Çukurova Üniversitesi Mühendislik Fakültesi Dergisi
Çukurova University Journal of the Faculty of Engineering
ISSN: 2757-9255 CİLT/VOLUME: 39 SAYI/ISSUE: 4 ARALIK/DECEMBER 2024

The Utilization of Single-Counter Systems Featuring Final Terminals with Non-Zero Counter Values

Mehmet KURUCAN [1,a], **Dominik WOJTCZAK** [2,b]

[1] *Ardahan University, Faculty of Engineering, Department of Computer Engineering, Ardahan, Türkiye,*
[2] *University of Liverpool, Faculty of Engineering, Department of Computer Science, Liverpool, UK*

[a]***ORCID***: *0000-0003-4359-3726;* [b]***ORCID*** *0000-0001-5560-0546*

Article Info
Received : 31.05.2024
Accepted : 23.12.2024
DOI: 10.21605/cukurovaumfd.1606100

Corresponding Author
Mehmet KURUCAN
mehmetkurucan@ardahan.edu.tr

Keywords
Hidden Markov models
Probabilistic Context-free Grammars
Probabilistic one counter automata
Probabilistic pushdown automata
Natural language processing

How to cite: *KURUCAN, M., WOJTCZAK, D., (2024). The Utilization of Single-Counter Systems Featuring Final Terminals with Non-Zero Counter Values. Cukurova University, Journal of the Faculty of Engineering, 39(4), 999-1014.*

ABSTRACT

Hidden probabilistic one counter models (HPOCA) that are a specific model where spotting between hidden Markov models (HMMs) and probabilistic context-free grammars (PCFGs) which is a subclass of probabilistic pushdown automata contains only one stack symbol In this study, we propose a new model in which the final terminal counter value is different from zero. With this proposed model, we enhance the existing HPOCA, making it more complex. Consequently, as there will be a greater number of paths to reach the final terminal, we also evaluate the probability of reaching the target through alternative routes based on the given observation sequence. It makes the model more expressive than default HPOCA due to providing alternative final terminals. However, the inference of the final counter value could easily go to an infinite number without any threshold. A boundary is applied to prevent the occurrence of this unexpected condition. By applying this threshold value, we ensured that the computational complexity of the model is quadratic rather than cubic.

Sıfırdan Farklı Sayaç Değerlerine Sahip Final Terminalleri İçeren Tek Sayaçlı Sistemlerin Kullanımı

Makale Bilgileri
Geliş : 31.05.2024
Kabul : 23.12.2024
DOI: 10.21605/cukurovaumfd.1606100

Sorumlu Yazar
Mehmet KURUCAN
mehmetkurucan@ardahan.edu.tr

Anahtar Kelimeler
Gizli Markov modelleri
Olasılıksal bağlamdan bağımsız gramerler
Olasılıksal tek sayaç otomatları
Olasılıksal itme otomatları
Doğal dil işleme

Atıf şekli: *KURUCAN, M., WOJTCZAK, D., (2024). The Utilization of Single-Counter Systems Featuring Final Terminals with Non-Zero Counter Values. Cukurova University, Journal of the Faculty of Engineering, 39(4), 999-1014.*

ÖZ

Gizli olasılıklı bir sayaç modeli (HPOCA), gizli Markov modelleri (HMMs) ile olasılıklı bağlamdan bağımsız gramerler (PCFGs) arasında tespitin yalnızca bir yığın sembolü içerdiği belirli bir modeldir. Bu çalışmada, son terminal sayaç değerinin sıfırdan farklı olduğu yeni bir model öneriyoruz. Önerilen bu modelle, mevcut HPOCA'yı geliştirerek daha karmaşık hale getiriyoruz. Sonuç olarak, son terminale ulaşmak için daha fazla yol olacağından, verilen gözlem dizisine göre alternatif yollar aracılığıyla hedefe ulaşma olasılığını da değerlendiriyoruz. Alternatif son terminaller sağladığı için modeli varsayılan HPOCA'dan daha anlamlı hale getiriyor. Ancak, son sayaç değerinin çıkarımı herhangi bir eşik olmaksızın kolayca sonsuz bir sayıya gidebilir. Bu beklenmeyen durumun oluşmasını önlemek için bir sınır uygulanır. Bu eşik değerini uygulayarak, modelin hesaplama karmaşıklığının kübik değil, ikinci dereceden olmasını sağlamış oluyoruz.

1. INTRODUCTION

Complex nested hierarchical structures are a major issue for modelling sequential data. They might be found in many different fields such as natural language, gene modelling, and queuing systems. There are two major types of models to challenge this issue: Hidden Markov Models (HMMs) and probabilistic context-free grammars (PCFGs). It Is known that HMMs are commonly used for both their computational complexity and simplicity, however, it is not sufficient while facing more complex problems. On the other hand, PCFGs are a sufficient model to be used for complex problems but they are slow learners (i.e. higher computational complexity than HMMs).

In the light of this consideration, we propose a model that has an attractive spot in between standard HMM and PCFG. This model is more suitable to face more complex problems than HMMs provide, and it has less computational complexity than PCFGs have. In this paper, we introduce the characteristic feature of the final terminal, which is substantially formed with zero counter value in the default HPOCA model. Probabilistic one counter automata (POCA) is a trending known model that is used to recognize the subclass language of probabilistic context-free languages. This aspect is used as a comprehensible model for queuing systems [1] or epidemic modelling [2]. In these systems the counter value is used as a key role when tracking the number of clients in the queue or figure out the number of infected patients. POCAs are also commonly used in the analysis of software such as [3] and [4].

The language recognition of this model is a subclass of probabilistic context-free languages as HPOCA recognises. It is called probabilistic one-counter languages. The parsing of these languages is notably expensive due the parsing algorithm used for these languages (i.e. known as CYK algorithm) has a cubic computational complexity. Reducing this computational complexity, the structure of HMM which has quadratic algorithms and recognizes the probabilistic regular languages encompassed by probabilistic context-free languages is adapted to HPOCA as mentioned in [6].

In systems with queues, the counter value is constrained as much as possible, despite the system potentially presenting an infinite appearance. In order to further approach the potential of an infinite system and to develop a more impactful model, a modification was made to the final terminal feature of the existing HPOCA [5]. In this update, even if a process ends at a final terminal and the counter value is different from zero, it still indicates that the process was accepted, unlike in the present HPOCA where the counter value should always be zero. This new HPOCA model renders the current model more complex. This augmented feature places the model into the spot where it challenges more complex problems than the current model does. Along with this enables it to parse the recognized language more accurately.

1.1. Related Work

The definition of deterministic one-counter automata (OCA) and its structure that accepts the languages are represented in [7]. According to this definition, we see that one-counter automatas, which has only one stack symbol, accept the subclass of context-free languages which is called one-counter languages. POCA is augmented by adding probability values to the transition functions of the OCA. It is extensively discussed in [8] and [9]. According to the definitions, the proposed model also sits between two distinct models which are Markov chains (MCs) and probabilistic pushdown automata (PPDA) [10,11] as the model discussed in this work.

When examining the applications used by OCA, there are several purposes. For instance, it is employed in parsing bracketed arithmetic expressions [12]. Additionally, OCA is utilized for validating XML documents [13]. The work presented in [14] adapts only the Viterbi algorithm for POCA decoding. However, it is less concise than [5]. Because, [5] studies adapt all algorithms of HMM to enable the updating of parameters and yielding more favourable results.

On the other hand, HMM is widely utilized in various fields. For instance, in natural language processing [15]. In other instances, some other extensions are applied to create HMM models of different structures. Lexicon free-HMMs [16] has been employed for handwriting recognition and Weighted features in HMM [17] model has been used in removing spam from SMS. The other extension applied to HMM is called hierarchical hidden Markov model. It is applied to detect the resistance gene in biolgy [18] and inferring behaviour [19].

None of the models under consideration in this study are capable of simulating the model proposed here. The reason is that these models have finite states. However, the proposed model contains infinite states and thus this makes it a more complex structure and being applicable to more intricate problems.

1.2. Structure of the Paper

This article is adapted from the thesis discussed in [6]. Unlike [5], it allows final terminals to have non-zero counter value. The structure of the article proceeds as follows: In Section 2, the definition of the model and its constraints are discussed. Adapted algorithms are addressed in Section 3. The results of the comparison with HMM and [5] are examined in Section 4. The last section, Section 5, contains the conclusion part.

2.2. MODEL DEFINITION

In [5], the distinctive feature of the final terminal of the model is produced with zero counter value. In this new model, we do, however, slightly strengthen the traditional acceptance rule of the counter machines by accepting that the counter value of the final terminal is non-zero. To accommodate non-zero counter values, we will introduce an appropriate notation for the final terminal of this model by defining as (Equation 1):

$$Q_F = \{(q_F, c) | c \geq 0\} \tag{1}$$

where $q_F \in Q$.

The formal definition of this proposed model is quite similar as defined in [5]. The model has tuple $H = (Q, \Sigma, A^0, A^+, B, q_0, q_F)$ where

- Q is a finite set of hidden states
- Σ is finite set of observations
- A^0 is transition function and it is enabled when the counter value equals to zero where $A^0: Q \times \Delta_0 \times Q \rightarrow [0,1]$
- A^+ is transition function and it is enabled when the counter value is non-zero where $A^+: Q \times \Delta_+ \times Q \rightarrow [0,1]$
- B is emission function where $B: Q \times \Sigma \rightarrow [0,1]$
- $q_0 \in Q$ is initial terminal
- $q_F \in Q$ is designated the final terminal

The Δ_0 and Δ_+ are possible changes for the counter value. They are enabled depending on the current counter value. If the current counter value is zero then Δ_0={0,1} is enabled. According to the possible changes, at the next step, the counter ought to be still zero or increase one. If the counter value is non-zero then Δ_+={-1,0,1} is enabled and at the next step the counter might be still zero, increase one or decrease one.

The probability distribution of every state $q \in Q$ we have;

- $\Sigma_{\Delta \in \Delta_0, q' \in Q} A^0(q, \Delta, q') = 1$
- $\Sigma_{\Delta \in \Delta_+, q' \in Q} A^+(q, \Delta, q') = 1$
- $\Sigma_{o \in \Sigma} B(q, o) = 1$

The form of the proposed model is shown as a pair (q, c). Here, $q \in Q$ is a hidden state, and $c \in \mathbb{N}_0$ is a non-negative integer. The initial state, it is unique, is formed as $(q_0, 0)$ and the final terminal is configured as (q_F, c). We will show these configurations by S_t where *t* is a time step. When we run this configurations at time *t* and counter value equals to zero, we will get a probabilistic value as (Equation 2).

$$Pr(S_{t+1} = (q', \Delta)|S_t = (q, 0)) = A^0(q, \Delta, q') \tag{2}$$

If the counter value is non-zero at time *t*, then the configuration is formed as (Equation 3)

$$Pr(S_{t+1} = (q', c + \Delta)|S_t = (q, 0)) = A^+(q, \Delta, q') \tag{3}$$

Here, q is the current hidden state at time *t*. q' is the next hidden state at time step *t+1*. Δ is the difference of the counter values between states. If counter value equals to zero $\Delta \in \Delta_0$ and if counter value is non-zero $\Delta \in \Delta_+$.

A processing (i.e. run) of T (i.e. length of given observation sequence) in this model is a finite trace as shown in Figure 1. It starts at initial terminal $(q_0, 0)$ at time *t=0*. It is terminated (i.e. accepted) if (q_T, c_T) = (q_F, c), $c \geq 0$.

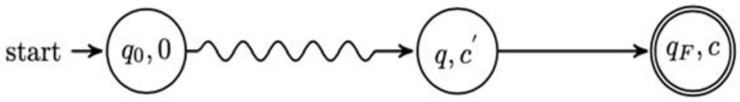

Figure 1. A simple trace of the model

The inference of the counter value in this model may potentially escalate to infinity without a predefined threshold. It is necessary to include a boundary to mitigate the occurrence of this unexpected condition during model running. The threshold for the counter value boundary is actually determined by the length of the given observation sequence. The running terminates at the final terminal with a non-zero counter value upon emitting the last output symbol of the given output sequence. For example, Figure 2, let T denote the length of the given observation sequence. A potential worst-case scenario related to this aspect is that the counter value in the model arises when the model terminates at a different hidden variable. On the other hand, the counter value is non-zero at time step *t=T*, yet. It is illustrated by the red line in Figure 2. In such a scenario, the probability of the processed output sequence becomes zero.

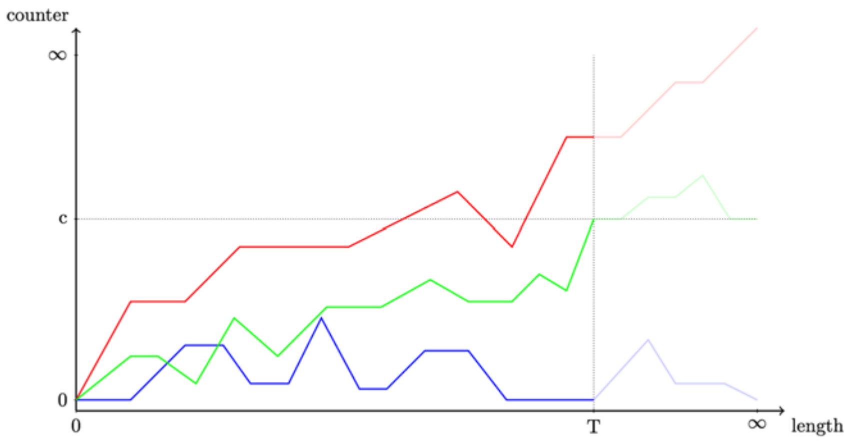

Figure 2. Three examples for counter threshold

2.1. Motivation Example

To enhance the comprehensibility of the proposed model from the reader's perspective, it would be beneficial to illustrate it using a simple motivating example. For this purpose, let us assume that we have a model with a single state (e.g., q) and a single observation symbol (e.g., a). The model parameters are defined as follows:

- Initial state distribution: $\pi(q) = 1$
- Transition probabilities:
  - $A^0(q, 0, q) = 0.4$; $A^0(q, 1, q) = 0.6$
  - $A^+(q, -1, q) = 0.4$; $A^+(q, 0, q) = 0.2$; $A^+(q, 1, q) = 0.4$
- Emission probabilities: $B(q, a) = 1$

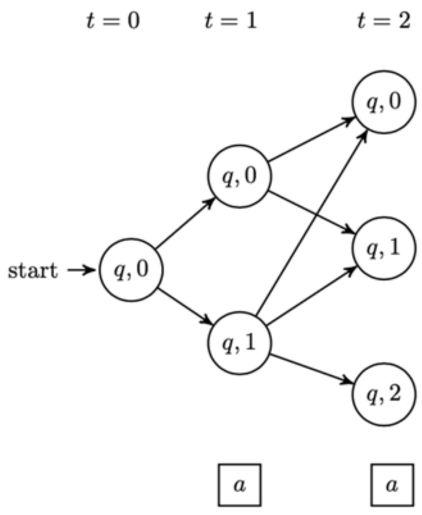

Figure 3. The trellis diagram of motivation example

Let us assume that the given observation sequence is "aa." The model will calculate the probabilistic value of this observation sequence (i.e., the forward likelihood value) by constructing a trellis diagram, as depicted in Figure 2. Since the model initiates at time *t=0*, the computation involves the initial probability of the state, the transition probabilities to possible state configurations at time *t=1*, and the emission probabilities for the symbol produced at time *t=1*. These probabilities are dynamically maintained as the product of the initial state probability, the transition probabilities, and the emission probabilities for the observed symbols.

The activation of the state transition matrices is contingent upon the counter value at time *t−1*. At *t=0*, the counter value is *0*, thus A^0 is activated. Consequently, the probability of producing the first symbol "a" in the given observation sequence at *t=1* is calculated as follows: $\pi(q) \cdot A^0(q,0,q) \cdot B(q,a)$ and $\pi(q) \cdot A^0(q,1,q).B(q,a)$. The reason for performing these two calculations is that during the transition from *t=0* to *t=1*, the counter value may either remain *0* or increase by *1*.

During the transition from *t=1* to *t=2*, it is essential to consider the transitions from *(q,0)* and *(q,1)*, as the appropriate transition matrix is activated based on the counter value (where the tuple indicates the state q alongside the counter value). The calculations are then aggregated and continued according to the length of the given observation sequence.

The main of this work, which is also mentioned in [5], is given a set of observation sequences $\boldsymbol{\mathcal{O}}$ to find a proper model $\boldsymbol{\mathcal{H}}$ with a set of hidden states $\boldsymbol{Q}$ that produces $\boldsymbol{\mathcal{O}}$. In simple terms, processing such calculations would require the multiplication of all possible hidden states with counter values. This would result in exponential computation. To avoid this extensive computation and bring it to a polynomial time we will adapt classic HMM algorithms in the next section.

3. ADAPTATION OF HMM ALGORITHMS

The learning problem in HMMs, there are three fundamental algorithms (i.e. Forward, Backward, and Baum-Welch) used to deal with it. In our proposed model, we consider the counter value that evolves the calculation of learning problems while adapting those algorithms.

When adapting the forward and backward algorithms, which are the first stage of the learning algorithm, we divided them into two separate phases. They are called preliminary and normalization, respectively. Here, the preliminary phase is similar to the calculations in the original algorithms. However, with this adaptation, considering the counter value results in a change in the calculation values. In the normalization phase, unnecessary paths are discarded to construct a complete trellis diagram connecting specific initial and final terminals.

The last adapted algorithm is the Baum-Welch algorithm. In the original version of this algorithm there are two distinct calculation functions. We use the same functions but also consider counter configuration. Among these functions, γ calculates the probability of being in a hidden state at time *t* based on the given observation sequence. The other function, ξ, calculates the transition probabilities between states based on the given observation sequence.

3.1. The Adaptation of Forward Algorithm

This algorithm is used to compute the probability value of a given observation sequence. This computation is obtained by summing the calculations of all possible paths that can generate the given observation sequence.

First Phase: Preliminary Forward Calculation

Let $o_{i:j} = o_i \ldots o_j$ is the part of the given observation sequence from *i* to *j*. In this calculation, we consider the joint probability distribution of the given observation sequence $o_{1:t}$, the hidden state-counter pair $S_t = (q, c)$ and all these are conditioned on initial terminal $S_0 = (q_0, 0)$. The calculation formally shown as (Equation 4);

$$\hat{\alpha}_t(q,c) = \Pr(\mathrm{o}_{1:t}, \mathrm{S}_t | \mathrm{S}_0) \tag{4}$$

Here, $\hat{\alpha}$ represents the preliminary Forward calculation.

The calculation start initial state at *t=0* as shown in Figure 4. It continues the calculation by summing all over the paths until reaches the hidden state S_t. The calculation is performed into two steps: Base step and Recursion step.

At Base step (i.e. *t=0*):

$$\hat{\alpha}_0(q,c) = \begin{cases} 1: & \text{if } q = q_0 \text{ and } c = 0 \\ 0: & \text{otherwise} \end{cases} \tag{5}$$

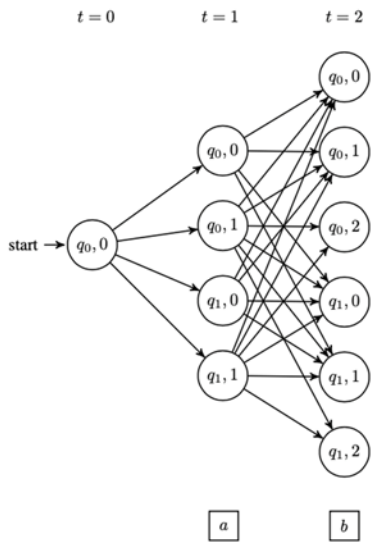

Figure 4. Trellis diagram of preliminary Forward calculation

At the Recursion step (i.e. *t>0*) :

If $c - \Delta = 0$:

$$\hat{\alpha}_t(q,c) = \sum_{q' \in Q} \sum_{\Delta=0}^{1} \hat{\alpha}_{t-1}(q', c-\Delta) A^0(q', \Delta, q) B(q, o_t) \tag{6}$$

If $c - \Delta > 0$:

$$\hat{\alpha}_t(q,c) = \sum_{q' \in Q} \sum_{\Delta=-1}^{1} \hat{\alpha}_{t-1}(q', c-\Delta) A^{+}(q', \Delta, q) B(q, o_t) \tag{7}$$

Theorem-1: $\hat{\alpha}_t(q,c)$ calculates $\Pr(\mathrm{o}_{1:t}, \mathrm{S}_t | \mathrm{S}_0)$.

Proof-1: The calculation start *t=0*, thus the function calculates $\Pr(\mathrm{o}_{1:0}, \mathrm{S}_0 | \mathrm{S}_0)$. Here $o_{1:0}$ is irrelevant. $\Pr(\mathrm{S}_0|\mathrm{S}_0) = \Pr(\mathrm{S}_0) = 1$ and at the base step if $q = q_0$ and $c = 0$, then $\hat{\alpha}_0(q,c) = 1$ otherwise it is equal to zero. Let assume that *t=k* is correct. Then we need to prove that *t=k+1*. If we put the *t* into calculation formulae, we get: $\hat{\alpha}_{k+1}(q,c) = \Pr(\mathrm{o}_{1:k+1}, \mathrm{S}_{k+1}|\mathrm{S}_0)$. If we apply chain rule, then we get: $\hat{\alpha}_{k+1}(q,c) = Pr(\mathrm{o}_{1:k+1}|S_0, S_{k+1}) Pr(S_{k+1}|S_0)$. Here o_{k+1} is conditionally independent from S_0 according to the *d-separation* rule. We get, $\hat{\alpha}_{k+1}(q,c) = Pr(\mathrm{o}_{1:k}|S_0, S_{k+1}) Pr(S_{k+1}|S_0) \Pr(o_{k+1}| S_{k+1})$. If we apply first chain rule, then apply sum rule over on S_k, and then again apply chain rule, then the calculation becomes $\hat{\alpha}_{k+1}(q,c) = \sum_{S_k} Pr(\mathrm{o}_{1:k}, S_{k+1}|S_k, S_0) \Pr(S_k|S_0) \Pr(o_{k+1}| S_{k+1})$According to the *d-separation* rule, S_{k+1} is independent from S_0 condition on S_k. Thus we got the proof if apply again chain rule $\hat{\alpha}_{k+1}(q,c) = \sum_{S_k} Pr(\mathrm{o}_{1:k}, S_k|S_0) \Pr(S_{k+1}|S_k) \Pr(o_{k+1}| S_{k+1})$. Here $Pr(\mathrm{o}_{1:k}, S_k|S_0)$ is previous calculation $\hat{\alpha}_k(q,c)$, $\Pr(S_{k+1}|S_k)$ is state transition and $\Pr(o_{k+1}| S_{k+1})$ is an observation transition. ■

Second Phase: Normalization of Forward Calculation

This second phase of the adapted algorithm is based on discarding the irrelevant path between initial and final terminals. Thus, this part calculates the joint probability distribution of the given observation sequence $\mathrm{o}_{1:t}$ and being in hidden state S_t condition on initial terminal S_0 and final terminals $S_T \in Q_F$.

The formal definition of this phase is shown as follow (Equation 8):

$$\alpha_t(q,c) = \Pr(\mathrm{o}_{1:t}, S_t|S_0, S_T \in Q_F) \tag{8}$$

The calculation of this phase starts at the end of the trellis diagram (i.e. designated final terminal with non-zero counter value) $S_T \in Q_F$ as shown in Figure 5.

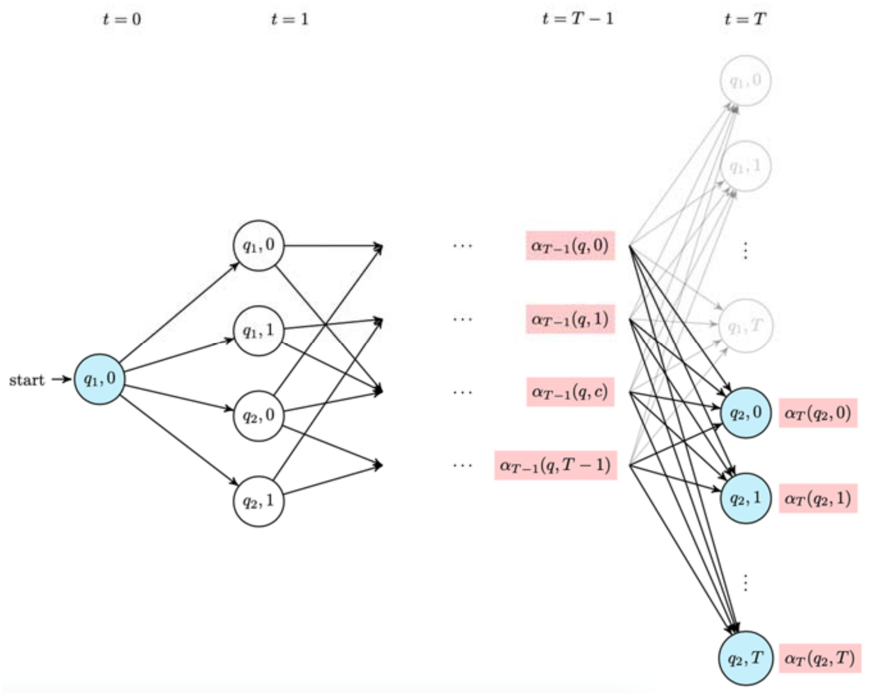

Figure 5. Trellis diagram of normalization phase of the Forward flow

As mentioned above, this model has a unique final terminal but the counter value can be non-zero depending on the length of the given observation sequence. Consequently, a run is terminated in the corresponding final terminal with non-zero counter value which is ranging from $0 \leq c \leq T$, where T denotes the length of the output sequence. The computation of this phase stars at time *t=T*. It is called Base step (Equation 9):

$$\alpha_T(q,c) = \frac{\hat{\alpha}_T(q,c)}{\sum_{c'=0}^{T} \Pr(S_T = (q,c')|S_0)} \tag{9}$$

where $q = q_F$ and c is non-zero counter value: $0 \leq c \leq T$. The computation continues to the first terminal at the recursion step (i.e. $t < T$), taking into account whether the counter value is zero or non-zero due to enable the transition matrices:

- if c=0:

$$\alpha_t(q,c) = \sum_{q' \in Q} \sum_{\Delta=0}^{1} \frac{\alpha_{t+1}(q',c+\Delta) A^0(q,\Delta,q') \hat{\alpha}_t(q,c)}{\hat{\alpha}_{t+1}(q',c+\Delta)} \tag{10}$$

- if c>0:

$$\alpha_t(q,c) = \sum_{q' \in Q} \sum_{\Delta=-1}^{1} \frac{\alpha_{t+1}(q',c+\Delta) A^+(q,\Delta,q') \hat{\alpha}_t(q,c)}{\hat{\alpha}_{t+1}(q',c+\Delta)}$$

Theorem-2: $\alpha_t(q,c)$ calculates the joint probability of $\Pr(o_{1:t}, S_t | S_0, S_T \in Q_F)$.

Proof-2: At the base step *t=T*: $\alpha_T(q,c) = \Pr(o_{1:T}, S_T = (q,c) | S_0, S_T \in Q_F)$. If we apply production rule then we get $\alpha_T(q,c) = \frac{\Pr(o_{1:T}, S_T=(q,c), S_T \in Q_F | S_0)}{\Pr(S_T \in Q_F | S_0)}$. Here, if the hidden state q is the final state and the counter value is $0 \leq c \leq T$, then $\alpha_T(q,c) = \frac{\Pr(o_{1:T}, S_T=(q,c)|S_0)}{\Pr(S_T \in Q_F|S_0)}$. The nominator of this fraction equals to preliminary calculation of forward flow at time *t=T*, and the denominator equals the sum of the probability of whole paths from initial state to final states. Thus, we get $\alpha_T(q,c) = \frac{\hat{\alpha}_T(q,c)}{\sum_{c'=0}^{T} \Pr(S_T=(q,c')|S_0)}$. At the recursion part, we again assume that the calculation is true for *t=k* where $\alpha_k(q,c) = \frac{\Pr(o_{1:k+1}, S_{k+1}|S_0, S_T \in Q_F)\Pr(S_{k+1}|S_k)\Pr(o_{1:k}, S_k|S_0)}{\Pr(o_{1:k+1}, S_{k+1}|S_0)}$ and we would like to proof the calculation is also correct for *t=k-1* and equals to $\alpha_{k-1}(q,c) = \Pr(o_{1:k-1}, S_{k-1} | S_0, S_T \in Q_F)$. We apply two production rules separately. First we apply to $\Pr(o_{1:k}, S_k | S_0, S_T \in Q_F)$ and get $\frac{\Pr(o_{1:k}, S_k, S_0, S_T \in Q_F)}{\Pr(S_0, S_T \in Q_F)}$, then the second production rule is applied to $\Pr(o_{1:k}, S_k, S_0, S_T \in Q_F)$ as $\Pr(o_{1:k}, S_0, S_T \in Q_F | S_k)\Pr(S_k)$. According to the *d-separation* rule o_k is independent from S_0 and S_T depending on S_k. Thus we get $\Pr(o_{1:k-1}, S_0, S_T \in Q_F | S_k)\Pr(o_k|S_k)$. If we apply chain rule to $\Pr(o_{1:k-1}, S_{k-1}|S_0)$ then we get $\Pr(o_{1:k-1}|S_0, S_{k-1})\Pr(S_{k-1}|S_0)$. We can merge $\Pr(S_k|S_{k-1})$ with $\Pr(o_{1:k-1}|S_0, S_{k-1})$ according to the *d-separation* rule. We get here $\Pr(o_{1:k-1}, S_k|S_0, S_{k-1})$. We can again merge $\Pr(S_{k-1}|S_0)$ with $\Pr(o_{1:k-1}, S_k|S_0, S_{k-1})$ and we get $\Pr(o_{1:k-1}, S_k, S_{k-1}|S_0)$. If we apply chain rule to denominator item $\Pr(o_{1:k}, S_k|S_0)$ then we get $\Pr(o_{1:k}|S_0, S_k)\Pr(S_k|S_0)$. According to *d-separation* rule o_k is independent from S_0 depending on S_k. Then we get $\Pr(o_{1:k-1}|S_0, S_k)\Pr(o_k|S_k)$. We can eliminate each $\Pr(o_k|S_k)$ from both numerator and denominator. In $\Pr(o_{1:k-1}, S_0, S_T \in Q_F|S_k)$, $S_T \in Q_F$ is independent from both $o_{1:k-1}$ and S_0 depending on S_k according to the *d-separation* rule. We can get $\Pr(o_{1:k-1}, S_0|S_k)\Pr(S_T \in Q_F|S_k)$. We can merge both $\Pr(S_k)$ and $\Pr(o_{1:k-1}, S_0|S_k)$ in the numerator section. Then we get $Pr(o_{1:k-1}, S_0, S_k)$. If we apply product rule to $Pr(o_{1:k-1}|S_0, S_k)$ and $Pr(S_k|S_0)$ then we get $Pr(o_{1:k-1}, S_k| S_0)$. After that we can apply again chain rule to $Pr(o_{1:k-1}, S_0, S_k)$ as $Pr(o_{1:k-1}, S_k|S_0)Pr(S_0)$. Now, we can eliminate $Pr(o_{1:k-1}, S_k|S_0)$ from both numerator and denominator. We can apply production rule to merge $Pr(o_{1:k-1}, S_k, S_{k-1}|S_0)$ and $Pr(S_0)$ in the numerator, then we get $Pr(o_{1:k-1}, S_k, S_{k-1}, S_0)$. If we apply chain rule depending on S_k, we get $Pr(o_{1:k-1}, S_{k-1}, S_0|S_k)Pr(S_k)$. According to the *d-separation* rule, $Pr(S_T \in Q_F|S_k)$ can be merged with $Pr(o_{1:k-1}, S_{k-1}, S_0|S_k)$ depending on S_k. Thus, we can get $Pr(o_{1:k-1}, S_{k-1}, S_0, S_T \in Q_F|S_k)$. After that, we can merge $Pr(S_k)$ with it, then we get $Pr(o_{1:k-1}, S_{k-1}, S_0, S_T \in Q_F, S_k)$. In here, actually we have $\sum_{S_k} \frac{Pr(o_{1:k-1}, S_{k-1}, S_0, S_T \in Q_F, S_k)}{\Pr(S_0, S_T \in Q_F)}$. If we look at this fraction carefully, it is actually equals to $\sum_{S_k} Pr(o_{1:k-1}, S_{k-1}, S_k | S_0, S_T \in Q_F)$. In here we can remove sum rule production over S_k variable, then we have $Pr(o_{1:k-1}, S_{k-1}, S_k|S_0, S_T \in Q_F)$ where it is equivalent to $\alpha_{k-1}(q,c)$. ■

3.2. The Adaptation of Backward Algorithm

This suggested model might be implemented using the second modified dynamic programming method backward algorithm, which consists of two primary computing phases again: preliminary backward calculation and normalisation of backward flow. The probability of the future observation sequence $o_{t+1:T}$ condition hidden variable S_t is being computed in this calculation.

First Phase: Preliminary Backward Calculation

In this phase, the calculation begins from the final terminal. Therefore, if we consider the length of a given observation sequence as T, and since this computation also calculates the probability of future observation sequences, the calculation starts from the final terminal would be appropriate.

In the proposed model, since the counter value at the final terminal will be non-zero. Thus, the counter value of the final terminal in the range $0 \leq c \leq T$ might be considered. Therefore, the computation will take into account all counter values, as illustrated in Figure 6.

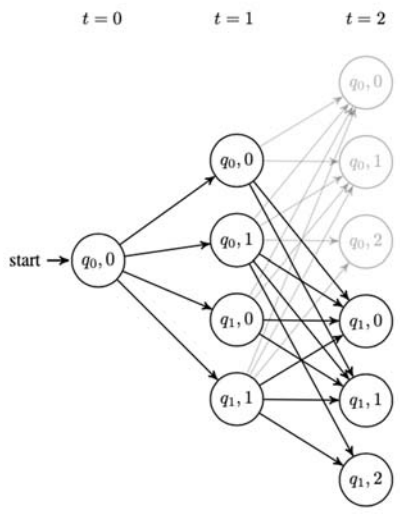

Figure 6. Trellis diagram of preliminary calculation of adapted Backward algorithm

This calculation is denoted as $\widehat{\beta}_t(q,c)$. It starts from the final terminal towards the initial terminal. At time *t=T*, the preliminary Backward calculation value of the final terminal, considering all designated counter values, is set to one. This knowledge is obtained from the initiation of the calculation at the final terminal, whose location is also known. It differs from Forward flow normalisation in that there is no additional symbol probability added to the computation.

The calculation can be shown formally as follow (Equation 11):

$$\widehat{\beta}_t(q,c) = Pr(S_T \in Q_F | S_t = (q,c), S_0)$$

At Base Step (t=T):

$$\widehat{\beta}_t(q,c) = 1 \text{ If } q = q_F \text{ and } c: 0 \leq c \leq T \tag{11}$$

At Recursion Step (t<T):

- If $c = 0$:

$$\hat{\beta}_t(q,c) = \sum_{q' \in Q} \sum_{\Delta=0}^{1} \hat{\beta}_{t+1}(q', c+\Delta) A^{0}(q, \Delta, q')$$

- If $c > 0$:

$$\hat{\beta}_t(q,c) = \sum_{q' \in Q} \sum_{\Delta=-1}^{1} \hat{\beta}_{t+1}(q', c+\Delta) A^{+}(q, \Delta, q') \tag{12}$$

Theorem-3: $\widehat{\beta_t}(q,c)$ computes the probability of $Pr(S_T \in Q_F | S_t = (q,c), S_0)$ correctly.

Proof-3: We can start with the base step of the calculation. For *t=T*, the calculation becomes as $\widehat{\beta_T}(q,c) = Pr(S_T \in Q_F | S_T = (q,c), S_0)$. Here, $S_T \in Q_F$ is conditionally independent from initial terminal S_0. Thus, we get $Pr(S_T \in Q_F | S_T = (q,c))$. Here, if the hidden variable $q = q_F$ and the counter value $0 \leq c \leq T$ then $Pr(S_T \in Q_F | S_T \in Q_F)$. Thus the probability equals 1. At the recursion part (i.e. *0<t< T*), we assume that the calculation is correct for *t=k*, and now we need to prove that the same equation also provides the correct calculation for *t=k-1*. We can start the proof of this part by adding sum rule over hidden state S_k as $\sum_{S_k} \Pr(S_T \in Q_F, S_k | S_{k-1}, S_0)$. Then we can apply chain rule as $\sum_{S_k} \Pr(S_T \in Q_F | S_{k-1}, S_k, S_0) \Pr(S_k | S_{k-1}, S_0)$. The first order of the Markov assumption: the future state only depends on the current state. Thus, $\Pr(S_k | S_{k-1}, S_0)$ becomes to $\Pr(S_k | S_{k-1})$. According to the *d-separation* rule, any $S_T \in Q_F$ is conditionally independent from S_{k-1} condition on S_k and we get $\sum_{S_k} \Pr(S_T \in Q_F | S_k, S_0) \Pr(S_k | S_{k-1})$. Here, $\Pr(S_T \in Q_F | S_k, S_0)$ is $\widehat{\beta_k}(q,c)$ and $\Pr(S_k | S_{k-1})$ is state transition probability distribution which depends on whether the counter value is zero or non-zero. If counter equals to zero then A^0 is enabled and $\Delta = \{0,1\}$. If counter is non-zero then A^+ is enabled and $\Delta = \{-1,0,1\}$. ■

Second Phase: Normalization of Backward Calculation

In this computation phase, the probability value of the future observation sequence is calculated both with respect to the current hidden variable and conditionally with respect to the terminals (i.e., initial and final) as shown in Figure 7. It is shown formally as (Equation 13)

$$\beta_t(q,c) = Pr(o_{t+1:T} | S_t = (q,c), S_T \in Q_F, S_0) \tag{13}$$

In the recursion part of the above formulae is used as follows:

$$\beta_t(q,c) = \sum_{S_{t+1}} \Pr(o_{t+2:T} | S_{t+1}, S_T \in Q_F, S_0) \Pr(S_{t+1} | S_t) \Pr(o_{t+1} | S_{t+1}) \frac{\Pr(S_T \in Q_F | S_{t+1}, S_0)}{\Pr(S_T \in Q_F | S_t, S_0)} \tag{14}$$

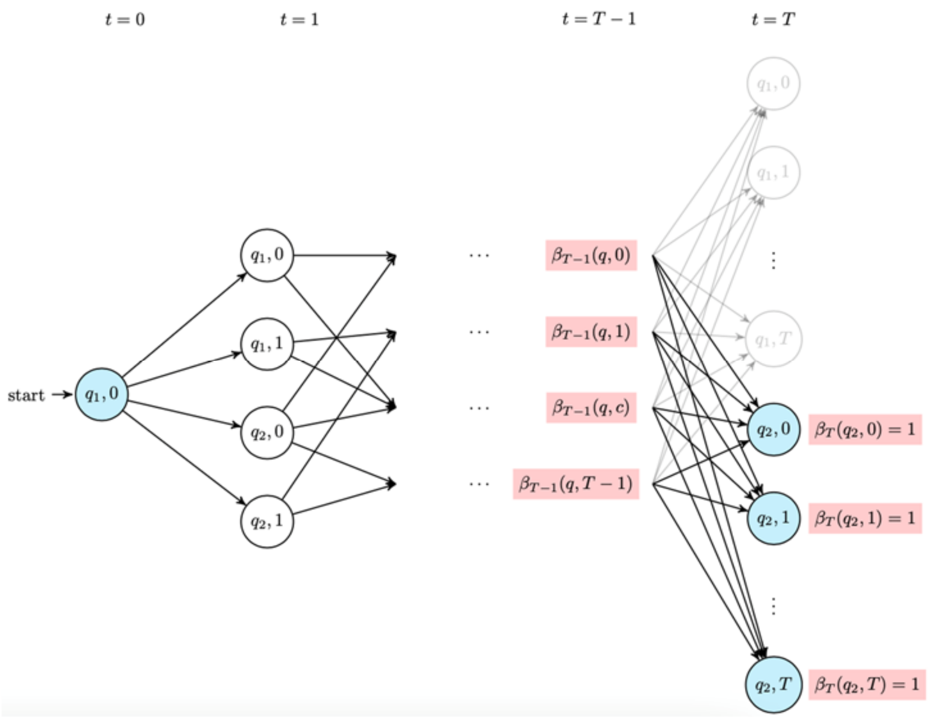

Figure 7. The calculation of Backward normalization

The normalization part of Backward flow also initiates at the end of the time scale of *t=T* as preliminary Backward flow does.

At Base Step (t=T):

$\beta_t(q,c) = 1$ If $q = q_F$ and c: $0 \leq c \ \leq T$.

At Recursion Step (t<T):

- If $c = 0$:

$$\beta_t(q,c) = \sum_{q'\in Q}\sum_{\Delta=0}^{1} \frac{\beta_{t+1}(q',\Delta)A^{0}(q,\Delta,q')B(q',o_{t+1})\hat{\beta}_{t+1}(q',\Delta)}{\hat{\beta}_t(q,c)}$$

- If $c > 0$:

$$\beta_t(q,c) = \sum_{q'\in Q}\sum_{\Delta=-1}^{1} \frac{\beta_{t+1}(q',c+\Delta)A^{+}(q,\Delta,q')B(q',o_{t+1})\hat{\beta}_{t+1}(q',c+\Delta)}{\hat{\beta}_t(q,c)} \tag{15}$$

Theorem-4: $\beta_t(q,c)$ calculates the joint probability of the future observation sequence $Pr(o_{t+1:T}|S_t = (q,c), S_T \in Q_F, S_0)$ correctly.

Proof-4: First at *t=T* the calculation becomes $\beta_T(q,c) = Pr(o_{T+1:T}|S_T = (q,c), S_T \in Q_F, S_0)$. Here, $o_{T+1:T}$ is trivial and $S_T = (q,c)$ is one if the hidden variable is final terminal otherwise, it is zero. Let assume that the calculation is correct at *t=k* and we need to proof that it is also correct for *t=k-1*. If we put to time value to Equation 9 and applying the product rule to $Pr(o_{k+1:T}|S_k, S_T \in Q_F, S_0)$ then we get $\frac{\Pr(o_{k+1:T},S_k,S_T\in Q_F,S_0)}{\Pr(S_k,S_T\in Q_F,S_0)}$. Again applying second time production rule to $Pr(o_{k+1:T}, S_k, S_T \in Q_F, S_0)$ and we get $Pr(o_{k+1:T}, S_T \in Q_F, S_0|S_k)Pr(S_k)$. We can merge these two arguments depending on S_k: $Pr(o_{k+1:T}, S_T \in Q_F, S_0|S_k)$ and $Pr(o_k|S_k)$ then we get $Pr(o_{k:T}, S_T \in Q_F, S_0|S_k)$. If we merge $Pr(S_k)$ with it then it becomes $Pr(o_{k:T}, S_T \in Q_F, S_0, S_k)$. If we apply product rule to $Pr(S_T \in Q_F|S_k, S_0)$ then it becomes $\frac{Pr(S_T\in Q_F,S_k,S_0)}{Pr(S_k,S_0)}$. We can eliminate $Pr(S_T \in Q_F, S_k, S_0)$ argument from both numerator and denominator. Again we can apply chain rule to $Pr(o_{k:T}, S_T \in Q_F, S_0, S_k)$ then it becomes $Pr(o_{k:T}, S_T \in Q_F|S_0, S_k)\ Pr(S_0, S_k)$. We can again eliminate the same argument $Pr(S_0, S_k)$ from both numerator and denominator. If we use chain rule for the argument of denominator $Pr(S_T \in Q_F|S_0, S_{k-1})$ then we get $\frac{\Pr(S_0,S_{k-1})}{Pr(S_T\in Q_F,S_0,S_{k-1})}$. Here, if we can apply chain rule to $\Pr(S_0, S_{k-1})$ as $\Pr(S_0|S_{k-1})\ \Pr(S_{k-1})$. The following arguments $\Pr(S_0|S_{k-1})$ and $\Pr(S_k|S_{k-1})$ can be merged according to *d-separation* rule conditionally S_{k-1}. Thus, we can get $\Pr(S_0, S_k|S_{k-1})$. Now we can convert the conditional distribution to joint distribution by merging $\Pr(S_0, S_k|S_{k-1})$ and $\Pr(S_{k-1})$. Then we get $\Pr(S_0, S_k, S_{k-1})$. If we again apply the chain rule this last argument as $\Pr(S_{k-1}|S_0, S_k)$ $\Pr(S_0, S_k)$ and then we can merge $\Pr(S_{k-1}|S_0, S_k)$ with $Pr(o_{k:T}, S_T \in Q_F|S_0, S_k)$ depending on the same conditional states. Then we can get $Pr(o_{k:T}, S_{k-1}S_T \in Q_F|S_0, S_k)$. After this, we can merge again $Pr(o_{k:T}, S_{k-1}S_T \in Q_F|S_0, S_k)$ with $Pr(S_0, S_k)$ to handle the joint probability as $Pr(o_{k:T}, S_{k-1}S_T \in Q_F, S_0, S_k)$. Now we can again use chain rule to get rid of the denominator part $Pr(S_T \in Q_F, S_0, S_{k-1})$. Here we get the numerator arguments as $Pr(o_{k:T}, S_k|S_{k-1}S_T \in Q_F, S_0)$ $\Pr(S_{k-1}S_T \in Q_F, S_0)$ after applying chain rule. Now we can eliminate the same argument, $\Pr(S_{k-1}S_T \in Q_F, S_0)$, from both numerator and denominator. Here, we have $\sum_{S_k} Pr(o_{k:T}, S_k|S_{k-1}S_T \in Q_F, S_0)$ and we can remove the sum rule then we will get $Pr(o_{k:T}, S_k|S_{k-1}S_T \in Q_F, S_0)$ which is equal to $\beta_{k-1}(q,c)$ calculation at time *t=k-1*. ■

3.3. The Adaptation of Baum-Welch Algorithm

The original form of the Baum-Welch algorithm, which is a special case of the EM algorithm, is used to update the model's parameters in HMMs. The parameters are made to converge to their respective values by applying a certain number of iterations. In our proposed model, we will use the adapted version that takes counter values into account to update the parameters.

The algorithm consists of two steps as its original version has. The first step, called the E-step, involves two distinct calculations: the probabilities of being in hidden states and the transition probabilities between hidden states. The second step, known as the M-step, involves updating the parameters.

E-step

In this step, the two different calculations used are denoted as $\gamma_t(q,c)$ and $\xi_t((q',c'),(q,c))$, respectively. The $\gamma_t(q,c)$ calculation provides the probabilities of being in the current hidden states for a given observation sequence. Formally (Equation 16),

$$\gamma_t(q,c) = \Pr(S_t = (q,c) | O, S_0, S_T \in Q_F) \tag{16}$$

here O is the given observation sequence. The consideration of the non-zero counter value of the final terminal is proceed in this model, then there need to be sum over the counter value at time $t=T$. Thus, the formula is represented as follow (Equation 17):

$$\gamma_t(q,c) = \frac{\hat{\alpha}_t(q,c)\beta_t(q,c)\hat{\beta}_t(q,c)}{\sum_{c'=0}^{T} \hat{\alpha}_T(q_F,c')} \tag{17}$$

Theorem-5: $\gamma_t(q,c)$ calculates correctly the marginal posterior distribution of S_t where $\Pr(S_t = (q,c) | O, S_0, S_T \in Q_F)$.

Proof-5: The proof can start by applying the production rule to Equation 10. We can get $\frac{\Pr(S_t, O, S_T \in Q_F | S_0)}{\Pr(O, S_T \in Q_F | S_0)}$. The observation sequence can be expended into two separated parts as $o_{1:t}$ and $o_{t+1:T}$. We can apply this separated parts into numerator as arguments: $\Pr(S_t, o_{1:t}, o_{t+1:T}, S_T \in Q_F | S_0)$. After this adding if we apply production rule again, we get $\Pr(o_{1:t}, o_{t+1:T}, S_T \in Q_F | S_0, S_t) Pr(S_t|S_0)$. According to *d-separation* rule, $o_{1:t}$ is conditionally independent from both $o_{t+1:T}$ and , $S_T \in Q_F$ condition on S_t. Thus, we have $\Pr(o_{1:t}|S_0, S_t)$ $\Pr(o_{t+1:T}, S_T \in Q_F | S_0, S_t) Pr(S_t|S_0)$. If we apply chain rule to $\Pr(o_{t+1:T}, S_T \in Q_F | S_0, S_t)$, then it becomes $\Pr(o_{t+1:T}|S_0, S_t, S_T \in Q_F) Pr(S_T \in Q_F | S_0, S_t)$. We can merge $\Pr(o_{1:t}|S_0, S_t)$ and $\Pr(S_t|S_0)$ depending on initial terminal S_0, then we get $\Pr(o_{1:t}, S_t|S_0)$. At last, here, $\Pr(o_{1:t}, S_t|S_0)$ is equivalent calculation of preliminary Forward algorithm $\hat{\alpha}_t(S_t)$ (i.e. equation 4), $\Pr(o_{t+1:T}|S_0, S_t, S_T \in Q_F)$ is equivalently equal to the normalization of Backward algorithm $\beta_t(S_t)$ (i.e. equation 8), and $Pr(S_T \in Q_F | S_0, S_t)$ is equivalent to preliminary Backward flow $\hat{\beta}_t(S_t)$ (i.e. equation 7). The denominator part of the proof (i.e. $Pr(O, S_T \in Q_F | S_0)$) is a preliminary Forward flow by summing over all the counter values of the final terminal. ■

The $\xi_t((q',c'),(q,c))$ calculation, on the other hand, computes the transition probabilities between hidden states for the given observation sequence. Formally (Equation 18),

$$\xi_t\big((q',c'),(q,c)\big) = \Pr(S_{t-1}, S_t | O, S_0, S_T \in Q_F) \tag{18}$$

here O is the given observation sequence, $S_{t-1} = (q',c')$ previous hidden state and counter pair, $S_t = (q,c)$ current hidden state and counter pair. The multiple counter values of the final terminal is also considered in this calculation. Therefore, the formula becomes as follow (Equation 19) :

$$\xi_t\big((q',c'),(q,c)\big) = \frac{\hat{\alpha}_t(q',c')A^{\delta(c')}(q',\Delta,q)B(q,o)\hat{\beta}_t(q,c)\beta_t(q,c)}{\sum_{c'=0}^{T} \hat{\alpha}_T(q_F,c')} \tag{19}$$

Here, $\delta(c')$=0 if c'=0 and $\delta(c')$= + if c'>0. Also, Δ={0,1} if c'=0 and Δ ={-1,0,1} if c'>0.

Theorem-6: $\xi_t\big((q',c'),(q,c)\big)$ correctly calculates the joint posterior distribution of two hidden states $S_{t-1} = (q',c')$ and $S_t = (q,c)$ where $\Pr(S_{t-1}, S_t | O, S_0, S_T \in Q_F)$.

Proof-6: The proof can start by applying the production rule to equation 12. We can get $\frac{\Pr(S_{t-1}, S_t, O, S_T \in Q_F | S_0)}{\Pr(O, S_T \in Q_F | S_0)}$. We can expand the given observation sequence as we already did in proof-5 as $o_{1:t}$ and $o_{t+1:T}$. The numerator part becomes $\Pr(S_{t-1}, S_t, o_{1:t}, o_{t+1:T}, S_T \in Q_F | S_0)$. After that we can apply chain rule as $\Pr(o_{1:t}, o_{t+1:T}, S_T \in Q_F | S_0, S_{t-1}, S_t) Pr(S_{t-1}, S_t | S_0)$. According to the *d-separation* rule, o_t is conditionally independent from $o_{1:t-1}$, $o_{t+1:T}, S_T \in Q_F$, S_{t-1} ,and S_0 condition on S_t. Thus, we have

$\Pr(o_{1:t-1}, o_{t+1:T}, \mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0}, \mathrm{S_{t-1}}, \mathrm{S_t})Pr(S_{t-1}, S_t|S_0)\Pr(\mathrm{o_t}|\mathrm{S_t})$. According to the *d-separation* rule, the previous observation sequence part $o_{1:t-1}$ is conditionally independent from $o_{t+1:T}, \mathrm{S_T} \in \mathrm{Q_F}$, and $\mathrm{S_t}$ condition on $\mathrm{S_{t-1}}$. Thus, we have $\Pr(\mathrm{o_{1:t-1}}|\mathrm{S_{t-1}}, \mathrm{S_0})\Pr(o_{t+1:T}, \mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0}, \mathrm{S_{t-1}}, \mathrm{S_t})Pr(S_{t-1}, S_t|S_0)\Pr(\mathrm{o_t}|\mathrm{S_t})$.. Here, $\mathrm{S_{t-1}}$ is irrelevant in $\Pr(o_{t+1:T}, \mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0}, \mathrm{S_{t-1}}, \mathrm{S_t})$ due to the future observation sequence part and the set of final terminals are conditionally independent with it. Thus, we have $\Pr(o_{t+1:T}, \mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0}, \mathrm{S_t})$. Now, we can apply chain rule it and we get $\Pr(\mathrm{o}_{t+1:T}|\mathrm{S_0}, \mathrm{S_t}, \mathrm{S_T} \in \mathrm{Q_F})\Pr(\mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0}, \mathrm{S_t})$. We can apply chain rule to $Pr(S_{t-1}, S_t|S_0)$ as $Pr(S_t|S_0, S_{t-1})\Pr(\mathrm{S_{t-1}}|\mathrm{S_0})$. Then, we can merge $\Pr(\mathrm{o_{1:t-1}}|\mathrm{S_{t-1}}, \mathrm{S_0})$and $\Pr(\mathrm{S_{t-1}}|\mathrm{S_0})$ to handle $\Pr(\mathrm{o_{1:t-1}}, \mathrm{S_{t-1}}|\mathrm{S_0})$.According to Markov property $\mathrm{S_0}$ is irrelevant in $Pr(S_t|S_0, S_{t-1})$. We can remove it and this argument becomes $Pr(S_t|S_{t-1})$. At the end of this proof, $\Pr(\mathrm{o_{1:t-1}}, \mathrm{S_{t-1}}|\mathrm{S_0})$ is preliminary Forward calculation at time *t-1*: $\hat{\alpha}_{t-1}(S_{t-1})$, $\Pr(\mathrm{o}_{t+1:T}|\mathrm{S_0}, \mathrm{S_t}, \mathrm{S_T} \in \mathrm{Q_F})$ is normalization of Backward flow at time *t*: $\beta_t(S_t)$, $\Pr(\mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0}, \mathrm{S_t})$ is preliminary Backward calculation at time *t*: $\hat{\beta}_t(S_t)$, $Pr(S_t|S_{t-1})$ is the probability of state transition $A^{\delta(c')}(S_{t-1}, \Delta, S_t)$ where $\delta(c')$=0 if c'=0 and $\delta(c')$= + if c'>0. Also, Δ={0,1} if c'=0 and Δ ={-1,0,1} if c'>0. The last argument $\Pr(\mathrm{o_t}|\mathrm{S_t})$ is symbol emitting probability where is equivalent to $B(S_t, o_t)$. The denominator part of the proof (i.e. $Pr(O, \mathrm{S_T} \in \mathrm{Q_F}|\mathrm{S_0})$) is a preliminary Forward flow by summing over all the counter values of the final terminal.■

M-step

In the M-step, the model's parameters (θ) are updated using the previous parameters (θ'). This process is similar to the approach in [5]. Multiple observation sequences are used to ensure convergence of the parameters when updating them.Here, D represents the observation sequence number. This calculation proceeds through the joint probability distribution over the hidden variables, counter values and the set of given observation sequences (Equation 20):

$$\Pr(S, C, O|\theta) = \prod_{d=1}^{D}\prod_{t=1}^{T} B\left(S_t^{(d)}, o_t^{(d)}\right)A^{\delta\left(C_t^{(d)}\right)}\left(S_t^{(d)}, C_{t+1}^{(d)} - C_t^{(d)}, S_{t+1}^{(d)}\right) \tag{20}$$

where $\delta\left(C_t^{(d)}\right)$=0 if $C_t^{(d)}$=0 and $\delta\left(C_t^{(d)}\right)$=+ if $C_t^{(d)}$>0. The Lagrange Multipliers is used in equation 14 to get the converged parameters of the model. The update formulas of the parameters are shown as follows, respectively:

$$\hat{B}(q, o) = \frac{\sum_{d=1}^{D}\sum_{t=1}^{T}\gamma_t^{(d)}(S_t^{(d)})I(o_t^{(d)}=o)}{\sum_{d=1}^{D}\sum_{t=1}^{T}\gamma_t^{(d)}(S_t^{(d)})} \tag{21}$$

$$\hat{A}^0(q, \Delta, q') = \frac{\sum_{d=1}^{D}\sum_{\{t|C_t^{(d)}=0\}}\xi_t^{(d)}(S_t^{(d)}, S_{t+1}^{(d)})}{\sum_{d=1}^{D}\sum_{S_{t+1}^{(d)}}\sum_{\{t|C_t^{(d)}=0\}}\xi_t^{(d)}(S_t^{(d)}, S_{t+1}^{(d)})} \tag{22}$$

$$\hat{A}^+(q, \Delta, q') = \frac{\sum_{d=1}^{D}\sum_{\{t|C_t^{(d)}>0\}}\xi_t^{(d)}(S_t^{(d)}, S_{t+1}^{(d)})}{\sum_{d=1}^{D}\sum_{S_{t+1}^{(d)}}\sum_{\{t|C_t^{(d)}>0\}}\xi_t^{(d)}(S_t^{(d)}, S_{t+1}^{(d)})} \tag{23}$$

4. IMPLEMENTATION AND RESULTS

The proposed model is compared with the HMM. The original algorithms and adapted algorithms were implemented in Python. 10 different models were created in this work and they are illustrated as $\boldsymbol{\mathcal{H}}$ to represent all models. $\mathcal{H}_k = (Q_k, \Sigma_k, A_k^0, A_k^+, B_k, q_0, q_F)$ where k=1,,10. Here, A_k^0, A_k^+, and B_k are used as parameters in the models were generated randomly. q_0 and q_F were selected manually.

Subsequently, each model generated 30,000 observation sequences. Since each run is terminated at a final terminal with multiple counter values. Thus, we kept the length of each observation sequence fixed at 14. As mentioned in [5], to have a fair and quality comparison, the HMM must have an equal number of parameters (i.e., according to the Bayesian Information Criterion) as the proposed model has. Therefore, we set the number of states in the HMM to be 2.24 times the number of states in the proposed model. We used the technique which was applied in [5] during the training models. In the first step, 100 different models were randomly used. Then, in each iteration, 25% of these models were discarded as measured by the probability value. This process continued until convergence, leaving only one model at the end. Thus, we prevent the local minima issue using this method.

Table 1. KL scores of the models after testing step

Model	$\mathcal{H}1$	$\mathcal{H}2$	$\mathcal{H}3$	$\mathcal{H}4$	$\mathcal{H}5$	$\mathcal{H}6$	$\mathcal{H}7$	$\mathcal{H}8$	$\mathcal{H}9$	$\mathcal{H}10$
Proposed Model	0.000693	0.001459	0.002913	0.003941	0.010453	0.009671	0.011952	0.016158	0.013957	0.010571
H1MM	0.01047	0.02183	0.03021	0.04023	0.12098	0.10879	0.12103	0.15941	0.14184	0.11210
HMM	3.4107	4.1067	4.5739	5.0137	4.7218	5.2167	5.7846	6.1146	6.0478	5.8362

In terms of time consumption, times used in the learning process can be shown as evidence that considering the counter value in models has higher time consumption compared to HMM.

In terms of the efficiency of the comparison, the learning durations for each of the three models were evaluated on the same task, taking into account the counter value. As shown in Table 2, for models that include a counter, the learning process was manually initialized by the user (as shown "---" in the table) after a certain threshold, due to the extended duration of learning beyond that point.

Table 2. The comparison of time consumption in learning phase

Model	$\mathcal{H}1$	$\mathcal{H}2$	$\mathcal{H}3$	$\mathcal{H}4$	$\mathcal{H}5$	$\mathcal{H}6$	$\mathcal{H}7$	$\mathcal{H}8$	$\mathcal{H}9$	$\mathcal{H}10$
Proposed Model	1h 5m	7h 17m	26h 51m	84h 48m	236h 57m	---	---	---	---	---
H1MM	58m	5h 43m	23h 7m	79h 21m	218h 36m	---	---	---	---	---
HMM	9m	28m	1h 5m	4h 11m	7h 3m	9h 47m	13h 17m	16h 28m	21h 3m	27h 23m

It should also be noted that since the counter value of the final terminal is non-zero, it can approach the length of the given observation sequence. This will lead to an increase in computational complexity (though it will still remain quadratic). We set a boundary to prevent this issue: the feasible maximum counter values will be equal to or less than half the length of the given observation sequence.

The adapted Baum-Welch algorithm used as the learning algorithm consists of multiple iterations. When we consider the length of the observation sequence as T, the feasible counter M value will approach T since we determine the counter value of the final terminal to be non-zero. In the proposed model, as noted in Section 2, we can constrain this approach using a threshold. In light of this information, considering an observation sequecne of length T and $n = |Q|$ states with n state transitions, the computational complexity of the proposed model will be $O(n^2TM)$ where the counter value can exceed half of the length of the given observation sequence.

In the testing phase, the Kullback-Leibler (KL) divergence was used as the metric to compare the performance values of the trained models (Equation 24).

$$KL(P||Q) = P(x)log\frac{P(x)}{Q(x)} \tag{24}$$

Here, the trained models calculate the probability values $Q(x)$ of the observation sequences (i.e. for each x) of the test set. Then, these values are compared logarithmically with the original probability values $P(x)$ to generate a score. If this score is close to zero, it indicates that the trained model is closely similar to the original model.

As shown in Table-1, the KL-scores of the 10 different models indicate that the proposed model is quite similar to the original model, while the HMM is significantly distant from the original model. The trade-off in our model comparison is time consumption. However, we are confident that our model is still faster than PCFG. Additionally, the fact that it is more complex than HMM. Thus, it addresses the research question of this study.

5. CONCLUSIONS

In this study, we propose a new model with a final terminal counter value that is non-zero, similar to a hidden one-counter automaton, which reduces the computational cost of the learning algorithm for probabilistic one-counter languages—a subclass of stochastic context-free languages—from cubic to quadratic. The presence of a final terminal with a non-zero counter value enables the creation of alternative final terminals. This not only renders the existing model more complex but also allows for the evaluation of non-zero counter values with a higher likelihood through various alternative paths leading to the final terminal. The learning and testing algorithms of the model have been adapted from the forward, backward, Baum-Welch, and Viterbi algorithms used in hidden Markov models. This adaptation allows us to benefit from the quadratic computational complexity of these algorithms. Although the proposed model achieves better results in practical applications compared to hidden Markov models (HMMs), it has been observed that the time required for learning is longer than that of HMMs. Additionally, due to the final terminal having a non-zero counter value, the counter value can increase significantly depending on the length of the given observation sequence. To address this issue, a threshold has been implemented, thus maintaining the computational complexity at a quadratic level.

6. REFERENCES

1. Etessami, K., Wojtczak, D., Yannakakis, M., 2010. Quasi-birth-death processes, tree-like qbds, probabilistic 1-counter automata, and pushdown systems, Perform. Eval., 67(9), 837-857.
2. Peng, L., Xie, P., Tang, Z., Liu, F., 2021. Modeling and analyzing transmission of infectious diseases using generalized stochastic petri nets. Applied Sciences, 11(18), 8400.
3. Ouaknine, J., Sousa-Pinto, J., Worrell, J., 2014. On termination of integer linear loops. Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms, 957-969.
4. Ben-Amram, A.M., Genaim, S., 2013. On the linear ranking problem for integer linear-constraint loops. In Proceedings of the 40th Annual ACM Symposium on Principles of programming languages, Rome, Italy, 51-62.
5. Kurucan, M., Özbaltan, M., Schewe, S., Wojtczak, D., 2022. Hidden 1-counter Markov models and how to learn them. International Joint Conferences on Artificial Intelligence Organization, Wien Austria, 4857-4863.
6. Kurucan, M., 2020. Hidden probabilistic one-counter automata. PhD Thesis, University of Liverpool, 157.
7. Valiant, L.G., Paterson, M.S., 1975. Deterministic one-counter automata. Journal of Computer and System Sciences, 10(3), 340-350.
8. Etessami, K., Wojtczak, D., Yannakakis, M., 2010. Quasi-birth-death processes, tree-like qbds, probabilistic 1-counter automata, and pushdown systems. Perform. Eval., 67(9), 37-857.
9. Wojtczak, D., 2009. Recursive probabilistic models: efficient analysis and implementation. PhD Thesis, University of Edinburgh, UK, 201.
10. Dubslaff, C., Baier, C., Berg, M., 2012. Model checking probabilistic systems against pushdown specifications. Information Processing Letters, 112(8-9), 320-328.
11. Brázdil, T., Esparza, J., Kiefer, S., Kucera, A., 2013. Analyzing probabilistic pushdown automata. Formal Methods in System Design. 43(2), 1-43.
12. Ford, B., 2003. Parsing expression grammars. ACM SIGPLAN Notices, 39(1), 1-12.
13. Tan, Z., 2009. Validating XML constraints using automata. Proceedings of the 2009 8th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2009. 1205-1210.
14. Sakharov, A., Sakharov, T., 2018. The viterbi algorithm for subsets of stochastic context-free languages. Information Processing Letters, 135, 68-72.
15. Siddalingappa, R., Hanumanthappa, P., Reddy, M., 2018. Hidden Markov model for speech recognition system-A pilot study and a naive approach for speech-to-text model. Advances in Intelligent Systems and Computing, 77-90.
16. Fischer, A., Keller, A., Frinken, V., Bunke, H., 2012. Lexicon-free handwritten word spotting using character HMMs. Pattern Recogn. Lett., 33(7), 934-942.

17. Xia, T., Chen, X., 2021. A weighted feature enhanced hidden Markov model for spam SMS filtering. Neurocomputing, 444, 48-58.
18. Lakin, S.M., Kuhnle, A., Alipanahi, B., 2019. Hierarchical hidden Markov models enable accurate and diverse detection of antimicrobial resistance sequences. Commun Biol., 2, 294.
19. Giada, S., Ben, S., 2021. Toward efficient Bayesian approaches to inference in hierarchical hidden Markov models for inferring animal behavior. Frontiers in Ecology and Evolution, 9, 62373.