Complexity of Comparing Hidden Markov Models

Rune B. Lyngsø[1,⋆] and Christian N. S. Pedersen[2,⋆⋆]

[1] Baskin Center for Computer Science and Engineering, University of California, Santa Cruz, CA 95064, U.S.A. E-mail: rlyngsoe@cse.ucsc.edu

[2] BiRC[⋆⋆⋆], Department of Computer Science, University of Aarhus, Ny Munkegade, DK-8000 Århus C, Denmark. E-mail: cstorm@daimi.au.dk.

Abstract The basic theory of hidden Markov models was developed and applied to problems in speech recognition in the late 1960's, and has since then been applied to numerous problems, e.g. biological sequence analysis. In this paper we consider the problem of computing the most likely string generated by a given model, and its implications on the complexity of comparing hidden Markov models. We show that computing the most likely string, and approximating its probability within any constant factor, is **NP**-hard, and establish the **NP**-hardness of comparing two hidden Markov models under the L_∞- and L_1-norms. We discuss the applicability of the technique used to other measures of distance between probability distributions. In particular we show that it cannot be used to prove **NP**-hardness of determining the Kullback-Leibler distance between the probability distributions of two hidden Markov models, or of comparing them under the L_k-norm for any fixed even integer k.

Keywords Hidden Markov Models, Consensus String, Distance Measures, NP Hardness

1 Introduction

A hidden Markov model is a description of a probability distribution over a set of strings. It is convenient to consider a hidden Markov model as a generative model in which a run generates a string with a certain probability. A run starts in a special start-state, and continues by following a first order Markov chain of states, called the path, until a special end-state is reached. A symbol from a finite alphabet is emitted according to some probability distribution each time a non-silent state is entered. The theory of hidden Markov models was developed and applied to problems in speech recognition in the late 1960's and early 1970's. In [14], Rabiner gives a good overview of the theory of hidden Markov models and its applications to problems in speech recognition. Hidden Markov models

⋆ Supported by grants from Carlsbergfondet and the Program in Mathematics and Molecular Biology

⋆⋆ Partially supported by the IST Programme of the EU under contract number IST-1999-14186 (ALCOM-FT)

⋆⋆⋆ Bioinformatics Research Center, www.birc.dk, funded by Aarhus University Research Foundation

are also applied in other areas than speech recognition. One prominent example of this is the area of computational biology where they have found many applications, e.g. modeling of DNA sequences [5], protein secondary structure prediction [2], gene finding [11], recognition of transmembrane proteins [16], and characterization of biological sequence families by modeling how the biological sequences relate by substitutions, insertions, and deletions to the consensus sequence of the family [12].

Applications of hidden Markov models are often based on two fundamental questions. Given a hidden Markov model and a string we might want to determine the probability of the string under the model, i.e. determine the probability that the model has generated the string. This can be used for *classification* of the string as either belonging to the family of strings represented by the model or not. Or we might want to determine the most likely path of states through the model that generates the string. This can be used for *annotating* the string with states from the model. Dynamic programming algorithms for solving these two problems are described in e.g. [14].

In this paper we consider the problem of determining the most likely string generated by a given hidden Markov model, i.e. of determining the *consensus* string of the model, and its implications on the problem of comparing hidden Markov models. We show that in polynomial time we cannot for any $\epsilon > 0$ approximate the probability of the most likely string under a hidden Markov model with n states within a factor of $n^{1/4-\epsilon}$ unless $\mathbf{P} = \mathbf{NP}$, and within a factor $n^{1/2-\epsilon}$ unless $\mathbf{ZPP} = \mathbf{NP}$. The hardness results hold even if we restrict the hidden Markov model to a model without silent states generating only strings over a binary alphabet. The problem of determining the consensus string of a hidden Markov model has not been addressed previously in the literature. However, it is useful for studying the hardness of comparing the probability distributions given by two hidden Markov models. Comparing two hidden Markov models is an interesting theoretical problem with practical applications as well, for example by comparing two profile hidden Markov models, e.g. from the Pfam protein families database [3], we compare entire sequence families instead of just individual members. In [13] we present methods for comparing hidden Markov models, and describe how to compute the Euclidean distance (the L_2-distance) between two models in polynomial time.

Using the hardness of determining the consensus string, we show that comparing two hidden Markov models under the L_∞-norm is hard, and construct a system of linear equations which links the probability of the consensus string of the model constructed to prove the hardness of determining the consensus string to the problem of comparing two hidden Markov models under any L_k-norm with $k \in \mathbf{R}_+$. We utilize this link to prove the hardness of comparing two hidden Markov models under the L_1-norm but show that it cannot be used to establish the hardness of comparing two hidden Markov models under the L_{2k}-norm for any $k \in \mathbf{N}$. The L_1-distance between two probability distributions defined on the same set, e.g. two probability distributions given by two hidden Markov models over the same alphabet, is of special interest because it equals twice the *variation distance* between the two distributions. The variation distance describes the maximum numerical difference between the probability of any set of events under the two distributions, see e.g. [6], and is a commonly used measure between probability distributions. Comparing probability distributions by L_k-distances is a well-studied problem, see e.g. [4,8,9] for algorithms for comparing probability distributions over finite sets.

The rest of the paper is organized as follows. In Sect. 2 we discuss hidden Markov models in more detail. In Sect. 3 we show that computing the most likely string, and approximating its probability within any constant factor, is **NP**-hard. In Sect. 4 we consider the general problem of comparing hidden Markov models, and show that comparison under the L_∞- and L_1-norms is **NP**-hard. In Sect. 5 we summarize the status of the tractability of comparing hidden Markov models by various well-known distances.

2 Hidden Markov Models

Let M be a hidden Markov model that generates strings over some finite alphabet Σ with probability distribution P_M, i.e. $P_M(s)$ denotes the probability of $s \in \Sigma^*$ under model M. Like a classical Markov model, a hidden Markov model consists of a set of interconnected states. We use $a^M_{q,q'}$ to denote the probability of a transition from state q to state q' in model M. These probabilities are called *state transition probabilities*. The transition structure of a hidden Markov model is often shown as a directed graph with a node for each state, and an edge between two nodes if the corresponding state transition probability is non-zero. Unlike a classical Markov model, a state in a hidden Markov model can emit a symbol according to a local probability distribution over all possible symbols. We use $e^M_{q,\sigma}$ to denote the probability of emitting symbol $\sigma \in \Sigma$ in state q in model M. These probabilities are called *symbol emission probabilities*. A state without symbol emission probabilities is called a *silent* state.

It is convenient to consider a hidden Markov model as a generative model in which a *run* generates a string. A run of a hidden Markov model begins in a special start-state and continues from state to state according to the state transition probabilities until a special end-state is reached. Each time a non-silent state is entered, a symbol is emitted according to the symbol emission probabilities of the state. In the exposition of our methods we also use the notion of a *partial run* which is a run that does not necessarily start in the special start-state or end in the special end-state. We refer to the Markovian sequence of states in a run as the *path* followed by the run. The string generated by a run is the concatenation of the symbols emitted along its path. The name "*hidden* Markov model" comes from the fact that the Markovian sequence of states followed by a run, the path, is hidden while only the emitted symbols, the generated string, is observable.

The probability $P_M(\pi)$ of following a path $\pi = (\pi_0, \pi_1, \ldots, \pi_k)$ in model M is given by the state transition probabilities as

$$P_M(\pi) = \prod_{i=1}^{k} a^M_{\pi_{i-1},\pi_i}. \tag{1}$$

The probability $P_M(\pi, s)$ of following a path $\pi = (\pi_0, \pi_1, \ldots, \pi_k)$ in model M and emitting string s depends on the subsequence $(\pi_{i_1}, \pi_{i_2}, \ldots, \pi_{i_l})$ of non-silent states on the path π. If the length of string $s = s_1 s_2 \cdots s_{l'}$ is different from the number of non-silent states along path π, the probability $P_M(\pi, s)$ is zero. Otherwise, the probability of following path π and emitting string s is

$$P_M(\pi, s) = P_M(\pi) \cdot P_M(s \mid \pi) = \prod_{i=1}^{k} a^M_{\pi_{i-1},\pi_i} \cdot \prod_{j=1}^{l} e^M_{\pi_{i_j},s_j}. \tag{2}$$

Since a run r of a hidden Markov model M is identified by a path π_r through the model and an emitted string s_r, we can define the probability of a run as

$$P_M(r) = P_M(\pi_r, s_r). \tag{3}$$

Finally, the probability $P_M(s)$ of model M generating a string s is the probability of following any path and emitting string s, that is

$$P_M(s) = \sum_{\pi} P_M(\pi, s). \tag{4}$$

3 Finding the Most Likely String

In this section we will establish the hardness of finding the most likely string of a hidden Markov model, a question one might naturally ask about hidden Markov models. We show that computing the probability of the most likely string generated by a hidden Markov model is **NP**-hard, but first we observe why there has to exist a most likely string: Let M be a hidden Markov model in which a run r emits a string s, i.e. $P_M(s) = \delta$ for some $\delta > 0$. Since $\sum_{s \in \Sigma^*} P_M(s) = 1$, there can be at most $1/\epsilon$ strings s' where $P_M(s') \geq \delta$. This implies that there cannot exist an infinte series of strings with increasing probabilities all greater than δ. Hence, there has to exist a most likely string.

We will show the hardness of computing the probability of the most likely string by a reduction from MaxClique, the problem of computing the size of the maximum clique in an undirected graph. The proposed reduction essentially preserves approximations. Hence, the approximation hardness results of [7, 10] for MaxClique can be translated into approximation hardness results for computing the probability of the most likely string. Since the probability of a given string can be computed in polynomial time, cf. [14], the result also implies that finding the most likely string, and not only its probability, is **NP**-hard. We start with a lemma describing how to construct a hidden Markov model that by its probability distribution over finite strings captures the clique sizes of a graph.

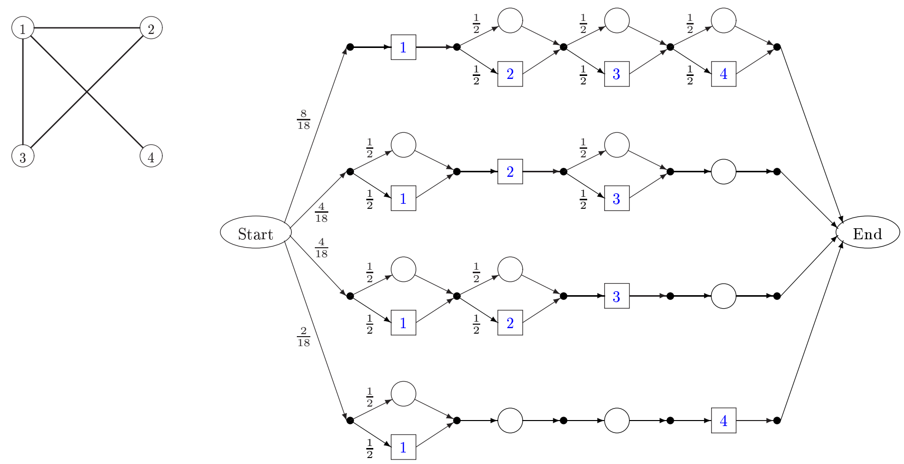

Figure 1. A graph, $G = (\{1,2,3,4\}, \{\{1,2\},\{1,3\},\{1,4\},\{2,3\}\})$, of four nodes, and the hidden Markov model M_G constructed from it by the construction of Lemma 1. The square states are the non-silent states $n_{u,v}$ with the symbol they emit with probability 1 written inside. The large, hollow circular states are the silent states $s_{u,v}$ while the small, black circular states are the silent states $i_{u,v}$.

Lemma 1. *For any graph $G = (V, E)$ we can in polynomial time construct a hidden Markov model M_G generating strings over the alphabet $\Sigma = V$ such that $\exists s \in \Sigma^* : P_{M_G}(s) = k/\gamma_G$ if and only if G has a clique of size k, where $\gamma_G = \sum_{v \in V} 2^{\deg(v)}$ and $\deg(v)$ is the degree of v in G.*

Proof. For simplicity, we will assume that $V = \{1, 2, \ldots, |V|\}$. The basic idea of the construction of M_G, illustrated in Fig. 1, is for each node v in G to construct a submodel with $2^{\deg(v)}$ possible paths of equal probability. Each of these paths generate one of the $2^{\deg(v)}$ ordered sequences of nodes where each node occurs at most once, v occurs exactly once, and all other occurring nodes are connected by an edge to v. We can construct these submodels simply by having a state emitting each of the possible symbols that can occur in the sequence with probability 1, and then choose to either enter or skip these states in turn with equal probability 1/2. By choosing the submodel of v with probability $2^{\deg(v)}/\gamma_G$ in the aggregate model consisting of all the submodels, all paths in the aggregate model will have probability $1/\gamma_G$. Thus, the probability of a string will be k/γ_G where k is the number of submodels that can generate it. Hence, the probability of a string "counts" the number of submodels that can generate it.

Formally, the model M_G consists of a start-state *start*, an end-state *end*, a set of silent states S (not essential for the construction but serves to make the structure of the model more clear), and a set of non-silent states N, where

$$S = \{s_{u,v} \mid u, v \in V, u \neq v\} \cup \{i_{u,v} \mid u \in V, v \in V \cup \{0\}\},$$
$$N = \{n_{u,v} \mid u, v \in V, \{u, v\} \in E \vee u = v\}.$$

The non-zero transition probabilities are

$$
\begin{aligned}
a_{\text{start},i_{u,0}} &= 2^{\deg(u)}/\gamma_G \\
a_{i_{u,v-1},n_{u,v}} &= \begin{cases} 1 & \text{if } u = v \\ 1/2 & \text{if } \{u,v\} \in E \\ \textit{undefined} & \text{otherwise } (n_{u,v} \text{ is not a state in } M_G) \end{cases} \\
a_{i_{u,v-1},s_{u,v}} &= \begin{cases} \textit{undefined} & \text{if } u = v \ (s_{u,v} \text{ is not a state in } M_G) \\ 1/2 & \text{if } \{u,v\} \in E \\ 1 & \text{otherwise} \end{cases}
\end{aligned}
$$

where $u, v \in V$ and

$$a_{n_{u,v},i_{u,v}},\ a_{s_{u,v},i_{u,v}},\ a_{i_{u,|V|},\text{end}} = 1$$

whenever they are well defined, i.e. whenever both states are in M_G. The non-zero emission probabilities are

$$e_{n_{u,v},v} = 1$$

for all non-silent states $n_{u,v} \in N$. This model can evidently be constructed in polynomial time.

We still need to prove the connection between clique sizes in G and string probabilities in M_G. We first observe that any run through M_G has probability $1/\gamma_G$, hence the probability of any string must be k/γ_G for some k. Now assume that there is a string s with $P_{M_G}(s) = k/\gamma_G$. Hence, s can be generated by the submodels of k nodes. Let $\{u_i\}_{1 \le i \le k}$ be the set of nodes whose submodels can generate s. We claim that $\{u_i\}_{1 \le i \le k}$ must be a clique in G. First, all u_i for $1 \le i \le k$ must occur in s, as all sequences generated by the submodel of node u_i contains u_i. Secondly, each u_i must be connected by an edge to all nodes, apart from itself, occurring in s. Hence, all pairs of nodes $u, v \in \{u_i\}_{1 \le i \le k}$ are connected by an edge.

Conversely, assume that $C = \{u_i\}_{1 \le i \le k}$, where $i < j \Leftrightarrow u_i < u_j$, is a clique in G. We claim that $P_{M_G}(u_1 u_2 \dots u_k) = k/\gamma_G$. First, $u_1 u_2 \dots u_k$ can be generated by the submodel of any node $u_i \in C$ as it contains u_i and as u_i is connected by an edge to any other node occurring in $u_1 u_2 \dots u_k$. Secondly, $u_1 u_2 \dots u_k$ cannot be generated by the submodel of any node $v \notin C$ as it does not contain v. Hence, $u_1 u_2 \dots u_k$ can be generated by precisely k submodels and thus has probability k/γ_G in M_G. □

The model M_G constructed in Lemma 1 satisfies that the end-state can be reached from any other state with non-zero probability, i.e. there does not exist a state p where $a^{M_G}_{p,p} = 1$. The next two lemmata simplify the model. More precisely, we show that only non-silent states and a binary alphabet is necessary.

Lemma 2. *Lemma 1 still holds if we restrict the model to generate strings over a binary alphabet.*

Proof. In the proof of Lemma 1 we used an alphabet $\Sigma = V$. We can encode this alphabet in binary such that there to each $\sigma \in \Sigma$ corresponds a unique string in $\{0,1\}^{\lceil \log |V| \rceil}$. Each non-silent $n_{u,v}$ state in M_G is now replaced with a sequence of $\lceil \log |V| \rceil$ non-silent states, where the i'th state emits the i'th bit in the binary encoding of v and has probability 1 for the transition to the $i+1$'st state. □

Lemma 3. *Given a hidden Markov model M, where the end-state can be reached from any other state with non-zero probability, we can construct a hidden Markov model M' with no silent states and $P_M = P_{M'}$.*

Proof. We prove the lemma by describing the simple procedure of removing one silent state, thus transforming M into a model M'' with one less silent state. This procedure can then be applied to all the silent states of M in turn. Let p be a silent state, i.e. it does not emit any symbols. Hence, the only effect p has is to allow going from one state q via p to another state r without emitting any symbols on the way. But this might as well be done with a direct transition. When eliminating p we thus have to update all other transition probabilities as

$$a_{q,r}^{M''} = a_{q,r}^{M} + a_{q,p}^{M} \cdot a_{p,r}^{M} / (1 - a_{p,p}^{M})$$

if $a_{p,p}^{M} < 1$. Since the end-state by assumption can be reached from any state with non-zero probability, we can ignore the case $a_{p,p}^{M} = 1$. Hence, the described update yields a new model M'' with $P_{M''} = P_M$ but one less silent state. □

Corollary 1. *Lemma 2 still holds if we restrict the model to not having any silent states.*

We are now ready to present the main result of this section, that the probability of the most likely string of a hidden Markov model is hard to approximate.

Proposition 1. *Let M be a hidden Markov model with n states generating strings over an alphabet Σ, where $|\Sigma| \geq 2$. For any $\epsilon > 0$ we cannot in polynomial time*

- *approximate the probability of the most likely string under M within a factor of $n^{1/4-\epsilon}$ unless* $\mathbf{P} = \mathbf{NP}$
- *approximate the probability of the most likely string under M within a factor of $n^{1/2-\epsilon}$ unless* $\mathbf{ZPP} = \mathbf{NP}$

Proof. In [10] it is proved that we in polynomial time cannot approximate the largest clique of a graph $G = (V, E)$ within a factor of $|V|^{1/2-\epsilon}$ unless $\mathbf{P} = \mathbf{NP}$ and cannot approximate it within $|V|^{1-\epsilon}$ unless $\mathbf{ZPP} = \mathbf{NP}$. By Lemma 3 and an inspection of the proofs of Lemmata 1 and 3 we can construct a model M_G with less than $|E|\lceil \log |V| \rceil$ states such that the most likely string in M_G has probability k/γ_G if and only if the largest clique of G is of size k. Assume we can approximate $\max\{P_{M_G}(s) \mid s \in \Sigma^*\}$ within a factor of n^c in polynomial time, i.e. that we can find $p \leq k/\gamma_G$ such that $p \cdot n^c \geq \max\{P_{M_G}(s) \mid s \in \Sigma^*\} = k/\gamma_G$. As $|E| \cdot \lceil \log |V| \rceil > n$ it follows that $p \cdot \gamma_G \cdot (|E| \cdot \lceil \log |V| \rceil)^c \geq k$. Furthermore, $|E| \cdot \lceil \log |V| \rceil = \mathrm{o}(|V|^{2+\delta})$ for any $\delta > 0$. Hence, we can approximate the size of the largest clique in G within a factor $|V|^{2c+2\delta}$. The result now follows by choosing $c = \frac{1}{2}(1 - \epsilon - 2\delta)$ and $c = \frac{1}{2}(\frac{1}{2} - \epsilon - 2\delta)$, respectively. □

4 Comparing Hidden Markov Models

What the most probable string of a hidden Markov model is, is a very natural question to ask. However, it does not seem to be of high practical importance. At least, it does not appear that any previous work has been concerned with this problem. Indeed, our main motivation for studying the problem of finding the most probable string was that we can use the hardness of this problem – and some specific details from the reduction proving the hardness – as basis for proving the results of this section, developing hardness results for comparing the probability distributions of two hidden Markov models under L_k-norms.

Usually, hidden Markov models are considered tools for analyzing data. But we may also simultaneously view them as a compact representation of a probability distribution over finite sequences. E.g. in computational biology, a hidden Markov model for classifying sequences can be viewed as a representation of the family of sequences belonging to the class. Hence, the hidden Markov model itself can be considered data. As a common task in computational biology is that of comparing data, it is thus interesting, both from a theoretical and a practical viewpoint, to investigate how to compare two hidden Markov models, i.e. how to compare the probability distributions described by the two models.

The L_k-norm between two models M and M' over the same alphabet Σ is $\|P_M - P_{M'}\|_k = \sum_{s\in\Sigma^*} \sqrt[k]{(|P_M(s) - P_{M'}(s)|^k)}$, and the L_∞-norm is $\|P_M - P_{M'}\|_\infty = \max_{s\in\Sigma^*} |P_M(s) - P_{M'}(s)|$. That L_k-norms are well-defined for hidden Markov models, even over infinite countable sets, follows from the comparison criteria for series. For the L_∞-norm the well-definedness can be argued similar to the well-definedness of a most likely string. In [13] we describe how to compute the L_2-distance between two models in polynomial time. In this section we will extend this result by proving that if $\mathbf{P} \neq \mathbf{NP}$ then neither the L_∞-norm nor the L_1-norm can be computed in polynomial time, but that the L_k-norm can be computed in polynomial time for k any fixed even integer. We conjecture these to be the only efficiently computable L_k-norms, i.e. the L_k-norm between the probability distributions of two hidden Markov models can be computed in polynomial time if and only if k is fixed and an even integer. We start with the L_∞-norm. This is closely linked to the probability of the most likely string by the following lemma.

Lemma 4. *Given a hidden Markov model M that generates finite strings over an alphabet Σ, we can construct another hidden Markov model M' that generates finite strings over an alphabet $\Sigma \cup \{\$\}$ such that*

$$\max\{P_M(s) \mid s \in \Sigma^*\} = \|P_M - P_{M'}\|_\infty .$$

Proof. The model M' we construct will be an almost exact copy of M. The only difference is an extra state that emits a special symbol $\$ \notin \Sigma$ just prior to entering the end-state. Hence, we add a new state q to M', as compared to M, with $e^{M'}_{q,\sigma} = \delta(\sigma, \$)$ and $a^{M'}_{q,p} = \delta(p, \text{end})$. The transition probabilities are updated to go to the new state q instead of to the end-state, i.e. $a^{M'}_{p,q} = a^{M}_{p,\text{end}}$ and $a^{M'}_{p,\text{end}} = 0$ for all $p \neq q, \text{end}$; otherwise emission as well as transition probabilities are the same for M and M'. With this construction the sets of sequences emitted by M and M' are disjoint and $P_M(s) = P_{M'}(s\$)$ for all s. Hence,

$$\begin{aligned}\|P_M - P_{M'}\|_\infty &= \max\left\{|P_M(s) - P_{M'}(s)| \mid s \in (\Sigma \cup \{\$\})^*\right\} \\ &= \max\left\{P_M(s) \mid s \in \Sigma^*\right\}.\end{aligned}$$

□

Corollary 2. *The hardness results of Proposition 1 for determining the most likely string of a hidden Markov model transfers directly to the problem of comparing the probability distributions of two hidden Markov models M and M' under the L_∞-norm.*

In general, the L_∞-norm between two probability distributions, P and Q, defined on the same set, Ω, is defined as $\|P - Q\|_\infty = \max_{s\in\Omega} |P(s) - Q(s)|$. Hence, it measures the largest difference in probabilities we can obtain for any possible single observation. Another, seemingly similar, way to compare two probability distributions P and Q is by the *variation distance*, see e.g. [6], defined as $\|P - Q\| = \max_{\mathcal{A}\subseteq\Omega} |P(\mathcal{A}) - Q(\mathcal{A})|$. This measures the largest difference in probabilities we can obtain for any subset of observations. It is well-known that the variation distance between two probability distributions is half the L_1-distance between the two probability distributions, i.e. $\|P - Q\| = \frac{1}{2} \cdot \|P - Q\|_1$.

Proposition 2. *Comparing two hidden Markov models M and M' under the L_1-norm is **NP**-hard.*

Proof. The proof is again by a reduction from MaxClique. Or rather a reduction from the consensus string problem for the hidden Markov model M_G constructed from a graph $G = (V, E)$ in the previous section to establish the hardness of the consensus string problem. Recall that every string generated by M_G is a subsequence of $12\ldots|V|$ and have probability i/γ_G for some $i \in \{0, 1, \ldots, |V|\}$, where $\gamma_G = \sum_{v\in V} 2^{\deg(v)}$. Let a_i denote the number of subsequences of $12\ldots|V|$ which the hidden Markov model M_G generates with probability i/γ_G. If we know the maximum k such that $a_k \neq 0$, we can conclude that the probability of the most likely string under M_G is k/γ_G, and by the result of the previous section, that the maximum clique in G has size k.

To clarify our proof technique, we initially ignore the fact that the probabilities of all sequences generated by a model has to sum to 1. For any $x \in \mathbf{R}$ we can construct a model $M^x_{|V|}$ that assigns a uniform probability of x to all subsequences of $1\,2\ldots|V|$ and probability 0 to all other sequences. Comparing $M^{i/\gamma_G}_{|V|}$ to M_G under the L_1-norm we get

$$\|P_{M^{i/\gamma_G}_{|V|}} - P_{M_G}\|_1 = \sum_s \left|P_{M^{i/\gamma_G}_{|V|}}(s) - P_{M_G}(s)\right| = \sum_{j=0}^{|V|} a_j \frac{|i-j|}{\gamma_G}. \qquad (5)$$

Hence, comparing M_G with $M_{|V|}^{i/\gamma_G}$ under the L_1-norm for all $i = 0, 1, \ldots, |V|$ we obtain a system of linear equations

$$Ma = l \tag{6}$$

for determining the a_i, where M is the $(|V|+1) \times (|V|+1)$ matrix with entries $M_{ij} = |i-j|$, and l is the $(|V|+1)\times 1$ vector with entries $l_i = \gamma_G \| P_{M_{|V|}^{i/\gamma_G}} - P_{M_G} \|_1$. The matrix M is invertible with inverse

$$(M^{-1})_{ij} = \begin{cases} (1-|V|)/2|V| & \text{if } i = j = 1 \text{ or } i = j = |V|+1. \\ -1 & \text{if } 1 < i = j \leq |V|. \\ 1/2 & \text{if } j = i \pm 1. \\ 1/2|V| & \text{if } i = 1, j = |V|+1 \text{ or } i = |V|+1, j = 1. \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

Thus, knowing l we can compute a in time polynomial in the size of G, which in turn allows us to determine the size of the largest clique in G.

Let us now extend the above technique when keeping in mind that the probabilities of all sequences generated by a model have to sum to 1. When the probabilities of all sequences generated by a model are required to sum to 1, the only model $M_{|V|}$ with a uniform probability distribution over all subsequences of $1\,2 \ldots |V|$ and all other strings having probability 0 is $M_{|V|} = M_{|V|}^{2^{-|V|}}$.

However, if we embed a model M as a submodel of an aggregate model M' that chooses model M with probability x and chooses another submodel that always generates a string with a single symbol $\$ \notin \Sigma$ with probability $1-x$, we have scaled down the probabilities of all sequences originally generated by M by a factor x. Hence, for any $i = 0, 1, \ldots, |V|$ we can make $M_{|V|}$ generate all subsequences of $0\,1 \ldots |V|$ with probability equal to the probability of a string in M_G that can be generated by exactly i different paths in M_G. Either by rescaling $M_{|V|}$ by a factor $\frac{i/\gamma_G}{2^{-|V|}}$ if $i/\gamma_G \leq 2^{-|V|}$, or by rescaling M_G by a factor $\frac{2^{-|V|}}{i/\gamma_G}$ if $i/\gamma_G > 2^{-|V|}$. Denoting the rescaled models $M_{G,i}$ and $M_{|V|,i}$ we get

$$\begin{aligned} \|P_{M_{|V|,i}} - P_{M_{G,i}}\|_1 &= \left|P_{M_{|V|,i}}(\$) - P_{M_{G,i}}(\$)\right| + \sum_{s \neq \$} \left|P_{M_{|V|,i}}(s) - P_{M_{G,i}}(s)\right| \\ &= (1-b_i) + c_i \sum_{j=0}^{|V|} a_j |i-j| \end{aligned} \tag{8}$$

where $b_i = \min\left\{\frac{i/\gamma_G}{2^{-|V|}}, \frac{2^{-|V|}}{i/\gamma_G}\right\}$ and $c_i = \min\left\{\frac{1}{\gamma_G}, \frac{2^{-|V|}}{i}\right\}$. By comparing $M_{G,i}$ and $M_{|V|,i}$ under the L_1-norm for $i = 0, 1, \ldots, |V|$ we thus obtain a system of linear equations identical to (6), except for the vector l. In the system of linear equations based on comparing rescaled models, one observes from (8) that the entries of l are $l_i = \frac{1}{c_i}\left(\|P_{M_{|V|,i}} - P_{M_{G,i}}\|_1 + b_i - 1\right)$. But the system being solvable does not depend on the value of l, only on the matrix M. Hence, being able to compare two hidden Markov models under the L_1-norm in polynomial time would allow us to determine a and thus the size of the largest clique in a graph in polynomial time. □

In the proof of Proposition 2 we do not directly relate the L_1-distance between two models to the most likely string of one (or both) of the models. Hence, the approximation hardness is not preserved by the reduction, and the hardness result obtained is only for the computation of the exact L_1-distance. In the rest of the paper we examine the proof technique of Proposition 2 in more details.

Let P and Q be two probability distributions defined on a set Ω, and D be a distance between probability distributions defined as a sum of point-wise comparisons, i.e. $D(P,Q) = \sum_{s\in\Omega} d(P(s), Q(s))$. One can consider using the same technique as in the proof of Proposition 2 to prove it hard to compare two hidden Markov models under the measure D. We can set up a system of linear equations similar to (6), as the distance between $M_{|V|,i}$ and $M_{G,i}$ under D is

$$D\left(P_{M_{|V|,i}}, P_{M_{G,i}}\right) = (1-b_i) + \sum_{j=0}^{|V|} a_j \cdot d(c_i \cdot i, c_i \cdot j)\,. \tag{9}$$

Whether this allows us to deduce anything about the hardness of comparing two hidden Markov models under D depends on whether the resulting system of linear equation system, with matrix M defined by $M_{ij} = d(c_i \cdot i, c_i \cdot j)$, can be solved with sufficient precision to determine the maximum clique size of G in time polynomial in the size of G.

As an example we can consider the L_k-norm for an even integer k. In general, for the L_r-norm we have

$$\begin{aligned}\left(\|P_{M_{|V|,i}}, P_{M_{G,i}}\|_r\right)^r &= (1-b_i) + \sum_{j=0}^{|V|} a_j \left|c_i \cdot i - c_i \cdot j\right|^r \\ &= (1-b_i) + {c_i}^r \sum_{j=0}^{|V|} a_j \left|i-j\right|^r\,.\end{aligned} \tag{10}$$

Hence, when considering the L_k-norm for an even integer k, the matrix M of the resulting system of linear equations has entries $M_{ij} = |i-j|^k = (i-j)^k$. Since any $m \times m$ matrix A defined by $A_{ij} = (x_i - y_j)^k$ is singular if $m > k+1$ (a fact which can be proved e.g. using the fact that a polynomial of degree k is uniquely determined by its value in $k+1$ points, hence we can choose a set of values in $m > k+1$ points that does not agree with any polynomial of degree k), the system of linear equations defined by $M = (i-j)^k$, for an even integer k, cannot be solved for a unique solution if $|V| > k$, and we fail to prove the **NP**-hardness of comparing two hidden Markov models under the L_k-norm for an even integer k. This failure is no surprise because the algorithmic technique for comparing two hidden Markov models under the L_2-norm described in [13] can easily be extended to compare two hidden Markov models under the L_k-norm,

k a fixed even integer, in time $O(n^{3k})$ where n is the number of states in the two models. The algorithmic technique of [13] cannot be applied to the computation of any other L_r-norm than those where r is an even integer.

Based on the **NP**-hardness of comparing two models under the L_1-norm proved above, and that the L_r-norms where r is an even integer stands out as the only ones where the absolute value operation can be ignored, we conjecture that it is **NP**-hard to compare two models under any L_r-norm where r is not an even integer. This claim immediately follows, if we for a fixed r not an even integer can solve a linear equation system as in (6) with an $m \times m$ matrix M with entries $M_{ij} = |i - j|^r$ in time polynomial in m.

Similarly to the L_k-norm for k an even integer, we can rule out the Kullback-Leibler divergence, or relative entropy, as a case where the technique used in the proof of Proposition 2 is useful. Here the matrix M of the linear equation system can be defined by $M_{ij} = i \log \frac{i}{j} = i(\log i - \log j)$. Setting $x_i = \log i$ and $y_j = \log j$, by the above discussion the $m \times m$ matrix M' defined by $M'_{ij} = \log i - \log j$ is singular if $m > 2$. Hence, so is any matrix obtained by multiplying one or more rows of M' by scalars.

5 Conclusion

When choosing how to compare two probability distributions, one important consideration is how efficiently a given distance can be computed. In [1] a number of commonly used measures of distances between probability distributions are listed: the χ^2, variation, quadratic, Hellinger and Kullback-Leibler distances. In [13] we show how to compute the quadratic distance between the probability distributions of two hidden Markov models in polynomial time. The Hellinger distance – the distance in Euclidean space between the two probability distributions after they have been normalized to 1 – can also be computed in polynomial time based on the methods in [13], as we show how to compute the angle in Euclidean space between the two probability distributions when interpreted as infinite dimensional vectors.

In this paper we have proved that the variation distance is **NP**-hard to compute, and furthermore that the L_∞-distance is hard even to approximate. Furthermore, we briefly mentioned that the distance based on the L_k-norm can be computed in polynomial time if k is an even integer. To our knowledge, the complexity of comparing the probability distributions of two hidden Markov models under the χ^2 distance, the Kullback-Leibler distance and the distance based on the L_r-norm for any r not 1, ∞, or an even integer remains an open problem, though a heuristic for approximating the Kullback-Leibler distance was proposed in [15].

Acknowledgements

The authors would like to thank Peter Bro Miltersen for bringing our attention to the problems discussed in this paper. We furthermore acknowledge the help of Ole Østerby and Kevin Karplus.

References

1. N. Abe and M. K. Warmuth. On the computational complexity of approximating distributions by probabilistic automata. *Machine Learning*, 9:205–260, 1992.
2. K. Asai, S. Hayamizu, and K. Handa. Prediction of protein secondary strucuture by the hidden markov model. *Computer Applications in the Biosciences (CABIOS)*, 9:141–146, 1993.
3. A. Bateman, E. Birney, R. Durbin, S. R. Eddy, K. L. Howe, and E. L. L. Sonnhammer. The Pfam protein families database. *Nucleic Acid Research*, 28:263–266, 2000.
4. T. Batu, L. Fortnow, R. Rubinfeld, W. D. Smith, and P. White. Testing that distributions are close. In *Proceedings of the 15th Annual Symposium on Theory of Computing (STOC)*, pages 259–269, 2000.
5. G. A. Churchill. Stochastic models for heterogeneous DNA sequences. *Bulletin of Mathematical Biology*, 51:79–94, 1989.
6. T. M. Cover and J. A. Thomas. *Elements of Information Theory*. John Wiley & Sons, New York, 1991.
7. L. Engebretsen and J. Holmerin. Clique is hard to approximate within $n^{(1-o(1))}$. In *Proceedings of the 27th International Colloquium om Automata, Languages and Programming (ICALP)*, volume 1853 of *Lecture Notes in Computer Science*, pages 2–12, 2000.
8. J. Feigenbaum, S. K. ad M. Strauss, and M. Viswanathan. An approximate l^1-difference algorithm for massive data streams. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 501–511, 1999.
9. J. Fong and M. Strauss. An approximate l^p-difference algorithm for massive data streams. In *Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, 2000.
10. J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182:105–142, 1999.
11. A. Krogh. Two methods for improving performance of an HMM and their application for gene finding. In *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 179–186, 1997.
12. A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler. Hidden markov models in computational biology: Applications to protein modeling. *Journal of Molecular Biology*, 235:1501–1531, 1994.
13. R. B. Lyngsø, C. N. S. Pedersen, and H. Nielsen. Metrics and similarity measures for hidden Markov models. In *Proceedings of the 7th International Conference on Intelligent Systems for Molecular Biology (ISMB)*, pages 178–186, 1999.
14. L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, 1989.
15. Y. Singer and M. K. Warmuth. Training algorithms for hidden Markov models using entropy based distance functions. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9 (NIPS)*, pages 641–647. Morgan Kaufmann Publishers, 1996.
16. E. L. L. Sonnhammer, G. von Heijne, and A. Krogh. A hidden Markov model for predicting transmembrane helices in protein sequences. In *Proceedings of the 6th International Conference on Intelligent Systems for Molecular Biology (ISMB)*, 1998.