

# Sequential learning using temporal context

Karthik H. Shankar, Udaya K. K. Jagadisan, and Marc W. Howard  
Syracuse University

## Abstract

The temporal context model (TCM) has been extensively applied to recall phenomena from episodic memory. Here we use the same formulation of temporal context to construct a sequential learning model called the predictive temporal context model (pTCM) to extract the generating function of a language from sequentially presented words. In pTCM, temporal context is used to generate a prediction vector at each step of learning and these prediction vectors are in turn used to construct semantic representations of words on the fly. The semantic representation of a word is defined as the superposition of prediction vectors that occur prior to the presentation of the word in the sequence. Here we create a formal framework for pTCM and prove several useful results. We explore the effect of manipulating the parameters of the model on learning a sequence of words generated by a bi-gram generating function. In this simple case, we demonstrate that feeding back the constructed semantic representation into the temporal context during learning improves the performance of the model when trained with a finite training sequence from a language with equivalence classes among some words. We also describe pTCM<sub>A</sub>, a variant of the model that is identical to pTCM at steady state. pTCM<sub>A</sub> has significant computational advantages over pTCM and can improve the quality of its prediction for some training sequences.

## Introduction

Sequential learning algorithms that extract the generating function of a language from sample sequences of symbols, are of high significance not only to the machine learning community, but also to the cognitive psychology community. A variety of experiments indicate that humans are adept at reconstructing the generating functions of toy languages with artificial grammars (e.g. Mintz, 2002) and are able to discover underlying equivalence classes among the symbols if they exist. The success of any model in learning the generating function of the language can be directly measured either by letting the generat-

---

Address correspondence to Karthik Shankar, karthik@memory.syr.edu, or Marc Howard, marc@memory.syr.edu. Supported by NIH grant 1-R01 MH069938 to MWH.

ing function reconstructed by the model generate novel sequences or by letting the model predict subsequent symbols on a test sequence.

Training neural networks on the sample sequences based on supervised learning paradigm has been a very popular approach to model the underlying learning mechanisms (e.g. Elman, 1991; Plate, 2003). Christiansen and Chater (2001) review the successes of such connectionist modeling methods in making contact with a variety of psycholinguistic data. The slowness of the supervised learning algorithms make them very inefficient in learning the structure of natural languages. On the other hand, the unsupervised learning models can be very fast and have proven successful in capturing certain features of natural languages. Some of these models, like the hyperspace analogue of language (HAL, Lund & Burgess, 1996), the bound encoding of the aggregate language environment (BEAGLE, Jones & Mewhort, 2007) and the constructed semantics model (CSM, Kwantes, 2005), can be classified as incremental learning algorithms, because they construct the semantic representation of symbols “on the fly”.<sup>1</sup> That is, as the sequence of words (symbols) from the training corpora is presented, the semantic representations of the words are gradually updated. Other models like latent semantic analysis (LSA, Landauer & Dumais, 1997), the Topic model (Griffiths, Steyvers, & Tenenbaum, 2007), and the automatic distillation of structure (ADIOS) model (Solan, Horn, Ruppin, & Edelman, 2005), need to access the entire corpus of training sequences prior to beginning their process of constructing semantic representations. Hence, they do not describe the process of learning *per se*, but rather describe statistical computations that are taken after all of experience has already occurred.

A common feature of all these models is that they all rely either explicitly or implicitly on being able to maintain a context and vary it as the sequence is presented. For instance, simple recurrent networks (SRN, Elman, 1991) and holographic recurrent networks (HRN, Plate, 2003) accomplish the task of sequence prediction primarily because there exists a context layer of neurons which sort of stores the prior states of the sequence. In LSA, the Topic model and CSM, words (symbols) that appear in a particular document are defined as appearing in a common context. Although it is not explicit in these treatments, these models are consistent with a representation of context that changes quite slowly—words at the beginning and end of a document with hundreds of words are treated as being part of the same context.<sup>2</sup> In HAL and BEAGLE, the context in which a word is presented is defined as a combination of words presented in closer temporal contiguity. For instance, in BEAGLE, words that appear within the same sentence are rated to be in the same context—context changes over a more rapid time scale in these models than in LSA, the Topic model or CSM.

In this paper, we describe a sequential learning model called pTCM (predictive temporal context model), which can be viewed as a neural network model with a context layer. The context layer has much in common with context in the SRNs. It encodes the prior states of the sequence and changes gradually as the sequence is presented. The precise

<sup>1</sup>The inclusion of the Kwantes (2005) model in this list of incremental learning models relies on its description of a semantic representation that is computed (but not stored) in response to a probe item.

<sup>2</sup>A term-by-document matrix, such as that used as the starting point of these models, is consistent with an explicit representation of context in a very large embedding dimension in which the state of context for each document is chosen randomly. If these context vectors form basis vectors for the embedding space, then a row of a term-by-document matrix can be understood as the sum of the contexts in which that word appeared.

formulation of the context and its evolution is based on the temporal context model (TCM, Howard & Kahana, 2002; Howard, Fotedar, Datey, & Hasselmo, 2005; Sederberg, Howard, & Kahana, 2008). A crucial difference between pTCM and SRNs is that the learning process in pTCM is unsupervised and does not use backpropagation as do the SRNs.

As the training sequence is presented to the model, at each step, the model generates a prediction for the subsequent symbol that is to occur in the sequence and simultaneously builds up a semantic representation vector for the symbol that is presented. It turns out that the semantic representation gives a natural way to categorize symbols into equivalence classes (we demonstrate this property with respect to a toy language). The semantic representation is built based on the idea that words occurring in similar contexts are interchangeable (Harris, 1951) and paradigmatically related to each other. For instance, consider the sentence, *The baker reached into the oven and pulled out the floob*. To the extent that the model predicts effectively, it will construct the semantic representation of the unknown word *floob* with words like *pizza* and *bread*, that are predicted to occur in that temporal context. An important aspect that distinguishes pTCM from other models is that the semantic representation constructed on the fly is fed back into the learning process. For example, if the word *floob* occurs in the corpus for the second time, its previously constructed semantic representation (made up of words like *pizza* and *bread*) influences the context that follows the second presentation of the word. In the spirit of the distributional hypothesis, where words that occur in similar contexts carry similar meanings, we expect this procedure to speed learning through generalization. When the sample sequence is too short for sufficient statistics of the language to be sampled, such a “learning through generalization” could be very useful (we demonstrate this property with respect to a toy language).

This paper is split into three parts. In the first part, we formally describe the mathematical aspects of pTCM. In the second part, we describe the behavior of pTCM when trained on a corpus long enough to have sampled all possible statistics of the language, and describe a variant of the model called pTCM<sub>A</sub> (which is not a sequential learning model). It has been a challenge to employ pTCM to learn (natural) large languages, because the vector and matrix multiplications involved in the algorithm are computationally very time expensive. pTCM<sub>A</sub> alleviates much of the computational complexity and can potentially scale up to learn large languages. In this paper, we do not illustrate the performance of the model on a real language (but see Howard, Shankar, and Jagadisan (submitted)). In the third part of the paper, we illustrate the properties of the model by training it on a toy language with a precisely defined generating function, this helps us to evaluate the model with mathematically precise measures. The ability of the model to construct useful semantic representations after training on corpus of a real language and the utility of this semantic representation in describing semantic effects on episodic memory tasks will be addressed elsewhere.

### The predictive temporal context model (pTCM)

In this section, we will formalize the predictive temporal context model (pTCM). We first lay out the mathematical framework, then discuss the learning rules that will be employed. Future sections will derive important results and explore the effect of the model’s parameters on learning.

Consider a language which has a finite number of unique words. We shall refer to the unique words as “items” and denote them by Greek letters ( $\alpha, \beta, \dots$ ). We shall pick a long sequence of words using the generating function of the language and refer to it as a token list. We will refer to the word (item) in the  $i^{\text{th}}$  position of the token list as the “ $i^{\text{th}}$  token”. The challenge faced by any statistical learning model is to use the token list to learn to reproduce the generating function that specifies the language. We will evaluate the model by asking how nearly the sequences generated by the model resemble the sequences generated by the language. In general, the token list used for learning will only describe a small fraction of all possible meaningful sequences that can be generated from the language. A better (more efficient) model will be able to approximate the generating function of the language with a shorter token list. The accuracy of the prediction can be estimated by comparing the model’s generating function with the true generating function of the language.

### *Mathematical framework*

In this paper, we will adopt the Dirac notation by denoting column vectors as a “ket,” e.g.,  $|\mathbf{v}\rangle$  and row vectors as a “bra,”  $\langle\mathbf{v}|$ . The primary advantage of this notation is that it becomes intuitive to visually identify the inner product as a “braket”  $\langle\mathbf{v}|\mathbf{u}\rangle$ ; the outer product appears as a “ketbra”:  $|\mathbf{u}\rangle\langle\mathbf{v}|$ . That is,  $\langle\mathbf{v}|\mathbf{u}\rangle$  “compresses” to a scalar whereas  $|\mathbf{u}\rangle\langle\mathbf{v}|$  “expands” to a matrix. Figure 1 provides a graphical summary of the model that may facilitate the following description.

We define an orthonormal vector  $|\mathbf{f}_\alpha\rangle$  corresponding to each word  $\alpha$ . The set of all these vectors form the basis of the word space. The vector corresponding to the  $i^{\text{th}}$  token in the token list is  $|\mathbf{f}_i\rangle$ , which is just the  $|\mathbf{f}_\alpha\rangle$  corresponding to the word presented as the  $i^{\text{th}}$  token. Next, we define a context space. We shall represent the context vector right after learning the  $i^{\text{th}}$  token by  $|\mathbf{t}_i\rangle$ . We assume that the context vector is always normalized such that the sum over all its components is equal to one. This differs from most previous treatments of TCM (but see Howard, Jing, Rao, Probyn, & Datey, 2009). The choice of  $\mathcal{L}^1$  norm is important in enabling many of the steps of the proofs here. The dimensions of the  $\mathbf{f}$  space and  $\mathbf{t}$  space can in principle be different.

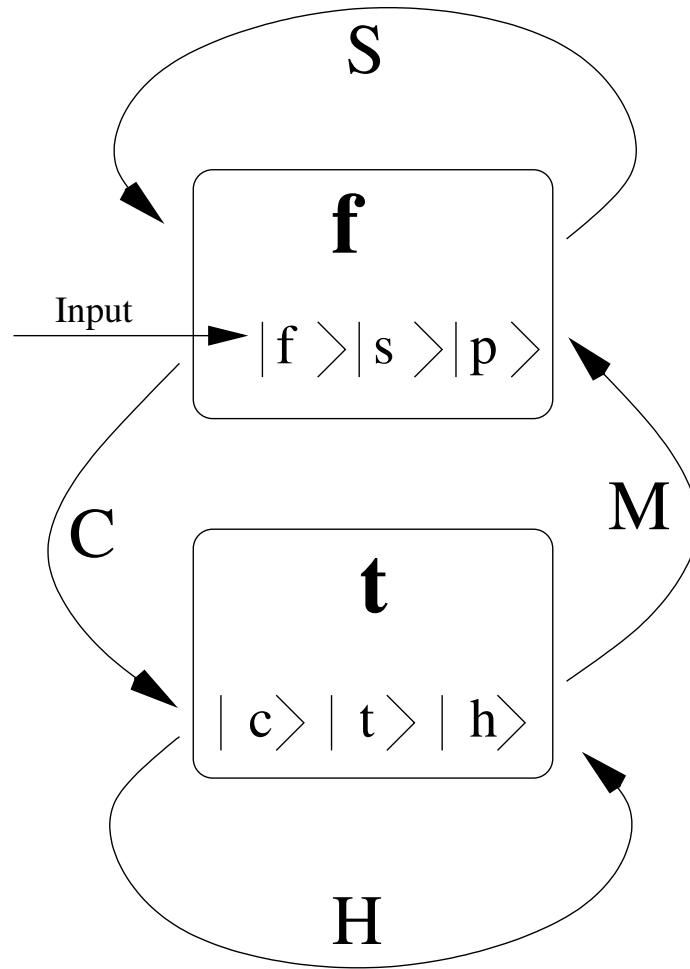
We next define a set of operators,  $\mathbf{C}$ ,  $\mathbf{M}$ ,  $\mathbf{H}$  and  $\mathbf{S}$ , that act on the  $\mathbf{f}$  space and  $\mathbf{t}$  space. The operator  $\mathbf{C}$  transforms each of the basis vectors of the  $\mathbf{f}$  space,  $|\mathbf{f}_\alpha\rangle$  into a vector in the  $\mathbf{t}$  space, which we denote by  $|\mathbf{c}_\alpha\rangle$ .

$$|\mathbf{c}_\alpha\rangle = \mathbf{C}|\mathbf{f}_\alpha\rangle \tag{1}$$

Unless otherwise noted, we will assume that  $\mathbf{C}$  is the identity operator and that the dimensionality of the  $\mathbf{t}$  space is the same as the dimensionality of the  $\mathbf{f}$  space. The operator  $\mathbf{M}$  acts on a context vector from  $\mathbf{t}$  space and yields a vector in the  $\mathbf{f}$  space. More specifically, at any step of the learning process,  $\mathbf{M}$  acts on the prevalent context vector and yields a prediction vector

$$|\mathbf{p}_i\rangle = \mathbf{M}|\mathbf{t}_i\rangle. \tag{2}$$

Note that  $|\mathbf{p}_i\rangle$  refers to a particular vector in the  $\mathbf{f}$  space. We will see that the properties of  $\mathbf{M}$  are such that  $|\mathbf{p}_i\rangle$  can be thought of as a prediction of the item that will be presented at time step  $i + 1$ .



*Figure 1.* A pictorial representation of the mathematical framework underlying the model. The two vector spaces  $\mathbf{f}$  and  $\mathbf{t}$  are denoted by blocks, and the four operators  $\mathbf{M}$ ,  $\mathbf{C}$ ,  $\mathbf{H}$ ,  $\mathbf{S}$  are denoted by arrows: the tail of the arrow indicates the space on which the operator acts, and the head of the arrow indicates the space onto which the operator projects. The vectors  $|f\rangle$ ,  $|s\rangle$  and  $|p\rangle$  live in the  $\mathbf{f}$ -space, while the vectors  $|c\rangle$ ,  $|h\rangle$  and  $|t\rangle$  live in the  $\mathbf{t}$ -space.

The operator  $\mathbf{S}$  acts on a vector from the  $\mathbf{f}$  space and yields another vector in the  $\mathbf{f}$  space. More specifically,  $\mathbf{S}$  acts on the basis vectors of the  $\mathbf{f}$  space,  $|\mathbf{f}_\alpha\rangle$  to yield a vector  $|\mathbf{s}_\alpha\rangle$  which we can think of as a semantic representation of word  $\alpha$ .

$$|\mathbf{s}_\alpha\rangle = \mathbf{S}|\mathbf{f}_\alpha\rangle. \quad (3)$$

The operators we have discussed so far,  $\mathbf{C}$ ,  $\mathbf{M}$ , and  $\mathbf{S}$  are all linear operators.

For completeness, we also include an operator  $\mathbf{H}$  that acts on a vector from the  $\mathbf{t}$  space and yields another vector in the  $\mathbf{t}$  space. More specifically,  $\mathbf{H}$  acts on  $|\mathbf{c}_\alpha\rangle$  to yield  $|\mathbf{h}_\alpha\rangle$ .

$$|\mathbf{h}_\alpha\rangle = \mathbf{H}|\mathbf{c}_\alpha\rangle. \quad (4)$$

In general,  $\mathbf{H}$  can be a (highly) nonlinear operator that recovers some combination of the previous states of temporal context in which item  $\alpha$  was presented. We hypothesize that  $\mathbf{H}$  describes the output of the hippocampal formation (Howard et al., 2005). We hypothesize that this act of recovery corresponds to successful episodic retrieval. Here our focus is on discovery of the consistent structural relations among words in a language rather than on episodic memory for specific instances in which words were presented, so we will not dwell much on  $\mathbf{H}$  except to locate it in the formal framework we develop.

### *Learning Rules*

Having defined the spaces, operators and the vectors in the model, we now describe the learning rules that specify the values of these quantities in response to a token list. At every step of learning, the operators  $\mathbf{M}$ ,  $\mathbf{S}$ , and  $\mathbf{H}$ , and the context vector  $|\mathbf{t}_i\rangle$  are updated. To keep the model simple, we assume that the operator  $\mathbf{C}$  does not evolve with learning. To further simplify matters, assume throughout the remainder of this paper that the dimensions of the  $\mathbf{t}$  space and  $\mathbf{f}$  space are the same and the operator  $\mathbf{C}$  is the identity matrix. We should point out that there are much more complex learning rules that can be imagined within the mathematical framework of pTCM; we use the simplest possible rules here to illustrate the basic properties of the model.

*Evolution of  $\overline{\mathbf{M}}$ .* We first define an operator  $\overline{\mathbf{M}}$  which stores the item-context association at each step of the learning process. As we step through the token list, the operator  $\overline{\mathbf{M}}$  is updated by adding on to itself an outer product of the corresponding item vector and the prevalent context vector. Hence, at any stage of learning,  $\overline{\mathbf{M}}$  is given by

$$\overline{\mathbf{M}} = \sum_i |\mathbf{f}_i\rangle\langle\mathbf{t}_{i-1}|, \quad (5)$$

where the summation goes from the beginning of the token list to the appropriate point on the token list.  $\overline{\mathbf{M}}$  is essentially unchanged from the outer product matrix referred to as  $\mathbf{M}^{TF}$  in other applications of TCM (e.g., Sederberg et al., 2008; Howard, Kahana, & Wingfield, 2006). By factoring out the word frequency of item  $\alpha$ ,  $w_\alpha$ , we can re-express  $\overline{\mathbf{M}}$  as

$$\overline{\mathbf{M}} = \sum_\alpha w_\alpha |\mathbf{f}_\alpha\rangle\langle\mathbf{t}_\alpha|, \quad (6)$$

where  $|\underline{\mathbf{t}}_\alpha\rangle$  is the normalized average context vector.

$$w_\alpha |\underline{\mathbf{t}}_\alpha\rangle = \sum_i \langle \mathbf{f}_i | \mathbf{f}_\alpha \rangle |\mathbf{t}_{i-1}\rangle$$

Since  $|\mathbf{f}_\alpha\rangle$ 's form the orthonormal basis of the  $\mathbf{f}$  space, note that the term  $\langle \mathbf{f}_i | \mathbf{f}_\alpha \rangle$  acts as a Kronecker delta function whose value is unity whenever the  $i^{\text{th}}$  token is  $\alpha$  and is zero otherwise. Hence, the above equation is essentially the summation over all those context vectors that existed prior to the presentation of the word  $\alpha$ . By defining a diagonal matrix of word frequencies

$$\mathbf{W} = \sum_\alpha w_\alpha |\mathbf{f}_\alpha\rangle \langle \mathbf{f}_\alpha|,$$

we can write  $\overline{\mathbf{M}} = \mathbf{W}\mathbf{M}$ , where

$$\mathbf{M} = \sum_\alpha |\mathbf{f}_\alpha\rangle \langle \underline{\mathbf{t}}_\alpha|, \quad (7)$$

That is,  $\mathbf{M}$  is the row normalized version of  $\overline{\mathbf{M}}$  (i.e., each row of  $\mathbf{M}$  sums to one). At each step of learning  $\overline{\mathbf{M}}$  is updated, and consequently  $\mathbf{M}$  is also updated. In fact, we can think of the operator  $\mathbf{M}$  as being updated in an online fashion without having to store the unnormalized  $\overline{\mathbf{M}}$ . If the word  $\alpha$  occurs for the  $w^{\text{th}}$  time in the token list, and if the prevalent context vector is  $|\mathbf{t}\rangle$ , then the corresponding row of  $\mathbf{M}$ , denoted by  $\langle \underline{\mathbf{t}}_\alpha|$  is updated in the following way.

$$\langle \underline{\mathbf{t}}_\alpha| \rightarrow \left(1 - \frac{1}{w}\right) \langle \underline{\mathbf{t}}_\alpha| + \frac{1}{w} \langle \mathbf{t}|. \quad (8)$$

It is straightforward to see that the operator  $\mathbf{M}$  constructed this way is essentially equivalent to the  $\mathbf{M}$  constructed from the steps leading to Eq. 7.

*Evolution of  $\mathbf{S}$ .* At each stage of learning,  $\mathbf{M}$  operates on the prevalent context vector  $|\mathbf{t}_i\rangle$  to yield a prediction vector  $|\mathbf{p}_i\rangle$ , and this prediction vector is used to update the operator  $\mathbf{S}$ . We first define an operator  $\overline{\mathbf{S}}$  which is updated at each step of learning by adding on to itself an outer product of the prior prediction vector and the item vector. Hence, at any stage of learning,  $\overline{\mathbf{S}}$  is given by

$$\overline{\mathbf{S}} = \sum_i |\mathbf{p}_{i-1}\rangle \langle \mathbf{f}_i|, \quad (9)$$

$\mathbf{S}$  is then defined as the column normalized version of  $\overline{\mathbf{S}}$ , i.e, normalize each column to sum to one.  $\mathbf{S}$  can then be expressed as

$$\mathbf{S} = \|\overline{\mathbf{S}}\|_{\text{col}}, \quad \mathbf{S} = \sum_\alpha |\mathbf{s}_\alpha\rangle \langle \mathbf{f}_\alpha|, \quad (10)$$

where we use the  $\|\cdot\|$  notation to reflect normalization such that the sum of all components of the vector within it is one. The subscript ‘‘row’’ or ‘‘col’’ implies that the matrix inside the double bar is normalized such that every row or column sums up to one. In the above equation,  $|\mathbf{s}_\alpha\rangle$  is the normalized sum over all the prediction vectors evaluated just before the item  $\alpha$  occurs each time in the token list.

$$|\mathbf{s}_\alpha\rangle = \left\| \sum_i \langle \mathbf{f}_i | \mathbf{f}_\alpha \rangle |\mathbf{p}_{i-1}\rangle \right\| \quad (11)$$

This vector  $|\mathbf{s}_\alpha\rangle$  can be thought of as the semantic representation of the item  $\alpha$ . The learning rule works to make  $|\mathbf{s}_\alpha\rangle$  similar to other items that were presented in similar contexts. In this way, the model implements a definition of semantic similarity that is based on contextual interchangeability (Harris, 1951; Miller & Charles, 1991), at least with respect to preceding contexts.

*Evolution of the context vector  $|\mathbf{t}_i\rangle$ .* In TCM, the current state of context changes gradually from moment to moment. A parameter  $\rho$  that lies in the range  $(0, 1)$  encodes the rate at which the context changes. At each step of learning, the context vector is updated in the following way.

$$|\mathbf{t}_i\rangle = \rho|\mathbf{t}_{i-1}\rangle + (1 - \rho)|\mathbf{t}^{\text{IN}}\rangle \quad (12)$$

As the value of  $\rho$  grows closer to one, context changes more slowly; as the value of  $\rho$  approaches zero, context changes more rapidly. The vector  $|\mathbf{t}^{\text{IN}}\rangle$  is the input that drifts the context away from its previous state, and it depends on the current item from the token list.

$$|\mathbf{t}^{\text{IN}}\rangle = \gamma|\mathbf{h}^{\text{IN}}\rangle + (1 - \gamma)|\mathbf{c}^{\text{IN}}\rangle. \quad (13)$$

Here  $|\mathbf{h}^{\text{IN}}\rangle$ , the hippocampal input, corresponds to the input from the  $\mathbf{H}$  operator and  $|\mathbf{c}^{\text{IN}}\rangle$ , the cortical input, corresponds to the input from the  $\mathbf{C}$  operator. If the current item is  $\alpha$ , then  $|\mathbf{h}^{\text{IN}}\rangle$  is taken to be  $|\mathbf{h}_\alpha\rangle$  and  $|\mathbf{c}^{\text{IN}}\rangle$  is taken to be

$$|\mathbf{c}^{\text{IN}}\rangle = \mathbf{C}\left[\phi|\mathbf{s}_\alpha\rangle + (1 - \phi)|\mathbf{f}_\alpha\rangle\right] \quad (14)$$

That is, the cortical input to the context is taken to be a combination of the semantic representation of item  $\alpha$  and the fixed representation of item  $\alpha$ . The parameter  $\phi$  is a free parameter of the model that lies in the range  $(0, 1)$ ; its value determines the ratio of the semantic component to the fixed component in the input pattern  $|\mathbf{c}^{\text{IN}}\rangle$ . The presence of the fixed representation avoids a collapse of the context vectors to a point (see Howard et al., 2005, 2009, for a discussion), hence we shall take  $\phi$  to be strictly less than one. In this paper, we will see that the presence of the semantic representation ( $\phi \neq 0$ ) enables the model to generalize from the sequences it has observed to sequences it has not yet observed, analogous to the recovery of  $|\mathbf{t}_{i-1}\rangle$  in previous applications of TCM (Rao & Howard, 2008; Howard et al., 2009).

In this paper, we focus on construction of semantic memory based on sequential learning of a token list. Generalization and integration across disparate experiences is a key feature of declarative memory (e.g. Eichenbaum, 2000). In our treatment of semantic memory, we use  $|\mathbf{s}_\alpha\rangle$  to accomplish this function rather than  $|\mathbf{h}_\alpha\rangle$ . So, throughout this paper, we will always take  $\gamma$  to be zero. Our semantic representation  $|\mathbf{s}_\alpha\rangle$  is composed of the prediction vectors that were present when item  $\alpha$  was presented rather than the prior contexts in which item  $\alpha$  was presented (Rao & Howard, 2008; Howard et al., 2009). This is essential for sequentially organized materials, such as language. The semantic representation for a word  $\alpha$ ,  $|\mathbf{s}_\alpha\rangle$  becomes similar to the semantic representation of other words that would fit into similar contexts, rather than becoming similar to the preceding contexts themselves. To illustrate the utility of this, imagine that we have learned English and we are presented with a novel word in the following sentence: *The baker reached into the oven and pulled out the floob.* Here, the semantic representation of *floob* should not be made similar to the preceding

context, i.e., *pulled, baker, oven*, etc. Rather, the semantic representation of *floob* should be made similar to words that would have fit into the same context like *bread, pizza, pie*, etc.

### Steady state behavior of pTCM

Here we prove three useful properties of pTCM by examining its steady-state behavior after training on a language. First, we demonstrate that at the steady state, the semantic representation  $\mathbf{S}$  can be reexpressed in terms of the prediction operator  $\mathbf{M}$ . Second, we derive a method for calculating the steady state of the model without explicitly calculating  $|\mathbf{p}\rangle$  at each time step. We refer to the results of this method as pTCM<sub>A</sub>. For languages described by large training sets, this can result in saving a tremendous amount of processing time. Finally, we prove that for languages that can be characterized by a bi-gram transition matrix, at steady state pTCM can generate sequences with precisely the same statistics as the generating function. In the following section, we explore the utility of nonzero values of  $\phi$  and pTCM<sub>A</sub> in learning languages far from steady state.

Let us suppose that the generating function of our language is ergodic. That is, all possible states are sampled according to their specified probability of occurrence in a sufficiently long sequence of words. One can understand that a probabilistic random number generator will do this as a consequence of law of large numbers and the central limit theorem. At the same time, one can also imagine several scenarios where the assumption of ergodicity would fail. (i) Consider a generating function that is not finite. That is, the total number of words might be finite, but the number of rules are infinite (like a Markov chain of infinite order) or keep changing with time. Quite possibly, natural languages could fall into such a category. (ii) Even in the case of a finite generating function with finite number of words, one can imagine pathological generating functions that can fail to be ergodic. For instance, imagine a generating function that with probability 1/2 selects an infinitely long sequence from the words 1, 3, and 5, and with probability 1/2 selects an infinitely long sequence from the words 2, 4, and 6. Any sequence from this generating function will get stuck in one of the two subsets of words. In the case of bi-gram languages, discussed in the next section, we will show that such a generating function can be represented as a reducible transition matrix. We will ignore such pathological languages except to note that, in this case it can be understood as two separable sublanguages that would each be ergodic.

For ergodic generating functions, if the number of times a specific word (say  $\alpha$ ) occurs in a token list of length  $N$  is  $n_\alpha$ , then for sufficiently large  $N$ , the probability distribution of  $n_\alpha/N$  is approximately normal with variance proportional to  $1/N$ . We shall justify this statement with respect to a generating function with trivial grammatical structure where the probability for any word  $\alpha$  to occur in any given position is  $\mathcal{P}_\alpha$ , independent of the preceding or following words in the sequence. In a token list of length  $N$  generated from this language, the number of times the word  $\alpha$  occurs ( $n_\alpha$ ) is simply given by the binomial distribution, with mean  $N\mathcal{P}_\alpha$  and variance  $N\mathcal{P}_\alpha(1 - \mathcal{P}_\alpha)$ . For large  $N$ , such that  $N\mathcal{P}_\alpha \gg 1$ , this is an excellent approximation to the normal distribution. The probability distribution of  $n_\alpha/N$  will then be approximately normal with mean  $\mathcal{P}_\alpha$ , and variance  $\mathcal{P}_\alpha(1 - \mathcal{P}_\alpha)/N$ . For more complicated languages where the probability for any word  $\alpha$  to occur in a given position depends on the preceding words,  $n_\alpha$  will be more complicated than a binomial distribution. Nevertheless, the central limit theorem ensures that the probability distribution of  $n_\alpha/N$  is approximately normal with a variance proportional to  $1/N$ .

By choosing  $N$  large, we can make the probability distribution of  $n_\alpha/N$  arbitrarily narrow around the mean. Let the area under this probability distribution curve within an  $\epsilon$ -neighborhood either side of the mean be  $1 - p$ , and the area under the curve in the region outside the  $\epsilon$ -neighborhood (tails of the distribution) be  $p$ . By increasing  $N$ , we can simultaneously push  $\epsilon$  and  $p$  arbitrarily close to zero. More formally, for any given  $\epsilon > 0$  and  $p > 0$ , we can choose a large  $N$  such that, for all  $n \geq N$ ,  $n_\alpha/n$  can be fixed within an  $\epsilon$  neighborhood of its mean value (say  $\omega_\alpha$ ) with a probability greater than  $1 - p$ . That is, for all  $n \geq N$ ,

$$\mathbf{Prob} \left( \left| \frac{n_\alpha}{n} - \omega_\alpha \right| < \epsilon \right) > 1 - p. \quad (15)$$

Note that for each word  $\alpha$  in the language,  $\omega_\alpha$  is positive and  $\sum_\alpha \omega_\alpha = 1$ .

An important point to note is that the above equation is not the only property that emerges out of ergodicity. We can do much more than just predict accurately the number of times a word occurs in a sufficiently long sequence generated by an ergodic generating function. We can even predict the entire distribution of (finite length) sequences which precede or follow a word in a sufficiently long sequence. Temporal context in pTCM is constructed from the preceding sequence of words whose distributions are subject to ergodicity. Hence, we should be able to accurately predict the average temporal context in which a word occurs, and even the statistical distribution of the temporal contexts in which a word occurs in sufficiently long sequence.

Since the operators  $\mathbf{M}$  and  $\mathbf{S}$  in pTCM are essentially constructed from the temporal context vectors of each token in the sequence, and since each token is weighed equally (that is, irrespective of whether it occurs early in the sequence or later in the sequence) in constructing these operators (see Eq. 5 and 9), they will converge to a steady state almost certainly when trained on a sufficiently long sequence. Although the non-normalized operators  $\bar{\mathbf{M}}$  and  $\bar{\mathbf{S}}$  will grow without bounds as the token list becomes longer, the normalized operators  $\mathbf{M}$  and  $\mathbf{S}$  will converge to a steady state.

One can think of alternative learning rules in pTCM. Consider a learning rule wherein the tokens are weighed unequally while constructing the  $\mathbf{M}$  and  $\mathbf{S}$  operators (unlike Eq. 5 and 9). For instance, we might imagine a rule where a token is encoded with a weight that depends on the current word frequency of the corresponding word, thus depending on the position of the token on the token list. In such a situation, there is no reason to expect a convergence to steady state. But for the learning rules in pTCM (Eqs. 5 and 9), the operators  $\mathbf{M}$  and  $\mathbf{S}$  will converge to a steady state almost certainly.

To analyze the convergence of matrices to steady state values, we define a scalar measure  $s$ , which just sums over the absolute values of all the entries of the matrix. Denoting the  $(i, j)^{th}$  element of a matrix  $\mathbf{X}$  by  $X_{ij}$ , we define the measure  $s$  through its action on  $\mathbf{X}$ ,

$$s(\mathbf{X}) = \sum_{i,j} |X_{ij}|, \quad (16)$$

Clearly,  $s(\mathbf{X} - \mathbf{Y})$  serves as a distance measure between  $\mathbf{Y}$  matrices. We can define an analogous scalar measure that acts on vectors by summing over the absolute values of all its components. For notational simplicity, we shall denote this measure also by  $s(\cdot)$ . If  $\mathbf{u}$  and  $\mathbf{v}$  are two vectors, then clearly  $s(\mathbf{u} - \mathbf{v})$  will serve as a distance measure between the two vectors.

The statement that  $\mathbf{M}$  and  $\mathbf{S}$  converge to a steady state limit almost certainly as the token list becomes arbitrarily long can now be precisely formulated. For any given arbitrarily small  $\epsilon > 0$ , and  $p > 0$ , there exists a  $N$ , such that for any token list of length  $n \geq N$ , we have

$$\mathbf{Prob}\left(\left[s(\mathbf{M} - \mathbf{M}_n) < \epsilon\right] \& \left[s(\mathbf{S} - \mathbf{S}_n) < \epsilon\right]\right) > 1 - p \quad (17)$$

Here  $\mathbf{M}_n$  and  $\mathbf{S}_n$  denote the operators obtained from training the model on a token list of length  $n$ , and  $\mathbf{M}$  and  $\mathbf{S}$  correspond to the steady state values.

Let us now restrict our focus to those generating functions, which when used to train pTCM will satisfy the convergence criteria, namely Eq. 15 and 17. In this section, we show that we can obtain analytic relationships between the steady state values  $\mathbf{M}$  and  $\mathbf{S}$ . First, we show that  $\mathbf{S}$  can be written as a function of  $\mathbf{M}$ . Next, we derive a convenient expression for  $\mathbf{M}$  as a function of the  $\mathbf{M}$  obtained with  $\phi = 0$ .

*Claim 1:  $\mathbf{S}$  can be calculated from  $\mathbf{M}$*

The operators  $\mathbf{M}$  and  $\mathbf{S}$  to which the sequence of  $\mathbf{M}_n$ 's and  $\mathbf{S}_n$ 's converge almost certainly (as defined in Eq. 17) satisfy the following relationship.

$$\mathbf{S} = \|\mathbf{M}\mathbf{M}'\|_{\text{col}} \quad (18)$$

Before proving Claim 1, let us first motivate it. Recalling the definition of  $\bar{\mathbf{S}}$  from Eq. 9, we can rewrite  $\bar{\mathbf{S}}$  after time step  $i$  as

$$\begin{aligned} \bar{\mathbf{S}}_i &= \sum_{j=0}^i |\mathbf{p}_{j-1}\rangle\langle \mathbf{f}_j| \\ &= \sum_{j=1}^i \mathbf{M}_{j-1} |\mathbf{t}_{j-1}\rangle\langle \mathbf{f}_j|, \end{aligned} \quad (19)$$

where the second line follows from the definition of  $|\mathbf{p}\rangle$  (Eq. 2). The sum of the outer product  $|\mathbf{t}_{j-1}\rangle\langle \mathbf{f}_j|$  is just  $\bar{\mathbf{M}}_i'$ . If  $\mathbf{M}_{j-1}$  were not changing over  $j$ , then we would be able to quickly verify Claim 1. After the steady state has been reached,  $\mathbf{M}_j$  is constant with respect to  $j$  and the claim is easily established. We now proceed more formally along these lines.

For a given small  $\epsilon > 0$  and  $p > 0$ , let us choose a  $N$  large enough such that Eq. 15 and Eq. 17 holds true. Now consider a token list of length  $2N$  from our ergodic generating function. We will use the first  $N$  tokens of the list to train the model according to the learning rules from the previous section and store the operators as  $\mathbf{M}_N$  and  $\mathbf{S}_N$ . Then we will continue the learning process over the remaining  $N$  tokens and refer to the operators aggregated during the second  $N$  tokens as

$$\bar{\delta\mathbf{M}} = \sum_{i=N+1}^{2N} |\mathbf{f}_i\rangle\langle \mathbf{t}_{i-1}|, \quad \bar{\delta\mathbf{S}} = \sum_{i=N+1}^{2N} |\mathbf{p}_{i-1}\rangle\langle \mathbf{f}_i|. \quad (20)$$

It will now be convenient to define  $\delta\mathbf{M}$  and  $\delta\mathbf{S}$  to be the row normalized and column normalized versions of  $\overline{\delta\mathbf{M}}$  and  $\overline{\delta\mathbf{S}}$ . Since  $\overline{\mathbf{M}}$  and  $\overline{\mathbf{S}}$  are defined to be incrementally additive with the presentation of tokens, we have

$$\begin{aligned}\overline{\mathbf{M}}_{2N} &= \overline{\mathbf{M}}_N + \overline{\delta\mathbf{M}}, \\ \overline{\mathbf{S}}_{2N} &= \overline{\mathbf{S}}_N + \overline{\delta\mathbf{S}},\end{aligned}\tag{21}$$

Let us now explicitly row-normalize the first line of the above equation. Note that the sum of each row of the matrix  $\mathbf{M}$  is proportional to the word frequency of the corresponding word (that is, the number of times that word occurred in the token list). From Eq. 15, we see that the word frequency of any word is the same (up to an order of  $N\epsilon$  difference, with a probability  $1-p$ ) for the first  $N$  tokens and the second  $N$  tokens. Hence while normalizing each row of  $\overline{\mathbf{M}}_{2N}$ , we will have equal contribution (up to an order of  $\epsilon$ ) from  $\overline{\mathbf{M}}_N$  and  $\overline{\delta\mathbf{M}}$ . Thus, with a probability greater than  $1-p$ , we have,

$$\mathbf{M}_{2N} = \frac{1}{2}\mathbf{M}_N + \frac{1}{2}\delta\mathbf{M} + O(\epsilon).\tag{22}$$

Here, we use the notation  $O(\epsilon)$  to denote a matrix with terms of order  $\epsilon$ . If the number of unique words in the language is  $D$ , and if  $\omega$  is the least value of all the  $\omega_\alpha$ 's (described in Eq. 15), it can be shown that  $s(O(\epsilon)) < D\epsilon/\omega$ . Moreover, from Eq. 17, we have  $s(\mathbf{M} - \mathbf{M}_N) < \epsilon$  and  $s(\mathbf{M} - \mathbf{M}_{2N}) < \epsilon$ . Using these inequalities and subtracting  $\mathbf{M}$  from both sides of the above equation, and applying the measure  $s()$ , we have

$$\begin{aligned}\mathbf{M} - \delta\mathbf{M} &= 2(\mathbf{M} - \mathbf{M}_{2N}) - (\mathbf{M} - \mathbf{M}_N) + 2O(\epsilon) \\ s(\mathbf{M} - \delta\mathbf{M}) &\leq 2s(\mathbf{M} - \mathbf{M}_{2N}) + s(\mathbf{M} - \mathbf{M}_N) + 2s(O(\epsilon)) \\ &< (2D/\omega + 3)\epsilon.\end{aligned}\tag{23}$$

The inequality in the second line of the above equation is a consequence of the fact that the measure  $s()$  obeys the triangle inequality. By a similar argument, it also turns out that  $s(\mathbf{S} - \delta\mathbf{S}) < (2D/\omega + 3)\epsilon$ . This implies that as  $\epsilon$  goes to zero with increasing  $N$  the operators  $\delta\mathbf{M}$  and  $\delta\mathbf{S}$  converge almost certainly to the steady state  $\mathbf{M}$  and  $\mathbf{S}$  respectively.

Now, to prove Claim 1, let us explicitly evaluate  $\delta\mathbf{S}$ .

$$\begin{aligned}\delta\mathbf{S} &= \left\| \sum_{i=N+1}^{2N} \mathbf{M}_i |\mathbf{t}_{i-1}\rangle \langle \mathbf{f}_i| \right\|_{\text{col}} \\ &= \left\| \sum_{i=N+1}^{2N} \mathbf{M} |\mathbf{t}_{i-1}\rangle \langle \mathbf{f}_i| \right\|_{\text{col}} + O_1(\epsilon) \\ &= \left\| \mathbf{M} \overline{\delta\mathbf{M}}' \right\|_{\text{col}} + O_1(\epsilon) \\ &= \left\| \mathbf{M} \delta\mathbf{M}' \right\|_{\text{col}} + O_1(\epsilon)\end{aligned}\tag{24}$$

The first equality comes from the definition (Eq. 20). The second equality holds almost certainly (with a probability greater than  $1-p$ ) because for all  $i > N$ ,  $\mathbf{M}_i$  differs from  $\mathbf{M}$

only by an order of  $\epsilon$ . The third equality again comes directly from our definition of  $\overline{\delta\mathbf{M}}$  (Eq. 20). The fourth equality is justified because the columns of  $\delta\mathbf{M}'$  and  $\overline{\delta\mathbf{M}'}$  are the same up to a normalization constant, and under overall column normalization, a difference in normalization constant is eliminated.

The notation  $O_1(\epsilon)$  again denotes a matrix with terms of order  $\epsilon$ , and is different from the matrix  $O(\epsilon)$  in Eq. 22. The individual entries of the matrix  $O_1(\epsilon)$  are both positive and negative and each of its columns sum up to zero. Over and beyond noting the fact that the matrix  $O_1(\epsilon)$  can be pushed arbitrarily close to zero by choosing a large value of  $N$ , we can evaluate an upper bound for it. Since this is inconsequential to the continuity of the proof, we will just state the result without details. Recall that each row of  $\mathbf{M}$  sums up to one by construction, but no such constraint is imposed on the sum over its columns. If  $d$  is the lower bound for all the column sums, it turns out that  $s(O_1(\epsilon)) < 2\epsilon D/d$ . Note that, for the ergodic generating functions considered here,  $d$  is necessarily positive. Thus we can write,

$$s(\delta\mathbf{S} - \|\mathbf{M}\delta\mathbf{M}'\|_{\text{col}}) < 2\epsilon D/d. \quad (25)$$

Along with  $s(\mathbf{M} - \delta\mathbf{M}) < (2D/\omega + 3)\epsilon$  and  $s(\mathbf{S} - \delta\mathbf{S}) < (2D/\omega + 3)\epsilon$ , the above equation immediately leads to Eq. 18 as we take  $\epsilon$  to zero, thus proving claim 1.

*Claim 2:  $\mathbf{M}$  (for any  $\phi$ ) can be calculated from the model with  $\phi = 0$*

If we define  $\mathbf{M}^\circ$  to be the steady state value of  $\mathbf{M}$  obtained with  $\phi = 0$ , then we have the following relationship.

$$\mathbf{M} = (1 - \phi)\mathbf{M}^\circ + \phi\mathbf{M}^\circ\mathbf{S}', \quad (26)$$

This claim is only true if the operator  $\mathbf{C}$  is an identity operator, as assumed here. Hence the vectors  $|\mathbf{f}_i\rangle$  and  $|\mathbf{c}_i\rangle$  can be used interchangeably.

The utility of Claim 2 is that it has tremendous practical implications when scaling the model up to real-world languages. During training of pTCM it is necessary to evaluate the  $|\mathbf{p}_i\rangle$  vector at each time step, which requires multiplying a vector by a matrix. This can be extremely costly. If there are  $D$  dimensions, the cost of this calculation goes like  $ND^2$ . In contrast, computation of  $\mathbf{M}^\circ$  does not require any of these calculations during learning. Claim 2 states that  $\mathbf{M}$  can be calculated with only a few matrix multiplications, which go like  $D^3$ . Using Claim 2 rather than explicitly calculating each  $|\mathbf{p}_i\rangle$  speeds up the calculation by a factor of approximately  $\frac{N}{D}$ , which can be a tremendous advantage.

To prove claim 2, we again imagine training the model on a token list of length  $2N$  as before. Instead of just keeping track of the context vector  $|\mathbf{t}\rangle$  at each step, now we also keep track of the context vector  $|\mathbf{t}^\circ\rangle$  that evolves with  $\phi = 0$ , and construct the corresponding operators  $\mathbf{M}_N^\circ$  and  $\delta\mathbf{M}^\circ$  which converge to the steady state value  $\mathbf{M}^\circ$  almost certainly. For the remainder of the proof, the term ‘‘almost certainly’’ implies that a given statement holds true with a probability greater than  $1 - p$ , and that  $p$  can be pushed arbitrarily close to zero by choosing a sufficiently large  $N$ . From the learning rules, we have

$$|\mathbf{t}_i\rangle = \rho|\mathbf{t}_{i-1}\rangle + (1 - \rho)[(1 - \phi)|\mathbf{f}_i\rangle + \phi\mathbf{S}_{i-1}|\mathbf{f}_i\rangle] \quad (27)$$

$$|\mathbf{t}_i^\circ\rangle = \rho|\mathbf{t}_{i-1}^\circ\rangle + (1 - \rho)|\mathbf{f}_i\rangle \quad (28)$$

A linear combination of the above two equations would yield,

$$\begin{aligned} |\mathbf{t}_i\rangle - (1 - \phi)|\mathbf{t}_i^o\rangle &= \rho [|\mathbf{t}_{i-1}\rangle + (1 - \phi)|\mathbf{t}_{i-1}^o\rangle] \\ &\quad + (1 - \rho) [\phi \mathbf{S}_{i-1} |\mathbf{f}_i\rangle] \end{aligned} \quad (29)$$

With the definition that

$$|\mathbf{k}_i\rangle \equiv |\mathbf{t}_i\rangle - (1 - \phi)|\mathbf{t}_i^o\rangle,$$

we can write

$$|\mathbf{k}_i\rangle = \rho |\mathbf{k}_{i-1}\rangle + (1 - \rho) \phi \mathbf{S}_{i-1} |\mathbf{f}_i\rangle \quad (30)$$

Let us represent the individual columns of the steady state  $\mathbf{S}$  as  $|\mathbf{s}_\beta\rangle$ . Hence as in Eq. 10, we have

$$\mathbf{S} = \sum_{\beta} |\mathbf{s}_\beta\rangle \langle \mathbf{f}_\beta|,$$

Since  $\mathbf{S}$  is the steady state value to which  $\mathbf{S}_i$ 's converge almost certainly, from Eq. 17, we see that for all  $i > N$ ,

$$\begin{aligned} |\mathbf{k}_i\rangle &= \rho |\mathbf{k}_{i-1}\rangle + (1 - \rho) \phi \mathbf{S} |\mathbf{f}_i\rangle + O(\epsilon) \\ &= \rho |\mathbf{k}_{i-1}\rangle + (1 - \rho) \phi \sum_{\beta} \langle \mathbf{f}_\beta | \mathbf{f}_i \rangle |\mathbf{s}_\beta\rangle + O(\epsilon) \end{aligned} \quad (31)$$

From the definitions of  $\overline{\delta \mathbf{M}}$  and  $\overline{\delta \mathbf{M}^o}$ , we have

$$\overline{\delta \mathbf{M}} = \sum_{i=N+1}^{2N} |\mathbf{f}_i\rangle \langle \mathbf{t}_{i-1}|, \quad \overline{\delta \mathbf{M}^o} = \sum_{i=N+1}^{2N} |\mathbf{f}_i\rangle \langle \mathbf{t}_{i-1}^o|. \quad (32)$$

$$\Rightarrow \overline{\delta \mathbf{M}} - (1 - \phi) \overline{\delta \mathbf{M}^o} = \sum_{i=1}^N |\mathbf{f}_i\rangle \langle \mathbf{k}_{i-1}| \quad (33)$$

Since  $\overline{\delta \mathbf{M}}$  and  $\overline{\delta \mathbf{M}^o}$  are constructed based on the same token list, the rows corresponding to the same word in either matrices will have the same sum. Hence row-normalization of the matrices on the LHS of the above equation will yield,

$$\delta \mathbf{M} - (1 - \phi) \delta \mathbf{M}^o = \phi \left\| \sum_{i=N+1}^{2N} |\mathbf{f}_i\rangle \langle \mathbf{k}_{i-1}| \right\|_{\text{row}} \quad (34)$$

Now, note that claim 2 (Eq. 26) is equivalent to the claim that  $\delta \mathbf{M} - (1 - \phi) \delta \mathbf{M}^o$  converges to  $\phi \delta \mathbf{M}^o \mathbf{S}'$  almost certainly. We can now explicitly expand

$$\phi \delta \mathbf{M}^o \mathbf{S}' = \phi \left\| \sum_{i=N+1}^{2N} \sum_{\beta} |\mathbf{f}_i\rangle \langle \mathbf{t}_{i-1}^o | \mathbf{f}_\beta \rangle \langle \mathbf{s}_\beta| \right\|_{\text{row}} \quad (35)$$

We need to show that for large  $N$ , Eq. 34 converges to Eq. 35 almost certainly. Note that these two expressions are row-normalized matrices. So let us explicitly compare a

specific row of the two matrices corresponding to the word  $\alpha$ . The  $\alpha^{th}$  row of Eq. 34 can be transposed to give the following column vector

$$\phi \left\| \sum_{i=N}^{2N-1} \langle \mathbf{f}_{i+1} | \mathbf{f}_\alpha \rangle | \mathbf{k}_i \rangle \right\| \quad (36)$$

and the  $\alpha^{th}$  row of Eq. 35 can be transposed to give the following column vector

$$\phi \left\| \sum_{i=N}^{2N-1} \langle \mathbf{f}_{i+1} | \mathbf{f}_\alpha \rangle \sum_{\beta} \langle \mathbf{f}_\beta | \mathbf{t}_i^o \rangle | \mathbf{s}_\beta \rangle \right\| \quad (37)$$

We shall now show that Eq. 36 converges to Eq. 37 in the large  $N$  limit. Let us expand them out. First note that

$$\begin{aligned} |\mathbf{t}_i^o\rangle &= \rho |\mathbf{t}_{i-1}^o\rangle + (1 - \rho) |\mathbf{f}_i\rangle \\ &= (1 - \rho) \sum_{j=0}^{i-1-N} \rho^j |\mathbf{f}_{i-j}\rangle + \rho^{i-N} |\mathbf{t}_N^o\rangle, \end{aligned} \quad (38)$$

where the second line follows from repeatedly expanding  $|\mathbf{t}_{i-1}^o\rangle$ . Then, we find

$$\begin{aligned} \sum_{\beta} \langle \mathbf{f}_\beta | \mathbf{t}_i^o \rangle | \mathbf{s}_\beta \rangle &= (1 - \rho) \sum_{j=0}^{i-1-N} \sum_{\beta} \rho^j \langle \mathbf{f}_\beta | \mathbf{f}_{i-j} \rangle | \mathbf{s}_\beta \rangle \\ &\quad + \rho^{i-N} \sum_{\beta} \langle \mathbf{f}_\beta | \mathbf{t}_N^o \rangle | \mathbf{s}_\beta \rangle, \end{aligned} \quad (39)$$

Similarly, we can repeatedly expand Eq. 31 to obtain

$$\begin{aligned} |\mathbf{k}_i\rangle &= (1 - \rho) \phi \sum_{\beta} \sum_{j=0}^{i-1-N} \rho^j \langle \mathbf{f}_\beta | \mathbf{f}_{i-j} \rangle | \mathbf{s}_\beta \rangle \\ &\quad + \rho^{i-N} |\mathbf{k}_N\rangle + (i - N) O(\epsilon) \end{aligned} \quad (40)$$

The factor  $(i - N)$  occurs in front of  $O(\epsilon)$  because Eq. 31 has to be iteratively used  $(i - N)$  times to completely expand  $|\mathbf{k}_i\rangle$ . The normalized vectors in Eqs. 36 and 37 can be constructed from Eqs. 40 and 39 respectively by summing over the  $i$ 's and subsequently normalizing them. Note that during normalization of both Eqs. 40 and 39, the second term in the RHS will pick up a factor of  $O(1/N)$  compared to the first term. This is because the first term has a summation over  $j$ . Thus it is basically the first term in the RHS of the Eq. 40 and Eq. 39 that would contribute to Eq. 36 and Eq. 37 respectively, and these two contributions are exactly the same. The contributions from the other terms in the RHS of Eqs. 40 and 39 to the Eqs. 36 and 37 respectively, is of the order  $O(\epsilon) + O(1/N)$ . For large  $N$ , both  $\epsilon$  and  $1/N$  go to zero, thus Eq. 36 and Eq. 37 converge on to each other almost certainly, hence proving Claim 2.

An important point to note is that Eqs. 18 and 26 yield the solution of  $\mathbf{M}$  and  $\mathbf{S}$  corresponding to pTCM only if the token list is long enough for the model to have reached

the steady state. If the token list is not long enough, then the expressions correspond to an approximate solution. We refer to the model calculated from  $\mathbf{M}^o$  using Claim 2 as  $\text{pTCM}_A$ . In practice, we evaluate  $\text{pTCM}_A$  by constructing an estimate of  $\mathbf{S}$  from Claim 1 using  $\mathbf{M}^o$ , then using this  $\mathbf{S}$  to make an estimate of  $\mathbf{M}$  using Claim 2; this process is repeated iteratively until the solution converges.

*Claim 3: pTCM with  $\rho = 0$  and  $\phi = 0$  perfectly reconstructs bi-gram languages*

Consider a simple language that can be completely specified by a bi-gram transition matrix among the  $D$  words in the language. We refer to this matrix describing the probability of transitioning from one word to another as  $\mathbf{T}$ . At steady-state, with  $\rho = 0$  and  $\phi = 0$ , the transition matrix derived from pTCM is equal to the transition matrix that describes the language:

$$\mathbf{T}_{\text{model}} = \mathbf{T} \quad (41)$$

While we cannot prove that pTCM develops the ability to perfectly reconstruct all possible generating functions that define a language (nor indeed is it capable of doing so), we can illustrate its ability to reconstruct a restricted class of generating functions. In particular, bi-gram languages for which the probability of the next word depends only on the identity of the previous word. Such a language is characterized by a Markov chain of order one and can be fully specified by a transition matrix  $\mathbf{T}$ , such that if word  $\alpha$  is presented at time step  $i$  then  $\mathbf{T}|\mathbf{f}_\alpha\rangle$  describes the probability distribution of words that will be presented at time step  $i + 1$ : i.e.,  $\langle \mathbf{f}_\beta | \mathbf{T} | \mathbf{f}_\alpha \rangle$  is the conditional probability of word  $\beta$  being presented at time step  $i + 1$  given that word  $\alpha$  was presented at time step  $i$ . We start by specifying more thoroughly some useful properties of the transition matrix  $\mathbf{T}$ . Then we describe how the transition matrix ( $\mathbf{T}_{\text{model}}$ ) can be extracted from pTCM, and finally compare the two.

*Specifying  $\mathbf{T}$  for a bi-gram language.* Consider a simple example with just three words ‘A’, ‘B’, and ‘C’. Let us denote the probability of transition from ‘A’ to ‘B’ by  $P_{AB}$ , the probability of transition from ‘A’ to ‘C’ by  $P_{AC}$ , and so on. We can arrange these probabilities in the form of a matrix such that each column of the matrix corresponds to transition probabilities from a given word. As a consequence, the sum of each column of this matrix should be one (a positive matrix with each column summing up to one is known as a column stochastic matrix). This is precisely the transition matrix.

$$\mathbf{T} = \begin{bmatrix} P_{AA} & P_{BA} & P_{CA} \\ P_{AB} & P_{BB} & P_{CB} \\ P_{AC} & P_{BC} & P_{CC} \end{bmatrix} \quad (42)$$

A straightforward observation is that any transition matrix has at least one eigenvalue equal to one. To observe this, first note that any matrix has the same spectrum of eigenvalues as its transpose. If  $\mathbf{T}$  is a transition matrix, then each row of its transpose  $\mathbf{T}'$  sums up to one. Clearly, a uniform vector (whose every component is one), is an eigenvector of  $\mathbf{T}'$  with an eigenvalue equal to 1. Hence,  $\mathbf{T}$  should also have an eigenvalue equal to one.

A slightly more subtle observation is that the eigenvector of  $\mathbf{T}$  corresponding to eigenvalue of 1 corresponds to the relative word frequency vector  $|\mathbf{w}\rangle$  at the steady state.

Here, we define the word frequency vector as a vector with each component being the relative word frequency of the corresponding word, namely  $(\mathbf{w}_A, \mathbf{w}_B, \mathbf{w}_C)$ . Note that in the steady state limit (when the list generated by the transition matrix  $\mathbf{T}$  is long enough), we can use the following equations to obtain the relative word frequencies.

$$\begin{aligned} P_{AA}\mathbf{w}_A + P_{BA}\mathbf{w}_B + P_{CA}\mathbf{w}_C &= \mathbf{w}_A \\ P_{AB}\mathbf{w}_A + P_{BB}\mathbf{w}_B + P_{CB}\mathbf{w}_C &= \mathbf{w}_B \\ P_{AC}\mathbf{w}_A + P_{BC}\mathbf{w}_B + P_{CC}\mathbf{w}_C &= \mathbf{w}_C \end{aligned}$$

The above equations can be concisely expressed as  $\mathbf{T}|\mathbf{w}\rangle = |\mathbf{w}\rangle$ , which shows that the word frequency vector is indeed the eigenvector of  $\mathbf{T}$  with eigenvalue 1.

One can imagine pathological situations where  $\mathbf{T}$  is reducible, meaning the nonzero entries of  $\mathbf{T}$  are separable subspaces. In such a case, the indices of  $\mathbf{T}$  can be reordered such that it can be written as a block diagonal matrix. If we start out with a word in one of the blocks, the generating function can never generate a word from outside that block. This would violate our condition of ergodicity that we have placed on the languages that we consider here. However, for each block, there would exist an associated word frequency vector. In this case, the eigenvalue=1 is degenerate, with degeneracy being equal to the number of blocks. We shall just focus on irreducible  $\mathbf{T}$  here, because in principle, a reducible  $\mathbf{T}$  can be viewed as a direct sum of transition matrices of disjoint languages.

*Specifying  $\mathbf{T}_{\text{model}}$  for  $pTCM$ .* We conjecture that the prediction vector  $|\mathbf{p}\rangle$  is an estimate of the conditional probability distribution of the subsequent word predicted by a particular context vector. More specifically,  $w_\alpha\langle\mathbf{f}_\alpha|\mathbf{p}\rangle$  gives an estimate of the probability (up to a normalization factor) that the subsequent word will be  $\alpha$ , where  $w_\alpha$  is the relative word frequency and the prediction vector  $|\mathbf{p}\rangle$  is calculated from the knowledge of the prevalent context vector  $|\mathbf{t}\rangle$ .

At steady state, the preceding context can be taken to be the average context in which the current word would occur. Suppose the current word is  $\beta$ . Then the context used to cue for the next word is on average

$$|\mathbf{t}\rangle = \rho|\underline{\mathbf{t}}_\beta\rangle + (1 - \rho)\mathbf{C}[\phi|\mathbf{s}_\beta\rangle + (1 - \phi)|\mathbf{f}_\beta\rangle]. \quad (43)$$

We can write explicit expressions for each of these terms using the operators that have been defined. From the definition of  $\mathbf{M}$  (Eq. 7), we have  $|\underline{\mathbf{t}}_\beta\rangle = \mathbf{M}'|\mathbf{f}_\beta\rangle$ . From the definition of  $\mathbf{S}$  (Eq. 4), we have  $|\mathbf{s}_\beta\rangle = \mathbf{S}|\mathbf{f}_\beta\rangle$ . Finally, from the definition of  $\mathbf{C}$  (Eq. 1), we have  $|\mathbf{c}_\beta\rangle = \mathbf{C}|\mathbf{f}_\beta\rangle$ . Putting everything together, we can calculate an estimate of the probability distribution function (up to a normalization factor) for the word following  $\beta$  to be  $\mathbf{W}|\mathbf{p}\rangle = \mathbf{W}\mathbf{M}|\mathbf{t}\rangle$ , which can be expanded out as

$$\mathbf{W}\mathbf{M}\left[\rho\mathbf{M}' + (1 - \rho)\left(\phi\mathbf{C}\mathbf{S} + (1 - \phi)\mathbf{C}\right)\right]|\mathbf{f}_\beta\rangle \quad (44)$$

By normalizing this vector, we get the subsequent word probability distribution for each word  $\beta$ . Arranging these vectors as columns of a matrix we have an estimate of the bi-gram transition probability derived from the model

$$\mathbf{T}_{\text{model}} = \left\| \left\| \mathbf{W}\mathbf{M}\left[\rho\mathbf{M}' + (1 - \rho)\left(\phi\mathbf{C}\mathbf{S} + (1 - \phi)\mathbf{C}\right)\right] \right\|_{\text{col}} \right\| \quad (45)$$

*Proof of Claim 3.* Consider the case where  $\rho = 0$ ,  $\phi = 0$  and  $\mathbf{C}$  is an identity operator. We have  $\mathbf{T}_{\text{model}} = \|\mathbf{WM}\|_{\text{col}}$ . We will now show that, at the steady state,  $\mathbf{T}_{\text{model}} = \mathbf{T}$ . Firstly, note that after the presentation of the item  $\beta$ , the context vector becomes  $|\mathbf{f}_\beta\rangle$ . From the definition of the transition matrix we see that, given an item  $\beta$  has occurred, the probability of the subsequent item to be  $\alpha$  is given by  $P_{\beta\alpha} = \langle \mathbf{f}_\alpha | \mathbf{T} | \mathbf{f}_\beta \rangle$ . Secondly, note that the average context in which the item  $\alpha$  occurs in the token list is  $|\underline{\mathbf{t}}_\alpha\rangle$ . If the word vectors  $|\mathbf{f}_\alpha\rangle$  are chosen as bases, then the average context vectors  $|\underline{\mathbf{t}}_\alpha\rangle$  form the rows of  $\mathbf{M}$ .

It is now straightforward to calculate  $|\underline{\mathbf{t}}_\alpha\rangle$ . Given an item  $\alpha$ , the probability that it occurs immediately following item  $\beta$  can be denoted by  $P(\beta|\alpha)$ . Then,

$$|\underline{\mathbf{t}}_\alpha\rangle = \sum_{\beta} P(\beta|\alpha) |\mathbf{f}_\beta\rangle \quad (46)$$

Note that  $P(\beta|\alpha)$  is different from  $P_{\beta\alpha}$ , which is the conditional probability of the next word being  $\alpha$ , given the current is  $\beta$ . We can now use Bayes' theorem to write

$$P_{\beta\alpha} P(\beta) = P(\beta|\alpha) P(\alpha) \quad (47)$$

where  $P(\alpha)$  and  $P(\beta)$  are the marginal probabilities of occurrence of items  $\alpha$  and  $\beta$  respectively, which are exactly the relative word frequencies  $w_\alpha$  and  $w_\beta$ . Hence, under the assumptions we have made here,

$$|\underline{\mathbf{t}}_\alpha\rangle = \sum_{\beta} P_{\beta\alpha} \frac{w_\beta}{w_\alpha} |\mathbf{f}_\beta\rangle \quad (48)$$

Let us now write out the column vectors of  $\mathbf{T}_{\text{model}}$ .

$$\begin{aligned} \mathbf{T}_{\text{model}} |\mathbf{f}_\beta\rangle &= \left\| \|\mathbf{WM}\|_{\text{col}} |\mathbf{f}_\beta\rangle \right\| \\ &= \left\| \sum_{\alpha} w_\alpha |\mathbf{f}_\alpha\rangle \langle \underline{\mathbf{t}}_\alpha | \mathbf{f}_\beta \rangle \right\| \\ &= \left\| \sum_{\alpha} P_{\beta\alpha} w_\beta |\mathbf{f}_\alpha\rangle \right\|. \end{aligned} \quad (49)$$

Since the index  $\beta$  is a constant on the columns and  $P_{\beta\alpha}$  is normalized such that  $\sum_{\alpha} P_{\beta\alpha} = 1$ , column normalization of the above equation will automatically get rid of the factor  $w_\beta$ . Thus

$$\langle \mathbf{f}_\alpha | \mathbf{T}_{\text{model}} | \mathbf{f}_\beta \rangle = P_{\beta\alpha} = \langle \mathbf{f}_\alpha | \mathbf{T} | \mathbf{f}_\beta \rangle \quad (50)$$

Thus we have shown that  $\mathbf{T}_{\text{model}} = \mathbf{T}$  when  $\rho = 0$  and  $\phi = 0$ . Note that, we have used the steady state  $\mathbf{M}$  and  $\mathbf{S}$  to construct the  $\mathbf{T}_{\text{model}}$ . If we had used the  $\mathbf{M}$  and  $\mathbf{S}$  created from a finite token list to construct  $\mathbf{T}_{\text{model}}$ , then we wouldn't have perfect equality as in Eq. 50. Instead we would have, for a given  $\epsilon > 0$  and  $p > 0$ , for a sufficiently long token list,

$$\mathbf{Prob}\left(s(\mathbf{T} - \mathbf{T}_{\text{model}}) < \epsilon\right) > 1 - p \quad (51)$$

This derivation also supports our conjecture that the prediction vector  $|\mathbf{p}\rangle$  is an estimate of the conditional probability distribution of the subsequent word predicted by a particular

context vector. For this class of generating function and these parameters, the estimate is precisely correct at steady state.

In the general case when  $\rho$  and  $\phi$  are nonzero,  $\mathbf{T}_{\text{model}}$  and  $\mathbf{T}$  are not equal. However, this should not be interpreted as meaning that these parameters are useless.  $\rho$  provides a measure of flexibility to capture generating functions that are not simply described by bi-gram models. One can think of the model with nonzero  $\rho$  as roughly analogous to a weighted N-gram model (e.g. Bengio, Ducharme, Vincent, & Jauvin, 2003). The parameter  $\phi$  provides the ability to rapidly discover equivalence classes among words. We will characterize this property with simulations in the following section.

### Simulations on a toy language

One of the major challenges in learning a language is overcoming the curse of dimensionality. To illustrate one aspect of this problem, let us imagine that we have a bi-gram language with  $D$  words. We want to estimate the value of  $\langle \mathbf{f}_\alpha | \mathbf{T} | \mathbf{f}_\beta \rangle$ . The number of tokens we need to present to estimate this value from the data to a given level of reliability goes up with  $D^2$ . The problem is worse if the generating function is not a bi-gram model but an N-gram model. Nonzero values of the parameters  $\rho$  and  $\phi$  cause the steady state behavior of the model to deviate from optimality for a bi-gram model. However, non-zero  $\phi$  works to mitigate against the curse of dimensionality by allowing generalization among words with similar roles in the language.

The ability to rapidly generalize among synonymous words is of course only an advantage for languages that contain synonyms. Here we illustrate the utility of non-zero  $\phi$  by explicitly simulating the model on training sequences generated from a small toy language with classes of synonymous words. We also compare pTCM<sub>A</sub> to the results of explicit simulation of this toy language and the form of convergence to the steady state.

#### Methods

For these simulations,  $\mathbf{T}$  is a  $9 \times 9$  matrix which is composed of three categories A, B and C, with three words each.  $\mathbf{T}$  can be written as  $\mathbf{T}^{ABC} \otimes \mathbf{X}$ , where  $\mathbf{T}^{ABC}$  contains the probabilities of transition between the three word categories and  $\mathbf{X}$  is a uniform  $3 \times 3$  matrix. In the simulations we used

$$\mathbf{T}^{ABC} = \begin{bmatrix} 0.1 & 0.4 & 0.1 \\ 0.7 & 0.0 & 0.8 \\ 0.2 & 0.6 & 0.1 \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \\ 1/3 & 1/3 & 1/3 \end{bmatrix} \quad (52)$$

Note that because  $\mathbf{X}$  is uniform, words within a category are perfectly interchangeable.

We implemented the model in the R programming language. We chose each of the nine  $|\mathbf{f}\rangle$  vectors to be a basis function for our space and  $\mathbf{C}$  to be the nine-dimensional identity matrix. As a consequence,  $\mathbf{M}$ ,  $\overline{\mathbf{M}}$ ,  $\mathbf{S}$  and  $\overline{\mathbf{S}}$  are all  $9 \times 9$  matrices. Each point in the simulations reflects an average over 2,000 randomly chosen token lists. The variability in the mean value is essentially zero for all simulated points. We initialized  $|\mathbf{t}_0\rangle$  and the operators  $\overline{\mathbf{M}}$  and  $\overline{\mathbf{S}}$  to be all zeros at the beginning of each token list. We also assumed that if  $|\mathbf{s}_i\rangle = 0$ , then  $|\mathbf{t}_i^{IN}\rangle = |\mathbf{c}_i\rangle$ . Note that this initialization leads to nonsensical predictions at the first several presentations. However, the form of the initialization is rapidly lost over

the scale we report here. Essentially indistinguishable results were observed if the  $|\mathbf{t}_0\rangle$  and the operators were initialized to be uniform or small random values.

The optimal value of  $\rho$  for a bi-gram generating function such as the one considered here is zero. That is, the model’s capability of discovering equivalence classes and its accuracy of prediction is maximal when  $\rho$  is zero. In this paper, we only report the results of the simulations with  $\rho = 0$ .

### Results

We explicitly demonstrate three aspects of pTCM when trained on the bi-gram language (eq. 52). First we show that for sufficiently long training sequences, pTCM converges to a steady state. Then we show that nonzero values of  $\phi$  give a significant advantage for short training sequences in discovering equivalence classes among words and in improving the quality of prediction.

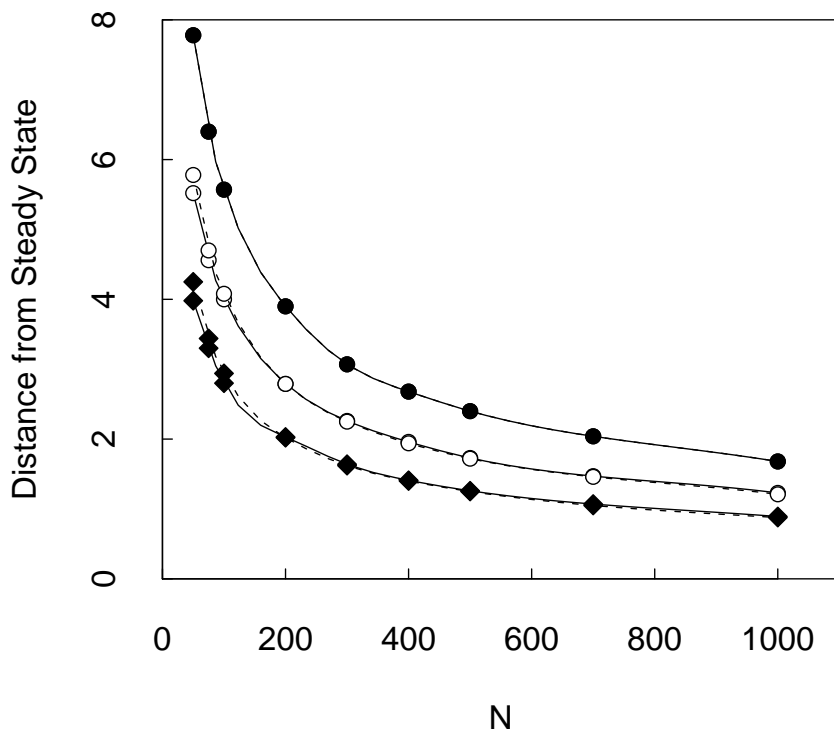
*Convergence to steady state.* As discussed previously, if the number of tokens  $N$  is large enough, the operators  $\mathbf{M}_N$  and  $\mathbf{S}_N$  will stabilize to their steady state values  $\mathbf{M}$  and  $\mathbf{S}$  respectively. Also recall that the steady state  $\mathbf{S}$  can be expressed as a function of the steady state  $\mathbf{M}$  (Claim 1), so evaluating the convergence of  $\mathbf{M}_N$  is sufficient to establish the convergence of  $\mathbf{S}_N$ . An important point to note is that the generating function  $\mathbf{T}$  does not uniquely fix the steady state values  $\mathbf{M}$  and  $\mathbf{S}$ ; they depend strongly on  $\rho$  and  $\phi$ . Moreover, it turns out that the rate of convergence to the steady state is faster for higher values of  $\phi$  and  $\rho$ . To measure the distance from the steady state, we use the measure  $s()$  (defined in Eq. 16). To give a sense of the scale of this measure, we note that since the matrices  $\mathbf{T}$ ,  $\mathbf{M}$  and  $\mathbf{S}$  are all  $9 \times 9$  (row/column) stochastic matrices, we have  $s(\mathbf{T}) = s(\mathbf{M}) = s(\mathbf{S}) = 9$ .

Figure 2 shows the distance from the steady state value of  $\mathbf{M}$  for pTCM and pTCM<sub>A</sub> as a function of the number of tokens in the tokens list for three values of  $\phi$  with  $\rho$  fixed at zero. There are two features of the model that can be observed from Figure 2. First, the convergence to steady state is faster for greater values of  $\phi$ . Second, there are at most modest differences in convergence between pTCM and pTCM<sub>A</sub>.<sup>3</sup> As  $N$  grows without bound, the curves end up at a distance of zero from the steady state value of  $\mathbf{M}$ . For  $\phi = 0$ , the distance to steady state decreases to .2 after a token list of 80,000 tokens. For  $\phi = 0.5$ , the number of tokens needed to approach within the same distance is 15,000.

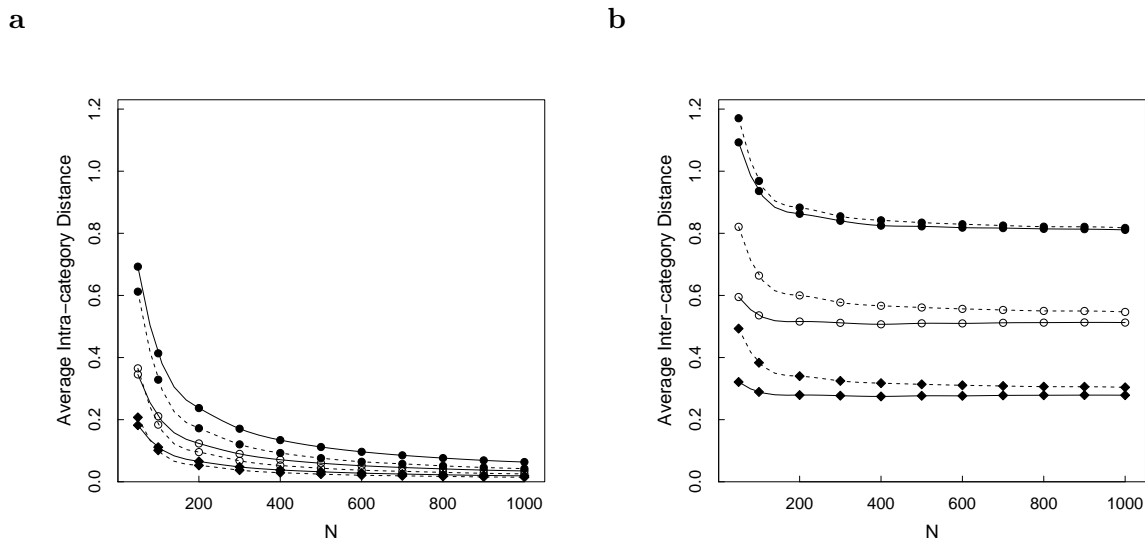
*Discovering equivalence classes.* There are two distinct steps involved in categorizing words into equivalence classes. The first is to recognize that the words within a category are sufficiently similar to each other, and the second is to recognize that the categories are sufficiently distinct from each other. Here we show that the equivalence classes among words can be directly read out from the semantic representation matrix  $\mathbf{S}$ .

Recall that the semantic representation of any word is given by the corresponding column of the  $\mathbf{S}$  matrix. Since our language is made of three categories of three words each, we expect the  $\mathbf{S}$  matrix to reflect this category structure. As expected, we found that the semantic representations of any two words  $\alpha$  and  $\beta$ , are similar to each other if they are from the same category, and less similar if they are from different categories. More

<sup>3</sup>For small values of  $N$ , pTCM converges more quickly. For large values of  $N$ , pTCM<sub>A</sub> converged slightly more quickly.

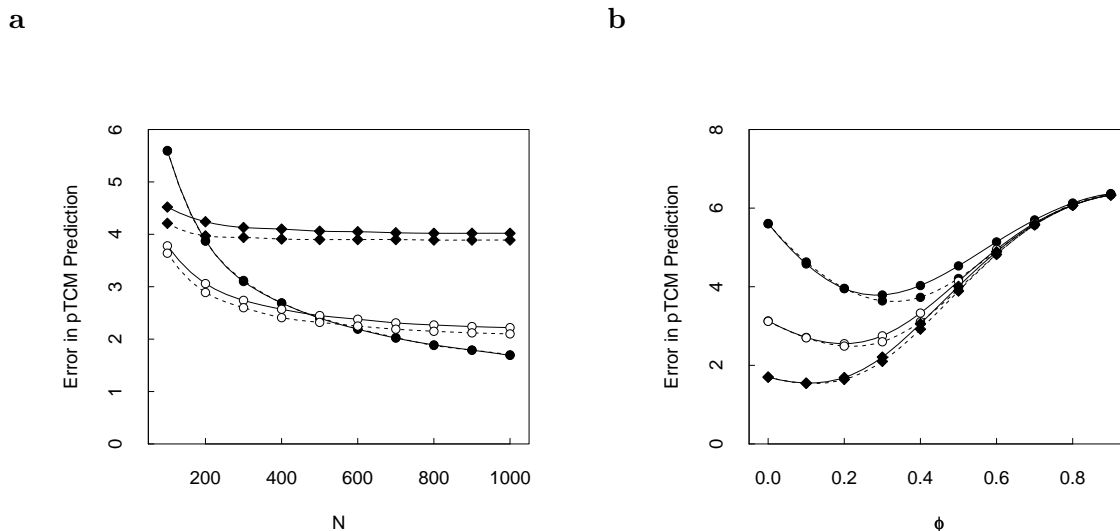


*Figure 2.* The discrepancy between the matrix  $\mathbf{M}_N$  and its steady state value  $\mathbf{M}$  defined as  $s(\mathbf{M}-\mathbf{M}_N)$  is plotted as a function of  $N$ , the number of tokens in the training set. Each point reflects the average over many training sets chosen independently from the generating function. Filled circles, open circles and filled diamonds correspond to values of  $\phi$  of 0, 0.3, and 0.5, respectively.  $\rho = 0$  in all the cases. Points connected by solid lines show the average of simulations using pTCM. Points connected by dashed lines show the average of simulations using pTCM<sub>A</sub>.



*Figure 3.* The distance on the y-axis gives  $s(|\mathbf{s}_\alpha\rangle - |\mathbf{s}_\beta\rangle)$  averaged over different words of the language. Each point reflects the average over many training sets chosen independently from the generating function. Points connected by solid lines are from simulations of pTCM and points connected by dashed lines are from simulations using pTCM<sub>A</sub>. Curves with different types of points correspond to different values of  $\phi$ : filled circles,  $\phi = 0$ ; open circles,  $\phi = 0.3$ ; filled diamonds,  $\phi = 0.5$ . **a.** The average intra-category distance or the average distance between words of different categories is calculated by averaging  $s(|\mathbf{s}_\alpha\rangle - |\mathbf{s}_\beta\rangle)$  over all possible combinations of  $\alpha$  and  $\beta$  when they both belong to the same category. **b.** The average inter-category distance is calculated by averaging over all combinations of  $\alpha$  and  $\beta$  that belong to distinct categories.

specifically, as we increase the length of the token list, the distance between the semantic representations of  $\alpha$  and  $\beta$ , namely  $s(|\mathbf{s}_\alpha\rangle - |\mathbf{s}_\beta\rangle)$  converges to zero if  $\alpha$  and  $\beta$  are in the same category. If not, this distance converges to a nonzero number. In Figure 3a, we plot the average intra-category distance, which we define as  $s(|\mathbf{s}_\alpha\rangle - |\mathbf{s}_\beta\rangle)$  averaged over all possible combinations of  $\alpha$  and  $\beta$  such that they both belong to the same category. We see that the average intra-category distance converges to zero for all values of  $\phi$  as  $N$  grows large. We also find that for higher values of  $\phi$ , the convergence is faster. Hence, for short training sequences, the words within a category can be grouped together much more reliably with higher  $\phi$  values. In figure 3b, we plot the average inter-category distance, which is defined as  $s(|\mathbf{s}_\alpha\rangle - |\mathbf{s}_\beta\rangle)$  averaged over all possible combinations of  $\alpha$  and  $\beta$  such that they belong to two distinct categories. We see that the average inter-category distance converges to a nonzero value. Moreover, we find that this value is lower for higher values of  $\phi$ . This implies that for higher values of  $\phi$ , even though the intra-category convergence is faster, the distinguishability among the categories is weaker. So if our goal were to discriminate categories from a generating function according a specific metric, we would choose a  $\phi$  such that the intra-category distance is sufficiently small, while at the same time the inter-category distance is sufficiently big to discriminate between the distinct categories. We also note that pTCM<sub>A</sub> pushes down the intra-category distance (from figure 3a), while at



*Figure 4.* The measure of the error in prediction on the y-axis is  $s(\mathbf{T}_{\text{model}} - \mathbf{T})$ . Each point reflects the average over many training sets chosen independently from the generating function. Points connected by solid lines are from simulations of pTCM, and points connected by dashed lines are from simulations using pTCM<sub>A</sub>. **a.** For different values of  $\phi$ , the error in prediction generated by pTCM is plotted with respect to the number of tokens in the token list  $N$ . Filled circles,  $\phi = 0$ ; open circles,  $\phi = 0.3$ ; filled diamonds,  $\phi = 0.5$ . **b.** For different values of  $N$ , the error in prediction is shown as a function of  $\phi$ . For small  $N$ , there is an optimal nonzero value of  $\phi$ . Filled circles,  $N = 100$ ; open circles,  $N = 300$ ; filled diamonds  $N = 1000$ .

the same time pushes up the inter-category distance (from figure 3b) relative to pTCM. This shows that pTCM<sub>A</sub> performs better than pTCM in reliably discovering the equivalence classes and discriminating amongst the distinct categories.

A point worth mentioning is that the inter-category distance to which the different curves in Figure 3b converge, is very much dependent on the generating function. The generating function we have chosen for these simulations (Eq. 52) is such that the inter-category transitions is non-deterministic, that is the columns of  $\mathbf{T}^{ABC}$  are not orthogonal. As a consequence, for higher values of  $\phi$ , the generalization process incorporated in pTCM makes different categories similar to each other. If we had chosen a deterministic inter-category transition matrix  $\mathbf{T}^{ABC}$  (unlike Eq. 52), then the categories will be maintained completely distinct from each other in  $\mathbf{S}$  matrix, and the inter-category distance would be independent of  $\phi$ . In such a situation, the highest possible value of  $\phi$  is the most optimal for discovering the equivalence classes.

*Accuracy of prediction.* Figure 2 established that nonzero  $\phi$  results in a faster convergence to steady state. However, the steady state limit to which the model with nonzero  $\phi$  converges to is non-optimal in the sense that it does not yield a correct prediction, that is  $\mathbf{T}_{\text{model}} \neq \mathbf{T}$ . However, far from steady state, it turns out that nonzero  $\phi$  improves the predictiveness of the model (at least for the type of generating function used here) by

discovering and exploiting the equivalence between the words that are part of the same category.

The scalar measure  $s(\mathbf{T}_{\text{model}} - \mathbf{T})$  can be used to measure the difference between  $\mathbf{T}$  and  $\mathbf{T}_{\text{model}}$ . This can be considered as an “error in prediction” by the model.<sup>4</sup> Figure 4a shows the error induced by pTCM and pTCM<sub>A</sub> as a function of  $N$  for several values of  $\phi$ . Figure 4a reveals several interesting properties. As expected, for large values of  $N$ ,  $\phi = 0$  (filled circles) provides the best prediction. However, for smaller values of  $N$ , the model with  $\phi = 0.3$  outperforms the model with  $\phi = 0$ . Although it performs considerably worse asymptotically, at very small values of  $N$ , the model with  $\phi = 0.5$  (filled diamonds) even outperforms the model with  $\phi = 0$ . For the two curves in Figure 4a with nonzero  $\phi$ , pTCM<sub>A</sub> yielded a better prediction than pTCM for all values of  $N$ .

Figure 4b illustrates the advantage of nonzero  $\phi$  for small  $N$  more clearly. For several values of  $N$ , the error is plotted as a function of  $\phi$ . Note that for small values of  $N$  (see especially  $N = 100$ ), the error initially decreases as  $\phi$  increases from zero, reaching an optimal value of  $\phi$  for which the error is minimal, beyond which increasing  $\phi$  increases the error. For this specific transition matrix  $\mathbf{T}$  and  $N = 100$ , the optimal value of  $\phi$ , the point at which the curve reaches a minimum, appears to be close to 0.3. For  $N = 300$ , the optimal value of  $\phi$  is a bit smaller. In the limit as  $N$  increases without bound, the curve would be monotonically increasing such that the optimal value of  $\phi$  was zero. For each value of  $\phi > 0$ , pTCM<sub>A</sub>, illustrated by the dashed lines in Figure 4b, performs slightly better than pTCM. In some cases this advantage is substantial. Also note that the optimal value of  $\phi$  for pTCM<sub>A</sub> is not necessarily the optimal value of  $\phi$  for pTCM (see especially the filled circles, for  $N = 100$ ).

## General Discussion

In this paper we presented a formal sequential learning model for extracting the generating function from sequentially-presented words. This model utilizes a representation of temporal context taken from the temporal context model (TCM, Howard & Kahana, 2002; Howard et al., 2005; Sederberg et al., 2008). This extension, which we refer to as the predictive temporal context model (pTCM) generates a prediction about the next word that will be presented at any time on the basis of the prior history. The semantic representation of a word is the superposition of all the prediction vectors generated just before the word was presented.

We prove several useful results in this paper. Among these, we show that if presented with a sequence of words generated by a bi-gram transition matrix, the model yields a perfect prediction if  $\phi$ , which determines the degree to which the semantic representation is included in context, is set to zero. Conversely, if  $\phi$  is nonzero, the model yields an imperfect prediction in the limit as the number of tokens in the training list goes to infinity. Under these circumstances, the semantic representation generated by the operator  $\mathbf{S}$  is appropriate, but its inclusion in the context cue leads to incorrect predictions. There are some generating functions for which this result does not hold. For instance, if we had a

<sup>4</sup>There are more complex measures of error in prediction that can be adopted. For instance,  $\mathbf{T}_{\text{model}}$  and  $\mathbf{T}$  could be compared with a metric that calculates the expected Kullback-Leibler divergence. Nonetheless,  $s()$  should suffice for our purposes.

generating function in which  $\mathbf{T}^{ABC}$  was composed of deterministic transitions, i.e., if the columns of  $\mathbf{T}^{ABC}$  were orthogonal to one another, then the model with nonzero  $\phi$  yields the same quality of prediction at steady state as does the model with  $\phi = 0$ .

Far from steady state, when  $N$  is relatively small, non-zero  $\phi$  yielded a superior fit to the generating function. This is because the generating function was chosen to have redundancy that can be exploited by non-zero  $\phi$ . In this example, all words that were part of the same category were perfectly interchangeable and thus synonymous. Non-zero  $\phi$  is advantageous for small  $N$  because the benefit of “filling in” the category structure outweighs the costs of distortions to the nontrivial across-category transition structure, at least in the case of our generating function.

In addition to pTCM, we also introduced pTCM<sub>A</sub>, a variant of the model. If the training sequence is long enough for pTCM to reach a steady state, then pTCM<sub>A</sub> exactly matches the final state of pTCM. If the training sequence is not long enough for pTCM to reach a steady state, the solution reached by pTCM with nonzero  $\phi$  depends on the order of presentation of the token list. That is, a particular sequence learned very early in the token list, can have a different effect on the final state of the model from the same sequence learned at a later point in the token list. This is so because the  $|\mathbf{p}\rangle$  vector that is generated at a particular stage of learning depends on the development of  $\mathbf{M}$  up to that point. In contrast, in pTCM<sub>A</sub>, the information about when a particular sequence is learned has no effect on the final state of pTCM<sub>A</sub>. In other words, far from steady state, unlike pTCM, pTCM<sub>A</sub> is history-independent. We can think of pTCM<sub>A</sub> as the steady state value that pTCM would attain if the token list were a precise estimate of the transitions that take place in the language. With this in mind, one can imagine pTCM<sub>A</sub> as the result of pTCM if it were repeatedly trained multiple times with the same token list. As mentioned previously, pTCM<sub>A</sub> has dramatic practical benefits over pTCM for large languages.

### *Extending the model*

We have shown in this paper that pTCM can capture the bi-gram generating functions quite well. But the model is sufficiently simple that it doesn’t have enough structure to capture various aspects of more complex languages. Here we suggest several routes for extending the model to enable it to extract the generating function of more complex languages.

*Nonzero  $\rho$ .* In pTCM, the prediction at any time is determined by the history of the context states and the words presented in association with them. The parameter  $\rho$  controls the rate of contextual drift, and thus how the context retrieved by previous tokens contribute to the current context. For the bi-gram languages considered here, the optimal value of  $\rho$  is zero—information about tokens before the most recently-presented one contribute no additional information about the subsequent word. When  $\rho$  is zero, only the most recently presented token gets encoded into the context vector. When  $\rho$  is nonzero, the tokens prior to the most recently presented one also contribute to the context vector, weighted by the recency of the token. Hence, in some sense, nonzero  $\rho$  helps to maintain word order information from prior tokens. To efficiently learn complex languages, storing word order information is very essential. Unlike pTCM, wherein the word order information is built into the context vector, models like BEAGLE (Jones & Mewhort, 2007) store the word order

information separately from the context vector. BEAGLE uses convolution vector products to capture the word order information. Sahlgren, Holst, and Kanerva (2008) showed that a computationally simpler strategy of random permutation of word vectors is sufficient to do the same. pTCM’s strategy of capturing the word order information is much simpler than these approaches.

Although the temporal context vector discards detailed information about word order, this may not be a crippling disadvantage to the extent that a language has stronger near neighbor correlations than distant correlations among words. On the other hand a language with stronger distant correlations than near neighbor correlations, will present a serious challenge to pTCM. In order to capture such non-adjacent dependencies in generating functions, varying the value of  $\rho$  from word to word might be an effective strategy. A more complicated strategy is to simultaneously consider multiple sets of pTCM with different values of  $\rho$ , where each set would carry a temporal context vector with a different scale of word order information. In principle, the information on non-adjacent dependencies should reside in a combination of across-scale word order information, hence developing heuristics on reconstructing them would be of interest for future work.

*Quick and optimal learning with variable  $\phi$ .* There is some conflict in the role of  $\phi$  for the bi-gram languages considered here. With  $\phi = 0$ , learning is guaranteed to be asymptotically perfect, but will take a very long time. On the other hand, non-zero  $\phi$  results in faster learning if there are redundancies in the language, but results in asymptotically imperfect reconstruction of the generating function, if the inter-category transitions are non-deterministic. This suggests a learning strategy in which  $\phi$  starts out large and gradually decreases. Unfortunately, the optimal  $\phi$  for a particular  $N$  would most likely depend on the precise choice of generating function, which is not known in advance. Developing useful heuristics for this dynamic learning rule is a major challenge for future work.

#### *Connection to episodic memory*

The present framework develops a representation of semantic structure from sequentially-presented materials. This uses a representation of temporal context originally developed for its use in describing episodic recall phenomena. The semantic representation utilized here bears some resemblance to the recovery of temporal context. If the item presented at time step  $i$  is repeated, recovery of temporal context has been described as the successful recovery of  $|\mathbf{t}_{i-1}\rangle$ . Here, the semantic representation is built up from the superposition of  $|\mathbf{p}_i\rangle$ , the prediction vector at time step  $i$ , which is just  $\mathbf{M}|\mathbf{t}_{i-1}\rangle$ . The constructed semantic representation differs from the recovery of temporal context in the sense that we envision the buildup of the  $\mathbf{S}$  operator to take place over a long period of time, whereas recovery of temporal context for episodic memory need not reflect the superposition of a great many states but rather reflect a more restricted range of events. It is our view that  $\mathbf{S}$  and  $\mathbf{H}$  reflect complementary modes of learning (Norman & O’Reilly, 2003) and that both are necessary for a complete description of declarative memory.

## Conclusions

We describe a formal approach to extracting generating function of a language from sequentially-presented words. This model, which we refer to as pTCM, utilizes a distributed

representation of temporal context to create a prediction vector. The semantic representation of a word is constructed by aggregating the prediction vectors that are available when it is presented. We prove several useful results and describe an approximation to pTCM, which we refer to as pTCM<sub>A</sub>, that can be advantageous for learning large languages that require extensive training sets. We demonstrate that when the training set leaves the model far from steady state, nonzero  $\phi$  enables the model to make a better prediction for languages that contain equivalence classes among words. Because the representation of temporal context is inherited from a model that has been extensively applied to episodic recall tasks, pTCM may eventually prove to represent an important step towards a unified model of declarative memory.

### References

- Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). A neural probabilistic language model. *Journal of Machine Learning Research*, 3, 1137-1155.
- Christiansen, M. H., & Chater, N. (2001). Connectionist psycholinguistics: Capturing the empirical data. *Trends in Cognitive Science*, 5, 82-88.
- Eichenbaum, H. (2000). A cortical-hippocampal system for declarative memory. *Nature Reviews, Neuroscience*, 1(1), 41-50.
- Elman, J. L. (1991). Distributed Representations, Simple Recurrent Networks, and Grammatical Structures. *Machine Learning*, 7, 195-225.
- Griffiths, T. L., Steyvers, M., & Tenenbaum, J. B. (2007). Topics in semantic representation. *Psychological Review*, 114(2), 211-44.
- Harris, Z. (1951). *Methods in Structural Linguistics*. Chicago, IL: University of Chicago Press.
- Howard, M. W., Fotedar, M. S., Datey, A. V., & Hasselmo, M. E. (2005). The temporal context model in spatial navigation and relational learning: Toward a common explanation of medial temporal lobe function across domains. *Psychological Review*, 112(1), 75-116.
- Howard, M. W., Jing, B., Rao, V. A., Probyn, J. P., & Datey, A. V. (2009). Bridging the gap: Transitive associations between items presented in similar temporal contexts. *Journal of Experimental Psychology : Learning, Memory, and Cognition*(35), 391-407.
- Howard, M. W., & Kahana, M. J. (2002). A distributed representation of temporal context. *Journal of Mathematical Psychology*, 46(3), 269-299.
- Howard, M. W., Kahana, M. J., & Wingfield, A. (2006). Aging and contextual binding: Modeling recency and lag-recency effects with the temporal context model. *Psychonomic Bulletin & Review*, 13, 439-445.
- Howard, M. W., Shankar, K. H., & Jagadisan, U. K. K. (submitted). Constructing semantic representations from a gradually-changing representation of temporal context. *Topics in Cognitive Science*.
- Jones, M. N., & Mewhort, D. J. K. (2007). Representing word meaning and order information composite holographic lexicon. *Psychological Review*, 114, 1-32.
- Kwantes, P. J. (2005). Using context to build semantics. *Psychonomic Bulletin & Review*, 12(4), 703-10.
- Landauer, T. K., & Dumais, S. T. (1997). Solution to Plato's problem : The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, 104, 211-240.
- Lund, K., & Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instruments & Computers*, 28(2), 203-208.
- Miller, G. A., & Charles, W. G. (1991). Contextual correlates of semantic similarity. *Language and Cognitive Processes*, 6, 1-28.

- Mintz, T. H. (2002). Category induction from distributional cues in an artificial language. *Memory and Cognition*, *30*(5), 678-686.
- Norman, K. A., & O'Reilly, R. C. (2003). Modeling hippocampal and neocortical contributions to recognition memory: a complementary-learning-systems approach. *Psychological Review*, *110*(4), 611-46.
- Plate, T. A. (2003). *Holographic Reduced Representation: Distributed Representation for Cognitive Structures*. Stanford, CA: CLSI Publications.
- Rao, V. A., & Howard, M. W. (2008). Retrieved context and the discovery of semantic structure. In J. Platt, D. Koller, Y. Singer, & S. Roweis (Eds.), *Advances in neural information processing systems 20*. Cambridge, MA: MIT Press.
- Sahlgren, M., Holst, A., & Kanerva, P. (2008). Permutations as a means to encode order in word space. In *Proceedings of the 30th annual conference of the cognitive science society* (p. 1300-1305). Austin, TX: Cognitive Science Society.
- Sederberg, P. B., Howard, M. W., & Kahana, M. J. (2008). A context-based theory of recency and contiguity in free recall. *Psychological Review*, *115*, 893-912.
- Solan, Z., Horn, D., Ruppin, E., & Edelman, S. (2005). Unsupervised learning of natural languages. *Proceedings of the National Academy of Science*, *102*, 11629-11634.