An introduction to Hidden Markov Models

Christian Kohlschein

Abstract

Hidden Markov Models (HMM) are commonly defined as stochastic finite state machines. Formally a HMM can be described as a 5-tuple $\Omega = (\Phi, \Sigma, \pi, \delta, \lambda)$. The states Φ, in contrast to regular Markov Models, are *hidden*, meaning they can not be directly observed. Transitions between states are annotated with probabilities δ, which indicate the chance that a certain state change might occur. These probabilities, as well as the starting probabilities π, are discrete. Every state has a set of possible emissions Σ and discrete/continuous probabilities λ for these emissions . The emissions can be observed, thus giving some information, for instance about the most likely underlying hidden state sequence which led to a particular observation. This is known as the Decoding Problem. Along with the Evaluation and the Learning Problem it is one of three main problems which can be formulated for HMMs. This paper will describe these problems, as well as the algorithms, like the Forward algorithm, for solving them. As HMMs have become of great use in pattern recognition, especially in speech recognition, an example in this field will be given, to help understand where they can be utilized. The paper will start with an introduction to regular Markov Chains, which are the base for HMMs.

1 Introduction

This paper gives an introduction into a special type of stochastic finite state machines, called Hidden Markov Models (HMMs). Nowadays HMMs are commonly used in pattern recognition and its related fields like computational biology. Towards an understanding of HMMs the concept of Markov Chains is fundamental. They are the foundation for HMMs, thus this paper starts with an introduction into Markov Chains in section 2. Section 3 addresses HMMs, starting with a formal definition in 3.1. After an example of a HMM is given in 3.2, section 3.3 continues with a description of the standard problems which can be formulated for HMMs. Section 3.4 describes the algorithms which can be used to tackle these problems. Finally section 4 closes with an example of an actual application of HMMs in the field of speech recognition.

2 Markov Chains

This section introduces Markov Chains, as well as the necessary definitions like stochastic process and Markov property.

2.1 Definition

Let (Ω, Σ, P) be a probability space and $(S, Pot(S))$ a measurable space. A set X of stochastic variables $\{X_t, t \in T\}$ defined on the probability space, taking values $s \in S$ and indexed by a non empty index set T is called a **stochastic process**. If T is countable, for instance $T \subseteq \aleph_0$, the process is called time discrete, otherwise time continuous. This section only addresses time discrete stochastic processes. A stochastic process which fulfils the **Markov property**:

$$P(X_{t+1} = s_{t+1} \mid X_t = s_t) = P(X_{t+1} = s_{t+1} \mid X_t = s_t, X_{t-1} = s_{t-1}, \ldots, X_0 = s_0) \quad (1)$$

is called a first order **Markov chain**. The Markov property states that the probability of getting into an arbitrary state at time $t+1$ only depends upon the current state at time t, but not on the previous states. A stochastic process which fulfils:

$$P(X_{t+1} = s_{t+1} \mid X_t = s_t, \ldots, X_n = s_n) = P(X_{t+1} = s_{t+1} \mid X_t = s_t, X_{t-1} = s_{t-1}, \ldots, X_0 = s_0) \quad (2)$$

is called a n-th order Markov chain. In this process the probability of getting into the next state depends upon the n previous states. Commonly the term Markov chain is used as a synonym for a first order Markov chain.

For the following consideration it is assumed that the chains are **time-homogeneous**:

$$p_{ij} := P(X_{t+1} = i | X_t = j) = P(X_t = i | X_{t-1} = j) \; \forall t \in T, \; \forall i, j \in S \quad (3)$$

This means that transition probabilities between states are constant in time. Vice versa in non-time-homogeneous Markov chains p_{ij} may vary over time. Time-homogeneous chains are often called homogeneous Markov chains.
For a homogeneous Markov chain the transition probabilities can then be noted in a time independent stochastical matrix M:

$$M = (p_{ij}), \; p_{ij} \geq 0 \; \forall i, j \in S \text{ and } \sum_{j \in S} p_{ij} = 1, \; (i \in S) \quad (4)$$

M is called the **transition matrix**. Along with the **initial distribution vector** π:

$$\pi = (\pi_i, i \in S), \text{with } \pi_i = P(X_0 = i) \quad (5)$$

it follows that the common distribution of the stochastic variables is well-defined, and can be computed as:

$$P(X_0 = s_0, \ldots, X_t = s_t) = \pi_{s_0} p_{s_0 s_1} p_{s_1 s_2} \ldots p_{s_{t-1} s_t} \quad (6)$$

It can be shown that the probability of getting in m steps to state j, starting from state i:

$$p_{ij}^m := P(X_{t+m} = j | X_t = i) \quad (7)$$

can be computed as the m-th power of the transition matrix:

$$p_{ij}^m = M^m(i, j) \quad (8)$$

Recapitulating, a first-order time-homogeneous Markov Chain can be defined as a 3-tuple, consisting of the set of states S, the transition matrix M and the initial distribution vector π:

$$\theta = (\mathrm{S}, \mathrm{M}, \pi) \quad (9)$$

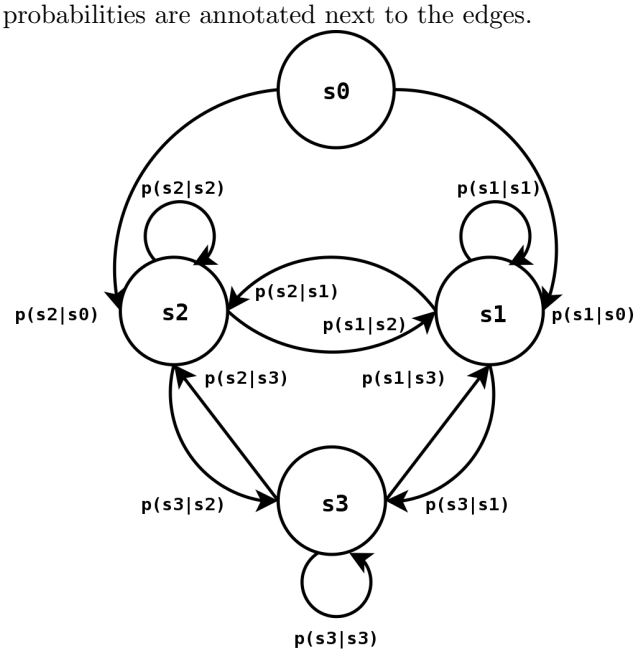
An example of a Markov Chain, represented by a directed graph, is shown in figure 1. The following section shows an example of the appliance of Markov Chains.

2.2 A common example

Although there is a wide range of tasks where Markov Chains can be applied, a common simple example of a time-discrete homogeneous Markov Chain is a weather forecast. For a more fancy application of Markov chains the interested reader is referred to the famous PageRank[1] algorithm used by Google. It gives a nice example of utilizing Markov Chains to become a billionaire.

[1] http://dbpubs.stanford.edu:8090/pub/1998-8

Figure 1: Example of a Markov Chain, represented by a directed graph. Starting state is s_0, and the conditional transition probabilities are annotated next to the edges.



Example 1

Let S be a set of different weather conditions:

$$S = \{sunny, overcast, rainy\}$$

The weather conditions can now be represented in a transition matrix, where the different entries are representing the possibility of a weather change. Note that the transition matrix is a stochastical matrix, thus the row entries sum up to 1.

Let M_{Aachen} be the transition matrix:

$$M_{\text{Aachen}} = \begin{pmatrix} & sunny & overcast & rainy \\ sunny & 0.1 & 0.2 & 0.7 \\ overcast & 0.2 & 0.2 & 0.6 \\ rainy & 0.1 & 0.1 & 0.8 \end{pmatrix}$$

For instance, in this transition matrix the chance of sunny weather after a rainy day is 0.1 % and the chance that the rain continues on the next day is 0.8 %.[2] Along with the initial distribution vector $\pi = (P(\text{sunny}), P(\text{overcast}), P(\text{rainy}))$:

$$\pi = (0.2, 0.3, 0.5)$$

the specific Markov chain can now be denoted as a 3-tuple

$$\theta_{\text{Ac}} = (S, M_{\text{Aachen}}, \pi)$$

A common application of this Markov model would be that someone is interested in the probability of

[2]It shall be mentioned, that the Markov Chain of the weather conditions in Aachen is a good example of a homogeneous Markov Chain. The probabilities of a weather change stay constant over the year, thus there is no season dependent transition matrix like M(spring,summer,autumn,winter).

a certain weather sequence, i.e. the chance for the sequence (sunny,sunny,overcast). Due to equation 6, this probability computes to:

$$P(X_0 = \text{sunny}, X_1 = \text{sunny}, X_2 = \text{overcast}) = \pi_{\text{sunny}} p(\text{sunny} \rightarrow \text{sunny}) p(\text{sunny} \rightarrow \text{overcast}) = 0.004\,\%$$

Note that these probabilities can become very small quite fast, so that one has to find a way to ensure the numeric stability of the computation. Commonly this is done by computing the probabilities in log-space.
This introductory section about Markov chains concludes with the remark that Markov chains are a good way to model stochastic procedures which evolve over time. For a more detailed introduction into Markov chains, the reader is referred to [Kre].

3 Hidden Markov Models

This section starts with a formal definition of Hidden Markov Models. Afterwards an example for the appliance of HMMs is given.

3.1 Definition

Let $\theta = (\mathrm{S}, \mathrm{M}, \pi)$ be a first-order time-homogeneous Markov Chain, as it was introduced in section 2. Now it is assumed that the states S of the Markov chain can not be directly observed at time t, thus $s(t)$ is hidden. Instead of that it is assumed that in every point in time the system emits some symbol v with a certain probability. This property can be considered as an additional stochastical process which is involved. The emitted symbol v can be observed and thus $v(t)$ is visible. The probability for such an **emission** at time t depends only upon the underlying state s at that time, so it can be denoted as the conditional probability $p(v(t)|s(t))$. With these properties a **Hidden Markov Model** (HMM) can now be introduced formally as a 5-tuple:

$$\vartheta = (\mathrm{S}, \mathrm{M}, \Sigma, \delta, \pi) \tag{10}$$

Following the definition of a Markov Chain in section 2, S, M and π keep their meanings. The set of emission symbols Σ can be discrete and in this the case the emission probabilities δ could be denoted in a stochastical matrix where each entry represents the chance for a certain emission, given a certain state. If the set of emission symbols is continuous these probabilities are modeled through a probability density function, for instance a Gaussian distribution. Independent of the models emission probabilities, i.e. if they are discrete or continuous it is important to notice that they sum up to 1, given a certain state s(t). Figure 2 shows an example of an Hidden Markov Model, represented by a directed graph.
This section continues with a recapitulation of the weather forecast example given in section 2.

3.2 Weather speculations with Hidden Markov Models

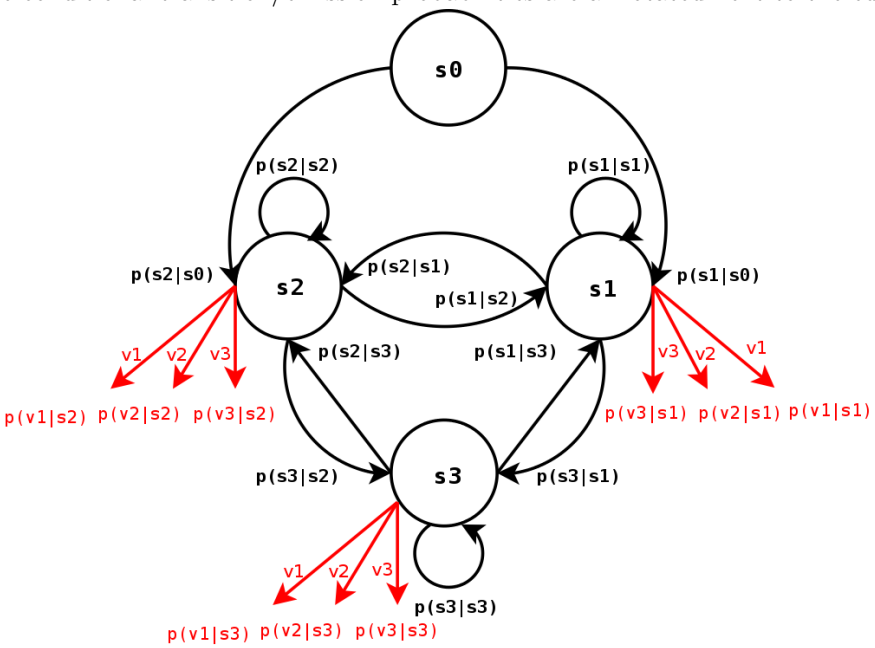
As stated in the above definition the states of a HMM are hidden and can only be indirectly observed by emissions, but what does this mean? The following example of a HMM with discrete emission probabilities tries to bring this to light.

Example 2

Let Alan be a computer scientist who lives in an apartment somewhere in Aachen, which has no direct connection to the outside world, i.e. it has no windows. Although Alan has no deeper interest in the outside world he is interested in its weather conditions. Since he is a computer scientist it is reasonable for him to assume that the change of the weather conditions over time can be described through a Markov Chain. Let

$$\theta_{\text{Ac}} = (S, M_{\text{Aachen}}, \pi)$$

Figure 2: Example of a Hidden Markov Model with three possible emission v_1, v_2, v_3. Starting state is s_0, and the conditional transition/emission probabilities are annotated next to the edges.



be the Markov Chain of example 1. Due to the fact that his flat has no possibilities of observing the current state of the weather (the state is hidden to him) his only chance of getting information about the weather is to look at his cat Knuth. Knuth daily leaves and accesses the apartment through a hatch in the door. Depending on the current weather and because it is a computer scientist cat, Knuths fur chances between only two states :

$$\Sigma = \{wet, dry\}$$

These emissions can now be observed by Alan. Additionally Alan also knows the chances of his cats fur being in one of the states, depending on the current weather. Thus the emission probabilities are also known to him, and can be denoted in a stochastical matrix δ:

$$\delta = \begin{pmatrix} & dry & wet \\ sunny & 0.7 & 0.3 \\ overcast & 0.5 & 0.5 \\ rainy & 0.1 & 0.9 \end{pmatrix}$$

Alan now augments the Markov Chain θ_{Ac}, given in example 1, with Σ and δ. The outcome is a Hidden Markov Model for Aachen:

$$\theta_{\text{Ac}} = (S, M_{\text{Aachen}}, \Sigma, \delta, \pi)$$

Alan can now use this specific HMM to guess what the weather conditions might have been over the last few days. Every day he records the state of Knuths fur and after a while he has a set, actual a sequence of these observations. Based on his notes he can now try to determine the most likely sequence of underlying weather states which led to this specific emission sequence. This is known as the **Decoding Problem** and it is one of the three standard problems which can be formulated for HMMs. Along with the algorithm for solving it and two other standard problems, it will be elaborated in the following section.

3.3 Standard problems for Hidden Markov Models

As stated in the previous example three problems can be formulated for HMMs:

- **The Decoding Problem**:
  Given a sequence of emissions V^T over time T and a HMM with complete model parameters, meaning that transition and emission probabilities are known, this problem asks for the most probable underlying sequence S^T of hidden states that led to this particular observation.
- **The Evaluation Problem**:
  Here the HMM is also given with complete model parameters, along with a sequence of emissions V^T. In this problem the probability of a particular V^T generally to be observed under the given model has to be determined.
- **The Learning Problem**:
  This problem differs from the two above mentioned problems in the way that only the elemental structure of the HMM is given. Given one or more output sequences, this problem asks for the model parameters M and δ. In other words: The parameters of the HMM have to be *trained*.

3.4 Algorithms for the standard problems

This section introduces the three main algorithms for the solution of the standard problems. The algorithms in pseudo code are taken from [DHS].

3.4.1 Forward algorithm

The Forward algorithm is a quite efficient solution for the Evaluation Problem.
Let $S_r^T = \{s(1), \ldots, s(T)\}$ be a certain sequence of T hidden states indexed by r. Thus, if there are c hidden states and they are fully connected among each other, there is a total of $r_{\max} = c^T$ possible sequences of length T. Since the underlying process in a HMM is a first-order Markov Chain where the probability of the system being in certain state $s(t)$ at time t only depends on its predecessor state $s(t-1)$, the probability of such a sequence r derives as:

$$p(s_r^T) = \prod_{t=1}^{T} p(s(t)|s(t-1)) \tag{11}$$

Let V^T be the sequence of emissions over time T. As mentioned, the probability for a certain emission to be observed at time t depends only on the underlying state $s(t)$, and it derives as $p(v(t)|s(t))$. Thus the probability for a sequence V_T to be observed given a certain sequence s_r^T derives as:

$$p(V^T|s_r^T) = \prod_{t=1}^{T} p(v(t)|s(t)) \tag{12}$$

and hence the probability for having the hidden state sequence s_r^T while observing sequence V^T given this sequence is:

$$p(V^T|s_r^T) \cdot p(s_r^T) = \prod_{t=1}^{T} p(v(t)|s(t)) \cdot p(s(t)|s(t-1)) \tag{13}$$

As mentioned, there are $r_{\max} = c^T$ possible hidden state sequences in a model, where all transitions are allowed and thus the probability for observing V^T given this model finally derives as:

$$p(V^T) = \sum_{t=1}^{r\max} \prod_{t=1}^{T} p(v(t)|s(t)) \cdot p(s(t)|s(t-1)) \tag{14}$$

Since the precondition in the Evaluation Problem is that the HMM is given with complete model parameters, the above equation could be evaluated straight forward. Nevertheless this is prohibitive, because the computational complexity is $O(c^T \cdot T)$. So, there is a need for a more efficient algorithm. In fact there is a approach with a complexity of $O(c^2 \cdot T)$, the Forward algorithm. It is derived from the observation, that in every term $p(v(t)|s(t)) \cdot p(s(t)|s(t-1))$ only $v(t)$,$s(t)$ and $s(t-1)$ are necessary. Hence the probability $p(V^T)$ can be computed recursively.
Let $a_{ij} = p(s_j(t)|s_i(t-1)$ be a transition probability, and $b_{jk} = p(v_k(t)|s_j(t))$ be a emission probability. The probability of the HMM being in state s_j at time t and having generated the first t emission of V^T will now defined as:

$$\alpha_j(t) = \begin{cases} 0, & \text{t = 0 and } j \neq \text{initial state} \\ 1, & t = 0 \text{ and } j = \text{initial state} \\ \left[\sum_i \alpha_i(t-1)a_{ij}\right] b_{jk}v(t) & \text{otherwise} \end{cases} \tag{15}$$

Here $b_{jk}v(t)$ means the emission probability selected by v(t) at time t.
The Forward algorithm can now be denoted in pseudo code:

Algorithm 1: Forward Algorithm
1 **init** t:=0, a_{ij}, b_{jk}, observed sequence V^T, $\alpha_j(0)$
2 **for** t:=t+1
3 $\alpha_i := \left[\sum_{i=1}^{c} \alpha_i(t-1)a_{ij}\right] b_{jk}v(t)$
4 **until** t=T
5 **return** $p(V^T) := \alpha_0(T)$ for the final state
6 **end**

The probability of the sequence ending in the known final state is denoted by α_0 in line 5.

3.4.2 Decoding algorithm

The Decoding problem is solved by the Decoding algorithm, which is sometimes called Viterbi algorithm. A naive approach for the Decoding problem would be to consider every possible path and to observe the emitted sequences. Afterwards the path with the highest probability that yield V^T would be chosen. Nevertheless this would be highly ineffective, because it is an $O(c^T \cdot T)$ calculation. A more effective and quite simple approach is the Decoding Algorithm given below in pseudo code:

Algorithm 2: Decoding Algorithm
1 **begin init** Path:={},t:=0
2 **for** t:=t+1
3 j:=j+1
4 **for** j:=j+1
5 $\alpha_j(t) := \left[\sum_{i=1}^{c} \alpha_i(t-1)a_{ij}\right] b_{jk}v(t)$
6 **until** j=c
7 $\hat{j} := \underset{j}{\operatorname{argmax}}\ \alpha_j(t)$
8 Append $s_{\hat{j}}$ to Path
9 **until** t=T
10 **return** Path
11 **end**

This algorithm is structural quite similar to the Forward Algorithm. In fact, they can both be implemented in one algorithm. A implementation of such an algorithm for evaluation/decoding in the python programming language can be found in the Wikipedia[3]

[3] http://en.wikipedia.org/wiki/Forward_algorithm

3.4.3 Baum-Welch algorithm

The Baum-Welch algorithm, also known as Forward-Backward Algorithm, is capable of solving the Learning Problem. From a set of training samples it can iteratively learn values for the parameters a_{ij} and b_{jk} of an HMM. This values are not exact, but represent a good solution.
Analog to the definition of $\alpha_i(t)$, $\beta_i(t)$ is now defined as the probability that the model is in state $s_i(t)$ and will generate the remaining elements of the target sequence:

$$\beta_i(t) = \begin{cases} 0, & \mathrm{s}_i(t) \neq s_0(t) \text{ and } t = T \\ 1, & \mathrm{s}_i(t) = s_0(t) \text{ and } t = T \\ \sum_j \beta_j(t+1) a_{ij} b_{jk} v(t+1) & \text{otherwise} \end{cases} \tag{16}$$

With the definition given above $\beta_i(T)$ is either 0 or 1 and $\beta_i(T-1) = \sum_j \beta_j(T) a_{ij} b_{jk} v(T)$.
After the determination of $\beta_i(T-1)$ the process is repeated and $\beta_i(T-2)$ is computed. This iteration is repeated, while "travelling back in time".
The calculated values for $\alpha_i(t)$ and $\beta_i(t)$ are just estimates. For the calculation of an improved version of these estimates, the auxiliary quantity $\gamma_{ij}(t)$ is introduced and defined as:

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1) a_{ij} b_{jk} \beta_j(t)}{p(V^T|\theta)} \tag{17}$$

Here θ denotes the HMMs model parameters (a_{ij} and b_{jk}), and therefore $p(V^T|\theta)$ is the probability that the model generated V^T. Hence the auxiliary quantity is the probability of a transition from $s_i(t-1)$ to $s_j(t)$, under the condition that the model generated V^T.
Using the auxiliary quantity, an estimated version $\hat{a}_{ij}$ of a_{ij} can now be calculated by:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T} \gamma_{ij}(t)}{\sum_{t=1}^{T} \sum_k \gamma_{ik}(t)} \tag{18}$$

Similar an estimated version $\hat{b}_{jk}$ of b_{jk} can be derived:

$$\hat{b}_{jk} = \frac{\sum_{t=1, v(t)=v_k}^{T} \sum_l \gamma_{jl}(t)}{\sum_{t=1}^{T} \sum_l \gamma_{jl}(t)} \tag{19}$$

Informally spoken, the Baum-Welch algorithm starts with the training sequence V^T and some rough or estimated versions of the transition/emission probabilities and then uses equation 18 and 19 for the calculation of improved estimates. This is then repeated, until some convergence criterion is achieved, i.e. until there are only slight changes in succeeding iterations. Expressed in pseudo code:

Algorithm 3: Baum-Welch algorithm

1 **begin init** estimated versions of a_{ij} and b_{jk},V^T, convergence criterion c,z:=0
2 **do** z:=z+1
3 compute $\hat{a}(z)$ from a(z-1) and b(z-1) by Eq. 18
4 compute $\hat{b}(z)$ from a(z-1) and b(z-1) by Eq. 19
5 $a_{ij}(z) := \hat{a}_{ij}(z-1)$
6 $b_{jk}(z) := \hat{a}_{jk}(z-1)$
7 **until** convergence criterium achieved
8 **return** $a_{ij} =: a_{ij}(z)$ and $b_{jk} =: b_{jk}(z)$
9 **end**

4 Application in Speech Recognition

Hidden Markov Models are used in a range of applications in computer science. One of the fields where they are most commonly used is statistical pattern recognition. Here they have become expedient in

such fields like machine translation or gesture recognition. This section presents their application in the field of Speech Recognition (ASR) on the example of isolated word recognition.

The general task of pattern recognition can be formally expressed as the problem of finding a decision function g capable of mapping an input vector $\vec{x} \in X \subseteq \Re^n$, were X is called the feature space, to some classification label $c \in C$, were C is a set of classification labels. For instance $\vec{x}$ might be the multi-dimensional representation of an email (i.e., a dimension for the length, a dimension for the sender address etc.) and the classification label could be spam or not spam. The sought-after function g is usually derived from a set of already classified input vectors $\{\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\}$ called the **training data**.

In **statistical pattern recognition** the decision function can now be formulated as:

$$\hat{c} = g(\vec{x}) = \underset{c}{\operatorname{argmax}}\{p(c|\vec{x})\} \tag{20}$$

The equation states that the class $\hat{c}$ which is yield by this function is the one which maximizes the probability $p(c|\vec{x})$. Using **Bayes Decision Rule**, equation 11 can be rewritten as:

$$\hat{c} = g(\vec{x}) = \underset{c}{\operatorname{argmax}}\left\{\frac{p(\vec{x}|c) \cdot p(c)}{p(\vec{x})}\right\} \tag{21}$$

Since the denominator has no influence on the maximization process, it can be omitted and the decision function can be finally written as:

$$\hat{c} = g(\vec{x}) = \underset{c}{\operatorname{argmax}}\{p(\vec{x}|c) \cdot p(c)\} \tag{22}$$

This decision function can now be utilized in a stochastical based ASR system. For the task of isolated word recognition, the goal is to map an acoustic signal to a written word. The acoustic signal has first to be transformed into a sequence of **acoustic feature vectors** $x_1^T = (x_1 \ldots x_T)$ in a process called **feature extraction**. These feature vectors can be imagined as suitable representation of the speech signal for this task, and typically have 16-50 dimensions. The most probable word $\hat{w}$ belonging to this vector sequence can be decided using the above stated decision rule:

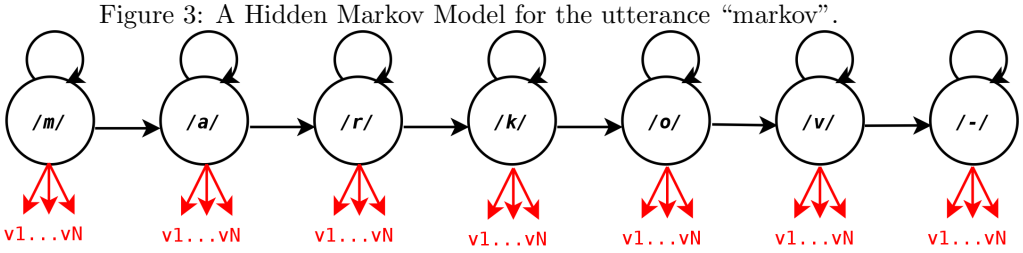
$$\hat{w} = \underset{w}{\operatorname{argmax}}\left\{p(x_1^T|w) \cdot p(w)\right\} \tag{23}$$

In isolated word recognition every word can now be represented by a Hidden Markov Model. The idea behind this is that a HMM gives a good representation of the task. In speech recognition the exact word, or more abstract, its components, can not be directly observed, because the only information given is the speech signal. In other words, the components are hidden. Every state of the HMM can thus be seen as a different component of the word. In contrast, the acoustic feature vectors can be observed and they can be seen as the emissions of the model. For instance, the utterance "markov" could be described by a 7 state HMM, as it can be seen in figure 3. This HMM is a left-to-right model, commonly used in speech recognition. It only allows transitions forward in time. Additionaly this model has a final silence state /-/, marking the end of the word.
The HMM shall now be denoted by θ, hence equation 14 is rewritten:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\left\{p(x_1^T|\theta) \cdot p(\theta)\right\} \tag{24}$$

As it can be seen in the above equation, the problem of finding the maximizing word sequence $\hat{w}$ is transformed to the problem of finding the maximizing HMM $\hat{\theta}$. This HMM approach brings up two questions, namely how to construct the HMMs for the different words, and how to estimate $p(\theta)$ respectively $p(x_1^T|\theta)$.
The HMMs for the different words are assembled using the Baum-Welch algorithm. As seen in section 3.4.3, this algorithm can estimate all the transition and emission probabilities which are necessary

Figure 3: A Hidden Markov Model for the utterance “markov”.



from training data.
The a-posteriori probability $p(x_1^T|\theta)$ for a certain HMM θ is yield by the forward algorithm, which gives the probability for observing x_1^T under this model. Finally the prior probability is commonly given by a so called language-model. In speech recognition the language model can contain information like the probability for a certain word given a predecessor, or the probability given a certain semantic context. In isolated word recognition it can be assumed that there is a uniform prior density for the distribution of θ, because no context information is available. Thus the prior probability could be omitted in the classification task within this example.

5 Conclusion

In this paper an extension of Markov Chains was introduced: Hidden Markov Models (HMMs).
After a short introduction in section 1 and revisiting Markov chains in section 2, HMMs were formally introduced in section 3.1. They were defined as a Markov Chain, with not directly observable states and an additional set of possible emissions, along with a set of corresponding emission probabilities. Next an example of a HMM was given in 3.2, and afterwards this paper continued in section 3.3 with a description of the standard problems, which can be formulated for Hidden Markov Models. Three problems were presented, namely the Decoding, Evaluation and the Learning Problem. In Section 3.4 the algorithms for the solution of these problems were described: The Forward, the Decoding and the Baum-Welch Algorithm. Section 4 finally gave an example of the actual application of Hidden Markov Models in the field of speech recognition.
This paper closes with the remark that HMMs are a good example for a theoretical concept where the direct application might be uncertain, at least at the first gaze. Nevertheless, over the years this concept has become an important part of such practical applications like for example machine translation, gesture recognition and, as presented in this paper, speech recogniton.

References

[DHS] R. Duda, P. Hart, and D. Stork. *Pattern Classification*. Wiley-Interscience, second edition. pages 128-139.

[Kre] Ulrich Krengel. *Einführung in die Wahrscheinlichkeitstheorie und Statistik*. Vieweg, second edition. pages 194-200.

[Ney06] Hermann Ney, editor. *Speech Recognition*. Chair of Computer Science 6, RWTH Aachen University, 2006.