maximum allowed character duration time in multiples of the number of hidden states. As an example, the character 'A' is defined by an HMM with 18 hidden states. Hence a character length limitation factor of $f = 5$ would indicate that no more than 90 feature vectors can be assigned to that character.

The used confidence function $c(f_k, w_k)$ is the recognition rate on the subset of the validation set $r_{val}(f, w)$ whose result is w when applying the limiting factor f, yet only if enough (more than τ) such samples exist:

$$c(f, w) = \begin{cases} r_{val}(f, w) & |w| > \tau, \\ f \cdot d & \text{otherwise.} \end{cases}$$

The values d and τ are free parameters to be optimized. The reason for setting the fall-back confidence as $f \cdot d$ is, that an unrestricted recognition works well in the general case. Hence, without a robust estimation on the impact, a less restricted decoding is to be preferred.

The exact recognition procedure is given in Fig. 6. The set of limiting factors was chosen as

$$F = \{2, 3, 4, 5, 6, 7, 8, 9, 10\},$$

and d and τ were validated on the validation set. The dictionary was chosen to contain all words occurring in the validation and test set, hence the task is closed vocabulary recognition.

A. Setup

Experiments are conducted using all instances of the 4 000 most frequent words of the IAM off-line database[2] [10]. All

[2]http://www.iam.unibe.ch/fki/databases/iam-handwriting-database

Require: A word image X; A dictionary of words W; A list of path constraints F; A confidence matrix $C(w, f)$
Ensure: A recognition hypothesis $R \in W$ is returned

1: **for all** $f \in F$ **do**
2: **for all** $w \in W$ **do**
3: compute $P(w, f|x)$ using Min-Cut/Max-Flow
4: **end for**
Consider the best word for each constraint:
5: $T[f] = \arg\max_w P(w, f|X)$
6: **end for**
Find the constraint leading to the highest confidence:
7: $f' = \arg\max_f C(T[f], f)$
Return the word recognized with this constraint:
8: **return** $T[f']$

Fig. 6. The recognition procedure

correctly segmented words among the 4 000 most frequent words according to the LOB corpus [7] are considered. The three sets, a working set (38 127 words), validation set (5 590 words) and training set (5 342 words) are writer disjunct, thus any person who contributed words to one of the three sets did not contribute to any of the other set.

The database itself consists of 1 539 pages of handwritten English text, written by 657 writers. All pages of the database are already segmented into individual text lines and words. The images are normalized prior to recognition in order to cope with different writing styles. First, the skew angle is determined by a regression analysis based on the bottom-most black pixel of each pixel column. Then, the skew of the text line is removed by rotation. Afterwards the slant is corrected in order to normalize the directions of long vertical strokes found in characters like 't' or 'l'. After estimating the slant angle based on a histogram analysis, a shear transformation is applied to the image. Next, a vertical scaling is applied to obtain three writing zones of the same height, i.e., lower, middle, and upper zone, separated by the lower and upper baseline. To determine the lower baseline, the regression result from skew correction is used, and the upper baseline is found by vertical histogram analysis. For more details on the normalization operations, we refer to [9].

The baseline system is a character-based, linear HMM, trained on the training set with a variable number of hidden states per character [18]. The output probability density function is estimated through a mixture of 14 Gaussian distributions.

B. Results

The overall performance of the baseline HMM system without any path restrictions is 74.28% while the proposed system performs slightly better at 74.94%. This increase might be only marginal, but behaves very differently for various word classes and restriction factors. In Fig. 7 the recognition accuracy of a few representative word classes and all tested restricting factors are shown.

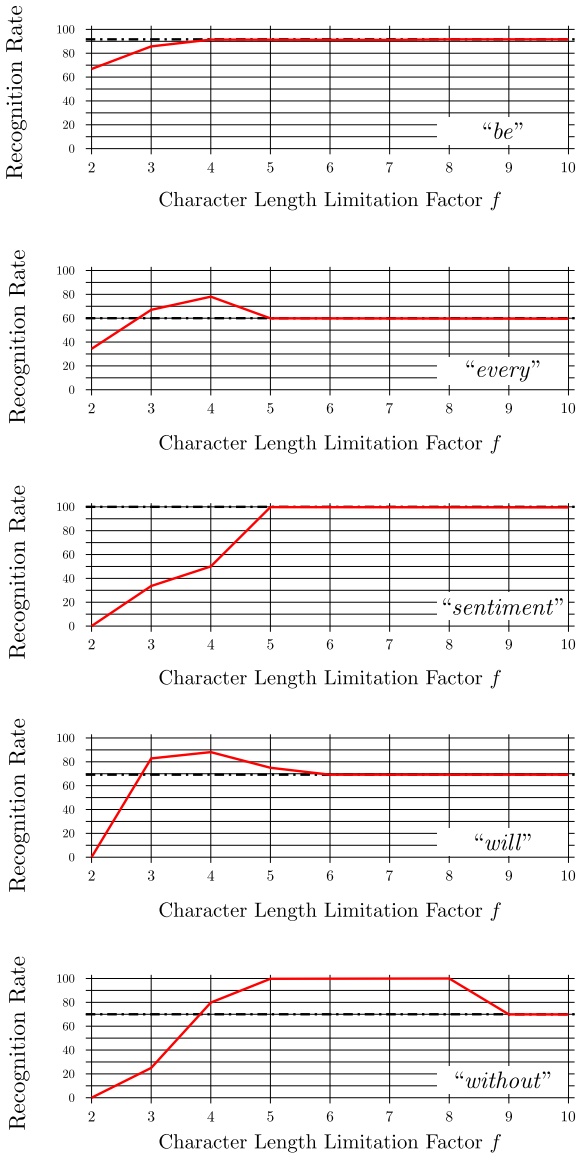


Fig. 7. The recognition rate of selected words as a function of the character length limitation factor f and the reference recognition rate (dotted lines).

For very small values of f, a massive degradation of the recognition accuracy can be observed. Small values of f signify tight character duration limits, hence the correct word might not be valid at all. For large value of f (10 or higher in these experiments), the duration limit does not have any effect. Hence, the recognition accuracy as a function of f starts therefore close to 0, increases, and levels off at the baseline value. For some of the words, the maximum recognition rate is significantly higher than the baseline.

Finally, in Fig. 8, a few input words are shown, along with the recognition output for several values of f. As can be seen from sample words, small values of f enforce the recognition of longer words, because a maximum character duration length implies a maximum word length. If the input image is too large, short words can not be recognized. On the other end of

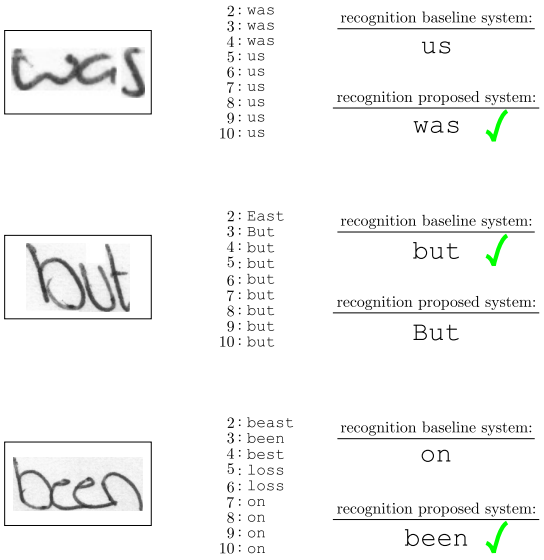


Fig. 8. Three samples from the test set. Next to each word, the recognition output for each character length limitation factor f is given and on the right side the final output of the proposed system and the baseline system.

the value range for f (relaxed constrains), the output is the same as the baseline's output.

VI. Conclusion

In this paper we present a novel HMM decoding algorithm for linear HMM topologies. Through the representation of the feature-to-hidden-state-assignment problem as a graph, we can use the Max-Flow/Min-Cut algorithm to find an optimal path, which is equivalent to the Viterbi path. However, as a fundamental improvement over the Viterbi-algorithm, this approach allows to easily set long-term constraints which restrict the space of admissible paths. While DP is not suited to solve the newly posed problem, Max-Flow/Min-Cut can find the optimal solution in polynomial time.

The application of the novel decoding technique including long-term constraints to handwriting recognition is demonstrated through experiments on the IAM off-line handwriting database. Carefully placed decoding constraints are able to limit the minimum and maximum character length. This can be used to not only restrict the possible words, but also to prevent pathological Viterbi-path with an unbalanced assignment in the number of feature vectors that are assigned to each character of a word. The experimental evaluation shows that it is possible to exploit this effect to improve the recognition rate for handwritten word recognition.

Currently a mismatch between the training and decoding algorithm can be observed, since the training is done using standard Baum-Welch algorithm. Obviously the inclusion of constrains in the training process, e.g. via *Viterbi Training*, is worth investigating, as well as methods to learn the long term constraints automatically. Additionally we will investigate in the future more sophisticated path restrictions, such as penalties instead of hard constraints and we will further extend the idea to more general HMM topologies, beyond the linear model. Additionally, further applications like keyword spotting, are also along the possible line of research.

Acknowledgment

This work is supported in part by the CREST project from Japan Society for the Promotion of Science (JSPS).

References

[1] Yuri Boykov and Vladimir Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[2] Will Chang and Matthias Zwicker. Automatic registration for articulated shapes. *Computer Graphics Forum*, 27(5):1459–1468, 2008.

[3] Olivier Duchenne, Armand Joulin, and Jean Ponce. A Graph-Matching Kernel for Object Categorization. In *Int'l Conf. on Computer Vision*, pages 1792–1799, 2011.

[4] Salvador España-Boquera, Maria José Castro-Bleda, Jorge Gorbe-Moya, and Francisco Zamora-Martinez. Improving Offline Handwritten Text Recognition with Hybrid HMM/ANN Models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33(4):767–779, 2011.

[5] Volkmar Frinken, Tim Peter, Andreas Fischer, Horst Bunke, Trinh-Minh-Tri Do, and Thierry Artieres. Improved Handwriting Recognition by Combining Two Forms of Hidden Markov Models and a Recurrent Neural Network. In *13th Int'l Conf. on Computer Analysis of Images and Patterns*, volume 5702/2009 of *Lecture Notes in Computer Science*, pages 189–196, 2009.

[6] T.E. Harris and F.S. Ross. Fundamentals of a Method for Evaluating Rail Network Capacities. Research Memorandum RM-1573, The RAND Corporation, Santa Monica, CA, USA, 1955.

[7] Stig Johanson, Geoffrey N. Leech, and Helen Goodluck. Manual of Information to Accompany the Lancaster-Oslo/Bergen Corpus of British English, for use with Digital Computers. Technical report, Department of English, University of Oslo, Norway, 1978.

[8] Michal Kozielski, Patrick Doetsch, and Hermann Ney. Improvements in RWTH's System for Off-Line Handwriting Recognition. In *Int'l Conf. on Document Analysis and Recognition*, pages 935–939, 2013.

[9] Urs-Victor Marti and Horst Bunke. Using a Statistical Language Model to Improve the Performance of an HMM-Based Cursive Handwriting Recognition System. *Int'l Journal of Pattern Recognition and Artificial Intelligence*, 15:65–90, 2001.

[10] Urs-Victor Marti and Horst Bunke. The IAM-Database: An English Sentence Database for Offline Handwriting Recognition. *Int'l Journal on Document Analysis and Recognition*, 5:39–46, 2002.

[11] Réjean Plamondon and Sargur N. Srihari. On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 22(1):63–84, 2000.

[12] Brian L. Price, Bryan S. Morse, and Scott Cohen. Geodesic Graph Cut for Interactive Image Segmentation. In *Int'l Conf. on Computer Vision and Pattern Recognition*, pages 3161–3168, 2010.

[13] Lawrence Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[14] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient Planar Graph Cuts with Applications in Computer Vision. In *Int'l Conf. on Computer Vision and Pattern Recognition*, pages 351–356, 2009.

[15] Frank R. Schmidt, Eno Töppe, Daniel Cremers, and Yuri Boykov. Efficient Shape Matching via Graph Cuts. In *Int'l Conf. on Energy Minimization Methods in Computer Vision and Pattern Recognition*, pages 39–45, 2007.

[16] Seiichi Uchida, Masahiro Fukutomi, Koichi Ogawara, and Yaokai Feng. Non-markovian Dynamic Time Warping. In *21st Int'l Conf. on Pattern Recognition*, pages 2294–2297, 2012.

[17] Alessandro Vinciarelli. A Survey On Off-Line Cursive Word Recognition. *Pattern Recognition*, 35(7):1433–1446, 2002.

[18] Matthias Zimmermann and Horst Bunke. Hidden Markov Model Length Optimization for Handwriting Recognition systems. In *Proc. 8th int. workshop on frontiers in handwriting recognition*, pages 369–374, 2002.