Hidden Markov Models With Set-Valued Parameters

Denis Deratani Mauá[a,*], Alessandro Antonucci[b], Cassio Polpo de Campos[c]

[a] *Universidade de São Paulo, São Paulo, Brazil*
[b] *Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, Manno, Switzerland*
[c] *Queen's University Belfast, Belfast, UK*

Abstract

Hidden Markov models (HMMs) are widely used probabilistic models of sequential data. As with other probabilistic models, they require the specification of local conditional probability distributions, whose assessment can be too difficult and error-prone, especially when data are scarce or costly to acquire. The imprecise HMM (iHMM) generalizes HMMs by allowing the quantification to be done by sets of, instead of single, probability distributions. iHMMs have the ability to suspend judgment when there is not enough statistical evidence, and can serve as a sensitivity analysis tool for standard non-stationary HMMs. In this paper, we consider iHMMs under the strong independence interpretation, for which we develop efficient inference algorithms to address standard HMM usage such as the computation of likelihoods and most probable explanations, as well as performing filtering and predictive inference. Experiments with real data show that iHMMs produce more reliable inferences without compromising the computational efficiency.

Keywords: hidden Markov models, imprecise probability, sensitivity analysis, credal networks, robust statistics.

*Corresponding author
Email addresses: denis.maua@usp.br (Denis Deratani Mauá), alessandro@idsia.ch (Alessandro Antonucci), c.decampos@qub.ac.uk (Cassio Polpo de Campos)

1. Introduction

Hidden Markov Models (HMMs) are popular probabilistic descriptions of paired sequences of states and observations [1], with applications in speech [1] and text processing [2], activity recognition [3] and computational biology [4], to name but a few. An HMM assumes that the states have been generated by a first-order Markov Chain process, while each observation is generated based only on the paired state. The specification of an HMM comprises an *initial state probability distribution*, which specifies the probability that the process originates in a given state, a *state transition probability distribution*, which specifies the probability that the process will transit from a given state to another, and a *symbol emission probability distribution*, which specifies the probability of observing a symbol conditional on a state.

In many domains, the transitions between consecutive hidden states and the relation between a hidden variable and the corresponding observation are affected by severe uncertainty. This is the case, for instance, when data are scarce [5], observations are missing not-at-random [6], and information is conflicting. In such cases, the use of probability distributions to represent uncertainty might be inadequate and lead to overly confident inferences [5, 7, 8].

Credal sets [9] are closed and convex sets of probability distributions that allow for a more general representation of uncertainty, including the situations just described. For instance, complete ignorance about a variable is represented as the credal set of all probability distributions on that variable, instead of the more common representation as a uniform probability distribution. The *imprecise (Multinomial) Dirichlet model* (IDM) learns credal sets from categorical data in a situation of near prior ignorance, providing a more reliable (although less informative) model of the underlying distribution than the more common Multinomial-Dirichlet model [10].

This paper presents efficient algorithms for inference with *imprecise hidden Markov models* (iHMMs), which allow the specification of a time- and state-discrete HMM with initial state, state transition and symbol emission credal sets in lieu of probability distributions. iHMMs provide a sound way to handle severe uncertainty, with two direct benefits. First, they allow us to suspend judgment when there is not sufficient statistical evidence to support a decision [11]. Second, they provide an efficient tool for performing sensitivity analysis [12] in standard non-stationary HMMs, allowing parameters to vary jointly, and in time.

In the rest of the paper, we review the related work (Section 2) and the basics about HMMs (Section 3) and iHMMs (Section 4); we then describe algorithms to deal with common uses such as comparing models according to the data likelihood (Section 5), predicting the current/next state given past observations (Section 6) and finding the most likely hidden state sequence for a given sequence of observations (Section 7). Experiments with speech and action recognition, text completion, and part-of-speech tagging (Section 8) provide evidence that iHMMs are indeed capable of making reliable decisions and evaluating the sensitivity of HMMs to the learning sample size. Conclusions and future work are described in Section 9.

2. Related Work

Bayesian networks are probabilistic models where conditional independences are represented by a graph whose nodes are identified with random variables [13]. HMMs are part of a special class of Bayesian networks (viz. tree-shaped Bayesian networks), one for which efficient inference algorithms are available. As with HMMs, Bayesian networks require uncertainty to be represented by conditional probability distributions. Credal networks [14] extend Bayesian networks to allow uncertainty to be modeled as credal sets. The iHMMs we discuss here are special cases of tree-shaped credal networks.

Drawing inferences from credal networks is a notoriously hard problem. Posterior inference is NP-hard already in tree-shaped credal networks, even when variables take on at most three values [15]; it is NP-hard also in polytree-shaped networks when there is no evidence, even if we allow (provably good) approximate results [16]. A few tractable cases appear in the literature. Fagiuoli and Zaffalon [17] developed a polynomial-time algorithm for polytree-shaped credal networks with binary variables. Zaffalon and Fagiuoli [18] described a method to compute unconditional posterior bounds in tree-shaped networks. Mauá, de Campos and Zaffalon [19] proved the existence of a fully polynomial-time approximation scheme for networks of bounded treewidth and bounded variable cardinality.[1]

[1] A fully polynomial-time approximation scheme is a family of algorithms parameterized by a rational $\epsilon > 0$, each returning, in time polynomial in the input and in $1/\epsilon$, a solution whose value is within a multiplicative factor of $(1 + \epsilon)$ of the optimum.

There has also been intense work on fast approximate algorithms (with no accuracy guarantees). The GL2U algorithm implements a message passing scheme similar to loopy belief propagation in Bayesian networks, that computes upper and lower probabilities in polynomial-time [20]. Other researchers have proposed the use of greedy heuristics [21, 22]. Recently, Antonucci et al. [23] developed an approximate method based on linear programming relaxations that was shown to outperform other approximate methods for marginal inference.

The algorithmic techniques discussed in the previous paragraphs deal with the interpretation of imprecision in the parameters known as strong independence. Strong independence, which we adopt in this work, assumes the existence of an ideal probability distribution which we cannot characterize for lack of resources. Epistemic irrelevance (or its symmetrical counterpart epistemic independence) makes no such claim, and allows for the possibility that there might not be any single probability distribution capable of representing our (uncertain) knowledge. De Cooman et al. [24] presented an efficient algorithm for single-query marginal inferences in tree-shaped credal networks under epistemic irrelevance. Their algorithm can be used to efficiently perform filtering (i.e., estimating the marginal probability of the future state given a sequence of observations). Recently, it was shown that filtering on iHMMs provides the same results whether one adopts strong independence or epistemic irrelevance [15, 25]. Hence, filtering is also polynomial-time computable under strong independence. We develop later an alternative algorithm for filtering in iHMMs under strong independence. Our algorithm follows more closely the syntax of HMMs and strong independence, and it is arguably easier to understand and implement for a non-expert in imprecise probability models.

De Bock and de Cooman [26] designed an algorithm that computes the maximal joint state sequences of an iHMM under epistemic irrelevance in time polynomial in the input and linear in the number of maximal sequences. A state sequence is maximal if there is no other state sequence with greater probability under any distribution induced by the model. De Boom et al. [27] devised an analogous algorithm for iHMMs under strong independence with similar time complexity. Finding maximal state sequences is a conservative generalization of the most likely state sequence inference in HMMs. As the number of maximal sequences can be exponential in the number of state variables, their algorithm does not qualify as efficient (i.e., polynomial-time) inference in a strict sense (unless we are satisfied with selecting an arbitrary

bounded subset of maximal sequences). In Section 7, we present polynomial-time algorithms for computing unconditional maximin and maximax state sequences; these can be seen as another possible generalization of the most likely state sequence inference in HMMs.

Yet another generalization of the most likely state sequence inference is the computation of E-admissible state sequences. A state sequence is E-admissible if it is a most likely state sequence for at least one distribution induced by an iHMM. Very recently, De Bock et al. [28] developed an algorithm that efficiently decides whether the set of E-admissible state sequences has cardinality strictly greater than one in bounded treewidth models. They showed how this algorithm can be used to measure the sensitivity of MAP inferences to perturbations in the parameters.

The use of credal sets in modeling sequential data is not new. Kozine and Utkin [29] investigated Markov chains with interval-valued transition probabilities. De Cooman et al. [30] used credal sets for sensitivity analysis in Markov chains. Škulj [31, 32] defined imprecise Markov chains, and analyzed some basic asymptotic behaviors such as regularity and ergodicity. Crossman et al. [33] studied imprecise Markov chains with absorbing states. Antonucci et al. [34] investigated the use of iHMMs under epistemic irrelevance for tracking tasks. Benavoli et al. [8] defined an iHMM over continuous variables aimed at robust filtering. An imprecise version of the Baum-Welch procedure [1], used to estimate the parameters of an HMM when the state sequence is not observable, was developed by Antonucci et al. [35], and tested on an activity recognition task. Van Camp and de Cooman [36] extended the learning of iHMMs from data to the case of epistemic irrelevance. In [37], the authors designed a method for comparing two iHMMs according to their asymptotic data likelihood, and also applied it on an activity recognition task.

3. Hidden Markov Models

A Hidden Markov model (HMM) describes a stochastic process over a sequence of *state variables* $Q_1, \ldots, Q_T$ and *manifest variables* $O_1, \ldots, O_T$. Each state variable Q_t, $t = 1, \ldots, T$, takes values in a finite set $\mathcal{Q} = \{1, \ldots, N\}$; each manifest variable O_t takes value in a finite set $\mathcal{O} = \{1, \ldots, M\}$.[2] We

[2]The constraint that all state variables or all manifest variables share the same sample space is introduced for simplicity of notation and because they are commonly observed in

denote an arbitrary value of state variable Q_t by q_t, i or j, and similarly for O_t. The parameter t that indexes either family of variables is called *time (step)*; we use a temporal metaphor and refer to the relative indexes of variables by using terms such as past, future and present in the usual way. The stochastic process satisfies the following properties:

P1 A state variable Q_t is stochastically independent of all the variables in the past given its immediate predecessor state variable Q_{t-1}, that is, $\mathbb{P}(Q_t = q_t | Q_{1:t-1} = q_{1:t-1}, O_{1:t-1} = o_{1:t-1}) = \mathbb{P}(Q_t = q_t | Q_{t-1} = q_{t-1})$, where the notation $X_{1:r} = x_{1:r}$ denotes the event $X_1 = x_1, \ldots, X_r = x_r$. By symmetry of independence, this implies that Q_t is also stochastically independent of all future variables given Q_{t+1}.

P2 A manifest variable O_t is stochastically independent of any other variable given the state variable Q_t: $\mathbb{P}(O_t = o_t | O_{1:t-1} = o_{1:t-1}, O_{t+1:T} = o_{t+1:T}, Q_{1:T} = q_{1:T}) = \mathbb{P}(O_t = o_t | Q_t = q_t)$.

Property P1 is known as the first-order Markov property, and it roughly states that the past is irrelevant to the future once the current state of the process is disclosed. It is also equivalent to stating that the variables $Q_1, \ldots, Q_T$ form a (first-order) Markov chain. Property P2 states that manifest variables are conditionally independent given the state variables. Formally,

Definition 1. *A hidden Markov model is a tuple*

$$\lambda = (a_2^1, \ldots, a_T^N, b_1^1, \ldots, b_T^N, \pi)\,,$$

where

$$\begin{aligned} a_t^i(j) &:= \mathbb{P}(Q_t = j | Q_{t-1} = i)\,, && i = 1, \ldots, N\,, t = 2, \ldots, T\,, \\ b_t^i(j) &:= \mathbb{P}(O_t = j | Q_t = i)\,, && i = 1, \ldots, N\,, t = 1, \ldots, T\,, \\ \pi(i) &:= \mathbb{P}(Q_1 = i)\,, && i = 1, \ldots, N\,. \end{aligned}$$

The functions a_t^i, b_t^i and π are called the transition, emission and initial probability distributions*, respectively. The model is said to be* stationary *if for any i, t and t' we have that $a_t^i = a_{t'}^i$ and $b_t^i = b_{t'}^i$.*

applications; they can be easily relaxed to generic discrete variables without invalidating any of the results developed in this paper. A further generalization, with some limitations, to the case of continuous manifest variables is also discussed in Section 8.

An HMM λ is a succinct representation of a stochastic process satisfying Properties P1 and P2; it defines a joint probability distribution over the variables in the process by

$$\begin{aligned} p_\lambda(q_{1:T}, o_{1:T}) &:= \mathbb{P}_\lambda(Q_{1:T} = q_{1:t}, O_{1:T} = o_{1:T}) \\ &= \pi(q_1) b_1^{q_1}(o_1) \prod_{t=2}^{T} a_t^{q_{t-1}}(q_t) b_t^{q_t}(o_t)\,. \end{aligned}$$

Rabiner [1] wrote a gentle and comprehensive presentation of HMMs and their usage. We assume the reader is familiar with Rabiner's presentation and we adopt similar terminology and notation.

4. Imprecise Hidden Markov Models

Conventionally, uncertainty is modeled by a single probability measure. Many authors have warned about the potential pitfalls of such a representation for describing situations involving scarcity of data or diverging opinions. A recent and gentle introduction to the topic can be found in [38]. Levi [9] advocated the use of convex and closed sets of probability measures as a more adequate representation of knowledge; he called such a set a *credal set*. We denote by $\mathbb{K}_{X_{1:r}}$ a credal set of probability measures over variables $X_{1:r}$. The set $\text{ext}\mathbb{K}_{X_{1:r}}$ denotes the extreme functions of the set, that is, the functions that cannot be written as convex combinations of other functions in the set. Credal sets can be manipulated in much the same way as probability measures. For instance, given a credal set $\mathbb{K}_{X,Y}$ we obtain a conditional credal set of X given $Y = y$ by point-wise conditioning:[3]

$$\mathbb{K}_X^{Y=y} := \{\mathbb{P}(X|Y=y) : \mathbb{P} \in \mathbb{K}_{X,Y}, \mathbb{P}(Y=y) > 0\}\,.$$

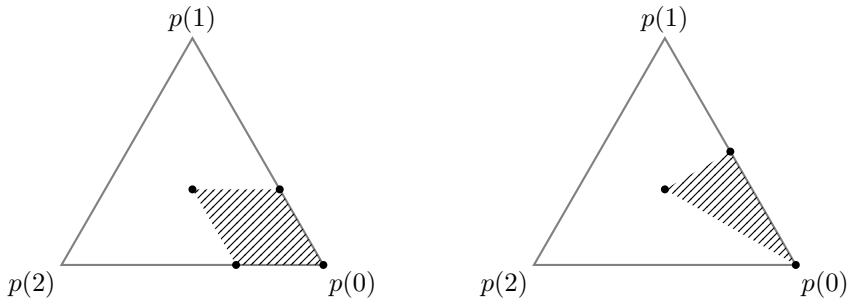
The lower and upper probability of an event $X = x$ are defined, respectively, by

$$\underline{p}(x) := \min_{\mathbb{P} \in \mathbb{K}_X^{Y=y}} \mathbb{P}(X = x|Y = y)\,,$$

and

$$\overline{p}(x) := \max_{\mathbb{P} \in \mathbb{K}_X^{Y=y}} \mathbb{P}(X = x|Y = y)\,.$$

[3] In principle, the conditional credal set could be empty; we assume in the remainder that this is never the case.



(a) $\mathbb{K}_X^{Y=0}$ (b) $\mathbb{K}_X^{Y=1}$

Figure 1: Barycentric coordinate-system visualization of the conditional credal sets specified by linear inequalities described in the text (hatched regions).

A (conditional) credal set $\mathbb{K}_X^{Y=y}$ is said to be specified by linear inequalities if it is characterized as the set of probability distributions $p(X)$ that satisfy

$$\sum_x f_k(x)p(x) \leq c_k\,, \quad k = 1, \ldots, C\,,$$

where $f_1, \ldots, f_C$ are arbitrary real functions on $\mathcal{X}$, and $c_1, \ldots, c_C$ are real values. As an example, consider variables X and Y taking values in $\{0, 1, 2\}$ and $\{0, 1\}$, respectively. The following conditional credal sets are specified by linear inequalities:

$$\begin{aligned}\mathbb{K}_X^{Y=0} &= \{p(X) : p(1) \leq 1/3, \quad p(2) \leq 1/3\}\,,\\ \mathbb{K}_X^{Y=1} &= \{p(X) : p(0) - p(1) \leq 0, \quad p(1) - p(2) \leq 0\}\,.\end{aligned}$$

These credal sets are depicted as hatched regions in Figure 1 using a barycentric projection of the probability simplex on $\{0, 1, 2\}$. Note that credal sets specified by finitely many linear inequalities are polytopes in the probability simplex.

A simple and commonly used type of inequalities specifying credal sets is of the form

$$0 \leq \ell(x) \leq p(x) \leq u(x) \leq 1, \quad \forall x \in \mathcal{X}\,,$$

where ℓ and u are functions from $\mathcal{X}$ to $[0, 1]$. Credal sets characterized in such a way are said to be specified by interval-valued probabilities. The credal set $\mathbb{K}_X^{Y=0}$ in the example is specified by interval-valued probabilities whereas the credal set $\mathbb{K}_X^{Y=1}$ is not. Finally, a credal set is said to be specified by a finite

set of distributions if it is defined as the convex hull of a set of (conditional) probability distributions $p(X)$. Represent a distribution $p(X)$ on $\{0,1,2\}$ by the triple $(p(0), p(1), p(2))$. Then a specification by finite set of distributions of the credal set $\mathbb{K}_X^{Y=0}$ in the example is the convex hull of

$$\{(1,0,0), (2/3,1/3,0), (1/3,1/3,1/3), (2/3,0,1/3)\}\,.$$

These distributions are the vertices (represented as black dots) of the hatched region on the left plot of Figure 1.

Given a credal set $\mathbb{K}_{X_{1:r}}$ we say that X_i and X_j are conditionally *strongly independent* given $X_k = x_k$ if they are stochastically independent under every probability measure in $\text{ext}\mathbb{K}_{X_{1:r}}$ [14]. This definition equates independence under an imprecise model with a set of independences under precise models that it contains, and it is thus adequate when we assume that our knowledge *can* in principle be captured by a (precise) probabilistic model but we lack the resources (expertise, time, money, etc) to do so.

An imprecise hidden Markov model (iHMM) is a concise description of the same stochastic process described by an HMM, except that we replace the representation of uncertainty using single probability measures by credal sets and the notion of stochastic independence by strong independence. In other words, we consider a credal set $\mathbb{K}_{Q_{1:T},O_{1:T}}$ and assume that:

I1 A state variable Q_t is strongly independent of all the variables in the past given its immediately predecessor state variable Q_{t-1}.

I2 A manifest variable O_t is strongly independent of any other variable given the state variable Q_t.

An alternative way of expressing the above assertions is to say that every extreme probability measure in $\mathbb{K}_{Q_{1:T},O_{1:T}}$ satisfies P1 and P2. A formal definition is provided next.

Definition 2. *An imprecise hidden Markov model (iHMM) is a tuple*

$$\Lambda = (A_2^1, \ldots, A_T^N, B_1^1, \ldots, B_T^N, \Pi)\,,$$

where

$$\begin{aligned} A_t^i &:= \mathbb{K}_{Q_t}^{Q_{t-1}=i} && i = 2,\ldots,N\,, t = 1,\ldots,T\,,\\ B_t^i &:= \mathbb{K}_{O_t}^{Q_t=i} && i = 1,\ldots,N\,, t = 1,\ldots,T\,,\\ \Pi &:= \mathbb{K}_{Q_1}\,. \end{aligned}$$

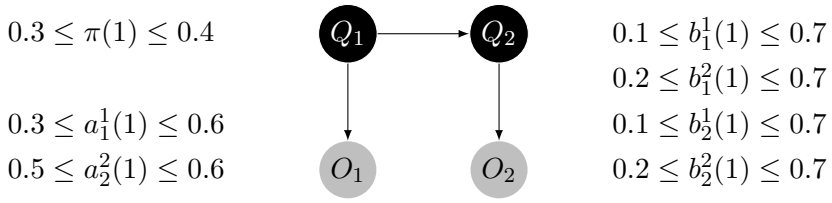


Figure 2: An example of an homogeneous iHMM with $N = M = T = 2$. State and manifest nodes are represented, respectively, as dark shaded and light shaded nodes.

The sets A_t^i, B_t^i *and* Π *are called* transition, emission and initial sets*, since they induce (closed convex sets of) corresponding distributions. An iHMM is said to be* homogeneous *if for all* i*: (i) transition credal sets* $A_2^i, \ldots, A_T^i$ *are equal, and (ii) emission credal sets* $B_1^i, \ldots, B_T^i$ *are equal.*

The above notation emphasizes the analogies with standard HMMs. In fact, an iHMM can be seen as a set of precise HMMs, one for each combination of probability distributions $a_2^1 \in A_2^1, \ldots, a_T^N \in A_T^N, b_1^1 \in B_1^1, \ldots, b_T^N \in B_T^N, \pi \in \Pi$. By construction, an iHMM Λ induces a credal set $\mathbb{K}_{Q_{1:T}, O_{1:T}}$ of probability measures over $(Q_{1:T}, O_{1:T})$ whose *extreme distributions* satisfy

$$\mathbb{P}(Q_{1:T} = q_{1:T}, O_{1:T} = o_{1:T}) = \pi(q_1) b_1^{q_1}(o_1) \prod_{t=2}^{T} a_t^{q_{t-1}}(q_t) b_t^{q_t}(o_t)\,,$$

where $a_2^1 \in \mathrm{ext}A_2^1, \ldots, a_T^N \in \mathrm{ext}A_T^N, b_1^1 \in \mathrm{ext}B_1^1, \ldots, b_T^N \in \mathrm{ext}B_T^N, \pi \in \mathrm{ext}\Pi$. This reinforces the view of iHMMs as sets of HMMs.

Figure 2 shows an homogeneous iHMM $(\Pi, A_2^1, A_2^2, B_1^1, B_1^2, B_2^1, B_2^2)$. All credal sets are specified by interval-valued probabilities. For instance, the initial credal set in the example is $\Pi = \{p(Q_1) : 0.3 \leq p(1) \leq 0.4\}$.

Given an iHMM $\Lambda = (A_2^1, \ldots, A_T^N, B_1^1, \ldots, B_T^N, \Pi)$, we write $A_t := \times_{i=1}^N A_t^i$ to denote the Cartesian product of transition credal sets at time t (analogously for B_t), $A_{t_0:t_f} := \times_{t=t_0}^{t_f} A_t$ to denote the Cartesian product of credal sets from time t_0 to time t_f (analogously for $B_{t_0:t_f}$), where by convention $A_1^i := \Pi$ for any i.

5. Data Likelihood

HMMs are commonly used to classify sequential data by choosing the model that best fits a sequence of observations according to the likelihood. For example, a common use of HMMs in speech recognition is to fit (or learn) a different HMM for each individual; the speaker is determined by selecting the corresponding HMM which maximizes the probability of observing a recorded passphrase (a sequence of observed manifest variables). Formally, we define the task as follows. Given a finite set Λ of (precise) HMMs, and a sequence of observations $o_{1:T} \in \times_{t=1}^{T} \mathcal{O}$, determine

$$\lambda^* = \operatorname*{argmax}_{\lambda \in \Lambda} p_\lambda(o_{1:T}) \,.$$

In order to generalize to the credal setting, we use the following notion of dominance.

Definition 3. *Given two iHMMs Λ_1 and Λ_2 and an observation sequence $o_{1:T}$, we say that* Λ_1 dominates Λ_2 *for $o_{1:T}$, denoted $\Lambda_1 \succ \Lambda_2$, if and only if*

$$\underline{p}_{\Lambda_1}(o_{1:T}) > \overline{p}_{\Lambda_2}(o_{1:T}) \,. \tag{1}$$

Dominance suggests that iHMMs can be used as *credal classifiers* [18] for reliable/robust sequence classification in the same way as HMMs are used for classifying sequential data. A class label associated to model Λ_1 is preferred as a classification of $o_{1:T}$ over a class label associated to model Λ_2 if and only if $\Lambda_1 \succ \Lambda_2$. Given a finite set $\Lambda_1, \ldots, \Lambda_K$ of iHMMs, credal classification outputs the set of undominated models

$$\Lambda^* = \{\Lambda_k : \nexists j \text{ such that } \Lambda_j \succ \Lambda_k\} \,. \tag{2}$$

The elements of Λ^* are determined by computing the upper and lower likelihood for each model Λ_k, $k = 1, \ldots, K$. In the rest of this section, we present an algorithm to compute such probabilities given an iHMM Λ, which is inspired on the algorithm of Zaffalon and Fagiuoli [18] for computing joint probability bounds in credal trees. For the sake of brevity, we present only the derivation for the lower bound. The algorithm for upper likelihoods is analogous.

Consider a sequence $o_{1:T}$ of observed symbols and an iHMM Λ. Our lower likelihood algorithm performs the following steps:

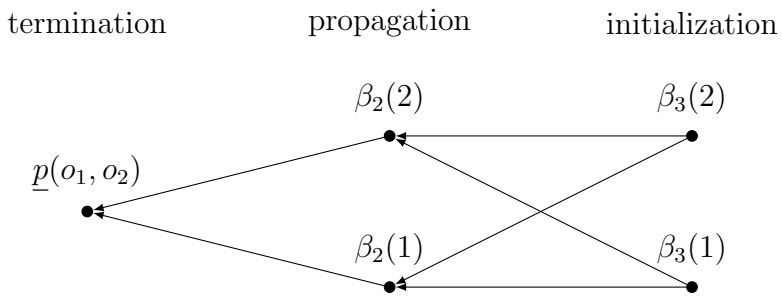


Figure 3: Illustration of lower likelihood computation for the iHMM in Figure 2.

1. **Initialization:**

$$\beta_{T+1}(i) := 1, \qquad i = 1, \ldots, N. \tag{3}$$

2. **Propagation:** For $t = T$ to $t = 2$ compute

$$\beta_t(i) := \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} \sum_{j=1}^{N} a_t^i(j) b_t^j(o_t) \beta_{t+1}(j), \qquad i = 1, \ldots, N. \tag{4}$$

3. **Termination:** Compute and output

$$\min_{\pi \in \Pi, b_1 \in B_1} \sum_{j=1}^{N} \pi(j) b_1^j(o_1) \beta_2(j) \; . \tag{5}$$

The variables $\beta_t(i)$ are called *lower backward variables*, and their definition is very similar to the backward variables used to compute the data likelihood of (precise) HMMs [1]. The computation is performed in layers and backwards in the time steps, that is, we start by computing the values of $\beta_T(i)$ for all i, then we compute the values of $\beta_{T-1}(i)$, and so on, until all variables have their value computed. The order in which variables are computed within a layer is arbitrary. As we show next the output is a tight lower bound on the likelihood. For example, the computation of lower backward variables for the iHMM in Figure 2 is shown in the trellis diagram in Figure 3. The nodes in the figure represent lower backward variables and the arcs represent temporal dependences (the value of the variable associated with a node needs to be calculated before those of its children).

The next result establishes the soundness and time complexity of the algorithm.

Theorem 1. *Given an iHMM Λ, the lower likelihood algorithm outputs $\underline{p}_\Lambda(o_{1:T})$ in time polynomial in the size of input.*

Proof. To show that the algorithm is correct, consider a mathematical induction in $t = T, \ldots, 2$ with the hypothesis

$$\beta_t(i) = \underline{p}(o_{t:T}|Q_{t-1} = i) \; . \tag{6}$$

The basis of induction for $t = T$ is straightforward:

$$\beta_T(i) = \min_{\substack{a_T^i \in A_T^i \\ b_T \in B_T}} \sum_{j=1}^{N} a_T^i(j) b_T^j(o_t) \tag{7}$$

$$= \underline{p}(o_T|Q_{T-1} = i). \tag{8}$$

To prove the inductive step, assume (6) holds at $t+1$ for $1 < t < T$. We thus have that

$$\beta_t(i) = \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} \sum_{j=1}^{N} a_t^i(j) b_t^j(o_t) \underline{p}(o_{t+1:T}|Q_{t+1} = j) \tag{9}$$

$$= \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} \sum_{j=1}^{N} a_t^i(j) b_t^j(o_t) \min_{\substack{a_{t:T} \in A_{t+1:T} \\ b_{t:T} \in B_{t:T}}} \sum_{i_{t+1:T}} \prod_{\tau=t+1}^{T} a_\tau^{i_{\tau-1}}(i_\tau) b_\tau^{i_\tau}(o_\tau) \tag{10}$$

$$= \underline{p}(o_{t:T}|Q_{t-1} = i). \tag{11}$$

In (9), we used the known fact that the value of $\underline{p}(o_{t+1:T}|Q_{t+1} = j)$ is attained at an extreme of the joint credal set [17] and hence factorizes to obtain (10). Each minimization inside the sum in (10) is either independent or constant with respect to the of value of j, and can thus be moved to front, obtaining (11). The correctness of the algorithm follows immediately by assuming (6) holds at $t = 2$ and then applying the same steps as in (9) and (10).

The complexity of computing (4) for given i and t depends on the specification of the corresponding transition and emission credal sets in the input. Define auxiliary variables $v_t^j = \beta_{t+1}(j) \min_{b_t^j \in B_t^j} b_t^j(o_t)$. Since every distribution b_t^j is selected from a different credal set B_t^j we can move the minimizations over b_t^j in (4) inside the sum and obtain

$$\beta_t(i) = \min_{a_t^i \in A_t^i} \sum_{j=1}^{N} a_t^i(j) v_t^j \; . \tag{12}$$

Consider the case where credal sets are specified by interval-valued probabilities. Then the values of v_t^j can be computed in constant time. Because each A_t^i is defined through intervals, the optimization in (12) reduces to a fractional knapsack problem, which can be solved in $O(N)$ time [18, 39]. Now consider the specification by finite sets of distributions. Each v_t^j can be computed by evaluating its value at a different distribution in the corresponding emission credal set (each evaluation takes constant time), taking linear time in the number of distributions. The right-hand side of (12) can then be solved by evaluating each possible transition distribution (each evaluation takes time linear in N), hence the whole procedure takes time linear in the size of the credal sets and in N. Finally, consider the case where credal sets are specified as a set of inequality constraints. Then computing v_t^j requires solving a simple linear program $\min b_t^j(o_t)$ s.t. $\sum_{o_t} f_k(o_t) b_t^j(o_t), k = 1, \ldots, K$, where f_k are part of the specification of the credal set B_t^j. Similarly, the optimization in (12) can be formulated as a linear program with objective $\sum_{j=1}^{N} a_t^i(j) v_t^j$ and the specification of A_t^i as constraints. Combined, both linear programs take time polynomial in the input size.

Let U denote the worst-case running time of computing (4) for any given i and t. The computation of the lower likelihood solves $O(TN)$ recursions of $\beta_t(i)$, taking a total time of $O(TNU)$, and is hence polynomial in the input size. □

The proof above shows that when credal sets are specified by interval-valued probabilities the total running time is $O(TN^2)$. This is the same time taken to compute likelihoods in HMM [1].

6. Filtering

Sequential data are often used to produce good estimates of the present given observation on the past. For example, in tracking, the hidden variables encode the true position of a target, while the manifest variables encode its sensed position. Due to noisy sensors, there is usually a mismatch between sensed and true positions, and one is interest in computing the probability of the current position given the past observations. This type of task is commonly know in the literature as filtering. Formally, given observations on the manifest variables for T time steps, we are interested in computing

$$\mathbb{P}(Q_T = q_T | o_{1:T}) \quad \text{for all } q_T = 1, \ldots, N \,. \tag{13}$$

When parameters are imprecisely-specified, hidden variable probabilities cannot be point-wise estimated; instead, we can compute their lower and upper bounds:

$$\underline{p}(q_T|o_{1:T}) \,, \quad \overline{p}(q_T|o_{1:T}) \,. \tag{14}$$

We can use the interval above as a measure of reliability of the (precise) probability in (13). For instance, we might consider a prediction $Q_T = i$ obtained through (13) reliable if $\underline{p}(i|o_{1:T}) \geq \overline{p}(j|o_{1:T})$ for $j = 1, \ldots, N$. This criterion is known as *interval dominance*. A more restrictive criterion is obtained by E-admissibility, which consider a prediction $Q_t = i$ reliable if $\min_{\mathbb{P}} (\mathbb{P}(Q_T = i|O_{1:T} = o_{1:T}) - \mathbb{P}(Q_T = j|O_{1:T} = o_{1:T})) \geq 0$ for all j. Even though E-admissibility is not defined in terms of upper and lower probabilities, it might also be obtained using similar techniques [23]; here we adopt interval dominance for the sake of simplicity [40].

The polynomial time algorithm developed in this section is based on the decision version of the problem. Formally, given an observation sequence $o_{1:T}$ of T samples, a hidden state $q_T \in \{1, \ldots, N\}$, and a rational number $k \in [0, 1]$, we wish to decide whether

$$\underline{p}(q_T|o_{1:T}) < k \,. \tag{15}$$

Conversely, we wish to decide whether

$$\overline{p}(q_T|o_{1:T}) > k \,. \tag{16}$$

We assume the model Λ is fixed from now on, and drop it from the notation. The lower (resp., upper) predictive probability in (14) can be obtained by finding through a binary search procedure the minimum (resp., maximum) k for which Ineq. (15) (resp., Ineq. (16)) is satisfied. Consider the problem of deciding whether the lower probability is beneath a constant. It is straightforward to see that Ineq. (15) is satisfied if and only if

$$\min_{p \in \mathbb{K}_{Q_T, O_{1:T}}} \left[p(q_T, o_{1:T}) - k \sum_{i_T} p(i_T, o_{1:T}) \right] < 0 \,. \tag{17}$$

Conversely, Ineq. (16) is true if and only if

$$\max_{p \in \mathbb{K}_{Q_T, O_{1:T}}} \left[p(q_T, o_{1:T}) - k \sum_{i_T} p(i_T, o_{1:T}) \right] > 0 \,. \tag{18}$$

Assume a sequence of observations $o_{1:T}$, a value q_T for Q_T and a value $\epsilon > 0$ specifying the precision of the binary search are given. The algorithm for performing filtering (w.r.t. the lower bound) in iHMMs is given next.

1. **Initialization:** Set $k_l \leftarrow 0, k_u \leftarrow 1$.
2. **Binary search:** While $k_u - k_l \geq \epsilon$ do:
   (a) **Initialization:** Set $k \leftarrow (k_l + k_u)/2$ and define

$$\gamma_{T+1}(q_T) := 1 - k, \tag{19}$$

$$\gamma_{T+1}(i) := -k, \qquad i = 1, \ldots, N,\ i \neq q_T. \tag{20}$$

   (b) **Propagation:** For $t = T$ to $t = 2$ compute

$$\gamma_t(i) := \min_{\substack{a_t^i \in A_T^i \\ b_t \in B_t}} \sum_{j=1}^{N} a_t^i(j) b_t^j(o_t) \gamma_{t+1}(j), \qquad i = 1, \ldots, N. \tag{21}$$

   (c) **Termination:** If

$$\min_{\substack{\pi \in \Pi \\ b_1 \in B_1}} \sum_{j=1}^{N} \pi(j) b_1^j(o_1) \gamma_2(j) > 0$$

   then set $k_l \leftarrow k$ else set $k_u \leftarrow k$.
3. **Termination:** Output $(k_l + k_u)/2$.

The steps 2(a) to 2(c) are very similar to the corresponding steps of the likelihood algorithm, with two noticeable differences: (i) the variables $\gamma_{T+1}(i)$ might differ with respect to the value of i (unlike the variables $\beta_{T+1}(i)$), and (ii) the variables $\gamma_t(i)$ can take on negative values. These differences introduce only minor changes to the implementation and to the time complexity of the algorithm.

The soundness and time complexity of the algorithm are given in the following result.

Theorem 2. *Given an iHMM Λ, the filtering algorithm outputs $\underline{p}(q_T|o_{1:T})$ in time polynomial in the size of the input.*

Proof. The fact that the binary search finds the desired value has already been shown in the discussion in the beginning of this section. Hence, it remains only to show that the algorithm computes (17). To this end we perform a mathematical induction in $t = T, \ldots, 2$ with the following hypothesis:

$$\gamma_t(i) = \min_{p} \sum_{i_T} \gamma_{T+1}(i_T) p(i_T, o_{t:T}), \tag{22}$$

where the optimization is carried out over the credal set $\mathbb{K}^{Q_{t-1}=i}_{Q_T,O_{t:T}}$. The basis for $t = T$ is immediate:

$$\begin{aligned}\gamma_T(i) &:= \min_{\substack{a^i_T \in A^i_T \\ b_T \in B_T}} \sum_{j=1}^{N} a^i_T(j) b^j_T(o_t) \gamma_{T+1}(j) \\ &= \min_{p \in \mathbb{K}^{Q_{t-1}=i}_{Q_T,O_T}} \sum_{i_T} p(i_T, o_T) \gamma_{T+1}(i_T) \ .\end{aligned} \tag{23}$$

To prove the inductive step, assume that (22) holds at $t+1$, for $1 < t < T$, that is, that

$$\gamma_t(i) = \min_{\substack{a^i_t \in A^i_t \\ b_t \in B_t}} \sum_{j=1}^{N} a^i_t(j) b^j_t(o_t) \min_{\{p_j\}} \sum_{i_T} p_j(i_T, o_{t+1:T}) \gamma_{T+1}(i_T) \ , \tag{24}$$

where p_j is selected in $\mathbb{K}^{Q_t=j}_{Q_T,O_{t+1:T}}$. By construction, each p_j factorizes as

$$p_j(i_T, o_{t+1:T}) = \sum_{i_{t+1}} a^j_{t+1}(i_{t+1}) b^{i_{t+1}}_{t+1}(o_{t+1}) \sum_{i_T} p_{i_{t+1}}(i_T, o_{t+2:T}) \gamma_{T+1}(i_T) \, ,$$

where $a^j_{t+1} \in A^j_{t+1}$, $b^{Q_{t+1}}_{t+1} \in B^{Q_{t+1}}_{t+1}$ and $p_{i_{t+1}}(i_T, o_{t+2:T}) \in \mathbb{K}^{Q_{t+1}=i_{t+1}}_{Q_T,O_{t+2:T}}$. The functions a^j_{t+1} can be selected independently for each value of $Q_t = j$. The choice of functions $b^{Q_{t+1}}_{t+1}$ and $p_{i_{t+1}}(Q_T, O_{t+2:T})$ is constant with respect to the value of j. Hence, all the corresponding minimizations can be moved out of the sum, obtaining

$$\begin{aligned}\gamma_t(i) &= \min_{\substack{a^i_t \in A^i_t \\ b_t \in B_t}} \min_{\{p_j\}} \sum_{j=1}^{N} a^i_t(j) b^j_t(o_t) \sum_{i_T} p_j(i_T, o_{t+1:T}) \gamma_{T+1}(i_T) \\ &= \min_{\{p_i\}} \sum_{i_T} p_i(i_T, o_{t:T}) \gamma_{T+1}(i_T) \ ,\end{aligned} \tag{25}$$

where $p_i \in \mathbb{K}^{Q_{t-1}=i}_{Q_T,O_{t:T}}$. The correctness of the algorithm follows immediately from the induction hypothesis for $t = 2$.

The time complexity analysis is similar to the analysis of the lower likelihood computation. The only difference is that the choice of the functions b^j_t depends on the sign of $\gamma_{t+1}(j)$ (which unlike the variables $\beta_t(i)$ they can take

on negative values). The analysis of running time is identical. If U is the time it takes to solve (21), then the total running time is $O(TNU)$, where U is always polynomial in the input size. In particular, when credal sets are specified as interval-valued probabilities the total running time is $O(TN^2)$. Again, this is the same time complexity of filtering in HMMs. The value of the lower posterior probability is obtained by performing a binary search in the value of k in the interval $[0, 1]$. One can show that the solution of the search is a ratio of polynomials in the numbers of the input. Hence, the binary search is guaranteed to finish in at most a polynomial number of steps in the size of the input. □

Even though we can decide a value for the constant ϵ which is polynomial in the size of the input and such that the binary search is guaranteed to find the correct value (apart from numerical inaccuracies introduced by floating-point arithmetic in the calculations performed), in practice we simply set ϵ to a reasonably small value, say 10^{-6}. This still leaves the time complexity polynomial and introduces only a negligible error.

6.1. Prediction

A task similar to filtering is that of *prediction* where one is interested in estimating the value not of the current state q_T but of a future state, say q_{T+K}, with $K > 0$. In the imprecise setting, this translates to e.g. computing upper and lower probabilities $\underline{p}(q_{T+K}|o_{1:T})$ and $\overline{p}(q_{T+K}|o_{1:T})$. It can easily been shown that the filtering algorithm we presented can be readily used to perform such computations by defining the emission credal sets $B^i_{O_t}$, for $t = T+1, \ldots, T+K$, as singletons containing the uniform distribution $p(O_t|Q_t = i) = 1/M$. In Section 8.4, we show experiments on a text completion task where we use such an approach to predict the upper and lower probabilities of the next state variable given a sequence of observations up to the current step.

7. Most Likely State Sequence

Yet another important use of (precise) HMMs is to infer the jointly most probable configuration of the hidden states for a given sequence of observations, that is, given an HMM λ and observations $o_{1:T}$ obtain

$$\operatorname*{argmax}_{q_{1:T}} \frac{p(q_{1:T}, o_{1:T})}{\sum_{i_{1:T}} p(i_{1:T}, o_{1:T})} \,. \tag{26}$$

For instance, the above task appears in the tagging of natural language sentences with part-of-speech labels (verb, noun, etc.) [2]. The Viterbi algorithm [41] can be used to efficiently solve (26) for the optimal sequence based on the fact that the denominator is constant with respect to the choice of $q_{1:T}$ and can thus be excluded from the computation. In other words, the Viterbi algorithm finds

$$\operatorname*{argmax}_{q_{1:T}} p(q_{1:T}, o_{1:T}) \ , \tag{27}$$

which coincides with the solution of (26). This approach can be generalized to iHMMs by adopting the following criteria.

Definition 4. *Given an iHMM* Λ*, the* joint maximin *and* joint maximax likely sequences *for an observation sequence* $o_{1:T}$ *are given by, respectively,*

$$\operatorname*{argmax}_{q_{1:T}} \underline{p}(q_{1:T}, o_{1:T}) \ , \tag{28}$$

$$\operatorname*{argmax}_{q_{1:T}} \overline{p}(q_{1:T}, o_{1:T}) \ . \tag{29}$$

The maximin and maximax likely sequences describe extreme scenarios where the distributions are chosen, respectively, in a pessimistic and optimistic way. We can use this fact to analyze the sensitivity of the Viterbi algorithm in precise HMMs with respect to fluctuations in the parameters of the model. In this sense, most likely state sequences found with the Viterbi algorithm can conservatively be regarded as reliable if the maximin and maximax sequences found by a corresponding iHMM coincide. A less conservative approach is to use the similarity of the maximin and maximax sequences element-wise: we consider that a state q_t is reliable if it appears in both maximin and maximax sequences. This is the approach we use in the experiments in Section 8.

Note that, unlike the case for precise HMMs, the state sequences obtained using (28) and (29) may differ from the sequences provided by

$$\operatorname*{argmax}_{q_{1:T}} \min_{p \in \mathbb{K}_{Q_{1:T}, O_{1:T}}} \frac{p(q_{1:T}, o_{1:T})}{\sum_{i_{1:T}} p(i_{1:T}, o_{1:T})} \ , \tag{30}$$

$$\operatorname*{argmax}_{q_{1:T}} \max_{p \in \mathbb{K}_{Q_{1:T}, O_{1:T}}} \frac{p(q_{1:T}, o_{1:T})}{\sum_{i_{1:T}} p(i_{1:T}, o_{1:T})} \ , \tag{31}$$

since in the above equations the denominators vary with the choice of state sequence (as p varies with them) and hence cannot be excluded. If instead

of analyzing the sensitivity of the Viterbi algorithm one wishes to study the sensitivity of the model itself to fluctuations in its parameters, then comparing the sequences returned in (30) and (31) is arguably a more principled approach. However, empirical results with artificial data described in the Appendix show that, at least for small chains, the sequences obtained using joint probabilities do not differ significantly from those obtained by the posterior probabilities. This suggests that the use of joint maximin and maximax sequences is a viable approximation even when one wants to assess the sensitivity of the model (and hence use (30) and (31)).

In what follows, we present an algorithm for computing joint maximin sequences as defined in (28) given a sequence $o_{1:T}$. A similar algorithm for computing (29) can be obtained by analogy.

1. **Initialization:** Define

$$\delta_{T+1}(i) := 1, \qquad i = 1, \ldots, N. \tag{32}$$

2. **Propagation:** For $t = T$ to $t = 2$ compute

$$\delta_t(i) := \max_j \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} a_t^i(j) b_t^j(o_t) \delta_{t+1}(j), \qquad i = 1, \ldots, N, \tag{33}$$

$$\phi_t(i) := \operatorname*{argmax}_j \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} a_t^i(j) b_t^j(o_t) \delta_{t+1}(j), \qquad i = 1, \ldots, N. \tag{34}$$

3. **Termination:** Compute

$$q_1 = \operatorname*{argmax}_j \min_{\substack{\pi \in \Pi \\ b_1 \in B_1}} \pi(j) b_1^j(o_1) \delta_2(j),$$

and output $q_{1:T}$, where

$$q_t = \phi_t(q_{t-1}), \qquad t = 2, \ldots, T.$$

The algorithm is similar to the algorithms for likelihood and filtering in that it computes variables in layers and backwards in time. Note that the values of the variables $\phi_t(i)$ can be obtained when computing the variables $\delta_t(i)$, dispensing with the extra overhead of solving the right-hand side of (34). The soundness and time complexity of the algorithm are given next.

Theorem 3. *Given an iHMM Λ, the maximin state sequence algorithm outputs* $\operatorname{argmax}_{q_{1:T}} \underline{p}(q_{1:T}, o_{1:T})$ *in time polynomial in the input size.*

Proof. The correctness of the algorithm is proved by mathematical induction in $t = T, \ldots, 2$ with the following hypothesis:

$$\delta_t(i) = \max_{q_{t:T}} \underline{p}(q_{t:T}, o_{t:T}|Q_{t-1} = i) \ , \tag{35}$$

$$\phi_t(i) = \operatorname*{argmax}_{q_{t:T}} \underline{p}(q_{t:T}, o_{t:T}|Q_{t-1} = i) \ . \tag{36}$$

The basis follows directly from the definition:

$$\begin{aligned} \delta_T(i) &= \max_j \min_{\substack{a_T^i \in A_T^i \\ b_T \in B_T}} a_T^i(j) b_T^j(o_T) \\ &= \max_{q_T} \underline{p}(q_T, o_T|Q_{T-1} = i), \end{aligned} \tag{37}$$

and therefore $\phi_T(i) = \operatorname{argmax}_{q_T} \underline{p}(q_T, o_T|Q_{T-1} = i)$. Now assume (35) and (36) are true at $t+1$ for some $1 < t < T$. Then,

$$\begin{aligned} \delta_t(i) &= \max_j \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} a_t^i(j) b_t^j(o_t) \max_{q_{t+1:T}} \underline{p}(q_{t+1:T}, o_{t+1:T}|j) \\ &= \max_j \max_{q_{t+1:T}} \underline{p}(q_{t+1:T}, o_{t+1:T}|j) \min_{\substack{a_t^i \in A_t^i \\ b_t \in B_t}} a_t^i(j) b_t^j(o_t) \\ &= \max_{q_{t:T}} \underline{p}(q_{t:T}, o_{t:T}|Q_{t-1} = i). \end{aligned} \tag{38}$$

In (38) we could move the minimization inside the expression because the term $\max_{q_{t+1:T}} \underline{p}(q_{t+1:T}, o_{t+1:T}|j)$ is nonnegative and does not depend on the choice of a_t^i or b_t. For this same reason the minimization over a_t^i and b_t could be moved in front of $\underline{p}(q_{t+1:T}, o_{t+1:T}|j)$. The last equality is obtained applying the definition of lower probability. The proof for the inductive step of $\phi_t(i)$ is analogous.

The algorithm runs in $O(N^2T)$ time, since the minimizations can be solved by choosing lower probability bounds, which are already available for interval credal sets or can be computed in advance for credal sets specified by probability distributions or sets of inequalities (in the latter case we have the time complexity is $O(N^2T+U)$ where U is the time to solve pre-compute the lower probability bounds, which requires solving linear programs of size polynomial in the input). □

8. Experiments

In this section, we describe the results of experiments with real data that provide evidence of the efficiency and applicability of the algorithms developed here.[4] Before detailing the experiments and reporting and commenting the results, we discuss two common issues: how to learn the parameters of iHMMs, and how to evaluate their output. We start with the learning of iHMMs.

8.1. Learning Imprecise Hidden Markov Models

The parameters of an iHMM (i.e., the transition, emission and initial credal sets) can be elicited from experts or estimated from a data set. There is a vast literature on eliciting expert knowledge (we refer the reader to the work of [42]). The same is true for learning credal sets from *complete data*. For instance, the Imprecise Dirichlet Model (IDM) developed by Walley [10] learns an interval-valued credal set $\mathbb{K}_X$ given i.i.d. data samples $x^1, \ldots, x^N$ of a Multinomial distribution by

$$\frac{n(X = x)}{N + s} \leq p(x) \leq \frac{n(X = x) + s}{N + s}\,, \tag{39}$$

where $n(X = x)$ is the number of data points such that $x^j = x$, and s is a parameter that arises from the use of Dirichlet priors and indicates the strength of prior beliefs. For a fixed data sample size N, small values of s lead to a smaller credal set, and large values of s lead to a larger set. The difference between the two bounds is given by $s/(N + s)$; as one would expect, the size of the credal set learned with IDM decreases with the increase in the amount of data. Conditional credal sets $\mathbb{K}_X^{Y=y}$ can be learned using the Equation (39) for each value of $Y = y$.

When data are incomplete (i.e., when some or all the values of some variables have not been observed), the IDM cannot be directly applied. Incomplete data are treated within the framework of HMMs using the standard Baum-Welch algorithm [1], which implements an Expectation-Maximization procedure to infer the parameters e.g. in the absence of observations of the state variables. We can use a similar approach to estimate the parameters of iHMMs when state variables are not observed. As already discussed in

[4] A software implementation of the inference algorithms together with learning and classification scripts is freely available at `https://github.com/denismaua/ihmm`.

[43], the fractional counts estimated by the Baum-Welch algorithm can be regarded as the result of (pseudo-)observations of the state variables. We can run Baum-Welch and fractional (expected) counts obtained as input to an IDM, so that credal sets are learned in much the same way as in Equation (39). The use of credal sets in this approach is justified by the fact that the fractional count estimates from the Baum-Welch algorithm are prone to inaccuracies introduced by the scarceness of data and other factors [35].[5] For example, the Baum-Welch-IDM-based interval-valued specification of transition credal sets is:

$$\frac{E[n(Q_{t-1}=i, Q_t=j)]}{\sum_j E[n(Q_{t-1}=i, Q_t=j)] + s} \leq a_t^i(j) \leq \frac{E[n(Q_{t-1}=i, Q_t=j)] + s}{\sum_j E[n(Q_{t-1}=i, Q_t=j)] + s}, \quad (40)$$

where $E[n(Q_{t-1} = i, Q_t = j)]$ are the expected counts for a transition from state i to state j obtained through Baum-Welch algorithm. A similar relation can be considered for the initial and emission credal sets. We can deal with continuous manifest data by assuming that the emission credal sets are precise. Under this assumption, the inference algorithms developed can easily be modified to handle continuous manifest variables, by simply replacing the emission lower probabilities with the corresponding densities. This is identical to the use of virtual (or soft) evidence in Bayesian networks [44, Ch. 3.7].

8.2. Evaluating Credal Classifiers

The performance of HMMs is usually evaluated through the accuracy of the predictions or some other loss function that computes the cost of predicting a certain value knowing the true value. For example, likelihood inference is typically used to select one of a set of HMMs, each HMM being associated to a label. The performance of such an approach is commonly measured by the percentage of times that the correct model was selected by the inference procedure.

When considering credal sets in lieu of single probability measures, inference often results in sets of decisions. For example, the domination criterion established in Definition 3 might lead to more than a single model being

[5]Note that we assume MAR as a valid hypothesis since all states are missing in a presumably non selective way. In fact, iHMMs can also deal with other learning approaches that do not make assumptions about the missingness process [6].

selected as appropriate for describing some observation sequence. In such cases, simple accuracy or loss measures cannot be used. Evaluating the performance of a credal classifier thus requires specific descriptors. In fact, inferences drawn with imprecise models can be analyzed in aspects other than simple quality of predictions. For instance, one might be interested in characterizing the level of *determinacy* achieved by a prediction: this is given by the percentage of instances classified with a single label. We can also measure the imprecision of inference by *average output size*, that is, the average number of classes on instances for which multiple labels are returned.

Of course, we are generally also interested in the accuracy of predictions. In the imprecise setting, we distinguish between *single accuracy*, which is the (standard measure of) accuracy over instances classified with a single label, and *set accuracy*, which is the accuracy over the instances classified with more than one label. In the latter case, classification is considered correct if the set of labels includes the true class.

Since more than one measure is associated with the predictions of an imprecise model, we cannot directly compare precise and imprecise models. To this end, we adopt the utility-based approach proposed in [45], which combines accuracy and reliability of predictions into a single measure, based on game-theoretic principles. The starting point is *discounted accuracy*, which rewards a prediction containing K classes with $1/K$ if it contains the true class, and with 0 otherwise. This indicator can already be compared to the accuracy achieved by a determinate classifier, but discourages imprecision in results: as shown in [45], an imprecise classier outperforms its precise version only if the latter behaves worse than a random classifier.

Risk-averse decision makers might assign higher utility for indeterminate-but-correct outputs compared with wrong-but-determinate ones [45]. For instance, one might prefer an expert who suspends judgement whenever shes is not confident enough than an expert who chooses an alternative completely at random. This can be obtained by modifying discounted accuracy with a (concave) transformation u_w with $w \in [.65, .80]$.[6] A conservative approach consists in evaluating the whole interval $[u_{.65}, u_{.80}]$ for each credal classifier and compare it with the (single-valued) accuracy of traditional classifiers. This can be shown to approximate the rewarding of accuracy while penalizing variance, a common procedure in finance models.

[6] We refer the reader to [45, Sect. 9] for the functional forms of these transformations.

When we compare the performance of a credal classifier with that of a precise counterpart (i.e., a classifier based on the HMMs returned by a precise learning procedure without combination with the IDM), we indeed consider the *precise single accuracy*, that is, the accuracy of the precise classifier when the credal returns a single label, and the *precise set accuracy*, the accuracy of the precise classifier when the credal returns multiple labels.

To better understand these concepts consider Table 1. Out of five instances, the credal classifier (second column) is determinate twice, i.e, its *determinacy* is 2/5. When indeterminate (three instances), the classifier returns three classes once and two classes twice. The *average output size* is therefore 7/3. By comparing the output with the ground truth (first column), we see that the credal classifier returns the true class only once when determinate (two instances); when indeterminate (three instances) the returned set of classes includes the true twice. This corresponds to a *precise accuracy* equal to 1/2 and a *set accuracy* equal to 2/3. The computation of the *discounted accuracy* leads to $(1/2 + 0 + 0 + 1/3 + 1)/5$. The precise classifier (third column) has accuracy 2/5. The accuracy becomes 1/2 if we consider only the instances on which the credal classifier is determinate, and 1/3 if we consider only the indeterminate ones. These are, respectively, the *precise single accuracy* and the *precise set accuracy*. Finally note that the output of the credal classifier always includes the class returned by the precise classifier (this is a general fact which is easily provable), thus the precise single accuracy coincides with the single accuracy.

True class	Credal classifier	Precise classifier
red	red, yellow	red
red	yellow, green	yellow
yellow	green	green
green	red, green, yellow	yellow
green	green	green

Table 1: The output of a credal classifier with a ternary class on five instances. True values of the class variable and the output of a precise classifier are also reported.

8.3. Data Likelihood

To evaluate the ability of iHMMs in reliably selecting good fits of sequences of observations using the algorithms for data likelihood inference,

we first learn the parameters of an iHMM using IDM or Baum-Welch-based IDM. We then compute with the data likelihood algorithm the lower and upper bounds of the likelihood probability of the test instances on the iHMMs associated to the training instances, Finally, we assign to each test instance the class labels of the iHMMs whose likelihood intervals are not dominated. The corresponding classifier is credal, i.e., multiple class labels might be assigned to a test sequence.

8.3.1. Speech Recognition

The first task we consider is the classification of phonemes using the *Japanese Vowels* dataset [46]. This dataset contains 640 sequences representing sound records from nine male speakers. Each speaker uttered two Japanese vowels successively. For each utterance a 12-degree linear prediction analysis was carried to obtain a discrete time series with 12 linear predictive coding (LPC) cepstrum coefficients. Hence, an observation at time step t (o_t) consists of a real vector of dimension 12.

We use a set of 270 time series for training and another of 216 series for testing. Such a down-sampling allows a uniform stratification of the data (i.e., each dataset has an approximately equal number of samples for each speaker). The length of the sequences ranges from 7 to 29 time steps.

The above-described learning procedure is adopted to obtain iHMMs from each training sequence. Stationary models are considered. The data are not discretized and the emission term is assumed to be precise. A multivariate Gaussian distribution with diagonal covariance matrix is used to model the emission distribution. The number of possible values for the state variables is decided by multivariate Gaussian clustering (hence the states are latent and represent abstract concepts). As suggested in [10], the value $s = 2$ is adopted for the IDM imprecision parameter. The same value is used in the learning of the precise model for the strength of the (symmetric) Dirichlet prior.

In the first experiment, the training sequences associated to each speaker are merged together. Because of the large amount of training data, the corresponding classifier is very precise. Out of 216 test instances, the credal classifier returns more than a single class only in three cases, when two classes are returned. Remarkably, in each of these cases one of the two classes returned is the correct one, while the other is the one returned by the (precise)

HMM.[7]

For a more detailed evaluation we reduce the size of the training set to nine instances only, one for each speaker. To do that, the 270 training instances are split in 30 subsets, each one containing a single sequence for each speaker. For each subset of the training set, we learn a classifier and test it over the 216 test instances. The aggregated results are reported in Table 2.

Descriptor			
Determinacy	87.5%	(3.0%)	(5672/6480)
Average output size	2.1	(0.3)	(out of 9)
Single accuracy (= precise single accuracy)	69.9%	(7.5%)	(3962/5672)
Set accuracy	67.0%	(0.1%)	(541/808)
Discounted accuracy	65.1%	(6.5%)	
Utility-based accuracy $u_{.65}$	66.4%	(6.6%)	
Utility-based accuracy $u_{.80}$	67.6%	(6.7%)	
Accuracy precise counterpart	65.6%	(6.8%)	(4254/6480)
Precise single accuracy	69.9%	(7.5%)	(3962/5672)
Precise set accuracy	36.1%	(0.1%)	(292/808)

Table 2: Performance of the data likelihood credal classifier on the Japanese vowels speech recognition dataset. Standard deviations (third column) and counts (fourth column) are also reported.

The value of $u_{.65}$ (i.e., the descriptor which is less in favor of credal classifiers) is higher than the accuracy of the precise counterpart. This advocates the choice of a credal approach to the problem. Moreover, it is worth noticing that the classifier returns a single class in most of the cases, and if this is not the case only two classes are typically provided. Furthermore, on the instances for which more than a class is returned by the credal classifier, the precise classifier has performances considerably worse than on average (see last row in the table). In other words, the iHMM-based classifier allows to distinguish between "easy" instances for which the classifier returns a single class label which is very likely to be the correct one, and "difficult" instances

[7]By construction, the class label returned by an HMM-based classifier is always included in the set of non interval-dominated classes returned by the iHMM-based credal classifier.

for which, to preserve the same level of accuracy, the credal classifier should return multiple outputs.

Figure 4 depicts a sensitivity analysis with respect to the parameter s. The $u_{.65}$ accuracy of the iHMM and the accuracy of the HMM are depicted together with the (iHMM) determinacy. As expected, increasing the value of s reduces the determinacy. Yet, this has no particular effect on both the accuracies.

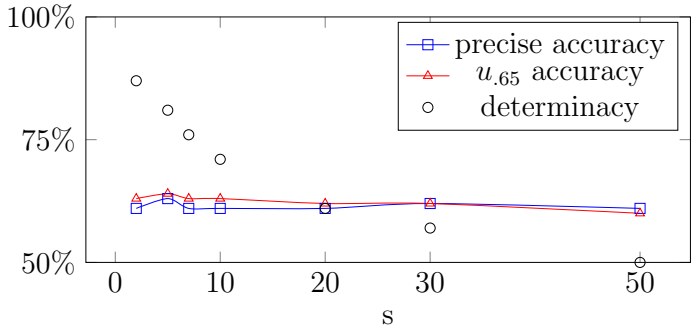


Figure 4: Sensitivity analysis of the Japanese Vowels dataset for different values of s.

8.3.2. Human Action Recognition

For a further validation of the likelihood algorithm we consider a classical computer vision benchmark: the Weizmann dataset for action recognition [47]. In this task, the class is the action depicted in the sequence (see Figure 5). These data are footage material which requires a *feature extraction* procedure at the frame level. Each frame is identified with a time step and the extracted features are the observable multivariate data. Our approach is based on histograms of oriented optical flows [48], a simple technique which describes the flows distribution in the whole frame as a histogram with 32 bins representing directions. The setup is the same as in the first benchmark apart from the number of features which, after PCA, is reduced to six. The dataset contains 80 sequences depicting eight individuals performing ten different actions. The results of a leave-one-out cross validation are depicted in Table 5. Basically the credal classifier behaves as in the first benchmark, with a utility-based accuracy higher than that of the precise classifier and a clear separation between single and set accuracy. Finally, let us note that the levels of accuracy of both the precise and the credal classifiers on this dataset are considerably worse than those reported in the literature (e.g.,

97.83% accuracy in [47]). This is a consequence of the very simple features considered here, our purpose being only to validate the iHMM algorithm and a comparison with its precise counterpart.

(a) Walk (b) Jumping Jack (c) Run

Figure 5: Three frames from sequences depicting human actions in the Weizmann dataset.[9]

Descriptor			
Determinacy	77.5%	(4.6%)	(62/80)
Average output size	2.4		(out of 10)
Single accuracy (= precise single accuracy)	35.5%	(6.0%)	(22/62)
Set accuracy	44.4%	(11.4%)	(8/18)
Discounted accuracy	32.1%		
Utility-based accuracy $u_{.65}$	33.5%		
Utility-based accuracy $u_{.80}$	35.0%		
Accuracy precise counterpart	31.3%	(5.2%)	(25/80)
Precise single accuracy	35.5%	(6.0%)	(22/62)
Precise set accuracy	16.7%	(8.5%)	(3/18)

Table 3: Performance of the data likelihood credal classifier on the Weizmann action recognition dataset. Standard deviations (third column) and counts (fourth column) are also reported.

8.4. Prediction: Text Completion

We take on the task of predicting the next letter in a sentence formed by the name of a movie. Our application regards smart TVs, where the user

[9] `http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html`

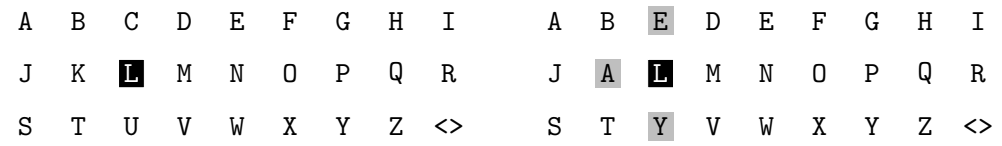


Figure 6: Example of a keyboard on the left with the cursor positioned at the letter L, where the user has, for example, typed "THE<>BIG<>". After clicking *OK* (on the right side), up to four suggestions appear around L. In this case, three suggestions are given. If suggestions are all wrong and the user wants to move in the direction of a suggestion in order to reach the desired letter, they *spend* an extra remote control click to move to the suggestion and then back to the keyboard.

has limited controlling options and wants to search for a movie to watch using only the remote control. We assume the following scenario, inspired by current implementations available in smart TVs: The user is presented with a search box and a virtual keyboard, which for simplification we assume to be composed of 26 letters plus the character *Space* denoted by <> (see Figure 6). In order to type a movie name, the user has four arrows to move a cursor over the keyboard letters orthogonally and an *OK* button to select a letter. Usually this is a tedious process, but most smart TVs have the option of enabling next-letter suggestions to the user. These suggestions (up to four of them) are shown around the last chosen letter in the four orthogonal neighbors. The drawback of such suggestions is that, in case they do not contain the desired letter and the desired letter is in the direction of a suggestion, the user loses one click in order to reach the actual desired letter (they need to move to the suggestion, and then move to the keyboard again in that direction, losing one click). We consider the problem of providing such suggestions to the user in order to minimize the loss of (undesired) clicks (note that the user only loses time if the desired letter is in the direction of a suggestion). We assume that at least one and at most four suggestions can be returned, following the layout of the graphical interface (to simplify matters, we assume the loss to be uniform over the four possible directions).

We take as data the whole list of movie names from the Internet Movie Database (`http://www.imdb.com`) and consider movie names containing letters and spaces only. This amounts to slightly more than 200 thousand names. For each name, we apply a perturbation algorithm to mimic the user input. This algorithm takes each word of a movie name and with probability

one third it either applies the corruption of one letter using one of its neighbor letters in the keyboard (to simulate a mistype, see Figure 6 for the layout) or the public map of misspelled words from *Wikipedia's list of common misspellings*[10] (to account for spelling errors from the user). Hence, state and manifest variables are very related, but are not exactly equal. Given the manifest variables from the past, we want to predict the state variable (that is, the next letter).

Using half of the complete data instances (each letter of a movie name is associated with a state variable, while each letter in the corresponding corrupted name is associated to a manifest one), we learn an iHMM without the assumption of stationarity, so letters appearing in position i and their predecessors in position $i-1$ are used only to learn the corresponding distribution of the state variable in time i given the state variable of time $i-1$. The same non-stationarity is used for manifest variables, so the corresponding pairs of state and manifest variable distributions are learned from data using only the data of that position in the movie names. The reason for non-stationarity is that the relations between letters in different positions of the movie names are different, as movie names have some patterns themselves (for instance, the article `THE` appears often in the beginning of movies). The other half of the data instances is used for testing the accuracy of our model. The accuracy is computed based on the advantage of the suggestions in terms of number of necessarily remote control clicks. To simplify calculations, we assume that we lose $K/4$ points if the correct letter is not among the K suggested letters (so suggesting more letters incur more loss but increases the chance of correctly identifying the next letter). The reasoning is that on average we lose $K/4$ clicks by presenting K suggestions, because to move in directions without a suggestion we do not lose clicks.

In order to obtain the desired predictions with the iHMM, we apply the algorithm for filtering as explained in Section 6 (to use the algorithm to predict Q_{T+1} instead of Q_T we set $\mathbb{P}(O_{T+1}|Q_{T+1})$ as a single uniform distribution for every value of Q_{T+1}, as described in the last paragraph of that section). The iHMM uses the set of non-dominated predictions (as explained before) as the suggestions to the user, and we report results with $s = 2$ and $s = 27$ (the number of letters in our alphabet). As comparison, we use the precise HMM, where the options are sorted by their posterior probability and the

[10] `http://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings`

best four options are returned (four suggestions is the standard approach currently used in the smart TVs with which we have experimented). As we see in Table 4, the iHMM obtains a better loss than the HMM by selecting the appropriate number of suggestions to output to the user. We focused on initial letters (as they are arguably the most important for the prediction and some specific values of s).

Descriptor	$s = 2$			$s = 27$		
Letter position T	2	3	4	2	3	4
Determinacy	99.8%	96.0%	98.6%	85.3%	56.4%	55.0%
Average K	4.0	2.6	2.3	2.7	3.2	3.5
HMM Loss	16,502.5	31,992.5	29,190.0	17,210.0	31,232.5	29,629.5
iHMM Loss	12,035.0	14,487.5	13,382.5	13,155.0	23,737.5	24,327.5

Table 4: Losses of HMM and iHMM to suggest the next letter in the smart TV example over 100 thousand test instances. Different letter positions in the movie name and values of s are used. Determinacy indicates the percentage of cases where a single suggestion was given. Average K is the average number of suggestions that were given by iHMM. The losses account by how many clicks were lost with each method, so smaller the better.

A sensitivity analysis of the difference between the two losses with respect to s is in Figure 7. Notably the HMM losses are always greater than the iHMM ones (i.e., their difference is positive). As soon as s increases, the difference decreases for the letter position T equal to 3 or 4, being almost stable for $T = 2$. An explanation is the stronger effect of s on the determinacy for $T > 2$ compared to the case $T = 2$ (see Table 4).

We also investigate the sensitivity of the difference between the losses with respect to the size of the training set (see Figure 8). For $s = 2$ (plot on the left), the difference increases for larger training sets, i.e., the iHMM takes more advantage from the higher availability of training data than the HMM does. This is less noticeable for $T = 2$. As in the previous case we relate this to the higher determinacy level for this position. For $s = 27$ (plot on the right), the behaviour is not monotone. A deeper analysis reveals that the iHMM starts to benefit of the higher data availability only with large (say > 25000) data sets, having an almost constant loss with fewer data. This is not the case of the HMM, which constantly reduces its losses with increasing amounts of data. An explanation is that, with few learning data,

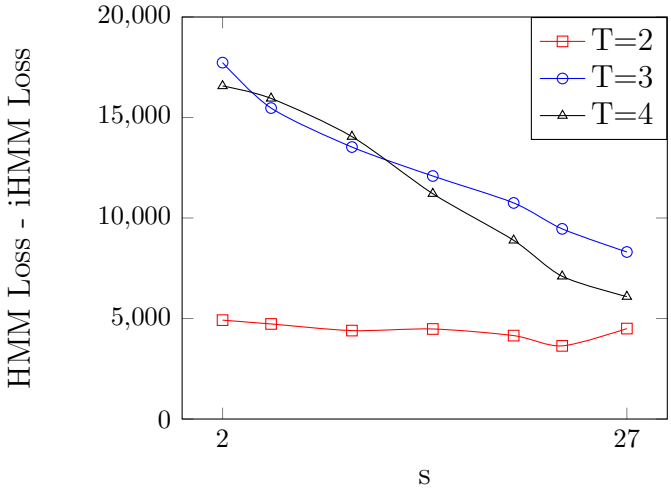


Figure 7: Difference between HMM and iHMM losses for different values of s.

IDM-based credal sets with $s = 27$ might be very large, thus producing the same inferences even if new data are available.

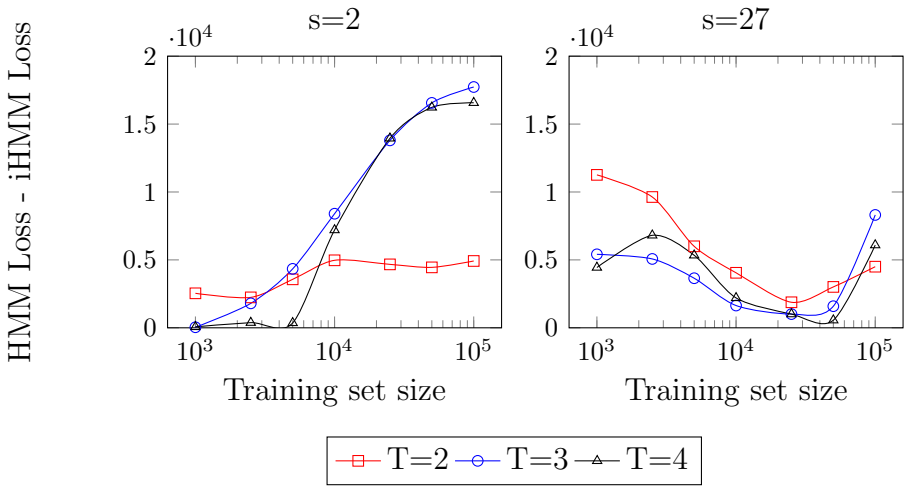


Figure 8: Difference between HMM and iHMM losses for different training set sizes and two values of s.

We emphasize however that our goal in this section is only to illustrate a scenario where iHMM might be useful, and not to defend it as a gen-

eral better predictor than the HMMs for this problem. For reaching the latter conclusion, we should extend the analysis to multiple data sets and experiment with a rejection rule for the HMM (that is, a threshold for the probability values for which only suggestions achieving higher probability are displayed). Given our goal in this work of introducing iHMM and presenting its capabilities, we leave a more detailed empirical study for the future.

8.5. Most Likely Sequence: Part-of-Speech Tagging

We evaluated the ability of iHMMs to discriminate between reliable and non reliable state sequences by comparing joint maximin and maximax state sequences in a part-of-speech (PoS) tagging task [2].

We performed experiments with reduced versions of two common data sets used for PoS that are freely available on the *nltk* package distribution.[11] The reduced versions of the Brown and Penn data sets contain, respectively, 38 and 31 distinct syntactic tags and about 5500 and 3500 distinct words, and allowed us to exploit the performance of HMMs with small training samples, where the impact of single probability distributions learned from data is greater. The interval-based credal sets were learned using local IDMs (with hyperparameter $s = 1$). Tags not occurring in the training data were omitted from the model (instead of having vacuous credal sets). Words appearing less than 4 times in the training data were collapsed into a single term and used for estimating the probability (or probability interval) of unseen words during classification. Coherently, the precise HMMs were learned with maximum likelihood smoothed by Perks' priors (with $s = 1$), and the same preprocessing steps. In order to assess the difficulty of the task we also performed tests with a simple unigram tagger. Figure 9 reports the results of numerical simulations for 5-fold cross validation and increasing size of the training set for the two data sets.

We evaluated the ability of discriminating reliable and non-reliable taggings by comparing the average accuracy of the predictions (PoS tags) of the precise HMM in the full test set and in two partitions of test set. The first, denominated *match set* consisted only of tokens for which the maximin and maximax criteria provided the same sequence. The second, denominated *mismatch set*, consisted of instances where maximin and maximax disagreed. According to our rationale, the accuracy of the precise should be much greater

[11] `http://www.nltk.org`

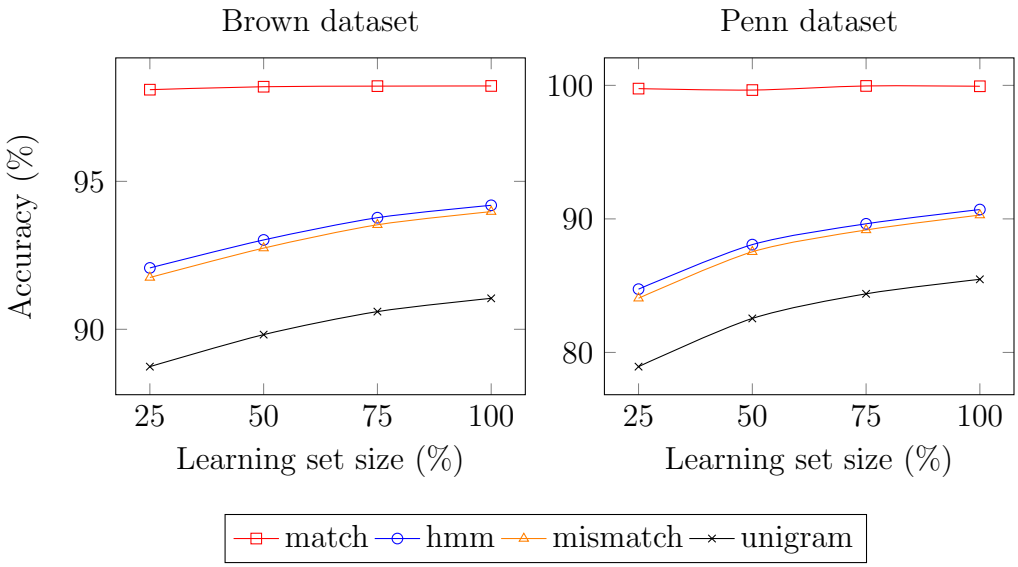


Figure 9: Results of the part-of-speech tagging experiments. Left and right plots show tagging accuracy for different criteria in the Brown and Penn data sets, respectively.

on the match set than on the mismatch set, as the former consisted of reliable predictions of the precise HMM. The results show that the predictions in the match set are considerably superior to the predictions in the mismatch set in both data sets. The rate of agreement, represented in the graph as the distance between the full set and the match set accuracy, was low in both data sets, indicating the unreliability of taggings generated by precise HMMs learned from very few data.

9. Conclusion

Imprecise hidden Markov models (iHMMs) are an extension of standard HMMs that arguably allows for proper handling of the imprecision in the parameters that arise in many domains. In this paper, we presented algorithms for standard usage such as computing likelihood, performing filtering/prediction, and finding optimistic and pessimistic most likely state sequences. When the parameters are specified as interval-valued probabilities, all algorithms run in quadratic time in the number of states and linear in the number of time steps. Remarkably, this is the same time complexity

of the analogous algorithms for standard HMMs. When imprecision takes a more complex form (e.g. as sets of linear inequalities), the time complexity grows only by the time of solving a small linear program of size linear in the input.

Experiments with real data showed that iHMMs can be used as "cautious" classifiers that suspend decision making when there is not enough statistical evidence to confidently support a decision. In addition, iHMMs can serve as valuable tools to perform analysis of the sensitivity of precise (non-stationary) HMMs to variations of the parameters.

The imprecision in the numerical parameters of the model translates to indeterminacy when using the models to make decisions as in the applications we show. We have adopted here interval dominance as the base criterion to suspend judgment. The literature counts with other criteria that are worth evaluating such as maximality and E-admissibility. Implementing such criteria will require developing efficient algorithms. We leave that as future work.

Acknowledgment

This work was partly supported by the São Paulo Research Foundation (FAPESP) grant no. 2013/23197-4 and the Swiss NSF grant no. 200021_146606 / 1.

We thank Marco Zaffalon for suggesting us the idea of using expected counts in the imprecise Dirichlet model to learn credal sets from incomplete data. We also thank Rocco De Rosa for his Matlab implementation of the Baum-Welch algorithm.

References

[1] L. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, IEEE 77 (2) (1989) 257–286.

[2] C. Manning, H. Schütze, Foundations of Statistical Natural Language Processing, MIT Press, 1999.

[3] L. Gorelick, M. Blank, E. Shechtman, M. Irani, R. Basri, Actions as space-time shapes, Transactions on Pattern Analysis and Machine Intelligence 29 (12) (2007) 2247–2253.

[4] D. Nur, D. Allingham, J. Rousseau, K. Mengersen, R. McVinish, Bayesian hidden Markov model for DNA sequence segmentation: A prior sensitivity analysis, Computational Statistics and Data Analysis 53 (2009) 1873–1882.

[5] P. Walley, Statistical Reasoning with Imprecise Probabilities, Chapman and Hall, 1991.

[6] M. Zaffalon, E. Miranda, Conservative inference rule for uncertain reasoning under incompleteness, Journal of Artificial Intelligence Research 34 (2009) 757–821.

[7] J. Kwisthout, L. van der Gaag, The computational complexity of sensitivity analysis and parameter tuning, in: Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI), 2008, pp. 349–356.

[8] A. Benavoli, M. Zaffalon, E. Miranda, Reliable hidden Markov model filtering through coherent lower previsions, in: Proceedings of the 12th International Conference on Information Fusion (FUSION), 2009, pp. 1743–1750.

[9] I. Levi, The Enterprise of Knowledge, MIT Press, 1980.

[10] P. Walley, Inferences from multinomial data: Learning about a bag of marbles, Journal of the Royal Statistical Society. Series B (Methodological) 58 (1) (1996) 3–57.

[11] M. Zaffalon, G. Corani, D. D. Mauá, Evaluating credal classifiers by utility-discounted predictive accuracy, International Journal of Approximate Reasoning 53 (8) (2012) 1282–1301.

[12] H. Chuan, A. Darwiche, Sensitivity analysis in Bayesian networks: From single to multiple parameters, in: Proceedings of the Twentieth Conference on Uncertainty in Artificial Intelligence (UAI), 2004, pp. 67–75.

[13] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.

[14] F. Cozman, Credal networks, Artificial Intelligence 120 (2000) 199–233.

[15] D. Mauá, C. de Campos, A. Benavoli, A. Antonucci, Probabilistic inference in credal networks: New complexity results, Journal of Artificial Intelligence Resesearch 50 (2014) 603–637.

[16] C. de Campos, F. Cozman, The inferential complexity of Bayesian and credal networks, in: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), 2005, pp. 1313–1318.

[17] E. Fagiuoli, M. Zaffalon, 2U: An exact interval propagation algorithm for polytrees with binary variables, Artificial Intelligence 106 (1) (1998) 77–107.

[18] M. Zaffalon, E. Fagiuoli, Tree-based credal networks for classification, Reliable Computing 9 (6) (2003) 487–509.

[19] D. Mauá, C. de Campos, M. Zaffalon, Updating credal networks is approximable in polynomial time, International Journal of Approximate Reasoning 53 (8) (2012) 1183–1199.

[20] A. Antonucci, Y. Sun, C. de Campos, M. Zaffalon, Generalized loopy 2U: A new algorithm for approximate inference in credal networks, International Journal of Approximate Reasoning 55.

[21] A. Cano, J. Cano, S. Moral, Convex sets of probabilities propagation by simulated annealing, in: Proceedings of the Fith International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU), 1994, pp. 4–8.

[22] A. Cano, S. Moral, A genetic algorithm to approximate convex sets of probabilities, in: Proceedings of the Fith International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU), 1996, pp. 859–864.

[23] A. Antonucci, C. de Campos, D. Huber, M. Zaffalon, Approximate credal network updating by linear programming with applications to decision making, International Journal of Approximate Reasoning 58 (2015) 25–38.

[24] G. de Cooman, F. Hermans, A. Antonucci, M. Zaffalon, Epistemic irrelevance in credal nets: the case of imprecise Markov trees, International Journal of Approximate Reasoning 51 (9) (2010) 1029–1052.

[25] D. Mauá, C. de Campos, A. Benavoli, A. Antonucci, On the complexity of strong and epistemic credal networks, in: Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI), 2013, pp. 391–400.

[26] J. De Bock, G. de Cooman, An efficient algorithm for estimating state sequences in imprecise hidden Markov models, Journal of Artificial Intelligence Research 50 (2014) 189–233.

[27] C. De Boom, J. De Bock, A. V. Camp, G. de Cooman, Robustifying the Viterbi algorithm, in: L. C. van der Gaag, A. J. Feelders (Eds.), Proceedings of the 7th Conference on Probabilistic Graphical Models (PGM 2014), Lecture Notes in Computer Science 8754, 2014, pp. 160–175.

[28] J. De Bock, C. de Campos, A. Antonucci, Global sensitivity analysis for MAP inference in graphical models, in: Advances in Neural Information Processing Systems 27 (NIPS), 2014, pp. 2690–2698.

[29] I. Kozine, L. Utkin, Interval-valued finite Markov chains, Reliable Computing 8 (2) (2002) 97–113.

[30] G. de Cooman, F. Hermans, E. Quaeghebeur, Imprecise Markov chains and their limit behavior, Probability in the Engineering and Informational Sciences 23 (2009) 597–635.

[31] D. Škulj, Discrete time Markov chains with interval probabilities, International Journal of Approximate Reasoning 50 (8) (2009) 1314–1329.

[32] D. Škulj, R. Hable, Coefficients of ergodicity for imprecise Markov chains, in: Proceedings of the Sixth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA), 2009, pp. 377–386.

[33] R. Crossman, P. Coolen-Schrijner, D. Škulj, F. Coolen, Imprecise Markov chains with an absorbing state, in: Proceedings of the Sixth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA), 2009, pp. 119–128.

[34] A. Antonucci, A. Benavoli, M. Zaffalon, G. de Cooman, F. Hermans, Multiple model tracking by imprecise Markov trees, in: Proceedings of

the 12th International Conference on Information Fusion (FUSION), 2009.

[35] A. Antonucci, R. De Rosa, A. Giusti, Action recognition by imprecise hidden markov models, in: Proceedings of the 2011 International Conference on Image Processing, Computer Vision and Pattern Recognition (IPCV), 2011, pp. 474–478.

[36] A. van Camp, G. de Cooman, A new method for learning imprecise hidden Markov models, in: Advances in Computational Intelligence, Vol. 299, 2012, pp. 460–469.

[37] A. Antonucci, R. De Rosa, A. Giusti, F. Cuzzolin, Temporal data classification by imprecise dynamical models, in: F. Cozman, T. Denoeux, S. Destercke, T. Seidenfeld (Eds.), Proceedings of the Eighth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA), 2013, pp. 13–22.

[38] T. Augustin, F. Coolen, G. de Cooman, M. Troffaes (Eds.), Introduction to Imprecise Probabilities, Wiley, 2014.

[39] B. Korte, J. Vygen, Combinatorial Optimization: Theory and Algorithms, Algorithms and Combinatorics, Springer, 2012.

[40] A. Antonucci, C. de Campos, D. Huber, M. Zaffalon, Approximating credal network inferences by linear programming, in: L. van der Gaag (Ed.), Proceedings of the 12th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU), Vol. 7958, Springer, 2013, pp. 13–25.

[41] G. Forney, The Viterbi algorithm, Proceedings of the IEEE 61 (3) (1973) 268–278.

[42] A. Piatti, A. Antonucci, M. Zaffalon, Building knowledge-based expert systems by credal networks: a tutorial, Vol. 11, Nova Science Publishers, New York, 2010.

[43] A. Antonucci, R. De Rosa, A. Giusti, F. Cuzzolin, Robust classification of multivariate time series by imprecise hidden Markov models, International Journal of Approximate Reasoning 56 (B) (2015) 249–263.

[44] A. Darwiche, Modeling and Reasoning with Bayesian Networks, Cambridge University Press, 2009.

[45] M. Zaffalon, G. Corani, D. Mauá, Evaluating credal classifiers by utility-discounted predictive accuracy, International Journal of Approximate Reasoning 53 (8) (2012) 1282–1301.

[46] M. Kudo, J. Toyama, M. Shimbo, Multidimensional curve classification using passing-through regions, Pattern Recognition Letters 20 (1999) 1103–1111.

[47] L. Gorelick, M. Blank, E. Shechtman, M. Irani, R. Basri, Actions as space-time shapes, Transactions on Pattern Analysis and Machine Intelligence 29 (12) (2007) 2247–2253.

[48] R. Chaudhry, A. Ravichandran, G. Hager, R. Vidal, Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions, in: In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.

Appendix A. An Empirical Comparison of State Sequences Selected Using the Joint and Posterior Distributions

As briefly discussed in Section 7, in the imprecise setting, state sequences provided by the extreme probability distributions in the posterior credal set $\mathbb{K}_{Q_{1:T}}^{o_{1:T}}$ may differ from the sequences provided by the extreme probability distributions in the joint credal set $\mathbb{K}_{Q_{1:T},O_{1:T}}$, due to the dependence of the likelihood on the denominator. However, some preliminary empirical results show that the effect of the likelihood in selecting the state sequence is small, making joint and posterior state sequences coincide in a great majority of cases, even under severe lack of data.

To test the divergence between posterior and joint state sequences we ran the following experiment consisting of several trials. For each trial, a precise HMM was randomly sampled and used to sample training and test sets of state and observation sequences. An iHMM was then learned from the training data and used to provide maximin and maximax sequence for the test set. The sequences for the posterior credal set were obtained by enumerating all possible state sequences and using an algorithm similar to

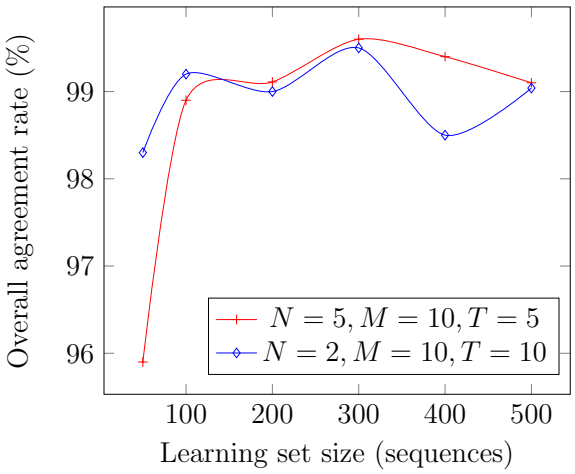


Figure A.10: Agreement rate of reliability of state sequences between posterior and joint probability.

the likelihood algorithms to compute lower and upper posterior probabilities, which limited generated models to small number of states and small length. For each configuration of N, M and T tested, 30 different model and data sets were generated.

Figure A.10 shows the agreement between joint and posterior state sequences as a function of the learning set size. The numbers in the y-axis indicate the proportion of cases where joint and posterior agreed about the reliability of the predictions, that is, where either the joint maximin and maximax differed and the posterior maximin and maximax also differed or the joint maximin and the maximax agreed and the posterior maximin and maximax also agreed. The curves show that joint and posterior sequences agree, on average, in approximately 99% of the cases.