Unsupervised Discovery of Student Learning Tactics

Benjamin Shih, Kenneth R. Koedinger, and Richard Scheines
{shih, koedinger, scheines}@cmu.edu
Carnegie Mellon University

Abstract. Unsupervised learning algorithms can discover models of student behavior without any initial work by domain experts, but they also tend to produce complicated, uninterpretable models that may not predict student learning. We propose a simple, unsupervised clustering algorithm for hidden Markov models that can discover student learning tactics while incorporating student-level outcome data, constraining the results to interpretable models that also predict student learning. This approach is robust, domain-independent, and does not require domain experts. The models have tecst-set correlations with learning gain as high as 0.5 and the findings suggest possible improvements to the scaffolding used by many software tutors.

1 Introduction

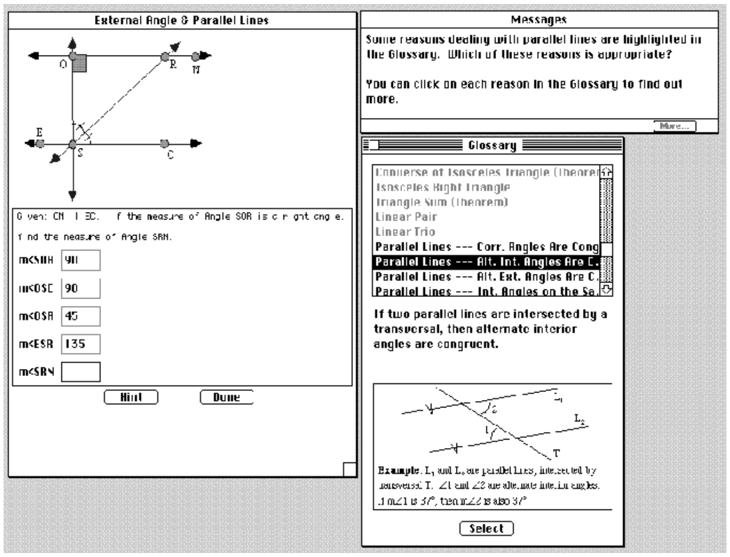
Since its inception as a field, educational data mining has consisted mostly of domain experts who use machine learning rather than machine learning experts who study education. The most commonly used methods are thus highly dependent on domain expertise. Examples include domain experts constructing data features [3], generating priors [5], and developing initial seed models [4]. An expertise-based approach is highly effective for educational data, but a reliance on domain experts has risks: if the domain expert's prior beliefs are wrong then the results will tend to be biased. The process can also be time-consuming and difficult for other researchers to replicate.

Alternatively, educational data mining without domain experts often results in uninterpretable or ungeneralizable models. Our solution is a novel unsupervised algorithm that incorporates student-level educational measures directly into the learning process, biasing the model search towards models that predict learning gain. Domain experts are only involved with the post-hoc interpretation of results. Further, in addition to predicting learning gain, the algorithm's models of student behavior suggest that the students who learn best tend to make persistent attempts rather than using software help.

2 Definitions

The data for this study comes from the Geometry Cognitive Tutor. A screenshot from a version of the tutor used in this study is shown in Figure 1. We will postpone most discussion of the data for later, but the tutor's representation of geometry problems is especially important. Each problem is shown on a separate page along with a geometry diagram. Students are expected to enter values, such as angle magnitudes, into answer cells. Solving

This work was supported in part by a Graduate Training Grant awarded to Carnegie Mellon University by the Department of Education (#R305B040063)


External Angle & Parallel Lines

Given: CN ‖ EC. If the measure of Angle SOR is a right angle.
Find the measure of Angle SRN.

m<SNH	90
m<OSC	90
m<OSR	45
m<ESR	135
m<SRN	

Messages

Some reasons dealing with parallel lines are highlighted in the Glossary. Which of these reasons is appropriate?

You can click on each reason in the Glossary to find out more.

Glossary

Converse of Isosceles Triangle (Theorem)
Isosceles Right Triangle
Triangle Sum (Theorem)
Linear Pair
Linear Trio
Parallel Lines --- Corr. Angles Are Cong
Parallel Lines --- Alt. Int. Angles Are C.
Parallel Lines --- Alt. Ext. Angles Are C.
Parallel Lines --- Int. Angles on the Sa.

If two parallel lines are intersected by a transversal, then alternate interior angles are congruent.

Example. L_1 and L_2 are parallel lines, intersected by transversal T. ∠1 and ∠2 are alternate interior angles. If m∠1 is 37°, then m∠2 is also 37°

Figure 1: Geometry Cognitive Tutor, circa 1998

Table 1: Mapping from ⟨Action,Duration⟩ to a single variable

	Attempt/Try	Hint
Fast	a	h
Slow	**A**	**H**

for and filling in one of these cells is called a "step". The first steps on a problem tends to involve givens; the later steps require values from the previous steps. Students can switch between steps at will, but can only switch problems by finishing them. Students can also request hints (fairly common) or read the glossary (very uncommon).

Within each step of each problem, a student performs actions (also called transactions in other literature). An action could be entering an answer (right or wrong), requesting a hint, or reading a definition from the glossary. For simplicity, we will group hints and the rare glossary request together, labeling them both as hints. The definition of an action has one additional wrinkle: each action has a corresponding duration. For example, a student might take 10 seconds to type an answer or 4 seconds to read a hint. Thus, actions are divided into two categories, long and short, using a threshold to be discussed later. Table 1 shows this mapping. For example, **A**aaaaa denotes one long attempt followed by many short attempts and might be considered a guessing tactic.

In sum, the data for this study consists of students working on geometry problems with each problem split into steps. The problems and steps are broken down into actions (a, **A**, h, or **H**). These rudimentary features are much simpler than human-constructed features. Beal et. al., for example, work from researcher-constructed features like "independent-inaccurate problem solving" [3]. Their approach, while successful, is dependent on the quality of the constructed features. Similarly, Baker et. al. use a set of human-constructed features as inputs to a feature construction algorithm [2]. Their approach can generate new

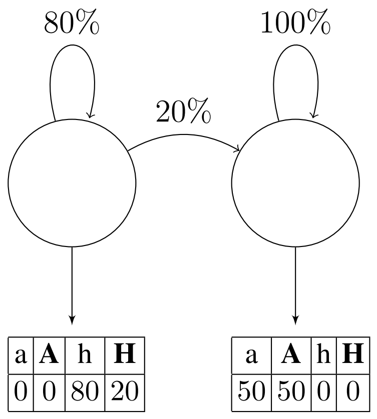


Figure 2: Example HMM

composite features, but the initial features must be expert-defined. Another distinction is that the actions used in this study are atomic: there is no lower granularity of data available. In contrast, Baker computes aggregate features over a roaming window of actions.

3 Hidden Markov Models

The goal of this study is to build models for student learning tactics. An example of a learning tactic, as defined in this paper, is, “The student requests hints quickly, over and over, until the tutor provides the solution. The student then enters the solution.” From this example, a learning tactic can be generalized to be an observable, predictable, and repeated *pattern* of behavior that is sufficiently abstract to include multiple *observed* instantiations. We implement learning tactics using hidden Markov models. A hidden Markov model (HMM) is a set of unobserved states, each state related to observations through a probability distribution. Here, the observations are student actions. Figure 2 shows an example HMM. Each unobserved state is represented by a circle; each arrow between states or looping back to a state represents a transition; the number above the arrow is a transition probability. The tables below the states show the probabilities of observing an action. When an HMM generates an action symbol, we say it *emits* the symbol.

Let a series of observed actions be a sequence. Sequences can be defined for either all actions in a problem or all actions in a step. Given a set of student sequences associated with an HMM, the Baum-Welch algorithm can relearn the parameters of that HMM to better fit the observed data. This is a standard method for learning a single HMM.

Single HMMs, as described above, have been used in many studies to model student behavioral traces. In a particularly relevant study, Beal et. al. used tutoring system log data to learn HMMs modeling patterns of student behavior [3]. Their study differs from this one in several ways: they define the structure of the HMMs by hand, they use outputs from another algorithm as inputs to their HMMs, they learn one HMM per student, and they perform clustering of students (not tactics) only after learning the HMMs. However, their key result is very relevant: HMMs work as both descriptive and predictive models for student learning behaviors and can find patterns without using cognitive models or domain content knowledge.

3.1 HMM Clustering

Let each individual HMM represent a single learning tactic. Discovering learning tactics requires discovering sets of HMMs. Let a set of HMMs be called a collection. In a collection, an observed sequence of actions is classified by whichever HMM is most likely to generate it. This results in a partitioning of the set of sequences, with each partition corresponding to one HMM. Each partition thus includes all observed examples of a given tactic.

The Baum-Welch algorithm can only learn parameters for a single HMM, but clustering algorithms can learn sets of HMMs, and thus sets of tactics. The usual objective of an HMM clustering algorithm is to maximize the total likelihood of generating the observed sequences. This type of problem has historically been tackled with Expectation-Maximization (E-M) algorithms and, for HMM clustering, given an initial set of HMMs, one iteration of the E-M algorithm is:

- Assign each sequence to the HMM most likely to generate it.
- For each HMM, relearn its parameters with Baum-Welch using the sequences in its partition.

This process begins with initial seed HMMs and repeats until a termination criterion is met, such as when an iteration results in fewer than 10 sequences being reclassified. A collection learned by this algorithm fits the data well if the likelihood of generating the observed sequences is high. This algorithm, here forth called *HMM-Cluster*, is provably guaranteed to converge to a local maximum. Further, *HMM-Cluster* will never change the number of HMMs in the collection (k) or the number of states per HMM (n); only the parameters and partitions will change.

There have been many prior uses of similar E-M HMM clustering algorithms, beginning with Rabiner et. al. for word recognition [6]. While there are newer variants, most HMM clustering is still done with Rabiner's original algorithm. A particularly illustrative study was done by Schliep et. al. to analyze gene expression data[9]. The paper discusses, amongst other things, the expressiveness of the models, the interpretation of results (for genetics), the inclusion of human labels, and the comparison of HMM clusters to other time series models.

3.2 Stepwise-HMM-Cluster

Unfortunately, naive *HMM-Cluster* has issues from both machine learning and educational perspectives:

- Like most E-M algorithms, *HMM-Cluster* gets trapped in local maxima.
- The choice of k and n determines the effectiveness of *HMM-Cluster*. If they are too

X:

	HMM 1	HMM 2
Student 1	80%	20%
Student 2	30%	70%
Student 3	25%	75%

Y:

Gain
73%
9%
5%

Figure 3: Example Stepwise Regression Inputs

large, the collection will overfit; if they are too small, no collection will fit the data.
- Collections that fit the data may not actually predict learning.

In principle, a better algorithm would search over values of k and n with a bias towards fewer, smaller HMMs, leading to better generalization and easier interpretation of the final collection. One such algorithm is *Stepwise-HMM-Cluster*, which is to *HMM-Cluster* what stepwise regression is to normal regression. An iteration of *Stepwise-HMM-Cluster*, for k HMMs and n states per HMM, proceeds:

- Begin with a collection of HMMs C.
- If $|C| < k$, generate $(k - |C|)$ new HMMs with n states per HMM.
- Run *HMM-Cluster* on C.
- Pick the "good" HMMs from C and use them for the next iteration.

A critical step in *Stepwise-HMM-Cluster* is the selection of "good models" from a collection C. This step allows *Stepwise-HMM-Cluster* to incorporate external data and iteratively improve its fit across iterations of the algorithm. For this study, HMMs are selected using forward stepwise linear regression: the total number of sequences classified by each HMM for each student is used as the independent variable and the pre-test to post-test learning gain is used as the dependent variable. A toy example is shown in Figure 3.

Stepwise-HMM-Cluster serves two goals at once: it tries to build a collection of HMMs to fit the observed sequences of actions, but also requires that the collection predict student learning gain. The incorporation of external data, such as pre-post learning gain, has been traditionally difficult when applying machine learning algorithms to educational data. This selection step addresses that issue, allowing student-level measures to influence the learning of much lower-level HMMs. In this case, learning gain is used to constrain the search for problem- and step-level HMMs. In future work, other data sources could be added, such as grade-point averages, survey information, or expert labels.

For this study, the parameters are restricted to $2 \leq k \leq 8$ and $2 \leq n \leq 8$. The limits of 8 HMMs and 8 states per HMM were chosen to maximize interpretability, but both limits exceed the complexity of any optimal collections actually found.

4 Data

This study uses two data sets, 02 and 06. Both data sets in this study originate in previous experiments, so only the control groups for each study are used. Both data sets involve

Table 2: Correlations with Learning, Best Collections, Test Data

τ	**02 Problem**	**02 Step**	**06 Problem**	**06 Step**
6	0.57	0.50	0.3	0.53
8	0.71	0.54	0.50	0.39
10	0.54	0.60	0.31	0.39

geometry tutoring systems that use the same general interface. However, the 02 data is from the angles unit, while the 06 data is from the circles unit. The 06 tutor also has some interface differences, including a minimum time per hint request. In the 06 data, students do fewer actions per step, complicating direct comparisons between the two data sets. Also, the 06 post-test used counter-balanced hint conditions between problems, e.g., sometimes students could get a hint at the cost of partial credit. This makes the test scores noisier and harder to predict.

- *02* data - First published in 2002, includes 21 students and 57204 actions [1].
- *06* data - First published in 2006, includes 16 students and 7429 actions [7].

5 Results

Stepwise-HMM-Cluster has several parameters. First, there is a threshold value between long and short actions. Let that threshold be denoted by τ. Second, *Stepwise-HMM-Cluster* can learn either problem-level or step-level tactics. For the former, HMMs are trained on sequences that include an entire problem. In the latter case, each sequence only contains actions from one step. The software implementation was built on the GHMM package [8] and, for a given search using a fixed value of τ, approximately 100 candidate collections reach the model selection stage.

Collections are learned from the first 80% of sequences per student; the remaining 20% are saved as test data. The main measure of a "good" collection is that it provides an accurate prediction of learning when applied to test data. To apply a collection of HMMs to test data, the HMMs are first used to classify test-data sequences. The total number of sequences per HMM per student is entered into the regression as shown earlier, now using parameters learned from training data. Table 2 shows the best correlations, for both data sets, between predicted learning gain and actual learning gain, as computed on the test data. Each column contains the best results for a specific run of *Stepwise-HMM-Cluster* with the rows split by value of τ.

In practice, Table 2 can be interpreted as showing upper bounds for predictions on withheld test data. However, even in this simple table, it's already clear that the 06 data is harder. The conclusion from Table 2 is that there are collections with a good fit to test data, if we can find them. The caveat to Table 2 is that it shows best collections picked after applying to test data; our actual goal is to find good collections using only the training data. To do so naively, however, invites overfit. This suggests the use of a selection heuristic: pick the collection with the best adjusted R^2 score on training data.

Table 3: Correlations with Learning, Selected Collections, 02 Test Data

τ	**Problem-Level**	**Step-Level**	**# of HMMs**	**Max # of States / HMM**
6	0.49	0.44	5	3
8	0.42	0.50	5	4
10	0.47	0.52	4	4

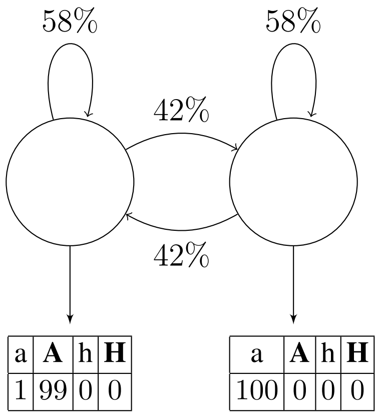


Figure 4: Dominant HMM for $\tau = 6$, 02 data

R^2, unadjusted, is defined as the sum-of-squared-error divided by the total sum of squares. For standard linear regression, R^2 is equal to the square of the correlation coefficient. The adjusted R^2 includes an additional term that grows in the number of model parameters, penalizing complex collections. Table 3 shows, for 02 data, test-set correlations for collections selected using adjusted R^2, the number of HMMs in the best collection (step-only), and the maximum number of states per HMM in the best collection (step-only).

The correlations in Table 3 are statistically significant ($\alpha < 0.05$) and almost as high as those in Table 2. They clearly show that, for the 02 data, it's possible to pick collections that generalize to with-held, within-student test data. However, the same table for the 06 data (not shown) is much less convincing. In 06 data, naively picking collections using the adjusted R^2 produces collections with poor predictions of learning. However, in 06, amongst the collections with the highest adjusted R^2, some collections do have a high test-set correlation with learning gain. For example, for $\tau = 6$, the fourth-best collection has a 0.46 test-set correlation with learning gain. The problem is selecting the right collection: while the adjusted R^2 is effective for 02 collections, it selects poorly from 06 candidates.

In the end, educational data mining requires interpretable results that have educational implications. Fortunately, *Stepwise-HMM-Cluster* outputs simple, interpretable HMMs. In particular, there is one HMM that occurs, with slightly different parameters, in every 02 collection shown in Table 3. This HMM-archetype always classifies a plurality of sequences, and so will be called the "dominant" HMM. Figure 4 shows a dominant HMM for $\tau = 6$, trained on 02 data at the step level. These dominant HMMs tend to be small, often only two states and never more than four.

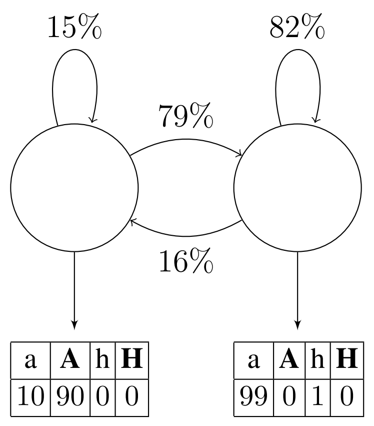


Figure 5: Repeated Guessing HMM for $\tau = 6$, 02 data

While interpreting the structure of individual HMMs is not actually meaningful (to be addressed), it is still a useful comprehension exercise. Here, the dominant HMM emits a and **A** with high probability, and emit both actions equally often (over the course of many sequences). One possible explanation is that the dominant HMMs select short sequences where the student already knows the answers and can solve each step in one attempt. However, the correlation between the frequency of first-try-correct sequences and learning gain is -0.24. Instead, an alternative interpretation for these HMMs is that they represent a persistence-trait. Students that attempt to solve repeatedly are more likely to learn the material than those that rely on hints. This is borne out by the resilience of the HMM to changes in τ, and by the duration-agnostic nature of the HMM, which emits both a and **A**.

However, this conflicts with common sense. The HMM shown in Figure 4 has a high probability[1] of emitting a sequence of type **A**aaaaa, i.e., a single long attempt followed by many short ones. This is generally considered poor learning behavior [2]. Intuitively, it represents a failed attempt followed by repeated, unthinking guessing. This paradox can be resolved by noting that no single HMM in any collection can be interpreted alone. Each HMM exists only as part of an entire collection and, thus, other HMMs in the collection can remove specific, degenerate sequences. Take the $\tau = 6$ collection as an example. It contains an HMM, shown in Figure 5, that has a high probability of emitting repeated-guessing type sequences. The repeated-guessing HMM, a highly specialized model, removes only the guessing sequences from the dominant HMM's partition. This relationship between HMMs in a collection, where a specific HMM can be tuned to special cases of a more general HMM, allows collections to be more expressive than the sum of their individual HMMs. However, this feature is also what makes the interpretation of the structure of individual HMMs meaningless, as a high probability sequence for one HMM may actually belong to another HMM's partition.

A more appropriate way of interpreting the HMM clusters is to directly examine the se-

[1]A high probability as compared to any other sequence of the same length. These HMMs are not assumed to model the distribution of sequence lengths, so comparing sequence probabilities is only meaningful for a fixed length.

quences classified by a particular HMM. For example, consider the dominant HMM for $\tau = 6, 02$ data, step-level. The five most commonly observed sequences in the HMM's partition are: **A**, **AA**, **A**a, **AAA**, **AA**a. None of these sequences are of the repeated-guessing type, yet they account for 95.5% of all sequences in the partition. Longer sequences in the partition follow the same pattern: example sequences include **AA**aa**AA** and **AA**a**A**a**A**. As noted above, guessing sequences, e.g., **A**aaaaa, are about as likely to be generated by the dominant HMM as the above sequences, but are actually captured by the repeated-guessing HMM. Similar results apply to the collections discovered for other values of τ.

The general interpretation of these results is that students learn more when using persistence-type tactics, as long as they don't guess repeatedly. Interestingly, this is largely independent of the choice of threshold τ. The most likely explanation is that very short or very long actions contains the most information about the student, and thus the actions that are reclassified by small changes in τ are relatively unimportant.

Finally, across all the best collections, hint-heavy tactics are negatively associated with learning. However, many of the more complex collections (3 or 4 HMMs) contain a "noise" HMM that generates all sequences with nearly uniform probability. Thus, hint-specific HMMs are actually very specialized, usually emitting mostly h actions. This explains the negative association with learning. Some "good" HMMs do involve hints, but those HMMs are not structurally consistent enough to permit conclusions without more data or analysis.

6 Conclusions

Most educational data mining methodologies either rely on domain experts or discover uninterpretable models. In contrast, *Stepwise-HMM-Cluster*, an unsupervised algorithm, can generate collections of HMMs that predict learning, but are also interpretable. For at least some data sets, *Stepwise-HMM-Cluster* produces collections of HMMs that can provide good predictions on with-held test data. This algorithm thus satisfies multiple educational data mining goals: it produces interpretable models, the models generalize (within-student), and the models not only fit data, but also predict learning outcomes.

Additionally, *Stepwise-HMM-Cluster* produced models with potential educational implications. Our results provide an additional argument that the most common type of hint-scaffolding in software tutors may be sub-optimal and that most learning may arise from persistent attempts to solve. This suggests a paradigm for tutoring systems that emphasizes attempts and provides hints or worked examples only when strictly necessary; however, there are other feasible explanations and more extensive exploration of this issue is required. In particular, there may exist learning tactics that are both productive and involve hints, but are difficult to detect due to noise or rarity.

There are many opportunities for future work. First, the evidence for statistical generalization is still weak; collections learned on one data set should be tested on another data set entirely. Second, adjusted R^2 appears to be a poor model selection criterion in some cases and other criteria may be more successful. Third, τ is a problematic parameter: while *Stepwise-HMM-Cluster* was robust to changes in τ in this study, it may not be robust in general. For example, some data sets may require three action strata (“Long”,“Medium”,“Short”) or require a different τ for hints versus attempts. Fourth, while there is already a procedure for interpreting clusters, there is significant room for improvement. A promising approach is to construct a visualization of the unfolding of sequences with sets of sequences with the same prefix or suffix grouped together. Finally, the algorithm itself is overly simple. The heart of *Stepwise-HMM-Cluster* is a strict assignment, E-M clustering algorithm using Baum-Welch; probabilistic mixture models or another method, such as spectral clustering, might improve the results, as might a better HMM learning algorithm.

Despite its limitations, *Stepwise-HMM-Cluster* has many potential applications. First, in its present form, the algorithm can already learn interesting models with little dependence on data. However, it is also flexible and extendible. Glossary requests could be separated from hints; new action types could be added for new data sets; tactics could be learned on the level of class sessions or curriculum units instead of problems and steps; human labels could be incorporated into the model selection step; actions could be redefined to include domain information, such as skill models. Further, there is the potential to develop a fully hierarchical algorithm that could simultaneously learn HMMs for individual step tactics, learn HMMs to classify problem tactics as a series of previously learned step tactics, and so on up, as far as sample size will permit.

However, the most important contribution of this study is neither the algorithm nor the educational implications. Rather, our results suggest an opportunity for a new paradigm where algorithms can simultaneously leverage multiple data sources at different granularities. In particular, constraining low-level models using student-level measures could potentially improve many existing algorithms and lead to important educational insights.

References

[1] ALEVEN, V., AND KOEDINGER, K. R. An effective meta-cognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. *Cognitive Science* (2002), 147179.

[2] BAKER, R. S., CORBETT, A. T., AND KOEDINGER, K. R. Detecting student misuse of intelligent tutoring systems. In *Proceedings of the 7th International Conference on International Tutoring Systems* (2004).

[3] BEAL, C. R., MITRA, S., AND COHEN, P. R. Modeling learning patterns of students with a tutoring system using hidden markov models. In *Proceedings of the 13th International Conference on Artificial Intelligence in Education* (2007).

[4] CEN, H., KOEDINGER, K. R., AND JUNKER, B. Learning factors analysis - a general method for cognitive model evaluation and improvement. In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (2006).

[5] CONATI, C., GERTNER, A. S., VANLEHN, K., AND DRUZDZEL, M. J. On-line student modeling for coached problem solving using bayesian networks. In *Proceedings of the Sixth International Conference on User Modeling* (1997).

[6] RABINER, L. R., LEE, C. H., JUANG, B. H., AND WILPON, J. G. Hmm clustering for connected word recognition. In *Proceedings of the IEEE ICASSP* (1989).

[7] ROLL, I., ALEVEN, V., MCLAREN, B. M., RYU, E., BAKER, R. S., AND KOEDINGER, K. R. The help tutor: Does metacognitive feedback improve students' help-seeking actions, skills and learning? In *Proceedings of the 8th International Conference on Intelligent Tutoring Systems* (2006).

[8] SCHLIEP, A., GEORGI, B., RUNGSARITYOTIN, W., COSTA, I. G., AND SCHNHUTH, A. The general hidden markov model library: Analyzing systems with unobservable states. In *Proceedings of the Heinz-Billing-Price* (2004).

[9] SCHLIEP, A., SCHNHUTH, A., AND STEINHOFF, C. Using hidden markov models to analyze gene expression time course data. *Bioinformatics 19* (2003), 255–263.