Evaluation of Hierarchical Temporal Memory for a Real World Application

Wim J.C. Melis, Shuhei Chizuwa and Michitaka Kameyama
Graduate School of Information Sciences, Tohoku University
6-6-05, Aoba, Aramaki, Aoba-ku
Sendai, 980-8579, JAPAN

Abstract

A large number of real world applications, such as user support systems, can still not be performed easily by conventional algorithms in comparison with the human brain. Such intelligence is often implemented, by using probability based systems. This paper focuses on comparing the implementation of a cellular phone intention estimation example on a Bayesian Network and Hierarchical Temporal Memory. It is found that Hierarchical Temporal Memory is a system that requires little effort for designing the application, and with some extra effort, further optimised results can easily be obtained.

1 Introduction

Real world applications have formed a continuous drive towards finding better algorithms. Some applications can however not be implemented as easily as the way the human brain would perform them. It is generally believed that the human brain operates on a probability bases, combined with the fact that probability systems are more general than deterministic systems, probability systems have recently known an increasing popularity.

Probability based systems have also proven to be useful for user support systems, presented in Figure 1, see for example [2]. In relation to this increasing popularity of probabilistic systems this paper focuses on taking a simple application and implementing it on two different probabilistic systems. Since probabilistic system are generally quite computationally demanding, one of the questions in this comparison is on how their requirements on hardware would trade-off with the benefits of using that particular algorithm. Since the ultimate goal of such an intelligent system is for it to help us humans in our daily tasks, and this with the least requirements on design time, power consumption, and also costs.

This paper will consider two different probability based systems, namely Bayesian Network (BN) [1, 5] and Hierarchical Temporal Memory (HTM) [4, 3]. Although HTM is to some extend based on BN, it has some features which clearly distinguish it from BN.

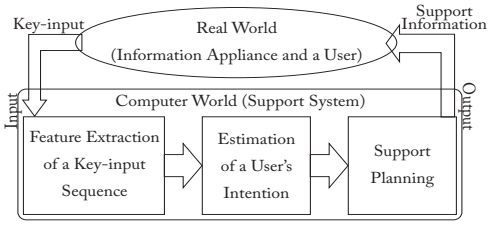

Figure 1: Components of a user support system.

This papers starts with describing HTM, which achieves intelligence by being trained with the data it should remember, similar to BN. During inference it then performs probabilistic calculations on finding the closest match between the stored system and the newly incoming pattern. Next is a short description of the phone intention estimation application, followed by a description of the algorithms on the two probability platforms: BN and HTM. The comparison section compares both algorithms in more detail, by looking at some results.

2 Hierarchical Temporal Memory

Hierarchical Temporal Memory is in essence a method of modelling the human brain. As the name implies it has a hierarchical structure and each node contains a large amount of memory, furthermore, time plays an important role in the system.

The hierarchical structure, as shown for example in Figure 3 b) & c), has its data or sensor input fed in at the bottom and category data is output at the top node. This category data would, in the case of HTM having been trained with pictures from cats and dogs, be an indication on whether a cat, a dog or neither has been recognised.

Within the HTM hierarchy, each node performs the same algorithm, but on different data. These nodes consist of two algorithmic parts, namely a spatial and a temporal pooler. The spatial pooler is responsible for storing common input patterns in space. For an image example, this could be adjacent pixels in a certain colour. Each one of these common

patterns is stored as a coincidence pattern in the node memory. When the spatial pooler has learned these coincidence patterns it will swap to normal operation and calculate how much the input matches to any of the stored patterns. This data forms input to the temporal pooler, which stores sequential information about these patterns stored in the spatial pooler. These patterns will be grouped together. During normal operation the temporal pooler determines which group the received coincidence belongs to.

The input to an HTM network is data at a very low level, and its standard hierarchy will work for any application. Though, altering the the HTM structure to fit the application might lead to better classification. In relation to a hardware implementation, these different hierarchical structures infer a considerable amount of flexibility for the communication part.

3 Cell Phone Intention Estimation

The purpose of this application is to predict user intention during the use of a cell phone. This is based on the sequence of menu choices that a user chooses while using its phone. The sample phone had several methods of reaching the same target action, which was the initiation of a mail or call action. The application could then be used to detect in an early stage what action the user might be intending to do and propose him this intention. Although the current application is designed for a particular phone and it's menu structure, the principle can be applied to any type phone and menu structure because these probability systems are taught, and therefore if the system is different, only the training data needs to be adapted. Ultimately, it would clearly be necessary for the system to adapt to the user, by performing extra training while the system is in use.

4 Different Algorithms

The performance of probability systems is highly dependent on the training data. Therefore, the training data was carefully composed to for example include an equal number of times each of the different paths to reach one of the intentions. For this specific application, three different paths were existing to reach these 2 actions. Two of these three paths directly target one of the actions, whereas the third one shares a common start, and the goal is only distinguishable during the last few key presses. The composed training data was used for both systems.

4.1 Bayesian Network

The design of a BN requires the design of a causal relationship network, which is shown in Figure 2 for this application. This causal relationship network was then implemented on the Bayonet software platform [6] that uses the K2 learning algorithm. Furthermore, a BN only performs the actual user intention estimation, requiring a separate algorithm for extraction of features from the key input sequence (see Figure 1). For hardware implementation of a BN based user support system, the BN algorithm could easily be implemented on a general BN hardware device, though the feature extraction device would be application specific.

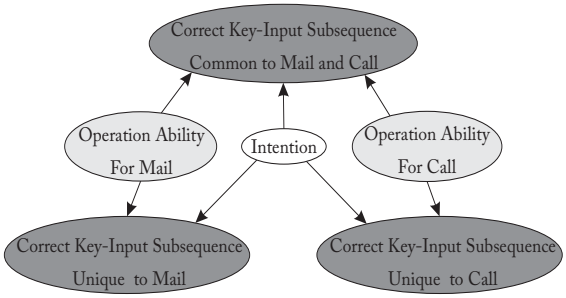

Figure 2: Causal relation between intention, operation ability and key-input sub-sequences.

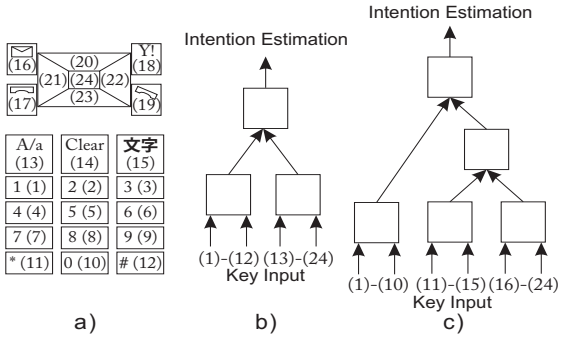

Figure 3: a) Layout of Phone Keyboard; Hierarchical structure of HTM implementations: standard (b) and function (c)

4.2 HTM

In contrast to the Bayesian Network implementation, the input to HTM is at a much lower level and comes directly from the real world, for this application being the key press actions. The numbering for the keys of the phone used for this application is shown in Figure 3 a).

In order to measure the influence of the structure of the HTM network, two different structures are studied. The first one, called standard, and shown in Figure 3 b), uses a normally balanced hierarhical tree structure. In this case the total number of key inputs are simply split among the two nodes. Each node will store the spatial/temporal information in relation to its inputs. Any information that would cross the boundary, due to the splitting of the key-inputs, is expected to be covered by the top level node.

The second architecture, shown in Figure 3 c), exploits the function of the phone's key's, hence named function. Namely, keys (1)-(10) relate to number or letter input, whereas key ranges (11)-(15) and (16)-(24) both have functional purpose. Therefore they are likely to have certain spa-

tial/temporal relations within their range as well as among themselves as a set of functional keys. In relation to the initiation of a mail or call action, keys (16)-(24) are directly linked with the selection of a particular function and therefore of high importance to the intention estimation. In this architecture the functional use of the keys is exploited by combining the key inputs that have a similar type function on the same node. The two nodes with function inputs are then combined at the next level to cover relations between these two groups of function keys, before being combined on the top level with the number key inputs.

For each of the above architectures, two different temporal pooler algorithms were tested. The difference in algorithms is in the fact that only the current input, or also past information is taken into account. Since this application has a lot of information in the sequence of key press actions, this test is of particular importance. The algorithm that takes past information into account has several parameters, which were also investigated.

5 Comparison

As mentioned before, BN as well as HTM require training, and both systems were trained with the same data. This data consisted of a list of actions taken and the resulting intention of the user. The results are expressed in probability numbers for BN as well as HTM. For BN these are the probability values in relation to one particular input being set high, e.g. "Correct Key input subsequence unique to mail", which provides the probability value for the estimation for that event. The results from HTM are generally expressed as accuracy values, but if the input data is the same as the training data, then the accuracy value measured is actually a probability value. Consequently, allowing comparison between BN and HTM.

In an ideal situation it would be expected to obtain a value of 100% for the estimation of mail or call when the correct unique input sequence is provided. In the case where the input sequence is shared for mail and call, then the optimal value is expected to be 50%. If the system is fed with a mixture of the three possible input sequences and assuming that each path is taken an equal number of times, this would lead to a theoretically expected optimal of 83%.

5.1 Bayesian Network

The results for the BN implementation are presented in Table 1. As can be seen from this table, the results are fairly close to the theoretical optimal values. For the case where the input sequence is shared for mail and call, two values are provided. The first is the one indicating the probability for mail (M), the second is the probability for selecting call (C). As mentioned before, BN is designed at a higher level than HTM. This is also one of the main reasons why BN performs closer to the theoretical optimal value, because BN does not include the feature extraction, which is actually

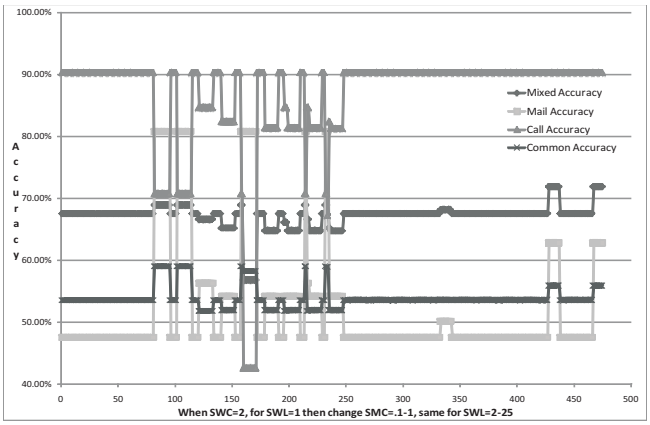

Figure 4: Indication of accuracy results for different SWL and SMC for the standard HTM with an SWC=2.

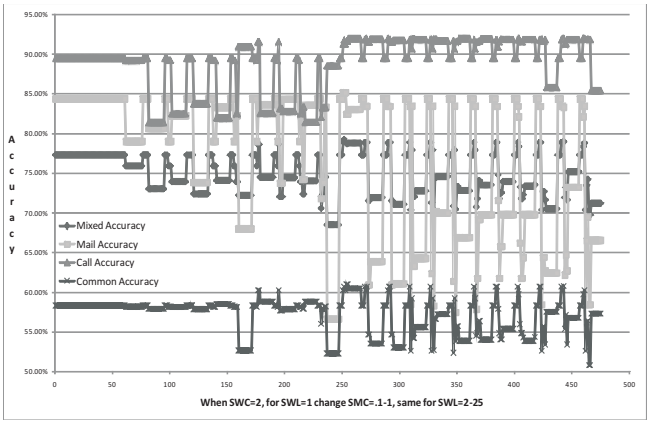

Figure 5: Indication of accuracy results for different SWL and SMC for the Functional HTM architecture with an SWC=2.

the more difficult part of this application.

5.2 HTM

When comparing the results of the different HTM structures that uses the standard temporal pooler algorithm, then little difference can be noticed. This is due to the fact that this application has mainly temporal information, which is not exploited in this system. Therefore, the fact that inputs are combined in such a way to exploit their temporal relationship has no effect on the system as such.

The second algorithm that exploits sequential information, has several parameters. A first parameter is the Sequence Window Length (SWL), for which the optimal value is expected to be related to the average or maximum length of the possible input sequences. Other parameters determine the number of stages to check for sequential information, Sequence Window Count (SWC), and a factor that influences the amount of sequential information stored during the learning phase, namely Sequence Model Complexity (SMC).

In case of the standard structure and for an $SWC = 1$,

Table 1: Results for Bayesian Network and Different types of HTM implementations

Algorithm/Architecture	Mail only Subsequence	Call only Subsequence	Subsequence Common to Mail & Call	Mixed Sequence
Bayesian Network	99.74%	99.31%	57.73%(M) / 42.27%(C)	—
Normal HTM, Stand. Alg.	80.79%	86.61%	59.43%	75.77%
Normal HTM, Seq. Alg. (SWC=1)	47.59%	90.33%	53.52%	67.52%
Funct. HTM, Stand. Alg.	80.79%	86.66%	59.45%	75.77%
Funct. HTM, Seq. Alg. (SWC=1)	84.39%	89.47%	58.31%	77.30%

then the accuracy values are as good as independent of the other two parameters (SWL & SMC). The results are however less good when compared to the standard algorithm, which gives a clear indication that these parameters are not optimal. For higher SWC values, approximately the same pattern can be noticed for different SWL and SMC values, therefore only $SWC = 2$ is displayed in Figure 4. This graph indicates that the accuracy changes for different SWL & SMC values. However, it is also clearly visible that if the mail accuracy improves, then the call accuracy reduces. This is a consequence of the algorithm learning to classify between only two groups. On average, the results are worse than for the standard algorithm, but for some specific parameter sets, the difference is rather small. The fact that mail accuracy is very bad implies that there is something within that mail sequence that is not well captured within the trained HTM, which is due to the keys used for this sequence being input to different lower level nodes. As expected the optimal sequence length is related to the sequence length of the mail and call sequences. The maximum for this length is 13, however on average most patterns fall within the range 9-11.

For the functional structure, the accuracy values for an $SWC = 1$ are again independent of SWL & SMC, but better than for the standard structure with the time algorithm, and the functional structure with the standard algorithm. These results clearly indicate that taking the structure of the application into account significantly improves the performance of the system. Similar to the standard architecture, this architecture shows similar irregularity of accuracy for different SWC when varying SWL & SMC. Therefore, again only $SWC = 2$ is displayed in Figure 5. When looking at the best sequence lengths for this case, it seems that the optimal sequence length is the maximum length of the input sequence for the application, namely 13. Other sequence lengths show less good accuracy, and that clearly also applies if the sequence length is double the average sequence length or longer. Neither of these latter are surprising results in relation to the features of the application.

6 Conclusions and Future Work

This paper described the implementation of a user support system implemented on a BN as well as the HTM platform. In comparing them at the design level, BN is designed from a much higher level and requires more interaction from the user. It also requires some pre-processing of the information coming from the real world. HTM on the other hand takes in the information from the real world, and its standard architecture and algorithm already provide a reasonably good accuracy. If the structure of the application is more reflected into HTM, then even better accuracy can easily be obtained.

HTM is a platform under development, but it should soon include another feature of the human brain, namely prediction. This would allow for this application to be extended with predicting the intention of the user and consequently speed up the device usage. A next step within this user support system would then be for the phone to also learn about the frequently used telephone numbers and words typed during mails, and predict these while typing.

Future work consists of implementing other applications in order to learn more about e.g. the creation of learning data. Furthermore, the used systems are software implementations, however these systems are quite computationally demanding, since on one hand they require high precision floating point multiplications and additions, besides having very high data-flow demands. The latter is due to the fact that for each provided input sequence, they need to calculate the matching probability with all stored patterns. Therefore, hardware implementations of these algorithms would significantly improve overall performance. From the results of this comparison it seems that HTM has the benefit of taking data in at a very low level. Furthermore each node is identical, which would allow for a cellular type hardware structure.

References

[1] E. Castillo, J. Gutierrez, and A. Hadi. *Expert Systems and Probabilistic Network Models*. 1997.

[2] S. Chizuwa and M. Kameyama. Bayesian network based intention estimation for a user support system of an information appliance. In *5th International Conference on Information Technology and Applications (ICITA)*, pages 71–76, 2008.

[3] D. George and B. Jaros. The HTM learning algorithms, 1/03/2007 2007. `http://www.numenta.com/for-developers/education/Numenta_HTM_Learning_Algos.pdf`.

[4] J. Hawkins and D. George. Hierarchical temporal memory: Concepts, theory, and terminology, 3/27/2007

2007. http://www.numenta.com/Numenta_HTM_Concepts.pdf.
[5] F. Jensen. *Bayesian Networks and Decision Graphs*. 2001.
[6] Y. Motomura. Bayesian network software bayo-net. *Journal of The Society of Instrument and Control Engineers*, 42(8):693–694, 2003.