Using Hidden Markov Models to Determine Cognitive States of Visual Analytic Users

Mohamad Aboufoul* University of North Carolina at Charlotte
Ryan Wesslen† University of North Carolina at Charlotte
Isaac Cho‡ University of North Carolina at Charlotte
Wenwen Dou§ University of North Carolina at Charlotte
Samira Shaikh¶ University of North Carolina at Charlotte

ABSTRACT

Many visual analytics tools exist to assist users in examining large amounts of information at once via coordinated views that include graphs, network connections and maps. However, the cognitive processes that those users undergo while using such tools remain a mystery. Many psychological studies suggest that individuals may undergo some planning stage followed by analysis before finally making conclusions when examining large amounts of analytical data with the goal of reaching a decision. While the general order of these cognitive states has been theorized, the exact states of individuals at specific points during their interaction with visual analytic systems remain unclear. In this work, we developed models to determine the cognitive states of users based solely on their interactions with visual analytics systems via Hidden Markov Models. Hidden Markov Models allow for the classification of observations through hidden states (cognitive states in our case) as well as the prediction of future cognitive states. We generate these models through unsupervised learning and use established metrics such as AIC and BIC metrics to evaluate our models. Our solutions are designed to help improve visual analytics tools by providing a better understanding of cognitive thought processes of users during data intensive analysis tasks.

Index Terms: Hidden Markov Models; Visual Analytics; Cognitive States; Sense-making; Intelligence Analysis

1 INTRODUCTION

Visual analytics (VA) tools allow individuals to collect, analyze, and make sense of large heterogeneous data and accomplish a task or reach a conclusion. The task of engaging with VA is cognitively demanding, requiring the user to undertake various cognitive processes at different points of their interaction. Researchers, however, can only access observable data such as user clicks, not users' actual cognitive processes. These cognitive processes involve some form of information retrieval, schema representation, developing insight and making conclusions, as theorized in prior research [19]. By understanding what a user is currently thinking or planning, as well as predicting future behavior through computational models, assistive technologies can be developed to provide users with information that would improve their decision-making.

In this paper, our goal is to use a statistical Markov model to identify cognitive states and understand their effects on users' decisions when interacting with VA systems. Specifically, our contribution is to link sequences of cognitive states and the transitions between them with a Hidden Markov Model (HMM) using interaction log data (mouse clicks). We use HMMs to interpret latent mental states that we cannot concretely observe – in this context, cognitive states – by using observable behavior or actions [7]. These observable actions can be used to make predictions of the most likely future hidden states. The probabilities associated with each hidden state are determined in the HMM by using the Baum-Welch algorithm, in which the values are initially estimated [11]. The values are then adjusted over iterations – by using a training set of observable sequences – until they converge. Once the model is complete, it can then be used to predict the hidden cognitive state that corresponds to the observable states in other sequences as well as predict future hidden states for those sequences.

To train and test our models, we used training data from three in-laboratory VA studies [5, 12, 23] that employ different, coordinated-multiple views VA systems designed for two complex decision-masking tasks: event detection (CrystalBall [6]) and misinformation identification (Verifi [12]). During all three studies, interaction logs recorded mouse actions and the time spent per action as participants were asked to examine large amounts of data in multiple coordinated views and accomplish a specific assigned task. A major difference between the studies was whether the decisions of users at critical stages of their interaction were recorded within the system, as this affected whether supervised or unsupervised training would be possible. For one study that used CrystalBall, we employed unsupervised learning to develop HMMs via the Baum-Welch algorithm as user decisions were made outside of the system (i.e., paper and pencil) and therefore do not have labeled data for each user's decisions. On the other hand, in the two latter studies [12, 23] that used the Verifi VA system, users were explicitly asked to submit their decisions within the system. We can thus use such actions as labels for supervised learning of HMM models. Accordingly, for these two studies, we were able to measure accuracy through decision-making actions as a proxy for a deciding state.

The structure of this paper is as follows. First, we review related work, specifically in HMMs. Next, we discuss the data used to develop and test our models, the process of developing and testing those models, and assumptions we made to implement our model. We then provide our results including the use of Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) to determine what models most accurately mapped the sense-making process (i.e., how many hidden states would be in an optimal model). The AIC and BIC criteria measure goodness of fit by producing values for all models and comparatively examining them based on maximized likelihood. The model with the lowest AIC or BIC value is considered the most optimal. We conclude the paper with a discussion on our major findings and future directions of Hidden Markov modeling in VA systems.

2 RELATED WORK

Hidden Markov Models have been applied in a wide range of contexts including stock market forecasting [10], understanding human intentions via mobile robots [13], and detection of emergencies in

*e-mail: maboufou@uncc.edu
†e-mail: rwesslen@uncc.edu
‡e-mail: icho1@uncc.edu
§e-mail: wdou1@uncc.edu
¶e-mail: sshaikh2@uncc.edu

crowds [4]. Studies have utilized HMMs to better understand human behavior via clickstreams, which are aggregated sequences of users click actions on web interfaces. These studies include user profiling on commercial sites and clickstream sequence clustering, all according to behavior [17,21,24] as well as predicting user sequences based on current clickstream data [14]. Aside from these applications of HMMs, other research has explored how to determine the optimal number of hidden states during model training [22].

However, in the context of sense-making, cognitive states of mind, and behavioral studies, very little research has considered Hidden Markov Models. Prior research does include the use of Hidden Markov Models on data gathered from visual analytics tools [16], though the researchers did not have the goal of determining the sequences of cognitive states of their users in accordance with their observable behavior (i.e., actions) but rather to distinguish sequential patterns of users in different experiment conditions.

Perhaps the most relevant work to our study is Dungs [7], who developed Hidden Markov Models to better understand the search process of users of a digital library search system, representing human cognition based on user actions, and predicting the future actions of users using current actions. Their goal was to provide suggestions to assist users in their decision-making. Although the data collected by the digital library search system includes events that the system can directly observe, such as the execution of a search query, the training data for the HMMs are user eye fixations on one of four panels of interest on the search system collected by an eye tracker as observational data. All fixations on the same area of interest are treated equally as observable data, regardless of differences in time spent on each. Additionally, their research utilizes the HMMs developed to examine correlations between certain hidden states and observable states based on the emission probabilities in each model. By contast, our research attempts to garner a better understanding of the sense-making process [19] represented by the hidden states (cognitive states) in accordance with their corresponding observable states (user mouse actions). We make further assumptions as to which hidden states from each model correspond to which potential stages of the sense-making process [19] using the start, transition, and emission probabilities.

In a later study, Dungs et al. [8] used a hybrid form of HMM utilizing both discrete and continuous observable data to better improve support systems and assistive features of search systems and interactive retrieval systems. This study focuses on utilizing HMMs to distinguish between the different phases of searching, which Dungs et al. [8] treat as a two phase model involving searching and finding data. However, this study still does not attempt to map the sense-making process for decision making as described by Pirolli and Card [19]. We use HMMs to accomplish this and discuss our process below.

3 METHOD

3.1 Training Data

The data used for training our initial set of HMMs comes from the user logs recorded in an anchoring bias study [5]. This study used CrystalBall [6], a social media event detection VA system, in which 81 users were randomly divided into four groups and exposed to one of two different treatments: a numerical anchor (i.e., the task is framed to participants around a high or low number) and a visual anchor (i.e., a training video that reviews straegies using either the geospatial or temporal Views). Thus, the four experiment conditions were High Number-Geospatial anchor, High Number-Temporal anchor, Low Number-Geospatial anchor, and Low Number-Temporal anchor.

In CrystalBall, participants could perform five mouse actions: Click, Navigate, Hover, Scroll and Login. The observable mouse actions were recorded for each participant and included the time and interface panel where the action took place. Using psychological research on human cognition [15, 18], our categories depict different cognitive processes operating at different intervals of time. Table 1 displays the categorized actions with respect to their cognitive significance according to the time spent on them. For our purposes, we modify cognitive behavior to include "unit tasks" which take ˜10000+ ms, and we add a non-significant category to represent actions that take 100 ms or less. Because the CrystalBall interface has a coordinated-multiple views interface (i.e., five panels as well as a menu, a menu datepicker, and an introduction pop-up), only the actions+cognitive significance were treated as observable data as opposed to action-panel combinations. This resulted in 120 possible combinations, making the observable data far too sparse. Furthermore, we reduced our fifteen action+cognitive significance observations (which were the combinations of the cognitive significance categories from Table 1 and the five possible mouse actions from CrystalBall) by excluding combinations that comprised less than 0.1% of the total user log data. This resulted in ten action+cognitive significance observations being used, as seen in Table 2. User sequences varied in length from 241 to 2,016 action+cognitive significance observations.

Table 1: Cognitive Significance Categories According to Time Spent

Nonsignificant	0 - 100 ms
Deliberate	101 - 1000 ms
Cognitive	1001+ ms

Table 2: Action + Cognitive Significance Observation Combinations

Cognitive Significance	Action
Cognitive	Click
	Navigate
	Hover
Deliberate	Click
	Navigate
	Hover
Nonsignificant	Click
	Navigate
	Hover
	Scroll

3.2 Baum-Welch Algorithm

In training an HMM, one must provide an observation sequence O and a set of possible hidden states Q for it to learn a **transition probability matrix** A, with each value a_{ij} in the matrix representing the probability of transitioning from hidden state i to hidden state j, and an **emission probability matrix** B, with each value $b_i(o_t)$ representing the probability an observation o_t is generated from hidden state i [11].

The Baum-Welch algorithm (also known as the forward-backward algorithm) is a special version of the Expectation Maximization (EM) algorithm used to learn matrices A and B through an iterative approach of calculating initial estimates for all of the values, using those estimates to produce better estimates, and repeating the process until they converge to an optimal set of values.

The value of each a_{ij} can be represented by the equation below in which $C(i \rightarrow j)$ represents the total number of times a transition from state i to state j took place and $C(i \rightarrow q)$ represents the total number of transitions from state i to any state q have taken place.

$$a_{ij} = \frac{C(i \rightarrow j)}{\sum_{q \in Q} C(i \rightarrow q)}$$

In an HMM, we do not know the total number of these transitions for certain as the states are hidden from us. Thus, we must use the iterative approach of expectation-maximization here. We begin with

the forward algorithm, which tells us the probability of being in state j after t observations, given the automaton λ.

$$\alpha_t(j) = P(o_1, o_2 ... o_t, q_t = j|\lambda)$$

This value, represented by $\alpha_t(j)$, is calculated using the following equation, which must be calculated recursively given that it relies on the α of the state preceding state q_t, which relies on the state preceding that, and so on.

$$\alpha_t(j) = \sum_{i=1}^{N} \alpha_{t-1}(i) a_{ij} b_j(o_t)$$

The second component of the Baum-Welch (forward-backward) algorithm is the backward algorithm. As its name suggests, it is the opposite of the forward algorithm in that it tells us the probability that a sequence of observations from time $t+1$ to time T (the end of the time of the sequence) will occur given the current state q_t and the automaton λ.

$$\beta_t(i) = P(o_{t+1}, o_{t+2} ... o_T|q_t = i, \lambda)$$

This value, represented by $\beta_t(i)$ is calculated using the following equation, which must also be calculated recursively given that it relies on the β of the state succeeding state q_t, which relies on the state succeeding that, and so on.

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

Going back to the equation for estimating the transition probability a_{ij} for transitions from state i to state j, we still need to determine how to calculate the numerator, the number of transitions from state i to state j. To find this number, we can estimate the probability that the transition from $i \rightarrow j$ took place at a certain time t. We can then sum these values over all times t to get this total count. This is represented in the equation below where ξ_t is the probability of being in state i at time t and state j at time $t+1$, given observation sequence O and automaton λ.

$$\xi_t(i, j) = P(q_t = i, q_{t+1} = j|O, \lambda)$$

ξ_t, however, must be calculated using a similar value *not-quite-*ξ_t, which differs in that the probability of the observation sequence O occurring is measured as opposed to O being a given event in the conditional. This new conditional probability can be found by multiplying the forward probability, the estimated transition probability, the estimated observation emission probability, and the backward probability.

$$\textit{not-quite-}\xi_t = P(q_t = i, q_{t+1} = j, O|\lambda)$$

$$\textit{not-quite-}\xi_t = \alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)$$

Using the laws of probability, ξ_t is found by dividing *not-quite-*ξ_t by $P(O|\lambda)$, which represents the probability of the observation sequence O occurring given the automaton λ.

$$P(O|\lambda) = \alpha_T(q_F) = \beta_T(q_0) = \sum_{j=1}^{N} \alpha_t(j) \beta_t(j)$$

$$\xi_t(i, j) = \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{\alpha_T(q_F)}$$

The numerator in our equation for a_{ij}, the number of transitions from $i \rightarrow j$, can be found by summing ξ over all t. The denominator, the number of transitions from state i to any state k, can be found the same way by simply replacing j with k as demonstrated below.

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T\text{-}1} \xi_t(i, j)}{\sum_{t=1}^{T\text{-}1} \sum_{k=1}^{N} \xi_t(i, k)}$$

To estimate the emission probability $b_j(v_k)$ we can simply estimate the number of times an observation v_k will be emitted when in state j and divide that by the total expected number of times of being in state j. We need to determine the probability of being in state j at time t, which is represented by $\gamma_t(j)$

$$\gamma_t(j) = \frac{\alpha_t(j) \beta_t(j)}{P(O|\lambda)}$$

The emission probability is found by summing $\gamma_t(j)$ over all time t where the state was j and the observation o_t was v_k divided by the summation of $\gamma_t(j)$ over all time t.

$$\hat{b}_j(v_k) = \frac{\sum_{t=1 s.t. O_t = v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}$$

With this means of estimating a_{ij} and $b_j(v_k)$, the expectation-maximization iterative process of the Baum-Welch (forward-backward) algorithm can be conducted until the transition and emission probability values converge [11].

3.3 Model Generation

For implementation of the Baum-Welch algorithm, we use the JAHMM library [2, 9]. The action+cognitive significance observation sequences for each user were fed into the KMeansLearner constructor as vectors. This class uses k-means clustering to identify groups of similar observations and develops an HMM with initial transition and observation emission probabilities based off a defined number of hidden states, a uniform distribution of the probabilities for the defined number of observable states, and the observation sequence that is fed into the constructor [9]. Additionally, it generates a start probability for each hidden state, which gives the likelihood that a sequence will begin with that state.

The initial HMM is then used as a starting point for the Baum-Welch learning algorithm. The number of iterations for the learner is defined by the user. For our purposes, we used ten iterations for all HMMs generated. Increasing the number of iterations resulted in negligible differences (less than 0.001%) on the transition and observation emission probabilities, indicating that the HMM values have converged. The number of observable states is set to ten given the action+cognitive significance combinations from our data as described in Section 3.1.

Using the parameters above, we developed HMMs with hidden states ranging from 3 to 6. We also developed other HMMs with 3-6 hidden states based on the four experiment conditions: the High Number-Geospatial anchor, the High Number-Temporal anchor, the Low Number-Geospatial anchor, and the Low Number-Temporal anchor, as described in Section 3.1. In addition, we developed HMMs by combining the anchor conditions to test whether participants assigned to different visual anchor treatments follow similar cognitive processes. For example, in one combination, we combined the High-Geospatial and High-Temporal groups as a high numerical anchor group. In total, we developed 36 HMMs.

Because the hidden states were undefined in the generation of these HMMs, we sought to infer the cognitive states corresponding to each hidden state generated. We referred to the sense-making process described by Pirolli et al. [19]. From this we show empirically that decision-making begins with some form of planning/data collecting state, followed by one or more intermediary analyzing states, ending with a concluding/deciding state. As described by

Pirolli et al. [19], such transitions are typically not concrete; for example, one may begin by planning, move on to analyzing, and may return to planning/data collecting once more after a certain point of analysis or realizing that more data must be collected. An example representation of this sense-making process (which is hidden to us) can be seen in Figure 1, where a possible corresponding sequence of behaviors and interactions (which can be observed) are displayed in parallel. We assumed that the state with the greatest probability of starting a sequence was the planning/data collecting state. We also assumed that the state with the lowest probability of starting a sequence was the concluding/deciding state, which was typically 0.00. Figures 2, 3, and 4 represent the 3-state HMM generated using all 81 user sequences as training data. We have labeled the hidden states based on the assumptions described above. These diagrams are of the same model; they are only separated to make the model easier to present and interpret.

We noticed that based on our assumptions, planning/data collecting states were typically associated with cognitive actions, given the high emission probabilities for those observable states and zero/near-zero emission probabilities for other observable states. Similarly, analyzing states corresponded to "deliberate" actions while concluding/deciding states corresponded with "nonsignificant" actions. This strongly points to the notion that longer thought processes go into the earlier stages than the later stages of intelligence analysis [19].

3.4 Measuring Accuracy

CrystalBall user log data are unlabeled with regards to the cognitive states corresponding to each action+cognitive significance observation. Therefore, we used data from another set of studies using a VA system called Verifi. The user log data from Verifi was used to test the accuracy of our models. In the Verifi studies [12, 23], participants were tasked to submit a form when they were ready to make their decisions. Each user was tasked with making eight decisions regarding the veracity of eight Twitter accounts; however, each user had the option to resubmit decisions. We therefore assumed that at the time when a user submitted a form, they were in a concluding/deciding state. HMM accuracy is measured by counting the number of accurate classifications of the hidden state corresponding to the "click" action associated with submitting a form (also known as a "submit" action) and dividing it by the total form submissions from the Verifi interaction logs. Moreover, testing our models with new data from a different visual analytics system provides a more stringent, out-of-sample evaluation of model performance.

3.5 Verifi Evaluation

The Verifi logs contained user actions at a finer level of granularity than the CrystalBall data (scroll down or scroll up instead of simply scroll). As such, we mapped them to their CrystalBall equivalents.

After mapping, the mouse actions were further grouped according to their cognitive significance as described in Section 3.1. Because our HMMs do not have "Cognitive Scroll" nor "Deliberate Scroll" in the observation vocabulary, all such instances were removed from the Verifi user observation sequences.

We evaluated our model on two sets of user log data for the Verifi interface. The first set consisted of logs including 60 users in the Verifi study involving confirmation bias, which we will refer to as Verifi 1.0. The second data set included 94 users of the same Verifi interface, but for the study of the anchoring bias, which we will refer to as Verifi 1.1 [23]. Verifi 1.0 user sequences varied in length from 631 to 2,829 action+cognitive significance observations, whereas Verifi 1.1 user sequences ranged from 569 to 3,531 action+cognitive significance observations. All of the HMMs generated (as described in Section 3.3) were tested on both Verifi user log data sets using the methodology described in Section 3.4.

4 Results

4.1 Viterbi Algorithm and Preliminary Results

For generation, we implemented the Viterbi algorithm for the cognitive states in which the action+cognitive significance observations from the user sequences in CrystalBall were used. The Verifi 1.0 and 1.1 data sets were used as the test data sets. The Viterbi algorithm is a type of dynamic programming that uses a given HMM along with its start, transition, and observation emission probabilities to determine the most probable sequence of hidden states corresponding to a given observation sequence [11].

Once the predicted hidden state sequences were generated, we measured the accuracy of each HMM. However, all HMMs had a 0% accuracy, making the preliminary results inconclusive with regards to which number of hidden states best represented the intelligence analysis and sense-making process. The inconclusive results are likely due to the "submit" action taking anywhere from 20000 ms to over 150000 ms. The "submit" action includes all of the events that took place from the time a user opened a form to the time they submitted it (individual actions within the form were not recorded). They are thus treated as "cognitive" actions, and because the HMMs classified "cognitive" actions as planning/data collecting or early analyzing cognitive states, these observations were likely misclassified for the same reason.

4.2 AIC and BIC

Given our inconclusive preliminary results, we turned to statistical estimators of models to measure the relative strengths of the HMM models of different hidden states conditioned on the model's complexity. We used the Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) for our measurements of model fit. AIC and BIC values depend on the "maximized likelihood" of a model (as derived from the likelihood function) and the number of parameters (or complexity) in the model [20]. The BIC value further depends on the length of the observation sequence in question. The equations for both can be seen below where l represents the log of the "maximized likelihood," K represents the number of parameters, and n represents the length of the observation sequence.

$$AIC = -2l + 2K$$

$$BIC = -2l + K\log(n)$$

When comparing models of varying parameters, the model with the lowest AIC (or lowest BIC) is said to be the model with the best fit. As can be seen in the equations above, a large "maximized likelihood" will lower the AIC and BIC value. A greater number of parameters will add to that value; this means that a model will have to have a larger "maximized likelihood" to justify the use of extra parameters in a model. Such measures help prevent overfitting when looking for the model with the greatest fit. The key difference between AIC and BIC is that BIC penalizes extra parameters (i.e., model complexity) more than AIC.

4.3 AIC and BIC Results

We use the HMMpa package in R [1] to find the AIC and BIC values for each user sequence from the CrystalBall, the Verifi 1.0, and the Verifi 1.1 data sets. Additionally, we find the AIC and BIC values on modified versions of the Verifi 1.0 and Verifi 1.1 data sets which include "Cognitive Scroll" and "Deliberate Scroll" in the observation vocabulary (as described in Section 4.1), and we treat all instances of the "submit" action as a separate observation, disregarding the cognitive significance associated with them (although they all have a significance of "cognitive" due to each taking no less than 20000 ms). This gives the latter two data sets 13 possible observable states as opposed to 10. The modified Verifi 1.0 data set has user sequences ranging from 683 to 3,164 action+cognitive significance

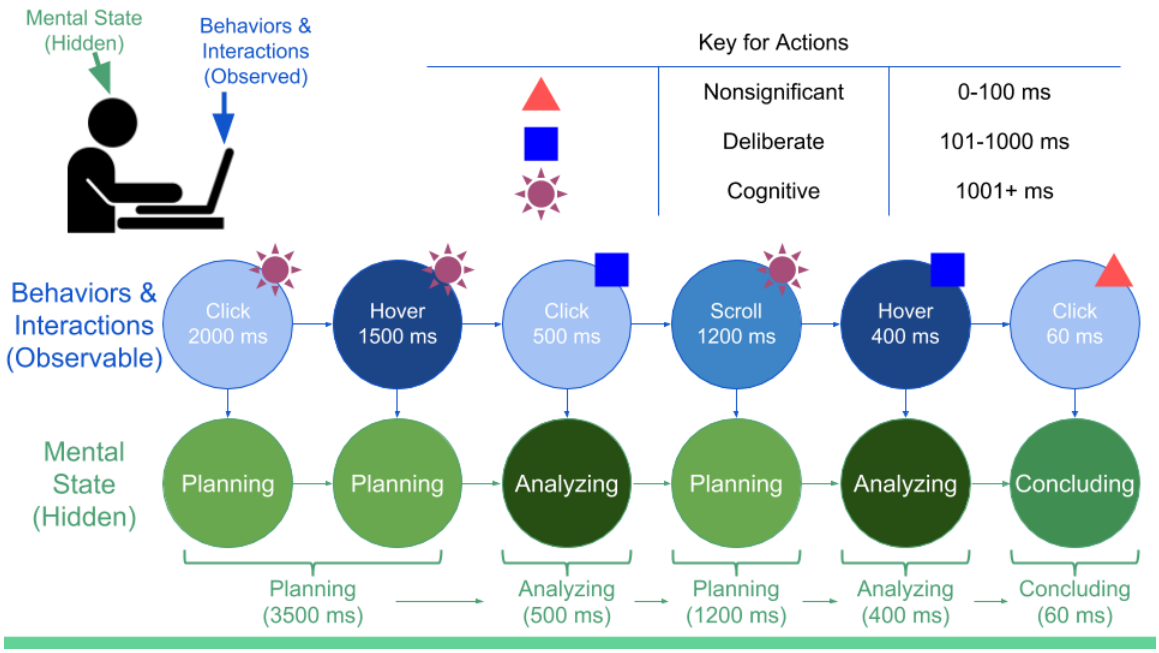

Figure 1: We use the behaviors and interactions of users as observable data, as described in Section 3.1, to predict the corresponding mental states, which are hidden to us. Any consecutive mental states that are the same can be treated as one large state.

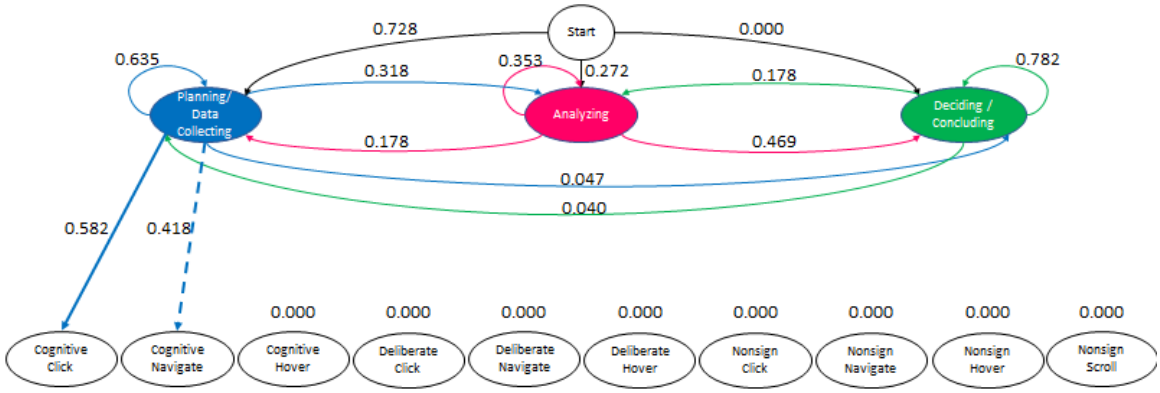

Figure 2: Planning/Data Collecting state emission probabilities

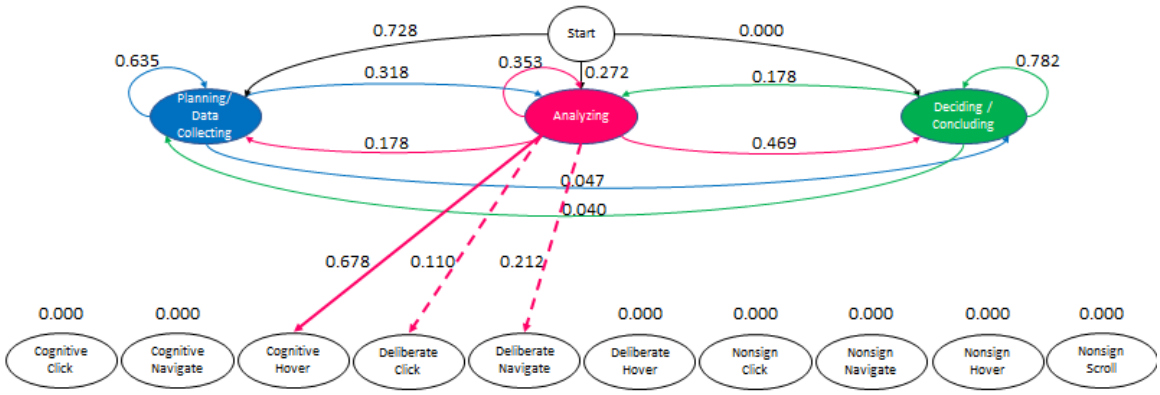

Figure 3: Analyzing state emission probabilities

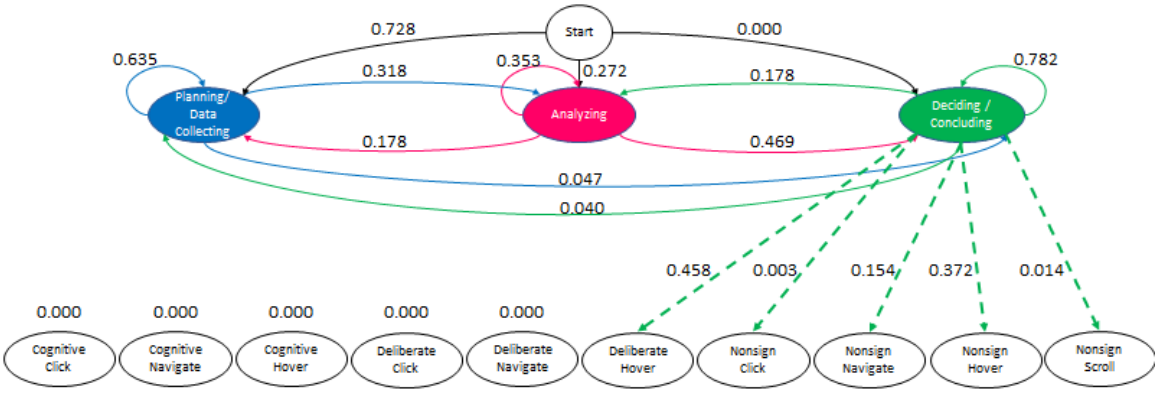

Figure 4: Deciding/Concluding state emission probabilities

observations, and the modified Verifi 1.1 data set ranges from 598 to 3,737 action+cognitive significance observations.

However, we cannot directly use the original HMMs in the HMMpa package, so we must instead use its HMM_training function which reads a single observation sequence and develops multiple HMMs, each with a predefined number of hidden states. It also calculates the corresponding AIC and BIC values for each model. A Poisson distribution is used for the probabilities of the observable states, and the Baum-Welch algorithm is used to find the optimal HMMs. All other parameters are left at their default values.

We ran this function on each user observation sequence for each of our five data sets. The models with the lowest AIC and BIC values for each sequence were recorded and the total counts were tallied up for each model. Additionally, the AIC and BIC values were averaged for each model in each data set, and the model with the lowest average was recorded as well. These results can be seen in Tables 3 and 4.

To examine whether or not AIC and BIC simply favored the models with the least parameters (the number of hidden states), we ran the HMM_training function again and had it generate a 2-state HMM along with the 3-6 state HMMs. Tables 5 and 6 provide the results.

We find that AIC largely favored the 3-state model while BIC found the 2-state model to be a better fit overall. Nevertheless, we select the 3-state model over the 2-state model for two reasons. First, BIC penalizes extra parameters more than AIC. Second, a 2-state model does not intuitively fit our application of intelligence analysis and the sense-making process. It is worth noting, however, that the 4-state model was the most favorable in the following instances:

- Verifi 1.1 with 13 observable states (Only in AIC when models with 3-6 hidden states are examined)
- Verifi 1.0 with 13 observable states (Only in AIC when models with 2-6 hidden states are examined)

While these instances are only favored with regards to AIC, they indicate that perhaps the best representation of intelligence involves somewhere between 3 and 4 cognitive states. Moreover, the different tasks to identify cognitive biases (anchoring, confirmation, etc.) may also influence this representation. Finally, the layout of a visual analytics interface could play another role in the sense-making process an individual undergoes when completing a task.

5 Conclusions and Future Work

In this paper, we find that a 3-state HMM provides the best fit among plausible models for intelligence analysis and the sense-making process. However, various assumptions were made in this study, which limited the strength and certainty of our conclusions overall. To further improve this work, certain aspects must be re-evaluated.

The data used are perhaps the most important detail that must be addressed. All of the user logs used in this study [5, 6, 12, 23] were not originally collected for the purposes of determining or predicting the cognitive states of mind of individuals. As such, the data were unlabeled for our purposes, leading to our assumptions regarding when a user was in a concluding or deciding state. Even then, the "form submit" action that we assumed corresponded with that state was not truly reflective of the submission of a form alone. The user logs did not record individual actions while completing the form and instead treated the entire filling of the form (and its submission) as one complete "form submit" action. Therefore, that action is thus recorded as taking far longer than it actually did. This affects that action's cognitive significance as it is an inaccurate observable state and led to inaccurate predictions from all of the original HMMs.

One solution could be to design future experiments that explicitly label decision-making steps. This could involve asking participants to state their cognitive state of mind during their analysis process. Of course, one possible complication would be that this could interfere with their decision-making process, leading to biased results (i.e., the Hawthorne Effect [3]). One way to mitigate such an effect would be for participants to perform tasks while being recorded and then retroactively be asked to state what their thought processes were as they watch the video recording of their actions. However, in both of these labeling procedures, one is assuming a set number of cognitive states that each user would have to choose from to properly, and consistently, label their actions in the interface. This could be solved by having different groups of participants, each given a separate list of possible cognitive states to choose from when labeling their data. Obviously there are other downsides such as participants not being aware of what cognitive state they were in as well as the length of time this entire process would take for each individual.

Another future aspect to examine is the type of user behavior recorded for observable states. In our study, we sought to use primitive data, thus focusing on mouse actions. However, eye tracking logs [7] were not included nor were keyboard actions recorded as neither were important for the studies' original purposes. If either approach were to be used, all possible actions would need to be defined properly and observation vocabularies would have to be set to ensure that they are not too long to avoid data sparsity problems. A reclassification method as described in Section 4.1 could also be utilized for such purposes.

Finally, other models could very well produce more improved results. This study only examined the effectiveness of Hidden Markov Models and the number of optimal states for such a model. However, Recurrent Neural Networks have offered promising results in more recent times and could very well produce improved models

Table 3: AIC and BIC Lowest Value Counts

	AIC				BIC			
	3-state	4-state	5-state	6-state	3-state	4-state	5-state	6-state
CrystalBall	63	17	1	0	81	0	0	0
Verifi 1.0 (10 observable states)	57	3	0	0	60	0	0	0
Verifi 1.1 (10 observable states)	60	34	0	0	91	3	0	0
Verifi 1.0 (13 observable states)	39	15	4	2	60	0	0	0
Verifi 1.1 (13 observable states)	48	29	14	3	92	2	0	0

Table 4: AIC and BIC Averages

	AIC				BIC			
	3-state	4-state	5-state	6-state	3-state	4-state	5-state	6-state
CrystalBall	3403.89	3412.13	3438.24	3475.64	3481.43	3553.53	3661.74	3799.49
Verifi 1.0 (10 observable states)	6370.17	6391.72	6422.35	6463.17	6458.27	6552.37	6676.29	6831.11
Verifi 1.1 (10 observable states)	7523.28	7527.15	7557.82	7597.76	7614.41	7693.32	7820.49	7978.35
Verifi 1.0 (13 observable states)	7672.69	7677.68	7700.55	7733.97	7762.94	7842.24	7960.67	8110.89
Verifi 1.1 (13 observable states)	8311.55	8308.48	8325.53	8358.56	8403.81	8476.71	8591.45	8743.87

Table 5: AIC and BIC Lowest Value Counts

	AIC					BIC				
	2-state	3-state	4-state	5-state	6-state	2-state	3-state	4-state	5-state	6-state
CrystalBall	8	51	19	3	0	46	35	0	0	0
Verifi 1.0 (10)	12	44	3	1	0	49	11	0	0	0
Verifi 1.1 (10)	16	75	2	0	1	79	15	0	0	0
Verifi 1.0 (13)	16	19	18	5	2	44	12	3	1	0
Verifi 1.1 (13)	22	34	22	14	2	76	17	1	0	0

Table 6: AIC and BIC Averages

	AIC					BIC				
	2-state	3-state	4-state	5-state	6-state	2-state	3-state	4-state	5-state	6-state
CrystalBall	3464.03	3407.19	3413.18	3439.09	3476.66	3495.96	3484.73	3554.58	3662.60	3800.51
Verifi 1.0 (10)	7404.52	6377.70	6395.91	6424.50	6463.57	6440.80	6465.80	6556.56	6678.43	6831.51
Verifi 1.1 (10)	7541.25	7514.41	7535.99	7567.93	7607.45	7578.77	7605.54	7702.17	7830.59	7988.04
Verifi 1.0 (13)	7732.62	7689.90	7680.67	7700.85	7732.69	7769.78	7780.15	7845.24	7960.98	8109.61
Verifi 1.1 (13)	8319.11	8291.61	8296.67	8314.92	8350.88	8357.10	8383.86	8464.90	8580.83	8736.18

of intelligence analysis and the sense-making process. With the proper labeled data, models can be better tested and model comparison can be done with much greater certainty. The generation of such accurate models could help researchers better understand cognitive thought processes and could offer significant applications in assistive technologies for various user interfaces including but not limited to visual analytics tools. Such assistive technologies could help users make better, more informed decisions by providing the right information at the right time in accordance with their thought processes.

REFERENCES

[1] Hmmpa. `https://rdrr.io/cran/HMMpa/man/HMM_training.html`. Last accessed 16 June 2018.

[2] Jahmm. `https://github.com/KommuSoft/jahmm`. Last accessed 16 June 2018.

[3] J. G. Adair. The hawthorne effect: a reconsideration of the methodological artifact. *Journal of applied psychology*, 69(2):334, 1984.

[4] E. L. Andrade, S. Blunsden, and R. B. Fisher. Hidden markov models for optical flow analysis in crowds. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, vol. 1, pp. 460–463. IEEE, 2006.

[5] I. Cho, R. Wesslen, A. Karduni, S. Santhanam, S. Shaikh, and W. Dou. The anchoring effect in decision-making with visual analytics. In *Visual Analytics Science and Technology (VAST), 2017 IEEE Conference on*, 2017.

[6] I. Cho, R. Wesslen, S. Volkova, W. Ribarsky, and W. Dou. Crystalball: A visual analytic system for future event discovery and analysis from social media data. In *IEEE Conference on Visual Analytics Science and Technology (VAST)*, 2017.

[7] S. Dungs. Describing user's search behaviour with hidden markov models. In *Bulletin of IEEE Technical Committee on Digital Libraries*, vol. Volume 12 Issue 2 November 2016 of *TPDL '16*. TCDL, 2016.

[8] S. Dungs and N. Fuhr. Advanced hidden markov models for recognizing search phases. In *Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval*, pp. 257–260. ACM, 2017.

[9] J.-M. Francois. *Jahmm v0.6.1 User Guide*, February 2009.

[10] M. R. Hassan and B. Nath. Stock market forecasting using hidden markov model: a new approach. In *Intelligent Systems Design and Applications, 2005. ISDA'05. Proceedings. 5th International Conference on*, pp. 192–196. IEEE, 2005.

[11] D. Jurafsky and J. H. Martin. *Speech and language processing*, vol. 3. Pearson London:, 2014.

[12] A. Karduni, R. Wesslen, S. Santhanam, I. Cho, S. Volkova, D. Arendt, S. Shaikh, and W. Dou. Can you verifi this? studying uncertainty and decision-making about misinformation using visual analytics. 2018.

[13] R. Kelley, A. Tavakkoli, C. King, M. Nicolescu, M. Nicolescu, and G. Bebis. Understanding human intentions via hidden markov models in autonomous mobile robots. In *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction*, pp. 367–374. ACM, 2008.

[14] S. Laxman, V. Tankasali, and R. W. White. Stream prediction using a generative model based on frequent episodes in event sequences. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 453–461. ACM, 2008.

[15] Z. Liu and J. Heer. The effects of interactive latency on exploratory visual analysis. *IEEE transactions on visualization and computer graphics*, 20(12):2122–2131, 2014.

[16] L. J. Mariano, J. C. Poore, D. M. Krum, J. L. Schwartz, W. D. Coskren, and E. M. Jones. Modeling strategic use of human computer interfaces with novel hidden markov models. *Frontiers in psychology*, 6:919, 2015.

[17] V. Melnykov. Model-based biclustering of clickstream data. *Computational Statistics & Data Analysis*, 93:31–45, 2016.

[18] A. Newell. *Unified theories of cognition*. Harvard University Press, 1994.

[19] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of international conference on intelligence analysis*, vol. 5, pp. 2–4, 2005.

[20] D. Posada and T. R. Buckley. Model selection and model averaging in phylogenetics: advantages of akaike information criterion and bayesian approaches over likelihood ratio tests. *Systematic biology*, 53(5):793–808, 2004.

[21] E. M. Schwartz, E. Bradlow, P. Fader, and Y. Zhang. children of the hmm: Modeling longitudinal customer behavior at hulu. com. 2011.

[22] S. M. Siddiqi, G. J. Gordon, and A. W. Moore. Fast state discovery for hmm model selection and learning. In *Artificial Intelligence and Statistics*, pp. 492–499, 2007.

[23] R. Wesslen, S. Santhanam, A. Karduni, I. Cho, S. Shaikh, and W. Dou. Anchored in a data storm: How anchoring bias can affect user strategy, confidence, and decisions in visual analytics. *arXiv preprint arXiv:1806.02720*, 2018.

[24] A. Ypma and T. Heskes. Automatic categorization of web pages and user clustering with mixtures of hidden markov models. In *International Workshop on Mining Web Data for Discovering Usage Patterns and Profiles*, pp. 35–49. Springer, 2002.