Programming Language Support for Probabilistic Associative Memory

Dept. of CIS - Senior Design 2014-2015*

Fan Yin
fanyin@seas.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

Haolin Lu
haolinlu@seas.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

Yukuan Zhang
yukuan@seas.upenn.edu
Univ. of Pennsylvania
Philadelphia, PA

ABSTRACT

The goal of this project is to create a programming language library that provides an interface for programming with a probabilistic associative memory, as well as various implementations of these memories. An associative memory allows queries by content rather than by address or location. Probabilistic associative memory returns a value based on a probability model for a given memory query. This project provides an intuitive interface to this probabilistic associative memory to facilliate programming efficiency for the end user.

1. INTRODUCTION

The standard way to access a stored value in memory is by address. This involves finding the physical position of that address in memory and returning the stored value. Meanwhile, associative memory allows searching for values based on the stored content. An analogous structure in programming languages is the hashtable, which returns an output value based on an input key.

Standard associative memories are useful for when one needs to query memory based on some input data or tag. For instance, router firmware often employs associative memories to store tables of Internet Protocol (IP) addresses, which serve as the "next hop" for incoming packets. Since routers looking for the "next hop" have a specific, standard query format – e.g. an IP address of the form www.xxx.yyy.zzz – it makes sense to store the routing tables based around this input string.

A probabilistic associative memory differs from an associative memory in the sense that values stored in the latter are not guaranteed to be recalled with 100-percent fidelity.

For instance, suppose there existed some database of words where "kat" appeared once and the word "cat" appeared 100 times. If this collection were stored within a regular associative memory, and the memory is queried for a string matching "kat", the memory would return "kat" because it was stored in the memory. With a probabilistic memory, the same query may return "cat", simply because "cat" matches closely with "kat" and it occurs much more often. The word "cat" would thus be considered the more "likely" match. Thus, with probabilistic memories, the most "likely" value is returned for a given query based on some probability model of the values stored in memory.

In a simplistic representation of a probability model, a frequency table of memory values can be constructed as memory values are stored. The queries would then consist of a set of values, and a likelihood would be generated based on both frequency and some matching metric in that query-set. Thus, the memory mechanism can extrapolate or interpolate a response to a query based on values previously inserted into memory.

The implementation may also include support for structured data in the probabilistic associative memory. In a standard associative memory scheme, a similarity query is done by comparing input bits to the values stored in memory. However, this does not allow for accurate similarity comparisons between trees or lists, as the memory is not aware of the underlying structure. As a result, many applications on structured data will first flatten the structure, perform the requisite calculations, then reconstruct the original structure of that data. With a built-in type system that supports structured data, the user will be able to directly manipulate the data in memory while keeping the structure intact.

Practical applications of probabilistic associative memories involve various approximation and prediction methods. These involve appliactions such as image reconstruction from noisy input data, or search autocompletion. This project strives to provide a simple, flexible interface to work with these probabilistic associative memories.

2. RELATED WORK

With regards to existing related work, the goal of this project is not to advance associative memory techniques. Rather, it is seeking to research existing models for associative memory, provide a simple probabilistic query interface, and create implementations that fit this interface. With that in mind, there are many works that address different forms of associative memory.

3. SYSTEM MODEL

One of the seminal works concerning associative memories is Willshaw's *Non-holographic Associative Memory* [5]. This paper describes a model for a probabilistic associative memory that holds relations between elements of a query space and elements of an output space. A query to the memory comes in the form of an element in the query space, and the memory returns the output that was most likely related to the query element, with a certain degree of confidence. This model also guaranteed that if a query element was never

*Advisor: Steve A. Zdancewic (stevez@cis.upenn.edu).

stored in the memory, a relation would not be found for that element, with the same degree of confidence as above.

Willshaw's paper was the first to present a model for emulating an associative memory probabilistically, without storing each element pair explicitly, and much of the following research on memory compression methods for probabilistic associative memory were based on this work. While Willshaw's model enforced a strict mapping between the stored elements, later work also found ways to give likely matches for elements which the memory has never seen.

4. SYSTEM IMPLEMENTATION

Research in associative memory has close connections to neural networks. Our brains function as associative memories that can retrieve memories based on similarity between stimuli and the memories themselves. Indeed, Norman and Bobrow [1] construct a memory model which describes how minds retrieve items in memory from partial descriptions, even if some part of the description is incorrect or inaccurate.

Applying this memory model to computation, Hinton [3] draws a connection between Norman and Bobrow's idea of retrieving memories based on partial descriptions and content-addressable memories. Hinton then notes that although this content-addressable memory is a useful representation, it is difficult to efficiently "search for content" through a memory space. Hinton ends by claiming that while this model is conceptually not difficult, it is difficult to actually implement it. He mentions that an approach to this problem that has arisen recently is with stochastic processes – heralding a probabilistic approach to this associative memory problem. Thus, both Willshaw and Hinton offer probabilistic models of dealing with associative memories.

5. SYSTEM PERFORMANCE

Later on, the development of Hopfield networks introduced a new form of neural network that handles associative memory. These neural networks may be used as content-addressable memory by storing memories on a constant number of nodes and connections, with a similar loss of fidelity as the Willshaw paper's model. While it had been shown that the theoretical maximum capacity for these networks can asymptotically approach twice the network size [2], the initial Hopfield model provided relatively low capacity. Work by Palm [4] details the history of neural associative memories (including the Willshaw model), the limitations of the initial Hopfield model, and the improvements that have been made upon it. Namely, it mentions using the Hebb learning rule over the original Hopfield learning rule and the importance of sparse coding of patterns.

Sparse coding in particular is quite important in Hopfield networks (and in unsupervised machine learning in general). For various applications, it is beneficial to utilize structural similarities to represent the input in a smaller space. In particular, quite accurate reconstructions can be created from damaged or noisy images, where details that were not even present in the input can be recovered. While this particular example primarily has its uses in, say, medical imaging [6], sparse coding has been incredibly useful in increasing the capacity of these neural associative memories.

6. REMAINING WORK

The OCaml language will be used to implement this probabilistic associative memory. Based on the specific probability distribution and model in use, the library would return a different memory value for each query. As such, our library will have user-defined support for different probability paradigms. After the library is created, applications will be created with the library to demonstrate the benefits of using this implementation of probabilistic associative memory.

6.1 Technical Challenges

7.

8. REFERENCES

[1] Norman D. A. and D. G. Bobrow. Descriptions: An intermediate stage in memory retreival. In *Cognitive Psychology*, 1979.

[2] E Gardner. The space of interactions in neural networks models. In *Journal of Physics A: Mathematical and General 21*, volume 21, 1987.

[3] Geoffrey Hinton. Distributed representations. In *Research Showcase @ CMU*, 1984.

[4] Günther Palm. Neural associative memories and sparse coding. In *Neural Networks*, volume 37, 2012.

[5] D. J. Willshaw, O. P. Buneman, and H. C. Longuet-Higgins. Non-holographic associative memory. In *Nature*, 1969.

[6] Shuyang Yang, Yaxin Sun, Yiguang Chen, and Licheng Jiao. Structural similarity regularized and sparse coding based super-resolution for medical images. In *Biomedical Signal Processing and Control*, 2012.