

Artifacts from Combining Hidden Markov Models

Rune B. Lyngsø

Hidden Markov models (HMMs) have found wide spread use in bioinformatics. In short, an HMM consists of a set of states; each state has a probability distribution over what state to move to next when being in this state (often an HMM is drawn as a directed graph with nodes representing states and edges showing transitions from one state to another with non-zero probability, cf. e.g. Figure 1) and a probability distribution over what symbol is emitted whenever the state is visited (unless the state is silent, in which case no symbol is emitted when the state is visited). In bioinformatics and other areas where we are mainly interested in finite strings, HMMs will usually be equipped with a start state and an end state. One can now think of an HMM as a string generating stochastic machine that starts in the start state; as long as the current state is

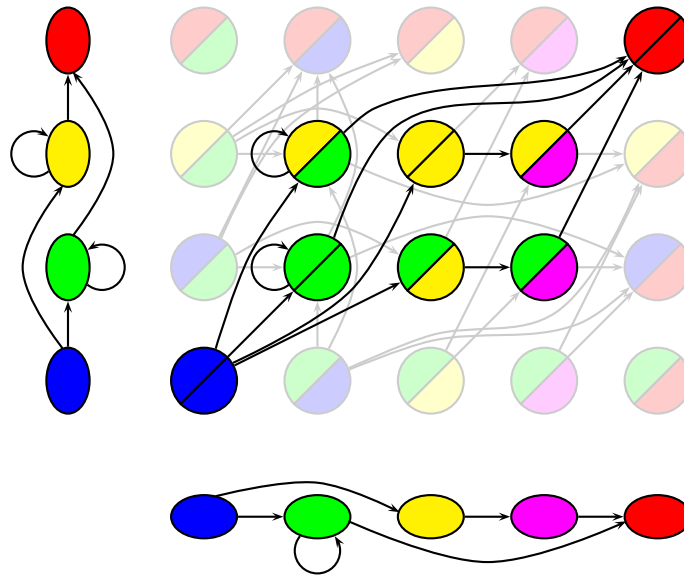


Figure 1: The transition structure of two simple HMMs with four and five states, respectively, are shown to the left and at the bottom. The transition structure of an HMM ‘combining’ them is shown above and to the right, with states to which it is not possible to get from the combined starts state or from which it is not possible to get to the combined end state faded. Blue states are start states and red states are end states.

not the end state, a symbol is emitted according to the current states symbol emission probability distribution and the next state is chosen according to the transition probability distribution of the current state; when the end state is reached, the sequence of symbols emitted is the generated string. Standard uses of HMMs are for annotating strings, where each symbol in the string is annotated with the state emitting it in the most probable way the HMM can generate the string, and for classifying strings, where the total probability of the HMM generating the string is used to assess whether it belongs to the class of strings modelled by the HMM. Efficient algorithms, known as the Viterbi algorithm and the forward algorithm, exist for solving the annotation and the classification problems. For a more extensive description of HMMs and their use in bioinformatics cf. [2].

Sometimes it is desirable to model more than one feature simultaneously. For example one could imagine modelling both secondary structure and surface accessibility for protein sequences. The purpose of this will usually be to improve the modelling power by taking more features into account at the same time, but can also be to simply study dependencies – or lack of same – between two features. The aim of this project is to investigate whether we by using standard techniques for simultaneous modelling of two or more features may inadvertently introduce artificial dependencies between these features.

In many such situations, HMMs modelling each of features individually will usually exist or at least be more readily obtained than a combined model. It is then natural to apply standard techniques from finite automata to obtain a combined model from the two individual models. Assume that we have HMMs M_1 and M_2 . To create the states and transition structure defining a combined model $M_{1 \times 2}$ we in essence take the Cartesian product: iff p_1 is a state in M_1 and p_2 is a state in M_2 , (p_1, p_2) becomes a state in $M_{1 \times 2}$; iff a transition between p_1 and q_1 is possible in M_1 and a transition between p_2 and q_2 is possible in M_2 , then a transition between (p_1, p_2) and (q_1, q_2) is possible in $M_{1 \times 2}$. This is illustrated in Figure 1.

This defines the general transition structure of the combined model, but not the actual probability distributions. If we have the probability distributions of the individual models, assuming independence there is again a natural way to combine things: we simply need to multiply probabilities from the two individual models. So if $\Pr(p_1 \rightarrow q_1)$ is the probability of choosing q_1 as next state when in state p_1 in M_1 and $\Pr(p_2 \rightarrow q_2)$ is the probability of choosing q_2 as next state when in state p_2 in M_2 , then the probability of choosing state (q_1, q_2) when in state (p_1, p_2) in $M_{1 \times 2}$ is simply defined as $\Pr((p_1, p_2) \rightarrow (q_1, q_2)) = \Pr(p_1 \rightarrow q_1) \cdot \Pr(p_2 \rightarrow q_2)$. This is illustrated in Figure 2.

One problem one can immediately spot with this approach to combining probability distributions is that it does not always yield a probability distribution in the combined model. For example the all-green state in the combined model in Figure 2 has probability of only $1/3$ of emitting a symbol and probability $5/9$ of choosing a next state. The problem stems from the loss in probability mass of ‘meaningless’ events, like emitting a symbol that is at the same time **a** and **b** and from the eliminated non-reachable states. Yet again we know of a simple natural way to resolve this problem: normalise the coefficients of the individual choices by their total sum to obtain a probability distribution. This is illustrated in Figure 3.

So it may seem that we have now described a rigorous way of combining two

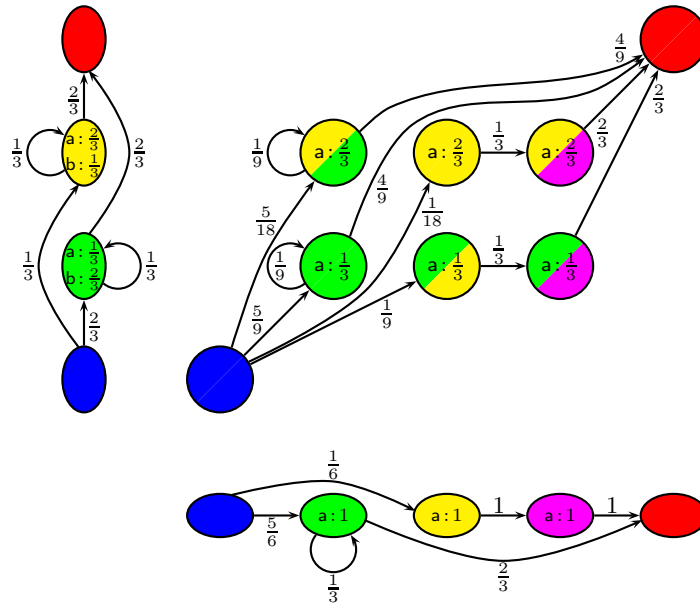


Figure 2: The HMMs from Figure 1 when probabilities are included and assumed independent. Non-reachable states (i.e. then ones faded in Figure 1) have been completely eliminated in this illustration. The start and end states silent, while the remaining states emit either a or b in the ‘vertical’ HMM and always emit a in the ‘horizontal’ HMM.

HMMs, assuming independence of the features they model. There is just one problem: by normalising we may inadvertently have messed up the ordering of the different annotations of a string. Take for instance the string aa . It has two possible annotations in each of the two individual HMMs. In the vertical HMM either both a ’s will be emitted by the green state or both a ’s will be emitted by the yellow state. In the horizontal HMM either both a ’s will be emitted by the green state or the first a will be emitted from the yellow state and the second a will be emitted from the purple state. The probabilities of any combination of these two pairs of annotations (also corresponding to the ‘probability’, or perhaps more correctly designated score, of the four possible annotations in the unnormalised combined HMM of Figure 2) is shown in the lefthand side of Figure 4. However, when scoring the annotations by the probabilities of the normalised combined HMM of Figure 3 we get the probabilities shown in the righthand side of Figure 4. A very undesirable effect of using the standard techniques for combining finite automata, combining independent probabilities and normalising to obtain a probability distribution is that the ordering of annotations can be completely reversed. This was initially observed for combining an HMM with a slightly more complex type stochastic models known as stochastic context-free grammars in [1].

That the ordering of annotations is reversed when not just the transition structure but also the probability distributions are attempted carried over to the combined model may not be to much of a concern. If all we wanted to do was model two features independently, there wouldn’t be a point in combining the models in the first place.

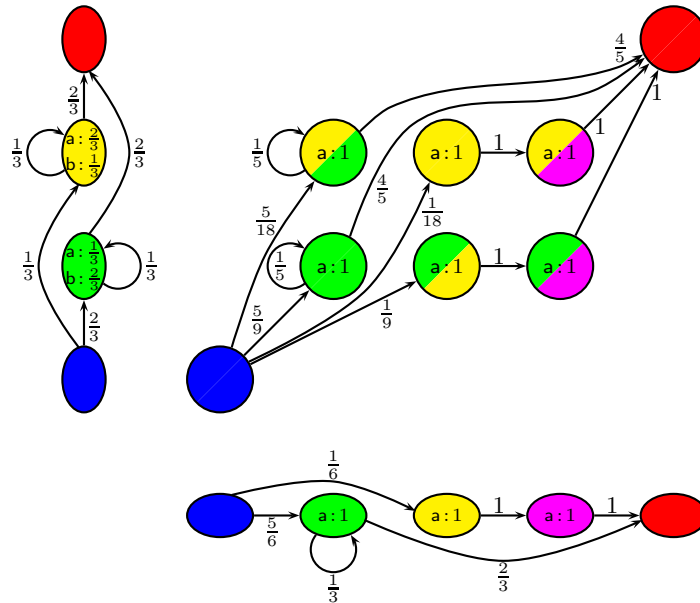


Figure 3: The HMMs from Figure 2 with the values of the combined model normalised to obtain proper probability distributions.

The standard approach to obtain probabilities for HMMs in bioinformatics is not based on theoretical considerations, but rather on estimating parameters from existing data – this is also known as training the HMM, and the data is usually referred to as training data. So the important question really is whether we still get unexpected behaviour of annotations when parameters of the combined model can be chosen freely.

The aim of this project is to investigate whether the first step of combining two models using the standard techniques from finite automata in itself introduces undesirable artifacts – whether the mere act of combined modelling makes capturing actual independence impossible. As is obvious from the many faded transitions and states in Figure 1, combining models does introduce some anomalies. Whether these can be offset by the freedom we have in choosing parameters is an open question. So in a

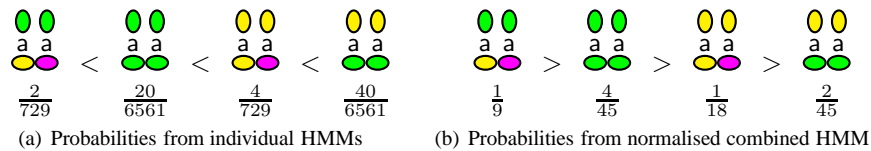


Figure 4: Probabilities of the four possible pairs of annotations of the sequence **aa** when multiplying annotation probabilities from the individual HMMs and when using the probabilities of the normalised combined HMM of Figure . Observe that the ordering of the annotations is completely reversed.

nutshell the project is to find a pair of HMMs for which there is no way to parametrise the combined model such that all sequences have the same ordering of their possible annotations in the two individual models and in the combined model, or to prove that it is always possible to parametrise the combined model with proper probability distributions in such a way that annotation orderings is preserved. An even stronger result would be to find a pair of HMMs for which there is no way to parametrise the combined model in such a way that all sequences have the same most probable annotation pair in the combined model as in the two individual models, as we are mostly interested in the most probable annotation.

Please note that this project comes with an author's warning. There are not really any simple stepping stones to proceed by – the success of the project essentially depends on being able to discover a crucial insight for settling the key question of the project.

References

- [1] J. Davies. Combining different grammars to make multiple annotations of a single sequence. 4th Year Dissertation, 2006.
- [2] R. Durbin, S. R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.