Learning location invariant orthographic representations for printed words

Frédéric Dandurand [a1], Jonathan Grainger [a2], and Stéphane Dufau [a3]

[a] Laboratoire de Psychologie Cognitive, Université d'Aix-Marseille, Pôle 3 C, LPC/CNRS/UMR 6146

3, place Victor Hugo, Bat. 9, Case D, 13331 Marseille Cedex 1

[1] Corresponding author. Email: frederic.dandurand@univ-provence.fr,

Phone: +33 (0) 4 88 57 68 94, Fax: +33 (0) 4 88 57 68 95

[1] Email: jonathan.grainger@univ-provence.fr, Phone: +33 (0) 4 88 57 68 92, Fax: +33 (0) 4 88 57 68 95

[1] Email: stephane.dufau@univ-provence.fr, Phone: +33 (0) 4 88 57 69 05, Fax: +33 (0) 4 88 57 68 95

Word Count (Including references and captions): 8016

Abstract

Neural networks were trained with backpropagation to map location-specific letter identities (letters coded as a function of their position in a horizontal array) onto location-invariant lexical representations. Networks were trained on a corpus of 1179 real words, and on artificial lexica in which the importance of letter order was systematically manipulated. Networks were tested with two benchmark phenomena – transposed-letter priming and relative-position priming – thought to reflect flexible orthographic processing in skilled readers. Networks were shown to exhibit the desired priming effects, and the sizes of the effects were shown to depend on the relative importance of letter order information for performing location invariant mapping. Presenting words at different locations was found to be critical for building flexible orthographic representations in these networks, since this flexibility was absent when stimulus location did not vary.

Keywords: reading, orthographic processing, supervised learning, artificial neural networks

1. Introduction

Several recent computational models of visual object recognition posit a hierarchical system of processing in which simple and local features are gradually integrated into more abstract and complex features using receptive fields of increasing size [1]. These hierarchical architectures account for the progressive invariance to size, shape, and location, that is achieved as one moves through the visual pathways from V1, V2 up to occipital and temporal cortex. The mechanisms that we develop to process printed words while learning to read, borrow heavily from the basic machinery of visual object recognition [2, 3]. Therefore visual word recognition shares many of the characteristics of object recognition. Location invariance is one such characteristic, since skilled readers are able to identity words that are displaced relative to a central fixation point without having to re-fixate the centre of the word. Given that even very small shifts of location imply a complete change in retinal activity, this implies that some form of non-retinotopic code is involved in visual word recognition. The key question concerns the precise nature of this location-invariant, word-centered code, and how it is activated by retinotopic features.

Some psychological models have postulated that the shift from a location-specific, retinotopic orthographic code to a location-invariant orthographic code is achieved by coding for combinations of letters in the correct order for both contiguous and non-contiguous letter sequences [2, 4]. For example, in the models of Grainger and van Heuven [5] and Whitney [6], so-called open bigrams code two-letter combinations in a position-independent yet ordered, but not necessarily contiguous fashion. For example, WITH is composed of following open bigrams: WI, WT, WH, IT, IH and TH. In certain versions of these models activation of open bigrams can be modulated by distance (i.e., contiguous bigrams like WI are more active than non-contiguous bigrams such as IH). An important characteristic of open bigrams is that, while they allow for non-contiguous letter combinations, they preserve letter order. For example, IW is not an open bigram for the word WITH.

The theoretical backbone of the present study is Grainger and van Heuven's [5] model of orthographic processing [see also 7] illustrated in Figure 1. In this model a bank of location-specific letter detectors perform parallel independent letter identification. A given configuration of visual features at a specific location along the horizontal meridian signals the presence of a given letter at that location (see [8, 9] for evidence in favor of such retinotopic letter detectors). These location-specific letter detectors then activate location-independent open-bigram units. Open-bigrams then send activation to all compatible word representations in an interactive-activation network.

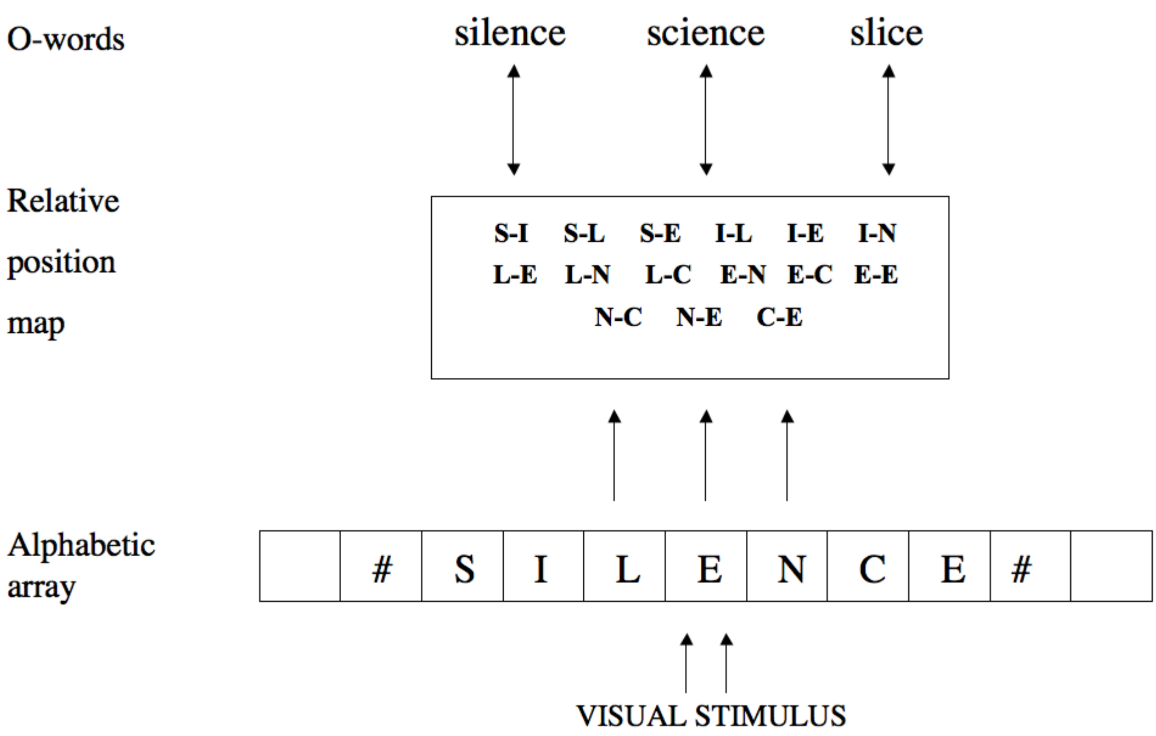

Figure 1 - Grainger and van Heuven's model of orthographic processing. Visual features extracted from a printed word feed activation into a bank of location-specific alphabetic character detectors (the alphabetic array). Each slot in the array codes for the presence of a given letter identity at a given location along the horizontal meridian. The next level of processing combines information from different processing slots in the alphabetic array to provide a relative position code for letter identities. These relative-position coded letter identities control activation at the level of whole-word orthographic representations (O-words) via bi-directional connections with all units at the relative position level.

In the present study we investigate, using backpropagation neural networks, to what extent the constraints of learning location-invariant lexical representations leads naturally to the development of the kind of flexible relative-position code described in the Grainger and van Heuven model. Following the Grainger and van Heuven model, we implemented location-specific letter detectors as input, and simulated presentation of the same word at different locations by activating different sets of letter detectors. The task consisted in recognizing these words presented at different locations as identical, location-independent lexical units (orthographic word forms). The network was trained on a corpus of real words, and on artificial lexica in which the importance of letter order was manipulated systematically.

One prior study has investigated the learning of location independent orthographic representations using backpropagation. Shillcock and Monaghan [10] used a task, which they called shift invariant identity mapping, that consists in mapping location-specific letters into a location independent representation of the same letters. For example, a neural network would learn to associate patterns WITH##, #WITH# and ##WITH (in which # represent blanks) to the common output WITH coded as a given letter identity at each of four possible positions (slot-coding). In their model, Shillcock and Monaghan simulated visual hemifields by splitting processing of the input slot at its center, sending these split inputs to two independent processing streams. Model splitting accounted for the superiority effect of exterior (i.e., first and last) letters of words in reading – network error was lower for exterior letters in the split model, but not lower in a non-split model. The present study provides an adaptation of Shillcock and Monaghan's modelling strategy, applied here to the learning of location-invariant orthographic representations.

We present three sets of simulations: (1) artificial lexica with 7 locations, (2) real word lexicon with 7 locations, and (3) real word lexicon in a single location. These simulations were designed to explore the nature of the internal representations that are developed when learning to map a location-specific orthographic representation onto a location-invariant lexical representation (whole-word orthographic representation); that is, learning certain ordered combinations of letters as representing words. The networks were tested with two key behavioral effects thought to reflect flexible orthographic coding in human participants: the transposed-letter priming effect, and the relative-position priming effect. Both effects have been observed using a masked priming paradigm that eliminates the role of various types of strategic responding associated with standard priming. The transposed-letter effect is a superior priming effect from primes formed by transposing two of the target's letters (e.g., gadren-garden) compared with a prime formed by substituting two of the target's letters (e.g., galsen-garden). The relative-position priming effect is a processing advantage for targets preceded by primes formed of a subset of the target's letters (e.g., grdn-garden) compared with a prime formed of the same subset of letters in the wrong order (e.g., gdrn - garden). Both of these priming effects argue against rigid slot-based coding schemes for letter encoding and are in favor of proposals for more flexible orthographic coding [e.g., 5, 6, 11, see 3 for a review].

Other research has also investigated flexible coding of letter order, attempting to account for phenomena such as letter transposition, letter migration, repeated letters, and relative-position priming. For example, Gomez, Ratcliff and Perea [11] account for such flexibility in their model using uncertainty about letter positions. In contrast to rigid, slot-based coding used in interactive-activation like models, letter position is represented as a probability distribution in their model, so that a letter present at a given position also provides evidence, albeit to a lesser extent, for the presence of that letter at neighbouring positions. However, letter position in the overlap model and similar approaches is defined as letter position in the word, independently of where the word is located. These models therefore fail to address the difficult issue of how information coded as being present at a particular location on the retina is mapped onto a word-centered representation. In the present study we train networks to map a set of location-specific letter identities (where location refers to location along the horizontal meridian) onto location-invariant lexical representations via a layer of hidden representations. We then examine whether these networks can simulate the kind of flexible orthographic processing seen in human experiments.

2. Methods

All simulations used standard feedforward neural networks that were trained with a standard gradient descent technique with momentum [12]. We used a learning rate of 0.1, and a momentum value of 0.9. The criterion for successful training was reaching a target level for the sum of squared errors (SSE) between targets and network outputs. For the number of hidden units, we used the square root of the number of training patterns, rounded up to closest integer. Connection weights were initialized randomly within a range of -1.0 and 1.0.

2.1. Input coding

We used sparse, local coding [13]: each letter slot was encoded using 26 binary values indicating the presence or absence of a given letter, in alphabetical order. For instance, presence of letter *A* was encoded as [1 0 0 ... 0], *B* as [0 1 0 ... 0], and *Z* as [0 0 … 1]. Blanks were coded using zeros in all positions [0 0 0... 0]. Words were presented in seven positions along a ten-slot input vector. As illustration, Table 1 presents the encoded input pattern for word WITH in central position (###WITH###). The input pattern presented to the network would simply have been concatenation of rows 1 to 10 into a 260 binary-valued vector.

	Presence of letter coding (1 bit per letter)																									
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\boxed{1}$	0	0	0
5	0	0	0	0	0	0	0	0	$\boxed{1}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$\boxed{1}$	0	0	0	0	0	0
7	0	0	0	0	0	0	0	$\boxed{1}$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 1 - Example of encoded input pattern for word WITH presented in central position (###WITH###). The first column indicates slot position. In this example, slots 1 to 3 and 6 to 10 contain blanks. This input vector is 260 bits long (10 slots x 26 letters per slot x 1 bit per letter).

2.2. Output coding

Each word is coded onto an output unit. Presence of the corresponding word is coded using a value 1, absence is coded as 0. For example, if target words are ABCD, EFGH, IJKL, MNOP and QRST, an input of #ABCD##### would correspond to output 1 0 0 0 0, whereas ####IJKL## would be associated with output vector 0 0 1 0 0. An output value of 1 coded for the presence of the word in the input vector, and 0 coded for its absence. As illustration, Table 2 presents a sample of training patterns from the target words only condition.

Input vector										Output vector				
A	B	C	D	#	#	#	#	#	#	1	0	0	0	0
#	A	B	C	D	#	#	#	#	#	1	0	0	0	0
#	#	#	#	A	B	C	D	#	#	1	0	0	0	0
E	F	G	H	#	#	#	#	#	#	0	1	0	0	0
#	#	#	#	#	#	E	F	G	H	0	1	0	0	0
#	#	#	M	N	O	P	#	#	#	0	0	0	1	0

Table 2 – Example of input and output for different words presented at different locations (#s represent blanks).

2.3. Composition of the training sets

In building training sets, we presented target words in different contexts to investigate if letter sequence or order influenced the representation built by backpropagation networks. We presented two types of training sets: (1) artificial lexica of four-letter strings in order to manipulate the relative importance for letter order for determining lexical identity, and (2) a realistic corpus of 1179 real four-letter words.

2.3.1. Artificial lexica

We used four types of artificial lexica. All these lexica comprised the following five target words: ABCD, EFGH, IJKL, MNOP and QRST. They also optionally included filler patterns designed to manipulate the importance of letter order, as illustrated in Figure 2.

Importance of letter order

min → max

Training sets	Target words	Target words & recombinations	Target words & anagrams	Target words & anagrams & recombinations
Examples containing letter A	ABCD	ABCD & QJAP	ABCD & DCBA	ABCD & DCBA & QJAP

Figure 2 - Importance of letter order in the artificial lexica.

The first training set contained the five target words only, which were the same across replications: (1) ABCD, (2) EFGH, (3) IJKL, (4) MNOP and (5) QRST, for a total of 35 training patterns (5 words x 7 positions). Networks learning this training set had 6 hidden units (that is, the square root of 35, rounded up). Replications differed in network initial conditions (random weights) only.

The second training set, dubbed “target words and letter recombinations” or “recombinations” for short, included five filler words made of the same letters as the target words, but randomly recombined. This was done by pooling letters A to T and making filler words by randomly drawing letters, without replacement. Although target words were the same across replications, due to random selection, filler words were different in each replication. An example of a recombinations training set is: (1) ABCD, (2) EFGH, (3) IJKL, (4) MNOP, (5) QRST, (6) TMEK, (7) QGAP, (8) CHNI, (9) BJFS, and (10) RDOL, for a total of 70 training patterns (10 words x 7 positions). Networks trained under this condition also had 9 hidden units. Replications differed in the composition of the filler words, and in network initial conditions (random weights).

The third training set, dubbed “target words and anagrams” or simply “anagrams” for short, included one anagram for each of the five target word. To maximize distance, we built the anagram by reversing letter order (i.e., ABCD → DCBA). The anagrams training set was the same across replications, and contained the following words: (1) ABCD, (2) EFGH, (3) IJKL, (4) MNOP, (5) QRST, (6) DCBA, (7) HGFE, (8) LKJI, (9) PONM, and (10) TSRQ, for a total of 70 training patterns (10 words x 7 positions). Networks trained under this condition had 9 hidden units (that is, the square root of 70, rounded up). Replications differed in network initial conditions (random weights) only.

The fourth training set, dubbed “target words, anagrams and letter recombinations”, or “combo” for short, include the five target words, five anagrams (as described above) and five recombinations (as described above). Combo sets contain 105 training patterns (15 words x 7 positions), and replications differed in the composition of the filler words, and in network initial conditions (random weights). Networks trained under this condition also had 11 hidden units.

With the target words only training set, a single letter determines lexical identity (for instance, letter A is evidence for one and only one word, ABCD, therefore letter combinations are not a requisite for successful learning. This is not the case with the recombinations training set also requires coding of letter combinations for successful learning, as the same letter appears in different words, but precise order information is not a requisite for successful encoding. Finally, the anagrams training set where letter order becomes critical for determining lexical identity, Therefore, the different training sets impose different levels of relevance of order information in learning to map location-specific letter identities onto location-invariant word representations. Order is least relevant in the target words only training set, more relevant in the recombinations condition and most relevant to the anagrams & combo conditions.

2.3.2. Real word lexicon

We also trained networks using a realistic corpus of 1179 real four-letter words, previously used by McClelland and Rumelhart [12].

2.4. Network evaluation

We investigated the nature of the internal representations built by the backpropagtion algorithm while learning to map location-specific letter representations onto location-invariant lexical representations. More precisely, we wanted to know whether the model would develop some form of flexible orthographic code similar to the type of code revealed in recent research on orthographic processing in skilled readers [see 3, for review]. We investigated these representations using test sets based on priming. In humans, priming effects are often explained by spreading activation among related or shared cognitive representations. These activation spreads may facilitate or hinder subsequent access to these related representations. This is generally measured as faster reaction times for better primes. In our models, we measured priming effects as the generalization from primes to target items using an accuracy measure: the more a given input primes a given target item, the more that target item will be activated, and thus the more accurate the network’s response will be. In other words, our operational definition of priming relates to the activation of the target word: (1) the discrepancy (MSE) between the network output and the output expected for this target word (a single one for the target word in a vector of zeros), or (2) the ability of the prime to activate the output unit associated with the target word more than any other output unit, measured here as accuracy.

We manipulated two factors: the composition of the training set (4 levels of artificial lexica: (1) target words only; (2) target words and recombinations; (3) targets words and letter anagrams; (4) target words, letter recombinations and anagrams; and (5) a real word lexicon), and the priming regime used for testing (2 levels: relative-position priming, and transposed-letter priming). We combined train and test regimes in a combinatorial fashion, for a total of 10 simulations.

We tested model performance under two priming manipulations: relative-position priming and transposed-letter priming.

2.4.1. Relative-position priming

We studied a network's ability to simulate relative-position priming using primes formed of a subset of the target's letters, namely three-letter sequences from four-letter target words. We manipulated two parameters of the prime letters: (1) order (2 levels: forward and backward) and (2) contiguity (2 levels: contiguous and non-contiguous). The exhaustive set of test patterns is given in Table 3.

	Order of letters in primes		
Contiguity	Forward	Backward	Target word
Contiguous	ABC, BCD	CBA, DCB	ABCD
	EFG, FGH	GFE, HGF	EFGH
	IJK, JKL	KJI, LKJ	IJKL
	MNO, NOP	ONM, PON	MNOP
	QRS, RST	SRQ, TSR	QRST
Non-contiguous	ABD, ACD	DBA, DCA	ABCD
	EFH, EGH	HFE, HGE	EFGH
	IJL, IKL	LJI, LKI	IJKL
	MNP, MOP	PNM, POM	MNOP
	QRT, QST	TRQ, TSQ	QRST

Table 3 – Exhaustive set of test patterns for the relative-position priming task.

We measured the amount, or quality, of priming using two methods (note that primes were never seen during training). First, we measured network output error (MSE), expecting that the better the input prime, the lower the output error would be (that is, the difference between the network output and the expected answer, consisting of having the output node corresponding to the target word fully activated, and all other nodes set to zero). Second, we computed accuracy as the proportion (or rate) of correct network responses. A network response was considered correct when, for some input prime, the associated target word was the most active item in the lexicon (i.e., the activation of the corresponding output unit was greater than all other units).

During training, input words were presented at seven locations but, as shown in Table 4, networks were tested on central locations only (i.e., slots no. 5, 6 and 7). With this design, each letter of some training word was seen exactly once at each testing location. Therefore, letters were seen equally frequently in any position they may appear in a test prime (i.e., regardless of contiguity and order). Thus, differences in network performance could not be attributed to certain letter-slot combinations trained more than others, and would therefore reflect relationships between letters.

	1	2	3	4	5	6	7	8	9	10
Relative-position test prime (3 characters)										
	#	#	#	#	**X**	**X**	**X**	#	#	#
Training patterns										
1	A	B	C	**D**	**#**	**#**	**#**	#	#	#
2	#	A	B	**C**	**D**	**#**	**#**	#	#	#
3	#	#	A	**B**	**C**	**D**	**#**	#	#	#
4	#	#	#	**A**	**B**	**C**	**D**	#	#	#
5	#	#	#	**#**	**A**	**B**	**C**	D	#	#
6	#	#	#	#	#	**A**	**B**	C	D	#
7	#	#	#	#	#	**#**	**A**	B	C	D

Table 4 – Illustration of training and test data for the relative-position priming task. Xs indicate where the three letters of the test strings were presented (always in the center). Each word in the training set was presented in the seven locations of the table. We see that central locations (slots 5 to 7) were trained on all the letters of the train data.

2.4.2. Transposed-letter priming

A network's ability to simulate transposed-letter priming was examined using primes formed by transposing the two central letters of targets (e.g., ABCD-ACBD) and comparing the effects of these primes with primes formed by replacing the two central letters with letters from a different word (e.g., AGFD). These priming effects were compared with simple repetition priming where the prime is the same stimulus as the target (e.g., ABCD) and another prime condition with different central letters (e.g., AFGD). Therefore two factors were manipulated. First, the origin of central, or inner, letters: (1) from the target word, or

(2) from a different word from the target word. Second, the order of central letters: (1) forward, or (2) backward. The exhaustive set of test patterns is presented in Table 5. It should be noted that in the condition with the same letters in the correct direction the prime is the same word as the target, a condition referred to as repetition priming in the behavioural literature.

Origin of central letters	Order of central letters Forward	Backward	Target word
Same word	ABCD	ACBD	ABCD
	EFGH	EGFH	EFGH
	IJKL	IKJL	IJKL
	MNOP	MONP	MNOP
	QRST	QSRT	QRST
Different word	AFGD	AGFD	ABCD
	EJKH	EKJH	EFGH
	INOL	IONL	IJKL
	MRSP	MSRP	MNOP
	QBCT	QCBT	QRST

Table 5 – Primes in the transposed-letter priming experiment.

Similarly to the experiment with relative-position priming, we presented primes in central locations, as shown in Table 6, such that letters in test patterns were seen exactly once per location during training.

	1	2	3	4	5	6	7	8	9	10
Transposed-letter priming test patterns (4 characters)										
	#	#	#	**X**	**X**	**X**	**X**	#	#	#
Training patterns										
1	A	B	C	**D**	**#**	**#**	**#**	#	#	#
2	#	A	B	**C**	**D**	**#**	**#**	#	#	#
3	#	#	A	**B**	**C**	**D**	**#**	#	#	#
4	#	#	#	**A**	**B**	**C**	**D**	#	#	#

5	#	#	#	**#**	**A**	**B**	**C**	D	#	#
6	#	#	#	#	#	**A**	**B**	C	D	#
7	#	#	#	#	#	**#**	**A**	B	C	D

Table 6 – Illustration of training and test data for the transposed-letter priming task. Xs indicate where the four letters of the test strings were presented (always in the centre). Each word in the training set was presented in the seven locations of the table. We see that central locations (slots 4 to 7) were trained on all the letters of the train data.

2.5. Discrimination of words and nonwords

Finally, as a general evaluation of a network's success in correctly learning to map letter representations onto lexical identity, we measured how well the network could discriminate words from nonwords. For a word to be considered correct, the activation of the correct corresponding lexical output unit had to be higher than a threshold value, empirically found to be appropriate at a level of 0.99. For a nonword to be considered correctly rejected, activations of all output lexical units had to be below threshold.

3. Results

3.1. Artificial lexica

For the artificial lexica, a sample of 20 networks was generated for each condition. The target SSE for successful completion was 1. We measured the network's ability to discriminate words from nonwords in the combo condition. Networks' accuracy for words was 98.0% while correctly rejecting 98.0% of nonwords.

3.1.1. Relative-position priming

A summary of accuracy results for the relative-position task are presented in Figure 3. As we can see, accuracy for the backward primes decreases as the importance of letter order increases (left to right), while accuracy for the forward primes remains high. The networks therefore reveal a relative-position priming effect the size of which is determined by the importance of letter order in the training set. Contiguity had an overall smaller influence on network performance, with an advantage for non-contiguous primes emerging in certain conditions. Network error (MSE) results are presented in Table 7, Table 8, Table 9 and Table 10, respectively.

	Order	
Contiguity	Forward	Backward
Contiguous	0.3 (0.1)	2.8 (2.5)

Non-contiguous	1.7 (1.6)	2.7 (2.9)

Table 7 - Network error (Mean Squared Error, or MSE) for the relative-position priming task, and the training set containing target words only. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the relative-position priming task and the training set containing target words only, accuracy is perfect in all four conditions. A two-way ANOVA on network error (MSE) with contiguity and order as repeated factors revealed a main effect of order, $F(1,19) = 8.5$, $p < 0.01$, a significant interaction, $F(1,19) = 5.2$, $p < 0.05$, but no effect of contiguity, $F(1,19) = 3.4$, $p > 0.05$. Error was lower in forward ($M = 1.0\text{x}10^{-3}$) than backward ($M = 2.7\text{x}10^{-3}$) primes, and the difference between forward and backward primes was larger in the contiguous ($2.4\text{x}10^{-3}$) condition than the non-contiguous condition ($0.1\text{x}10^{-3}$).

	Order	
Contiguity	Forward	Backward
Contiguous	1.2 (0.9)	11 (9)
Non-contiguous	6 (4)	12 (9)

Table 8 - Network error (Mean Squared Error, or MSE) for the relative-position priming task, and the training set containing letter recombinations. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the relative-position priming task and the training set containing recombinations, a two-way ANOVA on accuracy with contiguity and order as factors revealed a main effect of order, $F(1,19) = 12.8$, $p < 0.01$, but no effect of contiguity, $F(1,19) < 1$, and no interaction, $F(1,19) < 1$. In contrast, a two-way ANOVA on network error (MSE) revealed a main effect of order, $F(1,19) = 31$, $p < 0.001$, an effect of contiguity, $F(1,19) = 5.9$, $p < 0.05$, and no interaction, $F(1,19) = 4.8$, $p < 0.05$. Accuracy was larger for the forward primes ($M = 1.0$) than backward primes ($M = 0.97$) and error was lower for forward primes ($M = 0.003$) than backward primes ($M = 0.012$). In addition, error was lower in contiguous ($M = 6.1\text{x}10^{-3}$) than non-contiguous primes ($M = 9\text{x}10^{-3}$), and the difference between contiguous and non-contiguous primes was larger in the forward condition ($4.8\text{x}10^{-3}$) than the backward ($1.0\text{x}10^{-3}$) condition.

	Order	
Contiguity	Forward	Backward

Contiguous	28 (8)	125 (9)
Non-contiguous	14 (5)	140 (10)

Table 9 - Network error (Mean Squared Error, or MSE) for the relative-position priming task, and the training set containing anagrams. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the relative-position priming task and the training set containing anagrams, a two-way ANOVA on accuracy with contiguity and order as factors revealed a main effect of order, $F(1,19) = 1777$, $p < 0.001$, and a significant interaction, $F(1,19) = 96$, $p < 0.001$, but no effect of contiguity, $F(1,19) = 1.1$, $p > 0.05$. Similarly, a two-way ANOVA on network error (MSE) revealed a main effect of order, $F(1,19) = 2935$, $p < 0.001$, no effect of contiguity, $F(1,19) = 2.2$, $p > 0.05$, and a significant interaction, $F(1,19) = 94$, $p < 0.001$. Accuracy was larger for forward primes ($M = 0.88$) than backward primes ($M = 0.14$), and error was lower for forward primes ($M = 0.02$) than backward ($M = 0.13$) primes. The interaction stems from the fact that these differences were larger for non-contiguous primes than for contiguous primes.

	Order	
Contiguity	Forward	Backward
Contiguous	22 (6)	89 (6)
Non-contiguous	16 (5)	89 (9)

Table 10 - Network error (Mean Squared Error, or MSE) for the relative-position priming task, and the combo training set. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the relative-position priming task and the training set containing anagrams and recombinations (combo), a two-way ANOVA on accuracy with contiguity and order as factors revealed a main effect of order, $F(1,19) = 1650$, $p < 0.001$, and a significant interaction, $F(1,19) = 9.4$, $p < 0.01$, but no effect of contiguity, $F(1,19) = 2.1$, $p > 0.05$. Similarly, a two-way ANOVA on network error (MSE) revealed a main effect of order, $F(1,19) = 2045$, $p < 0.001$, no effect of contiguity, $F(1,19) = 3.8$, $p > 0.05$, and a significant interaction, $F(1,19) = 8.4$, $p < 0.01$. This pattern of results is identical to the anagrams condition: accuracy was larger for forward primes ($M = 0.86$) than backward primes ($M = 0.10$), and error was lower for forward primes ($M = 0.02$) than backward ($M = 0.09$) primes. The interaction stems from the fact that these differences were larger for non-contiguous primes than for contiguous primes.

In sum, we found a robust effect of letter order (namely, a higher accuracy on forward than on backward primes), a relative-position priming effect, whereby an ordered subset of the target’s letters provides a better match to the target than the same subset of letters in reversed order. As expected, this effect was strongest in the anagrams and combo conditions where letter order matters the most, but it was also present in the recombinations condition. It was nearly inexistent for the condition in which letter order does not matter, that is the target words only condition. We also found an order by contiguity interaction, which was significant only in the anagrams and combo conditions, and reflected an advantage for the non-contiguous primes in the forward condition.

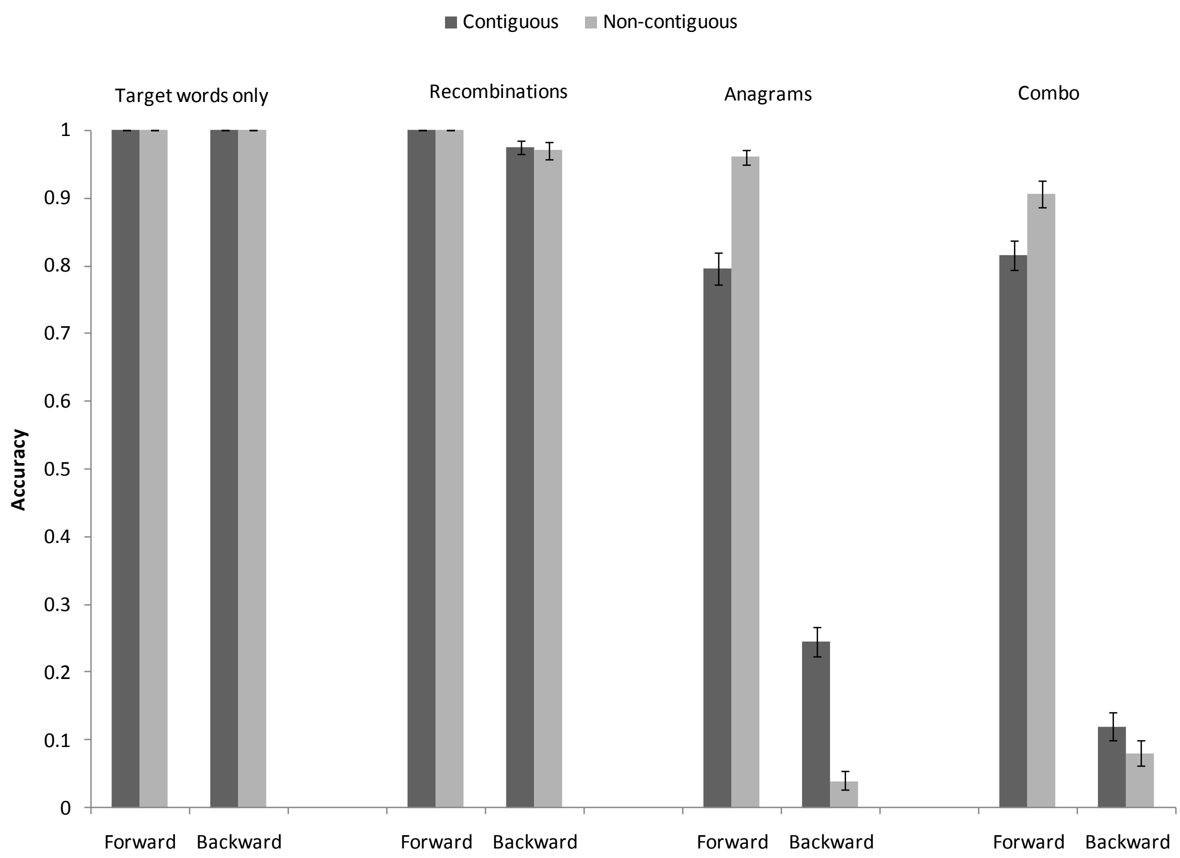

Figure 3 - Summary of accuracy results for the relative-position priming task. Example primes for the target ABCD are: ABC for the forward and contiguous condition, ABD for the forward and non-contiguous, CBA for the backward and contiguous and DBA for the backward and non-contiguous.

3.1.2. Transposed-letter priming

A summary of accuracy results for the transposed-letter priming task are presented in Figure 4. As can be seen in this figure, the networks successfully simulated the transposed-letter priming effect, and the size of this effect was practically as large as the effect of repetition priming. Network error (MSE) results are presented in Table 11, Table 12, Table 13 and Table 14, respectively for the training sets composed of target words only, containing recombinations, containing anagrams, and containing both (recombinations and anagrams).

Origin of central Letters	Order of central letters	
	Forward	Backward
Same	0.04 (0.01)	0.3 (0.3)
Different	140 (60)	130 (40)

Table 11 - Network error (Mean Squared Error, or MSE) for the transposed-letter priming task and the training set composed of the target words only. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the transposed-letter priming task and the training set composed of the target words only, a two-way ANOVA on accuracy with contiguity and order as factors revealed a main effect of origin, $F(1,19) = 107$, $p < 0.001$, but no main effect of order, $F(1,19) < 1$, and no interaction, $F(1,19) < 1$. Similarly, a two-way ANOVA on MSE revealed a main effect of origin, $F(1,19) = 187$, $p < 0.001$, but no main effect of order, $F(1,19) = 3.8$, $p > 0.05$, and no interaction, $F(1,19) = 4.0$, $p > 0.05$, although the latter two effects were trending. Accuracy was higher for primes with central letters from same word as the target ($M = 1.0$) than from a different word ($M = 0.49$) and smaller error ($M = 1.7\text{x}10^{-4}$ for same, and $M = 0.13$ for different).

Origin of central Letters	Order of central letters	
	Forward	Backward
Same	0.007 (0.003)	0.12 (0.2)
Different	80 (20)	90 (20)

Table 12 - Network error (Mean Squared Error, or MSE) for the transposed-letter priming task and the training set comprising recombinations. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the transposed-letter priming task and the training set comprising recombinations, two-way ANOVAs revealed a similar pattern of results as for the target words only condition: a main effect of origin on accuracy, $F(1,19) = 268$, $p < 0.001$, and on network error, $F(1,19) = 386$, $p < 0.001$, but no effect of order, nor interactions, Fs < 2.6, ps > 0.1.

Origin of central Letters	Order of central letters	
	Forward	Backward
Same	0.014 (0.010)	11 (10)

Different	60 (20)	60 (20)

Table 13 - Network error (Mean Squared Error, or MSE) for the transposed-letter priming task and the training set comprising anagrams. Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

With the transposed-letter priming task and the training set comprising anagrams, a two-way ANOVA on accuracy revealed a similar pattern of results, that is, a main effect of origin, $F(1,19) = 73$, $p < 0.001$, but no effect of order and no interaction, *F*s < 1.1. However, a two-way ANOVA on MSE revealed an additional significant effect of order, $F(1,19) = 5.4$, $p < 0.05$, in addition to the main effect of origin, $F(1,19) = 267$, $p < 0.001$. The interaction between order and origin was not significant, but trending, $F(1,19) = 3.7$, $p > 0.05$.

Origin of central Letters	Order of central letters	
	Forward	Backward
Same	0.0028 (0.0016)	9 (7)
Different	51 (12)	54 (10)

Table 14 - Network error (Mean Squared Error, or MSE) for the transposed-letter priming task and the training set comprising anagrams and recombinations (combo). Values presented in parentheses represent standard deviations. Values presented in table should be multiplied by 10^{-3}.

The pattern of results is the same with the training set comprising anagrams and recombinations (combo) as with the training set comprising only anagrams: a main effect of origin on accuracy, $F(1,19) = 180$, $p < 0.001$, but no effect of order nor interactions *F*s < 1.9. For network error (MSE), we found main effects of order, $F(1,19) = 26$, $p < 0.001$, and of origin, $F(1,19) = 564$, $p < 0.001$, but no interaction between order and origin, $F(1,19) = 3.1$, $p > 0.05$.

In short, the pattern across artificial lexica was consistent: higher accuracy and lower error when central letters were from the same word (i.e., the target) than from a different word, an intuitively appealing conclusion (see Figure 4). We also found a weaker effect of direction, which turned out significant only for network error only and when order was most relevant to the task (i.e., in conditions in which anagrams were included as fillers). This effect of direction reflects the stronger effects of repetition priming compared with effects of transposed-letter priming.

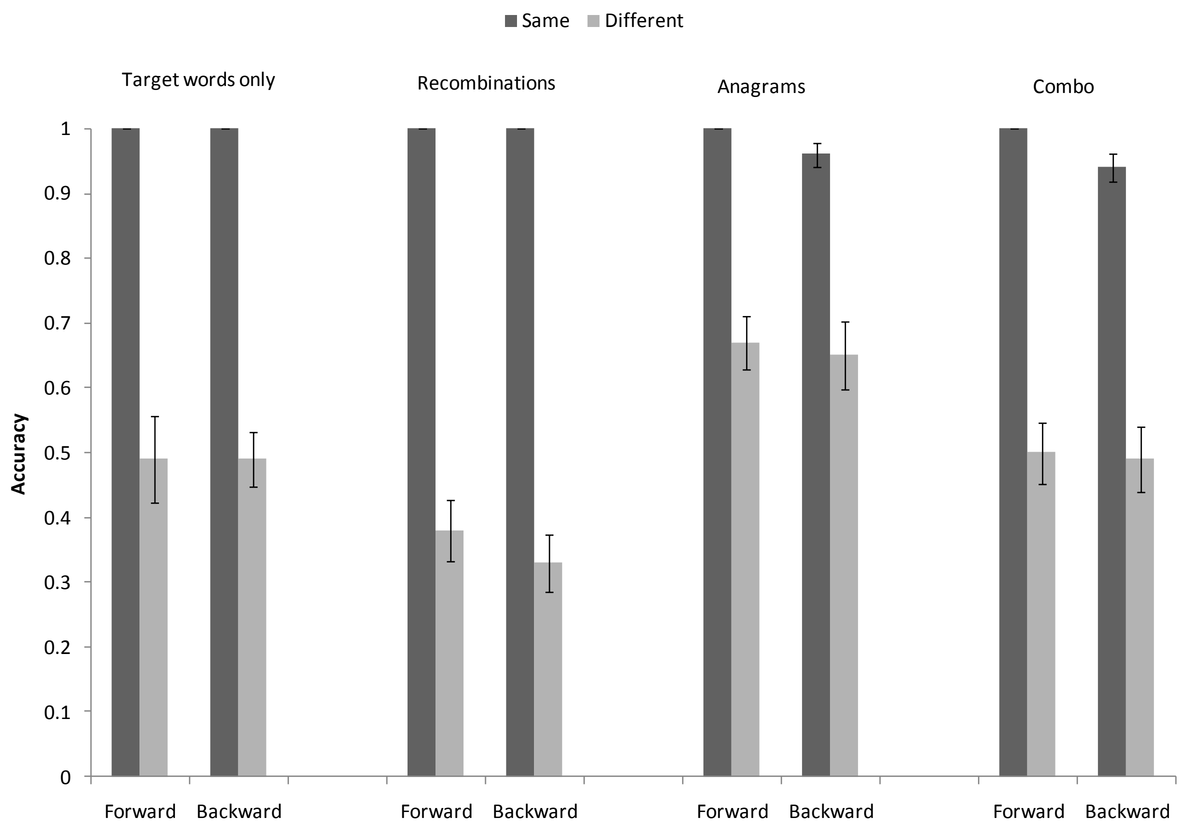

Figure 4 - Summary of accuracy results for the transposed-letter priming task. Example primes for the target ABCD are: ABCD for the forward and same condition, AFGD for the forward and different condition, ACBD for the backward and same condition, and AGFD for the backward and different condition. The forward-same vs. forward-different comparison measures repetition priming, and the backward-same vs. backward-different comparison measures transposed-letter priming.

3.2. Real word lexicon

A single network was trained with the real word training set of 1179 words of 4 letters in length. The target SSE for successful completion was 30. The training set contained 7 x 1179 = 8253 patterns, and the backpropagation network had 91 hidden units. In order to test word-nonword discrimination in this network a set of nonwords were derived from each real word by changing one letter (e.g., darm, stob), for a total of 1179 nonwords. The replacement location and identity of the substitution letter were randomly chosen. Our model exhibited perfect recognition accuracy for words (100.0%). The rate of correctly rejecting nonwords was 94.1%. Most incorrectly accepted nonwords (92.3%) were anagrams of real words (e.g., UDLY for DULY and ICOL for COIL).

3.2.1. Relative-position priming

Accuracy results for relative-position priming are presented in Figure 5 and error results in Table 15. The results show a relative-position priming effect with an advantage of forward primes over backward primes.

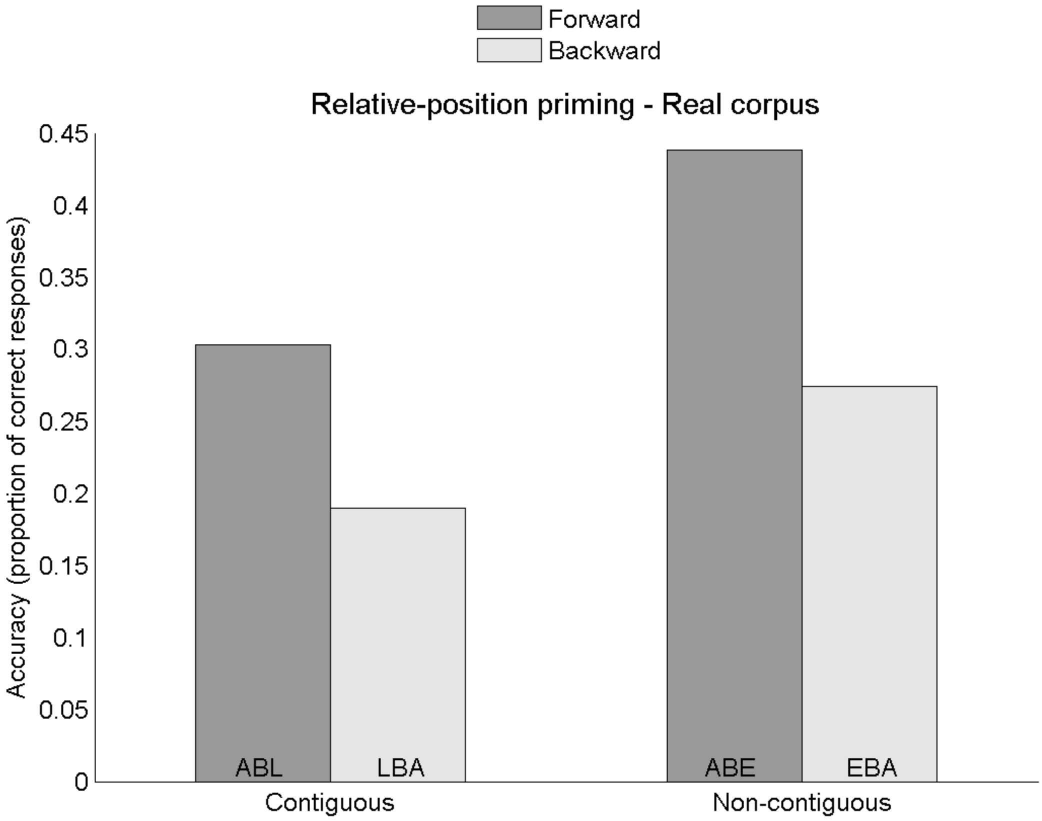

Figure 5 - Accuracy results for the relative-position priming task, and the training set containing real words. No error bar is provided for these single data points. Examples are given for real word ABLE.

	Order	
Contiguity	Forward	Backward
Contiguous	7.7	8.1
Non-contiguous	6.8	7.8

Table 15 - Network error (Mean Squared Error, or MSE) for the relative-position priming task, and the training set containing real words. Values presented in table should be multiplied by 10^{-4}.

3.2.2. Transposed-letter priming

Accuracy results for transposed-letter priming are presented in Figure 6 and error results in Table 16. The results show a transposed-letter priming effect. Accuracy is higher when central letters are from the target word than when they are from a different word even when the order of letters is reversed (backward condition). This transposed-letter priming effect is practically as strong as the repetition priming effect.

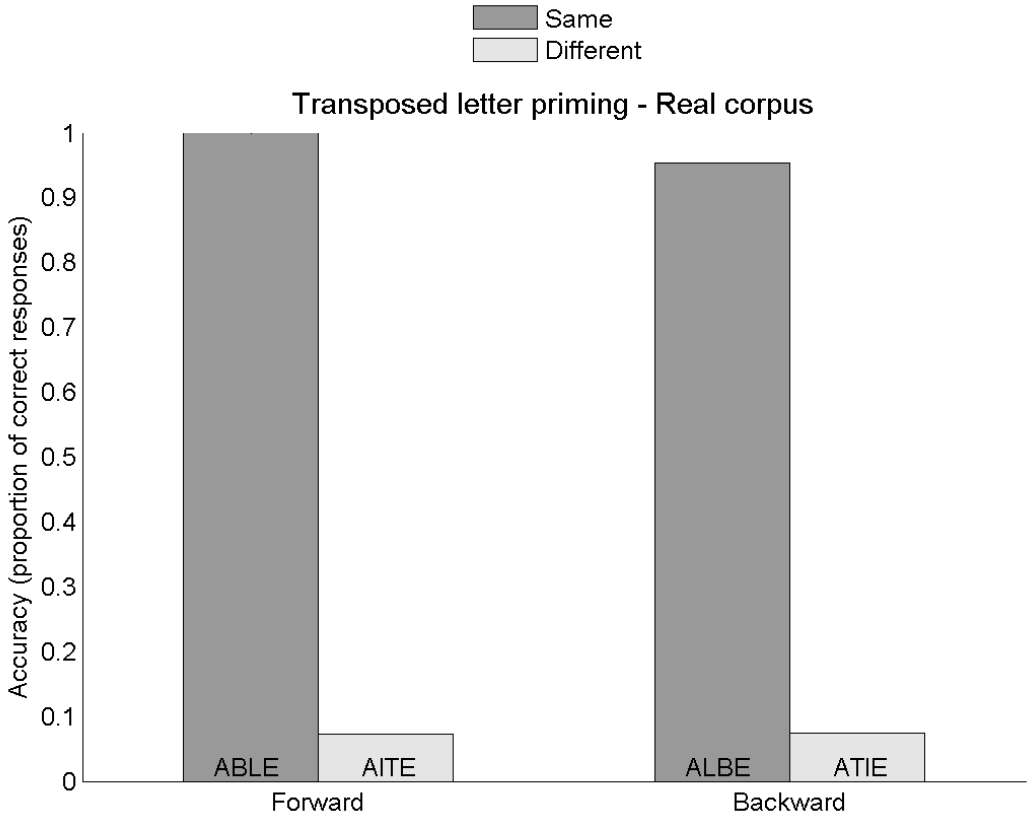

Figure 6 - Accuracy results for the transposed-letter priming task, and the training set containing real words. No error bar is provided for these single data points. Examples are given for the target word ABLE.

Origin of central Letters	Order of central letters	
	Forward	Backward
Same	0.0	0.1
Different	1.1	1.1

Table 16 - Network error (Mean Squared Error, or MSE) for the transposed-letter priming task and the training set composed of real words. Values presented in table should be multiplied by 10^{-3}.

3.2.3. Priming effects in a network trained with words at a single location

The network trained with a real word training set exhibited relative-position and transposed-letter priming effects. We interpret this ability to simulate such priming effects as reflecting an intervention of the type of flexible orthographic code that is developed when learning to map location-specific orthographic representations onto location-invariant representations. More precisely, the key hypothesis here is that it is the constraints involved in mapping totally independent sets of letter identities (i.e., the same letters appearing at different locations) onto the same lexical identity that forces the network to develop intermediate orthographic representations that acquire the kind of flexibility that is seen in experiments testing skilled readers. Therefore, such flexibility, as reflected in the simulated priming effects, should not be visible when the network is only trained at one location. Our final simulation study puts this prediction to test by training the network on only the central location.

In Figure 7, we present accuracy results for the relative-position priming task and the training set containing real words when words are only presented at the central location during training. As we can see, the priming effect is still present, as illustrated by a higher accuracy with forward than backward primes. However, accuracy for non-contiguous primes is now very small (less than 5%) suggesting that networks have not developed orthographic representations that are as flexible as when words are presented at different positions (compare Figure 5 and Figure 7).

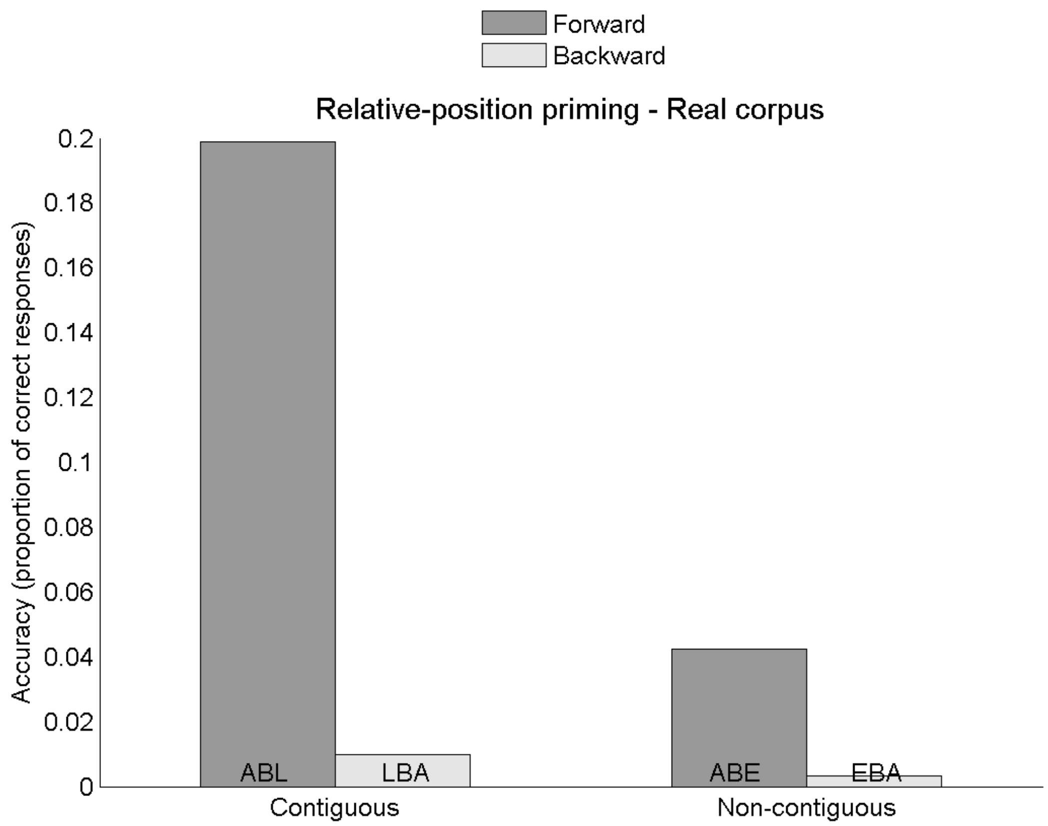

Figure 7 - Accuracy results for the relative-position priming task using the training set comprising real words but trained on central positions only.

In Figure 8, we present accuracy results for the transposed-letter priming task and the training set containing real words when words are only presented at the central location during training. As we can see, the transposed-letter priming effect effectively disappears when training only the central location (compare Figure 6 and Figure 8). This confirms our hypothesis that networks develop flexible orthogaphic representations only when the task involves processing input words presented in different positions.

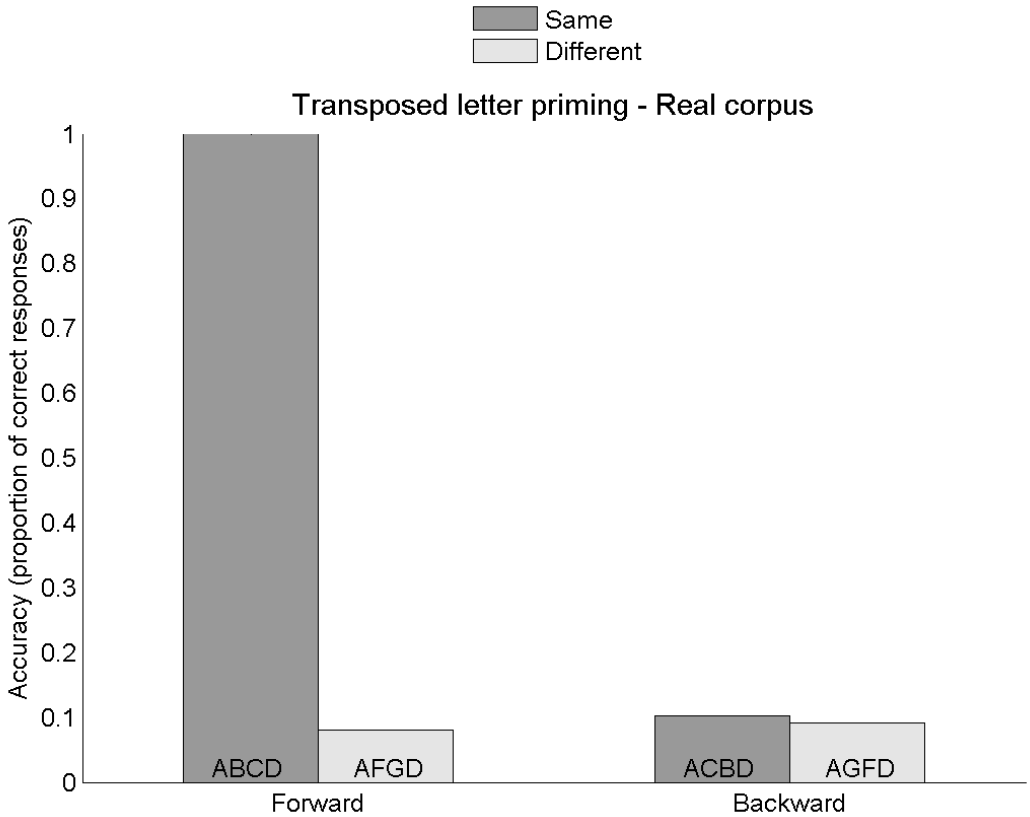

Figure 8 – Accuracy results for the transposed-letter priming task using the training set comprising real words but trained on central positions only.

4. General discussion

Neural networks were trained using backpropagation to map location-specific letter identities onto location-invariant lexical identities. The location-specificity of letter representations implied that when the same "word" input was presented at different locations, the network had to learn to map completely independent input representations onto the same output. In other words, the networks were trained to recognize that a given word is the same word independently of its location (location invariance). According to one account of orthographic processing in skilled readers [5], location-invariance is already achieved at a prelexical level of orthographic representation, where letter identities are coded independently of their position on the retina but relative to their position in the word. Furthermore, at this level of representation, letter position information is thought to be coded in a flexible manner, contrary to the rigid position-specific coding used in slot-based approaches. It is this flexibility that enables the Grainger and van Heuven model to capture empirical phenomena such as transposed-letter priming and relative-position priming.

The present study examined whether neural networks trained to map location-specific letter identities onto location-invariant lexical representations would acquire intermediate representations that would allow the network to exhibit the properties associated with flexible orthographic processing. The networks were trained with a variety of training regimes, including a real word lexicon, and artificial lexica in which the importance of letter order was systematically manipulated. These artificial training regimes forced the network to pay varying levels of attention to order information by varying the relevance of this information for the task. All networks were successful in learning the task, and the real word lexicon as well as the most complete artificial lexicon (the combo training set) exhibited accurate word – nonword discrimination following training. The networks were further evaluated on the two benchmark phenomena: transposed-letter priming and relative-position priming. Network accuracy (percentage of trials in which the target is the most activated output representation) and network error (MSE) revealed transposed-letter priming and relative-position priming. In the simulations run on the networks trained with artificial lexica, transposed-letter priming effects were found even when letter order was not important to solve the task (the target words only condition). In contrast, relative-position priming effects increased as the importance of letter order increased. The effects were small in the target words only condition, larger in the recombinations condition and largest in the conditions containing anagrams.

The simulations run on the network trained with a corpus of real words showed very large effects of transposed-letter priming that were practically as large as the effects of repetition priming. Relative-position priming effects were also evident, but the size of the priming effect was much smaller than that found with transposed-letter primes. This is in line with the results typically found with human participants [3]. Furthermore, relative-position priming effects were, if anything, greater for non-contiguous primes, a result that is line with certain models of orthographic processing [5, 6]. It could however be the case that the advantage for non-contiguous primes is the result of these primes having both of the target's outer letters appearing as outer letters in the prime (i.e., preceded or followed by a space). This was not the case for contiguous primes for which the last letter in the prime stimulus was not the last letter in the target word. Finally, both transposed-letter and relative-position priming effects disappeared with the network was trained on words presented at a single location, thus demonstrating the importance of shifts in location at the input for generating flexible, intermediate orthographic representations.

Summing up, two critical elements were found to be necessary for networks to develop flexible orthographic coding: (1) learning to map location-specific representations onto a location-invariant representation (i.e., having the same word presented at multiple locations in the input), and (2) training the network on a corpus in which letter position provides important information for constraining lexical identity. The real corpus had this characteristic, as many words differed only by a single letter, and the corpus included several anagrams. The results of the present simulations therefore suggest that, given the characteristics of natural language, flexible orthographic processing might emerge as a natural consequence of having to learn to map location-specific letter identities onto location-invariant lexical representations during reading acquisition.

5. Acknowledgements

This project was supported by the Agence Nationale de la Recherche (grant no. ANR-06-BLAN-0337).

6. References

[1] M. Riesenhuber, and T. Poggio, *Hierarchical models of object recognition in cortex*, in *Nature Neuroscience*, 1999, pp. 1019-1025.

[2] S. Dehaene, et al., *The neural code for written words: a proposal*, in *Trends in Cognitive Sciences*, 2005, pp. 335-341.

[3] J. Grainger, *Cracking the orthographic code: An introduction*, in *Language and Cognitive Processes*, 2008, pp. 1-35.

[4] J. Grainger, and C. Whitney, *Does the huamn mind raed wrods as a wlohe?*, in *Trends in Cognitive Sciences*, 2004, pp. 58-59.

[5] J. Grainger, and W.J.B. van Heuven, *Modeling letter position coding in printed word perception*, in *The Mental lexicon*, P. Bonin ed., Nova Science Publishers, New York, 2003, pp. 1-23.

[6] C. Whitney, *How the brain encodes the order of letters in a printed word: The SERIOL model and selective literature review*, in *Psychonomic Bulletin & Review*, 2001, pp. 221-243.

[7] J. Grainger, et al., *Letter position information and printed word perception: The relative-position priming constraint*, in *Journal of Experimental Psychology: Human Perception and Performance*, 2006, pp. 865-884.

[8] S. Dufau, et al., *An ERP investigation of location invariance in masked repetition priming*, in *Cognitive Affective and Behavioral Neuroscience*, 2008, pp. 222-228.

[9] I. Tydgat, and J. Grainger, *Serial position effects in the identification of letters, digits, and symbols*, in *Journal of Experimental Psychology: Human Perception and Performance*, 2009, in press.

[10] R. Shillcock, and P. Monaghan, *The computational exploration of visual word recognition in a split model*, in *Neural Computation*, 2001, pp. 1171-1198.

[11] P. Gomez, et al., *The overlap model: A model of letter position coding*, in *Psychological Review*, 2008, pp. 577-601.
[12] J.L. McClelland, and D.E. Rumelhart, *Explorations in parallel distributed processing*, MIT Press, Boston, MA, 1988.
[13] R.Q. Quiroga, et al., *Sparse but not 'Grandmother-cell' coding in the medial temporal lobe*, in *Trends in Cognitive Science*, 2008, pp. 87-91.