Bul. Ştiinţ. Univ. Baia Mare, Ser. B,
Matematică-Informatică, Vol. XV(1999), Nr. 1-2, 117-126

Dedicated to Professor Ion PĂVĂLOIU on his 60th anniversary

ITERATIVE DYNAMIC PROGRAMMING -A NEW SOLUTION FOR OPTIMIZATION PROBLEMS

Marius-Călin SILAGHI, Vasile BERINDE

Abstract

The Dynamic Programming (DP) is an old strategy that has been formalized for the first time in early fifties [2]. Since then, many applications of this generic method are constantly developed [6], so that it can be found in very different forms. There is a well-known form of DP which is largely used for computations with graphs [7]. Within this framework, DP highly optimize the computation of paths in graphs that maximize/minimize certain score functions. However, some scoring functions prove to be too complicated in order to be computed with only one DP run, especially when the score of one path depends on the score of other paths that are not related. These functions are specially useful in engineering and signal processing, and the exact optimization using DP proves to be expensive. Approximation techniques, even if often successful, are less robust. Here we present a solution to the problem of finding paths optimizing normalized costs in Hidden Markov Models. Our solution is based on a computation implying a simple and cheap DP run at each step. The solution is signaled by the convergence of a sequence generated by these steps. We prove that the sequence converges always. The quick convergence obtained in practical applications shows that the algorithm can be much more efficient that the straightforward computations with DP.

1 Introduction

Dynamic Programming (DP) is a generic method that was first formulated by a mathematician called Richard Bellman in [2]. The method has many instantiations for different types of problems and is the subject of many books [3, 6].

One form of Dynamic Programming [7] allows the computation of an optimal path in certain graphs, between predefined sets of nodes. This method may be applied as long as the cost function corresponds to the Bellman's principle of optimality [5]:

Definition 1 (Optimality Principle) *A function $f(C)$ defined on the domain of the paths in a graph corresponds to the optimality principle iff, $\forall a, b$, two nodes in the graph and if C is the path between them that maximizes the cost function $f(C)$, $C = \{a, x_1, x_2, ..., x_n, b\}$, then the paths $C_1 = \{a, x_1, ..., x_i\}$ and $C_2 = \{x_i, ..., x_n, b\}$ maximize the function f for all the paths between a and x_i respectively x_i and b, $\forall i \in \overline{1..n}$*

The method of Dynamic Programming can find between whatever nodes of the graph, the path that optimizes the function f. DP does this by computing iteratively the optimal path known at each node a toward each node b and visiting any of the neighbor nodes of a at which a path to b was computed in previous steps.

For example let us imagine a net of roads that can be represented with a matrix where we have a town in each element of the matrix. The entry in each town requires the payment of a certain toll and the road between two towns has a known cost as well. The access is possible only along the rows and the columns. Let us use DP to compute for this case the optimal path between the town in the left-down corner and the one in the right-up corner. If the roads are allowed only from left to right and upward, in that case the computation can be realized in only one iteration over all rows and columns [7].

This solution has an application in the computation of an optimal path through a Hidden Markov Model (HMM) [1] corresponding to a vector of observations X. The last problem may be represented with a matrix where each element of the observation vector X corresponds to a column and each of the states of the HMM M corresponds to a row. The matrix is filled with the probability of the corresponding tuples. The path in M that fits to X with maximal probability can be obtained by DP between the corner of the matrix corresponding to the first element of X and first state of M and the corner corresponding to the last observation, respectively the last state.

Often we search for the path in M and the sub-sequence of X that correspond optimally from all the possibilities. The score is provided by the average emission probability along the best match. The exact choice may be done by computing the probability of the optimal path in M for each sub-sequence of X, that is for each acceptable pair of beginning and end points in X. But this would require a high number of runs of the DP algorithm. We present an algorithm for the easier exact computation of the highest probability and of the pair that corresponds to it.

2 Notations

If $X = \{x_1, ..., x_N\}$ and M has the states $Q = \{q_1, ..., q_L\}$, then we note with $\overline{M}$ the HMM that extends M with the state q_G, $\{q_{G_1}, q_1, ..., q_L, q_{G_2}\}$ and having the probability of transition 1 between q_G and q_1, q_L and q_G, q_{G_1} and q_{G_1} as well as between q_{G_2} and q_{G_2}. The emission probability in the states q_G (q_{G_1} and q_{G_2}) are the same for all observations and equal with a constant ε.

We note with X_a^b the subsequence $X_a^b = \{x_a, ..., x_b\}$ where $1 \leq a \leq b \leq N$. One approximates often the log posterior of a model M given a subsequence X_b^e with the average posterior probability along the optimal path:

$$\begin{aligned} -\log P(M|X_b^e) &\simeq \frac{1}{e-b+1} \min_{\forall Q \in M} -\log P(Q|X_b^e) \\ &\simeq \frac{1}{e-b+1} \min_{\forall Q \in M} \{-\log P(q^b|q_G) - \sum_{n=b}^{e-1} [\log P(q^n|x_n) \\ &\quad + \log P(q^{n+1}|q^n)] - \log P(q^e|x_e) - \log P(q_G|q^e)\} \end{aligned} \quad (1)$$

where we have noted with $Q = \{q^b, q^{b+1}, ..., q^e\}$ one of the possible paths of length $e - b + 1$ in M and the HMM state visited at time n along Q, with $q^n \in Q$. q_G is simply used here as the non-emitting initial and final state of M.

For a specific sub-sequence X_b^e, expression (1) can easily be estimated by dynamic programming since the sub-sequence and the associated normalizing factor $(e - b + 1)$ are given. However, we sometimes look for:

$$S(M|X) = \min_{\forall \{b,e\}} -\log P(M|X_b^e). \quad (2)$$

The optimal {begin/end}points $\{b^*, e^*\}$, and the associated optimal path Q^*, are then given by:

$$< Q^*, b^*, e^* >= \underset{\{Q,b,e\}}{\text{argmin}} \frac{1}{e-b+1} - \log P(Q|X_b^e) \quad (3)$$

$\overline{Q} = \{\overbrace{q_G, ...q_G}^{b-1}, q^b, q^{b+1}, ..., q^e, \overbrace{q_G, ...q_G}^{N-e}\}$ is the path that extends Q with $b - 1$ states q_G before q^b and $N - e$ states q_G following q^e.

With these notations, the described method of Dynamic Programming which looks for the optimal pair between a path $\overline{Q_\varepsilon}$ in $\overline{M}$ and X computes:

$$\begin{aligned}
\overline{Q_\varepsilon} &= \underset{\forall \overline{Q} \in \overline{M}}{\operatorname{argmin}} \; -\log P(\overline{Q}|X) \\
&= \underset{\forall Q \in M}{\operatorname{argmin}} \; \{-\sum_{n=1}^{b-1} \log P(q_G|x_n) - \log P(q^b|q_G) \\
&\qquad -\sum_{n=b}^{e-1} \left[\log P(q^n|x_n) + \log P(q^{n+1}|q^n)\right] - \log P(q^e|x_e) \\
&\qquad - \log P(q_G|q^e) - \sum_{n=e+1}^{N} \log P(q_G|x_n)\} \qquad (4)
\end{aligned}$$

3 Iterative Dynamic Programming

$\overline{Q^*}$ do not visit the state q_G between the observations b^* and e^* and stays only in this state otherwise. For shortening the notation in demonstrations we use:

$$w = S(M|X) = -\log P(M|X_{b^*}^{e^*}). \qquad (5)$$

Proposition 1 *A path $\overline{Q}$ returned by the DP algorithm for X and $\overline{M}$ is completely determined by the first and the last observations of its Q - noted b respectively e - for all computations (all values of ε).*

It means that for whatever two paths $\overline{Q_{\varepsilon_1}}$, $\overline{Q_{\varepsilon_2}}$, obtained with Equation 4 for two different values of ε, ε_1 and ε_2, if they have the same first and last observation of their Q_{ε_1} and Q_{ε_2}, $b_1 = b_2$ and $e_1 = e_2$, then $\overline{Q_1} = \overline{Q_2}$.

Only the end states have different emission probabilities for different computations, as they depend on ε. Between the observations with indices b and e there is no end state q_G. The minimal path is always the same between indices b and e, while it is computed with the same deterministic DP algorithm. While, from the definition of Q, all the exterior observations match end states q_G, the results of the computations that have the same b and e represent the same path. We can therefore note a path $\overline{Q}$ as Q_b^e in order to determine it. $\overline{Q_{b^*}^{e^*}}$ is identical with $\overline{Q^*}$.

Lemma 2 *First, we show that if $\varepsilon = w$, then the DP algorithm over X and $\overline{M}$ chooses the sequence $\overline{Q^*}$.*

Proof. We see that choosing $\overline{Q^*}$, $-\log P(\overline{Q^*}|X) = N * w$. Another path $\overline{Q}$ would have yielded $-\log P(\overline{Q}|X) = N*w+(e-b+1)*(w'-w) > N*w$ while $w' > w$ from the definition of Q^* (if unique). Here we have noted by b, e, w' the values of the first frame, last frame respectively average probability in whatever path $\overline{Q}$ where Q_b^e is an alternative to Q^*.

$$w' = -\log P(M|X_b^e) = \frac{1}{e-b+1} - \log P(\overline{Q}|X_b^e)$$

While DP chooses the optimum (minimum) it chooses $\overline{Q^*}$. □

We recall that:

Proposition 3 *$\forall \varepsilon$, if the resulting DP path over X and $\overline{M}$ is $\overline{Q_b^\varepsilon}$, then $w' = -\log P(M|X_b^\varepsilon) \geq w$.*

The indices of the first and last observations in Q are numbered b and e. From the definition of Q^*, the average probability on Q, $-\log P(M|X_b^e)$, is w' higher or equal with w.

(6) $$\forall \varepsilon \Rightarrow w' \geq w$$

Equality appears when the path Q^* is not unique.

Lemma 4 *If $\varepsilon > w$ and the resulting DP path is Q_b^ε, then the found sequence will try to profit more in the states of Q:*

(7) $$e^* - b^* \leq e - b$$

Proof.

$$-\log P(\overline{Q^*}|X)) = (b^* + N - 1 - e^*) * \varepsilon + (e^* - b^* + 1) * w$$

We demonstrate that the path Q^* is better than any shorter one Q_b^ε:

$$-\log P(\overline{Q_b^\varepsilon}|X) = (b + N - 1 - e) * \varepsilon + (e - b + 1) * w'$$

If we assume that Q_b^ε is chosen with

$$e^* - b^* > e - b.$$

$((e^* - b^*) - (e - b)) > 0$. $w' \geq w$ from the definition of Q^*. $\varepsilon > w$ from hypothesis. Therefore:

$$(e^* - b^* + 1) * w < ((e^* - b^*) - (e - b)) * \varepsilon + (e - b + 1) * w', \ \forall\ e^* - b^* > e - b.$$

We get:

$$(b^* + N - 1 - e^*) * \varepsilon + (e^* - b^* + 1) * w < (b + N - 1 - e) * \varepsilon + (e - b + 1) * w',$$

showing that such a path $\overline{Q}$ would not have been preferred to $\overline{Q^*}$.

$\Rightarrow$ The assumption was wrong and the chosen $\overline{Q}$ will have $e^* - b^* \leq e - b$.

□

Lemma 5 *If $\varepsilon > w$ and the resulting DP path over X and $\overline{M}$ is $\overline{Q_b^e}$, then $w' = -\log P(M|X_b^e) < \varepsilon$.*

Proof.

$$-\log P(\overline{Q_b^e}|X) = (b-1) * \varepsilon + (e - b + 1) * w' + (N - e) * \varepsilon$$

$$\Rightarrow -\log P(\overline{Q}|X) = (N - 1 - e + b) * \varepsilon + (e - b + 1) * w'.$$

If the $\overline{Q^*}$ sequence was not preferred then:

$$-\log P(\overline{Q}|X) \leq -\log P(\overline{Q^*}|X)$$

$$(N - 1 - e + b) * \varepsilon + (e - b + 1) * w' \leq (N - 1 - e^* + b^*) * \varepsilon + (e^* - b^* + 1) * w$$

$$(e - b + 1) * w' \leq (N - 1 - e^* + b^*) * \varepsilon + (e^* - b^* + 1) * w - (N - 1 - e + b) * \varepsilon$$

$$(8) \qquad (e - b + 1) * w' \leq (e^* - b^* + 1) * w + ((e - b) - (e^* - b^*)) * \varepsilon$$

While $e^* - b^* + 1 > 0$ and from the hypothesis that $w < \varepsilon$:

$$(9) \qquad (e^* - b^* + 1) * w + ((e - b) - (e^* - b^*)) * \varepsilon < (e - b + 1) * \varepsilon$$

From the inequality 8 and 9:

$$(e - b + 1) * w' < (e - b + 1) * \varepsilon$$

$e - b + 1 > 0 \Rightarrow w' < \varepsilon$.

If Q^* was preferred, then again

$$w' = w < \varepsilon.$$

□

Theorem 6 *The next algorithm called Iterative Dynamic Programming (IDP) computes w and $\overline{Q^*}$ defined in the Equations 5, 2 and 3.*

1. $\varepsilon := InitialValue$

2. $< w', S > = DP(\varepsilon)$

3. *cycle:*
   $S' = S$
   $\varepsilon = w'$

4. $< w', S > = DP(\varepsilon)$

5. *if(S'!=S)*
   GoTo cycle
   otherwise
   report(S')

Where:
$DP(\varepsilon)$

{

1. *Compute the path* S_b^e *using DP.*

2. $w = -\log P(M|X_b^e)$

3. *return* $< w, S >$

}

Proof. From Proposition 3, after the 2^{nd} step we have $\varepsilon > w$. From Lemma 5, each further cycle of the algorithm decreases ε.

From Lemma 2 and Lemma 5, results that the convergence appear at $w' = w$.

While the number of possible paths is limited, the algorithm is sure to converge. □

If all the transition probabilities are 1, the value of convergence of ε is equal with the geometric average of the emission probabilities in the optimal pair between a path in M and a subsequence of X.

Similarly with Lemma 4 we ca show that:

Lemma 7 *If* $\varepsilon_1 > \varepsilon_2$ *then the sequence* Q_{ε_1} *is not shorter then* Q_{ε_2}.

$$e_2 - b_2 \leq e_1 - b_1 \tag{10}$$

Proof. While $\overline{Q_{\varepsilon_1}}$ was preferred to $\overline{Q_{\varepsilon_2}}$ for $\varepsilon = \varepsilon_1$, it means that:

$$\varepsilon_1(b_1 - e_1 + N - 1) + w_1(e_1 - b_1 + 1) \leq \varepsilon_1(b_2 - e_2 + N - 1) + w_2(e_2 - b_2 + 1)$$

$$(11) \quad \varepsilon_1((e_2 - b_2) - (e_1 - b_1)) + (e_1 - b_1 + 1)w_1 - (e_2 - b_2 + 1)w_2 < 0$$

If for ε_2, $\varepsilon_2 < \varepsilon_1$, we would obtain $\overline{Q_{\varepsilon_2}}$ with $e_2 - b_2 > e_1 - b_1$, $(e_2 - b_2) - (e_1 - b_1) > 0$, then by subtracting $(\varepsilon_1 - \varepsilon_2)((e_2 - b_2) - (e_1 - b_1)) > 0$ from the first part of the inequality 11 we obtain:

$$\varepsilon_2((e_2 - b_2) - (e_1 - b_1)) + (e_1 - b_1 + 1)w_1 - (e_2 - b_2 + 1)w_2 < 0$$

showing that $\overline{Q_{\varepsilon_1}}$ will still be preferred to $\overline{Q_{\varepsilon_2}}$. This is a contradiction and we can infer that $e_2 - b_2 \leq e_1 - b_1$ □

From the previous lemma and from Lemma 5 we notice that at each step before convergence, the length of Q_ε decreases. The number of steps is therefore roughly upper bounded by N which is the same with the number of DP runs needed in order to compute the matches for all the subsequences of X.

4 Conclusions

We have present an algorithm consisting in the development of a sequence converging toward an interesting value from the point of view of the applications. The sequence is important while in practical cases [4] it was found that for reaching convergence, much less computations are needed then by using the standard known method.

5 Acknowledgments

We acknowledge the help of Giulia Bernardis and professor Hérve Bourlard in performing the tests of the IDP algorithm and for providing test data. We want to thank professors Hérve Bourlard and Sabău Ştefan for revising our first manuscripts as well as for hints related to the used notations. While the article was written, the first author was working at the Artificial Intelligence Laboratory of the Swiss Federal Institute of Technology in Lausanne and was sponsored by the Swiss National Science Foundation under project number 21-52462.97.

References

[1] L.E. Baum and T. Petrie. Statistical Inference for Probabilistic Functions of Finite State Markov Chains. *Annals of Mathematical Statistics*, 37:1554–1563, 66.

[2] R. Bellman. On the theory of dynamic programming. *Proceedings of the National Academy of Sciences*, (38):716–719, 52.

[3] R. Bellman. *Dynamic Programming*. Princeton University Press, New Jersey, 57.

[4] M.C. Silaghi and H. Bourlard. Posterior based keyword spotting without Filler Models. Technical report, EPFL, 99.

[5] M. Sniedovich. Dynamic programming and the principle of optimality: a systematic approach. *Advances in Water Resources*, 3(1):131–139, 78.

[6] M. Sniedovich. *The Joy of Dynamic Programming*. http://www.maths.mu.oz.au/ moshe/dp/joy.html, 99.

[7] A.J. Viterbi. Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm. *IEEE Trans. on Information Theory*, 13(2):260–269, 67.

Received: 21.02.1999

† Swiss Federal Institute of Technology (EPFL), Lausanne (CH)
‡ North University of Baia-Mare (RO)
silaghi@lia.di.epfl.ch, vberinde@univer.ubm.ro