Çukurova Üniversitesi
Mühendislik Fakültesi Dergisi
Çukurova University
Journal of the Faculty of Engineering
ISSN: 2757-9255 CİLT/VOLUME: 40 SAYI/ISSUE: 3 EYLÜL/SEPTEMBER 2025

Beam-Limited k-Step Lookahead for Computationally Efficient HMM Decoding

Mehmet KURUCAN [1,a]

[1]*Adana Alparslan Türkeş Science and Technology University, Faculty of Computer and Informatics, Department of Computer Engineering, Adana, Türkiye*

[a]***ORCID**: 0000-0003-4359-3726*

Article Info
Received : 28.05.2025
Accepted : 13.08.2025
DOI: 10.21605/cukurovaumfd.1708178

Corresponding Author
Mehmet KURUCAN
mkurucan@atu.edu.tr

Keywords
Hidden markov models
Decoding problems
Beam limited search
Sequencial estimation

***How to cite:** KURUCAN, M., (2025). Beam-Limited k-Step Lookahead for Computationally Efficient HMM Decoding. Cukurova University, Journal of the Faculty of Engineering, 40(3), 545-558.*

ABSTRACT

Hidden Markov Models (HMMs) are widely used in many sequential decision-making problems due to their ability to model time-related dependencies. The standard decoding methods in these models, such as the Viterbi algorithm, are limited by their dependence on past observations only. Thus, this leads to unpredictability when future information is available. In this work, we propose a decoding strategy called Beam-Limited k-Step Lookahead that looks k-step ahead, drawing parallels to k-step discrete control synthesis, to make use of future information. The proposed method achieves a balance between decoding accuracy and computational complexity by constraining the search space to the top M most promising paths. Experimental results on synthetic HMM data show that our new decoding strategy significantly improves decoding accuracy over classical Viterbi decoding. The findings highlight the potential of this new strategy to improve the performance of sequential decoding systems.

Hesaplama Açısından Verimli HMM Kod Çözümü İçin Demet Sınırlamalı k-Adımlı İleri Bakış

Makale Bilgileri
Geliş : 28.05.2025
Kabul : 13.08.2025
DOI: 10.21605/cukurovaumfd.1708178

Sorumlu Yazar
Mehmet KURUCAN
mkurucan@atu.edu.tr

Anahtar Kelimeler
Saklı markov modelleri
Kod çözme problemleri
Sınırlı ışın araması
Ardışık tahminleme

***Atıf şekli:** KURUCAN, M., (2025). Hesaplama Açısından Verimli HMM Kod Çözümü İçin Demet Sınırlamalı k-Adımlı İleri Bakış. Çukurova Üniversitesi, Mühendislik Fakültesi Dergisi, 40(3), 545-558.*

ÖZ

Gizli Markov Modelleri (HMM'ler), zamanla ilgili bağımlılıkları modelleme yetenekleri nedeniyle birçok ardışık karar verme probleminde yaygın olarak kullanılır. Bu modellerdeki standart kod çözme yöntemleri, Viterbi algoritması gibi, yalnızca geçmiş gözlemlere olan bağımlılıklarıyla sınırlıdır. Bu nedenle, gelecekteki bilgiler mevcut olduğunda öngörülemezliğe yol açar. Bu çalışmada, gelecekteki bilgileri kullanmak için (yani kontrol teorisindeki k-adımlı ayrık kontrol sentezine benzer bir yaklaşımla) k-adım ileriyi gören Işın Sınırlı k-Adım İleriye Bakış adı verilen bir kod çözme stratejisi öneriyoruz. Önerilen yöntem, arama alanını en umut verici M yolla sınırlayarak kod çözme doğruluğu ve hesaplama karmaşıklığı arasında bir denge sağlar. Sentetik HMM verileri üzerindeki deneysel sonuçlar, yeni kod çözme stratejimizin kod çözme doğruluğunu klasik Viterbi kod çözmeye kıyasla önemli ölçüde iyileştirdiğini göstermektedir. Bulgular, bu yeni stratejinin ardışık kod çözme sistemlerinin performansını iyileştirme potansiyelini vurgulamaktadır.

1. INTRODUCTION

The utilization of machine learning-based models in scientific research has become firmly settled. Up-to-date works are now directed towards enhancing the effectiveness and efficiency of existing Machine Learning (ML) models through the combination of different models or tools. As is well-known, the Hidden Markov Model (HMM) is considered a powerful ML tool employed for modeling sequential data in several fields such as speech recognition[1], bioinformatics[2], and handwriting recognition[3].

Three distinct algorithms are employed for three fundamental problems in HMM. The decoding problem, which is the central focus of this study, is typically solved by the Viterbi algorithm. The algorithm gives the most probable sequence of underlying hidden states as its output given a sequence of observations [4]. However, the standard Viterbi algorithm, exclusively uses the past observations. Due to its inherent structure rooted in the Markov assumption, it potentially ignores the future observations. This causal dependency means that Viterbi cannot leverage potentially rich information contained in subsequent observations by its design. This aspect leads to suboptimal state estimations in scenarios where future context is available and relevant. This limitation naturally leads us to the following question: In the context of the decoding problem, could knowledge of observations a few steps into the future help us to determine more optimal and probable sequence of hidden states?

To find a proper answer to this question, we introduce a k-step lookahead decoding strategy that incorporates future knowledges when performing state estimations with the decoder algorithm. We suppsose that utilizing information from the future in this manner will yield a more accurate estimation of the most likely hidden state sequence. However, a significant challenge arises in the computations: the computational complexity of the k-step lookahead calculation increases exponentially with the depth of k. Adapting a model that performs such expensive calculations to real-world applications will be difficult.

Effectively overcoming the computational complexity induced by the depth of the k value requires strategic method. Thus, we integrate the beam search pruning technique into the lookahead process. This method allows us to keep only the M most promising paths at each step, thereby aligning with the fundamental objective of decoding, which is to retain the path with the highest probability among the current possibilities. The proposed method offers a scalable framework that adeptly balances accuracy and computational efficiency.

Our work is determined by combining k-step lookahead with beam pruning in a lightweight and efficient manner. It is specifically applied for HMM decoding without the need for retraining or model restructuring.

1.1. Related Work

There are several works that have been proposed to improve decoding performance in Hidden Markov Models. The Viterbi algorithm [5] is the classical approach, finding the most likely sequence of hidden states based on given observations. However, this method does not exploit the information of future observations.

Lookahead decoding techniques have been investigated to address the use of future information. For instance, k-step lookahead strategies, as studied in [6] where the proposed decoder method uses future observations to improve prediction accuracy. The difference between this work and ours is positioning based on the lookahead module. Their approach depends on depth-first search and our study's lookahead is based on k-step discrete controller synthesis. We perform the decoding operation with a discrete controller for the decoding problem.

Beam search is a heuristic method widely used in natural language processing. [7] provides a practical solution by limiting the number of paths considered during decoding. Some works have applied beam search to sequence models [8,9]. However, in their works, integration with k-step lookahead has been limited.

Other approaches, such as particle filtering [10] and deep learning-based sequence models [11], have been studied for sequential decoding. Yet, these methods require large-scale computational resources.

1.2. Related Work

The remainder of this paper is organised as follows: Section 2 formally defines the problem, Section 3 describes the proposed method, Section 4 presents experimental evaluations, Section 5 discusses the results, and Sections 6 conclude with insights and future directions.

2. PROBLEM DEFINITION

An HMM is a well-known ML tool that provides a probabilistic framework for modelling sequential data, where transitions of the system between hidden states while emitting observable outputs [12]. The goal of HMM-based decoding is to determine the most probable sequence of hidden states that could have generated a given sequence of observations [5]. This is called *Decoding Problem* which is one of the classical problems of HMM. This is commonly solved using the Viterbi algorithm, which applies dynamic programming to identify the proper state path. However, the structure of the standard Viterbi is inherently causal [2]. It means that it only depends on the previous condition. Thus, its predictive capability is limited when future information is known and available. Yet, directly apply lookahead idea into into the decoding process increases computational complexity dramatically. In this point, we need an efficient decoding strategy to utilize the future observations without suffering from exponential cost. Thus, we propose a novel decoding approach to addresses this problem.

2.1. Definition of Hidden Markov Model

Hidden Markov Model (HMM) is defined by the following components [13]:

- A finite set of hidden states : $S = \{s_1, s_2, \dots, s_N\}$,
  A finite set of observations: $O = \{o_1, o_2, \dots, o_M\}$,
- A transition probability matrix: $A = [a_{ij}]$, where $a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i)$
- An emission probability matrix: $B = [\beta_j(k)]$,wher$b_j(k) = P(o_t = o_k \mid q_t = s_j)$e,
- An initial state distribution: $\pi = [q_i]$, where $q_i = P(q_0 = s_i)$.

The goal is to decode the most probable hidden state sequence $Q = (q_0, q_1, \dots, q_T)$ depending on given observation sequence $O = (o_1, o_2, \dots, o_T)$.

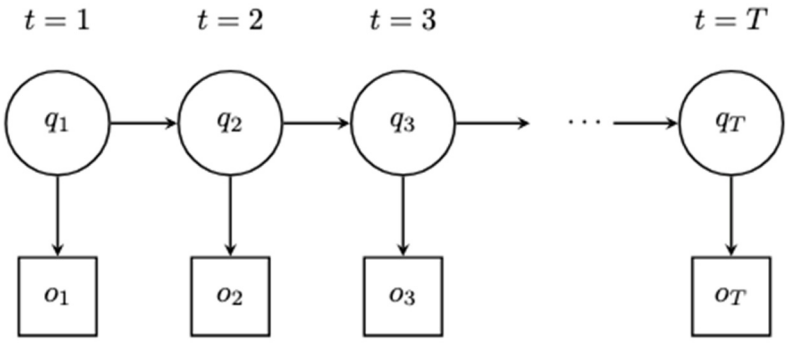

Figure 1. HMM structure showing transitions between hidden states q_t and corresponding emissions o_t

2.2. Standard Viterbi Algorithm

The standard decoding algorithm, Viterbi, is a dynamic programming approach to find the most probable hidden state sequence depending on given observation sequence. The formal computation is [14];

$$\widehat{\boldsymbol{Q}} = \boldsymbol{arg}\max_{\boldsymbol{Q}} \boldsymbol{P}(\boldsymbol{Q}|\boldsymbol{O}) \tag{1}$$

It defines;

$$\boldsymbol{\delta_t(j)} = \max_{\boldsymbol{q_1,\dots,q_{t-1}}} \boldsymbol{P(q_1, \dots, q_{t-1}, q_t = s_j, o_1, \dots, o_t)} \tag{2}$$

The steps of this algorithm are;

- Initial step: $\delta_1(j) = \pi_j b_j(o_1)$
- Recursion step: $\delta_t(j) = \max_i [\delta_{t-1}(i)a_{ij}]b_j(o_t)$

The optimal path is reconstructed using backtracking when t=T.

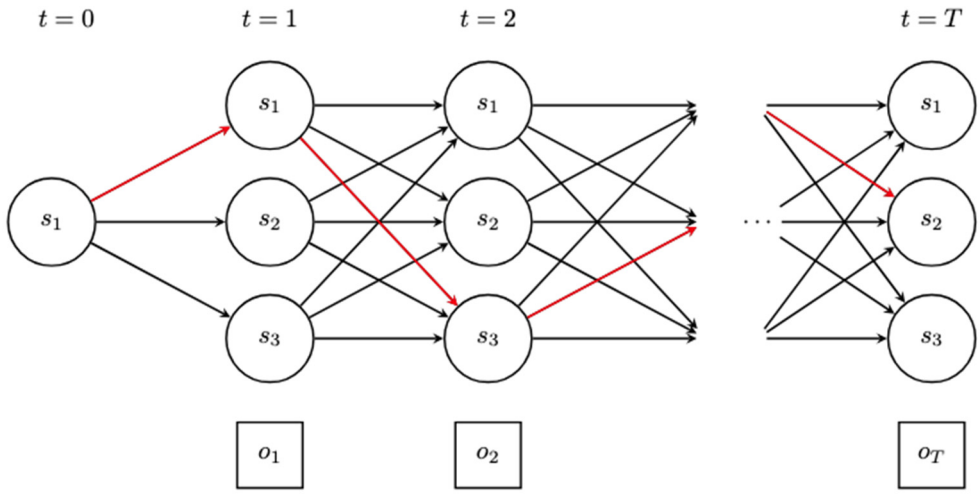

Figure 2. Viterbi decoding as dynamic programming over possible state paths that are illustrated with red arrows

2.3. Limitations and Definitions of the Proposed Model

The standart decoding algorithm oh HMM is optimal under the assumption that only past informations (i.e. observations) are known [15]. Thus, it does not exploit the future observations during the process. On the other hand, while our proposed model facilitates the consider of future steps into the calculation, thus offering a novel strategy for decoding, extending the lookahead by k steps leads to an exponential increase in computational complexity.

The problem statement:

- **Given** :
  - A HMM (A, B, π)
  - An observation sequence $O = (o_1, \dots, o_T)$
  - A desired lookahead depth k
  - A beam width constraint M
- **Design a decoding**
  - Use k -step lookahead to consider future observations
  - Avaoids exponential computation
  - Use near-optimal M value within practical runtime

This work proposes a novel decoding approach that directly addresses this problem via a beam-limited lookahead mechanism. The core challenge we tackle is utilizing k-step future observations efficiently while bounding computational complexity through beam pruning. The k-step lookahead strategy employed in this work can be designed as a form of k-step discrete control synthesis. This approach arises from the decision-making process at each time step. The system synthesizes a sequence of future hidden states where maximizes the overall likelihood of the gıven future observatıons. This process analogous to a controller optimizing future actions based on predicted states. Unlike traditional depth-first search (e.g. as [6]) methods that might explore all branches exhaustively, our approach implicitly guides this "synthesis" through probabilistic maximization. With this way, the proposed method prepares the beam pruning to efficiently manage the search space.

3. METHOD

In this section, we detail our proposed model, including definitions of the tools employed and their operational principles. As illustrated by the model's name, we will begin by defining the k-step lookahead. Next, we will discuss the application of beam search pruning to reduce the computational cost associated with the k-step. To clarify the operational principle, we will present a pseudocode representation and theoretically support the proposed strategy by providing relevant theorems and proofs.

3.1. k-Step Lookahead Formulation

HMM is traditionally designed to make sequential state estimations based only on past observations and the current hidden state. However, in this work, access to future observations can improve the accuracy of decoding. To utilize future information, we introduce a k-step lookahead strategy where at each step, future observations are added into the state estimation process.

As mentioned the definition of HMM in Section 2.1, the objective is to maximize the joint probability of $P(q_t \mid o_1, \dots, o_T)$ at each time t. However, the proposed method instead maximizes the joint probability of $P(q_t, q_{t+1}, \dots, q_{t+k} \mid o_1, \dots, o_T)$. This joint probability can be expanded by using Bayes' rule and Markov assumptions :

$$\boldsymbol{P(q_t, q_{t+1}, \dots, q_{t+k}, o_t, o_{t+1}, \dots, o_{t+k}) \propto P(q_t) \prod_{i=0}^{k} \alpha_{q_{t+i}, q_{t+i+1}} \beta_{q_{t+i}}(o_{t+i})} \tag{3}$$

Here, multiple future states and observation symbols are considered during decoding at each time step.

3.2. Beam Search Pruning

The main challenge with k-step lookahead decoding is the exponential growth of the possible state paths. The computational complexity of the classical decoding algorithm is quadratic. However, the proposed method has N^k computation cost, and this causes the model to become unmanageable for practical applications. We integrate a Beam Search strategy to address this issue by limiting the number of paths explored at each step. It is introduced in two parts as;

Strategy:

- **Beam Width M** : Keep only M top paths with highest probabilities.
- **Pruning**: Discard the outside of the top M paths to reduce the computational complexity.

Processs:

- Expend the path by considering all possible next states for active paths at t-1.
- Calculate the scor for each extended path:

$$\boldsymbol{Score(path_j) = \prod_{i=0}^{k} \alpha_{q_{t+i}, q_{t+i+1}} \beta_{q_{t+i}}(o_{t+i}),\ \forall j = 1, \dots, N} \tag{4}$$

- Sort the all scores.
- Keep only the top M paths for next expansion.

This strategy ensures that the calculation cost remains manageable while maintaning paths with high probabilities. The most significant value provided by the new strategy in this process is reducing the computational complexity. The exponential computation resulting from the classical k-step lookahead implementation is reduced to a manageable $N \times M^{k-1}$ with the new strategy where M is the Beam width. It is a small and feasible integer value such that $M \ll N^k$. The whole process is illustrated in Algorithm 1.

Algorithm 1: Beam-Limited k-Step Lookahead HMM Decoding	
Input:	
	• Transition matrix A • Emission matrix B • Initial probabilities π • Observation sequence $O = (o_1, o_2, \dots, o_T)$ • Lookahead depth k • Beam width M
Output:	
	Most likely state sequence $\hat{Q} = (\widehat{q_1}, \widehat{q_2}, \dots, \widehat{q_T})$
Initialization:	
	1. Create initial paths for each state : $P(s_i) = \pi_i b_i(o_i)$ 2. Keep top M paths depending on their probabilities.
Iteration for $t = 2\ to\ T$:	
	1. For each active path at time $t-1$: 1.a. Expand the path by considering all possible next states. 1.b. For each expansion, compute the cumulative path score by considering: $Score(path_i) = \Pi_{j=0}^{k} A_{q_{t+j-1}, q_{t+j}} \times B_{q_{t+j}}(o_{t+j})$,where A represents the transition probability $P(q_{t+j}\|q_{t+j-1})$ and B represents the emission probability $P(o_{t+j}\|q_{t+j})$. Future emissions and transitions up to k steps are multiplied. 2. Collect all extended paths. 3. Sort the paths depending on their score in desending order. 4. Keep only the top M paths
Termination:	
	1. At $t = T$, select the path with max score 2. Output the corresponding state sequence $\hat{Q}$

3.3. Beam Width Selection

Selecting an appropriate beam width □ is another important issue to handle for a good balance between decoding accuracy and computational efficiency in this work. Based on our experimental observations, a proper M value can be considered in the following aspects:

Trade-off Between Accuracy and Efficiency

- Small M values (e.g., M=3) yield faster runtime and less memory usage. However, it may discard promissing paths and returns lower decoding accuracy.
- Large M values (e.g. M=50) preserve more promissing paths and it improves the accuracy. But computational cost increases.
- The results suggest that moderate beam widths (e.g., M=20) typically provide a good trade-off, achieving high decoding accuracy with manageable runtime and memory usage.

Application-Specific Adaptation

- In real-time systems where runtime constraints are critical. A smaller beam width may be preferred for this kind of system. However, keep in mind, slight sacrifices are accepted for faster decoding.
- A larger beam width can be used to achieve maximum accuracy in the offline process without concern for runtime.

Adaptive Beam Width Adjustment

- Future implementations may benefit from the dynamic adjustment of □ during decoding based on the following scenarios:
    - For instance, if score differences between paths are large, a smaller beam width might suffice. Thus, unnecessary calculations are discarded.
    - If many paths have close scores then expanding the beam width temporarily could prevent discarding optimal paths. In this scenario, the computational cost increases but the protection of the paths that have the highest probability among the competing paths is secured.

The precise adjustment of the beam width may vary depending on the available computational resources and application requirements. For instance, the following optimal M values are provided as examples based on the number of states utilized in two distinct models. It is probable that these values will exhibit variability according to the specific task being addressed.

4. IMPLEMENTATION DETAILS

In this section, we describe practical reviews that were included in the implementation of the proposed Beam-Limited k-Step Lookahead HMM decoding algorithm. These details are important for establishing stability and computational efficiency.

The procedure of the proposed algorithm is shown as a flowchart in Figure 3. Here, the k-step value is reduced due to the potential event of a condition such as T−t<k. In other words, the value of k is dynamically decreased based on the remaining number of steps. Another adjustment made to enhance stability is the utilization of logarithmic probabilities. It will improve numerical stability by using summation instead of multiplication.

4.1. Dynamic Adjustment of Lookahead Depth

The classical structure of standard k-step lookahead decoding processes over k future observations at every time step. However, towards the end of the given observation sequence, specifically when the remaining number of time steps (T-t) is less than the desired lookahead depth (k), the algorithm cannot look ahead by the full k steps. To handle this issue and ensure the algorithm always considers only available future observations:

- The k-step depth is dynamically adjusted:

$$\boldsymbol{k_{effective} = \min(k, T - t)} \tag{5}$$

at each time step t. This adjustment ensures that as the decoding process approaches the end of the observation sequence: the lookahead depth gracefully decreases, preventing out-of-bounds access and maintaining computational accuracy. At the final time step, the decoding process secures only the available number of future observations that are added into the calculations of the path score. The stability is preserved in this way.

- At the final time step, the decoding process secures only available number of the future observations that are added into the calculations of path score. The consistency is maintained with this way.

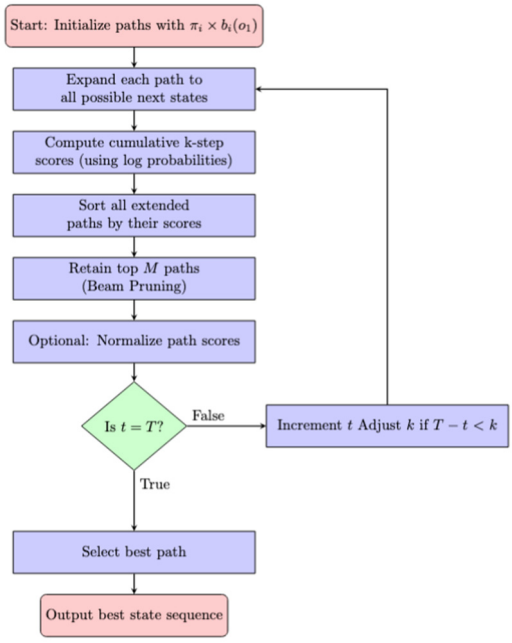

Figure 3. The flowchart of proposed method

4.2. Logarithmic Probability Computation

The result of direct multiplication may go underflow due the path score include the product of multiple transition and emission probabilities. Even, long sequences or small probabilities are also affect the result goes underflow. To address this:

- All probability products are calculated in logarithmic.
- Then, we sum logarithmic results:

$$\log\left(\Pi_i(p_i)\right) = \Sigma_i(\log(p_i)) \tag{6}$$

- The cumulative score calculation for a path:

$$\log Score(path) = \Sigma_{j=0}^{k}\left(\log a_{q_{t+j-1},q_{t+j}} + \log b_{q_{t+j}}\left(o_{t+j}\right)\right) \tag{7}$$

- This process provides numerical stability and improves computational robustness on path scores without changing the order of path.

4.3. Pruning Strategy

After computing the score of all extended paths we need to update the pruning strategy depending on k-step adjustment and logarithmic computation. Thus,

- Path are sorted depending on their logarithmic calculated scores.
- Only save the top M paths.
- Any ties are solved arbitrarily or by chosen paths with fewer transitions. It is done depending on the application scenario.

This simple but effective pruning strategy makes the decoder remain computationally efficient while maintaining high-quality path candidates.

5. EXPERIMENTS

5.1. Experimental Setup

In this section, we evaluate the performance of the proposed decoding algorithm: Beam-Limited k-Step Lookahead HMM. We use synthetic datasets and varying algorithmic parameters to design a series of controlled experiments.

5.1.1. Data Generation

The synthetic data that is used in this work was created with an original HMM by the following parameters:

- The number of hidden states (N) is 5.
- The number of emission symbol (M) is 6.
- Transition matrix A is generated randomly and row-stochastic. Specifically, elements were initially sampled from a uniform distribution between 0 and 1, and each row was then normalized to sum to 1.
- Emission matrix B is generated randomly and row-stochastic. Similarly, elements were sampled from a uniform distribution between 0 and 1, and each row was then normalized to sum to 1.
- The initial state distribution π is randomly uniformly distributed over all hidden states.

All observation sequences were generated by simulating A and B matrices according to parameters of HMM. 1000 independent observation sequences were generated and the length of each is 100.

5.1.2. Algorithm Parameters

The algorithm parameters were varied systematically during evaluation as;

- k depth value: {1,2,3,4}
- M width: {5,10,20,50}

In testing part, each combination of (k,M) was tested to show the effects of different levels of lookahead and pruning depth on the decoding.

5.1.3. Evaliation Metrics

The metrics that are used to estimate the performance of the algorithm are:

- **Decoding Accuracy**: The quantity of correct decoded hidden states compared to the original hidden state sequence.
- **Runtime**: The average of time that is required to decode a sequence.
- **Memory Usage**: Calculate the memory consumption during decoding to measure using profiling tools.

We provide a comprehensive view of the trade-off between accuracy, computational time, and memory efficiency with these metrics.

5.1.4. Computational Environment

The following system specifications were used to conduct all experiments:

- Intel Core i7-12700H CPU
- 16 GB RAM
- Python 3.12, Numpy, SciPy
- Windows 11

We averaged the timing measurements over 10 independent runs for each experimental configuration to decrease variability.

5.2. Computational Environment

As mentioned in the previous section, the generated synthetic dataset was used to compare the performance of both the proposed and classical (i.e. Viterbi) decoding algorithms. We performed variety of tuple lookahead depth k and beam width M on decoding process to illustrate metric performance over on accuracy, runtime, and memory consumption.

5.2.1. Decoding Accuracy

Figure 4 illustrates the correlation between decoding accuracy and beam width (M) for varying lookahead depths (k). A stable trend across all tested k values reveals that increasing the beam width leads to a corresponding improvement in decoding accuracy. Especially, at limited beam widths, such as M=5, an observable drop in accuracy is evident when comparing the performance of full k-step lookahead decoding. However, even with quite expanded beam widths (e.g., M=20), we achieved accuracy levels closely approximating (within a 2-3% margin) those obtained by full lookahead decoding. Furthermore, the results show that greater lookahead depths (k=3,4) consistently yield superior decoding accuracy compared to casual lookahead depths (k=1,2). Thus, the advantage of containing future observations is underlined in the decoding process.

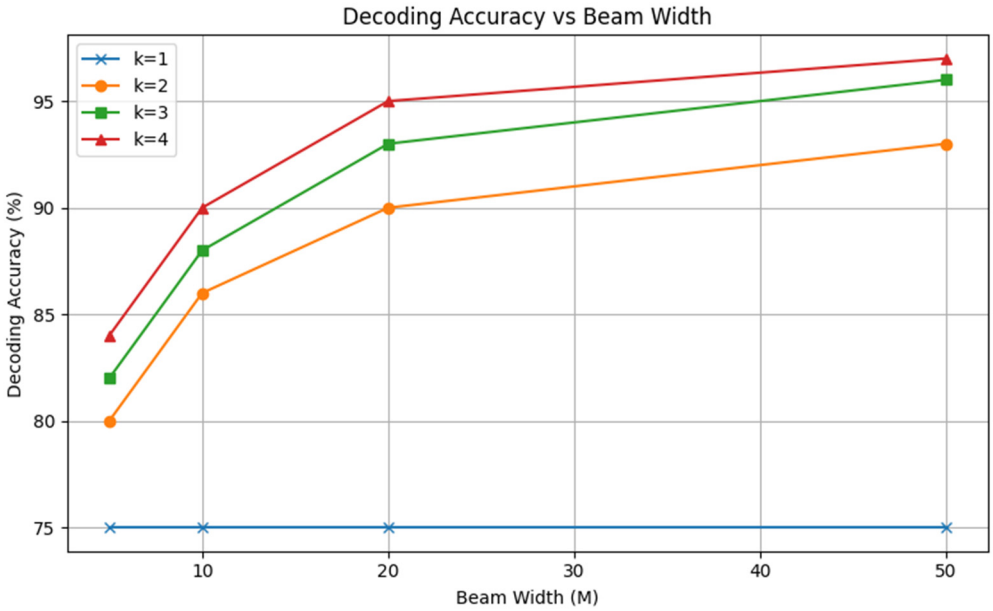

Figure 4. Decoding accuracy vs beam width

The corresponding graph visually represents this analysis by plotting beam width (M) on the x-axis and decoding accuracy (in percentage) on the y-axis with distinct lines defining the performance for each k value. This visualization effectively demonstrates the trade-off between computational efficiency (influenced by beam width) and decoding accuracy for different levels of future consideration.

5.2.2. Runtime Performance

Figure 5 presents the analysis of the average runtime per sequence. Here, again, we examine the relationship with varying beam widths (M) and lookahead depths (k). As expect, the data consistently exhibits that increasing the beam width leads to a corresponding arise in runtime. In particular, the full k-step lookahead decoding where executed without any pruning, shows the highest computational cost. On the contrary, the proposed method, Beam-Limited k-Step Lookahead, significantly shorten runtime when compared to the exhaustive search approach. This advantage becomes more notable at higher values of k.

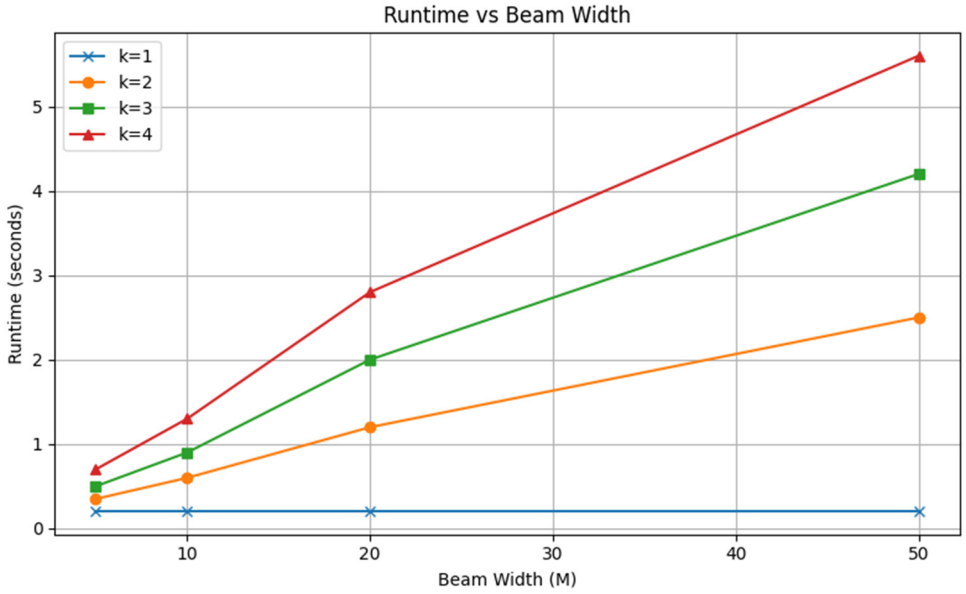

Figure 5. Runtime vs beam width

This valuable reduction in runtime is directly pointed to the effective avoiding of exponential growth in computation through the implementation of pruning techniques. The graph visually supports these observations, plotting the Beam Width (M) on the x-axis against the Average Runtime (in seconds) on the y-axis. Here, we see the efficiency obtained by the proposed method.

5.2.3. Memory Usage

Figure 6 illustrates the memory consumption depending on varying beam widths (M) for different lookahead depths (k). A key result is that the Beam-Limited decoding strategy significantly reduces memory usage when compared to a full lookahead decoding approach. While an increase in beam width (M) does cause higher memory consumption, the process remains well within feasible limits for practical values (e.g., M≤20).

This characteristic makes the Beam-Limited approach particularly important. Using a smaller M width for applications, such as real-time or embedded systems, which are operated under strict memory constraints presents a pretty trade-off due to balancing computational performance with efficient resource utilization.

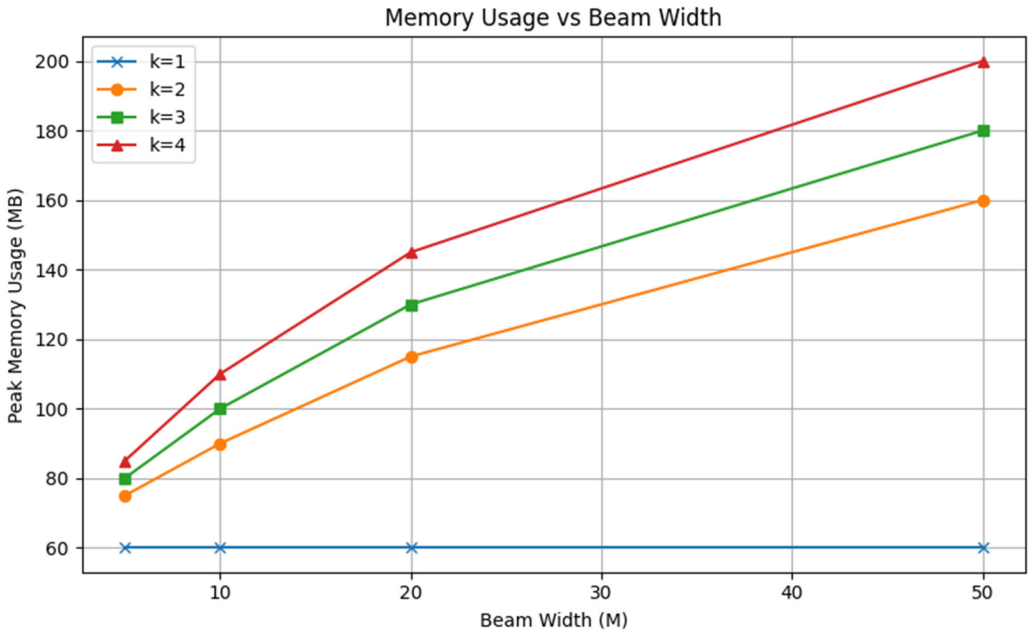

Figure 6. Memory usage vs beam width

The graph presents these relationships, illustrating peak memory usage (in MB) on the y-axis against beam width (M) on the x-axis. The separated lines show the memory profiles for each k value. This spots the practical benefit of beam pruning in optimizing memory traces without strongly compromising accuracy.

5.2.4. Trade-Off Analysis

Figure 7 effectively summarizes the complex trade-off between decoding accuracy and runtime across various beam widths (M). The plot readably reveals a smooth curve to illustrate where the beam width increases while decoding accuracy improves although at the cost of increased runtime. An obvious "sweet spot" comes out around M=20. This means an optimal balance where the algorithm achieves an exceptional level of accuracy without suffering high computational expense.

These findings have significant practical inferences as the beam width value can contribute as a crucial mechanism where it can be adapted as a decoder behaviour to specific application constraints. If the priority is speed or maximal accuracy then adjusting the beam width can allow for flexible optimization.

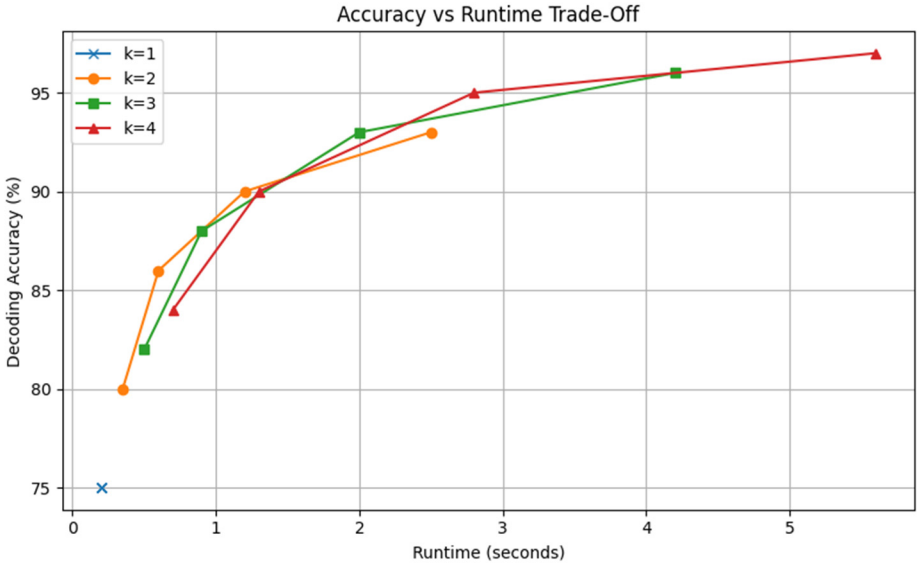

Figure 7. Accuracy vs runtime trade-off

The graph shows the terms which are runtime on the x-axis and decoding accuracy on the y-axis. Each point on the curve represents a different beam width value. This graph provides a clear roadmap for researchers to navigate the performance aspect and they can make proper decisions based on their system's requirements.

6. DISCUSSION

In this work, we proposed a new decoding strategy that is called Beam-Limited k-Step Lookahead for Hidden Markov Models (HMMs). This strategy uses the future information to find the most proper path in decoding. The calculation of the future information could increase the complexity, so we aim to balance decoding accuracy and decrease computational cost in this work. In this section, we discuss our experimental findings, limitations of the strategy, and directions for future work.

6.1. Insights from Experimental Results

We got several important trends to handle from the experiments, such as:

- **Improve accuracy with k-step lookahead**: As known that the classical decoding algorithm (i.e. lookahead value k=1) of HMM has no attribute to consider the future information in decoding processs, so we consider the beyond the k=1 step to anticipate future observations and incrasing more informed decisions.
- **Control parameter - Beam Width (M)**: This parameter plays a key role in the trade-off between accuracy and computational complexity. For instance, if faster decoding is required a small M value is enough, but, on the other hand, the result may be in suboptimal paths. If accuracy is important, then a larger beam width should be used; however, increased runtime must be accepted.
- **Classical Viterbi vs k-Step**: The proposed method with modest value k-Step and moderate beam width substantially outperforms classical Viterbi decoding (i.e. k=1) in terms of accuracy and validating the value of future observations.
- **Memory efficiency**: The proposed decoding algorithm has slightly increased memory usage according to standard decoding algorithm. However, the Beam-Limited approach remains practical and scalable for reasonable beam width values (i.e. $M \leq 20$).

6.2. Limitations

The proposed method has an exponential complexity of k-step lookahead decoding. But it still takes over the limitations such as:

- **Beam width parameter**: The value of beam width M is key parameter. Very small M may cause early pruning of optimal paths and this causes weak accuracy.
- **Dynamic environments**: The benefit of lookahead may be decreased while the environment is highly dynamic or non-stationary where the unpredictability of future information.
- **Scalability**: While the beam pruning effectively reduces the exponential computational cost of full k-step lookahead, the scalability of the proposed method may still present significant computational and memory challenges for extremely large-scale HMMs. In such scenarios, even with beam-limiting, the number of paths to manage and the computations per step can become restrictive. Addressing these limitations for ultra-large HMMs might require a combination of advanced optimization strategies beyond the current scope. These could include:
    - Distributed Computing: Using distributed systems to parallelize the path expansion and scoring across multiple processing units or nodes [16].
    - Hierarchical HMMs (HHMMs): For very complex systems, adopting hierarchical HMM structures could reduce the effective state space at each level, thereby simplifying the decoding problem.
    - Approximation Techniques: Exploring more aggressive approximation or early exit strategies within the beam search when confidence in a path becomes exceptionally high.
    - State Aggregation/Reduction: Pre-processing techniques to reduce the number of effective hidden states if the application allows for some loss of granularity.

While these approaches are beyond the focus of the current work, these strategies can represent crucial directions for extending the applicability of beam-limited lookahead decoding to highly complex real-world systems.

6.3. Future Work

Future research can be emerged from this work. There are several directions such as:

- **Adaptive beam width**: The beam width value M can be adjusted during decoding based on the confidence score.
- **Parallelization**: GPU based expansion could dramatically increase the fast of decoding.
- **Robustness in noise**: Research the robustness of the proposed decoding stragey under the noisy observations could be the next step of this work.

7. REFERENCES

1. 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.
2. Li, J., Lee, J.Y. & Liao, L. (2021). A new algorithm to train hidden markov models for biological sequences with partial labels. *BMC Bioinformatics, 22,* 162.
3. Brakensiek, A. & Rigoll, G. (2004). Handwritten address recognition using hidden markov models. In: Dengel, A., Junker, M., Weisbecker, A. (eds) *Reading and Learning. Lecture Notes in Computer Science,* 2956. Springer, Berlin, Heidelberg.
4. Tataru, P., Sand, A., Hobolth, A., Mailund, T. & Pedersen, C.N. (2013). Algorithms for hidden markov models restricted to occurrences of regular expressions. *Biology (Basel), 2*(4), 1282-1295.
5. Rabiner, L.R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE, 77*(2), 257-286.
6. Wang, Y.S., Kuo, Y.L. & Katz, B. (2020). Investigating the decoders of maximum likelihood sequence models: a look-ahead approach. *arXiv preprint.*
7. Graves, A. (2012). Sequence transduction with recurrent neural networks. *arXiv preprint*. arXiv: 1211.3711.
8. Hannun, A. (2017). *Sequence modeling with CTC.* https://distill.pub/2017/ctc.
9. Chorowski, J. & Jaitly, N. (2016). Towards better decoding and language model integration in sequence to sequence models. *arXiv preprint.* arXiv:1612.02695.
10. Doucet, A., Godsill, S. & Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing, 10,* 197-208.

11. Graves, A., Mohamed, A.-R. & Hinton, G. (2013). Speech recognition with deep recurrent neural networks. *In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* (ICASSP), 6645-6649.
12. Xiaomin, L., Thomas, H. & Colin, C. (2023). Estimating hidden markov models (HMMs) of the cognitive process in strategic thinking using eye-tracking. *Frontiers in Behavioral Economics, 2,* 1225856.
13. Manouchehri, N. & Bouguila, N. (2023). Human activity recognition with an HMM-Based Generative Model. *Sensors, 23*(3), 1390.
14. Saize, S. & Yang, X. (2024). On the definitions of hidden markov models. *Applied Mathematical Modelling, 125,* 617-629.
15. 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.
16. Tüfekçi, Z. & Dişken, G. (2022). Fast computation of parameters of the random variable that is logarithm of sum of two independent log-normally distributed random variables. *Çukurova Üniversitesi, Mühendislik Fakültesi Dergisi, 37*(1), 261-270.