WIKIPEDIA

Baum–Welch algorithm

In electrical engineering, statistical computing and bioinformatics, the **Baum–Welch algorithm** is a special case of the EM algorithm used to find the unknown parameters of a hidden Markov model (HMM). It makes use of the forward-backward algorithm to compute the statistics for the expectation step.

Contents

History
Description
  Algorithm
    Forward procedure
    Backward procedure
    Update
    Multiple sequences
Example
Applications
  Speech recognition
  Cryptanalysis
  Applications in bioinformatics
    Finding genes
      Prokaryotic
      Eukaryotic
    Copy-number variation detection
Implementations
See also
References
External links

History

The Baum–Welch algorithm was named after its inventors Leonard E. Baum and Lloyd R. Welch. The algorithm and the Hidden Markov models were first described in a series of articles by Baum and his peers at the Institute for Defense Analyses in the late 1960s and early 1970s.[1] One of the first major applications of HMMs was to the field of speech processing.[2] In the 1980s, HMMs were emerging as a useful tool in the analysis of biological systems and information, and in particular genetic information.[3] They have since become an important tool in the probabilistic modeling of genomic sequences.[4]

Description

A hidden Markov model describes the joint probability of a collection of "hidden" and observed discrete random variables. It relies on the assumption that the i-th hidden variable given the $(i-1)$-th hidden variable is independent of previous hidden variables, and the current observation variables depend only on the current hidden state.

The Baum–Welch algorithm uses the well known EM algorithm to find the maximum likelihood estimate of the parameters of a hidden Markov model given a set of observed feature vectors.

Let X_t be a discrete hidden random variable with N possible values (i.e. We assume there are N states in total). We assume the $P(X_t \mid X_{t-1})$ is independent of time t, which leads to the definition of the time-independent stochastic transition matrix

$$A = \{a_{ij}\} = P(X_t = j \mid X_{t-1} = i).$$

The initial state distribution (i.e. when $t = 1$) is given by

$$\pi_i = P(X_1 = i).$$

The observation variables Y_t can take one of K possible values. We also assume the observation given the "hidden" state is time independent. The probability of a certain observation y_i at time t for state $X_t = j$ is given by

$$b_j(y_i) = P(Y_t = y_i \mid X_t = j).$$

Taking into account all the possible values of Y_t and X_t, we obtain the $N \times K$ matrix $B = \{b_j(y_i)\}$ where b_j belongs to all the possible states and y_i belongs to all the observations.

An observation sequence is given by $Y = (Y_1 = y_1, Y_2 = y_2, \ldots, Y_T = y_T)$.

Thus we can describe a hidden Markov chain by $\theta = (A, B, \pi)$. The Baum–Welch algorithm finds a local maximum for $\theta^* = \operatorname{arg\,max}_\theta P(Y \mid \theta)$ (i.e. the HMM parameters θ that maximize the probability of the observation).[5]

Algorithm

Set $\theta = (A, B, \pi)$ with random initial conditions. They can also be set using prior information about the parameters if it is available; this can speed up the algorithm and also steer it toward the desired local maximum.

Forward procedure

Let $\alpha_i(t) = P(Y_1 = y_1, \ldots, Y_t = y_t, X_t = i \mid \theta)$, the probability of seeing the observations $y_1, y_2, \ldots, y_t$ and being in state i at time t. This is found recursively:

1. $\alpha_i(1) = \pi_i b_i(y_1)$,
2. $\alpha_i(t+1) = b_i(y_{t+1}) \sum_{j=1}^{N} \alpha_j(t) a_{ji}$.

Since this series converges exponentially to zero, the algorithm will numerically underflow for longer sequences.[6] However, this can be avoided in a slightly modified algorithm by scaling α in the forward and β in the backward procedure below.

Backward procedure

Let $\beta_i(t) = P(Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \mid X_t = i, \theta)$ that is the probability of the ending partial sequence $y_{t+1}, \ldots, y_T$ given starting state i at time t. We calculate $\beta_i(t)$ as,

1. $\beta_i(T) = 1$,
2. $\beta_i(t) = \sum_{j=1}^{N} \beta_j(t+1) a_{ij} b_j(y_{t+1})$.

Update

We can now calculate the temporary variables, according to Bayes' theorem:

$$\gamma_i(t) = P(X_t = i \mid Y, \theta) = \frac{P(X_t = i, Y \mid \theta)}{P(Y \mid \theta)} = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)},$$

which is the probability of being in state i at time t given the observed sequence Y and the parameters θ

$$\xi_{ij}(t) = P(X_t = i, X_{t+1} = j \mid Y, \theta) = \frac{P(X_t = i, X_{t+1} = j, Y \mid \theta)}{P(Y \mid \theta)} = \frac{\alpha_i(t) a_{ij} \beta_j(t+1) b_j(y_{t+1})}{\sum_{k=1}^{N} \sum_{w=1}^{N} \alpha_k(t) a_{kw} \beta_w(t+1) b_w(y_{t+1})},$$

which is the probability of being in state i and j at times t and $t+1$ respectively given the observed sequence Y and parameters θ.

The denominators of $\gamma_i(t)$ and $\xi_{ij}(t)$ are the same ; they represent the probability of making the observation Y given the parameters θ.

The parameters of the hidden Markov model θ can now be updated:

- $\pi_i^* = \gamma_i(1)$,

which is the expected frequency spent in state i at time 1.

- $a_{ij}^* = \frac{\sum_{t=1}^{T-1} \xi_{ij}(t)}{\sum_{t=1}^{T-1} \gamma_i(t)}$,

which is the expected number of transitions from state *i* to state *j* compared to the expected total number of transitions away from state *i*. To clarify, the number of transitions away from state *i* does not mean transitions to a different state *j*, but to any state including itself. This is equivalent to the number of times state *i* is observed in the sequence from *t* = 1 to *t* = *T* − 1.

- $b_i^*(v_k) = \frac{\sum_{t=1}^{T} 1_{y_t = v_k} \gamma_i(t)}{\sum_{t=1}^{T} \gamma_i(t)}$,

where

$$1_{y_t = v_k} = \begin{cases} 1 & \text{if } y_t = v_k, \\ 0 & \text{otherwise} \end{cases}$$

is an indicator function, and $b_i^*(v_k)$ is the expected number of times the output observations have been equal to v_k while in state i over the expected total number of times in state i.

These steps are now repeated iteratively until a desired level of convergence.

Note: It is possible to over-fit a particular data set. That is, $P(Y \mid \theta_{\text{final}}) > P(Y \mid \theta_{\text{true}})$. The algorithm also does **not** guarantee a global maximum.

Multiple sequences

The algorithm described thus far assumes a single observed sequence $Y = y_1, \ldots, y_N$. However, in many situations, there are several sequences observed: $Y_1, \ldots, Y_R$. In this case, the information from all of the observed sequences must be used in the update of the parameters A, π, and b. Assuming that you have computed $\gamma_{ir}(t)$ and $\xi_{ijr}(t)$ for each sequence $y_{1,r}, \ldots, y_{N_r,r}$, the parameters can now be updated:

- $\pi_i^* = \frac{\sum_{r=1}^{R} \gamma_{ir}(1)}{R}$
- $a_{ij}^* = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T-1} \xi_{ijr}(t)}{\sum_{r=1}^{R} \sum_{t=1}^{T-1} \gamma_{ir}(t)}$,
- $b_i^*(v_k) = \frac{\sum_{r=1}^{R} \sum_{t=1}^{T} 1_{y_{tr} = v_k} \gamma_{ir}(t)}{\sum_{r=1}^{R} \sum_{t=1}^{T} \gamma_{ir}(t)}$,

where

$$1_{y_{tr}=v_k} = \begin{cases} 1 & \text{if } y_{t,r} = v_k, \\ 0 & \text{otherwise} \end{cases}$$

is an indicator function

Example

Suppose we have a chicken from which we collect eggs at noon every day. Now whether or not the chicken has laid eggs for collection depends on some unknown factors that are hidden. We can however (for simplicity) assume that there are only two states that determine whether the chicken lays eggs. Now we don't know the state at the initial starting point, we don't know the transition probabilities between the two states and we don't know the probability that the chicken lays an egg given a particular state.[7][8] To start we first guess the transition and emission matrices.

Transition

	State 1	State 2
State 1	0.5	0.5
State 2	0.3	0.7

Emission

	No Eggs	Eggs
State 1	0.3	0.7
State 2	0.8	0.2

Initial

State 1	0.2
State 2	0.8

We then take a set of observations (E = eggs, N = no eggs): N, N, N, N, N, E, E, N, N, N

This gives us a set of observed transitions between days: NN, NN, NN, NN, NE, EE, EN, NN, NN

The next step is to estimate a new transition matrix. For example, the probability of the sequence NN and the state being S_1 then S_2 is given by the following, $P(S_1) * P(N|S_1) * P(S_1 \rightarrow S_2) * P(N|S_2)$.

Observed sequence	Highest probability of observing that sequence if state is S_1 then S_2	Highest Probability of observing that sequence	
NN	0.024 = 0.2 * 0.3 * 0.5 * 0.8	0.3584	S_2,S_2
NN	0.024 = 0.2 * 0.3 * 0.5 * 0.8	0.3584	S_2,S_2
NN	0.024 = 0.2 * 0.3 * 0.5 * 0.8	0.3584	S_2,S_2
NN	0.024 = 0.2 * 0.3 * 0.5 * 0.8	0.3584	S_2,S_2
NE	0.006 = 0.2 * 0.3 * 0.5 * 0.2	0.1344	S_2,S_1
EE	0.014 = 0.2 * 0.7 * 0.5 * 0.2	0.0490	S_1,S_1
EN	0.056 = 0.2 * 0.7 * 0.5 * 0.8	0.0896	S_2,S_2
NN	0.024 = 0.2 * 0.3 * 0.5 * 0.8	0.3584	S_2,S_2
NN	0.024 = 0.2 * 0.3 * 0.5 * 0.8	0.3584	S_2,S_2
Total	0.22	2.4234	

Thus the new estimate for the S_1 to S_2 transition is now $\frac{0.22}{2.4234} = 0.0908$ (referred to as "Pseudo probabilities" in the following tables). We then calculate the S_2 to S_1, S_2 to S_2 and S_1 to S_1 transition probabilities and normalize so they add to 1. This gives us the updated transition matrix:

Old Transition Matrix

	State 1	State 2
State 1	0.5	0.5
State 2	0.3	0.7

New Transition Matrix (Pseudo Probabilities)

	State 1	State 2
State 1	0.0598	0.0908
State 2	0.2179	0.9705

New Transition Matrix (After Normalization)

	State 1	State 2
State 1	0.3973	0.6027
State 2	0.1833	0.8167

Next, we want to estimate a new emission matrix,

Observed Sequence	Highest probability of observing that sequence if E is assumed to come from S_1		Highest Probability of observing that sequence	
NE	0.1344	S_2,S_1	0.1344	S_2,S_1
EE	0.0490	S_1,S_1	0.0490	S_1,S_1
EN	0.0560	S_1,S_2	0.0896	S_2,S_2
Total	0.2394		0.2730	

The new estimate for the E coming from S_1 emission is now $\frac{0.2394}{0.2730} = 0.8769$.

This allows us to calculate the emission matrix as described above in the algorithm, by adding up the probabilities for the respective observed sequences. We then repeat for if N came from S_1 and for if N and E came from S_2 and normalize.

Old Emission Matrix

	No Eggs	Eggs
State 1	0.3	0.7
State 2	0.8	0.2

New Emission Matrix (Estimates)

	No Eggs	Eggs
State 1	0.0404	0.8769
State 2	1.0000	0.7385

New Emission Matrix (After Normalization)

	No Eggs	Eggs
State 1	0.0441	0.9559
State 2	0.5752	0.4248

To estimate the initial probabilities we assume all sequences start with the hidden state S_1 and calculate the highest probability and then repeat for S_2. Again we then normalize to give an updated initial vector.

Finally we repeat these steps until the resulting probabilities converge satisfactorily.

Applications

Speech recognition

Hidden Markov Models were first applied to speech recognition by James K. Baker in 1975.[9] Continuous speech recognition occurs by the following steps, modeled by a HMM. Feature analysis is first undertaken on temporal and/or spectral features of the speech signal. This produces an observation vector. The feature is then compared to all sequences of the speech recognition units. These units could be phonemes, syllables, or whole-word units. A lexicon decoding system is applied to constrain the paths investigated, so only words in the system's lexicon (word dictionary) are investigated. Similar to the lexicon decoding, the system path is further constrained by the rules of grammar and syntax. Finally, semantic analysis is applied and the system outputs the recognized utterance. A limitation of many HMM applications to speech recognition is that the current state only depends on the state at the previous time-step, which is unrealistic for speech as dependencies are often several time-steps in duration.[10] The Baum–Welch algorithm also has extensive applications in solving HMMs used in the field of speech synthesis.[11]

Cryptanalysis

The Baum–Welch algorithm is often used to estimate the parameters of HMMs in deciphering hidden or noisy information and consequently is often used in cryptanalysis. In data security an observer would like to extract information from a data stream without knowing all the parameters of the transmission. This can involve reverse engineering a channel encoder.[12] HMMs and as a consequence the Baum–Welch algorithm have also been used to identify spoken phrases in encrypted VoIP calls.[13] In addition HMM cryptanalysis is an important tool for automated investigations of cache-timing data. It allows for the automatic discovery of critical algorithm state, for example key values.[14]

Applications in bioinformatics

Finding genes

Prokaryotic

The GLIMMER (Gene Locator and Interpolated Markov ModelER) software was an early gene-finding program used for the identification of coding regions in prokaryotic DNA.[15][16] GLIMMER uses Interpolated Markov Models (IMMs) to identify the coding regions and distinguish them from the noncoding DNA. The latest release (GLIMMER3) has been shown to have increased specificity and accuracy compared with its predecessors with regard to predicting translation initiation sites, demonstrating an average 99% accuracy in locating 3' locations compared to confirmed genes in prokaryotes.[17]

Eukaryotic

The GENSCAN webserver is a gene locator capable of analyzing eukaryotic sequences up to one million base-pairs (1 Mbp) long.[18] GENSCAN utilizes a general inhomogeneous, three periodic, fifth order Markov model of DNA coding regions. Additionally, this model accounts for differences in gene density and structure (such as intron lengths) that occur in different isochores. While most integrated gene-finding software (at the time of GENSCANs release) assumed input sequences contained exactly one gene, GENSCAN solves a general case where partial, complete, or multiple genes (or even no gene at all) is present.[19] GENSCAN was shown to exactly predict exon location with 90% accuracy with 80% specificity compared to an annotated database.[20]

Copy-number variation detection

Copy-number variations (CNVs) are an abundant form of genome structure variation in humans. A discrete-valued bivariate HMM (dbHMM) was used assigning chromosomal regions to seven distinct states: unaffected regions, deletions, duplications and four transition states. Solving this model using Baum-Welch demonstrated the ability to predict the location of CNV breakpoint to approximately 300 bp from micro-array experiments.[21] This magnitude of resolution enables more precise correlations between different CNVs and across populations than previously possible, allowing the study of CNV population frequencies. It also demonstrated a direct inheritance pattern for a particular CNV.

Implementations

- Accord.NET (http://accord-framework.net/docs/html/T_Accord_Statistics_Models_Markov_HiddenMarkovModel.htm) in C#
- ghmm (http://ghmm.org/) C library with Python bindings that supports both discrete and continuous emissions.
- HMMBase (https://github.com/maxmouchet/HMMBase.jl) package for Julia.
- HMMFit function in the RHmm (https://cran.r-project.org/src/contrib/Archive/RHmm/) package for R.
- hmmtrain (http://www.mathworks.com.au/help/stats/hmmtrain.html?searchHighlight=baum-welch) in MATLAB
- rustbio (https://docs.rs/bio/0.39.0/bio/stats/hmm/index.html) in Rust

See also

- Viterbi algorithm
- Hidden Markov model
- EM algorithm
- Maximum likelihood
- Speech recognition

- Bioinformatics
- Cryptanalysis

References

1. Rabiner, Lawrence. "First Hand: The Hidden Markov Model" (http://www.ieeeghn.org/wiki/index.php/First-Hand:The_Hidden_Markov_Model). IEEE Global History Network. Retrieved 2 October 2013.
2. Jelinek, Frederick; Bahl, Lalit R.; Mercer, Robert L. (May 1975). "Design of a linguistic statistical decoder for the recognition of continuous speech". *IEEE Transactions on Information Theory*. **21** (3): 250–6. doi:10.1109/tit.1975.1055384 (https://doi.org/10.1109%2Ftit.1975.1055384).
3. Bishop, Martin J.; Thompson, Elizabeth A. (20 July 1986). "Maximum likelihood alignment of DNA sequences". *Journal of Molecular Biology*. **190** (2): 159–65. doi:10.1016/0022-2836(86)90289-5 (https://doi.org/10.1016%2F0022-2836%2886%2990289-5). PMID 3641921 (https://pubmed.ncbi.nlm.nih.gov/3641921).
4. Durbin, Richard (23 April 1998). *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids* (https://books.google.com/books?id=R5P2GlJvigQC). Cambridge University Press. ISBN 978-0-521-62041-3.
5. Bilmes, Jeff A. (1998). *A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*. Berkeley, CA: International Computer Science Institute. pp. 7–13.
6. Rabiner, Lawrence (February 1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech recognition" (https://www.ece.ucsb.edu/Faculty/Rabiner/ece259/Reprints/tutorial%20on%20hmm%20and%20applications.pdf) (PDF). Proceedings of the IEEE. Retrieved 29 November 2019.
7. "Baum-Welch and HMM applications" (http://www.biostat.jhsph.edu/bstcourse/bio638/notes/HMMs_BaumWelch.pdf) (PDF). Johns Hopkins Bloomberg School of Public Health. Retrieved 11 October 2019.
8. Frazzoli, Emilio. "Intro to Hidden Markov Models: the Baum-Welch Algorithm" (http://ocw.mit.edu/courses/aeronautics-and-astronautics/16-410-principles-of-autonomy-and-decision-making-fall-2010/lecture-notes/MIT16_410F10_lec21.pdf) (PDF). Aeronautics and Astronautics, Massachusetts Institute of Technology. Retrieved 2 October 2013.
9. Baker, James K. (1975). "The DRAGON system—An overview". *IEEE Transactions on Acoustics, Speech, and Signal Processing*. **23**: 24–29. doi:10.1109/TASSP.1975.1162650 (https://doi.org/10.1109%2FTASSP.1975.1162650).
10. Rabiner, Lawrence (February 1989). "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition". *Proceedings of the IEEE*. **77** (2): 257–286. CiteSeerX 10.1.1.381.3454 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.381.3454). doi:10.1109/5.18626 (https://doi.org/10.1109%2F5.18626).
11. Tokuda, Keiichi; Yoshimura, Takayoshi; Masuko, Takashi; Kobayashi, Takao; Kitamura, Tadashi (2000). "Speech Parameter Generation Algorithms for HMM-Based Speech Synthesis". *IEEE International Conference on Acoustics, Speech, and Signal Processing*. **3**.
12. Dingel, Janis; Hagenauer, Joachim (24 June 2007). "Parameter Estimation of a Convolutional Encoder from Noisy Observations". *IEEE International Symposium on Information Theory*.
13. Wright, Charles; Ballard, Lucas; Coull, Scott; Monrose, Fabian; Masson, Gerald (2008). "Spot me if you can: Uncovering spoken phrases in encrypted VoIP conversations". *IEEE International Symposium on Security and Privacy*.
14. Brumley, Bob; Hakala, Risto (2009). *Cache-Timing Template Attacks*. *Advances in Cryptography*. Lecture Notes in Computer Science. **5912**. pp. 667–684. doi:10.1007/978-3-642-10366-7_39 (https://doi.org/10.1007%2F978-3-642-10366-7_39). ISBN 978-3-642-10365-0.
15. Salzberg, Steven; Delcher, Arthur L.; Kasif, Simon; White, Owen (1998). "Microbial gene identification using interpolated Markov Models" (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC147303). *Nucleic Acids Research*. **26** (2): 544–548. doi:10.1093/nar/26.2.544 (https://doi.org/10.1093%2Fnar%2F26.2.544). PMC 147303 (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC147303). PMID 9421513 (https://pubmed.ncbi.nlm.nih.gov/9421513).
16. "Glimmer: Microbial Gene-Finding System" (http://ccb.jhu.edu/software/glimmer/index.shtml). Johns Hopkins University - Center for Computational Biology.
17. Delcher, Arthur; Bratke, Kirsten A.; Powers, Edwin C.; Salzberg, Steven L. (2007). "Identifying bacterial genes and endosymbiont DNA with Glimmer" (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2387122). *Bioinformatics*. **23** (6): 673–679. doi:10.1093/bioinformatics/btm009 (https://doi.org/10.1093%2Fbioinformatics%2Fbtm009). PMC 2387122 (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2387122). PMID 17237039 (https://pubmed.ncbi.nlm.nih.gov/17237039).
18. Burge, Christopher. "The GENSCAN Web Server at MIT" (https://web.archive.org/web/20130906115338/http://genes.mit.edu/GENSCAN.html). Archived from the original (http://genes.mit.edu/GENSCAN.html) on 6 September 2013. Retrieved 2 October 2013.

19. Burge, Chris; Karlin, Samuel (1997). "Prediction of Complete Gene Structures in Human Genomic DNA". *Journal of Molecular Biology*. **268** (1): 78–94. CiteSeerX 10.1.1.115.3107 (https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.115.3107). doi:10.1006/jmbi.1997.0951 (https://doi.org/10.1006%2Fjmbi.1997.0951). PMID 9149143 (https://pubmed.ncbi.nlm.nih.gov/9149143).
20. Burge, Christopher; Karlin, Samuel (1998). "Finding the Genes in Genomic DNA". *Current Opinion in Structural Biology*. **8** (3): 346–354. doi:10.1016/s0959-440x(98)80069-9 (https://doi.org/10.1016%2Fs0959-440x%2898%2980069-9). PMID 9666331 (https://pubmed.ncbi.nlm.nih.gov/9666331).
21. Korbel, Jan; Urban, Alexander; Grubert, Fabien; Du, Jiang; Royce, Thomas; Starr, Peter; Zhong, Guoneng; Emanuel, Beverly; Weissman, Sherman; Snyder, Michael; Gerstein, Marg (12 June 2007). "Systematic prediction and validation of breakpoints associated with copy-number variations in the human genome" (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1891248). *Proceedings of the National Academy of Sciences of the United States of America*. **104** (24): 10110–5. Bibcode:2007PNAS..10410110K (https://ui.adsabs.harvard.edu/abs/2007PNAS..10410110K). doi:10.1073/pnas.0703834104 (https://doi.org/10.1073%2Fpnas.0703834104). PMC 1891248 (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1891248). PMID 17551006 (https://pubmed.ncbi.nlm.nih.gov/17551006).

External links

- A comprehensive review of HMM methods and software in bioinformatics – Profile Hidden Markov Models (https://web.archive.org/web/20110304083408/http://bioinformatics.oxfordjournals.org/content/14/9/755.full.pdf)
- Early HMM publications by Baum:
  - A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains (https://web.archive.org/web/20131005001241/http://projecteuclid.org/DPubS?verb=Display&version=1.0&service=UI&handle=euclid.aoms%2F1177697196&page=record)
  - An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology (https://web.archive.org/web/20131004233805/http://projecteuclid.org/DPubS?verb=Display&version=1.0&service=UI&handle=euclid.bams%2F1183528841&page=record)
  - Statistical Inference for Probabilistic Functions of Finite State Markov Chains (https://web.archive.org/web/20131004233211/http://projecteuclid.org/DPubS?verb=Display&version=1.0&service=UI&handle=euclid.aoms%2F1177699147&page=record)
- The Shannon Lecture by Welch, which speaks to how the algorithm can be implemented efficiently:
  - Hidden Markov Models and the Baum–Welch Algorithm (http://backup.itsoc.org/publications/nltr/it_dec_03final.pdf), IEEE Information Theory Society Newsletter, Dec. 2003.
- An alternative to the Baum–Welch algorithm, the Viterbi Path Counting algorithm:
  - Davis, Richard I. A.; Lovell, Brian C.; "Comparing and evaluating HMM ensemble training algorithms using train and test and condition number criteria" (http://dl.acm.org/citation.cfm?id=1011682.1011688), Pattern Analysis and Applications, vol. 6, no. 4, pp. 327–336, 2003.
- An Interactive Spreadsheet for Teaching the Forward-Backward Algorithm (http://www.cs.jhu.edu/~jason/papers/#eisner-2002-tnlp) (spreadsheet and article with step-by-step walkthrough)
- Formal derivation of the Baum–Welch algorithm (http://pandamatak.com/people/anand/771/html/node26.html)
- Implementation of the Baum–Welch algorithm (http://www.kanungo.com/software/software.html#umdhmm)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Baum–Welch_algorithm&oldid=1063639410"

This page was last edited on 4 January 2022, at 03:31 (UTC).